Command-control for Real-time Systems

Command-control for Real-time Systems

Edited by
Mohamed Chadli
Hervé Coppier

ISTE WILEY

First published 2013 in Great Britain and the United States by ISTE Ltd and John Wiley & Sons, Inc.

ISTE Ltd
27-37 St George's Road
London SW19 4EU
UK

www.iste.co.uk

John Wiley & Sons, Inc.
111 River Street
Hoboken, NJ 07030
USA

www.wiley.com

Library of Congress Control Number: 2013934628

British Library Cataloguing-in-Publication Data
A CIP record for this book is available from the British Library
ISBN: 978-1-84821-365-4

Printed and bound in Great Britain by CPI Group (UK) Ltd., Croydon, Surrey CR0 4YY

Table of Contents

Chapter 1

Introduction

The topic of this book is automation engineering applied to real systems. We use the term “real systems” to denote any complex system which forms an integral part of an industrial system, experimental system or onboard system in a vehicle or industrial machine. The peculiarity of these systems is that they are guided by real-time targets in a distributed environment.

Current research in the field of automation engineering relates mainly to systems of finite or of large dimensions, time-delayed systems, discrete event systems, hybrid dynamical systems, incomplete linear systems, etc., the modeling of such systems, identification of them, analysis of their stability, controlling them by coming up with different control laws such as:

– sliding mode control;

– predictive control;

– robust control;

– fuzzy control;

– etc.

The applications for such systems are many, and include applications in all sectors:

– electrical machines;

– environmental systems;

– vehicle dynamics;

– robotics;

– life sciences;

– process engineering;

– communications networks;

– aircraft;

– aeronautics and aerospace;

– etc.

The tools available in modern automation engineering serve many purposes, such as identification, parametric estimation, creation of correctors and observers, fault diagnosis, surveillance, etc.

The aim of the research reported herein relates to the computing of correctors for industrial systems of different physical natures, their implementation on real-time industrial targets (API/SCADA systems, embedded systems with distributed networks, Networked Control Systems (NCSs)) and their validation by means of simulation. When creating correctors, we use identification techniques or knowledge modeling. The primary approach in these various research projects is the optimization of industrial systems at the level of their control by making use as fully as possible of the resources available to us in industrial computing, communications networks and minimizing the realization time. In terms of control, 90% of regulation loops have a simple PID (Proportional/Integral/Derivative) control which, in addition, is often not optimized. Certain tools are lacking, as yet, for which we need to write control laws.

The considerable majority of procedures do not have knowledge models, so there is a clear advantage to developing efficient tools to identify knowledge on the basis of ground measurements.

The works presented in this book all stem from research carried out in an industrial context, and published in doctoral theses and masters dissertations:

– in the context of the regional project DIVA (research hub in Picardie, 2002–2005), the topics were: i) kinematic modeling of a hydraulic mechanical polyarticulated system; and ii) the building of a test array around a distributed computer structure for the excavator-loader created as part of the regional project "Aide à la conduite et détection de situations critiques pour engins intelligents de chantier" (driving support and critical situation detection for smart building machines);

– in the SEDVAC project (financed by the region of Picardie and FERDER, 2008–2012, UPJV-UTC collaboration; project leader: M. Chadli), the topic was the development of systems to support the driving of an automobile. The objective was to develop risk indicators based on the vehicle's dynamics and observer-based estimation techniques (estimation of road curvature, of slope, etc.);

– thermodynamic modeling of a cryogenic exchanger for the NA48 calorimeter at CERN was performed in the context of a partnership between the UPJV/ESIEE of Amiens and CERN to overhaul the control/command system of the NA48 experiment. Computation of the TDC (Time Delay Control) corrector for the overhaul of the control/command system of the NA48 experiment was done (thesis of Eng. M. Pezzetti, 2010). The collaboration with CERN also involved the description of the UNICOS framework object, the implementation of the object Multi-controller, the creation of digital models by identification for the gas mixing systems for the four LHC cryogenics experiments at CERN (the Gas Control System (GCS) project);

– a partnership between ESIEE-Amiens and Schneider Electric involved the computation of the internal model corrector (IMC) to regulate the output temperature of the superheater at an Alstom coal-burning power plant in Algeria;

– the works presented herein about modeling and multi-model control (also known as Takagi-Sugeno Fuzzy Models) are the result of many research projects carried out in the context of projects and theses supported in the past few years. These works relate to Takagi-Sugeno fuzzy control systems, fault-tolerant control systems, fault diagnostic systems and their applications in the automobile domain.

In more general terms, these works aimed to optimize industrial processes by using tools from automation engineering, industrial computing and communications networks. Indeed, in order to improve their product, industrialists have a never-ending need to optimize the regulating parameters of their procedures. Beyond the study of which control laws to use depending on the process to be modeled, it is also a question of providing generic tools which will work on any industrial computing platform (API/SCADA system) to guide the procedure(s), whilst integrating these tools as closely as possible into a clearly-defined development framework. In the particular case of an autonomous machine (area network or building machine), the computer structure is a system such as an embedded PC or microprocessor with a control area network that transmits distributed measurements to the mobile unit. The question then arises of the reliability and rapidity of area network control loops.

In order to study a real system, the following stages are necessary:

– understanding the specifications of the study that is to be carried out (description of a system's operation, constraints, operation point, the problem at hand and the objectives to be achieved);

– modeling the system with an appropriate type of modeling (relevance of the model, accessibility of the physical parameters, implementation of identification techniques and definition of the controllable variables);

– elaborating a strategy for controlling the system;

– calculating and simulating the type of control chosen in relation to the system being modeled;

– choosing the real-time target (industrial computing structure, use of networks) and its environment (real-time system, programming languages, the framework used);

– putting in place the real-time computing platform;

– implementing the control algorithms;

– simulating the whole of the controlled system on the real platform;

– testing and validating the whole system on a test array;

– installing and initializing the real industrial system.

All of this research contributes to the diffusion of modern automation techniques in industrial processes where, due to a lack of tools which make the connection between modeling, identification and implementation on real-time targets, optimization is as yet incomplete. Our work is intended precisely to fill that void, successively integrating new control laws so that the users can fully exploit the power of an engineering science such as automation engineering, to optimize the processes whilst retaining a high degree of maintainability of their installations. Furthermore, in terms of perspectives, on a topic which is of growing importance, such as energy efficiency in the field of sustainable development and construction, this research should be directly applicable, as demonstrated by numerous recent articles.

This manuscript is divided into seven chapters. Following this introductory chapter, the remaining chapters are as follows:

– Chapter 2 – Modeling tools: the aim of this chapter will be to present various techniques for mono- and multi-variable identification (ARX, ARMAX, etc.) for linear systems. The case of nonlinear systems is examined through the lens of the multi-model approach (also known as the multiple model approach or Takagi-Sugeno fuzzy approach). This is an interpolation of different linear models to approximate nonlinear behavior.

– Chapter 3 – Control tools: in this chapter, we examine different linear controllers (TDC, PFC, IMC, etc.). For nonlinear systems, techniques drawn from the domain of “soft computing” are put forward. Indeed, control laws for

Takagi-Sugeno fuzzy systems (also known as multi-model laws) are studied. The advantage to this approach is that it means the numerical tool LMI (Linear Matrix Inequalities) can be used to compute controllers.

– Chapter 4 – Application to cryogenic systems: the objective is thermodynamic modeling and control of a cryogenic exchanger for the NA48 calorimeter at CERN, and the modeling and control of the cryogenics of the ATLAS experiment being run at CERN.

– Chapter 5 – Application to thermal and gas systems: similarly, we shall observe the advanced control of the vapor temperature on output from a superheater at a coal-burning power station, and of gas systems.

– Chapter 6 – Application to vehicles: the aim in this chapter is to present two main domains: that of automobiles and that of excavator-loaders. Multi-controller-based techniques are applied to the dynamics of automobiles with a view to improving stability and safety. Driver assistance systems for an excavator-loader in a critical situation, kinematic modeling of the excavator-loader and control of the articulated arm are also subjects touched upon in this chapter.

– Chapter 7 – Real-time implementation (UNICOS, onboard systems): examples of distributed real-time architectures on a PLC-SCADA structure are examined. We also present the example of a universal controller: multi-controller, study of the reliability and rapidity of a CAN (Control Area Network)-based distributed architecture for an excavator-loader test array.

Chapter 2

Modeling Tools

2.1. Introduction

The literature about system modeling and identification goes back as far as does the literature about control. The first major papers to appear in the 1930s–1940s by Nyquist and Bode about frequency responses demonstrate this early interest. Ziegler and Nyquist's identifying work on the study of indicial responses dates from the 1940s. In addition, the progress made in terms of adaptive identification in the 1960s greatly contributed to the development of research in this domain. The research effort became organized, and in 1967, IFAC launched the first symposium on Identification and System Parameter Estimation. This and the series of symposiums which followed would produce a considerable number of articles about the aspects and problems surrounding system identification. Today, many books and articles dealing with modeling and identification are available, which give practical indications (for instance, see [BOR 01; EYK 74; LAN 02]).

Before speaking of models and identification, we shall quite deliberately discuss systems. L. Ljung [LJU 87] explains that if we wish to explicitly define the term "system", we could define it as being an object from which different interactions produce observable reactions. He adds that the determination of models by observation and study of the properties peculiar to a system is the very essence of science itself. It is indeed noteworthy that the goal of most scientific research projects since time began was merely to find representative models sufficiently accurate to describe natural phenomena. The view of a model as being unerringly true is therefore false in view of an (arbitrary) approach which defines a model for

---

Chapter written by Sébastien CABARET and Mohammed CHADLI.

any system. These philosophical considerations highlight the relative principle of a model and its relevance as regards a real system. All the approaches discussed in this chapter will take account of these hypotheses.

At the level of a control/command framework, the determination of a model is developed with a view to creating the control system [FLA 94]. In practice, a model is constructed on the basis of knowledge and observation of the data of the system subjected to stimuli (inputs) and its reactions (outputs). Experience is also a crucial factor in this process. The model, in industrial automation engineering, is intended to describe a system's behavior in order to assist the design and practical implementation of a control mechanism [BOR 01]. For this purpose, identification aims to determine the characteristics of a model, which essentially means producing a mathematical description of a system's dynamic and stationary behavior (if possible). Identification can therefore be summarized as the study and mathematical design of a model on the basis of observation, knowledge and the experience gained about the system.

System identification can be performed by way of six different treatments [GRA 76]:

– the distinction between a linear and a nonlinear system: all systems are naturally strictly nonlinear. Thus, a system is linear if for an area of its operation, its behavior is considered to be quasi-linear. In a linear regime, the principles of superposition are applicable, which makes system identification relatively simple;

– the difference between a stationary and a non-stationary system: non-stationarity must not be confused with nonlinearity. A stationary system is a system whose parameters do not vary under the same physio-chemical conditions within a given range of operation. This property is of fundamental importance in modeling and therefore in identification. Put more simply, we consider a system to be stationary if over the time taken to gather the data needed for the purpose of identification, the state of that system does not change;

– continuous or discrete systems: this aspect of the problem is rarely paid enough attention. Although it is not difficult to switch from a continuous to a discrete formulation, the implications for data processing and for the design of the control mechanism are great. The treatment regarding stability and robustness of the corrected systems is different depending on whether or not we are moving into a strictly continuous environment (programmable industrial robots can work in discrete mode);

– mono- or multi-variable treatment: the theoretical treatment of a mono- or multi-variable problem may be similar under certain conditions. The problem with multi-variable systems (or MIMO, for multiple input, multiple output) lies in the techniques for performing identification in concrete terms (complex algorithms and

programs). However, mono-variable identification techniques are significantly simpler;

– the deterministic or stochastic nature of a process: in practice, the measurements are contaminated by noise or by inapt smoothing/filtering techniques. We speak of deterministic identification when we can assume that the filters, noise and other disturbances are compensated for or taken into account. Reality frequently shows us that this assumption must be taken with a pinch of salt;

– the degree of prior knowledge of the system: this final point is probably the one which is most important not to overlook. Knowledge and experience are the best ways to success. They can help guide your choices of modeling techniques and identification strategies.

In the following section, we shall only deal with deterministic and stationary systems.

2.2. Models

In this chapter, we refer to various models which can be used to help design and develop the control system. The models which are helpful to do this are always those which represent the dynamics and the established regime. By contrast, static models can offer advantageous elements such as searching for the zone of operation.

There are a great many types of dynamic models, each of which is intended for specific applications. Nevertheless, we can divide such models into two broad categories:

– knowledge models, based on the laws of physics and chemistry;

– behavioral models, based on input/output data.

2.2.1. *Knowledge models*

Knowledge models are formulated on the basis of the laws of physics and chemistry. Knowledge models offer a reasonably complete description of systems. They are highly useful for the simulation of processes.

In order to convert a knowledge model into equation form, we need to define the physical phenomena which we wish to monitor [FLA 94]. In concrete terms, this means choosing the relevant variables to represent the process to be controlled. All the variables likely to be encountered and/or to interfere with the system must be defined and built into the knowledge model.

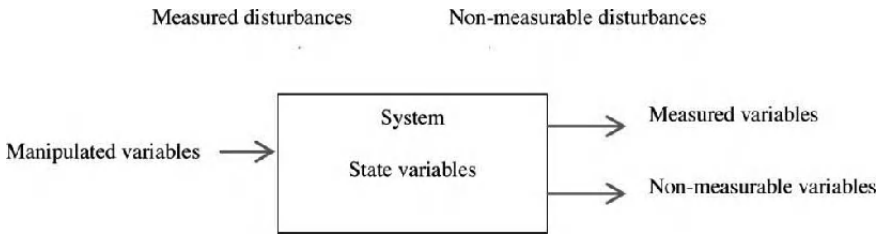


Figure 2.1. *The variables involved in the knowledge model*

Then comes the precise analysis of each variable with its particular function: state variables, input variables (manipulated) or output variables (measurable or otherwise), disturbance variables (measurable or otherwise).

The principle of the knowledge model is to find relations which link all the variables together with a view to establishing a set of mathematical equations.

Representation of the state of systems is a potent tool which can be used to simulate the operation of many systems – be they linear or otherwise, mono- or multi-variable, in continuous or discrete operation – and which also has the advantage of conserving the representation of the phenomena over time [GRA 03]. The advantage of this representation is that it is well developed for resolution and the study of knowledge-model automation problems.

EXAMPLE. – Consider the example in Figure 2.2 with (x_1, x_2 and x_3) the "internal" signals of the system. We can naturally conceive of controlling these different signals by way of the input signal e. The problem of control of the system can thus be dealt with by simultaneously controlling the change in the values of (x_1, x_2 and x_3).

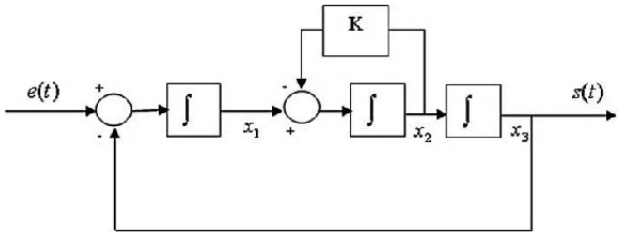


Figure 2.2. *Example: system and state representation*

We say that together, these three signals make up the state of the system. The internal variables are called the variables of state of a system. The state of a system is represented by the state vector $\underline{x}(t) = \underline{x} = [x_1\ x_2\ x_3]$.

For a system with n variables of state, m the number of inputs and p the number of outputs, the representation of the state is defined such that:

$$\begin{cases} \dot{\underline{x}}(t) = [A(t)].\underline{x}(t) + [B(t)].\underline{e}(t) \\ \underline{s}(t) = [C(t)].\underline{x}(t) + [D(t)].\underline{e}(t) \end{cases} \quad [2.1]$$

$$\begin{cases} A \in \mathbb{R}^{n \times n} & \begin{cases} C \in \mathbb{R}^{p \times n} \\ D \in \mathbb{R}^{m \times p} \end{cases} \\ B \in \mathbb{R}^{n \times m} \end{cases} \quad [2.2]$$

The first equation is the control equation; the second is the observation equation. When a system is modeled in the form of a representation of state, we show that it is possible to express the state of that system at a given time as a function of the input signal at that precise moment and of its past (i.e. its previous state).

For the above example, the representation of state becomes:

$$\begin{cases} \begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & -1 \\ 1 & -K & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} e(t) \\ s(t) = [0 \quad 0 \quad 1] \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \end{cases} \quad [2.3]$$

2.2.2. *Behavioral models*

Knowledge models are very often complex and unfortunately difficult to implement. The behavioral model or experimental model is an alternative which often proves better suited. The determination of this type of model is obtained by identification. These models, in themselves, have no physical meaning, but instead are intended merely to describe dynamic behavior by way of a mathematical formulation. We can distinguish two groups of behavioral models:

– *non-parametric* representative models: these models are mainly represented by curves which cannot be described by a finite set of numbers (e.g. frequential response, indicial response, etc.);

– *parametric* representative models: these models are characterized by a finite set of numbers (e.g. the polynomial coefficients for representation by transfer function, etc.).

Parametric models can be studied in different ways depending on their representations.

2.2.2.1. *Representation by difference equation: ARMA process*

For a system which does not exhibit pure delay, this representation is sometimes called a recurrence relation, linking the output *y(t)* to the input *u(t)*. The ARMA process (standing for AutoRegressive Moving Average) is defined by:

$$y(t) = \sum_{j=1}^{n_b+1} \beta_j u(t-j) - \sum_{j=1}^{n_a} \alpha_j y(t-j)$$

$$\Leftrightarrow y(t) = \sum_{j=1}^{n_b+1} b_{j-1} u(t-j) - \sum_{j=1}^{n_a} a_j y(t-j) \Big|_{\alpha_{j=} a_j}^{\beta_j = b_{j-1}} \qquad [2.4]$$

If we reason in the discretized domain, the output sample is known on the basis of the previous input and output samples:

$$y_k = \sum_{j=1}^{n_b+1} b_{j-1} u_{k-j} - \sum_{j=1}^{n_a} a_j \, y_{k-j}\text{f} \qquad [2.5]$$

This type of representation is particularly appropriate for parametric identification.

2.2.2.2. *Representation by transfer function*

The transfer function (continuous and discrete) is an essential tool in automation engineering today. Using transfer functions, we can study slave systems (performances) and design control mechanisms (disturbance rejection, precision, stability, robustness, etc.).

The continuous transfer function is defined for any linear invariant systems known as SISO (*Single Input Single Output*) describing a differential equation in accordance with the Laplace transform (initial conditions zero):

$$a_0 y(t) + a_1 \frac{dy(t)}{dt} + \cdots + a_{n_a} \frac{d^{n_a} y(t)}{dt^{n_a}} = b_0 u(t) + b_1 \frac{du(t)}{dt} + \cdots + b_{n_b} \frac{d^{n_b} u(t)}{dt^{n_b}} \qquad |n_a > n_b$$

[2.6]

$$\xrightarrow{Laplace} \qquad G(s) = \frac{b_0 + b_1 s + \cdots + b_{n_b} s^{n_b}}{a_0 + a_1 s + \cdots + a_{n_a} s^{n_a}} = \frac{Y(p)}{U(p)} \qquad [2.7]$$

The interesting use of the Laplace transfer function representation lies in the study and systematic solving of physical systems governed by a differential equation (linear SISO system).

The transfer function for discrete systems is obtained by the z transform of the continuous transfer function. The switch from continuous to discrete mode is described by the following equation:

$$H(z) = (1 - z^{-1})Z\left\{L^{-1}\left(\frac{G(s)}{s}\right)\right\} = \frac{Y(z)}{U(z)} \quad [2.8]$$

This transfer function can also be found by transforming the representation by the difference equation (equation [2.4]):

$$\overset{TZ}{\Rightarrow} \sum_{j=0}^{n_b} b_j U(z) z^{-j-1} - \sum_{j=1}^{n_a} a_j Y(z) z^{-j}$$

$$\overset{a_0=1}{\Longleftrightarrow} Y(z)\left(1 + \sum_{j=0}^{n_a} a_j z^{-j}\right) = \sum_{j=0}^{n_b} b_j U(z) z^{-j-1} \quad [2.9]$$

$$\frac{Y(z)}{U(z)} = z^{-1}\frac{\sum_{j=0}^{n_b} b_j z^{-j}}{\sum_{j=0}^{n_a} a_j z^{-j}} = z^{-1}\frac{B(z^{-1})}{A(z^{-1})}$$

NOTE.– If we use the notation q^{-1} for the delay operator, we obtain the polynomial representation of the discrete transfer function:

$$A(q^{-1})y(t) = B(q^{-1})u(t-1) \quad [2.10]$$

2.2.2.3. *Other interesting representations*

2.2.2.3.1. Use of discrete convolution

The advantage of this representation lies in the practical approach to obtaining a transfer function on the basis of an impulse response.

Let $\{f_i\}$ represent the coefficients of the impulse response. Assuming that the system is causal, using discrete convolution we can write:

$$y(t) = \sum_{i=-\infty}^{+\infty} f_i u(t-i) \overset{causal\ system}{\Longrightarrow} y(t) = \sum_{i=0}^{+\infty} f_i u(t-i) \quad [2.11]$$

Using the z transform, we obtain the transfer function:

$$y(t) = \sum_{i=-\infty}^{+\infty} f_i u(t-i) \quad \overset{TZ}{\Rightarrow} \quad Y(z) = \sum_{i=-\infty}^{+\infty}\left(\sum_{i=-\infty}^{+\infty} f_i u(t-i)\right) z^{-i}$$

$$\Leftrightarrow Y(z) = \left(\sum_{i=-\infty}^{+\infty} f_i z^{-i}\right)\left(\sum_{u=-\infty}^{+\infty} u(u) z^{-u}\right)\Big|_{t-i=u} = F(z)U(z). \quad [2.12]$$

This gives us:

$$F(z) = \sum_{i=-\infty}^{+\infty} f_i z^{-i} \xrightarrow{causal\ system} F(z) = \sum_{i=0}^{+\infty} f_i z^{-i} \quad [2.13]$$

In order to change to an incremental model, we need only introduce the operator $\Delta = 1 - q^{-1}$, and thus:

$$\Delta y(t) = \sum_{i=0}^{+\infty} f_i \Delta u(t-i) \quad [2.14]$$

2.2.2.3.2. Representation with disturbances

When disturbances are present, it is possible to give a global polynomial representation of the system:

$$A(q^{-1})y(t) = B(q^{-1})u(t-1) + v(t) \quad [2.15]$$

with $J(q^{-1})v(t) = e(t)$, where $e(t)$is a stochastic or deterministic disturbance. Furthermore, we often choose $J(q^{-1})=\Delta=1-q^{-1}$, which means we can introduce an integral action. Thus we obtain the CARIMA (*Controlled AutoRegressive Integrated Moving Average*) or ARIMAX (*AutoRegressive Integrated Moving Average with eXternal inputs*) model:

$$A(q^{-1})\Delta y(t) = B(q^{-1})\Delta u(t-1) + \Delta v(t) = B(q^{-1})\Delta u(t-1) + e(t) \quad [2.16]$$

2.3. The classic parametric identification methods

Identification of the behavioral model is intended to determine the model's parameters. On the basis of experimental measurements, the aim is to obtain a model whose behavior mimics that of the identified procedure as closely as possible.

2.3.1. *Graphic methods*

Graphic methods are very useful for the quick determination of a model. Most of the techniques in existence are based on the response of the procedure when given a gradated input. Therefore, these methods are often simple to implement on installations. They enable us to obtain answers about the enslavement behavior and thereby gain a clearer picture of the problem needing to be solved. Although they are not very precise, graphic methods of identification are often sufficient for regulating simple controllers such as PID loops.

There are four graphic identification techniques which offer interesting results:

– so-called intuitive graphic study, for both stable and unstable systems;

– the Broïda method, for first-order stable systems;

– the Strejc method, for *n*-order stable systems;

– the Ziegler–Nichols method for oscillating systems.

NOTE.– Before going on to use evolved methods of identification, graphic techniques are highly useful during the preliminary phases.

2.3.2. *Algorithmic methods*

Linear regression is a powerful tool for solving problems of parametric estimation. This technique enables us to describe the link that exists between controlled variables and response variables in order to then predict the responses on the basis of the controlled variables. The linear regression should give us the equation for the straight line which corresponds most closely to the points. The equation takes the following form:

$$k(t) = \Psi(t)\Lambda + \zeta(t) \qquad [2.17]$$

where $k(t)$ is the measurement vector, $\psi(t)$ the observation matrix, Λ the parameter vector and $\zeta(t)$ the error vector.

Methods which use this principle can thus be used to obtain the parameters of a linear dynamic model.

Parametric identification of dynamic systems involves finding the parameters whilst minimizing the errors between the models and the processes.

These methods for parametric identification require experimental measurements with excitation signals of varying amplitudes.

The key point with the implementation of the algorithms is the elaboration of a parameter adaptation algorithm (PAA) which has a recursive structure [LAN 02]. AAP is an approach based on linear regression which can be used to calculate the new parameter vector on the basis of the old estimation plus a correction term. Its structure is such that:

$$[A] = [B] + [C] \times [D] \times [E]. \qquad [2.18]$$

where:

– A: new estimation of the parameters (vector);

– B: previous estimation of the parameters (vector);

– C: adaptation gain (matrix);

– D: function of the measurements (vector) or "observation vector";

– E: function of the prediction error (scalar).

Consider the discrete model such that:

$$\boldsymbol{y}(\boldsymbol{t}+\boldsymbol{1}) = \boldsymbol{\theta}^{\boldsymbol{T}}\boldsymbol{\phi}(\boldsymbol{t}) \quad [2.19]$$

where θ is the vector of the unknown parameters and $\phi(t)$ is the vector of the measurements.

Let the *a priori* predictor be the adjustable prediction model constructed on the same structure as the previous model:

$$\hat{\boldsymbol{y}}^{\boldsymbol{0}}(\boldsymbol{t}+\boldsymbol{1}) = \hat{\boldsymbol{y}}\left(\boldsymbol{t}+\boldsymbol{1}\middle|\hat{\boldsymbol{\theta}}(\boldsymbol{t})\right) = \hat{\boldsymbol{\theta}}(\boldsymbol{t})^{\boldsymbol{T}}\boldsymbol{\phi}(\boldsymbol{t}) \quad [2.20]$$

where $\hat{\boldsymbol{\theta}}(\boldsymbol{t})$ is the vector of the parameters estimated at time t. The *a priori* prediction error is then defined by the relation:

$$\boldsymbol{\varepsilon}^{\boldsymbol{0}}(\boldsymbol{t}+\boldsymbol{1}) = \boldsymbol{y}(\boldsymbol{t}+\boldsymbol{1}) - \hat{\boldsymbol{y}}^{\boldsymbol{0}}(\boldsymbol{t}+\boldsymbol{1}) = \boldsymbol{\varepsilon}^{\boldsymbol{0}}\left(\boldsymbol{t}+\boldsymbol{1}, \hat{\boldsymbol{\theta}}(\boldsymbol{t})\right) \quad [2.21]$$

Finally, the objective is to minimize the *a priori* prediction error using the AAP. The structure of the AAP will therefore depend on the parameters estimated at time t, the observation vector and the *a priori* prediction error. The AAP will often be in the form:

$$\hat{\boldsymbol{\theta}}(\boldsymbol{t}+\boldsymbol{1}) = \hat{\boldsymbol{\theta}}(\boldsymbol{t}) + \boldsymbol{\Delta}\hat{\boldsymbol{\theta}}(\boldsymbol{t}+\boldsymbol{1}) = \hat{\boldsymbol{\theta}}(\boldsymbol{t}) + \boldsymbol{f}\left(\hat{\boldsymbol{\theta}}(\boldsymbol{t}), \boldsymbol{\phi}(\boldsymbol{t}), \boldsymbol{\varepsilon}^{\boldsymbol{0}}(\boldsymbol{t}+\boldsymbol{1})\right) \quad [2.22]$$

2.3.2.1. *Conventional methods*

2.3.2.1.1. Gradient algorithm

This algorithm is used in the identification of an ARMA-type model of the form:

$$\boldsymbol{A}(\boldsymbol{q}^{-\boldsymbol{1}})\boldsymbol{y}(\boldsymbol{t}) = \boldsymbol{q}^{-\boldsymbol{d}}\boldsymbol{B}(\boldsymbol{q}^{-\boldsymbol{1}})\boldsymbol{u}(\boldsymbol{t}) \quad [2.23]$$

The prediction error is defined such that:

$$\boldsymbol{\varepsilon(t\ +\ 1) = y(t\ +\ 1) - \hat{y}(t\ +\ 1)} \quad [2.24]$$

and the criterion of gradient minimization such that:

$$\min_{\hat{\theta}(t+1)} J(t\ +\ 1) = \left[\varepsilon(t\ +\ 1)\right]^2 \quad [2.25]$$

The APP is then found in the form:

$$\hat{\theta}(t+1) = \hat{\theta}(t) + \frac{F\phi(t)\varepsilon^0(t+1)}{1+\phi(t)^T F\phi(t)} \quad [2.26]$$

where $\boldsymbol{\hat{\theta}(t\ +\ 1)}$ is the new vector of the parameters estimated at time (t+ 1), $\phi(t)$ is the measurement vector and $F = \alpha I$, the matrix adaptation gain with weight α. The weight must satisfy the following equation [LAN 02]:

$$\alpha < \frac{2}{\phi(t)^T \phi(t)} \quad [2.27]$$

2.3.2.1.2. Non-recursive least-squares (LS) algorithm

This algorithm is used in the identification of an ARMA-type model. We minimize the "least squares" criterion such that:

$$\min_{\hat{\theta}(t)} J(t) = \frac{1}{t}\sum_{i=1}^{t}[y(i) - \hat{\theta}(t)^T \phi(i-1)]^2 = \frac{1}{t}\sum_{i=1}^{t} \varepsilon^2(i, \hat{\theta}(t)) \quad [2.28]$$

We obtain the parameter vector in the following non-recursive form:

$$\hat{\theta}(t) = \left[\sum_{i=1}^{t} \phi(i-1)\phi(i-1)^T\right]^{-1} . \sum_{i=1}^{t} y(i).\phi(i-1) = F(t)\sum_{i=1}^{t} y(i).\phi(i-1) \quad [2.29]$$

where:

$$F(t)^{-1} = \sum_{i=1}^{t} \phi(i-1)\phi(i-1)^T \quad [2.30]$$

2.3.2.1.3. Recursive least-squares (RLS) algorithm

In order for the LS algorithm to be recursive, we consider the estimation of $\boldsymbol{\hat{\theta}(t)}$ and $\boldsymbol{\hat{\theta}(t\ +\ 1)}$ such that:

$$\begin{cases} \hat{\theta}(t) = F(t)\sum_{i=1}^{t} y(i).\phi(i-1) \\ \hat{\theta}(t+1) = F(t+1)\sum_{i=1}^{t+1} y(i).\phi(i-1) \end{cases} \quad [2.31]$$

and we note that:

$$F(t+1)^{-1} = F(t)^{-1} + \phi(t)\phi(t)^T \quad [2.32]$$

We obtain the determination of the parameter vector in the recursive form by using the following APP:

$$\boldsymbol{\hat{\theta}(t+1) = \hat{\theta}(t) + F(t+1).\phi(t).\varepsilon^0(t+1)} \quad [2.33]$$

We then define the adaptation gain [LAN 02] with:

$$\begin{cases} F(t+1)^{-1} = \lambda_1(t)F(t)^{-1} + \lambda_2(t)\phi(t)\phi(t)^T \\ 0 < \lambda_1 \leq 1 \\ 0 < \lambda_2 \leq 2 \\ F(0) > 0 \end{cases} \quad [2.34]$$

and we find the adaptation gain:

$$F(t+1) = \frac{1}{\lambda_1(t)}\left[F(t) - \frac{F(t)\phi(t)\phi^T F(t)}{\frac{\lambda_1}{\lambda_2} + \phi^T F(t)\phi(t)}\right]. \quad [2.35]$$

2.3.2.1.4. Extended least-squares (ELS) algorithm

This algorithm is used in the identification of an ARMAX-type model in the form (model "process + disturbance"):

$$\boldsymbol{A(q^{-1})y(t) = q^{-d}B(q^{-1})u(t) + C(q^{-1})e(t)} \quad [2.36]$$

The aim of this identification is to model the process and the disturbance in order to obtain a prediction error of zero. The prediction error is such that:

$$\boldsymbol{\varepsilon(t+1) = e(t+1) = y(t+1) - \hat{y}(t+1)} \quad [2.37]$$

We minimize the criterion in the sense of the "least squares", now with the estimated parameter vector and the observation vector:

$$\begin{cases} \phi(t)^T = [-y(t) \ \ldots \ -y(t-n_A+1) \quad u(t-d) \ \ldots \ u(t-d+n_B+1) \quad \varepsilon(t) \ \ldots \ \varepsilon(t-n_C+1)] \\ \hat{\theta}(t)^T[\hat{a}_1(t) \ \ldots \ \hat{a}_{n_A}(t)\hat{b}_1(t) \ \ldots \ \hat{b}_{n_B}(t)\hat{c}_1(t) \ \ldots \ \hat{c}_{n_C}(t)] \end{cases}$$

[2.38]

and the convergence of the observation vector is subject to a sufficient but not necessary condition:

$$\begin{cases} \frac{1}{C(z^{-1})} - \frac{\lambda_2}{2} \\ 2 > \lambda_2 \geq max\ \lambda_2(t) \end{cases} \quad [2.39]$$

2.3.2.1.5. The recursive maximum likelihood (RML) algorithm

This algorithm is an improvement upon the ELS method. The idea is to filter the observation vector by $\mathbf{1}/\widehat{\boldsymbol{C}}(\boldsymbol{t}, \boldsymbol{q^{-1}})$, with $\widehat{\boldsymbol{C}}(\boldsymbol{t}, \boldsymbol{q^{-1}})$ being an estimation of $\boldsymbol{C}(\boldsymbol{t})$ at time *t*. The effect is to accelerate the decorrelation between the observation vector and the prediction error. The other effect is to remove the condition on $\boldsymbol{C}(\boldsymbol{z^{-1}})$ of the ELS method. The estimated parameter vector and the observation vector become:

$$\begin{cases} \phi(t)^T = \frac{1}{\hat{C}(t, q^{-1})} * \\ [-y(t) \quad \dots \quad -y(t-n_A+1) \quad u(t-d) \quad \dots \quad u(t-d+n_B+1) \quad \varepsilon(t) \quad \dots \quad \varepsilon(t-n_C+1)] \\ \hat{\theta}(t)^T \left[\hat{a}_1(t) \quad \dots \quad \hat{a}_{n_A}(t) \hat{b}_1(t) \quad \dots \quad \hat{b}_{n_B}(t) \hat{c}_1(t) \quad \dots \quad \hat{c}_{n_C}(t)\right] \end{cases}$$

[2.40]

2.3.3. *Validation and estimation of the model identified*

The elaboration of a model is, in reality, only one stage in the process of identification. Long before modeling the system, we need to:

– take an accurate reading of the input/output measurements. This stage is sometimes accompanied by additional treatments which lend themselves to a good estimation of the parameters (measurement conditioning);

– take account of the complexity of the model chosen for parameter identification (when graphic methods are limited);

– estimate the parameters of the model using a method of identification;

– validate the model found.

The quality of the measurements is a necessary (but not sufficient) condition for obtaining a good model. Also, signal conditioning enables us to circumvent additional constraints which hamper identification.

Initial conditions for the taking of measurements:

– the prediction errors or measurement errors must be negligible (or known!);

– the sampling must be fine in the interests of good resolution, but not too fine (coherent use of resources is to be desired);

– the system must be in its nominal operating conditions;

– it is sometimes necessary to add filtering functions (e.g. an anti-folding filter).

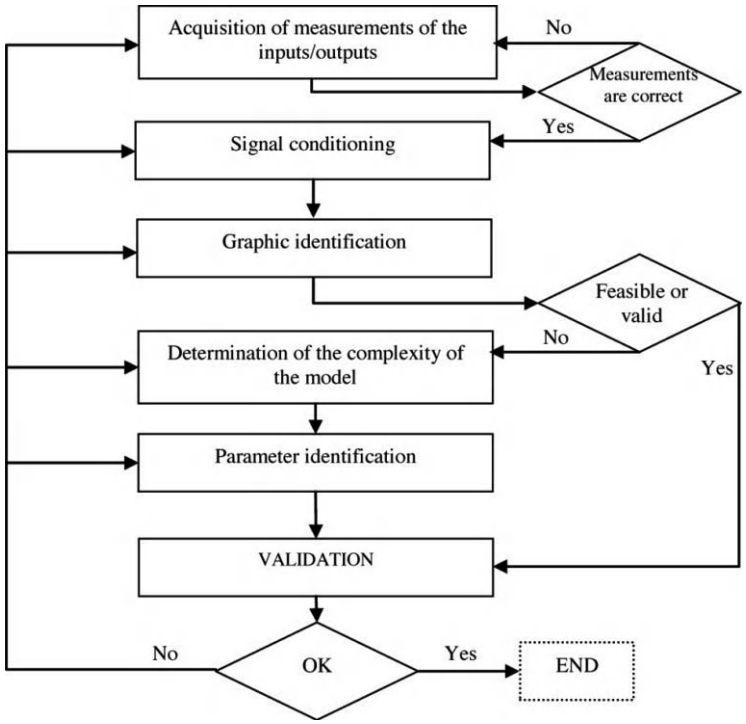


Figure 2.3. *Principle of the identification process and its validation*

Excitation of the system:

– we need to give the system an input which will cause it to generate enough information for the identification to be able to describe the dynamics of the system and the static gain. This input must be frequency-rich, and the standard theoretical solution is to apply a PRBS (pseudo-random binary sequence);

– in practice, it is not always easy to generate a PRBS. A succession of three stages (long enough for precise determination of the gain) is often sufficient.

Measurement conditioning:

– we need to get rid of the continuous component (if one is present) that results from the point of operation. The effect of this component may skew the parameter estimations for the dynamics of the system;

– aberrant values (other than noise) need to be removed (or adapted) so as not to corrupt the identification. These may be due to the measuring instruments or to a punctual problem with the tools used;

– it may be interesting to change the scales of the signals so as to obtain a good rate of convergence during the identification process;

– when the system has a pure derivator or a pure integrator, it is preferable to take this information into consideration (by way of an operation on the signals) in order to reduce the complexity of the model to be identified. This treatment can improve the precision of the identification.

2.3.3.1. *Determination of the complexity of the model*

The delay d can easily be determined with experiments. Based on the measurements, we can identify the reaction time when subjected to an excitation, and thus deduce that d = (reaction time/sampling).

The estimation of the order of the system (order of the polynomials A, B (or C?)) is done by way of operations on the measurements. In order to do this, we can use the instrumental determinant ratio test or the instrumental residuals test [BOR 01]. These tests involve exploiting pairs of input and output measurements. We construct the *information matrix* Q_m such that:

$$Q_m = \frac{1}{N}\sum_{k=1}^{N} \begin{bmatrix} u(k) \\ u(k+1) \\ u(k-1) \\ u(k+2) \\ \dots \\ u(k-m+1) \\ u(k+m) \end{bmatrix} [y(k+1) \quad u(k+1) \quad \cdots \quad y(k+m) \quad u(k+m)] \quad [2.41]$$

where m is the order and N the number of measurements. The test consists of calculating the *instrumental determinant ratio* (IDR) such that:

$$IDR(m) = \left| \frac{\det(Q_m)}{\det(Q_{m+1})} \right| \quad [2.42]$$

Then, as the order n of the process, we take the value m for which this ratio first shows a rapid increase.

The instrumental residuals (IR) test involves calculating the ratio:

$$IR(m) = \left|\frac{\det(Q_{m+1})}{\det(Q_m)}\right| \quad [2.43]$$

with Q'_{m+1} being the matrix obtained from matrix Q_{m+1} by removing the last row and the last column. The order n of the process is the value m when the IR ratio decreases sharply.

Then, in practical terms, we generally choose (if d is determinate):

$$\begin{cases} n_a = n \\ n_b = n_a \\ n_c = n_a \end{cases} \quad [2.44]$$

NOTE.– It is also possible to undertake the estimation of the complexity at the end of the identification process (without the tests) by performing successive approaches on the model and determining its relevance by validation.

2.3.3.2. *Validation of the model*

The model can be validated by a number of complementary approaches:

– the first is to subject the model identified to the input of measured values in order to view the output and to compare it to the real measured output (e.g. a graphic simulation). By this simple principle, we can tell at a glance whether the model is incorrect. Conversely, though, this simple stage is not enough to confirm the pertinence of the model;

– the second approach is to subject the model and the real system (in nominal conditions) to the same excitations in order to see whether the identification was sufficiently accurate to describe the complete dynamic behavior of the real system. Unfortunately, this stage is not always easy to perform;

– the third approach is to divide all of the measurements into two groups: the first is called the "identification set" and the second is usually called the "validation set". By applying the identification method to the first set and then comparing the model identified against the measurements of the validation set, it is easy to obtain results of performances (e.g. comparison of the standard deviation, etc.);

– the fourth approach is to verify the validity of the model by tests on the measurements. One such test is based on prediction error whitening (a method used for RLS, ELS, RML). We perform the calculation such that:

$$RN(i) = \frac{R(i)}{R(0)} = \left. \frac{\frac{1}{N}\sum_{t=1}^{N} \varepsilon(t)\varepsilon(t-1)}{\frac{1}{N}\sum_{t=1}^{N} \varepsilon^2(t)} \right|_{i=1,2,\ldots \max\,(n_{a,n_{b+d}})} \quad [2.45]$$

where N is the number of measurements.

In general, a model is acceptable from the point of view of the test if:

$$\left| RN(i) \leq \frac{2.17}{\sqrt{N}} \right| \quad [2.46]$$

2.4. Multi-model approach

2.4.1. *Introduction*

Since the works of Zadeh [ZAD 65], fuzzy logic has had a great deal of success for modeling of complex/nonlinear or poorly-known systems and also for the creation of fuzzy regulators [TAK 85]. The capacity of fuzzy logic to represent a broad class of systems has been demonstrated as a universal approximation. In that sense, many successes have been made at the level of the applications [BUC 93; CAS 95]. In the literature, we can pick out numerous fuzzy models. Yet we can still distinguish between two main types of fuzzy models: the Mamdani fuzzy model and the Takagi-Sugeno (T-S) model [TAK 85]. The Mamdani fuzzy model uses fuzzy subsets in the consequence part, whereas the T-S-type fuzzy model uses functions dependent on the input variables. The most popular T-S model is that which, in the consequence part, uses a linear model in state representation or in an autoregressive model. This type of representation, called multi-model representation [MUR 97], has come to enjoy a great deal of success in all domains of automation engineering (identification, control, FDI, FTC) [AKH 07a; AKH 07b; CHA 08b; CHA 09; CHA 10a; FRA 90; PAT 97].

2.4.2. *Techniques for obtaining multi-models*

Multi-models are obtained by interpolation between linear time-invariant (LTI) models. Each LTI model represents a domain of operation valid around a particular point of operation. We can speak of three methods which can be employed in order to obtain a multi-model:

– by identification when data are available about the inputs and outputs;

– by linearization around different points of operation;

– by a convex polytopic transformation when we have an analytical model.

These methods use the state representation. Thus, works relating to stability analysis of multi-models and the creation of control laws or observers adopt the state representation in order to extend to a nonlinear scenario the control techniques by state feedback, output feedback and the construction of nonlinear observers based on the conventional observers widely used in the linear domain.

The form generally adopted for multi-models in the continuous and discrete cases is, respectively:

$$\dot{x}(t) = \sum_{i=1}^{N} \mu_i(z(t))(A_i x(t) + B_i u(t)) \qquad [2.47]$$

$$x(t+1) = \sum_{i=1}^{N} \mu_i(z(t))(A_i x(t) + B_i u(t)) \qquad [2.48]$$

where $x(.) \in \mathbb{R}^n$ is the state variable vector, $u(.) \in \mathbb{R}^m$ the input vector, $z(.) \in \mathbb{R}^q$ is the decision variable vector. The $\mu_i(.)$ $i \in I_N$ are the activation functions such that $\sum_{i=1}^{N} \mu_i(z(t)) = 1,\ \mu_i(z(t)) \geq 0.$

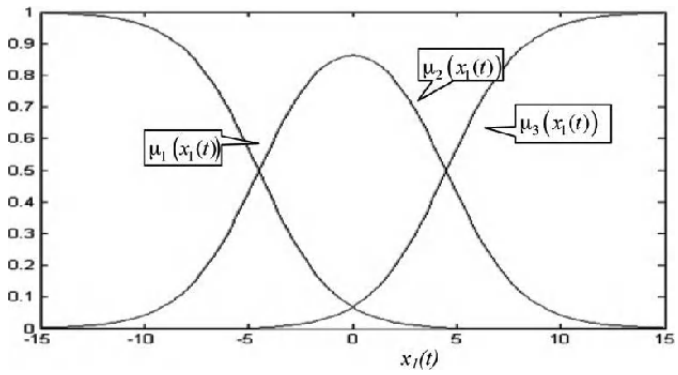


Figure 2.4. *Example of activation functions*

2.4.2.1. *Construction of multi-models by identification*

"Black box"-type models are identified on the basis of the data on the inputs and outputs around different points of operation. Independently of the type of model chosen, this identification requires a search for an "optimal" structure, estimation of

the parameters and validation of the final model [GAS 01; JOH 00; JOH 03; MUR 97; TAK 85]. In our case, the model is nonlinear in relation to the parameters. Iterative techniques of nonlinear optimization are used depending on the data available initially. The methods of identification of the unknown parameters are generally based on the minimization of a functional of the difference between the output estimated by the multi-model $y_m(t)$ and the measured output of the system $y(t)$. The criterion commonly used is quadratic error minimization:

$$J(\theta) = \frac{1}{2}\sum_{k=1}^{H} \varepsilon^2(k,\theta) = \frac{1}{2}\sum_{k=1}^{H} \left(y_m(k) - y(k)\right)^2 \qquad [2.49]$$

where H is the horizon of observation and θ is the parameter vector of the multi-model (LTI models and activation functions).

The algorithm for updating the parameter vector is described by the following general relation:

$$\theta(k+1) = \theta(k) - \eta D(k) \qquad [2.50]$$

where k is the iteration indicator, $\theta(k)$ the value of the solution at iteration k, η represents an adjustment factor, calculated by a heuristic method, which helps regulate the rate of convergence toward the solution. *D(k)* is the direction of searching, whose expression depends on the optimization method used. Here we recall the two most widely used optimization methods.

GRADIENT ALGORITHM.– This method is based on a development to the first order of the criterion $J(\theta)$:

$$D(k) = G\left(\theta(k)\right) = \left.\frac{\partial J}{\partial \theta}\right|_{\theta=\theta(k)} = \left.\sum_{k=1}^{H} \frac{\partial \varepsilon(k,\theta)}{\partial \theta} \varepsilon(k,\theta)\right|_{\theta=\theta(k)} \qquad [2.51]$$

NEWTON ALGORITHM.– This algorithm is based on a development to the second order:

$$D(k) = H^{-1}(k)G(\theta(k)) \qquad [2.52]$$

where $H(k)$ is the Hessian matrix of the criterion defined by:

$$H(k) = \sum_{k=1}^{H} \frac{\partial \varepsilon(k,\theta)}{\partial \theta} \frac{\partial \varepsilon(k,\theta)}{\partial \theta^T} + \sum_{k=1}^{H} \frac{\partial^2 \varepsilon(k,\theta)}{\partial \theta^2} \varepsilon(k,\theta) \Bigg|_{\theta=\theta(k)} \quad [2.53]$$

This algorithm has the advantage of simultaneously defining the direction and speed of searching. The main disadvantage lies in the calculation of the inversion of the Hessian matrix at each iteration.

GAUSS-NEWTON ALGORITHM.– This is a simplified form of the Newton algorithm. An approximate expression of the Hessian matrix is used, ignoring the second-order terms:

$$H_a = \sum_{k=1}^{H} \frac{\partial \varepsilon(k,\theta)}{\partial \theta} \frac{\partial \varepsilon(k,\theta)}{\partial \theta^T} \quad [2.54]$$

This algorithm guarantees a positive definite Hessian matrix, and consequently convergence toward a minimum.

These algorithms are sensitive to the initial choice of the parameter vector θ, and when the dimension of the parameter space is very large, there is a danger that the algorithms will converge toward local minima. For further information, the interested reader can refer to [BOR 90; BOR 92], for example.

EXAMPLE.– In order to illustrate the method, let us consider the following bicycle model for a four-wheel-drive vehicle [CHA 07b]:

$$\begin{pmatrix} \dot{\beta}(t) \\ \dot{r}(t) \end{pmatrix} = \begin{pmatrix} 2\dfrac{F_f(t) + F_r(t)}{mU} - r(t) \\ 2\dfrac{a_f F_f(t) - a_r F_r(t)}{I_z} \end{pmatrix} \quad [2.55]$$

where β denotes the sideslip angle, r is the differential of the yaw angle, F_f is the front pneumatic force and F_r the rear pneumatic force. U is the vehicle's speed, taken to be constant, I_z is the moment of inertia in relation to the z axis and m is the mass of the vehicle.

In the existing body of literature, there are a number of different models representing pneumatic forces. Of these models, the most widely used is that expressed as a function of the gradient angles by [CHA 07b]:

$$F_f = D_f \sin\left(C_f \tan^{-1}\left(\beta_f(1-E_f)\alpha_f + E_f \tan^{-1}(\beta_f \alpha_f)\right)\right) \quad [2.56]$$

$$F_r = D_r \sin\left(C_r \tan^{-1}\left(\beta_r(1-E_r)\alpha_r + E_r \tan^{-1}(\beta_r \alpha_r)\right)\right) \quad [2.57]$$

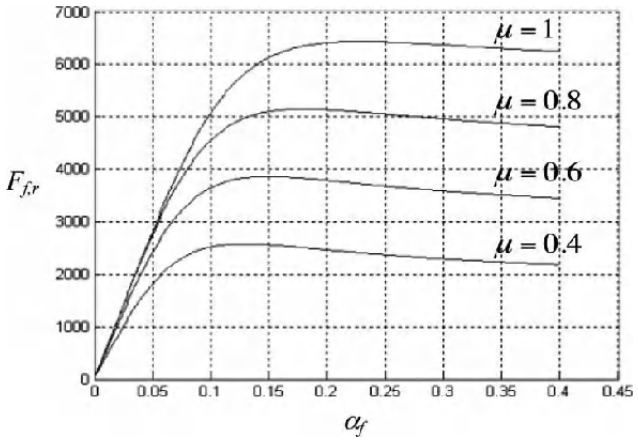


Figure 2.5. *Nonlinear behavior of contact forces*

Our objective is to approximate these forces of contact between the tire and the ground at the front and rear of the vehicle (F_f and F_r) (equations [2.56] and [2.57] using a multi-model dependent on the gradient angle $\alpha_i \left(i = f, r\right)$ and the friction coefficient μ by:

$$\begin{cases} F_f &= h_1(|\alpha_f|)C_{f1}(\mu)\alpha_f + h_2(|\alpha_f|)C_{f2}(\mu)\alpha_f \\ F_r &= h_1(|\alpha_f|)C_{r1}(\mu)\alpha_r + h_2(|\alpha_f|)C_{r2}(\mu)\alpha_r \end{cases} \quad [2.58]$$

where C_{fi} and C_{ri} represent the rigidity coefficients depending on the friction coefficient μ. The activation functions h_i (i = 1, 2) satisfy the following conditions:

$$\sum_{i=1}^{2} h_i(|\alpha_f|) = 1, \quad 0 \le h_i(|\alpha_f|) \le 1, \quad i = 1,2 \quad [2.59]$$

By choosing activation functions of the form:

$$h_1(|\alpha_f|) = \frac{\omega_1(|\alpha_f|)}{\omega_1(|\alpha_f|) + \omega_2(|\alpha_f|)} = 1 - h_2(|\alpha_f|) \quad [2.60]$$

where:

$$\omega_1(|\alpha_f|) = \frac{1}{\left(1+abs\left(\frac{|\alpha_f|-c_1}{a_1}\right)\right)^{2b_1}} \quad \omega_2(|\alpha_f|) = \frac{1}{\left(1+abs\left(\frac{|\alpha_f|-c_2}{a_2}\right)\right)^{2b_2}} \quad [2.61]$$

and using an algorithm such as those defined above, for μ = 0.7, we obtain the values given in Table 2.1.

Nominal rigidity coefficients	C_{f1}	C_{f2}	C_{r1}	C_{r2}
Values	60,712	4,812	60,088	3,455

Table 2.1. *Nominal rigidity coefficients*

and the following values for the parameters of the activation functions:

$$a_1 = 0.5077,\ b_1 = 0.4748,\ c_1 = 3.1893,\ a_2 = 5.3907\ b_2 = 0.4356,\ c_2 = 0.5633 \quad [2.62]$$

Figure 2.6 can be used to compare the two models: that described in equations [2.56] and [2.57] and the multi-model proposed here. This model will be used in the following chapters to create multi-regulators for the model of the vehicle [2.55] in a multi-model representation.

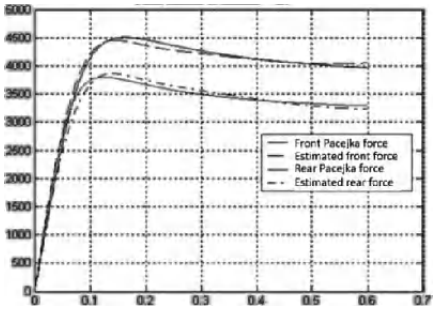


Figure 2.6. *Approximation of forces (equations [2.56] and [2.57]) by a multi-model*

2.4.2.2. *Construction of multi-models by linearization*

In this case, we assume we have a nonlinear mathematical model of the physical process, which we linearize around different carefully-chosen points of operation. Consider the following nonlinear system:

$$\dot{x}(t) = f\left(x(t), u(t)\right) \qquad [2.63]$$

where $f(x) \in C^1, f : \mathbb{R}^n \longrightarrow \mathbb{R}^n$. The linearization of the system [2.63] around a random point of operation $(x_i , u_i) \in \mathbb{R}^n \times \mathbb{R}^m$ is:

$$\dot{x}(t) = A_i(x(t) - x_i) + B_i(u(t) - u_i) + f(x_i, u_i) \qquad [2.64]$$

which can be rewritten in the form:

$$\dot{x}(t) = A_i x(t) + B_i u(t) + d_i \qquad [2.65]$$

with:

$$A_i = \left.\frac{\partial f(x,u)}{\partial x}\right|_{\substack{x=x_i \\ u=u_i}}, \; B_i = \left.\frac{\partial f(x,u)}{\partial u}\right|_{\substack{x=x_i \\ u=u_i}}, \; d_i = f(x_i, u_i) - A_i x_i - B_i u_i \qquad [2.66]$$

Assuming that the local models (also known as sub-models) result from a linearization around N points of operation (x_i, u_i), the multi-model formulation gives us:

$$\dot{x}(t) = \sum_{i=1}^{N} \mu_i\left(z(t)\right)\left(A_i x(t) + B_i u(t) + d_i\right) \qquad [2.67]$$

where the $\mu_i\left(z(t)\right)$, $i \in I_N$ are the activation functions and $z(t)$ is the decision variable vector, dependent upon the measurable variables of state and possibly on the control $u(t)$. Note that in this case, the number of local models (N) depends on the desired precision of modeling, the complexity of the nonlinear system and the choice of structure of the activation functions.

EXAMPLE.– Consider the following model for a vehicle which combines the longitudinal and lateral dynamics [AKH 07a]:

$$\begin{aligned}
\dot{u}(t) &= v(t)r(t) - fg + \frac{(fk_1 - k_2)}{M}u(t)^2 + c_f\frac{v(t) + ar(t)}{Mu(t)}\delta(t) + \frac{T(t)}{M} \\
\dot{v}(t) &= -u(t)r(t) - \frac{(c_f - c_r)}{Mu(t)}v(t)^2 + \frac{(bc_r - ac_f)}{Mu(t)}r(t) + \frac{(c_f\delta(t) + T\delta(t))}{M} \\
\dot{r}(t) &= \frac{(bc_r - dc_f)}{I_z u(t)}v(t) - \frac{(b^2 c_r - a^2 c_f)}{I_z u(t)}r(t) + \frac{(aT(t)\delta(t) + ac_f\delta(t))}{I_z}
\end{aligned} \qquad [2.68]$$

where *u(t), v(t)* and *r(t)* respectively represent the longitudinal speed, the lateral speed and the yaw velocity. $\delta(t)$ is the steering angle, *T(t)* is the traction/steering force. The other parameters of the vehicle are given in Table 2.2.

M	Mass of the vehicle (1480 kg)
I_z	Moment of inertia (2350 kg.m^2)
g	Force of gravity (9.81 m/s^2)
f	Rotational friction coefficient (0.02)
a	Distance from front axle to center of gravity (1.05 m)
b	Distance from rear axle to center of gravity (1.63 m)
c_f	Friction coefficient of front tires (135000 N/rad)
c_r	Friction coefficient of rear tires (95000 N/rad)

Table 2.2. *Parameters of the model*

The model of the vehicle is a nonlinear model in the form $\dot{x}(t) = f(x(t), w(t))$, where $x(t) = \left[u(t), v(t), r(t)\right]^T$, $w(t) = \left[\delta(t), T(t)\right]^T$. Our objective is to approximate this model with a T-S model in the form of equation [2.67]:

$$\dot{x}(t) = \sum_{i=1}^{N} \mu_i \left(z(t)\right)\left(A_i x(t) + B_i u(t) + d_i\right) \qquad [2.69]$$

where

$$\sum_{i=1}^{N} \mu_i \left(z(t)\right) = 1, \ \mu_i \left(z(t)\right) \geq 0 \qquad [2.70]$$

and:

$$A_i = \left.\frac{\partial f(x,w)}{\partial x}\right|_{\substack{x=x_i \\ w=w_i}}, \ B_i = \left.\frac{\partial f(x,w)}{\partial u}\right|_{\substack{x=x_i \\ w=w_i}}, \ d_i = f\left(x_i, w_i\right) - A_i x_i - B_i w_i \qquad [2.71]$$

$$A_i = \begin{bmatrix} \frac{2(fk_1 - k_2)u_i}{M} - \frac{c_f(v_i + ar_i)}{Mu_i^2}\delta_i & r_i + \frac{c_f\delta_i}{Mu_i} & v_i + \frac{ac_f\delta_i}{Mu_i} \\ -r_i + \frac{(c_f + c_r)}{Mu_i^2}v_i - \frac{(bc_f - ac_f)}{Mu_i^2}r_i & \frac{(c_f + c_r)}{Mu_i^2} & u_i + \frac{(bc_r - ac_f)}{Mu_i^2} \\ -\frac{(bc_r - ac_f)}{I_z u_i^2}v_i + \frac{(b^2c_r - a^2c_f)}{I_z u_i^2}r_i & \frac{(bc_r - ac_f)}{I_z u_i^2} & -\frac{(b^2c_r - a^2c_f)}{I_z u_i^2} \end{bmatrix} \quad [2.72]$$

$$B_i = \begin{bmatrix} c_f\frac{v_i + ar_i}{Mu_i} & \frac{1}{M} \\ \frac{c_f + T_i}{M} & \frac{\delta_i}{M} \\ \frac{aT + ac_f}{I_z} & \frac{a\delta_i}{I_z} \end{bmatrix} \quad [2.73]$$

$$D_i = \begin{bmatrix} v_i r_i - fg + \frac{fk_1 - k_2}{M}u_i^2 + c_f\frac{v_i + ar_i}{Mu_i}\delta_i + \frac{T_i}{M} \\ -u_i r - \frac{c_f + c_r}{Mu_i}v_i + \frac{bc_r - ac_f}{Mu_i}r_i + \frac{c_f + T_i}{M}\delta_i \\ \frac{bc_r - ac_f}{I_z u_i}v_i - \frac{b^2c_r - a^2c_f}{I_z u_i^2}r_i + \frac{aT + ac_f}{I_z}\delta_i \end{bmatrix} - A_i\begin{bmatrix} u_i \\ v_i \\ r_i \end{bmatrix} - B_i\begin{bmatrix} \delta_i \\ T_i \end{bmatrix} \quad [2.74]$$

Using an algorithm such as those defined above, we have chosen three local models which offer a good compromise between precision and complexity. The activation functions chosen are triangular in shape and depend on the longitudinal velocity $u(t)$ (Figure 2.7c). The numerical values obtained for the different matrices A_i, B_i and D_i are:

$$A_1 = \begin{bmatrix} 0.052 & 0.403 & 0.239 \\ -0.366 & -10.82 & -13.743 \\ 0.728 & 0.388 & -11.890 \end{bmatrix}, B_1 = \begin{bmatrix} 10.99 & 7.10^{-4} \\ 91.216 & 10^{-4} \\ 60.319 & 0 \end{bmatrix}, D_1 = \begin{bmatrix} -0.832 \\ 5.259 \\ -10.46 \end{bmatrix}$$

$$A_2 = \begin{bmatrix} 0.085 & 2.895 & 1.925 \\ -0.989 & -9.282 & -16.213 \\ 0.507 & 0.333 & -10.198 \end{bmatrix}, B_2 = \begin{bmatrix} 3.359 & 7.10^{-4} \\ 91.216 & 3.10^{-4} \\ 60.319 & 2.10^{-4} \end{bmatrix}, D_1 = \begin{bmatrix} -0.087 \\ 16.562 \\ -8.496 \end{bmatrix}$$

$$A_3 = \begin{bmatrix} 0.031 & 2.065 & 0.693 \\ -1.141 & -8.468 & -17.870 \\ 0.441 & 0.303 & -9.303 \end{bmatrix}, B_3 = \begin{bmatrix} 1.548 & 7.10^{-4} \\ 91.216 & 2.10^{-4} \\ 60.319 & 10^{-4} \end{bmatrix}, D_1 = \begin{bmatrix} -0.392 \\ 20.951 \\ -8.092 \end{bmatrix} \quad [2.75]$$

Figures 2.7d) and f) show the quality of the approximation with the same control signals (see Figures 2.7 a) and b)).

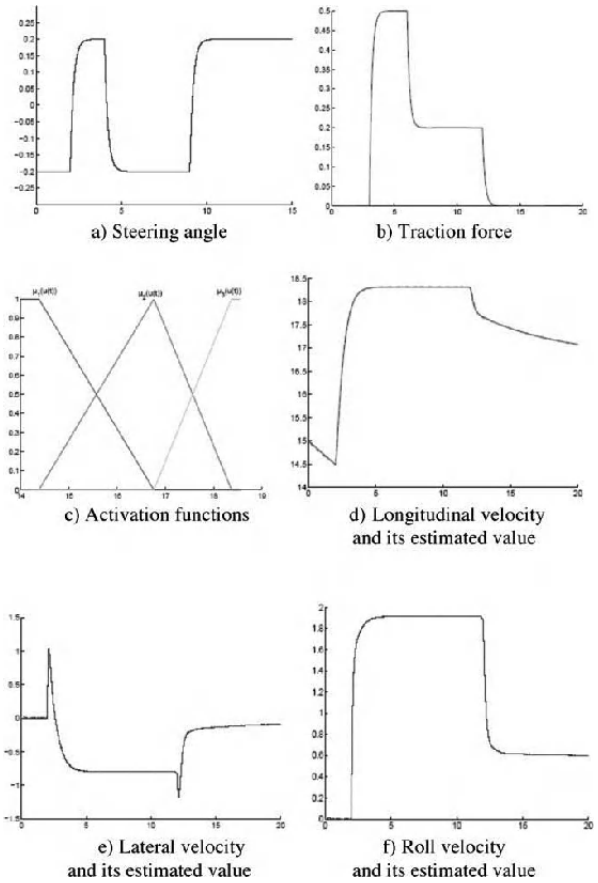


Figure 2.7. *State estimation of the vehicle dynamics*

2.4.2.3. *Obtainment of multi-models by transformation of a control affine nonlinear system*

Consider the general case of a control affine nonlinear system:

$$\begin{cases} \dot{x}(t) = f\left(x(t)\right) + B\left(x(t)\right)u(t) \\ y(t) = g\left(x(t)\right) + D(t)u(t) \end{cases} \quad [2.76]$$

where $x(.) \in \mathbb{R}^n$, $u(.) \in \mathbb{R}^m$, $y(.) \in \mathbb{R}^l$, $f\left(x(.)\right) \in \mathbb{R}^n$, $g\left(x(.)\right) \in \mathbb{R}^l$, $B\left(x(.)\right) \in \mathbb{R}^{n.m}$ and $D\left(x(.)\right) \in \mathbb{R}^{l.m}$.

The method proposed uses only the boundedness of the nonlinear terms [CHA 02b; MOR 00]. This method of transformation is based on a convex polytopic transformation of scalar functions at the root of the nonlinearity. It enables us to reduce the number of LTI models to its minimum, i.e. two LTI models. Indeed, for a function $h(x(t))$ bounded by $[a,b] \to \mathbb{R}$ for all values of $x \in [a,b]$ with $(a,b) \in \mathbb{R}^2$, there are two functions:

$$F^i(.) : [a,b] \to [0,1], i \in I_2 \quad [2.77]$$

$$x(t) \mapsto F^i\left(x(t)\right)$$

with $F^1\left(x(t)\right) + F^2\left(x(t)\right) = 1$ and two scalars α and β such that:

$$h\left(x(t)\right) = F^1\left(x(t)\right)\alpha + F^2\left(x(t)\right)\beta \quad [2.78]$$

An obvious decomposition of $h\left(x(t)\right)$ is to consider on $[a,b]$:

$$\beta \le h\left(x(t)\right) \le \alpha \quad [2.79]$$

where:

$$\beta = \min_{x \in [a,b]}\left(h\left(x(t)\right)\right), \alpha = \max_{x \in [a,b]}\left(h\left(x(t)\right)\right) \quad [2.80]$$

$$F^1\left(x(t)\right) = \frac{h\left(x(t)\right) - \beta}{\alpha - \beta}, F^2\left(x(t)\right) = \frac{\alpha - h\left(x(t)\right)}{\alpha - \beta} \quad [2.81]$$

This method of decomposition, which is not unique, will be used from hereon in.

Hereafter, we shall write the model [2.76] as:

$$\begin{pmatrix} \dot{x}(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} A(x(t)) & B(x(t)) \\ C(x(t)) & D(x(t)) \end{pmatrix} \begin{pmatrix} x(t) \\ u(t) \end{pmatrix} \qquad [2.82]$$

and take:

$$E(x(t)) = \begin{pmatrix} A(x(t)) & B(x(t)) \\ C(x(t)) & D(x(t)) \end{pmatrix} \qquad [2.83]$$

Taking the hypothesis that $E(x(t))$ is continuous and bounded and considering each of the non-constant terms, the matrix $E(x(t))$ can be transformed in the following form:

$$E(x(t)) = \sum_{i=1}^{N} \mu_i(x(t)) E_i, \; E_i = \begin{pmatrix} A_i & B_i \\ C_i & D_i \end{pmatrix} \qquad [2.84]$$

The number N of local models created by the transformation is consequently dependent on the number s of nonlinearities contained in the matrix $E(x)$. This number is equal to $N=2^s$. Examples will be given later on, illustrating the method.

The advantage to this method is that it does not cause an approximation error and it reduces the number of local models in comparison to the linearization method. Remember that reducing the number of LTI models enables us to reduce the number of matrix constraints (of stability or stabilization), which increases the likelihood of finding a solution.

EXAMPLE.– Consider the control affine nonlinear model:

$$\begin{aligned} \dot{x}(t) &= A(x(t))x(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned} \qquad [2.85a]$$

with:

$$\begin{aligned} x(t) &= \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix}, \; A(x) = \begin{pmatrix} -1 & \sin(x_2(t)) \\ 2 & -3 \end{pmatrix} \\ B &= \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \; C = \begin{pmatrix} 0 & 2 \end{pmatrix} \end{aligned} \qquad [2.85b]$$

The transformation method proposed here is based on the boundedness of the nonlinear continuous terms. Indeed, the nonlinear term is $h_1(x_2(t)) = \sin(x_2(t))$ is bounded $\forall\ x(t) \in \mathbb{R}^n$:

$$-1 \leq h_1(x_2(t)) \leq 1$$

Thus, we can transform this nonlinear term as follows:

$$h_1(x_2(t)) = -1.\mu_1(x_2) + 1.\mu_2(x_2) \quad [2.86]$$

where:

$$\mu_1(x_2) + \mu_2(x_2) = 1 \quad [2.87]$$

This enables us to find:

$$\mu_1(x_2) = \frac{1-\sin(x_2)}{2},\ \mu_2(x_2) = \frac{1+\sin(x_2)}{2} \quad [2.88]$$

Consequently:

$$\begin{aligned} \dot{x}(t) &= \sum_{i=1}^{N=2} \mu_i\left(x(t)\right) A_i x(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned} \quad [2.89]$$

where:

$$A_1 = \begin{pmatrix} -1 & -1 \\ 2 & -3 \end{pmatrix},\ A_2 = \begin{pmatrix} -1 & 1 \\ 2 & -3 \end{pmatrix} \quad [2.90]$$

Now, if we consider the same model [2.85] with:

$$B = \begin{pmatrix} 1 \\ x_1^2(t) \end{pmatrix} \text{ and } C = \begin{pmatrix} x_1^2(t) & 0 \end{pmatrix} \quad [2.91]$$

$h_1(x_2(t)) = \sin(x_2(t))$ and $h_2(x_1(t)) = x_1^2(t)$ represent the non-constant terms. Note that $h_1(x_2(t))$ is bounded no matter what the value of $x(t)$ in the state space, while

$h_2(x_1(t))$ can only be bounded on a bounded compact $[-a,a]$, $a>0$. This being the case, we can write $h_1(x_2(t))$ and $h_2(x_1(t))$ as follows:

$$h_1(x_2(t)) = F_{11}(x_2(t)).1 + F_{12}(x_2(t)).(-1)$$

$$h_2(x_1(t)) = F_{21}(x_1(t)).a^2 + F_{22}(x_1(t)).0 \qquad [2.92]$$

with:

$$F_{11}(x_2(t)) + F_{12}(x_2(t)) = 1$$

$$F_{21}(x_1(t)) + F_{22}(x_1(t)) = 1 \quad F_{21}(x_1(t)) = \frac{x_1^2(t)}{a^2};\ F_{22}(x_1(t)) = 1 - \frac{x_1^2(t)}{a^2}. \qquad [2.93]$$

$$F_{11}(x_2(t)) = \tfrac{1}{2}(1+\sin(x_2(t)));\ F_{12}(x_2(t)) = \tfrac{1}{2}(1-\sin(x_2(t)))$$

Then, we get four local models, obtained on the basis of the four possible combinations of the limits of the terms $h_1(x_1(t))$ and $h_2(x_2(t))$. These models are described by the following matrices A_i, B_i and C_i:

$$A_1 = \begin{pmatrix} -1 & 1 \\ 2 & -3 \end{pmatrix}, \quad B_1 = \begin{pmatrix} 1 \\ a^2 \end{pmatrix}, \quad C_1 = \begin{pmatrix} a^2 & 0 \end{pmatrix}$$

$$A_2 = \begin{pmatrix} -1 & -1 \\ 2 & -3 \end{pmatrix}, \quad B_2 = \begin{pmatrix} 1 \\ a^2 \end{pmatrix}, \quad C_2 = \begin{pmatrix} a^2 & 0 \end{pmatrix}$$

$$A_3 = \begin{pmatrix} -1 & 1 \\ 2 & -3 \end{pmatrix}, \quad B_3 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad C_3 = \begin{pmatrix} 0 & 0 \end{pmatrix} \qquad [2.94]$$

$$A_4 = \begin{pmatrix} -1 & -1 \\ 2 & -3 \end{pmatrix}, \quad B_4 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad C_4 = \begin{pmatrix} 0 & 0 \end{pmatrix}$$

The same number activation functions as the local models are obtained on the basis of the products $F_{1(1\,or\,2)}.F_{2(1\,or\,2)}$:

$$\mu_1(x(t)) = F_{11}(x_1(t)).F_{21}(x_2(t))$$

$$\mu_2(x(t)) = F_{11}(x_1(t)).F_{22}(x_2(t))$$

$$\mu_3(x(t)) = F_{12}(x_1(t)).F_{21}(x_2(t))$$

$$\mu_4(x(t)) = F_{12}(x_1(t)).F_{22}(x_2(t)) \qquad [2.95]$$

Finally, the corresponding multi-model is given by:

$$\begin{cases} \dot{x}(t) = \sum_{i=1}^{4} \mu_i(x(t))(A_i x(t) + B_i u(t)) \\ y(t) = \sum_{i=1}^{4} \mu_i(x(t)) C_i x(t) \end{cases} \qquad [2.96]$$

Note that the number of local LTI models depends on the number of non-constant terms contained in the matrices *A*, *B* and *C*. In general, if we have *s* non-constant terms then the multi-model comprises at most 2^s local models.

In this example, in response to a unit range, the multi-model offers a good representation of the nonlinear system in the domain $U = [-a, a] \times \mathbb{R}$ where $a > 0$ (see Figure 2.8).

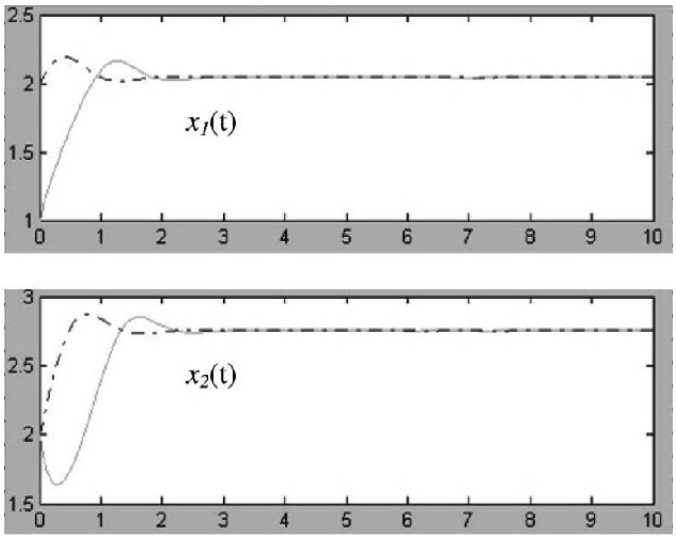


Figure 2.8. *Response of the nonlinear model [2.85] and its corresponding multi-model*

EXAMPLE.– Consider the following nonlinear model:

$$\begin{cases} \dot{x}_1(t) = 5x_1(t) + \cos\left(x_1(t)\right)x_2\left(t\right) + u(t) \\ \dot{x}_2(t) = -x_1(t) + 2x_2(t) + x_2^3(t)u(t) \\ y(t) = x_2^3(t)x_1\left(t\right) + x_2(t) \end{cases} \quad [2.97]$$

This can be rewritten in the form of equation [2.82] with:

$$\begin{aligned} &A\left(x(t)\right) = \begin{pmatrix} 5 & \cos\left(x_1(t)\right) \\ -1 & 2 \end{pmatrix}, B\left(x(t)\right) = \begin{pmatrix} 1 \\ x_2^3(t) \end{pmatrix} \\ &C\left(x(t)\right) = \begin{pmatrix} x_2^3(t) & 1 \end{pmatrix}, D\left(x(t)\right) = 0 \end{aligned} \quad [2.98]$$

The aim is to write $E\left(x(t)\right)$ as defined in equation [2.83] in the form of [2.84]. The matrix $E\left(x(t)\right)$ exhibits $s = 2$ non-constant terms:

$$h_1\left(x_1(t)\right) = \cos\left(x_1(t)\right), h_2\left(x_2(t)\right) = x_2^3(t) \quad [2.99]$$

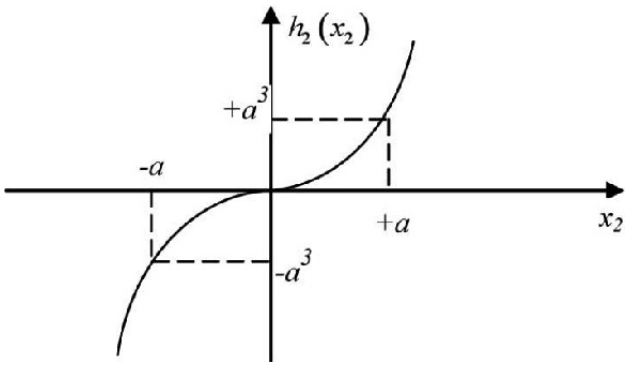


Figure 2.9. *The non-constant term* $h_2\left(x_2(t)\right) = x_2^3(t)$

Note that the non-constant term $h_1\left(x_1(t)\right)$ is bounded $\forall\ x(t) \in \mathbb{R}^2$, whereas $h_2\left(x_2(t)\right)$ can only be bounded on a bounded compact $\left[-a,a\right]$, $a > 0$ (see Figure 2.9). Thus, we can transform the nonlinear terms $h_1\left(x_1(t)\right)$ and $h_2\left(x_2(t)\right)$ $\forall x\left(t\right) \in U$ with $U = \mathbb{R} \times \left[-a,a\right], a > 0$ such that:

$$h_1(x_1(t)) = F_1^1(x_1).1 + F_1^2(x_1).(-1) \quad [2.100a]$$

$$h_2(x_2(t)) = F_2^1(x_2).a^3 + F_2^2(x_2).\left(-a^3\right) \quad [2.100b]$$

where:

$$F_1^1(x_1) = \frac{1}{2}\left(1+\cos(x_1(t))\right), F_1^2(x_1) = \frac{1}{2}\left(1-\cos(x_1(t))\right) \quad [2.101a]$$

$$F_2^1(x_2) = \frac{1}{2a^3}\left(x_2^3(t)+a^3\right), F_2^2(x_2) = \frac{1}{2a^3}\left(a^3-x_2^3(t)\right) \quad [2.101b]$$

Thus, we obtain four local models, obtained on the basis of the four possible combinations of the limits of the non-constant terms of $h_1(x_1(t))$ and $h_2(x_2(t))$. These four models are represented by the matrices E_i described in equation [2.83]:

$$E_1 = \begin{pmatrix} 5 & 1 & 1 \\ -1 & 2 & a^3 \\ a^3 & 1 & 0 \end{pmatrix}, E_2 = \begin{pmatrix} 5 & 1 & 1 \\ -1 & 2 & -a^3 \\ -a^3 & 1 & 0 \end{pmatrix} \quad [2.102]$$

$$E_3 = \begin{pmatrix} 5 & -1 & 1 \\ -1 & 2 & a^3 \\ a^3 & 1 & 0 \end{pmatrix}, E_4 = \begin{pmatrix} 5 & -1 & 1 \\ -1 & 2 & -a^3 \\ -a^3 & 1 & 0 \end{pmatrix}_2 \quad [2.103]$$

Taking the product operator as a conjunction operator, the four activation functions are obtained on the basis of the products $F_1^{1\,or\,2}.F_2^{1\,or\,2}$:

$$\mu_1(x(t)) = F_1^1(x_1).F_2^1(x_2),\ \mu_2(x(t)) = F_1^1(x_1).F_2^2(x_2) \quad [2.104]$$

$$\mu_3(x(t)) = F_1^2(x_1).F_2^1(x_2),\ \mu_4(x(t)) = F_1^2(x_1).F_2^2(x_2) \quad [2.105]$$

The multi-model description of the nonlinear system [2.97] is only valid in the domain $U = \mathbb{R}\times[-a,a]$ of the state space. Consequently, the analysis of the stability of [2.97] based on its multi-model [2.47] is only local, even if the stability established for this latter is absolute. In the case of matrices $E(x(t))$ which are

bounded $\forall x(t) \in \mathbb{R}^p$, the multi-model representation is identical to the nonlinear model. Remember that, as a general rule, if the matrix $E(x(t)$ presents s non-constant terms then the multi-model is composed of at most 2^s local models.

2.5. Bibliography

[AKH 07a] AKENAK A., CHADLI M., MAQUIN D., *et al.*, "Design of a sliding mode fuzzy observer for uncertain Takagi-Sugeno fuzzy model: application to automatic steering of vehicles", *International Journal of Vehicle Autonomous Systems (IJVAS)*, vol. 5, no. 3/4, pp. 288–305, 2007.

[AKH 07b] AKENAK A., CHADLI M., RAGOT J., *et al.*, "Multiple model approach modelling: application to a turbojet engine", *Journal of Engineering and Applied Sciences*, vol. 2, no. 4, pp. 798–807, 2007.

[BOR 92] BORNE P., DAUPHIN-TANGUY G., RICHARD J.-P., et al., Modélisation et identification des processus, tome 1, Technip, Paris, 1992.

[BOR 90] BORNE P., DAUPHIN-TANGUY G., RICHARD J.-P., et al., Commande et optimisation des processus, Technip, Paris, 1990.

[BOR 01] BORNE P., BEN ABDENNOUR R., KSOURI M., *et al.*, *Identification et commande numérique des procédés industriels*, Technip, Paris, 2001.

[BUC 93] BUCKLEY J.J., "Sugeno type controllers are universal controllers", *Fuzzy Sets and Systems*, vol. 53, no. 3, pp. 299–303, 1993.

[CAS 95] CASTRO J., "Fuzzy logic controllers are universal approximators", *IEEE Transactions on Systems, Man and Cybernetics*, vol. 25, pp. 629–635, April 1995.

[CHA 02a] CHADLI M., MAQUIN D., RAGOT J., "Output stabilization in multiple model approach", *International Conference on Control Applications*, vol. 2, pp. 1315–1320, 2002.

[CHA 02b] CHADLI M., Stabilité et stabilisation des multimodèles, Doctoral thesis, INPL-CRAN, Nancy, December 2002.

[CHA 07a] CHADLI M., BORNE P., "Design of robust static output controller for uncertain fuzzy model: an LMI formulation", *Studies in Informatics and Control*, vol. 16, no. 4, pp. 421–430, 2007.

[CHA 07b] CHADLI M., ELHAJJAJI A., "Moment robust fuzzy observer-based control for improving driving stability", *International Journal of Vehicle Autonomous Systems*, vol. 5, no. 3/4, pp. 326–344, 2007.

[CHA 08a] CHADLI M., AKHENAK A., RAGOT J., *et al.*, "On the design of observer for unknown inputs Fuzzy models", *International Journal of Automation and Control*, vol. 2, no. 1, pp. 113–125, 2008.

[CHA 08b] CHADLI M., "Multimodèles: origine et méthodes d'obtentions", *Revue d'électricité et d'électronique*, REE-SEE, no. 10, pp. 51–54, 2008.

[CHA 08c] CHADLI M., AKHENAK A., MAQUIN D., *et al.*, "Fuzzy observer for fault detection and reconstruction of unknown input fuzzy models", *International Journal of Modelling, Identification and Control*, REE-SEE, vol. 3, no. 2, pp. 193–200, 2008.

[CHA 08d] CHADLI M., BORNE P., "Stabilité des multimodèles", *Revue d'électricité et d'électronique*, no. 10, pp. 55–59, 2008.

[CHA 08e] CHADLI M., "Commande et observation des multimodèles", *Revue d'électricité et d'électronique*, REE-SEE, no. 10, pp. 60–65, 2008.

[CHA 09] CHADLI M., AKHENAK A., RAGOT J., *et al.*, "State and unknown input estimation for discrete time multiple model", *Journal of the Franklin Institute*, vol. 346, no. 6, pp. 593–610, 2009.

[CHA 10a] CHADLI M., "Chaotic systems reconstruction", *Evolutionary Algorithms and Chaotic Systems*, vol. 267, Springer-Verlag, Berlin, p. 560, 2010.

[CHA 10b] CHADLI M., "State and an LMI approach to design observer for unknown inputs Takagi-Sugeno fuzzy models", *Asian Journal of Control*, vol. 12, no. 4, 2010.

[EYK 74] EYKHOFF P., *System Identification: Parameter and State Estimation*, John Wiley, London, 1974.

[FRA 90] FRANK P.-M., "Fault diagnosis in dynamic system using analytical and knowledge based redundancy – a survey and some new results", *Automatica*, vol. 26, no. 3, pp. 459–474, 1990.

[GAS 01] GASSO K., MOUROT G., RAGOT J., "Structure identification in multiple model representation: elimination and merging of local models", *IEEE Conference on Decision and Control*, vol. 3, pp. 2992–2997, 2001.

[GRA 03] GRANJON Y., *Automatique, Systems linéaires, non linéaires, à temps continu, à temps discret, représentation d'état*, Dunod, Paris, 2003.

[GRA 76] GRAUPE D., *Identification of Systems – 2nd edition*, Robert E. Krieger Publishing Company, New York, NY, 1976.

[FLA 94] FLAUS J.M., *La régulation industrielle*, Hermès, Paris, 1994.

[JOH 00] JOHANSEN T.A., SHORTEN R., MURRAY-SMITH R., "On the interpretation and identification of dynamic Takagi-Sugeno fuzzy models", *IEEE Transactions on Fuzzy Systems*, vol. 8, no. 3, pp. 297–312, 2000.

[JOH 03] JOHANSEN T.A., BABUSKA R., "Multiobjective identification of Takagi-Sugeno fuzzy models", *IEEE Transactions on Fuzzy Systems*, vol. 11, no. 6, pp. 847–860, 2003.

[LAN 02] LANDAU I.D., *Commande des systèmes – conception, identification et mise en œuvre*, Hermès, Paris, 2002.

[LJU 87] LJUNG L., *System Identification – Theory for the User*, Prentice Hall PTR, Englewood Cliffs, NY, 1987.

[MOR 00] MORERE Y., Mise en œuvre de lois de commandes pour les modèles flous de type Takagi-Sugeno, Doctoral thesis, University of Valenciennes and Haut Cambrésis, Lille, 2000.

[MUR 97] MURRAY-SMITH R., *Multiple Model Approaches to Modelling and Control*, Taylor & Francis, London, 1997.

[PAT 97] PATTON R.J., CHEN J., "Observer-based fault detection and isolation: robustness and applications", *Control Engineering Practice*, vol. 5, no. 5, pp. 671–682, 1997.

[TAK 85] TAKAGI M., SUGENO M., "Fuzzy identification of systems and its application to modelling and control", *IEEE Transactions on Systems Man and Cybernetics*, vol. 15, no. 1, pp. 116–132, 1985.

[ZAD 65] ZADEH L., "Fuzzy sets", *Information Control*, vol. 8, pp. 338–353, 1965.

Chapter 3

Control Tools

3.1. Linear controls

3.1.1. *The PID corrector*

The PID is, without contest, the most popular and most widely-used controller. Its quick and simple approach is able to solve a number of control-loop problems. The earliest commercially-available PIDs appeared in the 1930s, with studies on their parametric optimization coming out in the 1940s [ZIG 42]. Since then, they have constantly been used in industry (analog, pneumatic, electronic and digital PIDs).

The popularity of the PID controller is based on simple points: widespread use, indicating that it has proven itself; relative simplicity of the regulation (three parameters); an implementation method (apparently) requiring little experimentation.

Hans Eder speaks precisely about the phenomenon surrounding the PID corrector. He explains that the controller is largely used by professionals because it "wrongly" appears to require no preparatory work or prior testing. This misguided approach can, unfortunately, lead to drastic errors in regulation. The most surprising realization is that most users have no more knowledge in the workings of the PID controller than twenty or thirty years ago [EDE 05].

The PID is a controller which, like most controllers, requires testing and planes of experiments to identify and model the process to be regulated. The simplicity of

---

Chapter written by Mohammed CHADLI and Hervé COPPIER

tuning a PID may deceptively appear to legitimize the absence of testing, but this is absolutely not so. Certainly, the strength of the PID lies in its use of only three parameters, but it does require effort to determine the correct regulation.

The strength of this controller is thus that it combines three elementary actions to correct the system: the proportional corrector, the integral corrector and the derivative corrector. All three have the merit of performing concrete actions on the measured difference without having a link (*a priori*) about the knowledge of the process. In the structure of PIDs, there is no part which can, at a certain level, integrate any form of simulation. It may be, for this reason, that experimentation may appear to be no longer a necessity, although this is absolutely not true.

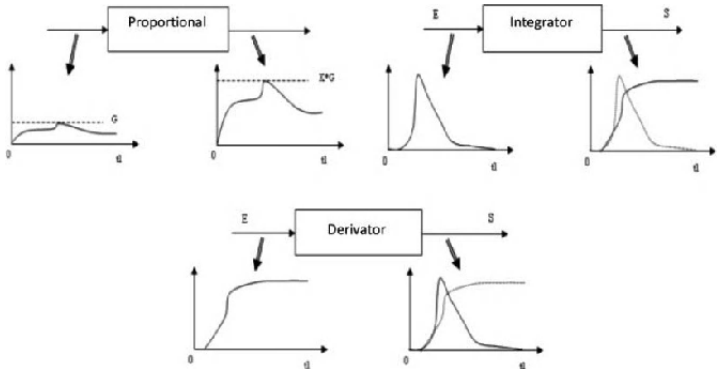


Figure 3.1. *Elementary actions of the PID corrector*

The methods for regulating PIDs are many [FLA 94; PIG 96]. One might cite:

– successive approach regulation;

– the Ziegler–Nichols method (temporal and frequential);

– the Takahashi method;

– the criterion optimization methods advanced by Zhuang and Atherton [ZHU 93].

3.1.2. *The Smith predictor*

3.1.2.1. *History*

The Smith predictor owes its name to O.J. Smith who, in 1959, put forward a technique to compensate for the delay in a system by adjusting a controller, used the

traditional PID [SMI 59]. This solution consists of eliminating the delay for the closed-loop system. Numerous methods called modified Smith predictors, internal model corrector (IMC) or dead time compensator (DTC), have since been developed.

In 1968, A.T. Fuller proposed an optimal nonlinear controller [FUL 68]. G. Alevisakis, in 1973, extended the SISO solution of the original Smith predictor to a multi-variable solution [ALE 73]. In 1977, J.F. Donoghue carried out an in-depth study of mono- and multi-variable methods and of the optimal approach to the use of the controller [DON 77]. In 1981, K. Watanabe developed a modified structure for the predictor, based on a PI or a PID so as to have good disturbance rejection and a null static error [WAT 81]. In 1994, K.J. Astrom put forward a structure for integrator systems subjected to a disturbance [AST 94]. M.R. Matausek, in 1996, found an effective way of parameterizing a slightly altered corrector, similar to that of Astrom [MAT 96].

3.1.2.2. *Choices and principles*

The original Smith predictor is used for the stable systems (first and second order) and the modified Matausek predictor for the integrator systems. These choices are the result of technical and programming compromises, made so as to cover the broadest possible range of scenarios likely to be encountered.

The principle of the Smith predictor is simple. It is to find a structure such that the pure delay no longer plays any part in the characteristic equation for the closed-loop system.

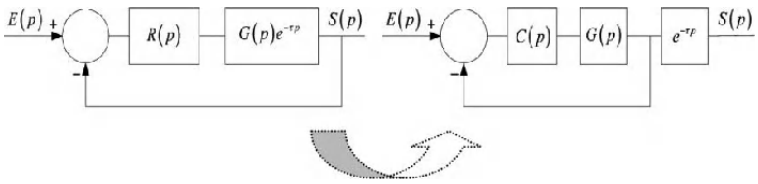


Figure 3.2. *Principle of the Smith predictor*

In all cases, we have:

$$S(p) = \frac{R(p)G(p)e^{-\tau p}}{1 + R(p)G(p)e^{-\tau p}} \qquad [3.1]$$

$$S(p) = \frac{C(p)G(p)e^{-\tau p}}{1 + C(p)G(p)} \qquad [3.2]$$

The equivalence of the two systems is then made for:

$$R(p) = \frac{C(p)}{1 + C(p)G(p)(1 - e^{-\tau p})} \quad [3.3]$$

3.1.2.2.1. Second-order example

Consider that the process is modeled by a delayed second-order system G:

$$G(p) = \frac{G_s . e^{-\tau p}}{(1 + Tp)^2} \quad [3.4]$$

The functional diagram of the predictor is defined in Figure 3.3.

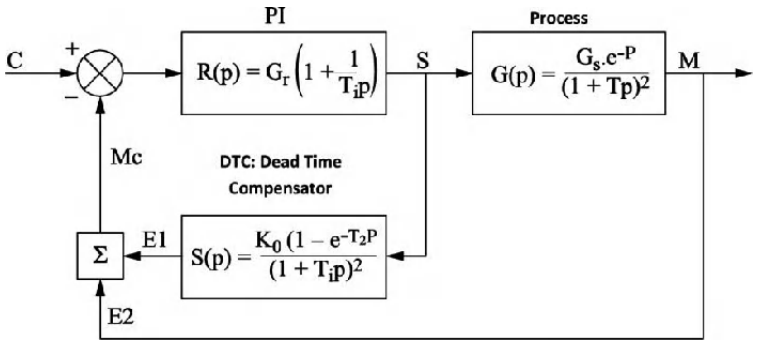


Figure 3.3. *Functional diagram of the Smith predictor for a second-order stable system*

The Smith predictor can also be represented by Figure 3.4.

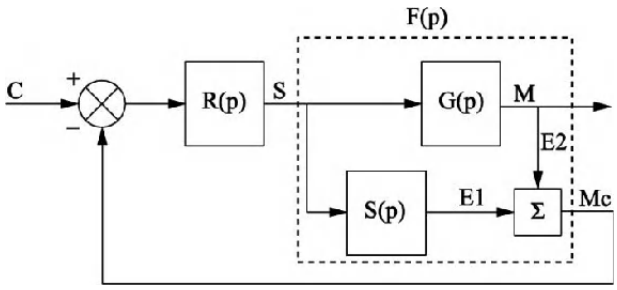


Figure 3.4. *Another functional diagram of the Smith predictor*

F is thus the transfer function experienced by the PI corrector such that:

$$\begin{aligned} F(p) = G(p) + S(p) &= \frac{G_s e^{-\tau p}}{(1+Tp)^2} + \frac{K_0(1-e^{-T_2 p})}{(1+T_1 p)^2} \\ &= \frac{G_s e^{-\tau p}}{(1+Tp)^2} + \frac{K_0}{(1+T_1 p)^2} - \frac{K_0 e^{-T_2 p}}{(1+T_1 p)^2} \\ \Leftrightarrow F(p) &= \left.\frac{G_s}{(1+Tp)^2}\right|_{\substack{K_0=G_s \\ T_1=T \\ T_2=\tau}} \end{aligned} \quad [3.5]$$

The delay is indeed eliminated from the point of view of the PI controller. Consequently, for a closed-loop system we have the following second-order equation:

$$\begin{aligned} BF = \frac{BO}{1+BO} &= \frac{F(p)\cdot R(p)}{1+F(p)\cdot R(p)} = \frac{\frac{G_s G_r}{(1+Tp)^2}\cdot\left(\frac{1+T_i p}{T_i p}\right)}{1+\frac{G_s}{(1+Tp)^2}\cdot\left(\frac{1+T_i p}{T_i p}\right)} = \frac{\frac{G_s G_r}{(1+Tp)}\cdot\frac{1}{Tp}}{1+\frac{G_s G_r}{(1+Tp)}\cdot\frac{1}{Tp}} \\ \Leftrightarrow BF &= \frac{G_s G_r}{(1+Tp)Tp + G_s G_r} = \frac{1}{1+\left(\frac{Tp+T^2p^2}{G_s G_r}\right)} \\ \Leftrightarrow BF &= \frac{1}{1+\frac{T}{G_s G_r}p + \frac{T^2}{G_s G_r}p^2} = \frac{1}{1+ap+bp^2} \end{aligned} \quad [3.6]$$

The damping factor z (often ~ 0.7) of this second-order system is thus defined such that:

$$\begin{aligned} z &= \frac{a}{2\sqrt{b}} = \frac{\frac{T}{G_s G_r}}{2\sqrt{\frac{T^2}{G_s G_r}}} = \frac{1}{2\sqrt{G_s G_r}} \\ \Rightarrow z^2 &= \frac{1}{4.G_s G_r} \Leftrightarrow G_r = \frac{1}{4.G_s.z^2} \end{aligned} \quad [3.7]$$

This means that for the regulation of the Smith predictor for a delay second-order system:

$$S(p) = \frac{K_0(1-e^{-T_2 p})}{(1+T_1 p)^2} = \frac{G_s(1-e^{-\tau p})}{(1+Tp)^2}$$
$$R(p) = G_r\left(1+\frac{1}{T_i p}\right) = \frac{1}{4.G_s.z^2}\left(1+\frac{1}{Tp}\right) \quad [3.8]$$

Matausek's method consists of employing a conventional Smith predictor with a modification of the difference between the process and the model with a view to compensating for that difference.

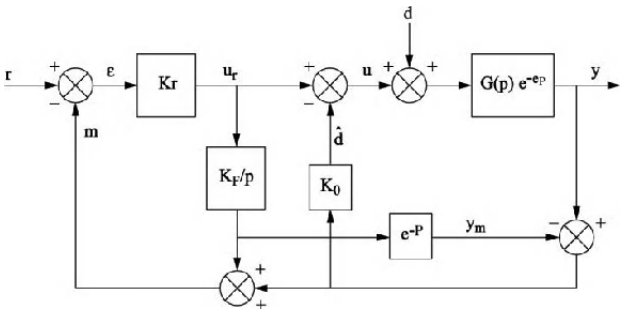


Figure 3.5. *Matausek's Smith predictor*

The system is defined such that:

$$Y(p) = H_r(p).R(p) + H_d(p).D(s) \quad [3.9]$$

If we consider that "model = system", then:

$$G(p) = \frac{K_p}{p}; \tau = \theta \quad [3.10]$$

Hence:

$$H_r(p) = \frac{K_p.K_r.e^{-\tau p}}{1+K_p.K_r} \quad [3.11]$$

$$H_d(p) = \frac{K_p.\left[p+K_pK_r\left(1-e^{-\tau p}\right)\right].e^{-\tau p}}{\left(p+K_p.K_r\right).\left(p+K_p.K_0.e^{-\tau p}\right)} \quad [3.12]$$

From equation [3.12], we can see that:

$$\lim_{p\to 0} H_d(p) = 0, \qquad K_0 \neq 0$$
$$\lim_{p\to 0} H_d(p) = \frac{1+K_p K_r \tau}{K_r}, \qquad K_0 = 0 \qquad [3.13]$$

Hence, we need to find a non-zero value for K_0 in order to ensure disturbance rejection in the stationary regime. In addition, in view of the structure and of equation [3.13], it can be said that:

$$\lim_{t\to+\infty} \hat{d} = d, \qquad K_0 \neq 0 \qquad [3.14]$$

The tunings suggested by Matausek are:

$$K_0 = \frac{\pi}{2.K_p.\tau}, \qquad Kr = \frac{1}{K_p.T_r} \qquad [3.15]$$

where T_r is the time constant chosen for the first-order closed-loop system.

Thanks to the Smith predictor (both conventional and modified), regulation loops with significant dead times can be corrected. Because of its simple and effective implementation, the Smith predictor offers a functionality that is highly interesting for the GCS project.

3.1.3. *Predictive functional control*

Predictive functional control (PFC) was introduced in the 1980s by J. Richalet [BOU 96]. It belongs to the family of predictive control systems using internal models (collectively referred to as model-based control – MBC). The principle of this control is structured around four crucial concepts: the model, the reference trajectory, structured control and compensation for the error between the model and the process (auto-compensator) [RIC 04; RIC 93].

3.1.3.1. *Principles*

In general, we consider the system to be regulated as a system that can be modeled in accordance with the ARMA representation:

$$y_M(n) = \sum a_i.y_p(n-i) + \sum b_j.u(n-j) \qquad [3.16]$$

Then, PFC is based on principles which are peculiar to the implementation of this method.

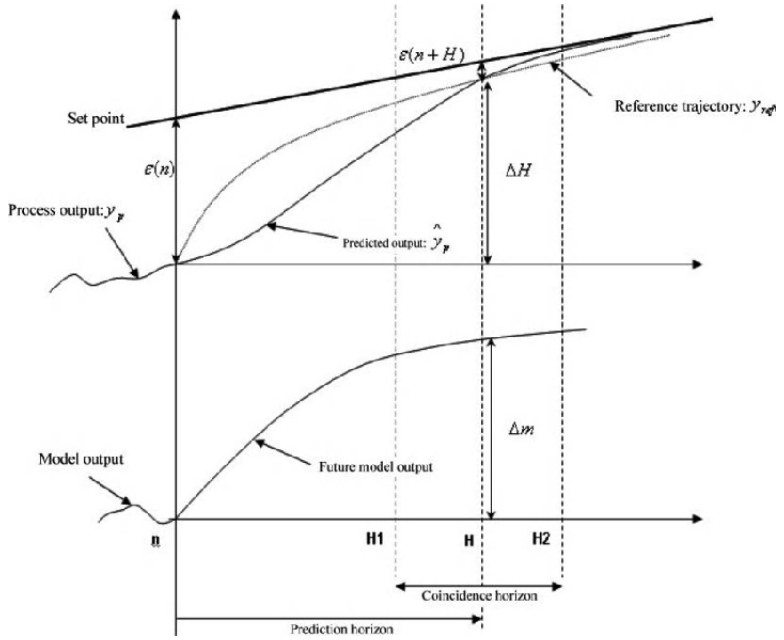


Figure 3.6. *Principle of predictive functional control (PFC)*

The first interesting principle of predictive functional control is the use of a reference trajectory that helps specify the dynamic behavior in a closed-loop system. Generally, the exponential is taken as a reference, so that with each passing sample, the difference between the measurement and the set point should be exponentially lesser (see Figure 3.6). If we use the notation $\varepsilon(n)$ for the difference at time n, we have:

$$\begin{aligned} &\varepsilon(n) = e^{-nT_e/T} \\ &\varepsilon(n+1) = e^{-nT_e/T}.e^{-\frac{T_e}{T}} = \varepsilon(n).e^{-\frac{T_e}{T}} = \varepsilon(n).\lambda \\ &\varepsilon(n+H) = \varepsilon(n).\lambda^H \end{aligned} \qquad [3.17]$$

The second fundamental principle is based on the coincidence horizon (see Figure 3.6). This horizon is a specification to the control system to attempt to match the reference trajectory and the measurement. We quite deliberately speak of a coincidence point if the horizon is limited to a particular point.

When the desired dynamic is specified (horizon and reference trajectory), it is necessary to use a control (a manipulated variable) to ensure the desired behavior.

PFC introduces a very particular notion: that of structuring the manipulated variable. This is tantamount to choosing the future control from a database of predefined functions $U_{Bk}(i)$, so:

$$MV(n+i) = \sum_{k=0}^{k_M} \mu_k U_{Bk}(i) \qquad [3.18]$$

By linear superposition, each basic function causes a basic output $y_{Bk}(i)$ from the model (known in advance, calculable or tabulated for a given model):

$$u_{Bk}(i) \rightarrow y_{Bk}(i) \qquad [3.19]$$

Thereafter, we simply need to put the control system in equation form. In order to illustrate this approach, we *have chosen to have the control system* constructed using a single coincidence point H. Thus, we search for the desired increment $\Delta m = \Delta H$ of the output of the process at time $n + H$ (see Figure 3.6). At H, we can say that:

$$\begin{aligned}
&\varepsilon(n) = C(n) - s_p(n) = C(n) - y_p(n) && \text{[reference point – process output at time } n\text{]}\\
&\varepsilon(n+H) = C(n) - y_{reference}(n+H) && \text{[at coincidence point } H\text{]}\\
&\Delta H = y_{reference}(n+H) - y_p(n) && \text{[incremental objective given to the model]}
\end{aligned} \qquad [3.20]$$

We therefore have:

$$\begin{aligned}
&\varepsilon(n+H) = C(n) - y_{reference}(n+H) = \varepsilon(n).\lambda^H = \left(C(n) + y_p(n)\right).\lambda^H\\
&\Leftrightarrow y_{reference}(n+H) = C(n) - \left(C(n) + y_p(n)\right).\lambda^H\\
&\Leftrightarrow y_{reference}(n+H) - y_p(n) = \left(C(n) - y_p(n)\right).\left(1 - \lambda^H\right)
\end{aligned} \qquad [3.21]$$

This gives the objective for the control system to achieve:

$$\Delta H = \left(C(n) - y_p(n)\right).\left(1 - \lambda^H\right) \qquad [3.22]$$

The control equation at point *H* can finally be summarized as:

$$\begin{aligned} &\Delta H = \Delta m \\ &\Leftrightarrow \left(C(n) - y_p(n)\right).\left(1-\lambda^H\right) = y_{Released,\,Model}(n+H) \\ &\quad + y_{Forced,\,Model}(n+H) - y_{Model}(n) \end{aligned} \qquad [3.23]$$

This last expression is valid for a single coincidence point.

3.1.3.2. *Resolution: examples*

3.1.3.2.1. First-order system

Consider a system governed by a first-order law and represented by its differential equation:

$$T\dot{y} + y = Ku \qquad [3.24]$$

The solution to this equation is:

$$y(t) = y_{released}(t) + y_{forced}(t) = y_0.e^{-t/T} + K(1-e^{-t/T}) \qquad [3.25]$$

Now we have to solve the control equation [3.23]. In order to successfully solve this problem, it is interesting to work on the discrete, first-order model and then to work on the manipulated variable. Thus, consider the first-order equation from [3.24] in a Laplacian regime:

$$G(p) = \frac{S(p)}{E(p)} = \frac{K}{1+Tp} \qquad [3.26]$$

In a discrete regime, we obtain the equation in terms of *z*:

$$\begin{aligned} &G(z) = \frac{S(z)}{E(z)} = \frac{b.z^{-1}}{1+a.z^{-1}} = \frac{K(1-e^{-T_e/T}).z^{-1}}{1-e^{-T_e/T}.z^{-1}} \\ &\Leftrightarrow S(z)(1-e^{-T_e/T}.z^{-1}) = E(z).K(1-e^{-T_e/T}).z^{-1} \\ &\Leftrightarrow S(n) = a_m S(n-1) + K(1-a_m)E(n-1)\big|_{a_m = e^{-T_e/T}} = S(n)_{Free} + S(n)_{Forced} \end{aligned} \qquad [3.27]$$

By recurrence, and considering that the basic function is the level *MV(n)=MV(n + 1)*, we can deduce that:

$$S(n+1) = a_m S(n) + K(1-a_m)E(n)$$
$$\Leftrightarrow S(n+1) = a_m \left[a_m S(n-1) + K(1-a_m)E(n-1)\right] + K(1-a_m)E(n)$$
$$\Leftrightarrow S(n+1) = a_m^2 S(n-1) + Ka_m(1-a_m)E(n-1) + K(1-a_m)E(n)\Big|_{E(n)=M(n)} \quad [3.28]$$

$$\Leftrightarrow S(n+1) = a_m^2 S(n-1) + K(1-a_m^2)MV(n)$$

By recurrence, we can deduce that:

$$S(n+h) = a_m^h S(n) + K(1-a_m^h)MV(n) \quad [3.29]$$

In the control equation, we thus obtain the manipulated variable *MV(n)*:

$$\Delta H = \left(C(n) - y_p(n)\right).\left(1-\lambda^H\right) = y_L(n+H) + y_F(n+H) - y_M(n)$$
$$\Leftrightarrow \left(C(n) - y_p(n)\right).\left(1-\lambda^H\right) = a_m^h S_M(n) + K(1-a_m^h)MV(n) - S_M(n) \quad [3.30]$$
$$MV(n) = \frac{\left(C(n) - y_p(n)\right).\left(1-\lambda^h\right) + S_M(n) - a_m^h S_M(n)}{K(1-a_m^h)}$$

For the use of PFC for a first-order system, the expression of the *MV(n)* control is simple. Its implementation is also simplified.

3.1.3.2.2. Application to a system of any order on the basis of its indicial response

Consider an asymptotic stable system defined by its representation of convolution on N coefficients a_i. The indicial response may have any shape:

$$y_P(n) = a_1^P .u(n-1) + \ldots + a_i^P .u(n-i) + \ldots + a_N^P .u(n-N) = \sum_{i=1}^{N} a_i^P .u(n-i) \quad [3.31]$$

(This system is an MA (Moving Average) real-time system $s_k = \sum_{i=0}^{p} a_i e_{k-i}$).

In order to determine the control, we suppose that the set point varies in level and that the reference trajectory is the exponential defined above by equation [3.16]. The basic function is the level, which brings a single unknown coefficient μ_0 which can be determined by coincidence limited to a single point.

Hence, the associated model is:

$$\begin{aligned} & y_M(n) = a_1^M.u(n-1) + ... + a_i^M.u(n-i) + ... + a_N^M.u(n-N) \\ & \Leftrightarrow y_M(n) = \sum_{i=1}^{N} a_i^M.u(n-i) \end{aligned} \quad [3.32]$$

At time $n + H$, the model becomes:

$$\begin{aligned} & y_M(n+H) = a_1^M.u(n-1+H) + ... + a_i^M.u(n-i+H) + ... + a_N^M.u(n-N+H) \\ & \Leftrightarrow y_M(n+H) = a_1^M.u(n-1+H) + a_2^M.u(n-2+H) + ... \\ & \qquad + a_H^M.u(n-H+H) \quad + a_{H+1}^M.u(n-1) \quad + ... \\ & \qquad + a_{H+i}^M.u(n-i) \quad + ... + \quad a_{H+i=N}^M.u(n-N+H) \end{aligned} \quad [3.33]$$

$$\Leftrightarrow y_M(n+H) = a_1^M.u(n-1+H) + .. + a_H^M.u(n) + ... + a_N^M.u(n-N+H)$$

and because for the future $u(n+i+H) = u(n), \forall i \in [1; H]$, we obtain the following result for the model:

$$y_M(n+H) = u(n).\sum_{i=1}^{H} a_i^M + \sum_{i=H+1}^{N} a_i.u(n-i+H) \quad [3.34]$$

We shall use the following notation:

$$\begin{aligned} & A_H^T = \left\{ a_{H+1}^M, ... a_N^M \right\} \\ & A^T = \left\{ a_1^M, ... a_N^M \right\} \\ & U^T(n) = \left\{ u(n-1), u(n-2), ... u(n-N+H) \right\} \end{aligned} \quad [3.35]$$

The control equation for an asymptotic stable system defined by its representation of convolution is naturally obtained:

$$\begin{aligned} & \Delta H = \left(C(n) - y_p(n) \right).\left(1 - \lambda^H \right) = y_L(n+H) + y_F(n+H) - y_M(n) \\ & \Leftrightarrow \left(C(n) - y_p(n) \right).\left(1 - \lambda^H \right) = u(n).\sum_{i=1}^{H} a_i^M + A_H^T.U(n) - A^T.U(n) \\ & \Leftrightarrow u(n) = \frac{\left(C(n) - y_p(n) \right).\left(1 - \lambda^H \right) - A_H^T.U(n) + A^T.U(n)}{\sum_{i=1}^{H} a_i^M} \end{aligned} \quad [3.36]$$

On the basis of a simple measurement of the system's response to a level of input (any auto-stable system), PFC enables us to compute a control law with chosen values for the horizon and reference trajectory.

3.1.3.3. *Regulation of PFC*

The regulation of PFC must be able to deliver acceptable performances. Richalet's method can be parameterized by choosing the reference trajectory, the coincidence horizon and the basic function. Table 3.1 gives an overview of the influences between performances and parameters.

	Precision	**Dynamic**	**Robustness**
Basic function	2	0	0
Reference trajectory	0	2	1
Coincidence horizon	0	1	2

Table 3.1. *Performances of PFC and parameters for the controller [RIC 93]*

3.1.4. *Generalized predictive control*

Generalized predictive control (GPC) is the term employed by Clarke, Mohtadi and Tuffs in the proposal for a formulation of model-based control [CLA 87]. The appellation GPC has since been widely adopted as a denomination for a group of adaptive predictive control methods [CLA 87].

GPC is a complete structure which ensures we get a stable corrected system for a particular set of elaborated parameters. This methodology can be used to solve problems in the area of:

– processes with non-minimal phase-shifting;

– intrinsically unstable systems, or which have incorrectly damped poles;

– systems whose dead times are variable or unknown;

– systems of unknown order [CLA 87].

The principle of predictive control is in the creation of an anticipative effect. The implementation of this control requires:

– the elaboration of a model of the system needing to be controlled, with a view to predicting its future behavior;

– the implementation of a sequence of future commands by use of a minimization criterion (quadratic cost function);

– the renewal of the procedure at each iteration. Only the first command is taken into account and applied to the system.

3.1.4.1. *Principles of the use of GPC [BOU 96; RAM 01]*

For historical reasons, GPC is put into practice on the basis of the model represented in the form of CARMA (controlled autoregressive moving average):

$$A(q^{-1})y(t) = B(q^{-1})u(t-1) + x(t) \qquad [3.37]$$

where *y(t)* is the process output, *u(t)* the command applied to the system, q^{-1} is the delay operator, $x(t)$ a term linked to the disturbances, usually chosen in the form $x(t)=C(q^{-1})\xi(t)$ with $\xi(t)$ being a centered uncorrelated random sequence.

and the polynomials defined such that:

$$\begin{aligned} A(q^{-1}) &= 1 + a_1 q^{-1} + \ldots + a_n q^{-n} \\ B(q^{-1}) &= b_0 + b_1 q^{-1} + \ldots + b_m q^{-m} \\ C(q^{-1}) &= 1 + c_1 q^{-1} + \ldots + c_l q^{-l} \end{aligned} \qquad [3.38]$$

The system becomes:

$$A(q^{-1})y(t) = B(q^{-1})u(t-1) + C(q^{-1})\xi(t) \qquad [3.39]$$

We introduce an integral action $\Delta(q^{-1}) = 1 - q^{-1}$ to eliminate the static error; thus the model which will be used to implement the sequence of future commands is:

$$A(q^{-1})y(t) = B(q^{-1})u(t-1) + C(q^{-1})\frac{\xi(t)}{\Delta(q^{-1})} \qquad [3.40]$$

This latter equation is the CARIMA model (standing for Controlled AutoRegressive Integrated Moving Average). Hereafter, we take $C(q^{-1}) = 1$, looking only at the input/output transfer functions for which this polynomial has no influence.

GPC uses the concept of an optimal predictor, the aim of which is to anticipate the behavior of the process on a j-step in the future on a finite horizon.

Based on the model from equation [3.39], we elaborate the predicted output:

$$y(t+j) = \underbrace{F_j(q^{-1})y(t) + H_j(q^{-1})\Delta u(t-1)}_{past} + \underbrace{+ G_j(q^{-1})\Delta u(t+j-1) + J_j(q^{-1})\xi(t)}_{future} \quad [3.41]$$

From equation [3.40], we have:

$$A(q^{-1})\Delta(q^{-1})y(t) = B(q^{-1})\Delta u(t-1) + \xi(t) \quad [3.42]$$

At $t+j$, we get:

$$A(q^{-1})\Delta(q^{-1})y(t+j) = B(q^{-1})\Delta u(t+j-1) + \xi(t) \quad [3.43]$$

On the basis of equation [3.41], we can write that:

$$\begin{aligned} y(t+j) &= q^{-j}F_j(q^{-1})y(t+j) + q^{-j}H_j(q^{-1})\Delta u(t+j-1) + \\ &\quad + G_j(q^{-1})\Delta u(t+j-1) + J_j(q^{-1})\xi(t+j) \\ \Leftrightarrow \left[1-q^{-j}F_j(q^{-1})\right]y(t+j) &= \left[G_j(q^{-1}) + q^{-j}H_j(q^{-1})\right]\Delta u(t+j-1) + \\ &\quad + J_j(q^{-1})\xi(t+j) \end{aligned} \quad [3.44]$$

From equation [3.43], if we multiply by $J_j(q^{-1})$, we get:

$$A(q^{-1})J_j(q^{-1})\Delta(q^{-1})y(t+j) = B(q^{-1})J_j(q^{-1})\Delta u(t+j-1) + \xi(t)J_j(q^{-1}) \quad [3.45]$$

Thus, from equations [3.44] and [3.45], we obtain the following two equations:

$$\begin{aligned} &A(q^{-1})J_j(q^{-1})\Delta(q^{-1}) = 1 - q^{-j}F_j(q^{-1}) \\ &\Leftrightarrow A(q^{-1})J_j(q^{-1})\Delta(q^{-1}) + q^{-j}F_j(q^{-1}) = 1 \end{aligned} \quad [3.46]$$

$$B(q^{-1})J_j(q^{-1}) = G_j(q^{-1}) + q^{-j}H_j(q^{-1}) \quad [3.47]$$

These equations are also known as Diophantine equations.

If we hypothesize that the best prediction of the term linked to the disturbances is its average (here zero, in the case of the centered white noise), the optimal predictor is defined uniquely as soon as the polynomials are known:

$$y(t+j) = F_j(q^{-1})y(t) + H_j(q^{-1})\Delta u(t-1) + G_j(q^{-1})\Delta u(t+j-1) \quad [3.48]$$

with:

$$\begin{cases} \deg\left[J_j(q^{-1})\right] = \deg\left[G_j(q^{-1})\right] = j-1 \\ \deg\left[F_j(q^{-1})\right] = \deg\left[A(q^{-1})\right] \\ \deg\left[H_j(q^{-1})\right] = \deg\left[B(q^{-1})\right] - 1 \end{cases} \quad [3.49]$$

In order to construct the control, GPC needs to put a minimization criterion in place.

To this end, GPC describes *J*, which is a quadratic minimization criterion at the finite horizon. *J* is defined such that:

$$J = \sum_{j=N_1}^{N_2} \left(y(t+j) - w(t+j)\right)^2 + \lambda \sum_{j=1}^{N_u} \Delta u(t+j-1)^2 \quad [3.50]$$

where $w(t+j)$ is the set point to be applied at time $(t+j)$; $y(t+j)$ is the predicted output at time $(t+j)$; $\Delta u(t+j-1)$ is the increment of the control at time $(t+j-1)$; N_1 is the minimum prediction horizon on the output; N_2 is the maximum prediction horizon on the output; N_u is the prediction horizon on the control; and λ is the weighting coefficient on the control.

By minimizing the criterion *J*, we get the sequence of future commands, of which only the first will actually be applied. The coefficient λ gives more or less weight to the control in relation to the output, ensuring convergence when the initial system exhibits a danger of instability.

3.1.4.2. *Matrix representation and calculation of the optimal control for GPC*

Equation [3.45] can be represented in matrix form between horizons N_1 and N_2, such that:

$$y = if(q^{-1}).y(t) + G.\tilde{u} + ih(q^{-1}).\Delta u(t-1) \quad [3.51]$$

where:

$$if(q^{-1}) = \left[F_{N_1}(q^{-1}) \quad \ldots \quad F_{N2}(q^{-1})\right]^T$$

$$ih(q^{-1}) = \left[H_{N_1}(q^{-1}) \quad ... \quad H_{N2}(q^{-1}) \right]^T$$

$$\tilde{u} = \left[\Delta u(t) \quad ... \quad \Delta u(t+N_u-1) \right]^T$$

$$G = \begin{bmatrix} g_{N_1}^{N_1} & g_{N_1-1}^{N_1} & \cdots & \cdots \\ g_{N_1+1}^{N_1+1} & g_{N_1}^{N_1+1} & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots \\ g_{N_2}^{N_2} & g_{N_2-1}^{N_2} & & g_{N_2-N_u+1}^{N_2} \end{bmatrix} \qquad [3.52]$$

G is the matrix of coefficients $\left\{ g_i^j \right\}$ of the polynomials G_j; $\left\{ g_i^j \right\}$ corresponds to the values of the coefficients $\left\{ g_i \right\}$ of the model's indicial response.

From equations [3.51] and [3.52], we can rewrite the quadratic criterion J such that:

$$\begin{aligned} J = & \left[if(q^{-1}).y(t) + G.\tilde{u} + ih(q^{-1}).\Delta u(t-1) - w \right]^T \\ & \left[if(q^{-1}).y(t) + G.\tilde{u} + ih(q^{-1}).\Delta u(t-1) - w \right] + \lambda.\tilde{u}^T\tilde{u} \end{aligned} \qquad [3.53]$$

where:

$$w = \left[w(t+N_1) \quad ... \quad w(t+N_2) \right]^T \qquad [3.54]$$

The optimal control is obtained by analytical minimization of the criterion in its matrix form:

$$\frac{\partial J}{\partial u} = 0 \qquad [3.55]$$

However, we have:

$$\frac{\partial}{\partial u}\left[if(q^{-1}).y(t) + G.\tilde{u} + ih(q^{-1}).\Delta u(t-1) - w \right]^T = G^T \qquad [3.56]$$

Hence:

$$\begin{aligned} & \frac{\partial J}{\partial u} = 2.G^T.\left[if(q^{-1}).y(t) + G.\tilde{u} + ih(q^{-1}).\Delta u(t-1) - w \right] + 2.\lambda.\tilde{u} = 0 \\ & \Leftrightarrow \tilde{u} = \left[G^T.G + \lambda.I_{N_u} \right]^{-1}.G^T\left[w - if(q^{-1}).y(t) - ih(q^{-1}).\Delta u(t-1) \right] \end{aligned} \qquad [3.57]$$

Thus we obtain the optimal control:

$$\tilde{u} = M.\left[w - if(q^{-1}).y(t) - ih(q^{-1}).\Delta u(t-1) \right] \quad [3.58]$$

with:

$$M = Q.G^T \text{ of dimension } N_u \times (N_2 - N_1 + 1)$$

$$Q = \left[G^T.G + \lambda.I_{N_u} \right]^{-1} \text{ of dimension } N_u \times N_u \quad [3.59]$$

Finally, only the first value in the sequence [3.58] is applied to the system, in accordance with the receding horizon strategy, whereby:

$$\Delta u_{opt}(t) = m_1^T.\left[w - if(q^{-1}).y(t) - ih(q^{-1}).\Delta u(t-1) \right] \quad [3.60]$$

with m_1 being the first row of the matrix *M*.

3.1.5. *The RST controller*

The RST polynomial form of a controller is a highly advantageous exploitation of the possibilities of digital control [BOR 93; LON 06].

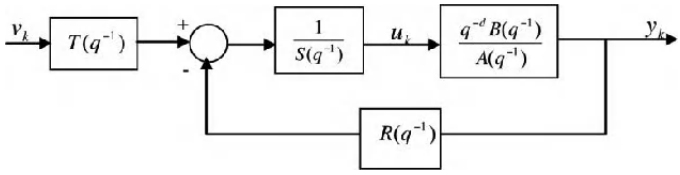


Figure 3.7. *Structure of the RST controller*

The controller possesses two degrees of freedom: the first classically defined on the signal of error between the set point and the measurement, and the second facilitating pursuit by a reference trajectory. The structure of the controller also makes it possible to impose poles and certain zero values in a closed-loop system.

3.1.5.1. *Principle*

The procedure is given by:

$$F(q^{-1}) = \left. \frac{q^{-d}B(q^{-1})}{A(q^{-1})} \right|_{\substack{\deg(B)=m \\ \deg(A)=n}} \quad [3.61]$$

and the polynomials of the RST controller by:

$$\begin{aligned} T(q^{-1}) &= t_0 + t_1 q^{-1} + t_2 q^{-2} + ... \\ R(q^{-1}) &= r_0 + r_1 q^{-1} + r_2 q^{-2} + ... \\ S(q^{-1}) &= s_0 + s_1 q^{-1} + s_2 q^{-2} + ... \end{aligned} \quad [3.62]$$

The open-loop system is written:

$$G(q^{-1}) = \frac{q^{-d}.T(q^{-1}).B(q^{-1}).R(q^{-1})}{S(q^{-1}).A(q^{-1})} \quad [3.63]$$

and the closed-loop system becomes:

$$\begin{aligned} H(q^{-1}) &= \frac{T(q^{-1}).F(q^{-1})/S(q^{-1})}{1+R(q^{-1}).T(q^{-1}).F(q^{-1})/S(q^{-1})} \\ \Leftrightarrow H(q^{-1}) &= \frac{q^{-d}.T(q^{-1}).B(q^{-1})}{S(q^{-1}).A(q^{-1}) + q^{-d}.R(q^{-1}).T(q^{-1}).B(q^{-1})} \end{aligned} \quad [3.64]$$

Thus, by using the notation P to denote the polynomial of the denominator:

$$P(q^{-1}) = S(q^{-1}).A(q^{-1}) + q^{-d}.R(q^{-1}).T(q^{-1}).B(q^{-1}) \quad [3.65]$$

Hence:

$$H(q^{-1}) = \frac{q^{-d}.T(q^{-1}).B(q^{-1})}{P(q^{-1})} \quad [3.66]$$

3.1.5.2 *Placement of pole and next steps*

The choice of P enables us to impose the poles of the system. In order to do so, we need only solve the equation:

$$P(q^{-1}) = 1 + p_1 q^{-1} + p_2 q^{-2} + ... = S(q^{-1}).A(q^{-1}) + q^{-d}.R(q^{-1}).T(q^{-1}).B(q^{-1}) \quad [3.67]$$

The solution of this equation requires the degrees of the polynomials to be compatible. We call this equation “Bézout’s identity” or a “Diophantine equation”.

Another possibility opened up by the polynomial structure of the RST corrector is the performance of pursuance. The idea is to impose a reference trajectory

corresponding to a reference model by synthesizing an RST controller. This involves solving:

$$F_m(q^{-1}) = \frac{B(q^{-1})T(q^{-1})}{A(q^{-1})R(q^{-1}) + B(q^{-1})S(q^{-1})} = \frac{B_m(q^{-1})}{A_m(q^{-1})} \quad [3.68]$$

3.1.5.3. *Applications: RST representation of generalized predictive control*

The RST corrector has an open-ended structure which is very effective for representing complex control systems. For instance, GPC can be transcribed (and therefore implemented!) with this polynomial structure.

Based on equation [3.60], we can write that:

$$\Delta u_{opt}(t)\left[1 + m_1{}^T.ih(q^{-1}).q^{-1}\right] = m_1{}^T.w - m_1{}^T.if(q^{-1}).y(t) \quad [3.69]$$

where:

$$w = \left[w(t+N_1) \quad \ldots \quad w(t+N_2)\right]^T = \left[q^{N_1} \quad \ldots \quad q^{N_2}\right]^T w(t) \quad [3.70]$$

Thus:

$$\Delta u_{opt}(t)\left[1 + m_1{}^T.ih(q^{-1}).q^{-1}\right] = m_1{}^T.\left[q^{N_1} \quad \ldots \quad q^{N_2}\right]^T w(t) - m_1{}^T.if(q^{-1}).y(t) \quad [3.71]$$

The RST formulation of the GPC controller is therefore defined in Figure 3.8.

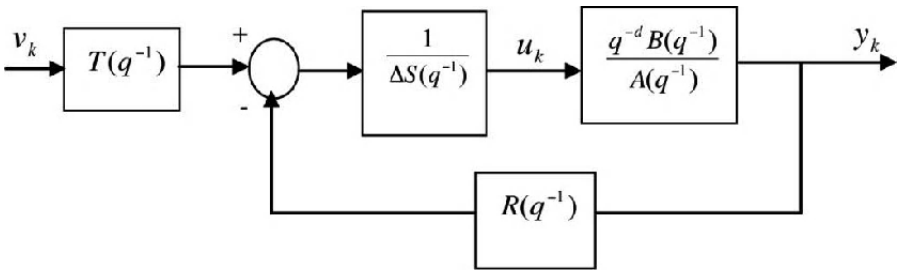


Figure 3.8. *RST structure of the GPC controller*

From this structure, we deduce that:

$$S(q^{-1}).\Delta(q^{-1}).u(t) = T(q^{-1}).w(t) - R(q^{-1}).y(t) \qquad [3.72]$$

By identification between equations [3.71] and [3.72], we obtain the expressions for the polynomials of the RST structure:

$$\begin{cases} S(q^{-1}) = 1 + m_1^T .ih(q^{-1}).q^{-1} \\ R(q^{-1}) = m_1^T .if(q^{-1}) \\ T(q) = m_1^T .\left[q^{N_1} \quad \ldots \quad q^{N_2} \right]^T \end{cases} \qquad [3.73]$$

NOTE.– $T(q)$ encapsulates the non-causal structure inherent in predictive control.

3.1.6. *Implementation of the advance algorithms on a programmable logic controller: results*

The aim of this section is to lay out the concrete results of corrections of various systems using algorithms which were implemented in the programmable logic controller (PLC) premium produced by Schneider.

3.1.6.1. *The Smith predictor*

The Smith predictor is a highly advantageous controller if the dead time of the system to be corrected is non-negligible. The implementation on an industrial PLC ensures the correction of stable systems (first- and second-order) or integrator systems (unstable).

3.1.6.1.1. Correction of a first-order stable system with delay

Consider a first-order system of the form:

$$G(p) = \frac{2.0e^{-\tau p}}{1+10p} \qquad [3.74]$$

The measurements illustrated in the figures below were taken for increasing delays (from 1 second to 12 seconds) and with a sampling time of 500 ms:

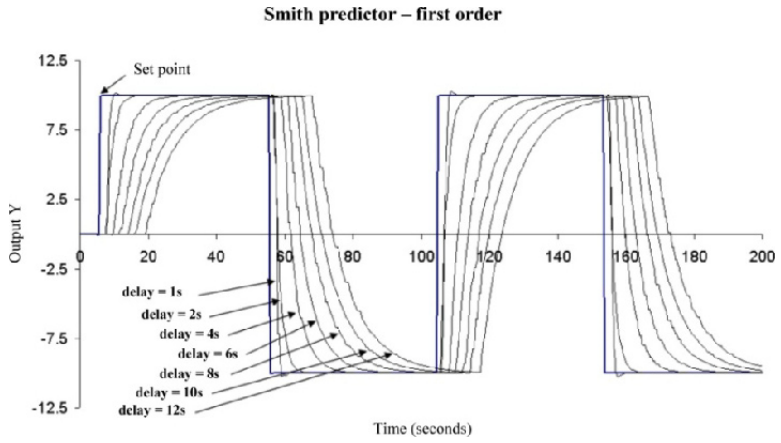


Figure 3.9. *Correction with a Smith predictor of a first-order stable system with delay – reading 1 – system output (measured)*

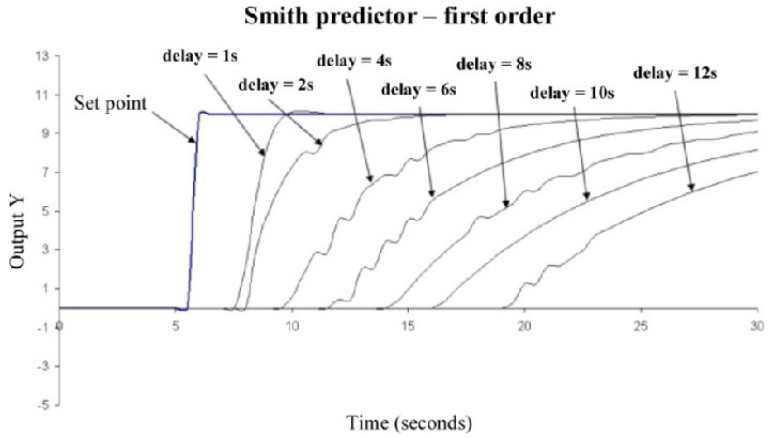


Figure 3.10. *Correction with a Smith predictor of a first-order stable system with delay – reading 2 – system output (measured)*

The curves shown in Figures 3.9 and 3.10 highlight the specificity of the Smith predictor: it is able to cancel out the typical undesirable effects of delay. The system's responses to changing reference points demonstrate the precision of the response and near non-existence of overflow.

The curves below illustrate the control strategies implemented for the correction of first-order systems with variable delays.

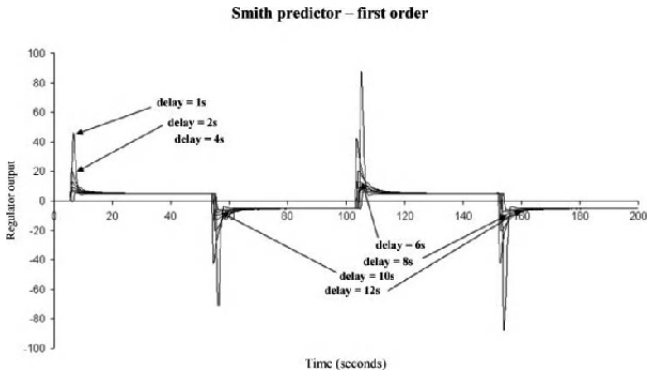


Figure 3.11. *Correction with a Smith predictor of a first-order stable system with delay – reading 1 – controller output (control)*

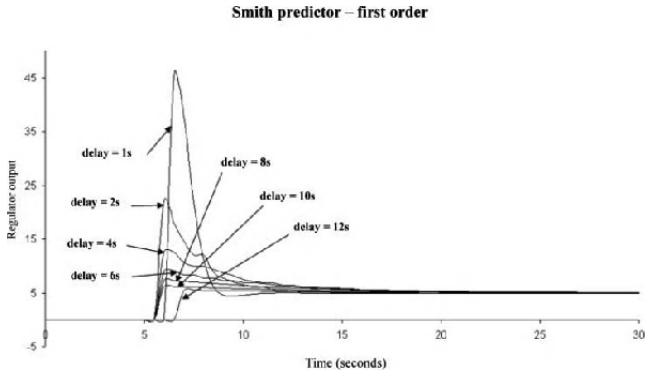


Figure 3.12. *Correction with a Smith predictor of a first-order stable system with delay – reading 2 – controller output (control)*

The readings from the successive tests for different lengths of delay are clear: the delay is anticipated more actively the longer it is. The control strategies take account of the delay in the system's reaction by putting in place what is called an "anticipatory" command. This function is to be found in the very structure of the controller with the dead time compensator (DTC).

3.1.6.1.2. Correction of a second-order stable system with delay

Consider a second-order system of the form:

$$G(p) = \frac{2.0e^{-\tau p}}{(1+2p)^2} \qquad [3.75]$$

The readings shown in the figures below were taken for increasing delays (from 1 second to 10 seconds) and with a sampling time of 500 ms.

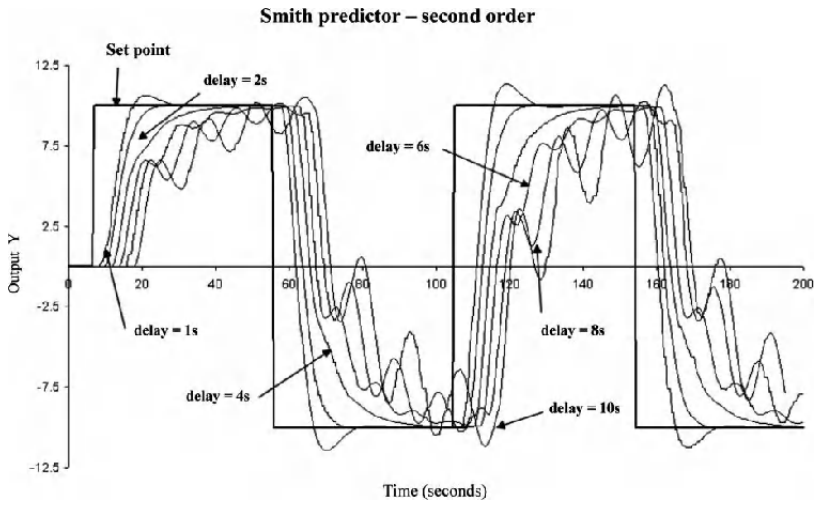


Figure 3.13. *Correction with a Smith predictor of a second-order stable system with delay – reading 1 – system output (measured)*

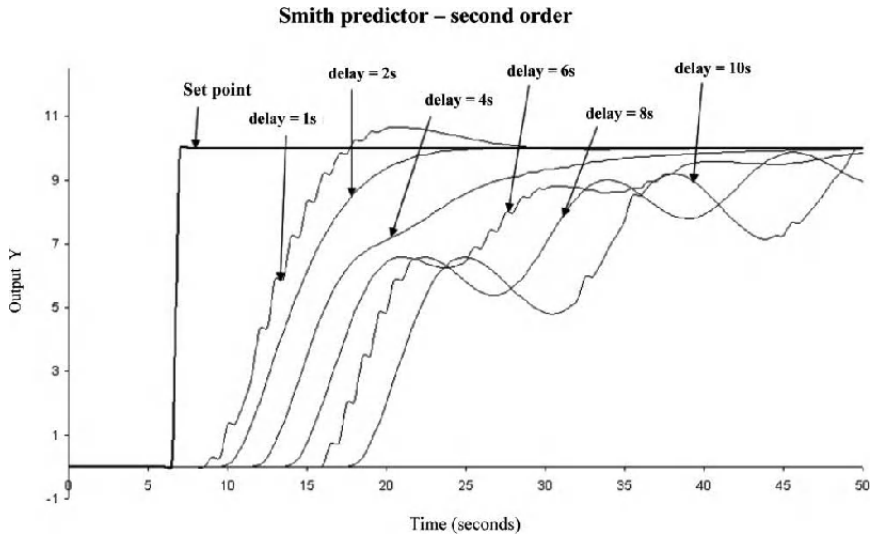


Figure 3.14. *Correction with a Smith predictor of a second-order stable system with delay – reading 2 – system output (measured)*

Just as for the case of a first-order system, the Smith predictor for a second-order system is able to decrease (or even eliminate) the undesirable effects of delay. It appears, however, that this correction is noticeably less effective in relation to the desired results. It can very clearly be seen that the corrected system converges, but with slower stabilization with some overflows.

The commands applied to the system are illustrated in the following figures.

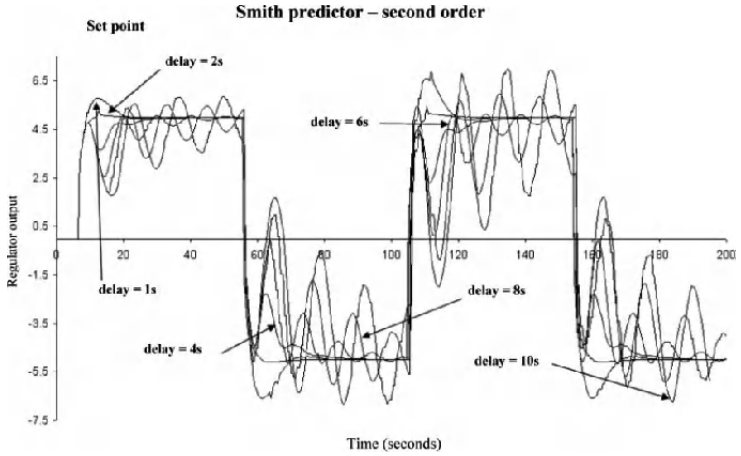


Figure 3.15. *Correction with a Smith predictor of a second-order stable system with delay – reading 2 – controller output (control)*

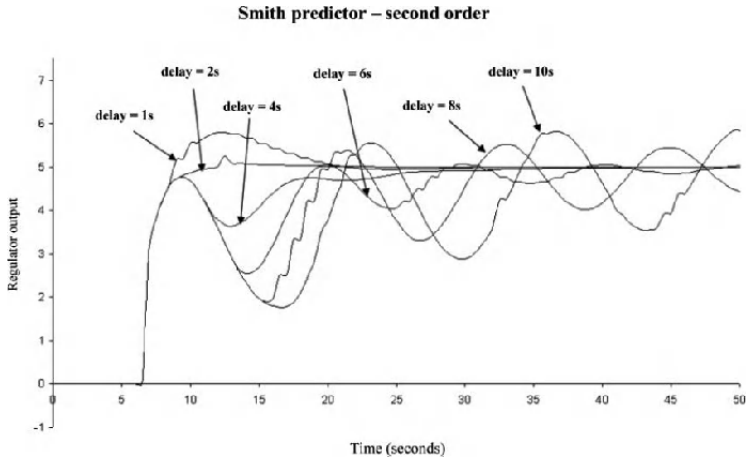


Figure 3.16. *Correction with a Smith predictor of a second-order stable system with delay – reading 1 – controller output (control)*

The anticipation function is present but performs less well. The DTC acts to limit the effects of the delay, but apparently does not fulfill its role to 100% capacity.

This example is interesting because it highlights the difficulties encountered with the practical application of the Smith predictor. This can be attributed to the very nature of the PLC and the discrete nature of the correction. The industrial PLC was not strictly built with the purpose of delivering perfect sampling. One must always take account of the size of the application. Indeed, the execution of the cycle adds a degree of uncertainty to the sampling. Finally, the discrete nature undeniably introduces a relative "instability" when dealing with fast systems that require extensive sampling.

3.1.6.1.3. Correction of an integrator system (unstable) with delay

Consider an integrator system of the form:

$$G(p) = \frac{0.2e^{-\tau p}}{p} \qquad [3.76]$$

The readings shown in the figures below were taken for increasing delays (from 1 second to 10 seconds) and with a sampling period of 500ms and a time constant in a closed loop of 5 seconds.

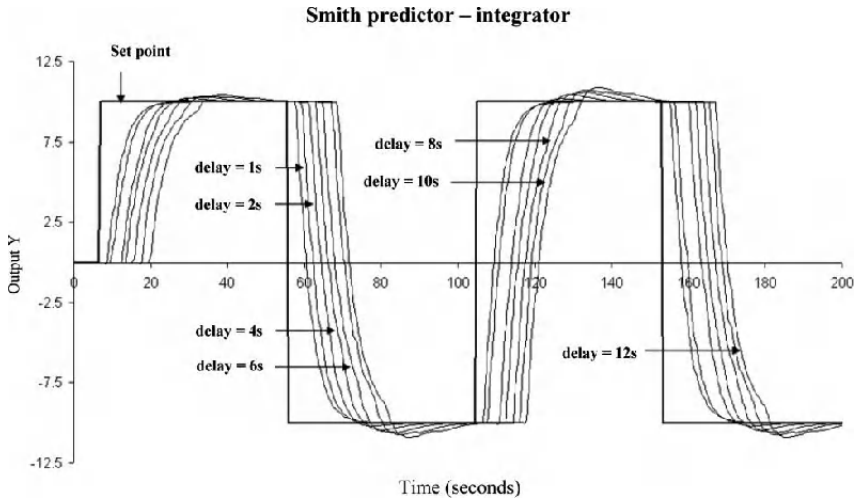


Figure 3.17. *Correction with a Smith predictor of an integrator system with delay – reading 1 – system output (measured)*

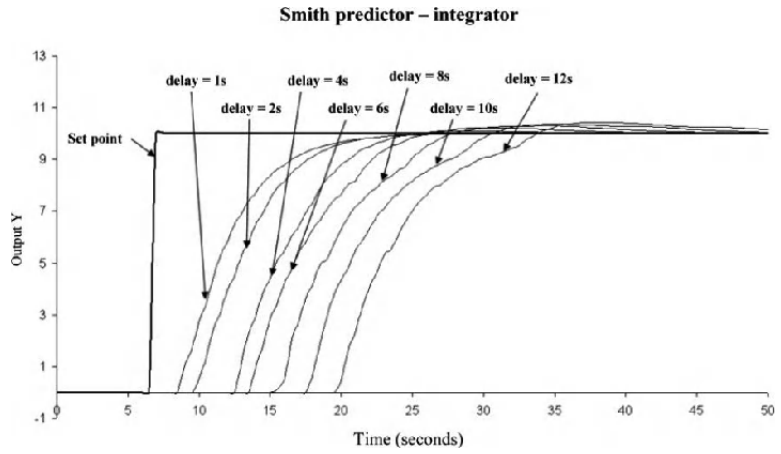


Figure 3.18. *Correction with a Smith predictor of an integrator system with delay – reading 2 – system output (measured)*

This example of a correction with the modified Smith predictor [MAT 96] demonstrates the potency of this method. In spite of slight overflows, due mainly to the problems with discrete executions (mentioned above), the responses of the systems, given differing reference points, are conclusive. The looped systems are stable and accurate.

The corresponding controls help to appreciate these results more fully.

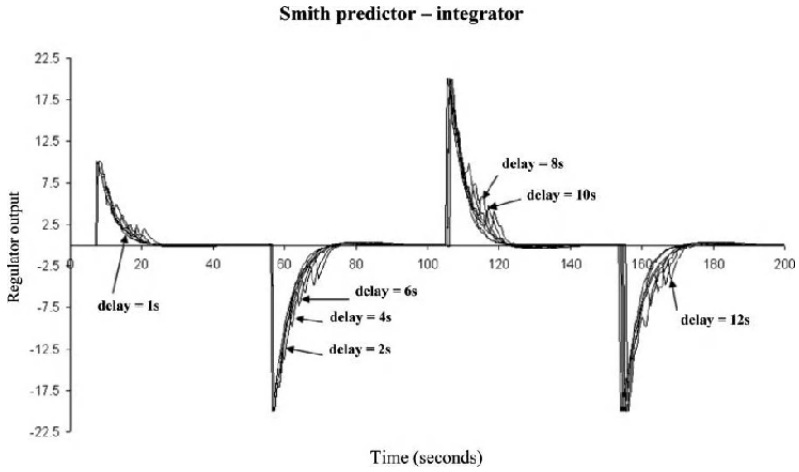


Figure 3.19. *Correction with a Smith predictor of an integrator system with delay – reading 1 – controller output (control)*

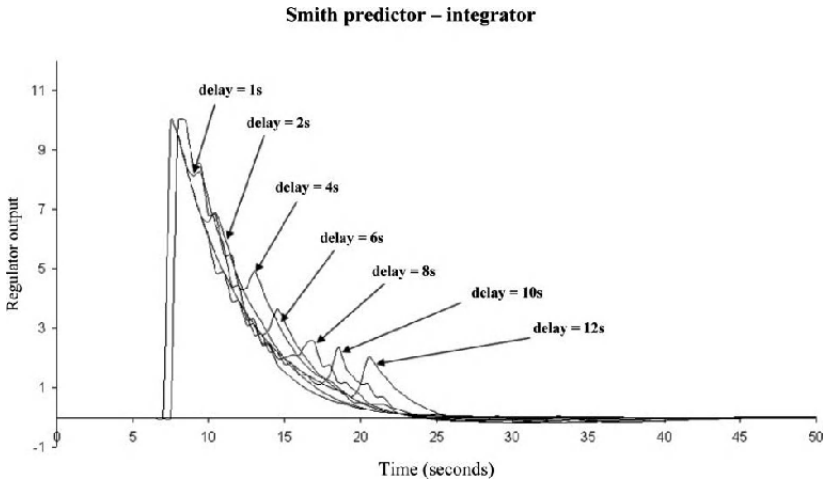


Figure 3.20. *Correction with a Smith predictor of an integrator system with delay – reading 2 – controller output (control)*

The Smith predictor for an unstable system produces an adapted correction. The results of the last curves (see Figures 3.19 and 3.20) demonstrate that the commands applied anticipate the undesirable effects of delay: the shapes of the derivatives of the commands reflect this behavior. Finally, by regulating the corrector, we can specify the output dynamic of the closed-loop system. Indeed, as we can see, the system takes around fifteen seconds to reach the set point (three times the specified time constant).

3.1.6.1.4. Correction of a slow first-order stable system with heavy delay

This example enables us to demonstrate the efficiency of the Smith predictor in extreme cases of severe delays. Consider a first-order system described by:

$$G(p) = \frac{2.0e^{-1200p}}{1+3600p} \qquad [3.77]$$

The reading of the output from the system and from the controller (1-second sampling frequency) is shown in Figure 3.21.

The output of the closed-loop system is accurate and converges toward the desired set point with no apparent overflow. In spite of a long delay of nearly twenty minutes, the controller is capable of producing a command which adapts to “cancel out” the effects of the delay.

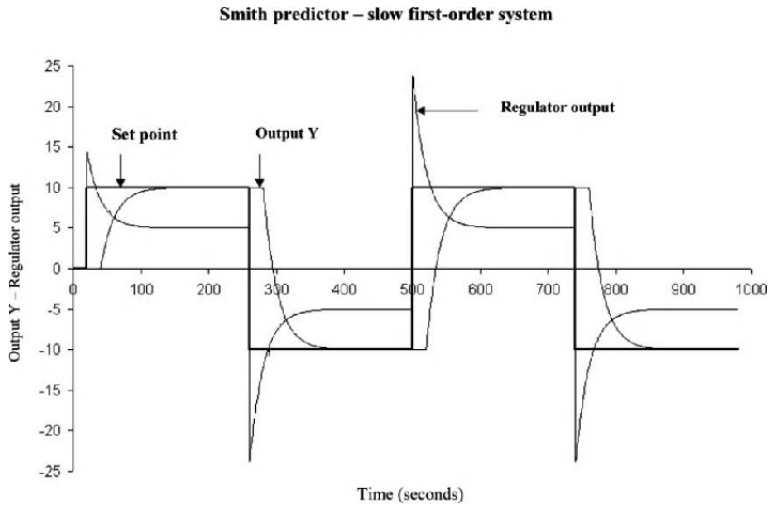


Figure 3.21. *Correction with a Smith predictor of a slow first-order stable system with heavy delay – controller output and system output*

It is also noteworthy that, as the sampling rate is very high in view of the nature of the system, there appear to be none of the harmful effects noted in the previous examples (slight overflows, noticeably oscillating commands, etc.).

The Smith predictor has a simple structure to compensate for delay. Its implementation for stable systems (first- and second-order) and unstable systems is entirely apt for an industrial PLC.

3.1.6.2. *Generalized predictive functional control*

Generalized PFC has the advantage of being able to construct a corrector on the basis of the indicial response of the system needing to be corrected. The necessary condition is that the system must be self-stable. From a basic reading, it is possible, by a simple calculation (performed in the PLC object), to implement a controller by way of the convolution representation. Then, we need only define a consistent horizon and reference trajectory in order to obtain good behavior from a closed-loop system.

3.1.6.2.1. Correction of a third-order system

Consider a third-order system described by:

$$G(p) = \frac{2.0}{24p^3 + 26p^2 + 9p + 1} \qquad [3.78]$$

When putting generalized PFC into practice, it is essential to model the system with an MA representation. The algorithm of this controller enables us to circumvent the calculation stage: as an input parameter, we feed in the values of the sampled y_i at a level applied at the input to the system. From the reading (see Figure 3.22), we find a sampling frequency of 2.5 seconds.

$y_1 = 0$	$y_6 = 1.5684$	$y_{11} = 1.9734$
$y_2 = 0.1126$	$y_7 = 1.7428$	$y_{12} = 1.9853$
$y_3 = 0.4873$	$y_8 = 1.8507$	$y_{13} = 1.992$
$y_4 = 0.9298$	$y_9 = 1.9149$	$y_{14} = 1.9956$
$y_5 = 1.3018$	$y_{10} = 1.9522$	$y_{15} = 1.9976$

Table 3.2. *Generalized PFC – third order – readings of y_i values*

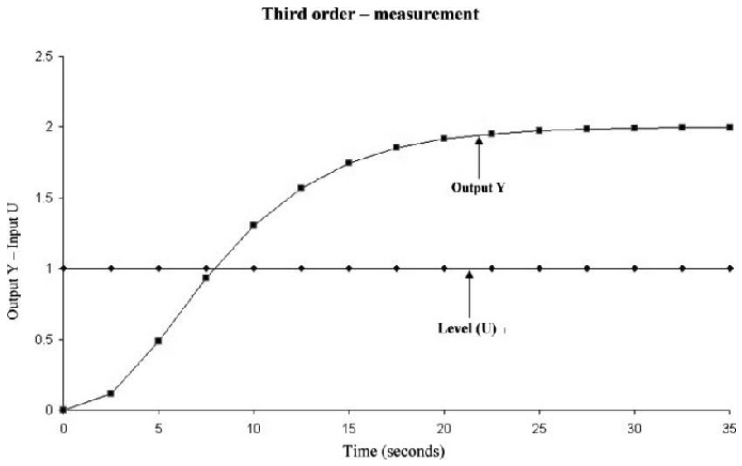


Figure 3.22. *Generalized PFC – third order – indicial response of the system for determination of the y_i values*

With a sampling frequency of 2.5 seconds, a horizon of 5 seconds and a time constant of 20 seconds for the reference trajectory, we obtain the curves shown in Figure 3.23.

The controller output is calculated on the basis of the desired performances. We can see that the system reaches the set point in around 60 seconds (three times the time constant of the reference trajectory). The looped system is accurate and presents no overflow.

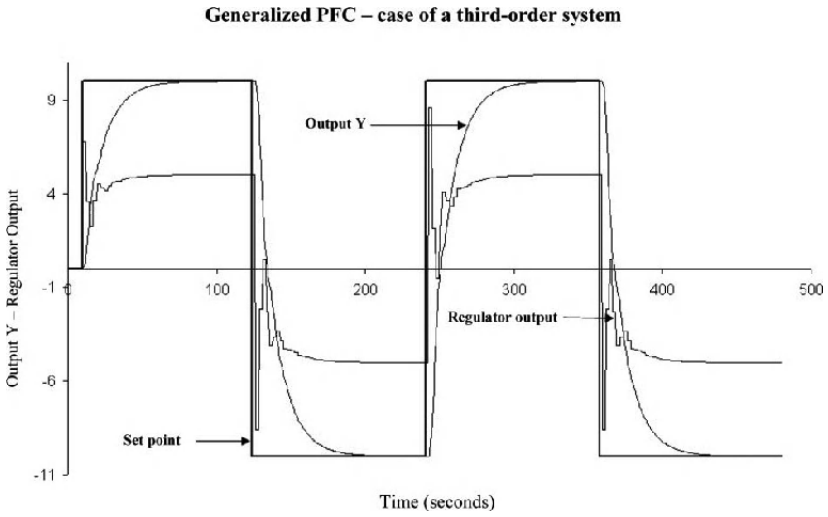


Figure 3.23. *Correction with generalized PFC – third order – Controller output and system output*

3.1.6.2.2. Correction of a fourth-order slow system

Consider a fourth-order system described by:

$$G(p) = \frac{2.0}{15000000\,p^4 + 6250000\,p^3 + 87500\,p^2 + 500\,p + 1} \qquad [3.79]$$

The measured values of the y_i sampled after an indicial response (75-second sampling frequency) is described in Table 3.3.

$y_1 = 0$	$y_8 = 1.2089$	$y_{15} = 1.9201$
$y_2 = 0.0163$	$y_9 = 1.3988$	$y_{16} = 1.9443$
$y_3 = 0.1042$	$y_{10} = 1.5538$	$y_{17} = 1.961$
$y_4 = 0.2723$	$y_{11} = 1.6755$	$y_{18} = 1.9724$
$y_5 = 0.496$	$y_{12} = 1.7679$	$y_{19} = 1.9802$
$Y_6 = 0.7434$	$y_{13} = 1.8362$	$Y_{20} = 1.9856$
$Y_7 = 0.9872$	$y_{14} = 1.8854$	

Table 3.3. *Generalized PFC – fourth order – reading of y_i values*

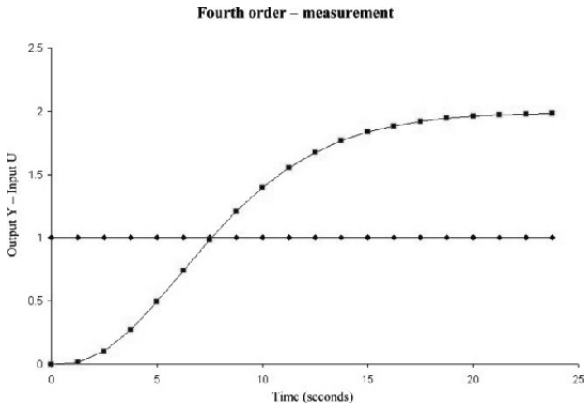


Figure 3.24. *Generalized PFC – fourth order – indicial response of the system for determination of the y_i values*

With a 75-second sampling frequency, a horizon of 150 seconds and a time constant of 10 minutes for the reference trajectory, we obtain the curves shown in Figure 3.25.

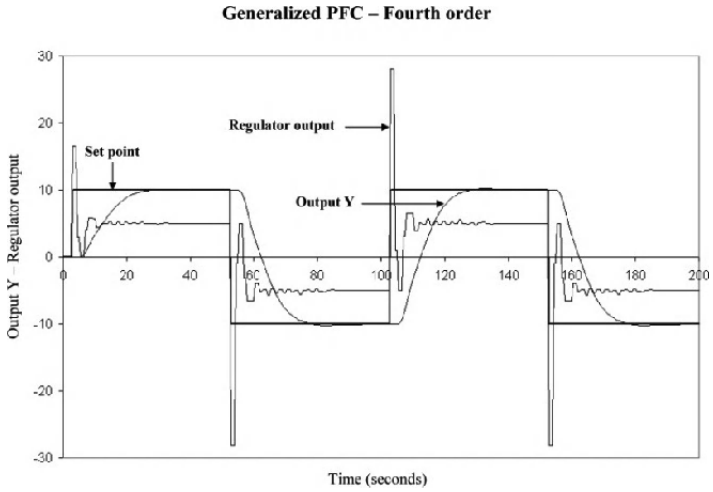


Figure 3.25. *Correction with generalized PFC – fourth order – Controller output and system output*

The output of the looped system is stable and "catches up to" the instruction in around 25 minutes. There is little overflow and the controller output converges with no problems toward its stabilization value.

Generalized PFC is a concrete example of the potency of a predictive method for a wide range of self-stable systems which requires less investment of effort in terms of implementation. The industrial PLC is entirely appropriate for this application.

3.1.6.3. *Generalized predictive control*

The results presented in this section relate to the correction of the same systems as discussed in the section on generalized PFC. In practice, GPC is advantageous in that it is possible to construct a corrector in RST form.

3.1.6.3.1. Correction of a third-order system

Consider the system [3.80]. The discrete representation of this model is also:

$$G(z) = \left. \frac{0.1126z^{-1} + 0.2334z^{-2} + 0.02904z^{-3}}{1 - 1.256z^{-1} + 0.5105z^{-2} - 0.06665z^{-3}} \right|_{T_e = 2.5s} \qquad [3.80]$$

The application of the control algorithms, for the implementation of GPC for three different horizons, is summarized in Table 3.4.

Case 1	**Case 2**	**Case 3**
N_1=1	N_1=1	N_1=1
N_2=3	N_2=5	N_2=8
N_u=1	N_u=1	N_u=1
λ=1.1147	λ=5.2659	λ=15.3747
T_e system = 10ms	T_e system = 10 ms	T_e system = 10 ms
T_e controller = 2.5s	T_e controller = 2.5 s	T_e controller = 2.5 s
RST polynomial = R=[2.5456 -2.7672 1.0364 -0.1286] S=[1.0000 -0.9440 0.4280 -0.4840] T=[0.0505 0.2186 0.4170] $T_{advised}$=0.6862	RST polynomial = R=[1.8351 -2.1483 0.8373 -0.1064] S=[1.0000 -0.9536 0.3631 -0.4094] T=[0.0107 0.0463 0.0883 0.1236 0.1488] $T_{advised}$=0.4177	RST polynomial = R=[1.5548 -1.8874 0.7515 -0.0968] S=[1.0000 -0.9578 0.3347 -0.3769] T=[0.0037 0.0159 0.0302 0.0423 0.0510 0.0566 0.0601 0.0622] $T_{advised}$=0.3220

Table 3.4. *GPC – application for three different horizons*

NOTE.– For the determination of the polynomial T, see [CAM 93].

The readings presented below demonstrate the behaviors of the output from the system and the controller.

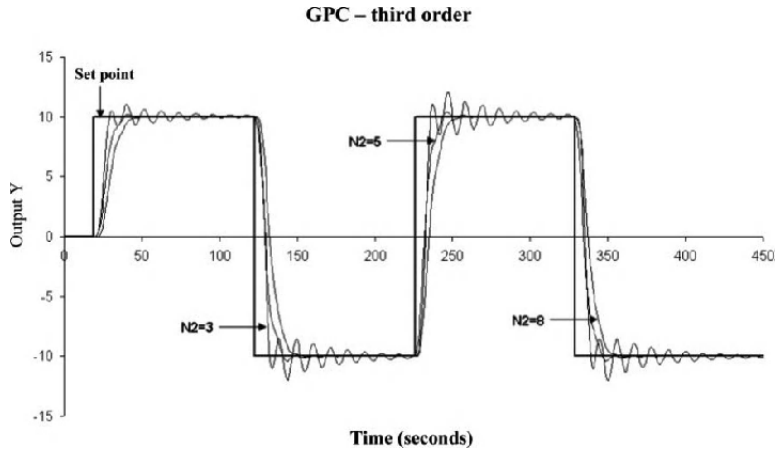


Figure 3.26. *Correction with GPC – third order – system output*

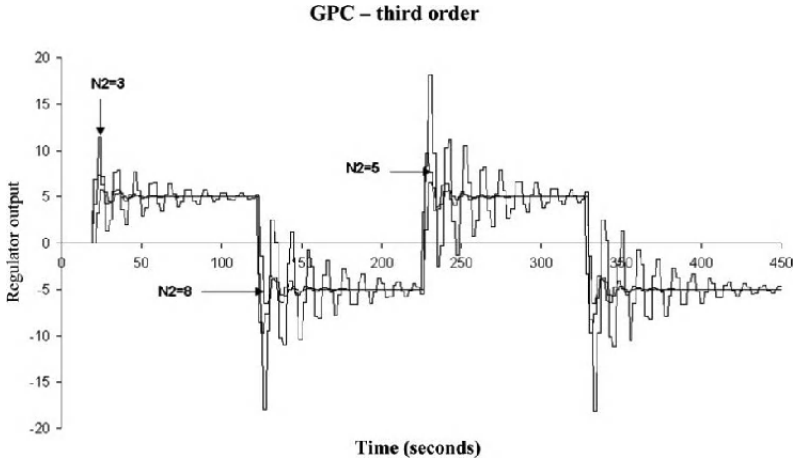


Figure 3.27. *Correction with GPC – third order – controller output*

The output of the looped system is dependent upon the regulating parameters of the GPC. The above curves illustrate the impact of the receding prediction horizon on the corrected system: the controller output converges more quickly toward an equilibrium state and the looped output becomes more stable as N_2 increases. This is not to say that a large value of this parameter guarantees good performances, but these curves highlight the predictive nature of GPC by the definition of the horizon of the optimal predictor. Finally, these results give a conclusive initial view of the practical implementation of GPC on the industrial PLC.

3.1.6.3.2. Correction of a fourth-order system

Consider the system [3.81]. The discrete representation of this model is also:

$$G(z) = \left. \frac{0.01627z^{-1} + 0.05466z^{-2} + 0.01162z^{-3} + 1.805e-006z^{-4}}{1 - 2.044z^{-1} + 1.428z^{-2} - 0.3423z^{-3} + 2.681e-014z^{-4}} \right|_{T_e = 75s} \quad [3.81]$$

The parameters for two different horizons which have been used for the implementation of GPC are summarized in Table 3.5.

Case 1	**Case 2**
$N_1 = 1$	$N_1 = 1$
$N_2 = 8$	$N_2 = 20$
$N_u = 1$	$N_u = 1$
$\lambda = 5{,}2626$	$\lambda = 47{,}0064$
T_e system = 30 s	T_e system = 30 s
T_e controller = 75 s	T_e controller = 75 s
RST polynomial = R = [8.2886 -16.0467 10.7251 -2.4709] S = [1.0000 -1.0000 0.0839 0.3804 -0.4643] T = [], $T_{advised} = 0.4961$	RST polynomial = R = [6.4684 -13.0737 9.0482 -2.1500] S = [1.0000 -1.0000 0.0730 0.3406 -0.4136] T = [], $T_{advised} = 0.293$

Table 3.5. *GPC – application for two different horizons*

The following readings evidentiate the behaviors of the system output and controller output.

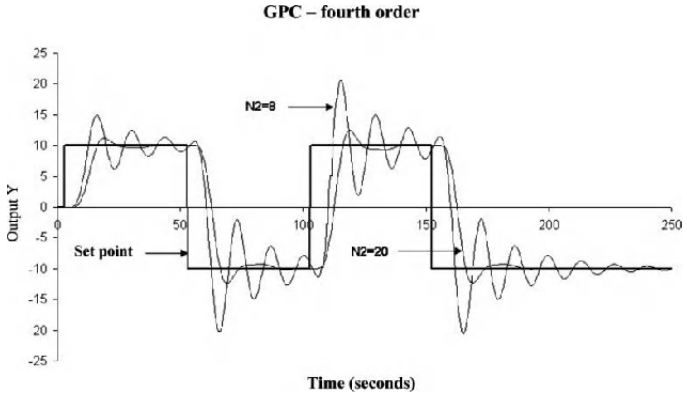


Figure 3.28. *Correction with GPC – fourth order – system output*

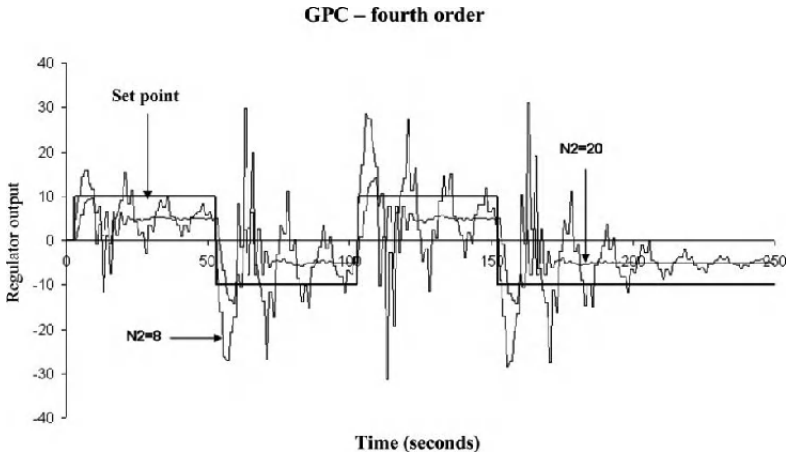


Figure 3.29. *Correction with GPC – fourth order – controller output*

Just as for the previous example, the output of the looped system is dependent on the horizon of the optimal predictor. The controls obtained converge more quickly for a broad horizon.

GPC is a predictive control method which is capable of controlling multiple systems by the use of an optimal predictor subject to a quadratic minimization criterion. Its strength lies in a non-restrictive implementation of any process modeled by an ARMA process. Its limitation is in the matrix calculations likely to be unwieldy for a PLC, and in the resolution of Diophantine equations which may be complex.

3.1.6.4. *The RST corrector*

The RST corrector has a structure that facilitates the implementation of many different controllers. In particular, it enables us to create correctors ensuring the desired performances in a closed-loop system by the placement of poles and the imposition of a pursuit trajectory.

3.1.6.4.1. Pole placement and reference model for pursuit

Consider the process described by the discretized model *H* (example adapted from [LAN 02]):

$$H_{sys}(z) = \left. \frac{-0.009627z^{-1} + 0.01637z^{-2}}{1 - 1.895z^{-1} + 0.8972z^{-2}} \right|_{T_{sys}=0.5s} \qquad [3.82]$$

H_m is the reference model for the dynamics of pursuit:

$$H_m(z) = \left. \frac{0.0927z^{-1} + 0.0687z^{-2}}{1 - 1.2451z^{-1} + 0.4066z^{-2}} \right|_{T_{sys}=4s} \quad [3.83]$$

and the polynomial P from Bézout's identity describes the dynamics of regulation (poles: 0.6225 +/- j0.138):

$$P(z) = 1 - 1.3741z^{-1} + 0.4867z^{-2} \quad [3.84]$$

The RST controller corresponding to these specifications is computed for a 4-second cycle and is such that:

$$\begin{aligned} R(z) &= 3 - 3.94z^{-1} + 1.3141z^{-2} \\ S(z) &= 1 - 0.3742z^{-1} - 0.6258z^{-2} \\ T(z) &= 3.333 - 4.5806z^{-1} + 1.6225z^{-2} \end{aligned} \quad [3.85]$$

We obtain the readings of the outputs from the RST corrector and the closed-loop system H_{sys} (see Figure 3.30).

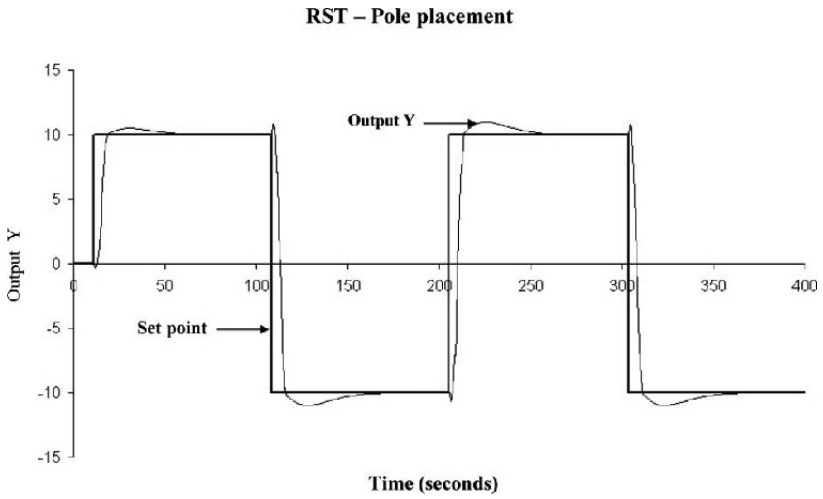


Figure 3.30. *RST correction – pole placement with model of pursuit – system output*

The curves shown in the above figures all appear to indicate that pole placement with the Schneider PLC is effective and functions normally.

In this example, it is interesting to perform the simulation in Matlab®. Thereby, it is possible to compare the behavior of the PLC regulation against the desired results. The curve in Figure 3.32 shows this comparison of the output.

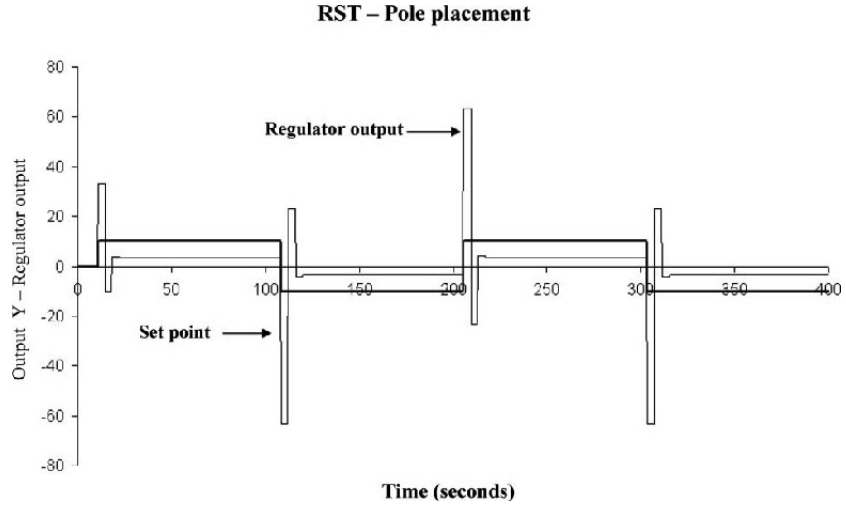


Figure 3.31. *RST correction – pole placement with model of pursuit – controller output*

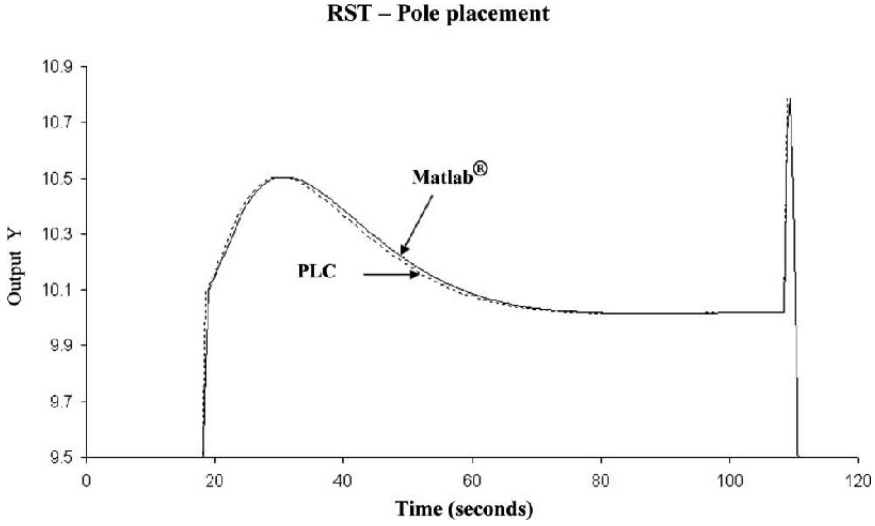


Figure 3.32. *RST correction – pole placement with model of pursuit – comparison of the system output in Matlab® and on the industrial PLC*

As we can see, the results between the Matlab® simulation (*offline*) and the PLC's behavior (*online*) are practically identical. In spite of certain disparities relating to the PLC's actual operation (synchronism), this example of pole placement stands to the credit of the implementation.

The creation of an RST corrector, in the case of pole placement, remains a complex task for an automation engineer. However, it does afford the opportunity to impose certain behaviors with the desired performances.

The PLC lends itself well to this polynomial structure. The RST corrector has the advantage of being able to represent a number of controllers in polynomial form – hence an additional advantage to the PLC in terms of openness.

3.1.6.5. *Review*

Table 3.6 gives a summary of the characteristics of the forms of regulation used:

	Advantages	**Disadvantages**	**Use**
PID	- Simple to implement; - Present in most PLCs; - Solves many control-loop problems.	- False simplicity of use; - Limited for more complex problems.	- On systems with little delay and which can be modeled by a first-order or integrator system.
Smith predictor	- Simple to implement in the PLC; - Simple to use; - Can greatly attenuate (or even eliminate) the undesirable effects of delay; - Robust.	- Not applicable to all systems, even with delay; - Certain troublesome effects for fast systems (problem of sampling).	- Good solution for systems of average speed with significant delays (first order, second order, double pole and integrator systems).
PFC	- Predictive control based on principles that are easy to understand; - Basic library available from Schneider (PCR library): first order, second order, third order, integrator system; - Generalized solution for any self-stable system of any order; - Robust and in some cases able to completely eliminate overflow.	- Tricky to implement in a PLC.	- Good solution to replace PID when problems of overflow and performance are encountered.

Table 3.6. *Overview of the regulations implemented in the PLC*

GPC	- Predictive control that can be generalized to any ARMA system; - Solution that can be modeled with the polynomial representation of the RST corrector.	- Algorithm highly complex to implement in a PLC (limitation in the order of the model to be solved); - Regulation requires a high level of expertise.	- Solution in cases when PFC cannot solve the problem.
RST	- Simple to implement in a PLC; - Open structure that can be used to simulate other control loops; - Solution that can fix the pole placement and pursuit of a reference trajectory.	- Expert level of use.	- Practical solution for the implementation of GPC control; - Solution made for specification of performance by pole placement and pursuit (robustness and dynamics of response).

Table 3.6. *(Continued) Overview of the regulations implemented in the PLC*

3.2. Multi-model control

3.2.1. *Introduction*

Our aim in this section is to examine the stability of multi-models. Many publications on the subject drew inspiration initially from the techniques developed in the linear domain. Indeed, Lyapunov's method and particularly the quadratic method, and the LMI formulation, have been intensively used [BOY 94; CHA 02b; CHA 08d; CHA 10; MA 98; MAR 99; TAN 98]. In that sense, transformation into a Lur'e problem, the techniques of interconnected systems and the properties of M-matrices have been adapted for nominal and uncertain multi-models [BRE 02; CHA 02b].

This section discusses the stability of continuous multi-models with and without uncertainties. Two types of Lyapunov functions are used: quadratic and non-quadratic functions. The section is organized as follows. In section 3.2.2, we present a number of sufficient conditions for the stability of standard multi-models based on the existence of quadratic Lyapunov functions. Next, stability conditions using non-quadratic functions are presented. Finally, robust stability conditions involving two types of uncertainties are put forward.

The topic of synthesis of "controllers" has been under active consideration in recent years, and has birthed a number of publications. These works, most of which were inspired by the control techniques found in the already-existing body of literature, dealt with state feedback control by static and dynamic output feedback

with and without uncertainties [CHA 06b; CHA 10; GUE 04; JAD 99; JOH 98; KRU 08; TAN 01; TAN 98]. Multi-criterion synthesis techniques were also touched upon.

3.2.2. *Stability analysis*

The approach put forward in this section is based primarily on the quadratic Lyapunov functions. It involves looking for a positive-definite symmetrical matrix, i.e. the associated quadratic Lyapunov function, which ensures the asymptotic stability of the multi-models.

Let us consider the continuous multi-model in the form:

$$\dot{x}(t) = \sum_{i=1}^{N} \mu_i \left(z(t)\right)\left(A_i x(t) + B_i u(t)\right) \qquad [3.86]$$

with:

$$\mu_i \left(z(t)\right) \geq 0, \sum_{i=1}^{N} \mu_i \left(z(t)\right) = 1 \qquad [3.87]$$

The open-loop multi-model corresponding to [3.86] is:

$$\dot{x}(t) = \sum_{i=1}^{N} \mu_i \left(z(t)\right) A_i x(t) \qquad [3.88]$$

By deriving the quadratic function $V(x)=x^T(t)Px(t)$ all along the trajectory of the multi-model [3.88], we obtain the following sufficient stability conditions:

$$P > 0, A_i^T P + P A_i < 0, \forall i \in I_N \qquad [3.89]$$

The existence of $P > 0$ depends on two conditions:

– the first relates to the stability of all the local models. It is necessary for each matrix A_i to be a Hurwitz matrix[1];

– the second condition relates to the existence of a Lyapunov function common to the N local models: the matrix $\sum_{i=1}^{N} A_i$ must be a Hurwitz matrix. This condition offers a means of rapid testing. Indeed, if there is an unstable matrix $A_i + A_j$,

1 A matrix whose eigenvalues belong to the left-hand demiplane of the complex plane.

conditions [3.89] cannot be satisfied (i.e. there is no symmetrical matrix $P > 0$ that satisfies conditions [3.89]).

Indeed, consider the following example with two local models defined by [CHA 12a; CHA 12b]: $A_1 = \begin{pmatrix} -1 & 4 \\ 0 & -2 \end{pmatrix}$, $A_2 = \begin{pmatrix} -1 & 0 \\ 4 & -2 \end{pmatrix}$, two asymptotically stable matrices.

The matrix $A_1 + A_2 = \begin{pmatrix} -2 & 4 \\ 4 & -4 \end{pmatrix}$ is unstable (eigenvalues: 1.12 and –7.12) and there is no matrix $P > 0$ that satisfies conditions [3.89].

We now consider the case of uncertain continuous-time multi-models in the form:

$$x(k+1) = \sum_{i=1}^{n} \mu_i(x(k)) A_i x(k) \qquad [3.90]$$

where the matrices $\Delta A_i(t)$ are unknown and time-variable matrices. Different types of uncertainties can be considered: *i*) norm-bounded uncertainties, i.e. $\| \Delta A_i(t)) \| \leq \delta_i, \delta_i > 0$; *ii*) uncertainties structured in the form $\Delta A_i(t) = D_i F_i(t) E_i$, $F_i(t)^T F_i(t) \leq I$ or *iii*) uncertainties in the form of intervals.

The following lemma will be used hereafter [CHA 06a].

LEMMA 3.1.– Consider constant matrices *D* and *E*, an unknown constant matrix *F* of appropriate dimension satisfying the constraint $F^T F \leq I$ where I is the identity matrix. The follow two propositions are equivalent:

i) $DFE + E^T F^T D^T < 0$

ii) $\varepsilon DD^T + \varepsilon^{-1} E^T E < 0$ for any $\varepsilon > 0$.

3.2.2.1. *Norm-bounded uncertainties*

To begin with, let us consider norm-bounded uncertainties:

$$\| \Delta A_i(t)) \| \leq \delta_i, \quad \forall i \in I_N \qquad [3.91]$$

where δ_i is a positive scalar.

The analysis of the asymptotic stability of [3.90] can be obtained directly by considering the following conditions:

$$\left(A_i + \Delta A_i\right)^T P + P\left(A_i + \Delta A_i\right) < 0 \qquad \forall i \in I_N \qquad [3.92]$$

Using Lemma 3.1, with $D = P, F = I_n$, $E = \Delta A_i$, the inequalities [3.92] become:

$$A_i^T P + P A_i + \varepsilon_i^{-1} P^2 + \varepsilon_i \delta_i^2 I_n < 0 \qquad [3.93]$$

with $\varepsilon_i > 0, \forall i \in I_N$.

By applying the Schur complement to [3.93], we obtain the stability conditions:

$$\begin{pmatrix} A_i^T P + P A_i + \varepsilon_i \delta_i^2 I_n & P \\ P^T & -\varepsilon_i \end{pmatrix} < 0 \qquad [3.94]$$

3.2.2.2. *Structured parametric uncertainties*

Parametric uncertainties are assumed to be norm-bounded and structured. They satisfy the following property:

$$\Delta A_i(t) = D_i F_i(t) E_i, \quad F_i(t)^T F_i(t) \le I \qquad [3.95]$$

where D_i, E_i are constant matrices of appropriate dimensions, $F_i(t)$ is an unknown matrix and I is the identity matrix.

The asymptotic stability conditions for [3.90] are:

$$\left(A_i + D_i F_i(t) E_i\right)^T P + P\left(A_i + D_i F_i(t) E_i\right) < 0 \qquad \forall i \in I_N \qquad [3.96]$$

Using Lemma 3.1, we obtain:

$$A_i^T P_j + P_j A_i + \sum_{k=1}^{n} \tau_{ijk}\left(P_j - P_k\right) < 0 \qquad [3.97]$$

Indeed, if there is a matrix $A_i^T P_j + P_j A_i + \sum_{k=1}^{n} \tau_{ijk}\left(P_j - P_k\right) < 0$ and scalars $A_i^T P_j + P_j A_i + \sum_{k=1}^{n} \tau_{ijk}\left(P_j - P_k\right) < 0$ such that the following conditions are satisfied:

$$A_i^T P_j + P_j A_i + \sum_{k=1}^{n} \tau_{ijk} \left(P_j - P_k \right) < 0 \qquad [3.98]$$

then the uncertain multi-model [3.90] is asymptotically stable.

This result is obtained by applying the Schur complement to [3.97]. These often-conservative conditions are relaxed by using other types of Lyapunov functions (partially quadratic, etc.) [CHA 12].

The stability of continuous-time multi-models with and without uncertainties is studied by using quadratic Lyapunov functions. Robust stability conditions have also been put forward with different types of uncertainties. The stability conditions are in LMI form. Other results using so-called poly-quadratic functions can be consulted in [CHA 00; CHA 06b; CHA 12; GUE 04; JAD 99; JOH 98; KRU 08; TAN 01].

3.2.3. *State feedback control*

Consider the nonlinear control law of the form:

$$u(t) = \sum_{i=1}^{N} \mu_i \left(z(t) \right) K_i x(t) \qquad [3.99]$$

A continuous multi-model equipped with this control law can be written:

$$\dot{x}(t) = \sum_{i=1}^{N} \sum_{j=1}^{N} \mu_i \left(z(t) \right) \mu_j \left(z(t) \right) \left(A_i - B_i K_j \right) x(t) \qquad [3.100]$$

The stability conditions for the closed-loop multi-model are written in the same way as those for the open-loop version, by:

$$R_{ii}^T P + P R_{ii} < 0 \qquad \forall i : 1,..,n \qquad [3.101]$$

These conditions have the disadvantage of being nonlinear in form and highly conservative. Various different techniques for relaxation have been introduced in linear form. The following result illustrates this case.

THEOREM 3.1.– If there is a matrix P such that $\forall (i,j) \in I_N^2$, $i < j$:

$$X A_i^T + A_i X + N_i^T B_i^T + B_i N_i < 0, \ \forall i \in I_N \qquad [3.102]$$

$$XA_i^T + A_iX + XA_j^T + A_jX + N_j^T B_i^T + N_i^T B_j^T + B_iN_j + B_jN_i \le 0,\ \forall (i,j) \in I_N^2,\ i < j \quad [3.103]$$

then multi-model [3.92] is asymptotically stable. The stabilization control law is thus defined by:

$$u(t) = \sum_{i=1}^{N} \mu_i(z(t))K_i x(t)\ ,\ K_i = N_i X^{-1} \quad [3.104]$$

so multi-model [3.100], with its stabilizing control law defined by $u(t) = \sum_{i=1}^{N} \mu_i(z(t))K_i x(t)$ is asymptotically stable.

PROOF.– Consider the quadratic Lyapunov function $V(x(t)) = x(t)^T Px(t)$. Its time derivative along the system [3.100] gives us:

$$\dot{V}(x(t)) = x(t)^T \sum_{i=1}^{n}\sum_{j=1}^{n} \mu_i(z(t))\mu_j(z(t))\left(R_{ij}^{\ T}P + PR_{ij}\right)x(t) \quad [3.105]$$

with $G_{ij} = A_i + B_iK_j$. Equation [3.105] can be rewritten as:

$$= x(t)^T \sum_{i=1}^{N} \mu_i^{\,2}(z)\left(G_{ii}^{\ T}P + PG_{ii}\right)x(t) +$$
$$x(t)^T \sum_{i=1}^{N}\sum_{j>i}^{N} \mu_i(z)\mu_j(z)\left(\left(G_{ij} + G_{ji}\right)^T P + P\left(G_{ij} + G_{ji}\right)\right)x(t) \quad [3.106]$$

Thus, the stability conditions are:

$$G_{ii}^T P + PG_{ii} < 0 \quad [3.106a]$$

$$\left(G_{ij} + G_{ji}\right)^T P + P\left(G_{ij} + G_{ji}\right) \le 0 \quad [3.106b]$$

By pre- and post-multiplying these bilinear conditions by $P^{-1} = X > 0$ and setting $N_i = K_iX$, we obtain the LMIs [3.107a] and [3.107b].

Note that these constraints need to be solved $\forall (i,j) \in I_N^2$ verifying $\mu_i(z(t))\mu_j(z(t)) \neq 0$, which generally means that we can reduce the number of constraints (as is the case for activation functions with bounded support).

The stability conditions for the above theorem are rather conservative, because they require all the interlinked subsystems to be stable. The following theorem can alleviate this conservatism by reducing the constraints on the LMIs [3.102].

THEOREM 3.2.– If there are matrices P, Y_{ij} and N_i which do verify the following LMIs $\forall (i,j) \in I_N^2$, $i < j$:

$$XA_i^T + A_iX + N_i^T B_i^T + B_iN_i + Y_{ii} < 0 \qquad [3.107a]$$

$$XA_i^T + A_iX + XA_j^T + A_jX + B_iN_j + B_jN_i +$$

$$N_i^T B_j^T + N_j^T B_i^T + Y_{ij} + Y_{ij}^T \leq 0 \qquad [3.107b]$$

$$Y = Y^T = \begin{pmatrix} Y_{11} & \cdots & Y_{1n} \\ * & \ddots & \vdots \\ * & * & Y_{nn} \end{pmatrix} > 0 \qquad [3.107c]$$

then multi-model [3.100], with its control law defined by $u(t) = \sum_{i=1}^{N} \mu_i(z(t)) N_i X^{-1} x(t)$, is asymptotically stable.

These results enable us to further reduce the conservatism by introducing the matrices Y_{ij}, which are not necessarily symmetrical or defined, under the conditions of [3.107b]. Other intermediary results are not presented here: the interested reader can consult them in [CHA 00; GUE 04; JAD 99; JOH 98; KRU 08; LIU 03; TAN 01]. An illustrative example is given at the end of this section.

In the case of multi-models with control matrices satisfying the property relation $B_i = \alpha_i B$, $\alpha_i > 0$, $i : 1..N$, it is more advantageous to use the control law:

$$u(t) = \frac{\sum_{i=1}^{N} \mu_i\left(z(t)\right) \alpha_i K_i}{\sum_{i=1}^{N} \mu_i\left(z(t)\right) \alpha_i} x(t) \qquad [3.108]$$

leading to the closed-loop multi-model described by:

$$\dot{x}(t) = \sum_{i=1}^{N} \mu_i \left(z(t)\right)\left(A_i + B_i K_i\right) x(t) \qquad [3.109]$$

Note that the multi-model obtained is written without the crossed terms $(i \neq j)$. The stability conditions presented above are reduced to the existence of a matrix $X > 0$ such that:

$$XA_i^T + A_i X + N_i^T B_i^T + B_i N_i < 0, \forall i \in I_N \qquad [3.110]$$

Note that if the pairs (A_i, B_i) are controllable, then there are gains K_i that enable us to put the eigenvalues of $(A_i - B_i K_i)$ in the same place.

These stability conditions are less conservative than those resulting from the conventional control $u(t) = \sum_{i=1}^{N} \mu_i(z(t)) K_i x(t)$.

3.2.3.1. *α-stability: decay rate*

The above results guarantee only the stability of the multi-model with no performance criterion. The α-stability criterion, which guarantees a certain degree of shrinkage, is often used.

This result regarding the decay rate imposes additional constraints, which bring in dominant terms (G_{ii}) and crossed terms (G_{ij}) [3.106]. These additional constraints at the level of the crossed terms obviously lead to more restrictive conditions.

The following result can be used to achieve this objective by modifying only the so-called dominant terms, i.e. constraints [3.107a]. Constraints [3.107b] are relaxed by taking account of the maximum number (r) of local models that can be activated simultaneously. This number r is equal to the maximum number of local models when the activation functions are infinitely supported: i.e. $r = N$.

THEOREM 3.3.– Let r be the maximum number of local models activated simultaneously. If there are matrices $X > 0$, Y_{ij} and N_i and a scalar $\alpha \geq 0$ which satisfy $\forall i < j \in I_N$:

$$XA_i^T + A_iX + N_i^T B_i^T + B_iN_i + Y_{ii} + 2r\alpha X < 0$$
$$XA_i^T + A_iX + XA_j^T + A_jX + N_i^T B_i^T + B_iN_i + \quad [3.111a]$$
$$N_i^T B_j^T + B_jN_i + Y_{ij} + Y_{ij}^T \leq 0$$

$$\begin{pmatrix} Y_{11} & \cdots & Y_{1N} \\ * & \ddots & \vdots \\ * & * & Y_{NN} \end{pmatrix} > 0 \quad [3.111b]$$

then multi-model [3.100], with its control law $u(t) = \sum_{i=1}^{N} \mu_i(z(t))K_i x(t)$, where $K_i = N_i X^{-1}$, is asymptotically stable.

Note that the maximum decay rate can be obtained by considering the problem of a GEVP (Generalized EigenValue Problem) in $X > 0$ and α:

maximize α under constraint

Note also that the same result can be obtained by using other relaxation techniques.

State feedback control requires complete availability of the variables of state of the system. As this condition cannot always be fulfilled, two techniques are considered: *i*) observer-based control and *ii*) static output feedback control.

3.2.4. *Reconstructed state feedback control*

The observer considered is of the form:

$$\begin{cases} \dot{\hat{x}}(t) = \sum_{i=1}^{N} \mu_i\left(z(t)\right)\left(A_i\hat{x}(t) + B_iu(t) + L_i\left(y(t) - \hat{y}(t)\right)\right) \\ \hat{y}(t) = \sum_{i=1}^{N} \mu_i\left(z(t)\right)C_i\hat{x}(t) \end{cases} \quad [3.112]$$

The reconstructed state feedback control is thus written:

$$u(t) = \sum_{i=1}^{N} \mu_i\left(z(t)\right)K_i\hat{x}(t) \quad [3.113]$$

The dynamics of the estimation error $e(t) = x(t) - \hat{x}(t)$ is written:

$$\dot{e}(t) = \sum_{i=1}^{N}\sum_{j=1}^{N}\mu_i(z(t))\mu_j(z(t))\left(A_i - L_iC_j\right)e(t) \qquad [3.114]$$

Taking account of the control law [3.113] and the estimation error, the observer [3.114] becomes:

$$\dot{\hat{x}}(t) = \sum_{i=1}^{N}\sum_{i=1}^{N}\mu_i(z(t))\mu_j(z(t))\left(\left(A_i + B_iK_j\right)\hat{x}(t) + L_iC_je(t)\right) \qquad [3.114a]$$

The extended model is expressed by:

$$\dot{\bar{x}}(t) = \sum_{i=1}^{N}\sum_{i=1}^{N}\mu_i(z(t))\mu_j(z(t))\bar{G}_{ij}\bar{x}(t) \qquad [3.114b]$$

with:

$$\bar{G}_{ij} = \begin{pmatrix} A_i + B_iK_j & L_iC_j \\ 0 & A_i - L_iC_j \end{pmatrix} \qquad [3.115a]$$

$$\bar{x}(t) = \left(\hat{x}(t)^T \quad \tilde{x}(t)^T\right)^T \qquad [3.115b]$$

The stability conditions for the extended multi-model [3.114] can easily be obtain from conditions [3.107] by substituting matrices $G_{ij} = A_i + B_iK_j$ by $\bar{G}_{ij}$. The conditions obtained are nonlinear in relation to the variables P, Q, L_i and K_i. They can be transformed into LMIs by considering the diagonal Lyapunov matrices. The property of separation is also guaranteed in this case, as shown by the following result.

THEOREM 3.4.– If there are matrices $X > 0$, $Q > 0$, Q_{ij}, Q_{ij}, M_i and N_i which satisfy the following LMIs $\forall(i,j) \in I_N^2$, $i < j$:

$$XA_i^T + A_iX + N_i^TB_i^T + B_iN_i + Y_{ii} < 0 \qquad [3.116a]$$

$$X\left(A_i^T + A_j^T\right) + \left(A_i + A_j\right)X + B_i N_j + B_j N_i$$
$$+ N_i^T B_j^T + N_j^T B_i^T + Y_{ij} + Y_{ij}^T \leq 0 \qquad [3.116b]$$

$$\begin{pmatrix} Y_{11} & \cdots & Y_{1N} \\ * & \ddots & \vdots \\ * & * & Y_{NN} \end{pmatrix} > 0 \qquad [3.116c]$$

$$QA_i + A_i^T Q^T - Y_i C_i - C_i^T Y_i^T + Q_{ii} < 0 \qquad [3.117a]$$

$$Q\left(A_i + A_j\right) + \left(A_i + A_j\right)^T Q^T - Y_j C_i - Y_i C_j$$
$$- C_i^T Y_j^T - C_j^T Y_i^T + Q_{ij} + Q_{ij}^T < 0 \qquad [3.117b]$$

$$\begin{pmatrix} Q_{11} & \cdots & Q_{1N} \\ * & \ddots & \vdots \\ * & * & Q_{NN} \end{pmatrix} > 0 \qquad [3.117c]$$

then the extended multi-model [3.114], with its control law $u(t) = \sum_{i=1}^{N} \mu_i(z(t)) K_i \hat{x}(t)$, $K_i = N_i X^{-1}$ and its observer [3.112] defined by $L_i = P^{-1} Y_i$, is asymptotically stable.

The proof of this result is obtained by considering a Lyapunov matrix of the form:

$$P = \begin{pmatrix} P_1 & 0 \\ 0 & P_2 \end{pmatrix} \qquad [3.118]$$

with $P_1 = X^{-1}, P_2 = Q$ and changing variables $N_i = K_i X$ and $Y = PL_{ii}$.

SEPARATION PRINCIPLE.– These conditions enable us to separately design the observer and the controller, whilst still ensuring the stability of the looped system.

EXAMPLE 3.1.– Consider the multi-model in the following example:

$$A_1 = \begin{pmatrix} 2 & -10 \\ 1 & 0 \end{pmatrix}, \quad B_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \qquad [3.119a]$$

$$A_2 = \begin{pmatrix} 49 & -10 \\ 1 & 1 \end{pmatrix}, \quad B_2 = \begin{pmatrix} 10 \\ 0 \end{pmatrix} \qquad [3.119b]$$

with:

$$\mu_1(x_1(t)) = 1 - \mu_2(x_1(t)) = \begin{cases} 1 - \dfrac{|x_1(t)|}{3}, \forall x_1(t) \in [-3,3] \\ 0, \text{ elsewhere} \end{cases}.$$

Conditions [3.116] and [3.117] enable us to determine:

$$\begin{aligned} &K_1 = (4.92 \quad -0.53), K_2 = (5.18 \quad -0.09) \\ &P_1 = \begin{pmatrix} 0.028 & 0.043 \\ 0.043 & 0.147 \end{pmatrix}, Q_1 = \begin{pmatrix} 0.032 & 0.064 \\ 0.064 & 0.171 \end{pmatrix} \\ &L_1 = (3.0398 \quad -9.0539)^T, L_2 = (50.0398 \quad -9.0539)^T \\ &P_2 = \begin{pmatrix} 29.9427 & 1.5509 \\ 1.5509 & 29.9427 \end{pmatrix}, Q_2 = \begin{pmatrix} 8.4517 & 0.00 \\ 0.00 & 8.4275 \end{pmatrix} \end{aligned} \qquad [3.120]$$

The simulations shown in Figure 3.33, with initial conditions $x(0) = (1.7, -0.7)^T$ and $\hat{x}(0) = (1.5, -0.5)^T$, illustrate the global exponential stability of the extended system.

3.2.5. *Static output feedback control*

Consider the multi-model:

$$\begin{cases} \dot{x}(t) = \sum_{i=1}^{N} \mu_i(z(t))(A_i x(t) + B_i u(t)) \\ y(t) = Cx(t) \end{cases} \qquad [3.121]$$

with the control law in the form:

$$u(t) = \sum_{i=1}^{N} \mu_i(z(t)) F_i y(t) \qquad [3.122]$$

The closed-loop multi-model can be explicitly expressed as:

$$\dot{x}(t) = \sum_{i=1}^{N}\sum_{j=1}^{N} \mu_i\left(z(t)\right)\mu_j\left(z(t)\right)\bar{A}_{ij}\, x(t) \qquad [3.123]$$

where:

$$\bar{A}_{ij} = A_i + B_i F_j C \qquad [3.124]$$

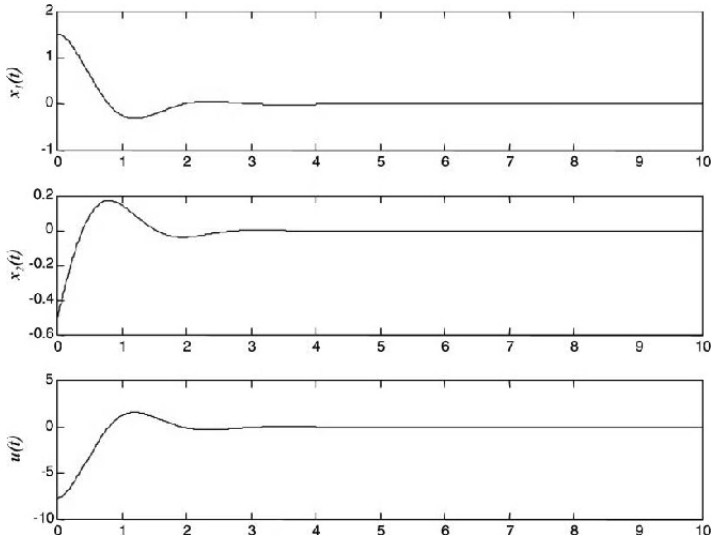


Figure 3.33. *Closed-loop system*

HYPOTHESIS 3.1.– Matrix C is full row rank.

The synthesis of this control law can be obtained directly by substituting matrix $G_{ij} = A_i - B_i K_j$ with matrix $\bar{A}_{ij}$ in the conditions of state feedback synthesis. For example:

$$P > 0,\ Q > 0 \qquad [3.125a]$$

$$\bar{A}_{ii}^T P + P\bar{A}_{ii} + Q_{ii} < 0 \qquad [3.125b]$$

$$\left(\bar{A}_{ij}+\bar{A}_{ji}\right)^T P+P\left(\bar{A}_{ij}+\bar{A}_{ji}\right)+Q_{ij}+Q_{ij}^T\le 0 \qquad [3.125c]$$

$$\begin{pmatrix} Q_{11} & \cdots & Q_{1N} \\ * & \ddots & \vdots \\ * & * & Q_{NN} \end{pmatrix} > 0 \qquad [3.125d]$$

These conditions are bilinear in P and $F_i \quad \forall i \in I_N$ and impossible to linearize by way of the conventional variable-change techniques. A linear formulation in the form of an LMI under linear algebraic constraints is suggested by the following result [CHA 02b].

THEOREM 3.5.– If there are symmetrical matrices $X > 0$, Q_{ij} and matrices Q_{ij}, N_i and M which satisfy the following conditions $\forall (i,j) \in I_N^2,\ i < j$:

$$A_i X + X A_i^T + B_i N_i C + C^T N_i^T B_i^T + Y_{ii} < 0 \qquad [3.126a]$$

$$X\left(A_i^T + A_j^T\right) + \left(A_i + A_j\right) X + B_i N_j C + B_j N_i C + C^T N_i^T B_j^T + C^T N_j^T B_i^T + Y_{ij} + Y_{ij}^T \le 0 \qquad [3.126b]$$

$$\begin{pmatrix} Y_{11} & \cdots & Y_{1N} \\ * & \ddots & \vdots \\ * & * & Y_{NN} \end{pmatrix} > 0 \qquad [3.126c]$$

$$CX = MC \qquad [3.126d]$$

then the closed-loop extended multi-model [3.123], with its control law [3.122] with $F_i = N_i M^{-1}$, is asymptotically stable.

Note that in the case of linear output feedback $F_i = F$, these conditions are reduced to the existence of matrices $X > 0$, N and M such that:

$$A_i X + X A_i^T + B_i N C + C^T N^T B_i^T < 0 \qquad [3.127a]$$

$$CX = MC \qquad [3.127b]$$

The parameters of the control law are defined by $F = NM^{-1}$. Note that as matrix C is assumed to be full row rank, we can deduce that $M = CXC^T\left(CC^T\right)^{-1}$, where $F_i = N_i M^{-1}$. Note also that the constraints obtained are linear and easily implemented. From a digital point of view, these conditions are easy to solve using already-existing numerical tools.

In the case of linear output feedback control $\left(F_i = F\right)$, it is sufficient to replace N_j with N in [3.126]. Note also that in the particular case of positive colinearity between the control matrices, the following control law is more advantageous than the output feedback control used [CHA 02b; CHA 07]:

$$u(t) = \frac{\sum_{i=1}^{N} \mu_i\left(z(t)\right)\alpha_i F_i}{\sum_{i=1}^{N} \mu_i\left(z(t)\right)\alpha_i} y(t) \qquad [3.128]$$

Indeed, the multi-model [3.121] with $B_i = \alpha_i B, \alpha_i > 0$ is written:

$$\dot{x}(t) = \sum_{i=1}^{N} \mu_i\left(z(t)\right)\overline{A}_{ii}x(t) \qquad [3.129]$$

where $\overline{A}_{ii} = A_i + B_i F_i C$. The conditions for synthesis of this control law with performance constraints become:

$$A_i X + XA_i^T + B_i N_i C + C^T N_i^T B_i^T < 0, \forall i \in I_N \qquad [3.130a]$$

$$CX = MC \qquad [3.130b]$$

The advantage to this control law lies in the fact that it requires only $(n+1)$ linear constraints instead of the $\left(n^2+1\right)$ required for the nonlinear control law [3.122].

EXAMPLE 3.2.– Consider multi-model [3.121] with $N = 2$ and:

$$A_1 = \begin{pmatrix} 2 & -10 \\ 1 & 0 \end{pmatrix}, \quad B_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

$$A_2 = \begin{pmatrix} 1 & -10 \\ 1 & 0 \end{pmatrix}, \quad B_2 = \begin{pmatrix} 10 \\ 0 \end{pmatrix}, \qquad C = \begin{pmatrix} 1 & 0 \end{pmatrix}$$

$$\mu_1(y(t)) = \frac{\left(1 - \tanh\left(y(t)\right)\right)}{2}, \ \mu_2(y(t)) = \frac{1 + \tanh\left(y(t)\right)}{2}$$

Note that $B_2 = \alpha B_1, \alpha = 10,$ control law [3.128] is used:

$$u(t) = \frac{\left(\mu_1\left(y(t)\right)F_1 + \mu_2\left(y(t)\right)\alpha F_2\right)}{\left(\mu_1\left(y(t)\right) + \mu_2\left(y(t)\right)\alpha\right)} y(t) \qquad [3.131]$$

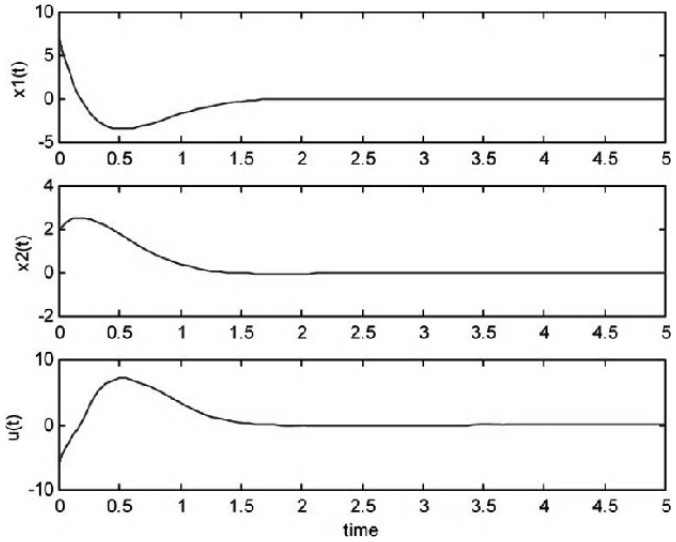


Figure 3.34. *Evolution of the states of the closed-loop system with initial conditions* $x(0) = (7,\ 2)$

Simulation of the evolution of the states of the multi-model with its control law demonstrates its asymptotic stability.

3.2.6. *Conclusion*

The earlier part of this chapter was given over to different control laws for linear systems. Thus, controllers such as the PID, the Smith predictor, the RST and

generalized predictive control were presented. Various examples were also given in order to illustrate these regulation techniques.

We then went on to examine the stability of nonlinear systems described by multi-models. We put forward conditions to be used to synthesize different control laws. To begin with, LMI conditions for the synthesis of state feedback control laws were determined. Then, controls based on reconstructed state feedback and static output feedback were examined. These controls were presented as LMI formulations. Illustrative examples were given.

3.3. Bibliography

[ALE 73] ALEVISAKIS G., SEBORG D.E., "An extension of the Smith predictor method to multivariable linear systems containing time delays", *International Journal of Control*, vol. 18, pp. 541–551, 1973.

[AST 94] ASTROM K.J., HANG C.C., LIM B.C., "A new Smith predictor for controlling a process with an integrator and long dead-time", *IEEE Transactions on Automatic*, vol. 39, no. 2, 1994.

[BOR 93] BORNE P., DAUPHIN-TANGUY G., RICHARD J.-P., *et al.*, *Analyse et régulation des processus industriels, Tome 2 – Régulation numérique*, Technip, Paris, 1993.

[BOU 96] BOUCHER P., DUMUR D., *La Commande Prédictive, Méthodes et Pratiques de l'Ingénieur*, Technip, Paris, 1996.

[BOY 94] BOYD S., EL GHAOUI L., FERON E., *et al.*, "Linear matrix inequalities in system and control theory", *SIAM*, Philadelphia, 1994.

[BRE 02] BREGSEN P., PALM R., DRIANKOV D., "Observers for Takagi-Sugeno fuzzy systems", *IEEE Transactions on Systems, Man and Cybernetics, Part B*, vol. 32, no. 1, pp. 114–121, 2002.

[CAM 93] CAMACHO E.F., "Constrained generalized predictive control", *IEEE Transactions on Automatic Control*, vol. 38, no. 2, 1 February 1993.

[CHA 00] CHADLI M., MAQUIN D., RAGOT J., "Relaxed stability conditions for the T-S fuzzy systems", *IEEE International Conference on Systems, Man, and Cybernetics*, vol. 5, pp. 3514–3519, 2000.

[CHA 02a] CHADLI M., MAQUIN D., RAGOT J., "Output stabilization in multiple model approach", *International Conference on Control Applications*, vol. 2, pp. 1315–1320, 2002.

[CHA 02b] CHADLI M., Stabilité et stabilisation des multimodèles, Doctoral thesis, INPL-CRAN, Nancy, December 2002.

[CHA 06a] CHADLI M., ELHAJJAJI A., "Observer-based robust fuzzy control of nonlinear systems with parametric uncertainties-comment on", *Fuzzy Sets and Systems Journal*, vol. 157, no. 9, pp. 1276–1281, 2006.

[CHA 06b] CHADLI M., "On the stability analysis of uncertain fuzzy models", *International Journal of Fuzzy Systems*, vol. 8, no. 4, pp. 224–231, December 2006.

[CHA 08a] CHADLI M., AKHENAK A., RAGOT J., *et al.*, "On the design of observer for unknown inputs fuzzy models", *International Journal of Automation and Control*, vol. 2, no. 1, pp. 113–125, 2008.

[CHA 08b] CHADLI M., "Multimodèles : origine et méthodes d'obtentions", *Revue d'électricité et d'électronique*, REE-SEE, no. 10, pp. 51–54, 2008.

[CHA 08c] CHADLI M., AKHENAK A., MAQUIN D., *et al.*, "Fuzzy observer for fault detection and reconstruction of unknown input fuzzy models", *International Journal of Modelling, Identification and Control*, vol. 3, no. 2, pp. 193–200, 2008.

[CHA 08d] CHADLI M., BORNE P., "Stabilité des multimodèles", *Revue d'électricité et d'électronique*, REE-SEE, no. 10, pp. 55–59, 2008.

[CHA 08e] CHADLI M., "Commande et observation des multimodèles", *Revue d'électricité et d'électronique*, REE-SEE, no. 10, pp. 60–65, 2008.

[CHA 09] CHADLI M., AKHENAK A., RAGOT J., *et al.*, "State and unknown input estimation for discrete time multiple model", *Journal of the Franklin Institute*, vol. 346, no. 6, pp. 593–610, 2009.

[CHA 10a] CHADLI M., "Chaotic systems reconstruction", *Evolutionary Algorithms and Chaotic Systems*, vol. 267, Springer-Verlag, Berlin, p. 560, 2010.

[CHA 10b] CHADLI M., "State and an LMI approach to design observer for unknown inputs Takagi-Sugeno fuzzy models", *Asian Journal of Control*, vol. 12, no. 4, 2010.

[CHA 12a] CHADLI M., BORNE P., *Multimodèles en Automatique : Outils avancés d'analyses et de synthèse*, Hermès, Paris, 2012.

[CHA 12b] CHADLI M., BORNE P., *Multiple Models Approach in Automation: Takagi-Sugeno Fuzzy Systems*, John Wiley & Sons, London, 2013.

[CHA 12c] CHADLI M., GUERRA T.M., "LMI solution for robust static output feedback control of Takagi-Sugeno fuzzy models", *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 6, 2012.

[CHA 12d] CHADLI M., KARIMI H.R., "Robust observer design for unknown inputs Takagi-Sugeno models", *IEEE Transactions on Fuzzy Systems*, 2012.

[CLA 87] CLARKE D.W., MOHTADI C., TUFFS P.S., "Generalized predictive control – Part I. the basic algorithm", vol. 23, no. 2, pp. 137–148, "Generalized predictive control – Part II. extensions and interpretations", vol. 23, no. 2, pp. 149–160, *Automatica*, 1987.

[DON 77] DONOGHUE J.F., "A comparison of the Smith predictor and optimal design approaches for systems with delay in the control", *IEEE Transactions on Industrial Electronics and Control Instrumentation*, vol. IECI-24, no. 1, pp. 109–117, February 1977.

[EDE 05] EDER H., "PID Tuning – science, art or both?", *Control Engineering Europe*, Tonbridge, September 2005.

[FUL 68] FULLER A.T., "Optimal nonlinear control of systems with pure delay", *International Journal of Control*, vol. 8, no. 2, pp. 145–168, 1968.

[FLA 94] FLAUS J.-M., *La régulation industrielle*, Hermès, Paris, 1994.

[GUE 04] GUERRA T.M., VERMEIREN L., "LMI-based relaxed nonquadratic stabilization conditions for nonlinear systems in the Takagi–Sugeno's form", *Automatica*, vol. 40, no. 5, pp. 823–829, 2004.

[JAD 99] JADBABAIE A., "A reduction in conservatism in stability and L2 Gain analysis of T-S fuzzy systems *via* Linear matrix inequalities", *IFAC, 14th triennial World congress*, Beijing, China, pp. 285–289, 1999.

[JOH 03] JOHANSEN T.A., BABUSKA R., "Multiobjective Identification of Takagi-Sugeno Fuzzy Models", *IEEE Transactions on Fuzzy Systems*, vol. 11, no. 6, pp. 847–860, 2003.

[KRU 08] KRUSZEWSKI A., WANG R., GUERRA T.M., "Nonquadratic stabilization conditions for a class of uncertain nonlinear discrete time TS fuzzy models: a new approach", *IEEE Transactions on Automatic Control*, vol. 53, no. 2, pp. 606–611, 2008.

[LAN 02] LANDAU I.D., *Commande des systèmes – conception, identification et mise en oeuvre*, Hermès, Paris, 2002.

[LON 06] LONGCHAMP R., *Commande Numérique de Systèmes Dynamiques – Cours d'automatique*, 2nd edition, Presses Polytechniques et universitaires romandes, Lausanne, 2006.

[MA 98] MA X.J., SUN Z.Q., HE Y.Y., "Analysis and design of fuzzy controller and fuzzy observer", *IEEE Transactions of Fuzzy Systems*, vol. 9, no. 1, pp. 41–51, 1998.

[MAR 99] TEIXEIRA C.M., STANISLAW H., "Stabilising controller design for uncertain nonlinear systems using fuzzy models", *IEEE Transactions of Fuzzy Systems*, vol. 7, no. 2, pp. 133–142, 1999.

[MAT 96] MATAUSEK M.R., MICIC A.D., "A modified Smith predictor for controlling a process with an integrator and long dead-time", *IEEE Transactions on Automatic Control*, vol. 41, no. 8, August 1996.

[PIG 96] PIGERON B., MULLOT H., CHAIX A., *et al.*, *Boucles de Régulation*, 3rd edition, Bhaly Autoédition, Saint-Rémy de Provence, 1996.

[RAM 01] RAMOND G., Contribution à la commande prédictive généralisée adaptative et applications", thesis University of Paris XI, UFR Scientifique d'Orsay, Paris, September 2001.

[RIC 93] RICHALET J., *Pratique de la commande prédictive*, Hermès, Paris, 1993.

[RIC 04] RICHALET J., DREYFUS G., LAVIELLE G., *et al.*, *La commande prédictive, Mise en oeuvre et applications industrielles*, Eyrolles, Paris, 2004.

[SMI 59] SMITH O.J.M., ATHERTON D.P., "A controller to overcome dead band time", *ISA J.*, vol. 6, no. 2, pp. 28–33, 1959.

[TAN 98] TANAKA K., IKEDA T., HE Y.Y., "Fuzzy regulators and fuzzy observers: relaxed stability conditions and LMI-based design", *IEEE Transactions on Fuzzy Systems*, vol. 6, no. 1, pp. 250–256, 1998.

[TAN 01] TANAKA K., HORI T., WANG H.O., "A fuzzy Lyapunov approach to fuzzy control system design", *IEEE American Control Conference*, vol. 6, pp. 4790–4795, 2001.

[WAT 81] WATANABE K., "A process-model control for linear systems with delay", *IEEE Transactions on Automatic Control*, vol. AC-26, no. 6, pp. 1261–1266, 1981.

[ZHU 93] ZHUANG M., ATHERTON D.P., "Automatic tuning of optimum PID controllers", *IEE Proceedings-D*, vol. 140, no. 3, May 1993.

[ZIG 42] ZIEGLER J.G., NICHOLS N.B., "Optimum settings for automatic controllers", *Transactions ASME*, 1942.

Chapter 4

Application to Cryogenic Systems

4.1. Introduction

4.1.1. *Cryogenics and its applications at CERN*

CERN, the European Center for Nuclear Research, is one of the largest and most respected centers for scientific research. Founded in 1954 by twelve European states, the organization now has twenty-eight member states. The laboratory is located on the Franco-Swiss border to the west of Geneva, at the foot of the Jura Mountains. Its purpose for existing is fundamental physics: the discovery of what the universe is made of and the way in which it works. At CERN, the largest and most complex scientific instruments in the world are used to study the basic components of matter: the fundamental particles. By investigating what happens when these particles collide, physicists are learning more about the laws of nature. At present, around 10,000 physicists from research institutions around the globe use the installations at CERN for their experiments.

The instruments used at CERN are particle accelerators and detectors. The accelerators shoot beams of particles at high energies, having them collide either with each other or with stationary targets. The detectors observe and record the results of these collisions. The Large Hadron Collider (LHC), the largest and most powerful accelerator ever built, is the final link in CERN's complex of accelerators (see Figures 4.1 and 4.2). It is a circular particle accelerator with a circumference of 27 km, located approximately 100 meters underground, used by scientists to study the smallest particles in existence, the fundamental building blocks of everything

---

Chapter written by Marco PEZZETTI, Hervé COPPIER and Mohammed CHADLI.

(see Figures 4.3a and 4.3b). The LHC makes use of the underground tunnel 27 km in circumference and the technical infrastructure of the old LEP (*Large Electron Positron collider*). The LHC is not perfectly circular: it is divided into eight arcs of 2.9 km with eight straight sections of around 500m (see Figure 4.4). Two beams of subatomic particles called "hadrons", which are either protons or lead ions, travel in opposite directions in the circular accelerator, gaining energy on each circuit. The machine is used to recreate the conditions of the instant after the Big Bang, by bringing two beams into direct collision at very high energy. Groups of scientists the world over analyze the particles created during these collisions using special detectors in numerous experiments devoted to the LHC.

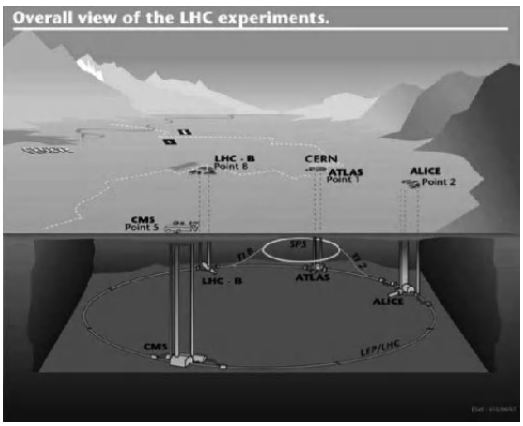


Figure 4.1. *Overall view of the LHC experiments*

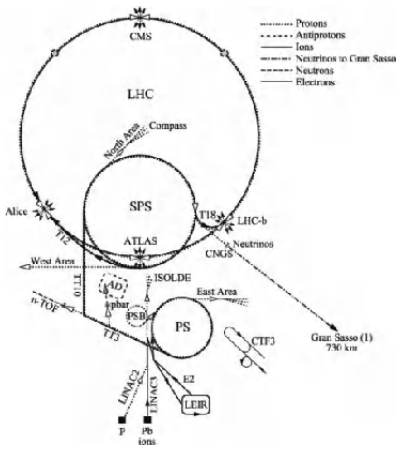


Figure 4.2. *The accelerators at CERN*

The six LHC experiments are all international collaborations. Each experiment is different, and characterized by its own specific particle detector. The two largest experiments, ATLAS and CMS, use polyvalent detectors to analyze the myriad of particles produced during collisions in the accelerator, and thus study the most diverse aspects of physics. These two detectors, designed independently of one another, are able to gather data in the case of a discovery. Two medium-scale experiments, ALICE and LHCb, are equipped with specialized detectors, and analyze specific phenomena during collisions in the LHC. Two other experiments on a far smaller scale, TOTEM and LHCf, study the hadrons which narrowly miss being involved in a head-on collision. Indeed, when two beams circulating in opposite directions reach the point of collision, only a relatively few particles actually collide head on. Others brush past each other, while the large majority of them continue on their way without encountering other particles. Those which merely touch lightly are very slightly diverted from the trajectory of the beam: these are "small-angle particles" and are analyzed by TOTEM and LHVoir. The ATLAS, CMS, ALICE and LHCb detectors are installed inside four enormous caverns situated along the circumference of the LHC (see Figure 4.4). The detectors for the TOTEM experiment are located near to the CMS detector, and those for the LHCf experiment are near the ATLAS detector.

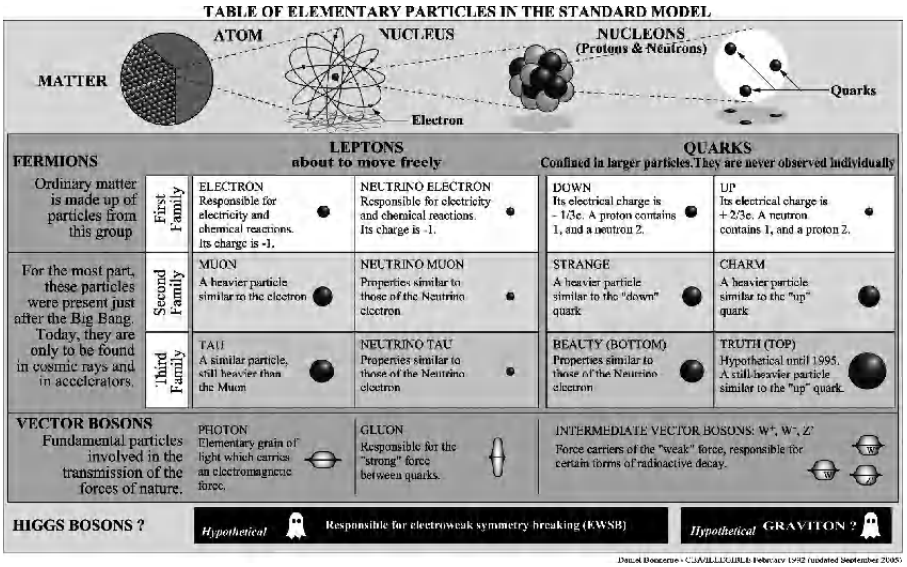


Figure 4.3a. *Description of the "old" standard model*

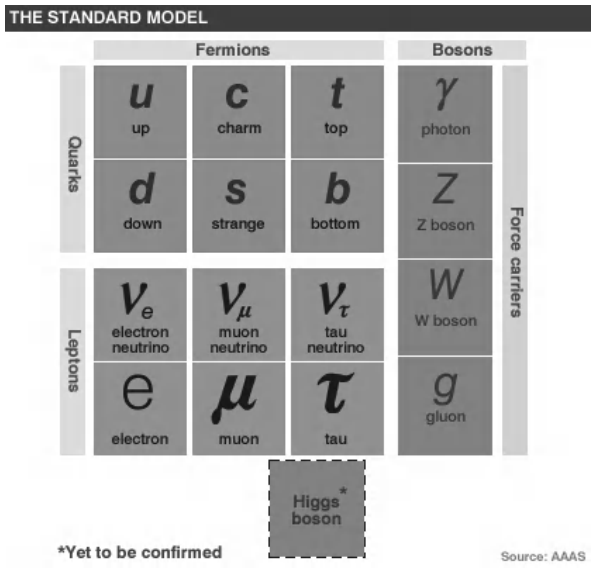


Figure 4.3b. *Description of the "new" standard model (after the July 2012 discovery of the Higgs boson)*

The beams circulating in the LHC are accelerated by radio-frequency (RF) cavities to over 99% the speed of light, thus attaining a nominal energy of 7 TeV per beam, which gives a nominal energy of collision of 14 TeV. They collide in the center of the enormous detectors built to collate the results. In order to avoid collisions with molecules of gas present in the accelerator, the particles circulate in vacuum chambers as void as outer space, called an ultra-vacuum (the pressure inside the LHC is 10^{-13} atmospheres). The vacuum is also necessary for the thermal insulation of the magnets in their cryostat.

The beams are curved and steered using a magnetic field, produced by dipole and quadripole superconductor magnets. The curving magnets, the dipoles, must include two magnets within the same body (double aperture – see Figure 4.5), for the two vacuum tubes where the protons are circulating in opposite directions. At their core, these magnets, 15 m in length and weighing around 35 tons, produce a 9-tesla magnetic field – around 200,000 times the strength of the Earth's magnetic field. They needed to be constructed on an industrial scale, because the LHC requires 1,232 of them. The beams are focused by quadripole magnets, which produce a gradient of 223 $T \cdot m^{-1}$. These magnets also include two vacuum tubes and are integrated into the curved and straight sections. There are over 500, ranging from 2 to 5 m in length. These main magnets are joined by all the correction systems, also based on superconductor magnets (dipoles, quadripoles, sextupoles, octupoles,

decapoles, dodecapoles, etc.). The total number of superconductor magnets at CERN is nearly 8,000 superconductive units. All these magnets have to produce reproducible fields, controlled to a degree of precision of up to 10^{-4}.

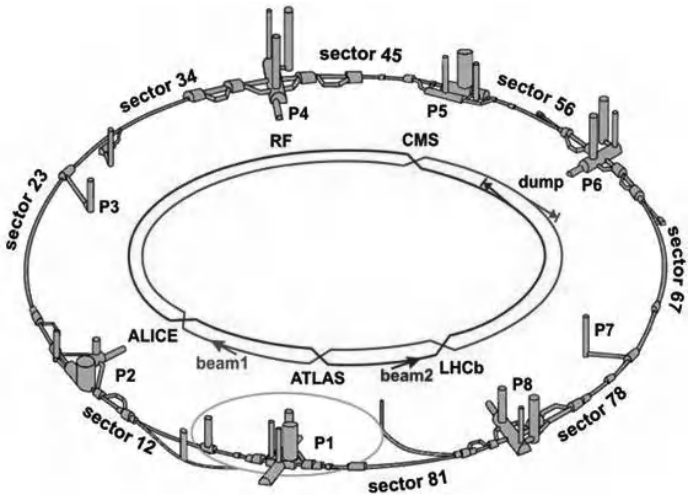


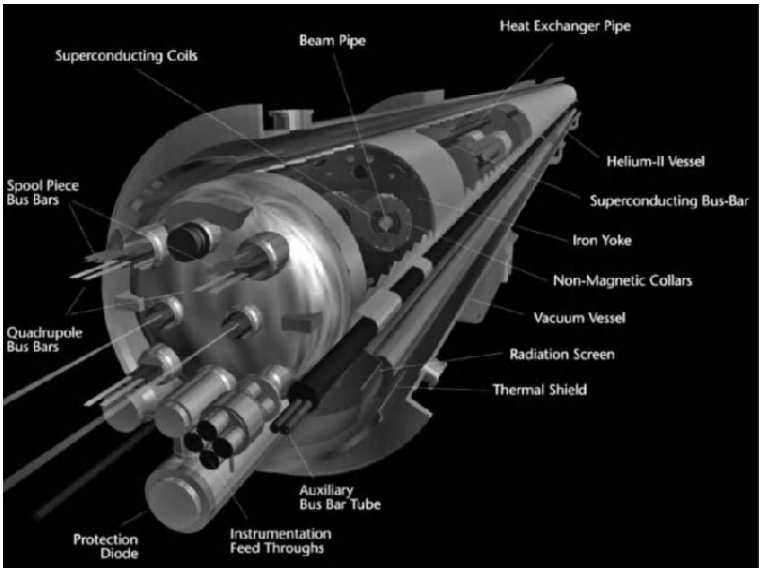
Figure 4.4. *Diagram of the LHC*

All the aforementioned performances, which are absolutely necessary in order to produce a stable beam with a long-enough life expectancy, are achieved by using superconductor magnets with Niobium-Titanium (NbTi) coils, cooled to 1.9 K so as to facilitate the circulation of a nominal current of approximately 12 kA. Furthermore, the safe and proper operation of the detectors and the RF cavities requires cryogenic temperatures.

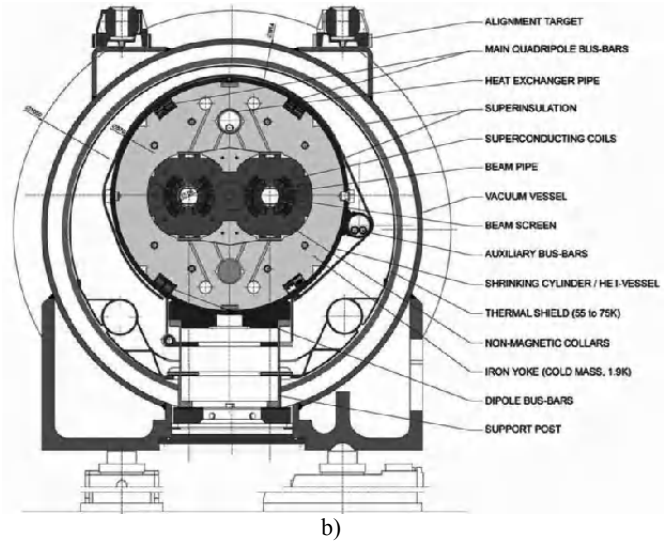
Consequently, the operation of the accelerator requires cryogenic systems that are capable of delivering extremely low temperatures. The LHC uses the largest cryogenic installation in the world, using helium as a cooling liquid. This installation is controlled by industrial "Programmable Logic Controllers" – PLCs. The manufacture both of the installations and of the control systems constituted a major challenge, as it was the first time that these cutting-edge technologies had been used in a complex system on such a vast scale.

This chapter presents the issue of models and control systems applied to certain cryogenic installations used at CERN, and draws on the experience accumulated over the course of over ten years during the construction, development and installation of the LHC's cryogenic equipment. One of the most important points about this work is the mathematical modeling of the physical phenomena involved

in the cryogenic processes. In such systems, the superposition principle is not often applicable, because of extensive nonlinearities in the relations between the causes which act on the system and their effects.



a)



b)

Figure 4.5. *a) The double-aperture dipole magnet at the LHC and b) a cross-section of it*

In large-scale installations such as those presented hereafter, the possibility for experimentation is greatly limited, both by the costs and by the risks involved in the manufacture of these installations. For this reason, in this chapter, extensive use will be made of simulation and modeling in the design process. However, under exceptional circumstances, it was possible to carry out a campaign of experiments on a nitrogen heat exchanger to identify the process being controlled. It should also be mentioned that, in the context described below – i.e. models and control applied on a massive scale to nonlinear systems – there may be problems in terms of the practical application of the traditional, well-established techniques. Furthermore, we can develop new advanced modeling and control techniques to optimize the management of the installations, with the aim of enhancing the performance of the control systems in terms of fidelity to the desired behavior, thereby reducing the undesirable effects which could, in the long term, drive up operational costs and affect the availability of the system.

4.1.2. *Some basics about cryogenics*

Up until now, the study and use of superconductors required the use of rather complex cryogenic devices. While such devices have begun to become less costly due to the discovery of so-called high-critical-temperature ceramic superconductors – which reach a superconductive state below the liquefaction temperature of nitrogen (77 K) – there is nothing to indicate that superconductivity at ambient temperature will become a possibility in the near future; and as Ortoli and Klein [ORT 94] write: "…the history of superconduction is inextricably linked with the history of the quest for low temperatures".

4.1.2.1. *Heat transport mechanisms*

A cryogenic device must be capable of maintaining a temperature far lower than ambient temperature, in a certain experimentation area which is thermally insulated from the outside. However, heat can pass from a hot body (at temperature T_2) to a cold body (T_1) by conduction, convection or radiation. These three contributions to transferred power obey different laws which can be expressed as follows, giving us a configuration similar to that shown in Figure 4.6.

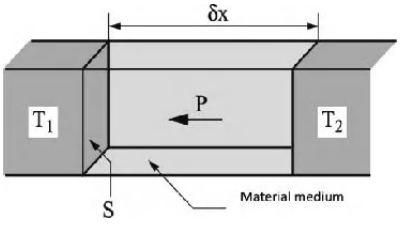


Figure 4.6. *Power transferred between two bodies at temperatures T_1 and T_2*

– *Conduction:*

$$\Delta P_{Conduction} = \lambda \tilde{\lambda} S \Delta T \quad [4.1]$$

– *Convection:*

For laminar flow of the fluid separating the two bodies:

$$\alpha(\Delta T) = \alpha_0 \Delta T^{1/4} \quad [4.2]$$

For a turbulent flow:

$$\alpha(\Delta T) = \alpha_0 \Delta T^{1/3} \quad [4.3]$$

provided the minimum temperature difference required for the initiation of convection (dependent on the viscosity, the expansion coefficient, etc.) is reached.

– *Radiation:*

The power radiated by a surface at temperature T is given by the Stefan–Boltzmann law:

$$P_{Radiation} = AS\sigma T^4 \quad [4.4]$$

with the Stefan–Boltzmann coefficient = 5.6710^{-8}[$Wm^{-2}K^{-4}$] and the absorption power, dependent upon the state of the surface: $A = 0$ for a mirrored surface; $A = 1$ for a black surface.

Thus, the balance of power transferred between the two surfaces will depend on their powers of absorption and their reflective coefficients.

Conductive and convective heat transport takes place in material media; we can therefore try to reduce these phenomena by surrounding the cryogenic reservoir with as large a vacuum as possible. As regards radiation, which is heavily dependent upon temperature, it can be controlled by the use of thermal screens.

4.1.2.2. *Thermal screens*

A thermal screen is constituted by an object placed between the cold zone and the warm zone, and kept at an intermediary temperature. Usually, it will reflect a large proportion of the heat emitted by the warm zone, whilst itself only radiating far less thermal power in the direction of the cold zone (owing to the T^4 dependency). The surface of the screen will therefore be treated so as to be as highly reflective as possible. The same is also true – and for the same reason – for the cold zone.

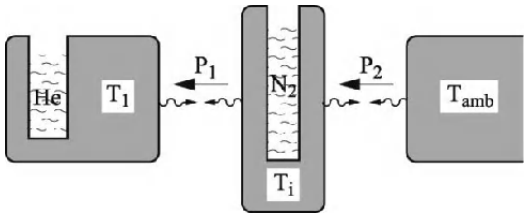


Figure 4.7. *Powers transported in the presence of a thermal screen*

4.1.2.3. *Cryogenic liquids*

The main cryogenic liquids used are liquid helium (to cool the conventional superconductors, between 0.5 and 4.2 K) and liquid nitrogen (for the thermal screens at 77 K and to cool high-critical-temperature superconductors), and less frequently, liquid hydrogen (which can be used to explore a temperature range between 14 and 20 K). In addition, the two helium isotopes (^{3}He and ^{4}He) are used, either separately or together, in the dilution cryostats. Table 4.1 mentions some of the properties of these elements.

	$T_{Liquefaction}$ [K]	$T_{Solidification}$ [K]	$T_{Inversion}$ [K]	Density
N_2	77.36	63.3	621	0.808
^{4}He	4.21	superfluid at 2.17 K	51	0.121
^{3}He	3.18	superfluid at 2.6 mK	23	0.082
H_2	20.36	14.02	205	0.070

Table 4.1. *Characteristics of the typical cryogenic liquids*

Because of the heat absorbed by the cryogenic bath(s), these first undergo evaporation which absorbs an amount of energy proportional to the quantity of liquid evaporated, due to the latent heat of this phase transition. The gas is then at its liquefaction temperature and begins to warm up again, until it reaches (at most) the outside temperature. Figure 4.8 gives an indication of the amounts of energy involved in these processes, for normal helium (^{4}He) and for nitrogen, and an idea of the order of magnitude of the corresponding prices in US dollars.

Figure 4.9 represents a complete cryogenic system using nitrogen and helium. Helium is often used in a closed circuit because of its price. It is liquefied during the course of a thermodynamic cycle including pre-cooling below its inversion temperature (see Table 4.1), and then undergoes a Joule-Thomson expansion: during this process the initial pressure (pressure at which gaseous helium is stored: 200 atmospheres) and final pressure (atmospheric pressure) are kept constant. Nitrogen,

for its part, is gleaned from the air when liquefied by a thermodynamic cycle, with the oxygen being separated because of its higher liquefaction temperature of 13 K.

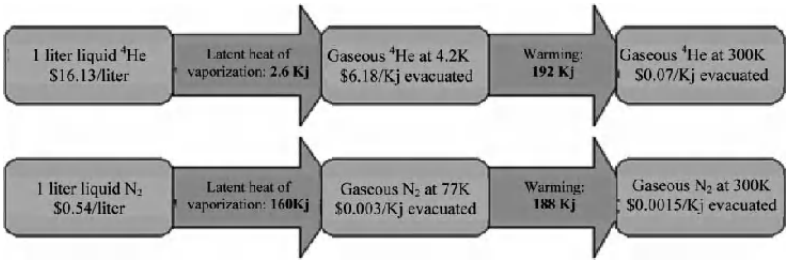


Figure 4.8. *Energies at play in a cryogenic device*

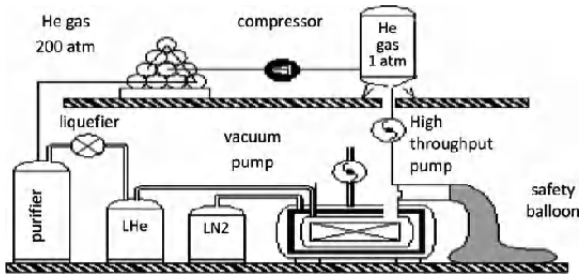


Figure 4.9. *Complete cryogenic device using ^{4}He and nitrogen*

4.2. Modeling and control of a cryogenic exchanger for the NA48 calorimeter at CERN

Installed at CERN in Geneva, the NA62 experiment is a continuation of the NA48 experiment, which was designed at CERN to study the dissymmetry between matter and antimatter by observing the direct violation of the CP (charge parity) symmetry, referred to simply as "CP violation" on the system of neutral kaons[1] or K0- K 0 (neutral kaon and antikaon), yielded by the SPS (super proton synchrotron) particle accelerator.

To this end, researchers observed the rates of decay of neutral kaons and antikaons and sought to demonstrate a difference between these decay rates. The

1 CP violation is said of particles which do not behave in exactly the same way as their antiparticle. A kaon is an elementary particle (K) of the meson family, comprising a quark and an antiquark. There are four possible types of kaons, depending on their charge: a positive kaon, a neutral kaon, a negative kaon and a neutral antikaon.

experiment used high-intensity beams of protons of 400GeV energy to produce neutral and charged kaons.

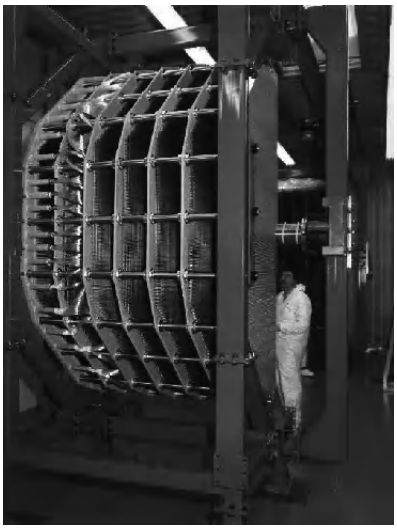


Figure 4.10. *Photograph of the NA48 liquid krypton calorimeter*

The experiment comprises a system to detect and identify particles (a spectrometer and calorimeter). The calorimeter, called the LKr (liquid krypton) calorimeter, is an ionization chamber containing $9m^3$ of liquid krypton used as an active medium and as an absorber.

It is able to detect neutral modes of decay by measuring the energy, position and time of the electromagnetic cascades caused by the photons in the krypton. Charged modes of decay are analyzed by a magnetic spectrometer. The energy absorbed by the krypton is measured on the basis of the current induced on electrodes by the ionization that took place. The electrodes are rectangular tapes of Cu-Be-Co, 18 mm wide and 40 μm thick, lain in the direction of increasing depth of krypton and subjected to a voltage of 2 N. The segmentation of the calorimeter is done by turns, with cathodes and anodes alternating at every 10 mm all along the horizontal. The ionization current induced on the anodes is measured by the electronic measuring array. The surface of the calorimeter has 13,212 cells on it. The electrodes are arranged in an accordion-like trajectory in relation to the projection lines. For this reason, the electrodes are held by spacers, placed at 21 cm intervals with a 1 cm gap. The angle thus formed between the accordion and the projective directions is 50 mrad (see Figure 4.11).

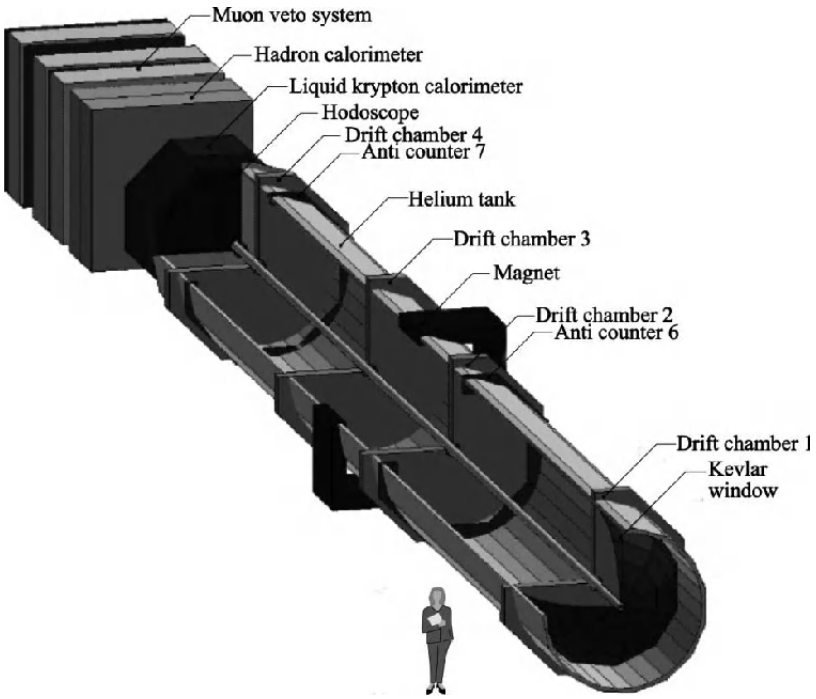


Figure 4.11. *Diagram of the whole of the different parts of the NA48 detector*

Approximately 9 m^3 of krypton are contained in a cryostat, to a depth of 125 cm, corresponding to 27 lengths of radiation. The showers of protons and electrons are therefore entirely contained within the krypton. At atmospheric pressure (1,013 bar), krypton boils at 119.8 K. The variation in drift rate of the secondary electrons with changing temperature is –0.87 %/K. We can therefore see that it is important to maintain the calorimeter at a constant temperature by using a thermal regulation system. The cryogenic system must make as little a contribution as possible to the presence of inert material in front of the krypton, not only to minimize the degradation of the uniformity of the electromagnetic cascades, but also to avoid a backsplash signal towards the charged hodoscope, the purpose of which is to provide additional information about the time of the neutral events and measure the efficiency of neutral release. The cryogenic installation in the calorimeter contains different cryogenic liquids in its reservoir, serving to maintain the state of the krypton. Consequently, an intermediary argon cooling circuit and a higher upper circuit of liquid nitrogen serve to keep the temperature of the krypton constant. In view of the high value of the krypton contained in the calorimeter, which is due to the rarity of that element, the control system is under study with a view to reducing faults which might cause a loss of liquid, as far as possible.

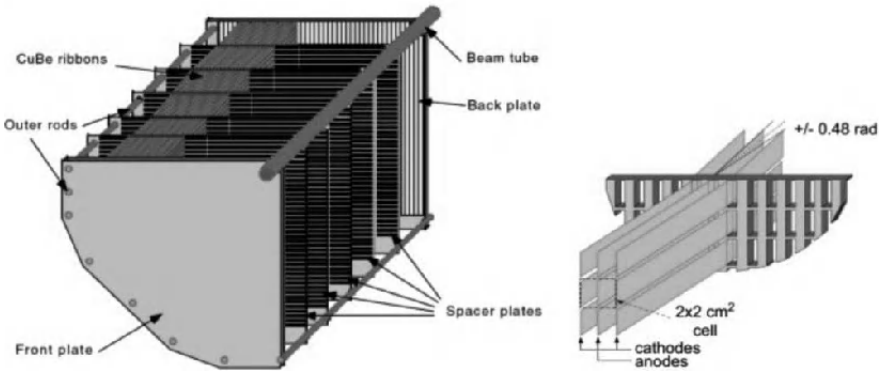


Figure 4.12. *Structure of the electrodes in the NA48 detector*

4.2.1. *Description of the cryogenic installations in the NA48 calorimeter*

We are going to describe the cryogenic installations which make the calorimeter work. The boiling point of krypton is 119.8 Kelvin at 1.013 bar. In addition, it is important to have a stable temperature in the calorimeter and a pressure regulated around 1.05 bar, to ±0.01 bar, so 117 K at saturation. The cryogenic system therefore must be able to deliver stable thermal conditions and ensure quality storage with no loss of liquid.

Under normal conditions, an intermediary cooler, containing liquid argon in saturation at around 10 bars (117 K) is able to recover the gaseous krypton to return it to the cryostat and obtain good thermal stability. The argon is then cooled by a nitrogen circuit. Direct cooling of the krypton in the cryostat by a liquid nitrogen circuit is done only in case of an emergency. It is difficult to constantly cool the krypton directly by using nitrogen, because this requires a pressure of nearly 25 bars in the exchanger, and the temperature difference between the two cryogenic fluids would cause the krypton to solidify on the surface of the exchanger, severely damaging its effectiveness. If the argon condenser is used, the liquid krypton is returned to the calorimeter at a temperature of 117 K and we avoid causing a temperature gradient in the cryostat, which would skew the physical measurements. The cryogenic system is made up of the cryostat of the calorimeter with its purification and cooling system, a storage reservoir for the krypton, two liquid nitrogen storage reservoirs, a condenser for the krypton storage tank and argon high pressure storage. The calorimeter is equipped with a cooling and heating system, used when the krypton is placed inside. Liquid transfer is performed by a centrifugal pump, and there are several systems in place to purify the krypton in both the gaseous and liquid phases.

4.2.1.1. *Cryostat of the NA62 calorimeter*

The cryostat has a capacity of 9,000 liters, and is filled with liquid krypton. It is possible to cool the liquid krypton directly using nitrogen, but it is not convenient to do so in this case, because the system requires a high degree of stability and precision, and the significant temperature difference between these two fluids would render control difficult or cause the krypton to solidify. For this reason, in order to solve the problem, the krypton passes into a condenser using liquid argon as an intermediary cooling agent. Liquid nitrogen could be used to cool the krypton in case of very rapidly climbing pressure, by way of the heat exchangers located directly on the calorimeter and the storage reservoir, as shown by Figure 4.13. During data acquisition, the pressure of the saturated krypton is regulated at around 1.05 bar, to ±0.02 bar.

4.2.1.2. *Krypton liquefier*

The krypton liquefier is situated outside the cryostat and is used to liquefy the krypton that evaporates out of the calorimeter. Using the heat exchangers below the bath of liquid argon saturated at 10 bars at 117 K (the melting point of krypton is 115.9 K at 0.72 bar), we introduce the gaseous krypton, which will condense and be recovered in liquid form. The argon, in turn, is cooled by liquid nitrogen circulating in a heat exchanger, passing into the gaseous space of the argon chamber, thus enabling the desired pressure to be maintained in the condenser.

The "cold surface" of the condenser, around 3m² for a cooling power of 4.5 kW, comprises 230 vertical pipes which join, at the top, to the argon tank. The krypton therefore condenses directly on the surface of these pipes, and the droplets fall into a collector and are returned to the calorimeter at a temperature of 117 K. The nitrogen circuit attached to the condenser provides saturated liquid nitrogen, which is used to evacuate the heat by evaporation, and the temperature of which can be regulated by altering the pressure. When it condenses, the argon passes heat to the nitrogen, which uses that heat to vaporize, as the nitrogen is at boiling point. The energy used by the nitrogen to vaporize corresponds to the energy given off by the argon to condense on the exchanger. There is no need to raise the temperature of the nitrogen because it is already at its boiling point. The nitrogen circuit in the condenser has two regulating valves – an inlet valve where the nitrogen enters the condenser, and an outlet valve. Using the first valve, we modify the flow of liquid nitrogen unto the exchanger in order to ensure that there is always a certain percentage (1 − Xvap) of liquid at the outlet; Xvap represents the percentage of nitrogen evaporated. Indeed, if the flowrate is too low and the nitrogen vaporizes as soon as it enters the exchanger, the effectiveness is greatly reduced. On the other hand, if we consider that a percentage of the mass of nitrogen cannot be vaporized on exiting the exchanger, then we can deduce from this that cooling took place throughout the whole of the exchanger. By adjusting this valve, we can control the pressure in the cryostat by

adjusting the pressure of the saturated argon in the condenser. The outlet valve, for its part, directly influences the pressure of liquid nitrogen inside the exchanger, and consequently its temperature.

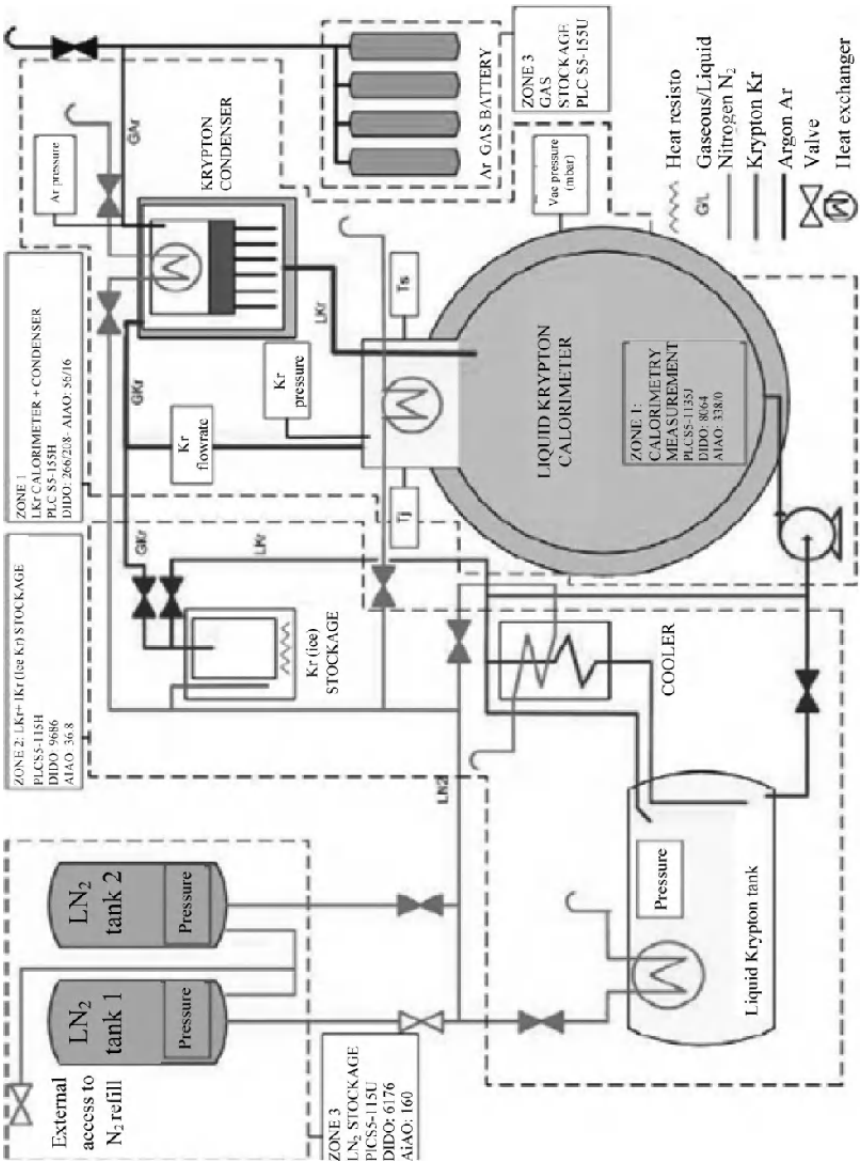


Figure 4.13. *Diagram of the cryogenic installations of the NA62 calorimeter*

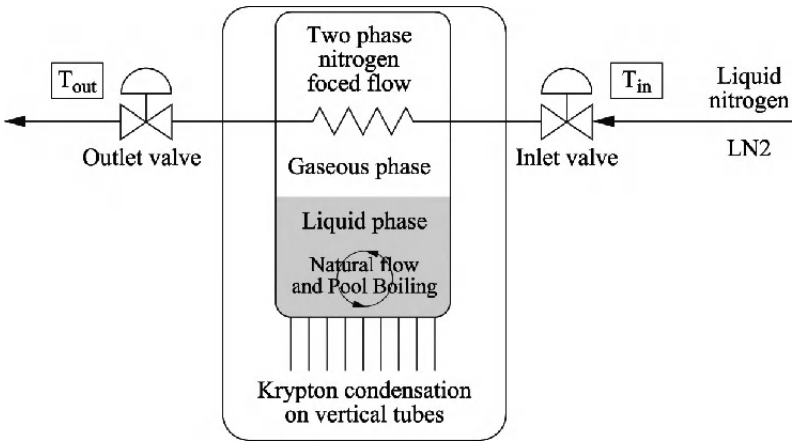


Figure 4.14. *Diagram of the argon condenser*

4.2.2. *Thermal model*

Once we have obtained the different usable coefficients, the system can be represented in equation form. We then have to think about various phenomena which will come into play in the simulation, and make a number of simplifying hypotheses.

4.2.2.1. *Thermodynamic equations*

To begin with, we need to make certain hypotheses about our system. The first is the presence of liquid nitrogen throughout the length of the exchanger. The nitrogen enters the exchanger with a mix quality of 0%, and exits with a variable quality which is usually around 10%. The percentage of mass evaporated is not constant, and depends on the values of the heat and mass flow on entering the exchanger. The cooling power is proportional to the latent heat of vaporization, because we are dealing with the phenomenon of Forced Convection Boiling. Ideally, in our first representation, the heat transfers will be modeled by convection. In the knowledge that all the systems are at saturation point, the simulation takes account of two parameters: the name of the saturating vapor x and the saturation temperature T. We can positively state that the evolution of the krypton and the argon is isochoric. Using the T-s curve for krypton, [JAC 97] (see Figure 4.15), we note that by following the isochors, it is possible to make a connection between the saturation temperature and the quality of the mixture. The system has only two actuators which can influence the pressure of the krypton.

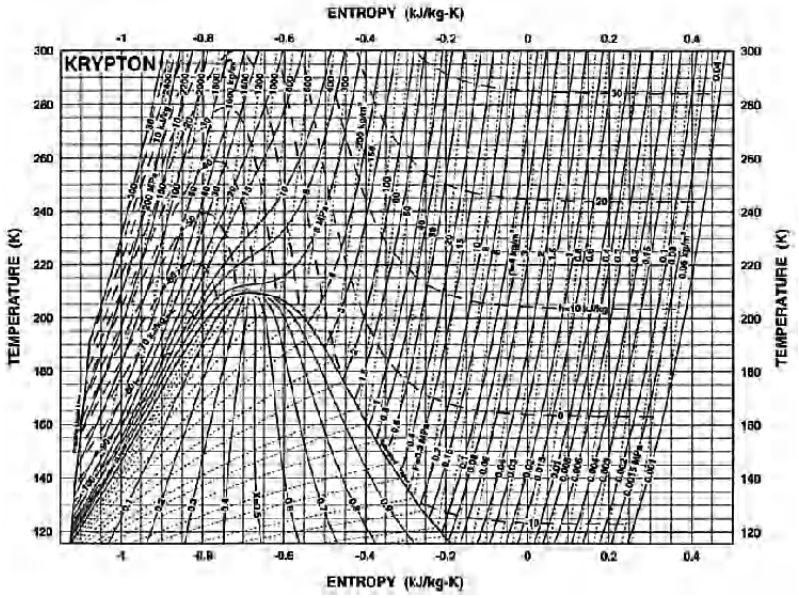


Figure 4.15. *Temperature/entropy diagram for krypton and visualization of isochors*

Thus, we have a valve to modify the flowrate at, therefore, evaporation in the nitrogen conduit, and another to adjust the temperature of the nitrogen and keep it at 88 K. This regulation is done by controlling the pressure of the krypton which, by way of data tables, can be used to work out the saturation temperature. We represent the first heat exchange between the krypton and argon as follows:

$$\delta Q \Rightarrow W(t) - h_{l.} S_1 . (T_{Kr} - T_{Ar}) = (\overline{C'_{Kr}} - \overline{C_{Kr}}) . x_{Kr} (\frac{dT_{Kr}}{dt}) + \overline{C_{Kr}} . \frac{dT_{Kr}}{dt} + (\frac{dx_{Kr}}{dt}) . L_{Kr} \quad [4.5]$$

The left-hand term represents, on the one hand, the "heat leaks", i.e. energy which penetrates into the cryostat by radiation, convection or conduction along the measuring cables found in the cryostat, and on the other hand the convection in the vicinity of the condenser along the pipes where the krypton condenses. The heat leaks result in a disturbance which we can estimate roughly, but which remains fairly unpredictable in terms of its variations. The right-hand term gives value to the variation in temperature and quality over time. The variation of x can be used to represent the evolution of the gaseous and liquid masses in the krypton circuit.

Similarly, we can express the variation in chemical proportion and temperature in the argon chamber:

$$h_{1.}S_1.(T_{Kr}-T_{Ar})-h_{2.}S_2.(T_{Ar}-T_{N2})=(\overline{C'_{Kr}}-\overline{C_{Kr}}).x_{Ar}(\frac{dT_{Ar}}{dt})+\overline{C_{Ar}}.\frac{dT_{Ar}}{dt}+(\frac{dx_{Ar}}{dt}).L_{Ar} \quad [4.6]$$

The equation is almost the same as for krypton, except that the heat leaks are not taken into account. A third equation completes the system – the nitrogen exchanger equation. At this stage, we consider the inlet and outlet temperature in the exchanger to be the same:

$$h_{2.}S_2.(T_{Ar}-T_{N2})=X_{Vap}(\frac{dm_{N2}}{dt}).L_{N2(l-g)} \quad [4.7]$$

The system equations obtained constitute a solid basis for the establishment of a state model which will be seen in the next section.

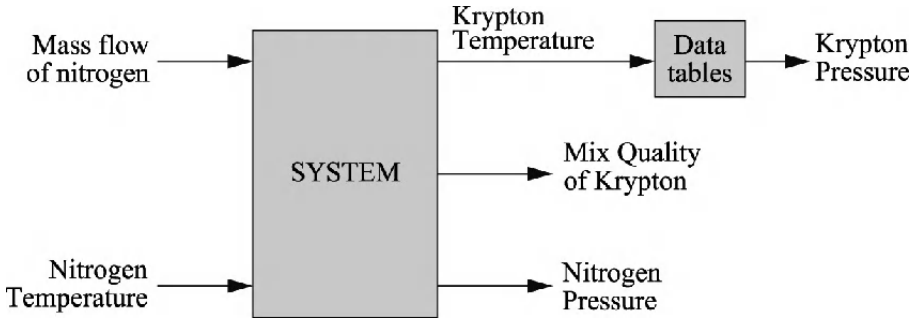


Figure 4.16. *Representation of the system in the form of an input/output diagram*

4.2.3. *The TDC (Time Delay Control) corrector: application to a liquid-krypton cryogenic exchanger*

4.2.3.1. *General operation*

In the context of our study, we propose to work with two regulations: that of the pressure of liquid krypton inside the calorimeter and that of the pressure of nitrogen inside the exchanger. The three cryogenic liquids – krypton, argon and nitrogen – are in their saturated state and therefore at an equilibrium state between liquid and vapor.

4.2.3.1.1. Description of the system

We have chosen to focus on the argon condenser of the calorimeter. It is this part which is responsible for the main form of regulation applied to maintaining the state of the krypton. In order to develop the appropriate control laws, we shall first focus on the working principle of the system and on the method currently used. To begin with, it is essential to explain the initial idea which represents the basis of the system. From the liquid/vapor equilibrium curves for the krypton and argon, shown in Figure 4.17, we can see that a certain argon pressure (*Ar P*) in the chamber corresponds to a saturation temperature of the bath of argon (*Ar T*). The process of liquefaction requires a certain temperature difference (D*T* = *Kr T* − *Ar T*) between the krypton and the argon, in order to condense the krypton by convection on the surface of the pipes located below the condenser. The saturated liquid krypton leaves the condenser at a temperature *Kr T* corresponding to a particular pressure *Kr P* of krypton in the cryostat.

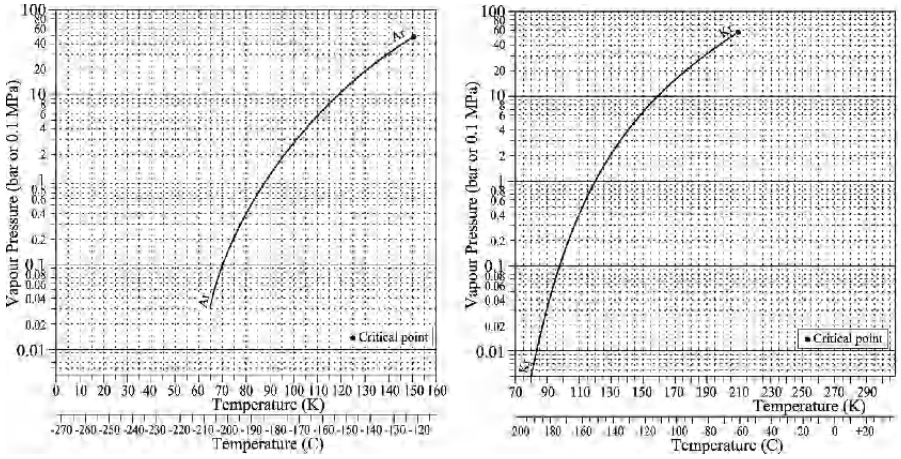


Figure 4.17. *Liquid/Vapor curves for argon and krypton*

If there is an increase in Kr T, and consequently an increase in Kr P, this DT will also be altered. In this case, the liquefaction ratio of the krypton increases, so Ar P must in turn decrease in order to compensate for this variation.

In order to obtain a stable krypton pressure, whilst taking account of the dynamic variations of the heat injections, we act on the argon pressure. A greater quantity of heat in the cryostat will increase the krypton pressure. The argon pressure is therefore decreased in order to regain the desired krypton pressure. The regulation

system envisaged is a two-stage control system. The first level consists of modeling the exchange system between the liquid nitrogen and argon to express the pressure of the argon bath as a function of the flowrate and temperature of the liquid nitrogen. The second level represents the argon/krypton system, wherein the krypton pressure is expressed as a function of the argon pressure. The two levels are interlinked and integrated into a higher regulation system which also includes regulation of the temperature of the nitrogen in the exchanger by means of the pressure, controlled by the outlet valve. Based on the variation in temperature and pressure in the calorimeter, we can determine the argon pressure which needs to be applied in the condenser, and therefore the flowrate and temperature of nitrogen to be reached. Figure 4.18 represents the system we intend to regulate as a set of inputs and outputs. The regulator will therefore give the position of the inlet and outlet valves, which can be expressed as the flowrate and temperature for the nitrogen.

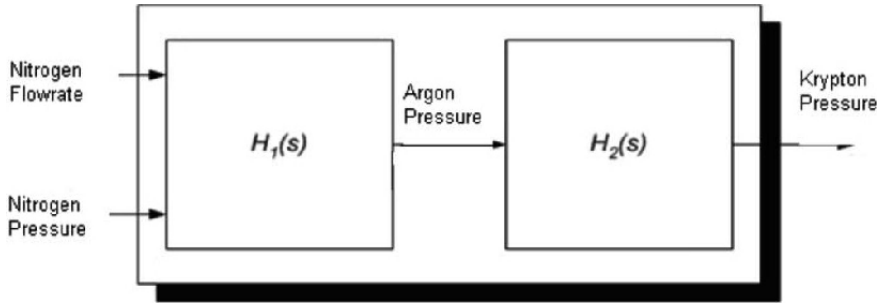


Figure 4.18. *Input/output diagram of the system*

The transfer function $H_1(s)$ represents the function $Ar\ P = f(N_2\ a, N_2\ T)$ where $N_2\ a$ is the mass flow of liquid nitrogen in the exchanger and $N_2\ T$ is the temperature of the nitrogen on entering the exchanger. The function $H_2(s)$ expresses $Ar\ P/Kr\ P$. The krypton temperature is closely linked to the pressure, and the goal is to maintain a constant temperature throughout the whole of the calorimeter. The krypton has to be liquefied with a degree of precision that means the liquid phase can be returned to the calorimeter without creating a temperature gradient.

4.2.3.1.2. Current form of control

Currently, the two regulatory operations are performed in separate loops. The first loop regulates the krypton pressure by altering the setting of the inlet valve; the second controls the nitrogen pressure in the exchanger, with feedback from the temperature sensors. The operation of the nitrogen cooling circuit is related

to the argon pressure set point (APSP). This point has to be calculated in relation to the heat injected into the krypton bath and therefore in relation to its own pressure. For this reason, it is necessary to find a direct relation between the pressure in the krypton bath and the APSP, and thus to regulate the valve of the nitrogen circuit on entering the condenser – both with a significant time constant in relation to the cryogenic systems. Similarly, we define a krypton pressure set point (KPSP), at 1.05 bar, which for its part is fixed and corresponds to the set reference point. Today, only PIDs are used for correction. The first regulation is done by two PIDs arranged in a cascade, which control the flow of liquid nitrogen and therefore the cold source.

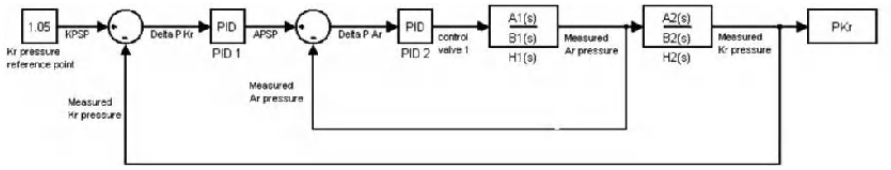


Figure 4.19. *Structure of the cascaded PID loops currently used for regulation*

4.2.3.2. *Objectives to be realized*

In order to maintain a constant nitrogen temperature in the exchanger and control the cooling power, the nitrogen pressure in the exchanger is also regulated by the outlet valve. The system takes account of the following regulatory points:

– the krypton pressure (KPSP) must remain constant, irrespectively of the injection of heat in the calorimeter. KPSP # 1.05 bar. Precision: ±0.01 bar;

– the corresponding argon pressure (APSP) is around 10 bars and must remain between the values of 9.8 bars – i.e. 0.1 K above the triple point of krypton (115.9 K) – and 13.0 bars – the point where the argon is at the same temperature as the krypton under nominal pressure;

– the nitrogen arrives at the level of the condenser at 4.5 bars (92 K). The regulation point can be set at 3.1 bars, i.e. 88 K (NPSP) (the triple point of argon is 83.8 K, i.e. 2.1 bars on the nitrogen curve).

The system manages two set reference points: KPSP and NPSP, two associated regulations, two adjustable variables, the mass flow and temperature of nitrogen, by way of two actuators – the inlet and outlet valves, whose aperture can be controlled directly.

4.2.3.3. *State equations*

The heat conservation equation for argon can be rewritten more precisely as follows:

$$\frac{dM_{g,Ar}}{dt} = \frac{q_{1c}}{h_{\lg,Ar}} - \frac{q_2}{h_{\lg,Ar}} \quad [4.8]$$

where:

– $M_{g,Ar}$ is the mass of the argon vapor;

– $\frac{q_{1c}}{h_{\lg,Ar}}$ represents the amount of liquid argon evaporating into gas form;

– $\frac{q_2}{h_{\lg,Ar}}$ represents the amount of argon vapor being liquefied;

By introducing the volume of argon vapor V_{Ar} and its density $\rho_{g\,Ar}$, which itself is a function of the argon temperature *Ar T,* we get:

$$V_{Ar} \cdot \frac{d\rho_{g,Ar}(T_{Ar})}{dT_{Ar}} \cdot \frac{dT_{Ar}}{dt} = \frac{q_{1c}}{h_{\lg,Ar}} - \frac{q_2}{h_{\lg,Ar}} \quad [4.9]$$

Hereafter, we shall use the coefficient k_1 such that:

$$k_1 = V_{Ar} \cdot \frac{d\rho_{g,Ar}(T_{Ar})}{dT_{Ar}} \quad [4.10]$$

Similarly, the same type of equation can be written for krypton:

$$\frac{dT_{Kr}}{dt} = \frac{q}{k2.h_{\lg,Ar}} - \frac{q_{1h}}{k2.h_{\lg,Ar}} \quad [4.11]$$

where:

– $\frac{q}{k2.h_{\lg,Ar}}$ represents the amount of liquid krypton evaporating into gas form;

$-\frac{q_{1h}}{k2.h_{\lg,Ar}}$ represents the amount of krypton vapor being liquefied;

which can be summarized in matrix form as follows:

$$\begin{bmatrix} T_{Kr} \\ T_{Ar} \end{bmatrix} = \begin{bmatrix} \frac{q_2 + q_d}{k2.h_{\lg,Kr}} - \frac{q_{1h}}{k2.h_{\lg,Kr}} \\ \frac{q_{1c}}{k1.h_{\lg,Ar}} - \frac{x_{Vap}.m_{l,N2}.h_{\lg,N2}}{k1.h_{\lg,Ar}} \end{bmatrix} \quad [4.12]$$

q_1 and q_{1h} can be written in accordance with the empirical Newton law as a function of the global heat transfer coefficient, the exchange surface and the difference in temperatures between the argon and krypton. On the basis of these equations, the state representation for the model of the exchanger is given below, with m_{lN2} representing the mass flow of nitrogen on entering the system and x_{vap} the percentage of liquid nitrogen being evaporated inside the heat exchanger. We suppose that the mix quality of the nitrogen is 0%:

$$X = A.X + B.U + w \quad [4.13]$$

where X is a state vector, $[T_{Kr}, T_{Ar}]^T$ the temperatures of the krypton and argon, U the control input and w represents the unknown terms *a priori*. This representation has the advantage of taking account of the imperfections of the model known *a priori*. A and B are described below:

$$A = \begin{bmatrix} \frac{-H_{1h}.S_{1h}}{k2.h_{\lg,Kr}} - \frac{H_{1h}.S_{1h}}{k2.h_{\lg,Kr}} \\ \frac{Hc.S_{1c}}{k1.h_{\lg,Ar}} - \frac{H_{1c}.S_{1c}}{k1.h_{\lg,Ar}} \end{bmatrix}$$

$$B = \begin{bmatrix} 0 \frac{x_{Vap}.h_{\lg,N2}}{k1.h_{\lg,Ar}} \end{bmatrix} \quad [4.14]$$

In analyzing matrix A, we note that one of the eigenvalues is null (an integrator system). When applying this formula digitally, it should be noted that the system can be controlled by approximating the numerical values of the physical constants contained in A and B.

4.2.3.4. *The TDC controller*

The Time Delay Control (TDC) technique was developed in the 1990s for systems of partial differential equations (PDE) with unknown terms. Put forward by Youcef-Toumi and Ito [ROG 08], it has been able to been applied to nonlinear systems on two main conditions: that all the state variables be measurable and that any continuous signal be constant over a sufficiently reduced period. These terms are estimated on the basis of past controls and measurements of the state variables. The principle of estimation of unknown terms is given by the below equation as a function of the delay operator D as follows:

$$\widehat{w}(t) = D.(\widehat{X}(t) - AX(t) - BU(t)) \qquad [4.15]$$

The corresponding Laplace transform for the delay is *e-Ls*, which can be estimated using the Padé approximation. According to our model, we shall take a second order, as indicated in the equation below. The resulting transfer function must be stable, causal and exhibit a static unitary gain:

$$\widehat{w}(t) = \frac{d_0 + p_1 s}{d_0 + d_1 s + d_2{}^2 s^2} \qquad [4.16]$$

In parallel, we choose a linear reference model for the trajectory to be followed in accordance with the following equation:

$$u = B^{+}[B_r c + (Ar - A)X - \widehat{w}] \qquad [4.17]$$

where the matrices $r\,A$ and $r\,B$ are constant, $r\,A$ is stable and the pair ($r\,A$, $r\,B$) is controllable. B^{+} is the pseudo-inverse of B: it is the reference vector, containing the reference for the static temperature of the krypton and the reference for the dynamic temperature of the argon, directly related to the control vector u.

In the Laplacian domain and with account taken of the Padé approximation, we get:

$$u(s) = B^{+}[c_1.(B_r c + ArX) - AX - c_2.X] \qquad [4.18]$$

where c_1 and c_2 are obtained by way of the Padé approximation after simplification.

As shown in Figure 4.20, the corrector uses the state variables which must be available physically or reconstructed by an observer with the help of the model.

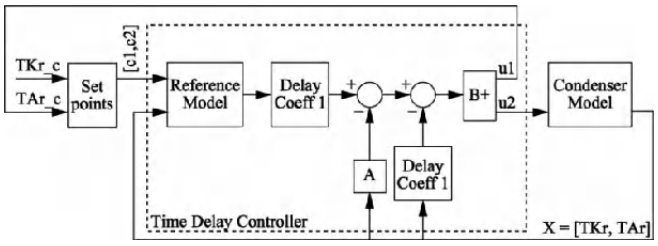


Figure 4.20. *Block diagram of the TDC controller and of the model of the process*

RESULTS.–

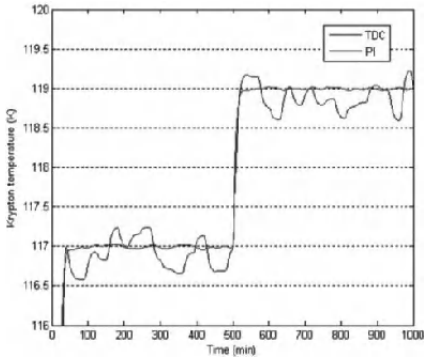


Figure 4.21. *Comparison between the PI and TDC controllers with variation of set reference point for small variations*

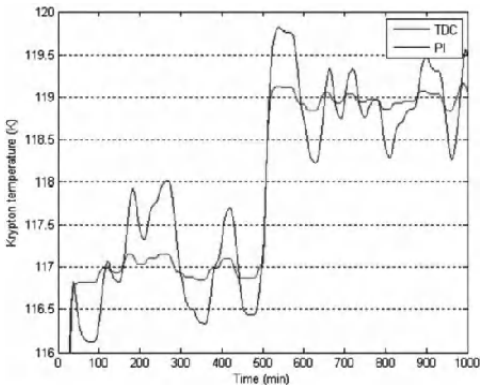


Figure 4.22. *Comparison between the PI and TDC controllers with variation of set reference point for large variations*

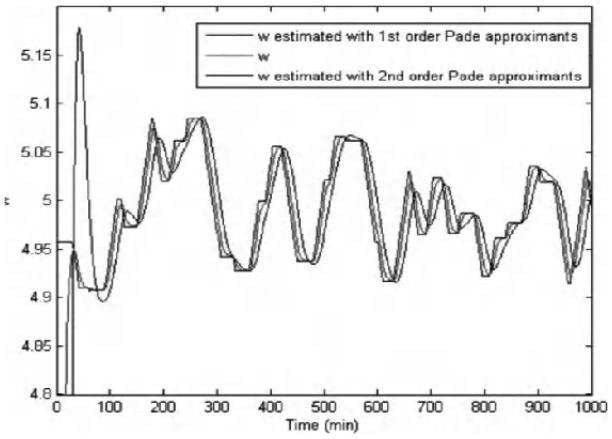


Figure 4.23. *Estimation of unknown parameters related to heat exchanges on entering the system*

4.3. Modeling and control of the cryogenics of the ATLAS experiment at CERN

4.3.1. *Context and objectives of the study*

The ATLAS experiment is one of the two large particle physics experiments intended to exploit the potential of the new large proton collider – proton LHC, which came into operation at CERN in 2007.

The domain of energy achieved (14 TeV) and the luminosity (10^{34} $cm^{-2}.s^{-1}$) open up opportunities for unparalleled discoveries – in particular in the area of the mechanism for electroweak symmetry breaking. The Higgs boson and/or supersymmetric particles are the spectacular manifestations expected on the basis of the underlying mechanisms. The ATLAS detector, which covers the whole of the solid angle around the point of collision, uses innovative techniques to exploit this potential. Muon detection is done by using a massive superconductive toroidal magnet. The electrons, photons and jets are identified and measured with a liquid argon precision calorimeter and the traces close to the vertex are measured by a set of layers of semi-conductor detectors, divided finely into pixels or "strips". The collaborative team that built this detector includes around 1800 researchers and engineers distributed among more than 150 laboratories on all continents. France (IN2P3 and CEA) is particularly involved with the liquid argon calorimeter. This calorimeter has been the focus of a highly active R&D program since 1990, to adapt the technique (noble-liquid ionization chamber, which is intrinsically rather slow) to the luminosity of the LHC, where collisions take place, creating an enormous

dataflow with a trigger system capable of selecting 100 interesting events per second out of one billion.

However, these technologies prove to be sensitive to inconsistencies in temperature – particularly in terms of the sensitive part of the modules (–2% per K). In order to obtain homogeneous responses in space and a stability over time to a few ‰ (PPT), it is crucial that the temperature in the sensitive modules be homogeneous and stable to a fraction of a degree (0.5 K).

The study we are presenting results from a collaboration between the LIMSI (*Laboratoire d'informatique pour la mécanique et les sciences de l'ingénieur* – Computer Science Laboratory for Mechanics and Engineering Sciences) and the LAL (*Laboratoire de l'accelerateur linéaire* – Linear Accelerator Laboratory). It involves carrying out 3D digital simulation of the heat transfers in the "barrel" section of the liquid argon calorimeter (LAC) in the ATLAS experiment. These transfers essentially take place by conduction in solids and by natural convective motions in the liquid phase (argon), which may be turbulent. The non-stationary nature of the flow and the different regimes that it may adopt (laminar-turbulent) caused us to choose a Large-Scale Modeling (LSM) approach.

The geometry of the LAC and the physical properties of the different elements which make it up imply a great disparity of scale on the evolution of the heat transfers, both on the spatial and temporal planes. Consequently, this study requires considerable resources in terms of computing, with the spatial resolution requiring around ten million meshes and the number of temporal iterations being around two millions. For this reason, the calculations were performed at the heart of the computer resources of IDRIS, on the NEC-SX5 vector supercomputer. This study is in line with the major realizations in digital calculations.

The primary objective of this study is to establish a map of the temperature in the LAC in order to check that the temperature difference in the modules does not exceed the set threshold of 0.5K in the sensitive part. Furthermore, this study should help improve our knowledge of the three heat transfers that take place within the LAC, thereby enabling us to better control the distribution of heat. As part of this report, we present a description of the LAC on the basis of the data provided by the LAL (Linear Accelerator Laboratory) [BRE 96] and the different elements to arrive at a simplified representation, in order to facilitate digital simulation. The physical phenomena involved in the heat transfers are explained, as is the way in which they are taken into account in the digital simulation. We also define the configuration of the simulation and the way in which we have obtained the results. We lead a discussion about the description of the average flow and the temperature fields obtained.

4.3.2. *Process of identification of cryogenic systems*

4.3.2.1. *Presentation of the ATLAS LAr calorimeter system*

In order to exploit the capabilities of the future accelerator at CERN (the LHC), with collisions of protons and heavy ions, four particle detectors were installed in the underground caverns. ATLAS is the largest of these detectors, including a complex cryogenic system for superconductivity of the magnets and thermal stability of the liquid argon. The liquid argon calorimeter is installed in three cryostats with a total volume of 83 m^3, cooled to a temperature of approximately 87 K.

The cryogenic systems at ATLAS represent different cryogenic circuits using helium, nitrogen and argon as fluids. The experimental system chosen is the central part of the ATLAS detector: the liquid argon calorimeter. It comprises a central (barrel) cryostat and two end cap cryostats, respectively containing 203 T and 2×269 T of cold mass. They are placed in three independent cryostats with 45 m^3 and 2×19 m^3 of liquid argon at 88 K. The cooling system for the central bath of liquid argon is made up of seven liquid nitrogen heat exchangers. The flowrate and pressure of nitrogen in each exchanger are regulated by valves at the inlet and outlet of the exchanger.

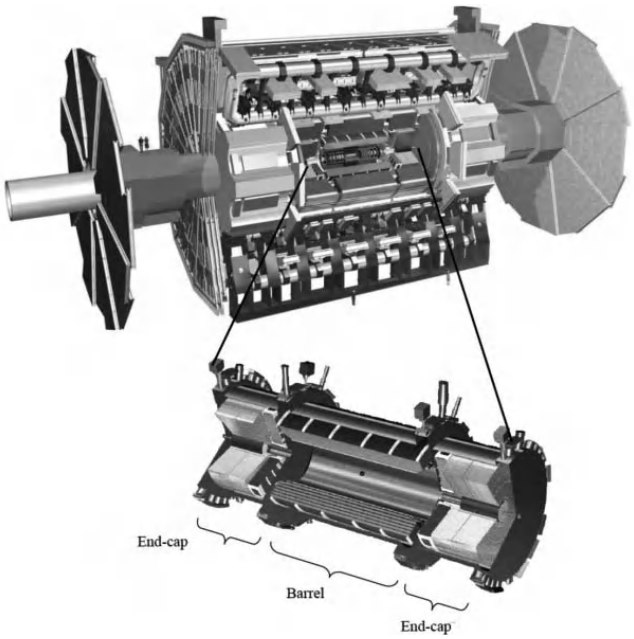


Figure 4.24. *ATLAS calorimeter*

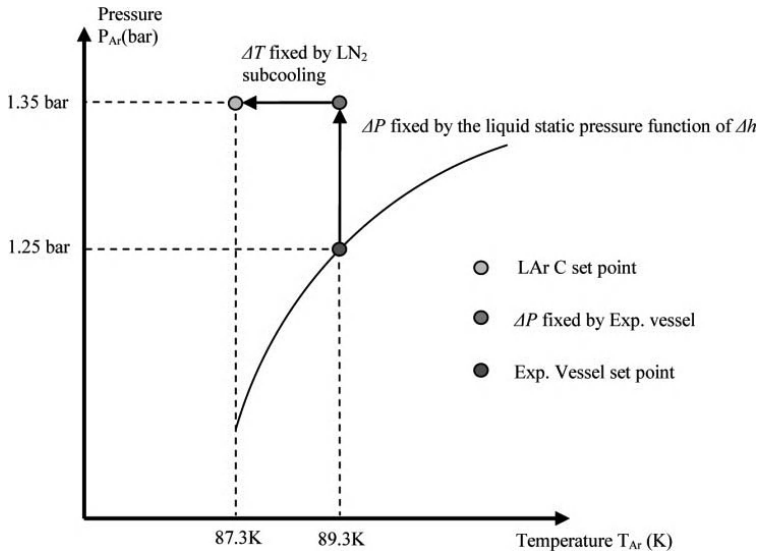


Figure 4.25. *Pressure/temperature diagram for the bath of liquid argon*

4.3.2.1.1. Cryogenic installation

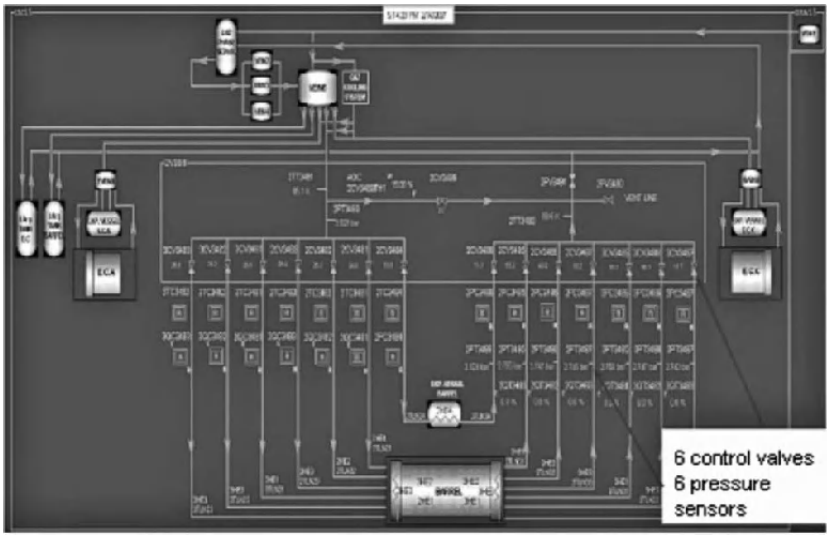


Figure 4.26. *PVSS panel – distribution of nitrogen to the argon barrel*

Input variables for the procedure:

Measurements and LAr Calorimeter actuators	Type of measurement	Alias (8 letters max)	Address
QXKBA_A_2CV3488	Control Valve HE Exp Vessel	2CV3488	%MW7669
QXKBA_A_2CV3485	Control Valve 2HE1	2CV3485	%MW7645
QXKBA_A_2CV3486	Control Valve 2HE2	2CV3486	%MW7653
QXKBA_A_2CV3487	Control Valve 2HE3	2CV3487	%MW7661
QXKBA_A_3CV3485	Control Valve 3HE1	3CV3485	%MW7709
QXKBA_A_3CV3486	Control Valve 3HE2	3CV3486	%MW7717
QXKBA_A_3CV3487	Control Valve 3HE3	3CV3487	%MW7725

Table 4.2. *Input variables*

Output variables for the procedure:

Measurements and LAr Calorimeter actuators	Type of measurement	Alias (8 letters max)	Address
QXKBA_A_2PT3488	Pressure transmitter	2PT3488	%MW4133
QXKBA_A_2PT3485	Pressure transmitter	2PT3485	%MW4121
QXKBA_A_2PT3486	Pressure transmitter	2PT3486	%MW4125
QXKBA_A_2PT3487	Pressure transmitter	2PT3487	%MW4129
QXKBA_A_3PT3485	Pressure transmitter	3PT3485	%MW4137
QXKBA_A_3PT3486	Pressure transmitter	3PT3486	%MW4149
QXKBA_A_3PT3487	Pressure transmitter	3PT3487	%MW4153

Table 4.3. *Output variables*

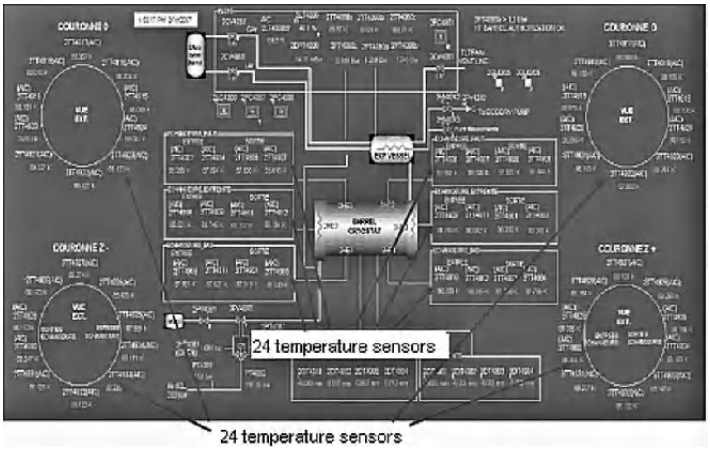


Figure 4.27. *PVSS panel – temperature sensors for the argon barrel*

Output variables for the procedure:

– Temperature of the bath of argon:

Measurements and LAr Calorimeter actuators	Type of measurement	Alias (8 letters max)	Address
QXKBA_A_2TT4915AIC	Detector Temperature 1	2TT4915	%MW5909
QXKBA_A_2TT4916AIC	Detector Temperature 2	2TT4916	%MW5913
QXKBA_A_2TT4917AIC	Detector Temperature 3	2TT4917	%MW5917
QXKBA_A_2TT4918AIC	Detector Temperature 4	2TT4918	%MW5921
QXKBA_A_2TT4919AIC	Detector Temperature 5	2TT4919	%MW5925
QXKBA_A_2TT4920AIC	Detector Temperature 6	2TT4920	%MW5929
QXKBA_A_2TT4921AIC	Detector Temperature 7	2TT4921	%MW5933
QXKBA_A_2TT4922AIC	Detector Temperature 8	2TT4922	%MW5937
QXKBA_A_2TT4923AIC	Detector Temperature 9	2TT4923	%MW5941
QXKBA_A_2TT4924AIC	Detector Temperature 10	2TT4924	%MW5945
QXKBA_A_2TT4925AIC	Detector Temperature 11	2TT4925	%MW5949
QXKBA_A_2TT4926AIC	Detector Temperature 12	2TT4926	%MW5953
QXKBA_A_2TT4927AIC	Detector Temperature 13	2TT4927	%MW5957
QXKBA_A_2TT4928AIC	Detector Temperature 14	2TT4928	%MW5961
QXKBA_A_2TT4929AIC	Detector Temperature 15	2TT4929	%MW5965
QXKBA_A_2TT4930AIC	Detector Temperature 16	2TT4930	%MW5969
QXKBA_A_2TT4931AIC	Detector Temperature 17	2TT4931	%MW5973
QXKBA_A_2TT4932AIC	Detector Temperature 18	2TT4932	%MW5977
QXKBA_A_2TT4933AIC	Detector Temperature 19	2TT4933	%MW5981
QXKBA_A_2TT4934AIC	Detector Temperature 20	2TT4934	%MW5985
QXKBA_A_3TT4915AIC	Detector Temperature 21	3TT4915	%MW6037
QXKBA_A_3TT4916AIC	Detector Temperature 22	3TT4916	%MW6041
QXKBA_A_3TT4917AIC	Detector Temperature 23	3TT4917	%MW6045
QXKBA_A_3TT4918AIC	Detector Temperature 24	3TT4918	%MW6049
QXKBA_A_3TT4919AIC	Detector Temperature 25	3TT4919	%MW6053
QXKBA_A_3TT4920AIC	Detector Temperature 26	3TT4920	%MW6057
QXKBA_A_3TT4921AIC	Detector Temperature 27	3TT4921	%MW6061
QXKBA_A_3TT4922AIC	Detector Temperature 28	3TT4922	%MW6065
QXKBA_A_3TT4923AIC	Detector Temperature 29	3TT4923	%MW6069
QXKBA_A_3TT4924AIC	Detector Temperature 30	3TT4924	%MW6073
QXKBA_A_3TT4925AIC	Detector Temperature 31	3TT4925	%MW6077

Table 4.4. *Temperatures of the bath of argon*

QXKBA_A_3TT4926AIC	Detector Temperature 32	3TT4926	%MW6081
QXKBA_A_3TT4927AIC	Detector Temperature 33	3TT4927	%MW6085
QXKBA_A_3TT4928AIC	Detector Temperature 34	3TT4928	%MW6089
QXKBA_A_3TT4929AIC	Detector Temperature 35	3TT4929	%MW6093
QXKBA_A_3TT4930AIC	Detector Temperature 36	3TT4930	%MW6097
QXKBA_A_3TT4931AIC	Detector Temperature 37	3TT4931	%MW6101
QXKBA_A_3TT4932AIC	Detector Temperature 38	3TT4932	%MW6105
QXKBA_A_3TT4933AIC	Detector Temperature 39	3TT4933	%MW6109
QXKBA_A_3TT4934AIC	Detector Temperature 40	3TT4934	%MW6113

Table 4.4. *(Continued) Temperatures of the bath of argon*

– Temperature of the heat exchangers:

Measurements and LAr Calorimeter actuators	Type of measurement	Alias (8 letters max)	Address
QXKBA_A_2TT4909	Detector Temperature 2HE1	2TT4909	%MW5893
QXKBA_A_2TT4911	Detector Temperature 2HE1	2TT4911	%MW5901
QXKBA_A_2TT4901	Detector Temperature 2HE1	2TT4901	%MW5861
QXKBA_A_2TT4903	Detector Temperature 2HE1	2TT4903	%MW5869
QXKBA_A_2TT4902	Detector Temperature 2HE2	2TT4902	%MW5865
QXKBA_A_2TT4904	Detector Temperature 2HE2	2TT4904	%MW5873
QXKBA_A_2TT4905	Detector Temperature 2HE2	2TT4905	%MW5877
QXKBA_A_2TT4907	Detector Temperature 2HE2	2TT4907	%MW5885
QXKBA_A_2TT4906	Detector Temperature 2HE3	2TT4906	%MW5881
QXKBA_A_2TT4908	Detector Temperature 2HE3	2TT4908	%MW5881
QXKBA_A_2TT4910	Detector Temperature 2HE3	2TT4910	%MW5897
QXKBA_A_2TT4912	Detector Temperature 2HE3	2TT4912	%MW5905
QXKBA_A_3TT4910	Detector Temperature 3HE1	3TT4910	%MW6025
QXKBA_A_3TT4912	Detector Temperature 3HE1	3TT4912	%MW6033
QXKBA_A_3TT4902	Detector Temperature 3HE1	3TT4902	%MW5993
QXKBA_A_3TT4904	Detector Temperature 3HE1	3TT4904	%MW6001
QXKBA_A_3TT4905	Detector Temperature 3HE2	3TT4905	%MW6005
QXKBA_A_3TT4907	Detector Temperature 3HE2	3TT4907	%MW6013
QXKBA_A_3TT4906	Detector Temperature 3HE2	3TT4906	%MW6009
QXKBA_A_3TT4908	Detector Temperature 3HE2	3TT4908	%MW6017
QXKBA_A_3TT4909	Detector Temperature 3HE3	3TT4909	%MW6021
QXKBA_A_3TT4911	Detector Temperature 3HE3	3TT4911	%MW6029
QXKBA_A_3TT4901	Detector Temperature 3HE3	3TT4901	%MW5989
QXKBA_A_3TT4903	Detector Temperature 3HE3	3TT4903	%MW5987

Table 4.5. *Temperatures of the heat exchangers*

4.3.2.1.2. Measuring method

The measurements will be taken in two different ways, depending on whether the data are needed for mono-variable or multi-variable identification.

In mono-variable identification, we use the DataStore program.

For multi-variable identification, the number of variables is too great to be dealt with by the same program. We therefore use the Oracle database from CERN's intranet, which constantly records the values from the sensors in the installations.

All the data are stored in the form of an Excel spreadsheet, and can be worked with directly by the programs Optireg or Matlab®.

4.3.2.1.3. Structure of the system identified

The whole of the system can be broken down into six mono-variable subsystems with a PI regulator and a multi-variable system with a regulator.

The whole system has a structure of interlocking regulators.

The mono-variable system comprises a PI regulator and a regulated valve.

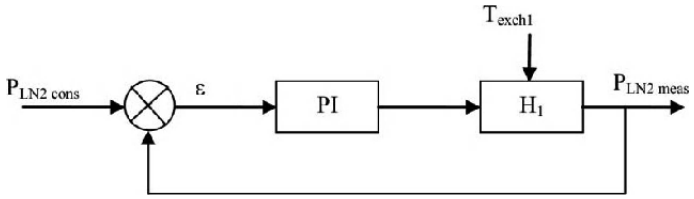


Figure 4.28. *Block diagram of the PI corrector and the procedure*

Each mono-variable system is equivalent to Figure 4.29.

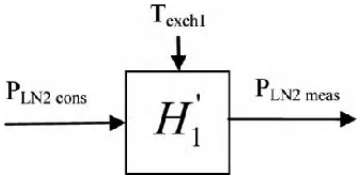


Figure 4.29. *Block diagram of a mono-variable process*

The whole system can be represented as shown in Figure 4.30.

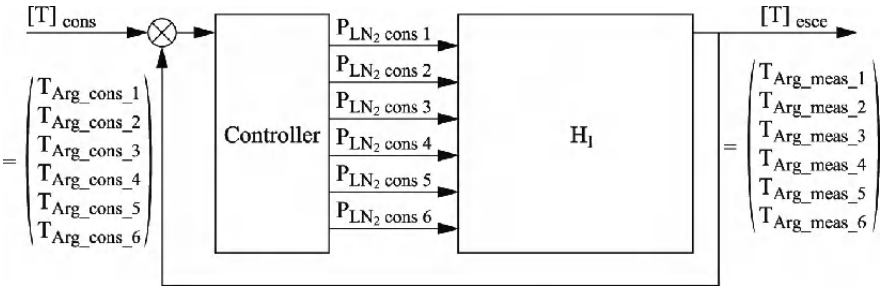


Figure 4.30. *Block diagram of the whole system*

Breakdown of the whole system:

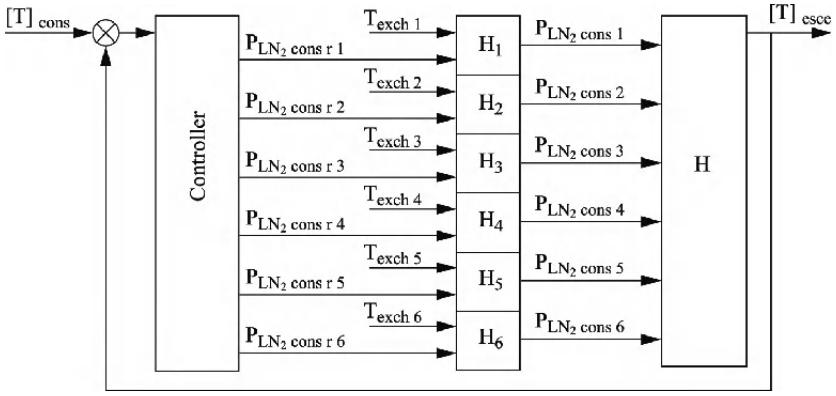


Figure 4.31. *Block diagram of the breakdown of the whole system*

4.3.3. *Experimental protocol of parametric identification*

4.3.3.1. *Parametric identification*

4.3.3.1.1. Principle and objectives

When designing a control system, it is necessary to have a model at your disposal to represent the dynamic behavior of the process to be controlled.

Based on the laws of physics, mechanics, hydraulics, etc., it is possible to obtain a model in the form of a set of differential algebraic equations (DAEs). This model, as it is based on the knowledge that we have of the process and of the laws governing it, is called a knowledge model, and the approach by which it is created is called modeling.

However, this model can rapidly become highly complex, and therefore unusable as such for the design of a control law. In this case, a modeling approach is essentially intended to simulate the process and study its behavior.

An alternative approach to modeling is identification. This is based on the determination of a mathematical model on the basis of measurements carried out on the process. The model obtained may be an empirical model whose coefficients have no physical meaning *a priori*. The parameters of a transfer function, for instance, rarely represent physical values.

It should be noted that these two approaches for obtaining a model of a dynamic system are not necessarily antinomic. An identification approach could be used to complement a modeling approach. The aim is then to determine certain physical parameters which are not available.

4.3.3.1.2. Procedure of identification

In order to obtain a consistent model, it is important to stimulate the process with all the frequencies in its operational range. The input signal applied must therefore be frequency-rich (that is, have a broad spectrum). In general, we tend to apply a pseudo-random binary sequence (PRBS).

When the system has numerous inputs/numerous outputs, it is important to apply decorrelated signals so as not to introduce bias into the identification process. One common idea, involving stimulating the inputs one after another, is a bad method because it introduces a bias of identification and does not account for the normal operation of the system. It is important to respect a rigorous procedure in order to identify a process:

– determination of a test protocol: statistical properties of the input signals in order to scan all the interesting frequencies, the signal-to-noise ratio must be high enough and the number of measurement points must be signification for the test (>1,000);

– determination of the structure of the model: type of model, order and delay;

– identification: choice of an algorithm to find the model by minimizing the disparity between the measurements and the model – usually an algorithm based on the least squares method (LS, RLS, RELS);

– validation of the model: numerous rounds of testing to verify its quality. It is necessary for this stage to use different measurements from those used during the identification process.

Other approaches are also possible – particularly those using the subspace matrices of a system – but these are less effective than the parametric identification method for nonlinear systems.

This technique can therefore easily give us a less "theoretical" model, and help improve the results of the control or prediction (for share prices in an economic system, for instance).

Matlab® and Scilab toolkits are available for algorithm-solving (such as ARMAX, for example).

4.3.3.1.3. The different types of models

The principle of a parametric identification is to extrapolate a mathematical model on the basis of observations.

The model must be able to compute the output from a process y at any time t if the initial conditions of the system are known. For this purpose, we can use the present and previous values of input ($u(t)$, $u(t\text{-}1)$, …) and the previous output values ($y(t\text{-}1)$, $y(t\text{-}2)$, …) in the case of a regressive model.

Nevertheless, it is important to have prior knowledge of the system in order to be able to choose an appropriate type of model:

– single-input/single-output (SISO) or multiple-input/multiple-output (MIMO) model;

– linear or nonlinear model (in this case, what is nonlinear, and on what basis?);

– continuous or discrete model;

– regressive or independent model: for a regressive model, the output at time t, $y(t)$, depends on the previous moments ($y(t\text{-}i)$);

– stochastic or deterministic model.

In general, the model is represented in the form of a transfer function using the Z transform.

Identification requires prior knowledge of the structure of the model, in order to identify different parameters within that structure. The following are the three most widely-used model structures:

– ARX (Auto Regressive model with eXternal inputs): this is an auto-regressive model which includes inputs *u(t)* and a white noise $\zeta(t)$ with zero average. In addition, the model includes a pure time delay of k timesteps. If the system is sampled with a sampling frequency T then the delay will be $k \times T$.

In temporal form:

$$Y(t) = B.\left[u(t-k), u(t-1-k), \ldots\right]^T - A.\left[y(t-1), y(t-2), \ldots\right]^T + \\ + A\left[\zeta(t), \zeta(t-1), \ldots\right] \quad [4.19]$$

In a discrete space using the Z transform:

$$y(z) = z^{-k}.\frac{B(z^{-1})}{A(z^{-1})}.u(z) + \zeta(z) \quad [4.20]$$

– ARMAX (Auto Regressive Moving Average with eXternal inputs): this model has all the attributes of the ARX model, but also includes a transfer function with a moving average for the white noise. In general, white noise can be used to model disturbances that cannot be measured in a model. However, these non-measurable disturbances (heat fluctuations, ground vibrations, etc.) rarely have a zero average and can also fit into a model:

$$y(z) = z^{-k}.\frac{B(z^{-1})}{A(z^{-1})}.u(z) + \frac{T(z^{-1})}{A(z^{-1})}\zeta(z) \quad [4.21]$$

– ARIMAX (or CARIMA): in the ARIMAX (Auto Regressive Integrated Moving Average with eXternal inputs) model, the noise model is directly integrated:

$$y(z) = z^{-k}.\frac{B(q^{-z})}{A(q^{-z})}.u(z) + \frac{T(z^{-1})}{(1-z^{-1}).A(z^{-1})}\zeta(z) \quad [4.22]$$

4.3.3.2. *Least squares method*

4.3.3.2.1. LS algorithm

The least squares method can be used to compare experimental data, which are usually tainted with measuring errors, to a mathematical model which is supposed to describe these data.

Let us suppose that the data recorded from the system we are seeking to identify are described by the equation:

$$y(k) = \varphi(k)^T \theta_0 + e(k) \quad [4.23]$$

where θ_0 represents the exact value of the parameter vector θ and e(k) is a white noise with zero average and variance λ^2.

In this case:

$-\hat{\theta}$, given by $\hat{\theta} = (\boldsymbol{\Phi}^T \boldsymbol{\Phi})^{-1} \boldsymbol{\Phi}^T Y$ is a non-biased estimation of θ_0;

– the covariance matrix of $\hat{\theta}$ is given by: $\text{cov}(\hat{\theta}) = \lambda^2 (\boldsymbol{\Phi}^T \boldsymbol{\Phi})^{-1}$;

– an estimation of λ^2 is given by: $\lambda^2 = \dfrac{2V(\hat{\theta})}{N-n}$.

Note that the covariance matrix of $\hat{\theta}$ is proportional to the variance of the signal $e(k)$. Furthermore, the matrix $\boldsymbol{\Phi}^T \boldsymbol{\Phi}$ must be invertible. An inapt measuring protocol may be the cause of non-verification of this condition.

An ARX-type model is a model corresponding to the structure defined in equation [4.23]. We can conclude from the previous section that the least squares algorithm gives a non-biased parametric estimation in the case of an ARX-type process, often also referred to as an LS process for this reason.

4.3.3.2.2. RLS algorithm

A recursive algorithm is an algorithm that can be used to obtain $\hat{\theta}(k)$ on the basis of $\hat{\theta}(k-1)$ and the information available at period k.

The conventional form of the Recursive Least Squares (RLS) algorithm is:

$$\begin{cases} \hat{\theta}(k) = \hat{\theta}(k-1) + K(k).\varepsilon(k) \\ \varepsilon(k) = (y(k) - \hat{\theta}(k-1)^T .\varphi(k)) \\ K(k) = \dfrac{P(k-1).\varphi(k)}{1+\varphi(k)^T P(k-1)\varphi(k)} \\ P(k) = P(k-1) - \dfrac{P(k-1)\varphi(k)\varphi(k)^T P(k-1)}{1+\varphi(k)^T P(k-1)\varphi(k)} \end{cases} \qquad [4.24]$$

The term $\varepsilon(k) = (y(k) - \hat{\theta}(k-1)^T .\varphi(k))$ represents the disparity between the measured output $y(k)$ and the estimation of the output given by the model $\hat{\theta}(k-1)^T .\varphi(k)$, which is notated as $\hat{y}(k)$.

4.3.3.2.3. RLS algorithm with forgetting factor

In the previous algorithm, the adaptation gain matrix is shrinking, and will tend toward a null matrix. In order to deal with this issue, modifications were made to the least squares algorithm. These modifications are strictly necessary if the objective is a real-time identification of those parameters of a process which are likely to change over the course of the system's operation. This situation arises, particularly, when implementing an adaptive control law.

In the RLS algorithm with forgetting factor, we minimize a weighted sum of the squares of the prediction errors:

$$V_\lambda(\theta,k)=\sum_{t=1}^{k}\lambda^{k-t}\varepsilon(t)^2 \qquad [4.25]$$

The recursive least squares algorithm with fixed forgetting factor takes the following form:

$$\begin{cases}\hat{\theta}(k)=\hat{\theta}(k-1)+L(k)(y(k)-\hat{\theta}(k-1)^T\varphi(k)\\ L(k)=\dfrac{P(k-1)\varphi(k)}{1+\varphi(k)^T P(k-1)\varphi(k)}\\ P(k)=\dfrac{1}{\lambda}\left\{P(k-1)-\dfrac{P(k-1)\varphi(k)\varphi(k)^T P(k-1)}{\lambda+\varphi(k)^T P(k-1)\varphi(k)}\right\}\end{cases} \qquad [4.26]$$

4.3.3.2.4. Recursive Extended Least Squares (RELS) algorithm

The method was developed in order to be able to identify processes described by an ARMAX-type model.

$$\begin{cases}\hat{\theta}(k)=\hat{\theta}(k-1)+K(k)\varepsilon(k)\\ \varepsilon(k)=y(k)-\hat{\theta}(k-1)^T\varphi(k)=\hat{\varepsilon}(k)\\ \varphi(k)^T=\left[-y(k-1),\ldots,-y(k-n),u(k-1),\ldots,u(k-m),\hat{\varepsilon}(k-1),\ldots,\hat{\varepsilon}(k-r)\right]\\ K(k)=\dfrac{P(k-1)\varphi(k)}{1+\varphi(k)^T P(k-1)\varphi(k)}\\ P(k)=P(k-1)-\dfrac{P(k-1)\varphi(k)\varphi(k)^T P(k-1)}{1+\varphi(k)^T P(k-1)\varphi(k)}\end{cases} \qquad [4.27]$$

or, if a set value for the forgetting factor is included:

$$P(k) = \frac{1}{\lambda}\left\{ P(k-1) - \frac{P(k-1)\varphi(k)\varphi(k)^T P(k-1)}{\lambda + \varphi(k)P(k-1)\varphi(k)} \right\} \quad [4.28]$$

4.3.4. *Mono-variable system*

4.3.4.1. *Objectives*

The objectives are to:

– identify the transfer functions H_1, H_2…, H_6;

– deduce the associated PI regulators;

– calculate the equivalent transfer functions H'_1, H'_2…, H'_6.

4.3.4.2. *Test protocol*

In order to identify the transfer functions H_1, H_2…, H_6, we carry out open-loop tests. The six systems for regulating the pressure in the heat exchangers are independent. It is therefore possible to carry out disturbances manually in these systems one after another. It should be noted, however, that the six heat exchanger systems can be grouped into three groups on the basis of their geographic location on the barrel:

– 1st group for the two exchangers located on the upper part of the calorimeter;

– 2nd group for the two exchangers located on the lower part;

– 3rd group for the two exchangers around the sides of the calorimeter.

Thus, we shall suppose we have three systems to identify in order to be able to model all of the heat exchangers of the liquid argon calorimeter together.

Our prior knowledge of the system leads us to perform a first-order identification of a system. Thus, our tests will consist of disturbing the system by staggered commands to open or close the valves.

When one valve is being tested, the actual regulator associated with that valve is disconnected. The other five valves remain active and continue to regulate the pressure.

The inlet valves to the heat exchangers are set at their nominal values for all of the tests.

We also record the four temperatures on each exchanger in order to model the disturbances.

The limits to the disturbance of the system which must not be surpassed are determined by the system's cryogenic properties. Indeed, the stability of the bath of liquid argon must be maintained. Thus, the pressure limits in the exchangers are:

2.6 bar < P < 2.9 bar.

For the valves, the limits of aperture are:

Valves	L_{min}	L_{max}
2CV3485	40%	60%
2CV3486	50%	70%
2CV3487	15%	30%
3CV3485	40%	60%
3CV3486	50%	70%
3CV3487	15%	30%

Table 4.6. *Limits of aperture of the valves*

We use the DataStore program to record the data from the tests, and the program PVSS to modify the aperture of the valves.

For the identification of the system, we shall use the program Optireg (marketed by Schneider Electric) and Matlab®. The two results will be compared and evaluated on the basis of their consistency with the real-world system.

4.3.4.2.1. The data server DataStore

Datastore is a software package developed by Schneider Electric, which uses an OFS server (OPC factory server) to handle data exchanges with programmable logic controllers and which periodically records the values of the variables in a file that can be used in Excel.

Configuration: for the exchanger 2HE1.

Type of PLC:	Quantum Unity
Type of connection:	TCP/IP
IPC address of target PLC:	172.18.34.171
Refresh period:	30s
Number of variables:	6

	Alias	Symbol	Type	Unit	Min scale	Max scale
Variable 1	2CV3485	%MF7645	REAL	%		
Variable 2	2PT3485	%MF4121	REAL	bar		
Variable 3	2TT4909	%MF5893	REAL	K		
Variable 4	2TT4911	%MF5901	REAL	K		
Variable 4	2TT4901	%MF5861	REAL	K		
Variable 5	2TT4903	%MF5869	REAL	K		

Table 4.7. *Excel spreadsheet generated by DataStore*

4.3.4.2.2. Data fusion

For each heat exchanger, four temperature sensors are installed to check its temperature distribution. It is necessary to fuse these data to create only one representative temperature T_{exch_x} for each exchanger for the mono-variable model.

In the case of heat exchangers, we shall use an average of the output temperatures to determine the representative temperature.

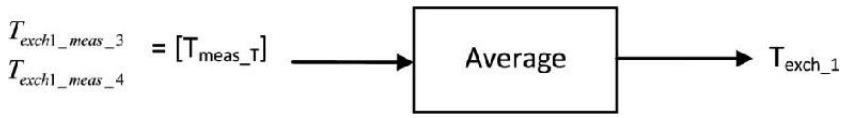


Figure 4.32. *Diagram of the way in which the average temperature is calculated*

4.3.4.2.3. Digital filtering

In order to eliminate the noise in the measurements of temperature and pressure, we use a first-order low-pass digital filter.

The low-pass digital filter IIR (Infinite Impulse Response) is characterized by the following function:

– low-pass digital filter:

$$S_n = a.e_n + (1-a).S_{n-1} \leftrightarrow H(z) = \frac{a}{1-(1-a).z^{-1}} \qquad [4.29]$$

The parameter *a* is determined by way of a program developed in Matlab® which uses the sampling and cutoff frequencies as calculation parameters.

In our case, we use the following parameters:

– sampling frequency = 0.033Hz;

– cutoff frequency = 0.002Hz.

We obtain the following result: $a = 0.3167$.

Below is a Matlab® function to determine the parameter a of the filter:

```
function a = RII(fech,fcoup) ;
tau = 1/(2*pi*fcoup) ;
per = 10/(10*fech) ;
fc1 = -per/tau ;
fc2 = exp(fc1) ;
det = 1-fc2 ;
nb = round(det*10000)/10000 ;
a = nb ;
```

Figure 4.33 is a diagram generated by Matlab® Simulink for the use of the digital filter with our measurements.

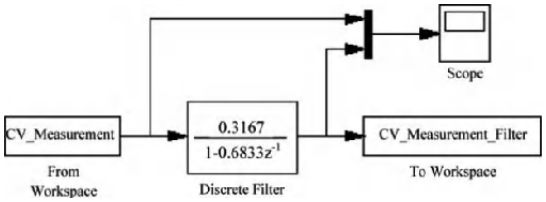


Figure 4.33. *Simulink diagram of the digital filter*

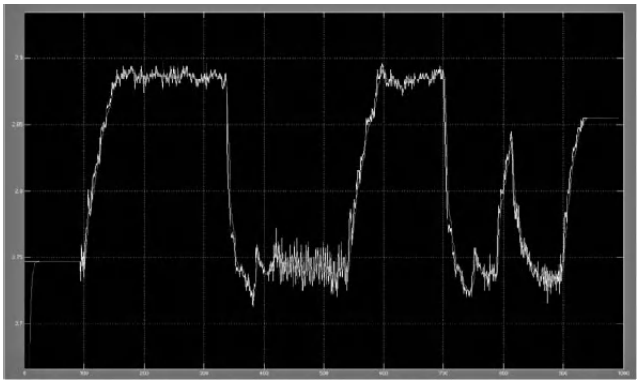


Figure 4.34. *Results of digital filtering*

4.3.4.2.4. Result of the measuring campaign

The test was performed on the exchanger 2HE1.

The test took place between 10:20 a.m. and 17:20 p.m. on May 8 2007.

Set reference point for valve 2CV3485:

Time	Reference point (%)
10:20	40%
12:15	60%
14:00	40%
15:20	60%
16:00	40%
16:20	60%
17:00	40%

Table 4.8. *Reference point for valve 2CV3485*

The system's response to pressure is the reverse of the set reference point. Also, in order to exploit our data for the purposes of parametric identification, we have inverted the reference signal.

The measured signals are heavily noised. Therefore, we used a digital filter in Matlab® Simulink to attenuate the effects of the noise and thereby work back to the clean signal that is usable for identification.

As we can see from the graph in Figure 4.35, the sensors 2TT4909 and 2TT4911 show constant temperatures throughout the measuring campaign. These sensors indicate the temperature at the inlet to the exchanger. Hence we can conclude that these temperatures are not influenced by the reference point in terms of aperture of the valve, by the pressure in the exchanger or by the output temperature from the exchanger. We choose to remove these measurements from our graphs and our identification. These data are unable to characterize the system's response.

The results of the inversion of the reference signal and filtering of the measurements are shown in Figure 4.36.

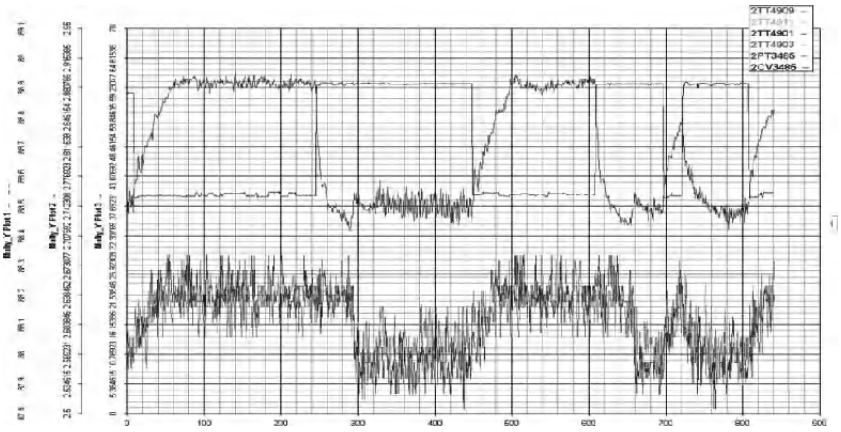


Figure 4.35. *Results from the 2HE1 exchanger – unfiltered data*

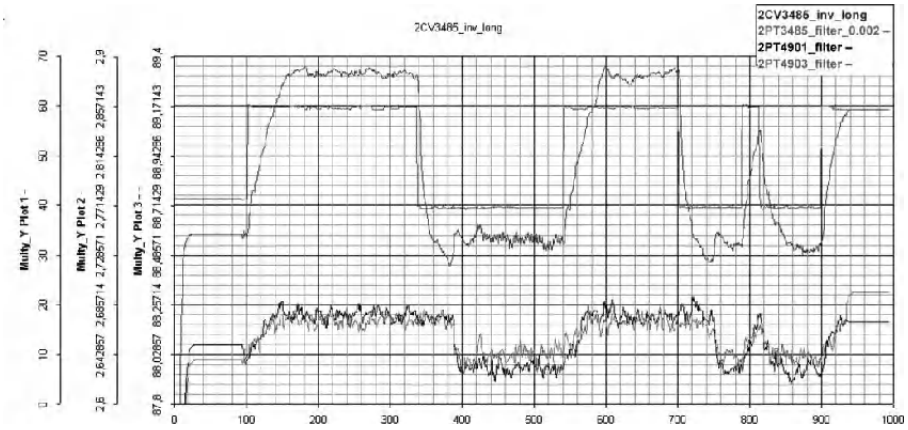


Figure 4.36. *Results from the 2HE1 exchanger – filtered data*

4.3.4.2.5. Mono-variable identification

4.3.4.2.5.1. Results with Optireg

The Optireg program can be used for identification of a SISO system. In our case, we have a SISO system with disturbance in the input. We shall perform an identification with Optireg without taking this disturbance into account.

We obtain the following modeled system:

$$H_1(Z^{-1}) = \frac{4.13\times10^{-4}\times Z^{-1}}{1-0.94\times Z^{-1}} \qquad [4.30]$$

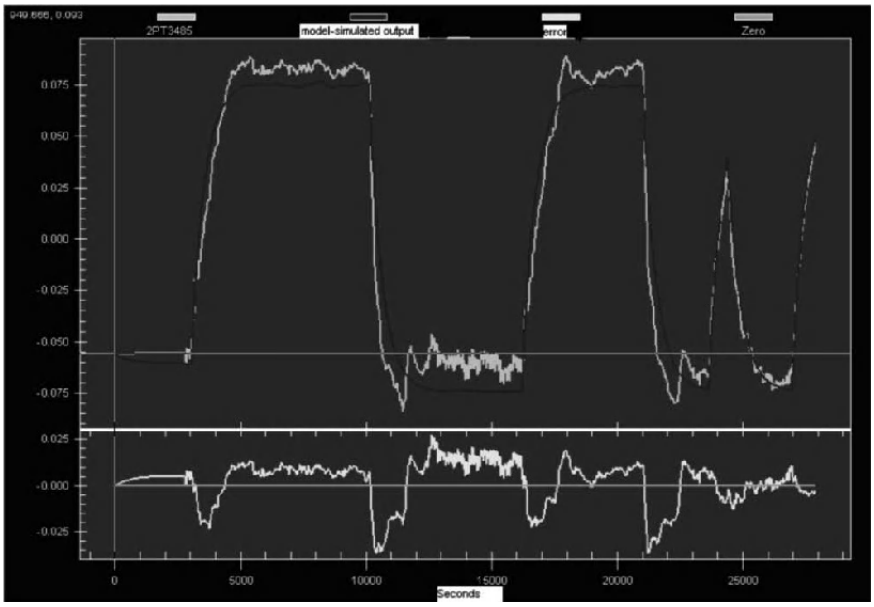


Figure 4.37. *Results of identification with Optireg*

4.3.4.2.5.2. Comparison of results

We note that the system identified contains temperature differences during the permanent regimes. These differences can be attributed to an error in the gain of the transfer function which can later be corrected by Matlab®.

We can also see that the dynamics of the identified model is consistent, to an acceptable degree, with the real-world dynamics of the system.

The significant variations in pressure that occur when the set reference point regarding the aperture of the valves change is not taken into account in the current model. This leads to significant errors at these times.

This problem could be solved if we were to use an identification program incorporating the input disturbances (such as the output temperature from the exchanger in our case).

4.3.5. *Compensation for the delay with a Smith controller based on the PI corrector UNICOS*

The Smith predictor Smith is a PID controller with an added internal model intended to deal with a greater delay. The main aim is to provide the PID with a correction to counteract the effect of the delay within the closed-loop response. This type of corrector can be used for first- and second-order (double pole) transfer functions or unstable systems with delay (see Figure 4.38).

During our work, we simulated this controller with data recorded experimentally and compared the results.

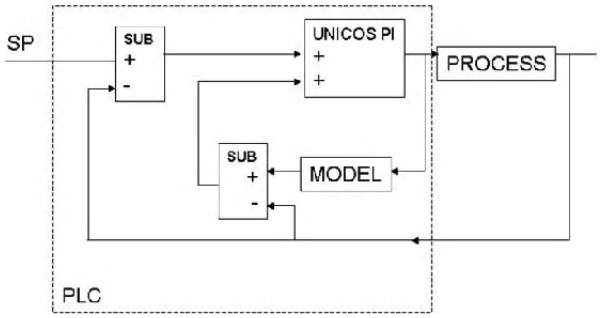


Figure 4.38. *Closed-loop Smith predictor*

4.3.5.1. *Results*

In this section, we present the results of the simulation of parametric identification and optimization of the PID for a real-world large-scale system: the nitrogen heat exchanger from the ATLAS experiment. The performance comparisons encapsulate three different configurations of control: (*i*) a conventional PI controller, (*ii*) a PI controller that has been optimized after a phase of parametric identification without taking account of the delay and (*iii*) a controller based on the Smith predictor, and optimized in order to compensate for the delay.

4.3.5.2. *Discussions about the PI controller and the model based on the Smith predictor*

In Figure 4.39, the experimental data are compared to the dynamic responses simulated by the controller models, with different configurations. Firstly, a conventional PI optimized by an operator from the cryogenics service (gain = 1, Ti = 300s); secondly a PI controller optimized after identification of the process without taking account of the delay; and finally, a third controller based on the Smith predictor approach [CAB 05] in order to compensate for the delay.

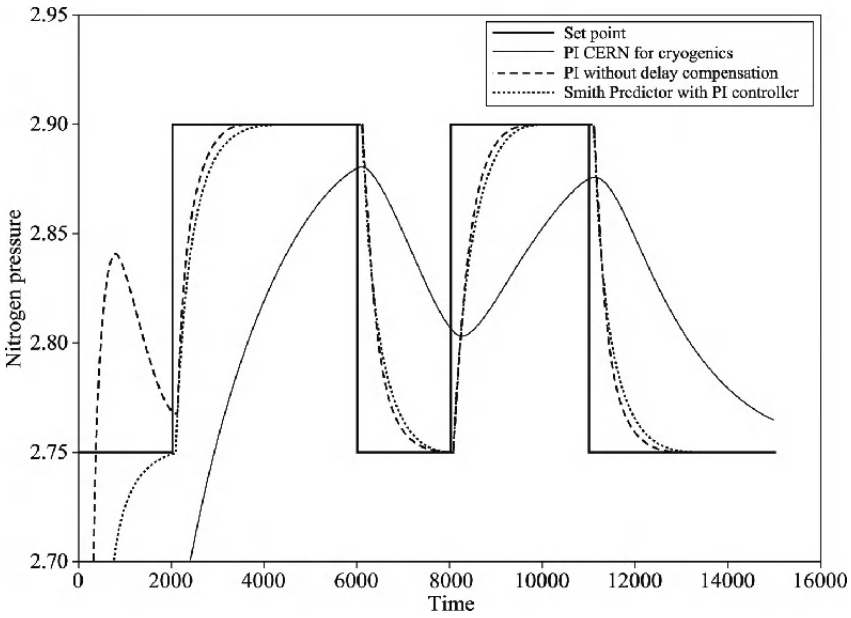


Figure 4.39. *Closed-loop response with a 60s time delay*

For a time-delay of less than two minutes, the closed-loop PI regulator is optimized in order to obtain a response comparable to a first-order system. It is particularly well suited to the process, as is the Smith predictor corrector. The PI regulator currently used has a relatively slow response time. This phenomenon is amplified for processes with even greater delays. Figure 4.40 shows a simulation of a process with a 300-second delay.

The current techniques used by the cryogenics service at CERN to control their PI regulator face problems of significant time delay or inversion of the response if the integral time Ti is increased. This method, which is widely used, yields modest but acceptable results for cryogenic processes with slow dynamics. The curve for the optimized PI regulator shows us these problems for a simulation with a 300s delay (see Figure 4.40) in relation to a simulation with a 60s delay (Figure 4.39).

In this case, only the Smith predictor regulator is able to give us a response which is sufficiently quick, precise and with no overflow. The PI regulator currently used remains acceptable even though it is not as accurate, while the optimized PI regulator may render the cryogenic process unstable.

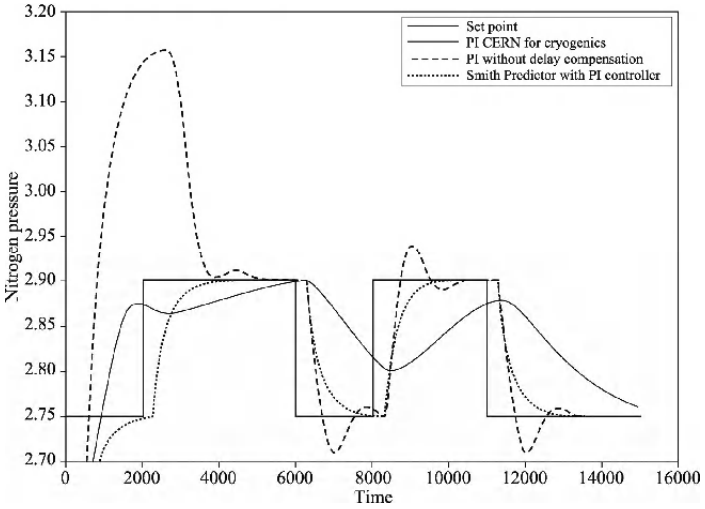


Figure 4.40. *Closed-loop response with a 300s delay*

4.3.6. *Multi-variable system*

4.3.6.1. *Objectives*

The multi-variable system has to regulate the temperatures in the bath of liquid argon in the ATLAS calorimeter. In order to do so, the system, on the basis of the temperature reference points and the measured temperatures, establishes the pressure reference points for the mono-variable regulators which, in turn, regulate the pressures in the exchangers and therefore the apertures of the nitrogen distribution valves.

The objectives are to:

– identify the transfer function H_T;

– deduce the regulator associated therewith.

4.3.6.2. *Test protocol*

In order to carry out an identification of the bath of liquid argon, we are going to use a PRBS signal to stimulate the reference points of the pressure regulators in the exchangers. The output values of the system are the forty temperatures from the bath of argon.

The values from the temperature sensors will then be reused in a data-fusion algorithm to determine the six representative temperatures for the bath of argon.

4.3.6.3. *PRBS*

4.3.6.3.1. Introduction to PRBS

PRBS is a pseudo-random signal because it is characterized by a "sequence length" within which the impulse length varies randomly, but, over a long time horizon, the impulses are periodic, with the period being defined by the sequence length.

PRBS is generated with the help of looped difference registries (in hard or software form). The maximum length of a sequence is $2N$-1, where N is the number of cells in the difference registry.

In order to properly identify the static gain of the process, the duration of at least one of the impulses (e.g. the longest impulse) must be greater than the rise time t_r of the process. The maximum duration of an impulse being $N.Te$, we get the condition:

$$N.Te > t_r \qquad [4.31]$$

where $t_r = 3\tau$ with τ being the time constant of the (first-order) system which is illustrated in Figure 4.41.

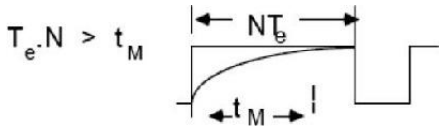


Figure 4.41. *Value of the PRBS signal*

On the basis of condition [4.31], we determine N and thus the sequence length 2^N-1. In addition, in order for the whole spectrum of frequencies to be scanned, the length of a test needs to be at least equal to that of the sequence. In many cases, we choose the length of the test (L) as being equal to the length of the sequence. If the length of the test is specified, then we need to ensure that:

$$(2^N\text{-}1).Te < L \ (L = \text{length of test}) \qquad [4.32]$$

It should be noted that condition [4.31] may yield rather large values for N, corresponding to sequence lengths that are prohibitively great – either because Te is very large or because there is a possibility that the system to be identified will change during the course of the test.

It is for this reason that in many practical situations, we choose a sub-multiple of the sampling frequency as a timer frequency for PRBS.

If:

$$f_{PRBS} = \frac{f_e}{k} \text{ ; k = 1, 2, 3…} \quad [4.33]$$

then condition [4.31] becomes:

$$k.N.Te > t_r$$

This approach is more advantageous than increasing the length of the sequence (increasing the value of N). Indeed, if we go from N to $N' = N + 1$, the maximum duration of an impulse goes from $N\,Te$ to $(N + 1)\,Te$, but the duration of the sequence doubles: $L' = 2L$. Conversely, if we choose $f_{PRBS} = f_e/2$, the maximum duration of an impulse goes from $N\,Te$ to $2\,N\,Te$ for a doubly-long sequence: $L' = 2L$.

By comparing these two approaches, it emerges that the second approach (division of frequency) results in a greater length of impulse for the same sequence length and therefore test length. If we use the notation p to represent the integer divisor of the frequency, we have division of the clock frequency (d_{max} = maximum impulse duration):

$$d_{max} = k\,N.Te;\ L' = k\,L;\ k = 1, 2, 3... \quad [4.34]$$

By increasing N by (k-1) and therefore increasing the sequence length by the same amount, we get:

$$d_{max} = (N + k\text{-}1)\,Te;\ L' = 2(k\text{-}1)\,L;\ k = 1, 2, 3... \quad [4.35]$$

4.3.6.3.2. Design of a PRBS for our system

We estimate (hypothesis about the cryogenic system):

– rise time: t_r = 6 hours = 360 minutes;

– non-distributed delay.

So as to have a large number of points in our identification, we shall take: $T_e = \frac{t_M}{10} = 36 \text{ min}$. We take Te = 30 min so as to avoid excessively long sequences.

We hypothesize that the minimum length of a stage needs to be two hours in order for the system to exhibit a measurable and meaningful response.

Thus: $T_{PRBS} = k.T_e$ = 2 hr.

Hence $k = 4$ and $T_{PRBS} = 120$ min.

We need:

$$N \geq \frac{t_M}{T_{SBPA}} = 3$$

We choose: $N = 5$

The longest impulse duration is: $L_{max} = k.N.Te = 600$ min = 10 hr.

The sequence length is therefore: $L = 2N\text{-}1 = 31$ states.

The sequence duration is therefore: $T_{seq} = L.k.Te = 3720$ min = 62 hr = 2.5 days.

We generated a PRBS signal for 6,400 points, which we then divided into eight sections comprising 800 points each.

The eight sections mean we can create eight control signals for the eight actuators in our system.

A verification of the correlation rate is then carried out to check that the signals are indeed independent. This non-correlation of the signals is vital so that no bias is caused in the identification of the system.

The 800 points in the graphs correspond to 24,000 minutes, which is equal to 400 hours ≅ 17 days of testing.

4.3.6.4. *PVSS script for control of the valves*

4.3.6.4.1. Objectives

The test lasts for seventeen days; it is therefore necessary to have an automated system to vary the set reference point regarding the pressure of the regulators for the duration of this period.

We have based the example below on a script which is already in use at CERN, employing PVSS to alter the positions of the valves.

Thus, we adjusted the program to use a matrix containing all of the values that must be assumed by the set reference point on the pressure of the regulators over time, and a function which is carried out every *Te* minutes (*Te* = sampling frequency).

4.3.6.4.2. Structure of program

The program uses a PVSS panel and a file to be inserted into the PVSS script folder.

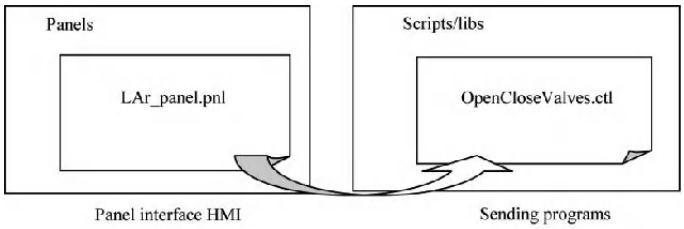


Figure 4.42. *Sending of orders via the HMI*

The PVSS panel contains UNICOS objects to visualize the state of openness of the valves in the system.

There are two buttons which the user can press to start or stop the script at any time. The two buttons contain small programs to initialize the variables and call the necessary functions.

The current script is able to modify the set reference point on the opening or closing of the valves. However, in the context of our identification, we need to modify the set reference point on the pressure of the valve regulators. This modification is not yet taken into account, but will be performed soon after.

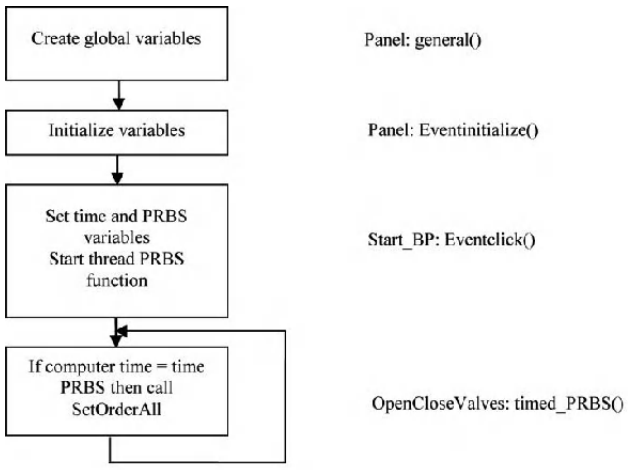


Figure 4.43. *Stages of operation*

4.3.6.5. *Data fusion*

4.3.6.5.1. Definition

Data fusion – a generic term covering a whole set of techniques – can be defined as "a process enabling us to best combine a set of multi-sourced and possibly heterogeneous data, for a higher quality of the resulting information". Data fusion (of raw or possible pre-treated data) enables us to handle a multitude of pieces of information, complementary, redundant or incomplete, drawn from heterogeneous sources, in order to obtain the "best" possible knowledge of the environment being studied.

4.3.6.5.2. Static fusion

A static fusion problem is one in which time is not explicitly involved, because:

– the system is static (immobile);

– all the data were acquired at the same time;

– all the data were acquired at the same place by different sensors, or if the system is static and we know it to be so beyond a doubt, redundancy in the data may result from measurements repeated over time.

Let $X \in R^n$ be the set of unknowns; let $Y \in R^m$ be the set of known values – either measured or known beforehand.

If $m > n$, then there is data redundancy: this is a static fusion problem.

The equation (or set of equations) $Y = H \cdot X$, or indeed $Y = h(X)$, between Y and X is called the observation equation. We draw a distinction between linear and nonlinear cases because, in the case of a linear problem, we already have well-known and effective tools and theories available to us.

4.3.6.5.3. Static fusion with no noise model: least squares method

Without knowledge about the noise and if the observation equation is linear ($Y = H \cdot X$), the fusion problem can be solved by the least squares method. In this case, we minimize the quadratic error:

$$e = \sum_{i=1}^{m} \left(y_i - H(i,:) \cdot X \right)^2 \qquad [4.36]$$

If we suppose all of the data to be consistent, the solution is:

$$\hat{X} = \left(H^t \cdot H \right)^{-1} \cdot H^t \cdot Y \qquad [4.37]$$

This calculation can produce a good estimation if the matrix $H^t \cdot H$ is properly conditioned.

If we know the average quadratic error of the measurements, there is a formalism that we can use called the *weighted least squares* method. If the observation equation is nonlinear, we can combine the least squares method with an iterative approach.

4.3.6.5.4. Static fusion with modeling of the imprecisions

There are three major formalisms:

– *bounded error model*: consider a measurement m of the value X with a maximum error e. Then, X is included in an interval $m\text{-}e < X < m + e$;

– *probabilistic model*: the imprecision is modeled by a probability density function denoted *f(x)*. The probability P such that $X \in [a,b]$, in the knowledge of m, is: $P = \int_a^b f(x)\,dx$. We sometime note this as $P([a,b]|m)$;

– *fuzzy model*: the imprecision is modeled by a possibility distribution, represented by a triangle or a trapezium. Let $\pi_m(a)$ represent the possibility that $x = a$ if the measurement is m. The set of values for which the possibility is 1 (certain) is called the kernel.

4.3.6.5.5. Dynamic fusion

We shall now look at dynamic systems – i.e. systems whose state changes over time.

We say that we are dealing with a dynamic fusion problem if we are able to model the evolution over time of all or some of the components of X. In this case, fusion takes place due to the taking into account of the movement.

In the bath of liquid argon, forty temperature sensors give us information about the temperature of the whole of the calorimeter. For our temperature regulation, we need to have six temperatures that are representative of the heat exchangers.

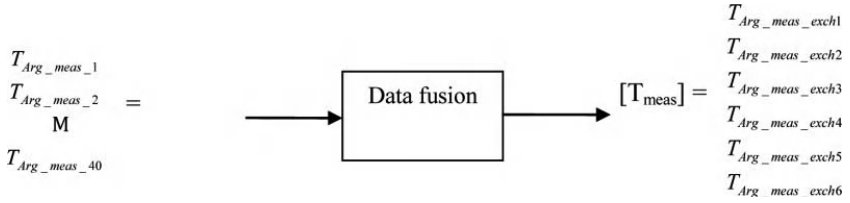


Figure 4.44. *Representation of data fusion*

4.3.6.5.6. Criteria for data fusion

In order to fuse data, it is necessary to define criteria by which to group the measurements together.

In the case of the bath of liquid argon, we can use the following criteria:

– geographic distribution of the sensors around the calorimeter on the basis of the temperature distribution;

– geographic distribution of the sensors around the calorimeter on the basis of the heat exchangers.

4.4. Conclusion

CERN, the European Center for Nuclear Research, is one of the largest and most respected centers for scientific research. Its purpose for existing is fundamental physics: the discovery of what the universe is made of and the way in which it works. At CERN, the largest and most complex scientific instruments in the world are used to study the basic components of matter: the fundamental particles. By investigating what happens when these particles collide, physicists are learning more about the laws of nature.

The instruments used at CERN are particle accelerators and detectors. The accelerators shoot beams of particles to high energies, having them collide either with each other or with stationary targets. The detectors observe and record the results of these collisions.

The Large Hadron Collider (LHC) is the largest and most powerful accelerator ever built. It is a circular particle accelerator with a circumference of 27 km, located approximately 100 meters underground, used by scientists to study the smallest particles in existence, the fundamental building blocks of everything.

Two beams of subatomic particles called “hadrons”, which are either protons or lead ions, travel in opposite directions in the circular accelerator, gaining energy on each circuit. The machine is used to recreate the conditions of the instant after the Big Bang, by bringing two beams into direct collision at very high energy. Groups of scientists the world over analyze the particles created during these collisions using special detectors in numerous experiments devoted to the LHC.

The beams circulating in the LHC are accelerated by radio-frequency (RF) cavities to over 99% light speed, thus attaining a nominal energy of 7 TeV per beam, which gives a nominal energy of collision of 14 TeV. They collide in the center of the enormous detectors built to collate the results. The beams are curved and steered using a magnetic field, produced by dipole and quadripole magnets. The maximum field required in order

to produce a stable beam with a long-enough life expectancy may be up to 8.33T in the dipole magnets. This value is achieved by using superconductor magnets with Niobium-Titanium (NbTi) coils, cooled to 1.9 K so as to facilitate the circulation of a nominal current of approximately 12 kA. Furthermore, the safe and proper operation of the detectors and the RF cavities requires cryogenic temperatures. Consequently, the operation of the accelerator requires cryogenic systems that are capable of delivering these extremely low temperatures. The LHC uses the largest cryogenic installation in the world, using helium as a cooling liquid. This installation is controlled by industrial "programmable logic controllers" – PLCs. The manufacture of both of the installations and of the control systems constituted a major challenge, as it was the first time that these cutting-edge technologies had been used in a complex system on such a vast scale.

4.4.1. *Motivations*

This chapter has presented the issue of models and control systems applied to certain cryogenic installations used at CERN, and drawn on the experience accumulated over the course of over ten years during the construction, development and installation of the LHC's cryogenic equipment.

One of the most important points is the mathematical modeling of the physical phenomena involved in the cryogenic processes. In such systems, the superposition principle is not often applicable, because of extensive nonlinearities in the relations between the causes which act on the system and their effects.

In large-scale installations such as those presented herein, the possibility for experimentation is greatly limited, both by the costs and by the risks involved in the manufacture of these installations. For this reason, in this chapter, extensive use has been made of simulation and modeling in the design process. However, under exceptional circumstances, it was possible to carry out a campaign of experiments on a nitrogen heat exchanger to identify the process being controlled.

It should also be mentioned that, in the context described above – i.e. models and control applied on a massive scale to nonlinear systems – there may be problems in terms of the practical application of the traditional, well-established techniques.

Furthermore, we can develop new advanced modeling and control techniques to optimize the management of the installations, with the aim of enhancing the performances of the control systems in terms of fidelity to the desired behavior, thereby reducing the undesirable effects which could, in the long term, drive up operational costs and affect the availability of the system.

4.4.2. *Main contributions*

In relation to the situation mentioned above, the controls applied on a large scale to cryogenic installations, the contributions made by this chapter is two-fold:

– in the context of the standard identification and control techniques, after examining the solutions presented in the high-level literature, this chapter shows the types of problems which may be caused by these traditional approaches when they are used on a massive scale. It proposes a solution by presenting the activities carried out at CERN for parametric identification and design of the control for the nitrogen heat exchanger used by the ATLAS experiment. Thus, an overview of the scientific and industrial approaches is given – particularly in relation to the implementation of the mathematical results in industrial controllers by PLC objects produced by Schneider. It should be mentioned that the heat exchanger is usually not available for experimentation, and that the work that was carried out was only possible due to exceptional authorization granted to the author and his collaborators. This, in itself, constitutes an unprecedented study and one of the main contributions;

– a new approach to the control of cryogenic installations is put forward, including the phases (*i*) modeling of the system by way of balance equations, describing the evolution of the mass flowrates and heat transfer within the time-lapse given under the hypothesis of spatial uniformity of the physical properties; (*ii*) design of the control system and its implementation, (*iii*) estimation of the parameters, i.e. the delays in communication used by the control algorithm; and (*iv*) results obtained by way of the new approach in comparison to those obtained with a traditional PID controller.

4.5. Appendices

4.5.1. *Appendix A*

4.5.1.1. *Matlab® program*

4.5.1.1.1. "PRBS" Matlab® program

```
function [U] = PRBS(Long,Ncel,h) ;
registry = ones(Ncel + 1,1) ;
registry(1)=1 ;
difference= [zeros(1,Ncel + 1) ;eye(Ncel,Ncel + 1)] ;
AdditReg=[1,1,1,3,3,2,4,4,5,8] ;
NbrHorl = fix(Long/h) ;
U = zeros(Long,1) ;
```

```
for i=1 :NbrHorl
 registry=difference*registry ;
 registry(1)=registry(Ncel + 1) + registry(AdditReg(Ncel)) ;
 if registry(1)==2
   registry(1)=0 ;
 end
 U((i-1)*h + 1 :i*h)=registry(Ncel)*ones(h,1) ;
end
```

4.5.1.2. *"Main" program for generating the command signals*

```
clear all

NbCmd = 8 ;
N=(NbCmd)*100 ;
Nreg=5 ;
clock=4 ;
U = SBPA(N,Nreg,clock) ;

u1 = U(1 :800) ;
u2 = U(801 :1600) ;
u3 = U(1601 :2400) ;
u4 = U(2401 :3200) ;
u5 = U(3201 :4000) ;
u6 = U(4001 :4800) ;
u7 = U(4801 :5600) ;
u8 = U(5601 :6400) ;

SUBPLOT(8,1,1), stairs(u1) ;
SUBPLOT(8,1,2), stairs(u2) ;
SUBPLOT(8,1,3), stairs(u3) ;
SUBPLOT(8,1,4), stairs(u4) ;
SUBPLOT(8,1,5), stairs(u5) ;
```

```
SUBPLOT(8,1,6), stairs(u6) ;
SUBPLOT(8,1,7), stairs(u7) ;
SUBPLOT(8,1,8), stairs(u8) ;
rho12 = corr(u1,u2)
rho13 = corr(u1,u3)
rho14 = corr(u1,u4)
rho15 = corr(u1,u5)
rho16 = corr(u1,u6)
rho17 = corr(u1,u7)
rho18 = corr(u1,u8)
rho23 = corr(u2,u3)
rho24 = corr(u2,u4)
rho25 = corr(u2,u5)
rho26 = corr(u2,u6)
rho27 = corr(u2,u7)
rho28 = corr(u2,u8)
rho34 = corr(u3,u4)
rho35 = corr(u3,u5)
rho36 = corr(u3,u6)
rho37 = corr(u3,u7)
rho38 = corr(u3,u8)
rho45 = corr(u4,u5)
rho46 = corr(u4,u6)
rho47 = corr(u4,u7)
rho48 = corr(u4,u8)
rho56 = corr(u5,u6)
rho57 = corr(u5,u7)
rho58 = corr(u5,u8)
rho67 = corr(u6,u7)
rho68 = corr(u6,u8)
rho78 = corr(u7,u8)
```

4.5.1.3. *Results from Matlab® program*

4.5.1.3.1. *PRBS signals*

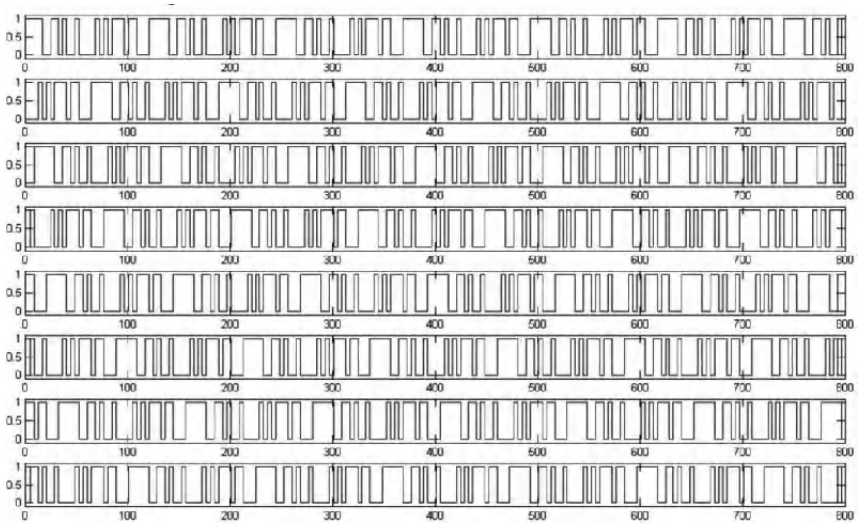


Figure 4.45. *PRBS signals*

4.5.1.3.2. Correlation of signals

```
rho12 = -0.0312
rho13 = -0.0216
rho14 = -0.0408
rho15 = -0.0120
rho16 = -0.0504
rho17 = -0.0321
rho18 = -0.0104
rho23 = -0.0312
rho24 = -0.0306
rho25 = -0.0215
rho26 = -0.0403
rho27 = -0.0215
rho28 = -0.0403
rho34 = -0.0408
rho35 = -0.0521
rho36 = -0.0504
rho37 = -0.0321
rho38 = -0.0304
```

```
rho45 = -0.0310
rho46 = -0.0302
rho47 = -0.0110
rho48 = -0.0302
rho56 = -0.0406
rho57 = -0.0426
rho58 = -0.0406
rho67 = -0.0205
rho68 = -0.0201
rho78 = -0.0406
```

4.6. Bibliography

[BRE 96] BREMER J., DAUVERGNE J.-P., DELIKARIS D., *et al.*, Cryogenics for CERN experiments: past, present and future, no CERN-LHC-96-006-ECR, *CERN*, p. 7, June 1996.

[CAB 05] CABARET S. *et al.*, "LHC GCS process tuning: selection and use of PID and Smith predictor for the regulations of the LHC experiments' gas systems", *ICALEPS*, Geneva, 10–15 October 2005.

[JAC 97] JACOBSEN R., PENONCELLO S., LEMMON E., *Thermodynamic Properties of Cryogenic Fluids*, New York, NY, Plenum, 312 p. 1997.

[ROG 08] ROGEZ E., BRADU B., MORAUX A., *et al.*, "A simulation case study for the virtual commissioning of the CERN central helium liquefier", *IEEE*, Seoul, South Korea, July 2008.

Chapter 5

Applications to a Thermal System and to Gas Systems

5.1. Advanced control of the steam temperature on exiting a superheater at a coal-burning power plant

5.1.1. *The issue*

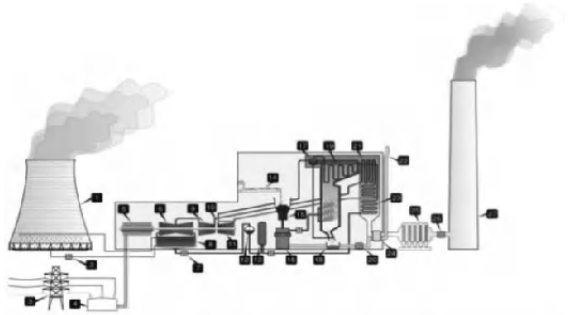
Our aim is to put a proposal to the company Alstom Power for a command/control solution for a new gas/steam power plant in Algeria, a diagram for which is presented below in Figure 5.1. Schneider Electric responded with their CPC (collaborative process control) solution corresponding to its new offer of a DCC (digital command control) system.

In terms of automation engineering, the main difficulty lies in regulating the temperature of the steam upon exiting the superheater (19 in Figure 5.1), i.e. as it enters the turbines (11 in Figure 5.1). The superheater has a 3rd- or 4th-order transfer function: the time constants vary with the steam flowrate, which is not a simple problem. Currently, there are solutions available, such as PIDD (Proportional Integral Double Derivative) controllers or state observers. However, these solutions exhibit drawbacks in terms of regulation and maintenance. For the first, PIDD-type solutions, the influence of the derivatives damages the robustness by amplifying the disturbances. With the second, state observer-type solutions, the regulations are particularly difficult to put in place for the maintenance engineers. Thus, we can see the advantage of putting forward a new internal model control (IMC) which satisfies

---

Chapter written by Sébastien CABARET and Hervé COPPIER.

the functional specifications whilst optimizing the regulations and the ease of maintenance.



1. Atmospheric coolant	10. Steam control valve	19. Superheater
2. Circulation pump	11. High-pressure steam turbine	20. Airflow fan
3. Triphase transmission cable	12. Deaerator	21. Re-superheater
4. Transformer	13. Water reheater	22. Air inlet for combustion
5. Alternator	14. Coal conveyor	23. Economizer
6. Low-pressure steam turbine	15. Coal silos	24. Air reheater
7. Water pump	16. Coal mill	25. Electric dust remover
8. Condenser	17. Boiler balloon	26. Draw fan
9. Steam turbine	18. Slag grate	27. Chimney

Figure 5.1. *Diagram and hardware components of a "standard" coal-burning power plant*

5.1.2. *The internal model corrector (IMC)*

In historical terms, Figure 5.2 shows the internal model regulator as presented by Menahem [MEN 81].

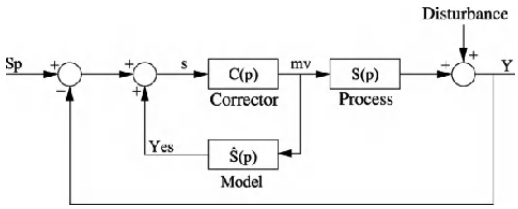


Figure 5.2. *General structure of the IMC with the process*

If we assume that the model is perfect (i.e. that there is no disturbance), the transfer function is then:

$$\frac{Y_{meas}}{S_p} = C(p) \cdot S(p) \qquad [5.1]$$

$C(p)$ thus acts as an open-chain corrector, which has the advantage of ensuring the stability of the process being regulated if chosen properly. In addition, the regulator is robust to the derivatives of the process, because all the disturbances and modeling errors are compensated for when put through the corrector.

In order to compute the corrector, we need to isolate those terms which cannot be compensated for in the process (e.g. pure time-delay).

Consider:

$$S(p) = S_0.S_1(p).S_2(p) \qquad [5.2]$$

where S_0 is the static gain, $S_1(p)$ a transfer function including all the compensable terms and $S_2(p)$ a transfer function including all the non-compensable terms.

$C(p)$ ideally would be equal to:

$$\frac{1}{S(p)} \qquad [5.3]$$

Of course, this is not possible with the non-compensable terms. By setting $S'(p)$ as the desired closed-loop transfer function (generally of the same order as the process), fixing the desired dynamics, the corrector is then written as:

$$C(p) = \frac{S'(p)}{S_1(p)} \qquad [5.4]$$

The corrective action must take account of the inherent limitations to the control (range, speed of variations), and the trajectory sought must be compatible with the process. The action has to be adapted to the order of the process. In order to ensure that this action is appropriate for the process, it is created on the basis of the process itself, so we choose a dynamic closed-loop behavior of the same order as the open-loop behavior.

The internal model corrector thus manifests itself in the form shown in Figure 5.3.

Another highly practical structure can be represented by Figure 5.4, by computing the corrector on the basis of pole placement by state feedback on the compensable part $S_1(p)$ of the model of the process.

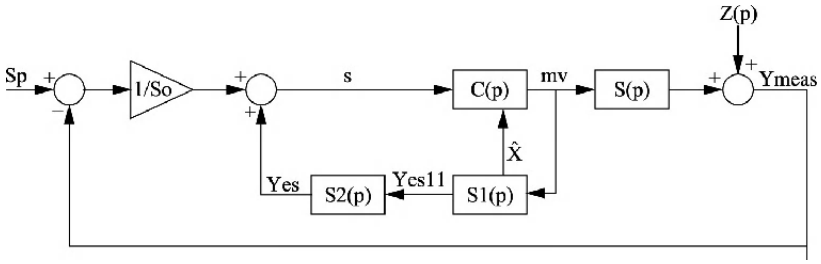


Figure 5.3. *Structure of the IMC (with compensable and non-compensable terms) with the process*

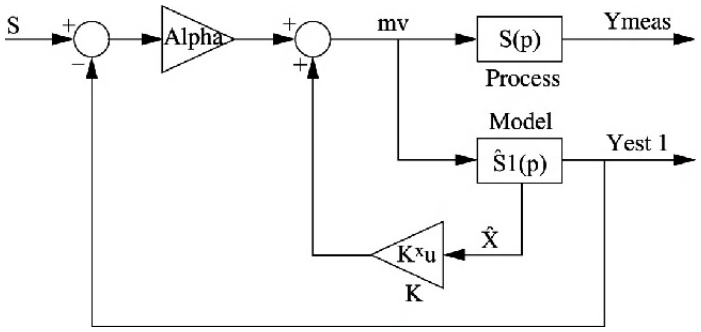


Figure 5.4. *Structure of the corrector by pole placement*

Thus, the scalar coefficient α and matrix coefficient K are found by identification with the desired dynamics (poles of $S'(p)$).

EXAMPLE OF CALCULATION.– Consider the transfer function of a process:

$$S(p)=\frac{1}{(p^2+0.8p+1)}e^{-2p} \qquad [5.5]$$

Knowing that in a closed-loop regime we wish to have the following transfer function:

$$S'(p)=\frac{1}{(p+1)^2}e^{-2p} \qquad [5.6]$$

The structure of the internal model corrector is given by the block diagram in Figure 5.5.

α =1 and K = [1.2, 1]

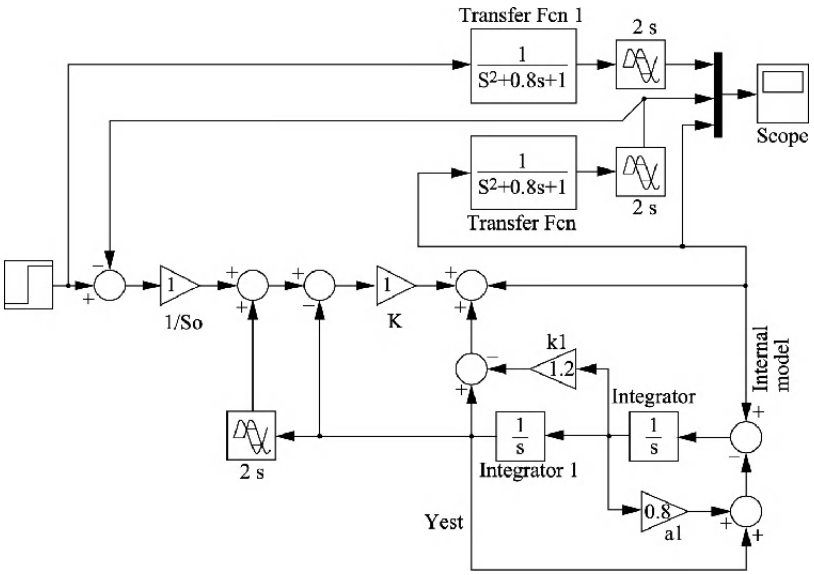


Figure 5.5. *Example of an IMC corrector*

The internal model regulator therefore enables us to impose a trajectory that is physically compatible (of the same order) on a process identifiable by a polynomial in the frequential domain. The advantages to IMC regulation are many. To begin with, this corrector enables us to later compensate for all the poles in a system whose real part is negative. Then, the desired trajectory respects the physics of the process. Finally, the equations obtained are simple.

5.1.3. *Multi-order regulator: 4th-order IMC*

In relation to the technical specifications and the measurements carried out on the process, it was decided to model the process as the product of the first four orders of different time constants. The use of a factorizable transfer function (real time constants) typically corresponds to the problems of regulation encountered with industrial processes.

After elaborating a number of possible solutions, we made the following decision: the regulator ought to be able to yield a transfer function for which all the time constants are divided by a common factor. This solution presents the advantage of having a single, easy-to-use regulatory parameter. In addition, from the point of view of implementation, the calculations are easy.

Consider the transfer functions identified for the process and for the desired dynamics:

$$S(p)=\frac{So}{(\theta_1 p+1)(\theta_2 p+1)(\theta_3 p+1)(\theta_4 p+1)}e^{-\tau p} \quad [5.7]$$

$$S'(p)=\frac{1}{\left(\frac{\theta_1}{n}p+1\right)\left(\frac{\theta_2}{n}p+1\right)\left(\frac{\theta_3}{n}p+1\right)\left(\frac{\theta_4}{n}p+1\right)}e^{-\tau p} \quad [5.8]$$

where *n* is the lone regulatory parameter of the desired time constants.

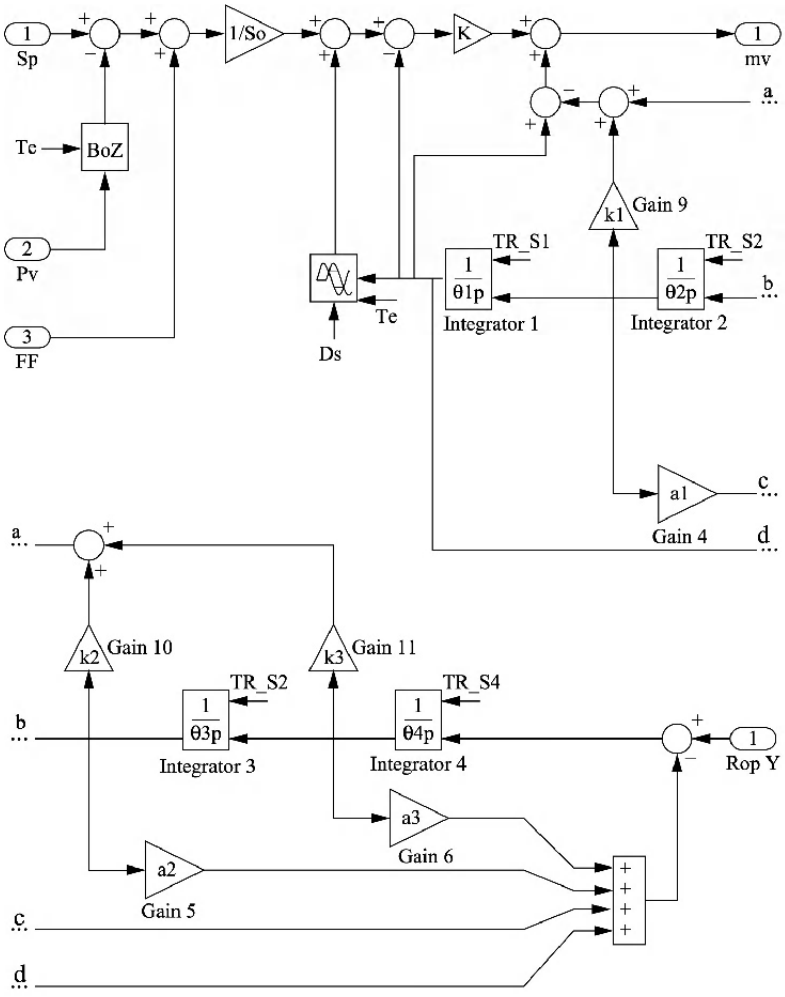


Figure 5.6. *Block diagram of 4th-order IMC*

The expressions of the different parameters in Figure 5.7 are given as follows:

$$K = n^3$$

$$k_1 = a_1 (n^3 - 1)$$

$$k_2 = a_2 (n^2 - 1)$$

$$k_3 = a_3 (n - 1)$$

where a_1, a_2 and a_3 represent the gain of the model:

$$a_1 = \frac{(\theta_1 + \theta_2 + \theta_3 + \theta_4)}{\theta_1}$$

$$a_2 = \frac{\theta_1\theta_2 + \theta_1\theta_3 + \theta_2\theta_3 + \theta_3\theta_4}{\theta_1\theta_2}$$

$$a_3 = \frac{\theta_1\theta_2\theta_3 + \theta_1\theta_2\theta_4 + \theta_1\theta_3\theta_4 + \theta_2\theta_3\theta_4}{\theta_1\theta_2\theta_3}$$

where:

– θ_1, θ_2, θ_3 and θ_4: time constants of the model;

– Ds, S_0: pure time-delay and gain of the process;

– Te: sampling frequency of the model.

5.1.4. *Results*

This algorithm was tested using Matlab Simulink and on Schneider Electric's PLC TSX57 Premium.

We can see that the closed-loop system is indeed twice as fast, as long as the static gain is unitary and the disturbance is properly damped.

We note that even in spite of the significant modeling errors, the algorithm still converges and the control law is not oscillatory. Thus, the algorithm is relatively robust.

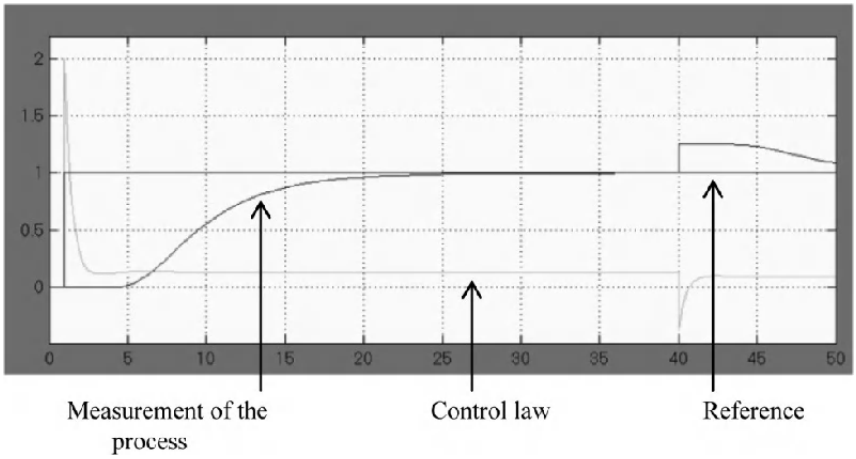


Figure 5.7. *Response to a reference value step and to a disturbance value step of 25%*

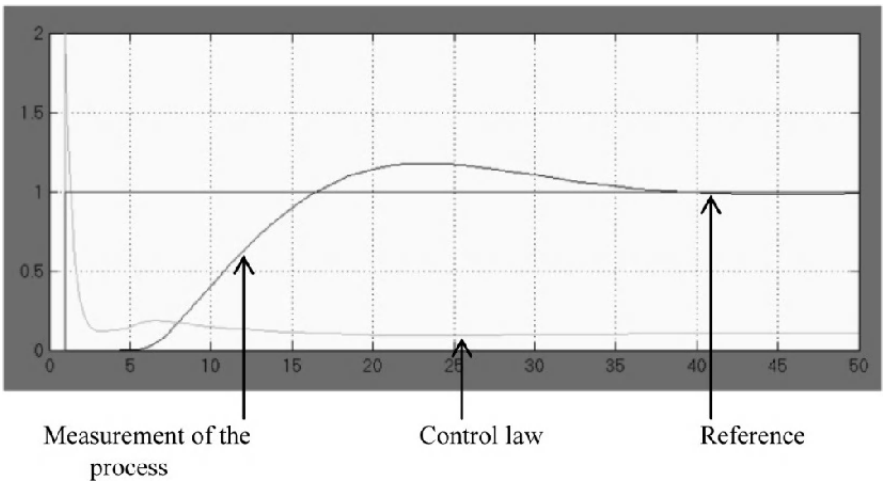


Figure 5.8. *Response to a reference value step and to a disturbance value step of 25% with 20% error in the parameters of the model*

5.1.4.1. *Response compared with a predictive algorithm for a first-order process*

The parameters of the two regulators were set up so as to obtain the same time constant in a closed-loop regime.

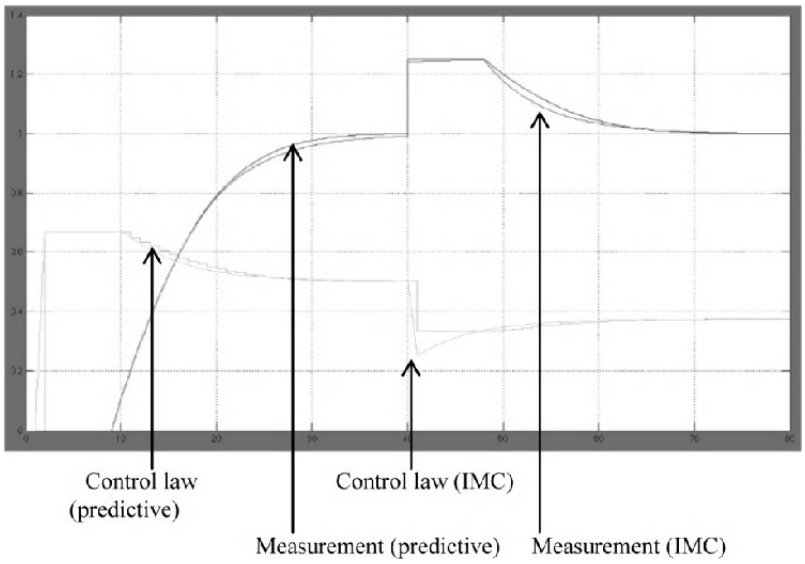


Figure 5.9. *Response to a unitary reference value step and to a disturbance value step of 25% and saturation of the control at 65%*

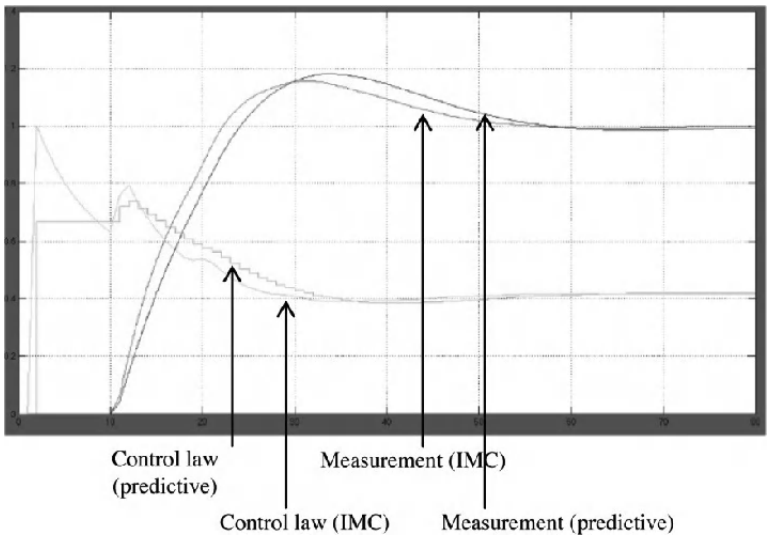


Figure 5.10. *Response to a unitary reference value step with 20% error on each of the model's parameters*

5.2. Application to gas systems

The JCOP (Joint Control Project) was created at CERN with the aim of finding common control solutions for all the experiments at the new accelerator. In this context, the LHC Gas Control System (GCS) project was set up to provide all of the control applications for the LHC's 23 gas systems.

The results and developments presented in this chapter use the identification methods described in Chapter 2.

5.2.1. *The gas systems*

The LHC's gas systems have the simple purpose of supplying the detectors and sub-detectors with gaseous mixtures maintained in very specific physical conditions of operation. To summarize its basic operation, a gas system at the LHC is designed in a nearly-closed loop for the circulation of the fluid. Feeding off the primary supply, it needs a *mixer* to adjust the required mixture. The gaseous mixture then needs to be distributed to the volume chambers (detector).

A volume *buffer* system adjusts the flow on the basis of the atmospheric fluctuations and the hazards of the process. Finally, the closed circuit is put in place by a circulation pump. The pump is not always essential for small systems, in which the flow can be ensured by a pressure differential maintained in the gas circuit.

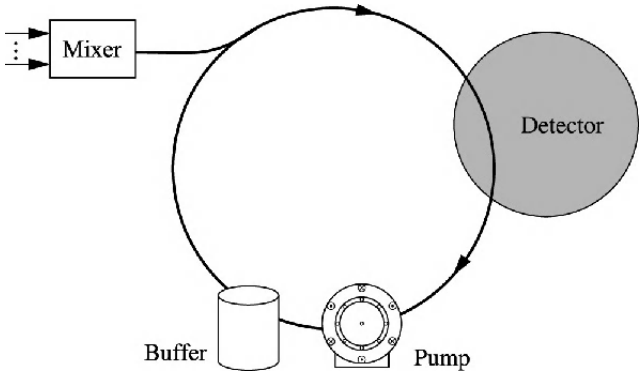


Figure 5.11. *The high-level principle of a gas system*

Regardless of the design of the body of the detector, it will never be 100% airtight. This simple observation explains the reasons for circulation to maintain the gaseous mixture at constant levels. For instance, air penetrates into the mixture

inside the detector, while the detector "loses" some of its mixture to the outside world. There is a sort of natural balancing act which takes place over time.

The crux of the issue with the LHC's gas systems is to maintain a constant gaseous mixture in the required physio-chemical conditions.

5.2.1.1. *Structure and design*

The gas is kept flowing by a pressure differential maintained throughout the gas circuit. This is created by an appropriate mechanical design (length, diameter of the pipes, pump, controlled valves, etc.).

The gas systems are identified by functional modules, enabling us to give an effective structural analysis of each of the 23 systems involved in the experiments at the LHC. For instance, the mixer module takes care of the mixing and flowrate of the gaseous fluids. The distribution module takes care of the distribution of the mixture to the sub-detector. The pump takes care of the circulation of the gas.

Certain systems have additional elements which are attached to the loop of the system. Thus, we can find a purifier module, a gas analysis module, etc.

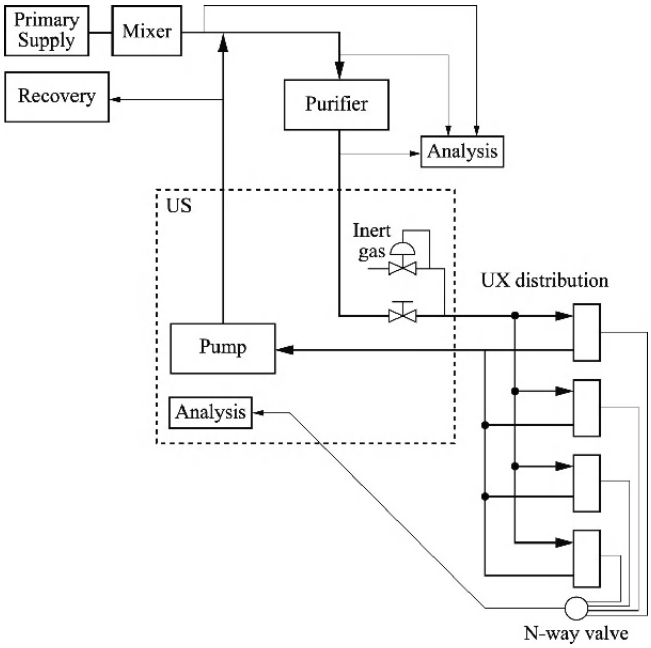


Figure 5.12. *Modular view of a gas system*

5.2.1.2. *Operation*

5.2.1.2.1. The primary gas module

Generally, there is a module for every experiment at the LHC. The gases are stored in bottles, batteries of bottles or *dewars*. This storage is located in the building at ground level and separate from the computing equipment and other hardware (dewars are usually stored in open air). In addition, sophisticated devices are installed nearby (a chromatograph, a monochromator, etc.). Each primary gas has two different sources of supply, and if one of the storage tanks becomes empty, the system automatically switches to using the other. The control of the primary gases is monitored by pressure, safety switches and control parameters to monitor the changes in the variables. A special monitor is installed for gases which have to be heated at the level of the bottles.

5.2.1.2.2. The mixer module

The mixer module is present for every installation. The process is passive. The mixer can handle up to three primary gas lines. It regulates the flowrates along those lines to achieve the desired ratios. The mixer mixes the gases in a turbulent manner. All the mixers are in the ground-level building and are separate from the computers and other electrical hardware for safety reasons.

The mixture is transported directly via a single pipe to the detector, which is located in the cavern (100 m below). The flowrate depends on the demand from the detector and the modules of the gas system in question. The mixer also attempts to maintain a constant operational pressure.

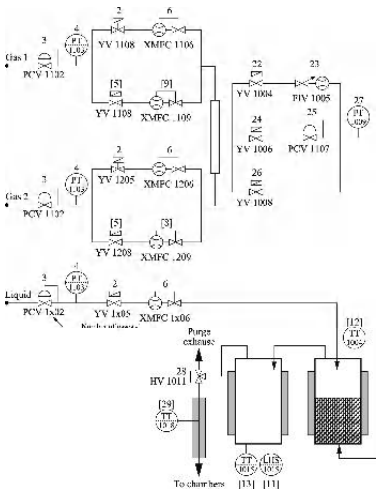


Figure 5.13. *The mixer module*

The main control parameters for the module are ratios, flowrates, detections of the presence of gases in the lines and independent instruments for controlling the composition of the gas. These instruments are shared between various systems and can furnish correction data for the mixture in the detector.

5.2.1.2.3. The distribution module

The distribution module is present for every gas system. The arrival of the gas corresponds to its exit from the mixer module. The distribution module is subdivided into racks. Each rack represents an area in the detector (up to 26 racks per detector). Each area is divided into several chambers (up to 36 chambers per rack and over 250 chambers for the largest system). The output from the distribution module is concentrated in a single pipe. The distribution racks are underground in the experiment and are completely inaccessible and always separate from the electronics and computers.

The regulation of the internal pressure in the detector can be done on the output line from the distribution module. The chambers are assimilated to channels. The output from each channel can be connected to the analysis chain, which measures the quality of the mixture.

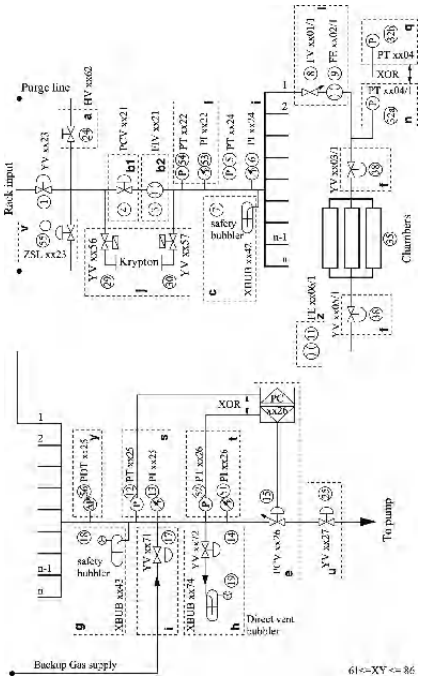


Figure 5.14. *The rack of the distribution module*

5.2.1.2.4. The pump and exhaust modules

The pump module takes care of the circulation of gas in the circuit. It is an active element located on the surface which mainly comprises a pump. It acts on the output from the detector and forms the link between high pressure and low pressure. The regulation of the pressure in the detector can be done here, depending on the configurations. This module may also have a variety of different architectures (an "all-or-nothing" or analog pump).

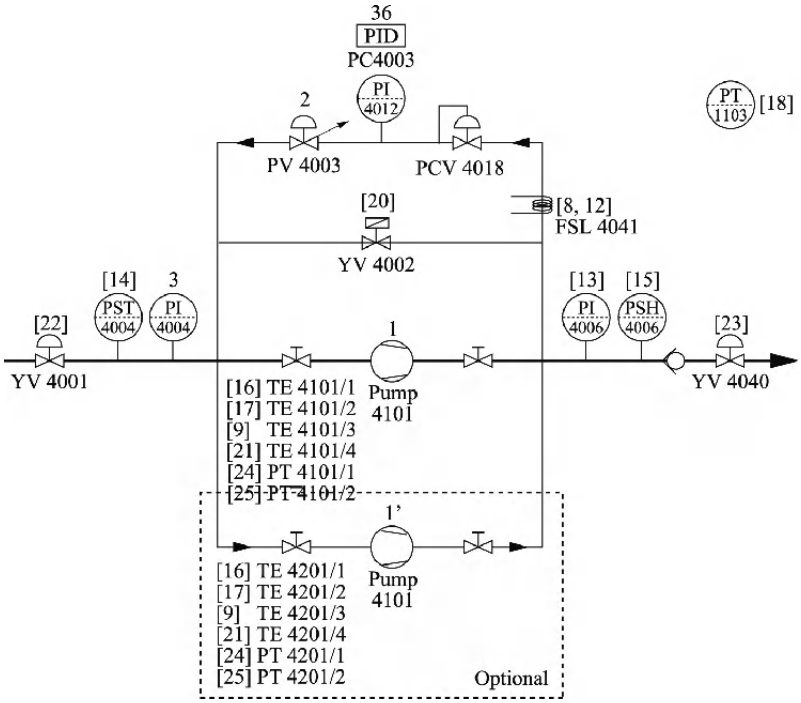


Figure 5.15. *The pump module*

The exhaust module can be used to evacuate the gaseous mixture from the main circuit. This function is optional, depending on the systems. It often allows greater flexibility in terms of operation. The exhaust, logically, is on the surface.

5.2.1.2.5. Purification and CO_2 removal module

The purifier serves certain systems which need to prevent the deterioration of the quality of the gas. This module is comprised of two columns which can function alternately on the circuit. This module is located on the surface and works at high pressure in the interests of a better efficiency of purification.

Purification is done by a chemical agent and a reactive grid which removes the water and oxygen contamination. When the column in question is saturated, it has to undergo a phase of regeneration at high temperature. In the interests of optimal operation and function, both columns ensure purification by functioning alternately (regeneration). The control of this module is essentially taken care of by sensors of temperature and of gas quality on entering and exiting the circuit. The peculiarity of this module is that it has substantial operational sequences which mean that the function of the process can be alternated between the columns.

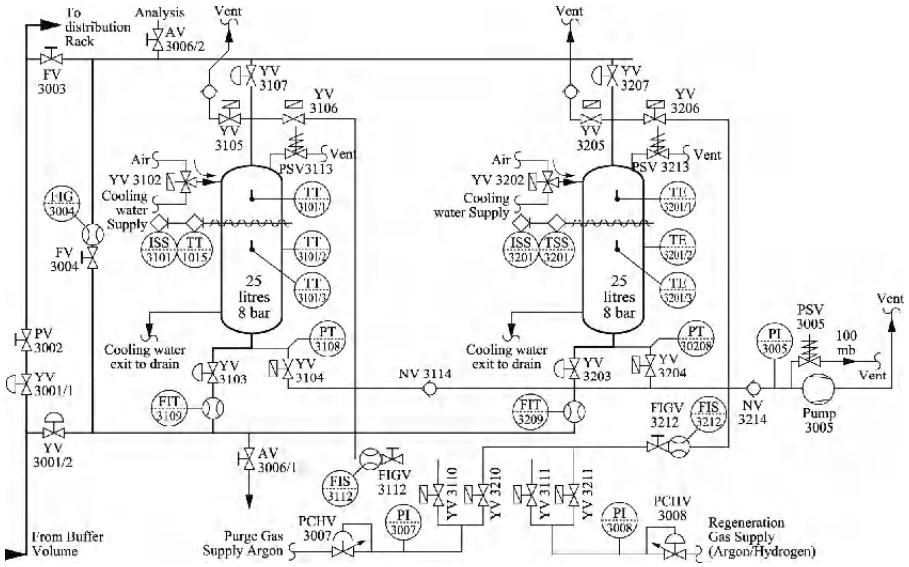


Figure 5.16. *The purifier*

The CO_2 removal module is built to the same blueprint as the purifier. It extracts CO_2 from the circuit loop. It also comprises two columns which can work alternately. This module is located on the surface and operates under high pressure for better efficiency of removal. The main difference between this module and the purifier is its operational period: the CO_2 removal module is in an operational state for a few weeks at a time (annual reset) whereas the purifier is in operation near constantly. The control of CO_2 removal is also similar to that of purification, with complex sequences of operation. The elements are constructed on the same schema with different active agents and a few optional pieces of equipment.

5.2.1.2.6. The gas analysis module

The analysis module consists of machinery to analyze mixtures of gases coming from various parts of the system. The module is located in the surface building. It is organized into analysis chains, which can be divided into two types. Type A corresponds to a selection of lines to be analyzed by standard pneumatic valves. Type B corresponds to a selection of lines to be analyzed by special N-way valves. Each chain is controlled independently.

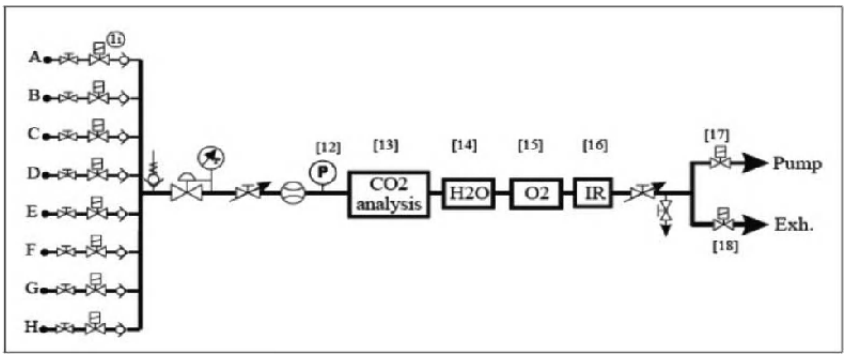


Figure 5.17. *The type-A gas analysis module*

5.2.1.2.7. The role of the buffer

The buffer compensates for the difference in atmospheric pressure. The aim of the gas system is to have the detector at ambient pressure. Any variation in pressure – however slight – must be compensated for (pressure regulation). The detector is in a fixed and rigid container which means the gas cannot expand or be compressed. If the variation is too great, the detector may be damaged. The buffer deals with excess pressure in the detector; and if the pressure is too low, the buffer "provides" (artificially) the volume of gas necessary to bring the detector back to equilibrium.

5.2.2. *The major regulations*

The 23 gas systems involved in the experiments at the accelerator are structured around a common pattern (the modular approach) and yet are all different. This duality gives us a window onto the regulatory issues surrounding these complex installations.

The gas systems defined above are intended to provide a gaseous mixture in predefined operating conditions. The design and elements which make up these

installations are, in themselves, able to ensure favorable operating conditions, but are not able to satisfy all of the requirements. It is crucial that certain physical values be controlled. We can distinguish a number of so-called "critical" regulations for the correct operation of the gas system.

5.2.2.1. *Cascaded control of the flow of the primary gases in the mixer*

This regulatory loop is essential, because it is needed to ensure a sufficient flowrate to obtain a constant ratio in the gaseous mixture. This cascade-type regulation takes place according to the principle illustrated in Figure 5.18.

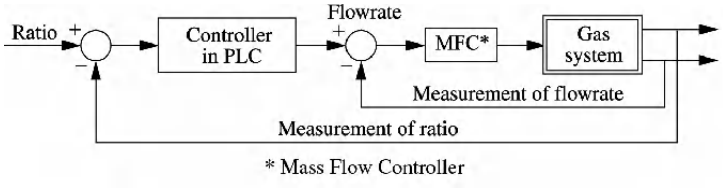


Figure 5.18. *Loop to regulate the ratio in the mixer module*

5.2.2.2. *Regulation of the pressure of the output from the detector – control of the pressure in general (pump, distribution and exhaust modules)*

The pressure is, without a doubt, the most difficult physical value to maintain with a view to the required performances. For good detection of the particles, the pressure in the detectors must be constant to a degree of precision often very near to a single millibar. The slightest variation in pressure has an effect on the celerity of the particles passing through the sub-detector. This phenomenon greatly decreases the relevance of the measurements taken by the physicists, so it is absolutely crucial that the pressure be correct. Each different gas system has different strategies (corresponding to their different configurations) for providing a constant level of pressure in the detector.

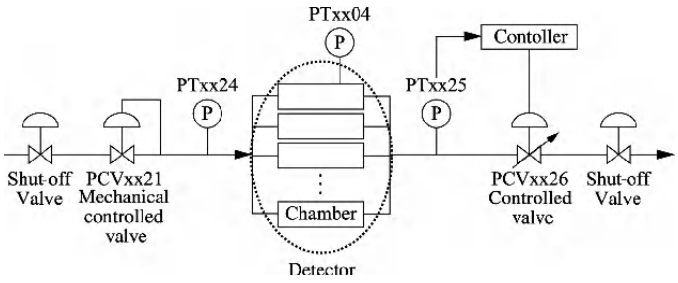


Figure 5.19. *Control of the pressure in the sub-detector by way of the valve to regulate the output from the racks in the distribution module*

The first way is to limit the output pressure (PTxx25) by controlling the degree of aperture of the valves regulating the output from the racks in the distribution module.

The second is to limit the pressure at the level of the pump module. This strategy uses the regulation valve to bypass the pump. By adjusting the aperture of this valve, the flow of gas on exiting the detector (and therefore on entering the detector as well) is affected. The inlet pressure (PT4004) can thus be modulated using the bypass valve.

It is also possible to combine the two valves (distribution rack output and pump bypass) to control the pressure.

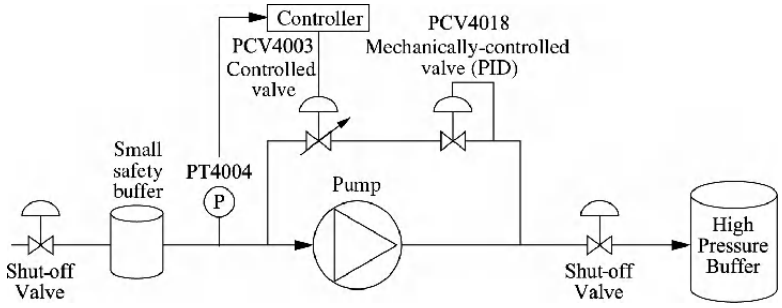


Figure 5.20. *Control of the pressure in the sub-detector by way of the valve to bypass the pump module*

Neither of these two strategies for controlling the pressure in the detector is perfectly insulated from all external disturbances. For instance, it should be noted that if there is an exhaust module in place, it plays an active role (cascade regulation on the MFC) in influencing the pressure on exiting the buffer. This pressure from the buffer may also be affected by the variations in atmospheric pressure. From cause to effect, a slight variation of the pressure in the buffer causes disturbances to the pressure in the detector itself.

The slightest change in flowrate or configuration (e.g. the number of racks connected, etc.) influences the pressure inside the detector. The regulation of the pressure is, without contest, the most complex and most difficult task to perform.

5.2.2.3. *Control of the temperature in the columns of the purifier and the CO_2 absorber modules*

The purification module (and/or the CO_2 removal module) may not be present in each and every system. Yet the regulation of the temperature of these modules

requires a certain amount of attention. During the regeneration phase of a column, the cartridge containing the active agent has to be heated to a threshold temperature for a sufficient period of time. For this purpose, each cartridge has a heating wire element and three temperature sensors, so that the interior containing the active agent can be heated sufficiently without damaging the exterior.

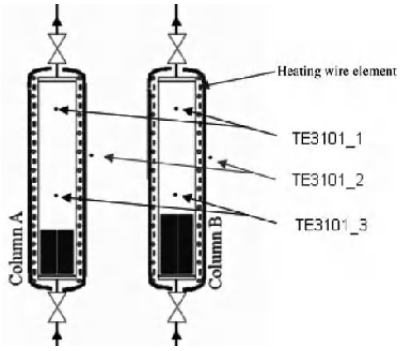


Figure 5.21. *Control of the temperature in the cartridges of the purifier*

5.2.2.4. *Control of humidity in the humidifier*

The humidifier is a module whose role is to inject the right amount of water into the gas process. The module regulates the humidity by adjusting the flow of water (using an MFC). Just as happens with the mixer, the humidity is controlled by cascaded regulation.

5.2.3. *The control system and acquisition of measurements*

5.2.3.1. *Architecture*

The control system for the GCS project uses industrial solutions. This arbitrary choice means that in designing the command/control system, we are able to focus on its implementation and development.

The control systems use industrial programmable logic controllers from Schneider Electric (TSX Premium). The associated SCADA, provided by ETM is PVSS II. The communication between these two modules uses the Modbus TCP protocol. The interfacing between the ground and the PLC is organized by deported modules made by WAGO. The input/output modules (local or deported) from Schneider are not used, and communication between the WAGO and TSX Premium modules runs through a Profibus communication module.

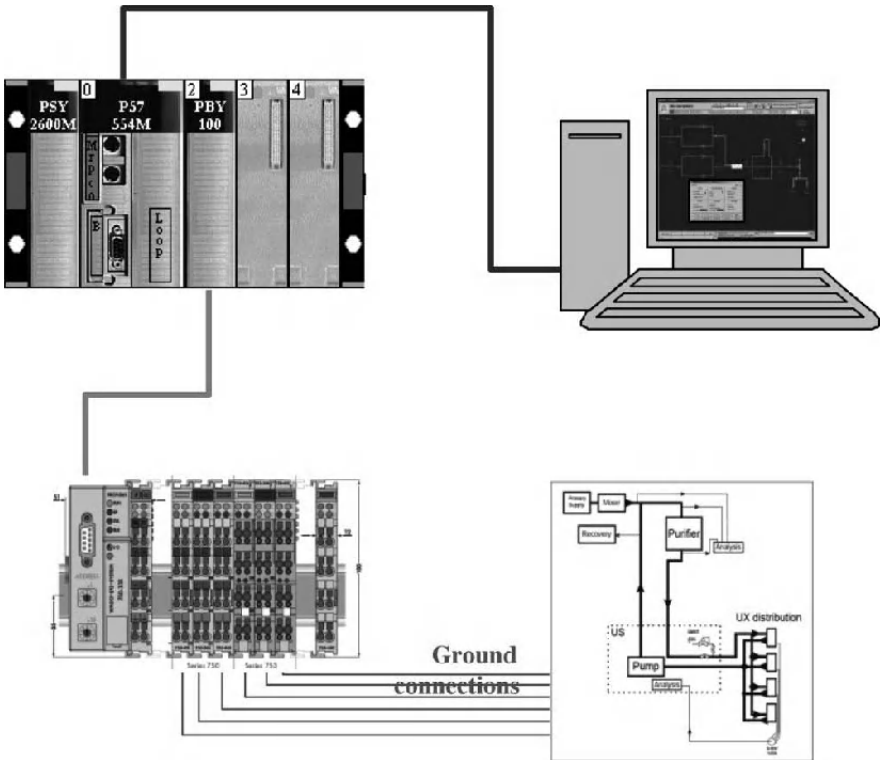


Figure 5.22. *The control system for a gas system as part of the LHC experiments*

5.2.3.2. *Measuring campaigns*

The PLC acquires measurements in real time.

The acquisition of data with a reliable experimentation protocol uses the tool DataStore, made by Schneider. This tool enables us to take readings in real time, connecting to the Schneider PLCs in local or remote mode. DataStore uses the communication standard (Modbus, Modbus TCP/IP) via OFS.

5.2.4. *Modeling, identification and experimental results*

5.2.4.1. *Choice for modeling*

The gas systems of the LHC experiments are all subject to physical constraints: pressures, flowrates, temperatures, etc. These variables are closely linked to the characteristics of the gaseous mixtures and to the structures of the systems (piping, actuators, etc.).

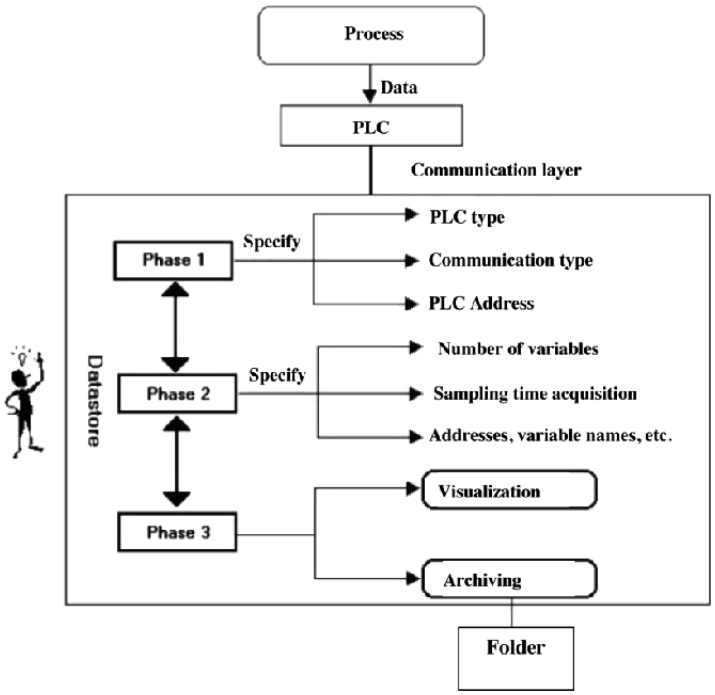


Figure 5.23. *Principle of operation of DataStore (source: Schneider)*

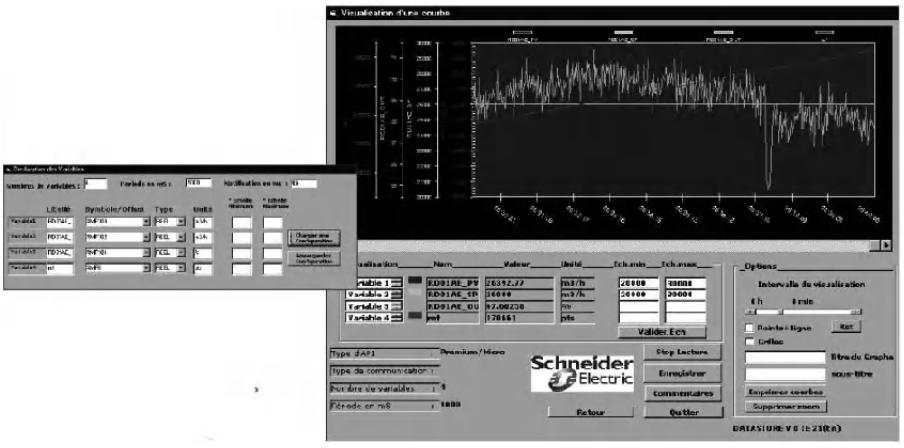


Figure 5.24. *Screenshot of DataStore in read/record mode (source: Schneider)*

A great many formulations in thermodynamics and fluid mechanics can be used to lay the foundations for modeling. Whether we use the Bernoulli equation or the perfect gas equation, there is a wide range of actions which we can undertake to implement a knowledge model. These approaches are indeed advantageous for the detailed study of the phenomena needing to be quantified, evaluated and understood.

In gas systems, the slightest variation in a variable relating to the system's behavior can have direct repercussions for the overall behavior. For instance, an alteration of the inlet flowrate may influence the pressure at the buffer on exiting the detector. This being the case, modeling by way of physio-chemical laws greatly complexifies the study of the control mechanisms.

Knowledge models also require us to have access to measurable and measured variables. Unfortunately, the gas installations do not always have all the sensors that are necessary to satisfy this condition.

Thus, in order to make it easier to study the regulatory loops of the gas systems, it is preferable to use behavioral models, which can be exploited more effectively to establish control laws.

For our purposes, we choose to use graphic and parametric methods of identification (see Chapter 2).

The gas systems in the LHC experiments have the peculiarity of only working in extreme conditions (e.g. the very, very low temperatures in cryogenics). Hence, the nonlinearities depend little on the physical phenomena relating to the operating conditions. Such nonlinearities depend more, in fact, on the properties of the gaseous mixtures.

EXAMPLE.– Influence of the flow ratios for an analog valve

Let Kv represent the coefficient of the valve, calculated using the constructive formula:

$$Kv = \frac{Q}{490}\sqrt{\frac{\rho_G * T_1}{\Delta P * P_2}} \qquad [5.9]$$

where Q = flowrate in m^3/hr, T_1 = temperature in K and ρ_G = gravity in kg/m^3.

The characteristic curve of the Equal Percentage flowrate for the valve is shown below.

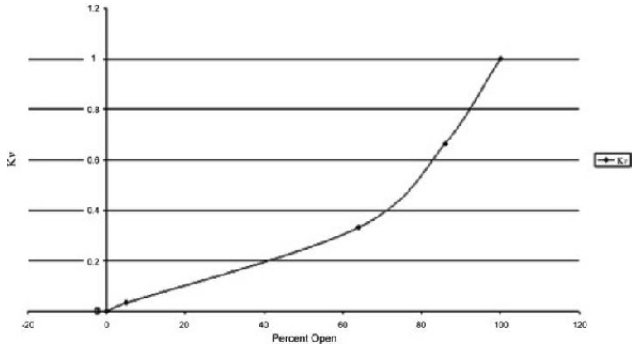


Figure 5.25. *Constructive characteristic curve for the valve*

For a mixture of Neon/CO_2 gases in the proportions 90/10 and at 30.0°C, we can say that the gravity is equal to:

$$\rho_{gas,90/10} = \frac{\rho_{gas}}{\rho_{air}} = \frac{0.93842}{1.20474} = 0.779 \qquad [5.10]$$

where:

– ρ_{gas} = 0.93842 kg/l, the density of the mixture;

– $V_m = RT/p$ = 24.36 liters/mole, the molar volume of the mixture;

– ρ_{air} = the density of air.

Consider a variation of 10 in the ratio (so the proportion becomes 80/20). The gravity then becomes:

$$\rho_{gas,\,80/20} = 0.8616 \qquad [5.11]$$

Thus, for a constant flowrate of 15 Nm^3/hr (e.g. $\Delta P = 1.0$; $P2 = 0.5$), we find:

$$Kv_{80/20} = 0.699 \;\; Kv_{90/10} = 0.665 \qquad [5.12]$$

Hence, we can see that:

$$Kv_{90/100} = 0.665 \rightarrow \approx 86\% \qquad [5.13]$$

$$Kv_{80/20} = 0.669 \rightarrow \approx 87.5\% \quad [5.14]$$

In conclusion for this example, a change in ratio from 90/10 to 80/20 leads to an error of around 1.5% in terms of the aperture of the valve (and hence a nonlinearity on the dynamic behavior!)

Hereafter, we shall consider that the nonlinearities are assimilated to measuring errors or disturbances in the system, hypothesizing that the operating conditions are close to the nominal conditions. Furthermore, the identification of the gaseous systems relates to mono-variable linear behavioral models.

5.2.4.2. *Implementation and analytical tools for gas systems*

This section discusses the analytical tools developed for Schneider PLCs with Unity (we choose to focus on the GCS project). The discussion highlights the solutions proposed to ensure practical implementation of an identification approach in the PLC without recourse to external design programs [CAB 01; CAB 02; CAB 03].

The applications developed use the modeling principles given in Chapter 2.

5.2.4.2.1. Determination of the complexity of the model

The object "RDI_sc" in the UnityV2.1 environment can be used to carry out online testing of the RDIs and RIs.

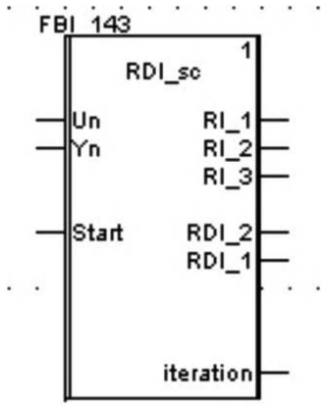


Figure 5.26. *Software object for a PLC to test RDIs and RIs*

The development tool for the PLC enables us to work with objects (DFBs) with particular attributes and functions. UnityV2.1, however, is limited to seven

successive encapsulations in the same DFB object. This restriction means it is possible to determine the RDI up to $i = 2$ and the RI up to $i = 3$. The limitation is due to the tests of the RDIs and RIs, which requires us to calculate the determinants of the information matrix $Q_m(Q_3|_{6\times6})$. Thus, the test enables us to evaluate the complexity up to a third-order process [BOR 01].

NOTE.– The order of industrial processes is usually less than three. Therefore, the RDI and RI test in UnityV2.1 may be sufficient. If the order is greater than this threshold, the GCS project has to use design tools such as, e.g., Matlab/Simulink from Mathworks.

5.2.4.2.2. Parametric identification

Parametric identification is done by way of three DFB objects developed for the project: "MCR_SC" (MCR standing for "*Moindres carrés récursifs*", recursive least squares), "MCE_SC" (for "*Moindres carrés étendus*", extended least squares) and "MVR_SC" (for "*Maximum de vraisemblance récursif*", recursive maximum likelihood).

While these three strategies for parametric identification are not the only possible solutions, these three objects are sufficient to deal with numerous issues encountered for the installations [LAN 02].

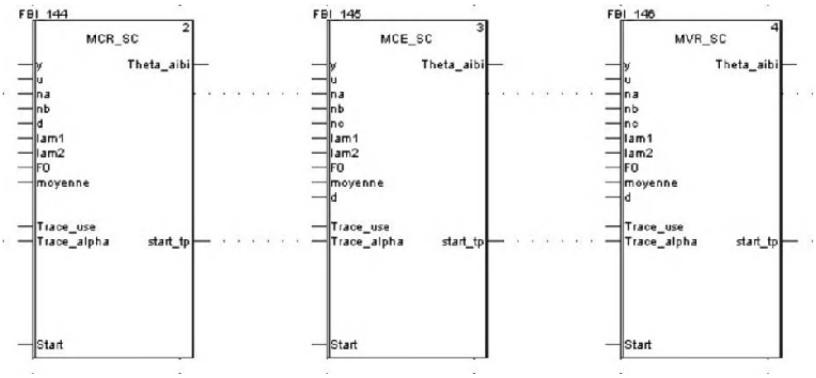


Figure 5.27. *Objects for the PLC for parametric identification with the UnityV2.1 package*

5.2.4.2.3. Stimulation signal

The use of PRBS may not strictly be necessary in order to obtain usable measurements. In practice, it happens that three successive value steps (from U_{min} to U_{max}) are sufficient to obtain good stimulation. The object "SBPAfix_sc" (SBPA

being the equivalent of PRBS) carries out this behavior (we need only specify the stabilization time and the limits). It also enables us to take a reading of the average commands *u* to be subtracted for the purposes of identification.

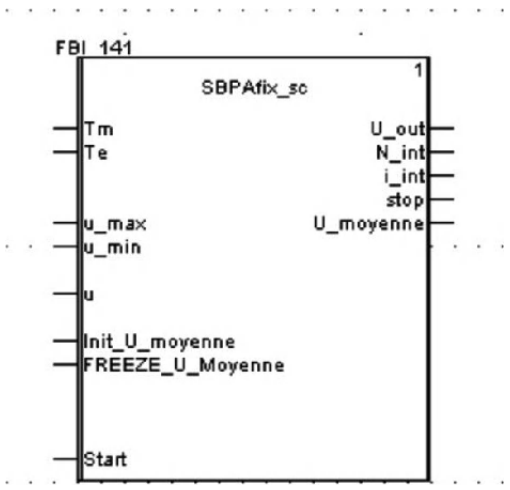


Figure 5.28. *Object for the PLC for generation of three successive value steps to stimulate the system*

5.2.4.2.4. Validation of the model

The validation of the model identified is taken care of by two DFB objects written for this purpose:

– the first, called "SimuSysteme_sc", enables us to simulate an ARMA-type system. It can also be used to construct a CARIMA model. The use of this object enables us to draw a comparison between the process and the model generated by the PLC for the first phase of validation;

– the second, called "RNtype1_sc", carries out the error whitening test for the three methods of identification developed for the GCS project. It is often used in conjunction with the first object.

5.2.4.3. *Application: results of online identification with a PLC for the modeling and identification of the pressure in the ALICE TPC detector*

ALICE TPC (TPC standing for Time Projection Chamber) is a sub-detector of the LHC experiment called ALICE. Its function is to track the trajectories and speed of the charged particles. Comprising a gas chamber, the gas system of ALICE TPC is one of the 23 command-control installations for the GCS project. The gas exiting

the TPC has to be compressed in the pump module in order for there to be effective recycling of the gas for purification and CO_2 absorption. It is also here that the pressure is regulated. The pump in itself does not regulate the pressure.

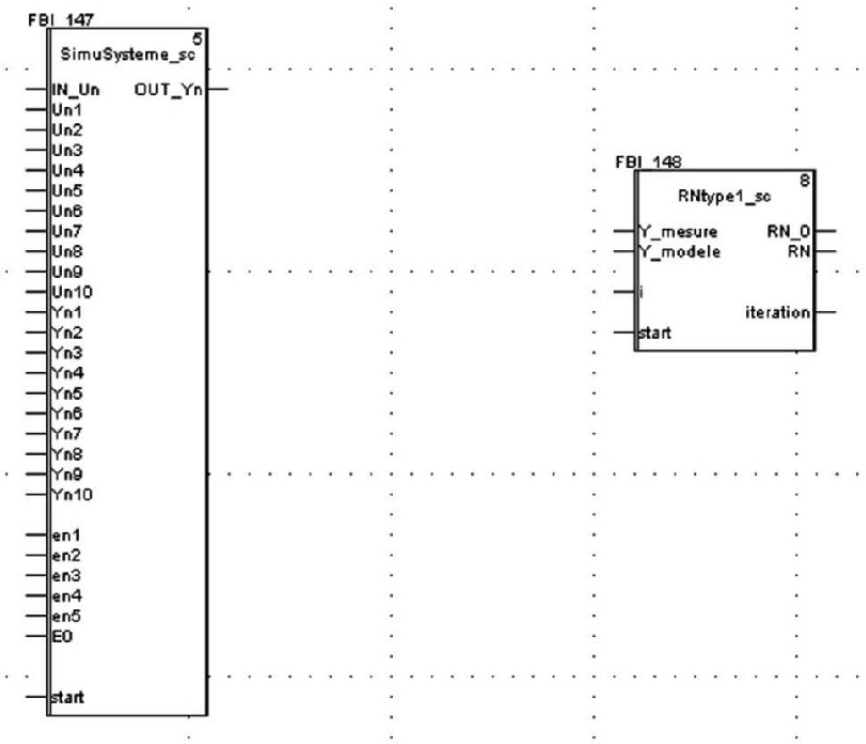


Figure 5.29. *Objects for the PLC for the validation of the model identified*

It is the pump module which takes care of this regulation, by way of the analog bypass valve PV4003. The goal of this regulation of pressure is to obtain a pressure of around 0.5 mbar inside the sub-detector.

The results are of two orders:

– to show a real-world approach to the online identification of a specific regulation of a gas system;

– to compare the results obtained by Unity (online) and Matlab® (offline).

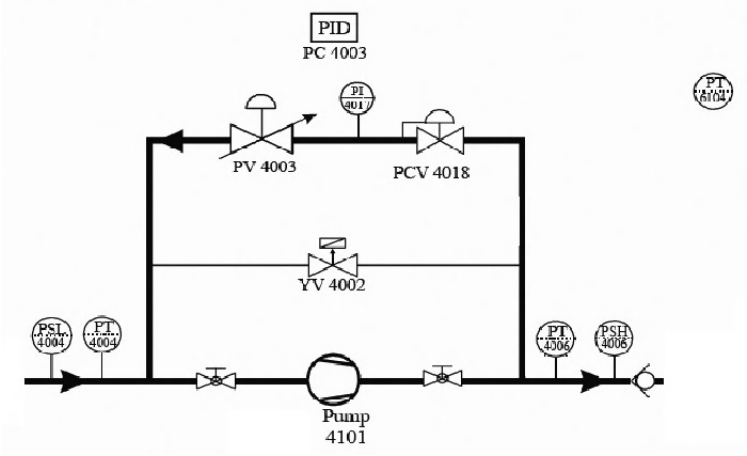


Figure 5.30. *The pump module for the sub-detector ALICE TPC*

Measurements were carried out in order to give a full view of the control of the pressure. The working method consists of manipulating the analog valve PV4003 and collecting measurements of the variations of pressure in the detector on exiting. To work with Matlab, the data acquisition is performed by the acquisition software DataStore.

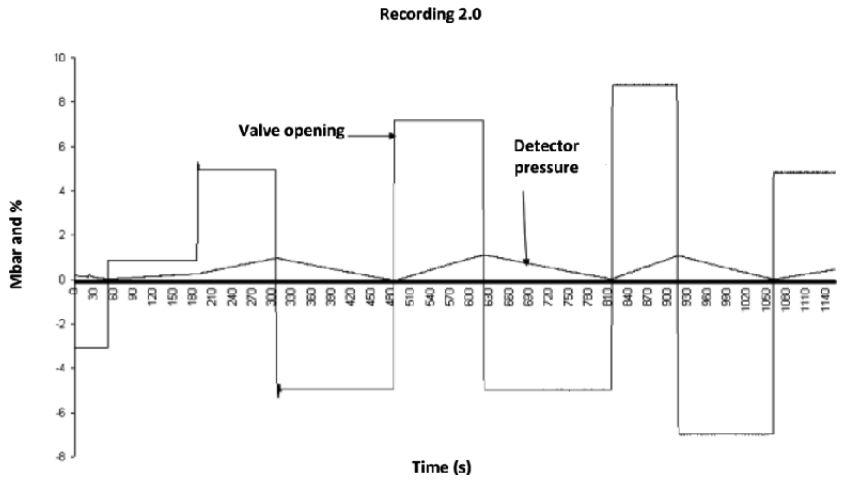


Figure 5.31. *Measurements of pressure in the ALICE TPC detector*

The tests were performed on the detector itself. The constraints were significant, because there was a very real risk of damaging the detector. Hence, the

measurements were performed in the presence of the gas expert, and within the authorized limits in terms of pressure [0 mbar to 1 mbar].

The volume of the TPC detector is 90 m^3. The chamber is filled with CO_2 and the flowrate of the gas is 4 m^3/hr. The pump is in continuous operation with a speed of 34.8 Hz. The average pressure recorded during the measuring campaign is 0.45 mbar with a standard deviation of 0.31 mbar. The average aperture of the valve is 82.13% with a standard deviation of 5.61%.

From these measurements, we can easily see that the system is fundamentally unstable. The system has a critical point of operation (82.13%), from which it diverges in opposite directions within the authorized limits. Above the point of aperture of the valve PV4003, the pressure increases divergently. Below it, the pressure decreases divergently. We can also see on the ground that the measurements are conditioned by the pump. The lower the speed of the pump, the more the critical point of function of the valve decreases. Finally, the system has a pure time-delay of around four seconds.

5.2.4.3.1. Determination of the complexity of the model

An online test of the RDIs was carried out for more than 2,000 iterations [BOR 01]. The below figures show the results obtained with Matlab and Unity.

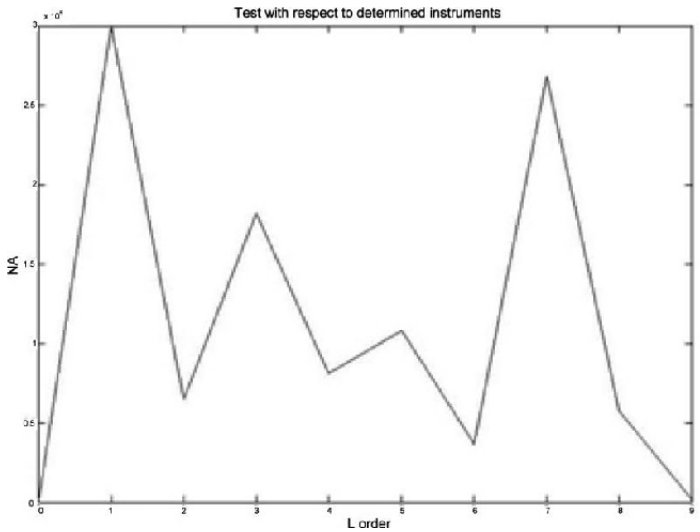


Figure 5.32. *Test of the RDIs using Matlab*

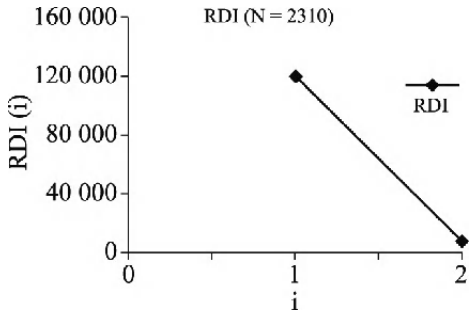


Figure 5.33. *Test of the RDIs using Unity*

These results reveal the order of the model needing to be identified. We can see that $n = 1$ and therefore that $n_a = n_b = n_c = 1$.

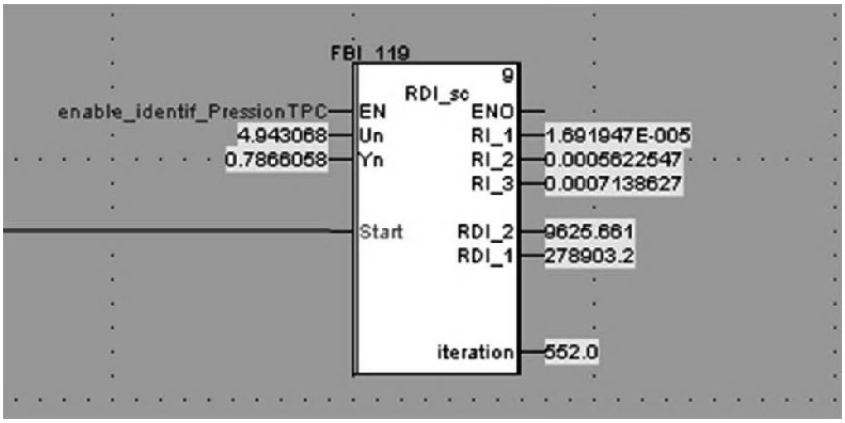


Figure 5.34. *Online determination of the order of the model using Unity for the pressure in the ALICE TPC detector*

5.2.4.3.2. Identification with RLS, ELS and RML

The model is a 1st-order integrator with a pure time-delay of four seconds. This first result helps guide the approach of identification using the three available recursive methods [LAN 02]. Thus, we find the following models with Unity and Matlab:

$$y_{RLS\_Unity}(t) = \left(\frac{0.000342q^{-1}}{1-1.0043q^{-1}}q^{-8}\right)u(t)$$
$$y_{ELS\_Unity}(t) = \left(\frac{0.000567q^{-1}}{1-1.00031q^{-1}}q^{-8}\right)u(t) + \left(\frac{1-0.256q^{-1}}{1-1.00031q^{-1}}\right)e(t) \quad [5.15]$$
$$y_{RML\_Unity}(t) = \left(\frac{0.000066q^{-1}}{1-0.95448q^{-1}}q^{-8}\right)u(t) + \left(\frac{1-0.063q^{-1}}{1-0.95448q^{-1}}\right)e(t)$$

$$y_{MCR\_Matlab}(t) = \left(\frac{0.0005533q^{-1}}{1-0.9998q^{-1}}q^{-8}\right)u(t)$$
$$y_{MCE\_Matlab}(t) = \left(\frac{0.0005476q^{-1}}{1-0.9999q^{-1}}q^{-8}\right)u(t) + \left(\frac{1-0.2411q^{-1}}{1-0.9999q^{-1}}\right)e(t) \quad [5.16]$$
$$y_{MVR\_Matlab}(t) = \left(\frac{0.0005654q^{-1}}{1-0.9476q^{-1}}q^{-8}\right)u(t) + \left(\frac{1-0.0656q^{-1}}{1-0.9476q^{-1}}\right)e(t)$$

It is interesting to note that the identifications carried out with the two different protocols yield similar results. The classic recursive least squares methods, however, appear to generate a difference in terms of the gains found (parameters b_1).

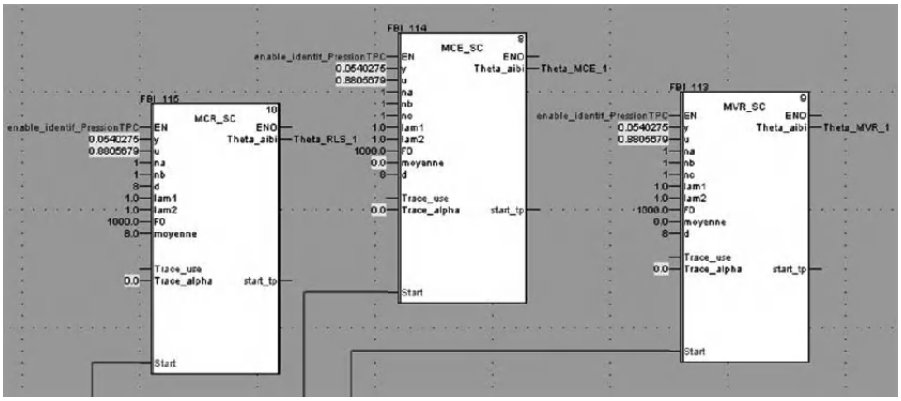


Figure 5.35. *Online identification using the PLC (RLS, ELS, RML) to find the pressure in the ALICE TPC detector*

NOTE.– From these identifications, the nature of the system is labeled: it is an integrator. In order to avoid any divergent behavior (i.e. $a_1 < -1$), it is preferable to take account of this information and set $a_1 = -1$. Thus, the models can be assimilated to:

$$
\begin{aligned}
y_{RLS\_Unity}(t) &= \left(\frac{0.000342q^{-1}}{1-q^{-1}}q^{-8}\right)u(t) \\
y_{ELS\_Unity}(t) &= \left(\frac{0.000567q^{-1}}{1-q^{-1}}q^{-8}\right)u(t)+\left(\frac{1-0.256q^{-1}}{1-q^{-1}}\right)e(t) \\
y_{RML\_Unity}(t) &= \left(\frac{0.000066q^{-1}}{1-q^{-1}}q^{-8}\right)u(t)+\left(\frac{1-0.063q^{-1}}{1-q^{-1}}\right)e(t)
\end{aligned}
\qquad [5.17]
$$

$$
\begin{aligned}
y_{RLS\_Matlab}(t) &= \left(\frac{0.0005533q^{-1}}{1-q^{-1}}q^{-8}\right)u(t) \\
y_{ELS\_Matlab}(t) &= \left(\frac{0.0005476q^{-1}}{1-q^{-1}}q^{-8}\right)u(t)+\left(\frac{1-0.2411q^{-1}}{1-q^{-1}}\right)e(t) \\
y_{RML\_Matlab}(t) &= \left(\frac{0.0005654q^{-1}}{1-q^{-1}}q^{-8}\right)u(t)+\left(\frac{1-0.0656q^{-1}}{1-q^{-1}}\right)e(t)
\end{aligned}
\qquad [5.18]
$$

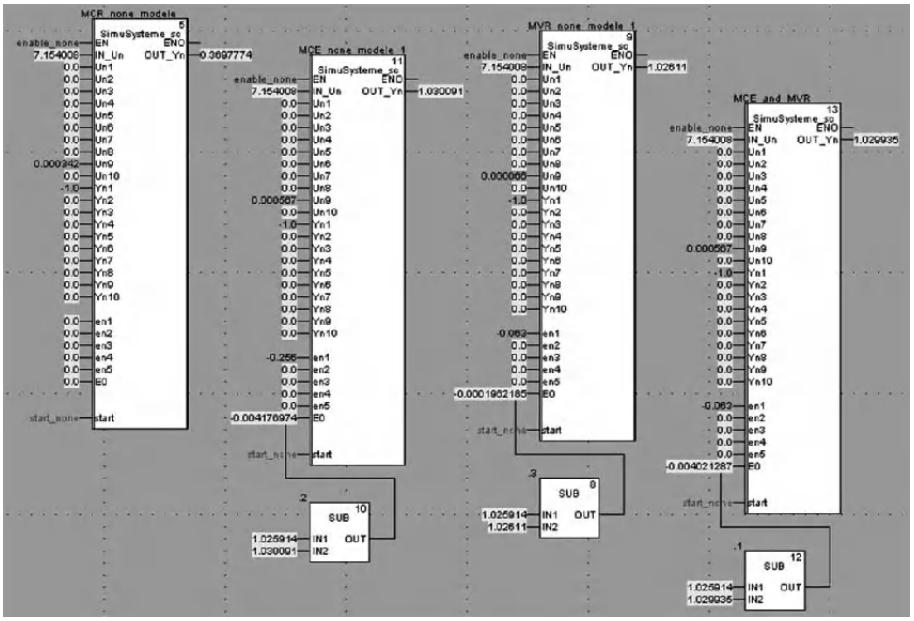


Figure 5.36. *Simulation of the models on a PLC for validation – pressure in the ALICE TPC detector*

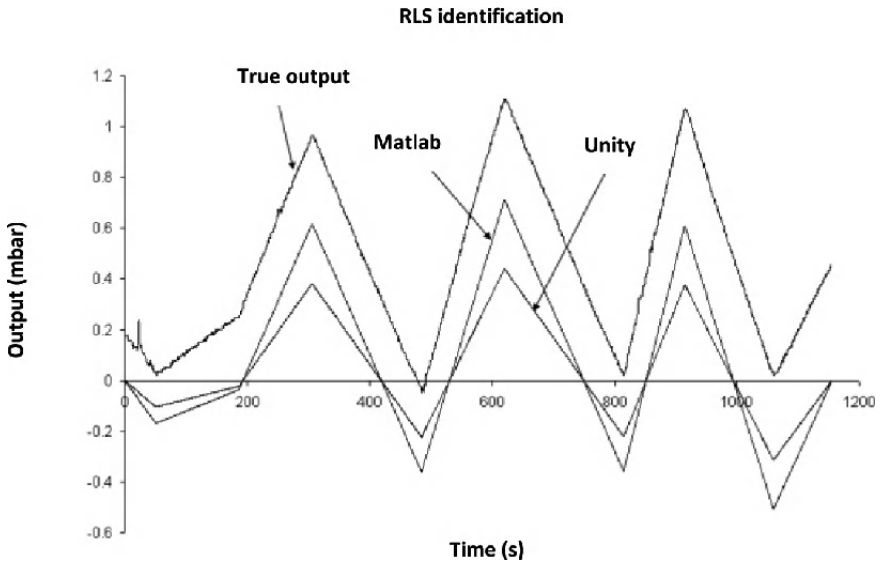


Figure 5.37. *Pressure in the ALICE TPC detector – RLS identification*

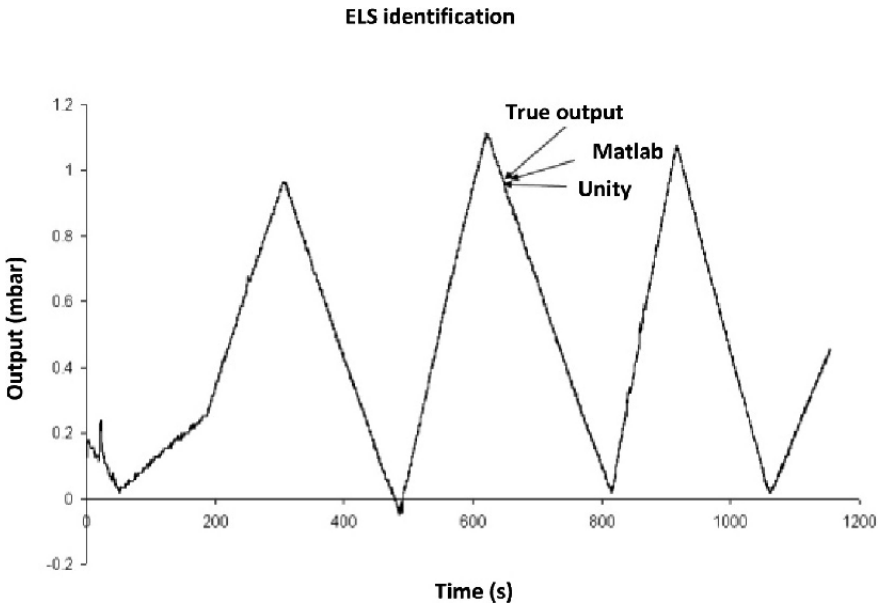


Figure 5.38. *Pressure in the ALICE TPC detector – ELS identification*

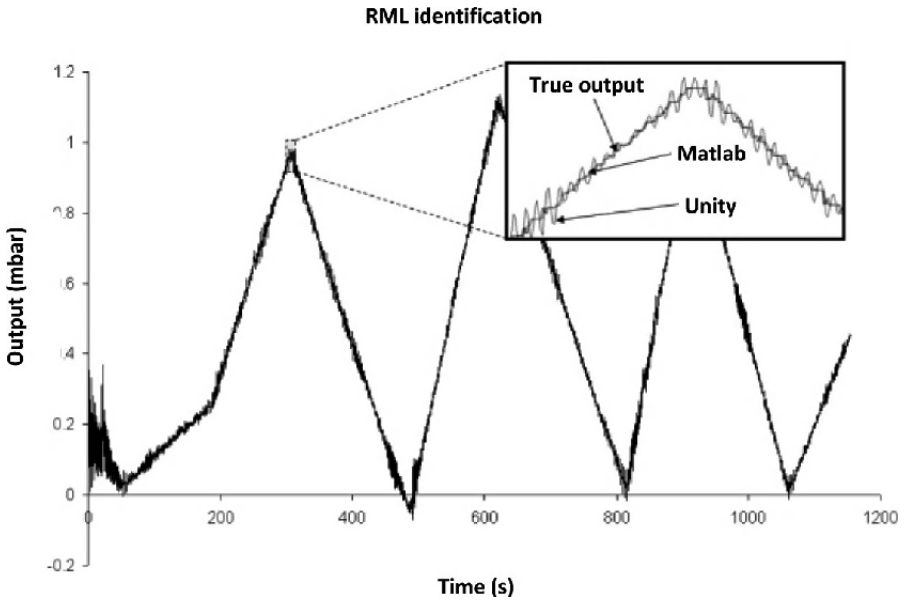


Figure 5.39. *Pressure in the ALICE TPC detector – RML identification*

The models identified here demonstrate that the RLS method is not appropriate. The ELS and RML methods, however, yield far more satisfactory results in terms of the differences between the model and the process than the previous figures.

NOTE.– It is important to understand that the identifications are not strictly identical between Unity and Matlab. An essential point must be understood: the PLC as it is functions with a cycle of execution. This means that the "loop" of the program being executed is not synchronous but instead depends on the size of the application. Thus, if we set a particular sampling time for the parametric identification, we are only sure to have a constant sample, with an error of one cycle of execution more or less. As the identification is not strictly rhythmic with a precise sampling frequency, it is unsurprising to find differences between the results given by Unity and Matlab.

5.2.4.3.3. Validation

The simulations of the models obtained by identification with the ELS and RML methods are highly satisfactory, and demonstrate the difficulty of differentiating the two curves. The error whitening test in Unity therefore seems to be required as an aid to decision-making and for the validation of the model [BOR 01].

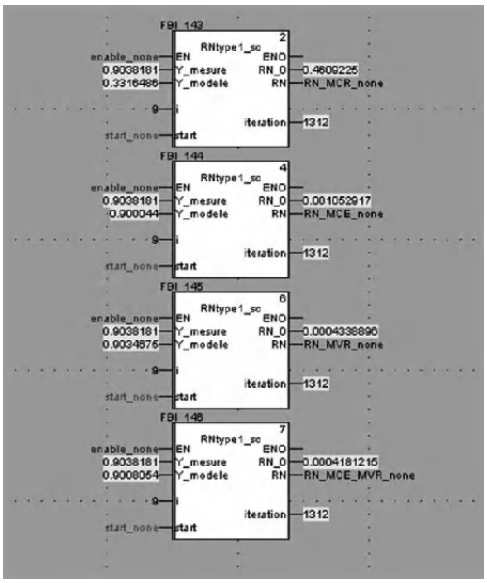


Figure 5.40. *Online error whitening test with a PLC – pressure in the ALICE TPC detector*

	ELS	RML
RN(0)	0.00107	0.000456
RN(1)	2.71^{E}-05	5.43^{E}-06
RN(2)	1.19^{E}-05	1.50^{E}-05
RN(3)	7.34^{E}-06	1.30^{E}-05
RN(4)	5.36^{E}-08	1.24^{E}-05
RN(5)	1.53^{E}-06	1.52^{E}-05
RN(6)	1.29^{E}-06	1.52^{E}-05
RN(7)	2.31^{E}-06	1.31^{E}-05
RN(8)	5.40^{E}-06	1.95^{E}-95

Table 5.1. *Results of the error whitening test with a PLC for the pressure in the ALICE TPC detector*

The prediction error whitening test applied shows that both methods are acceptable (as the covariances all fall below the theoretical threshold value of 0.0451). However, these results cannot help us decide *which* of these two models is the most relevant to use. In order to help come to a decision on this point, it is wise

to look at the curves of the convergences of the parameters for the ELS and RML methods.

As the value of the parameter a_1 is known (-1), we can look directly at the changes in b_1 and c_1 shown in Figures 5.41 – 5.44.

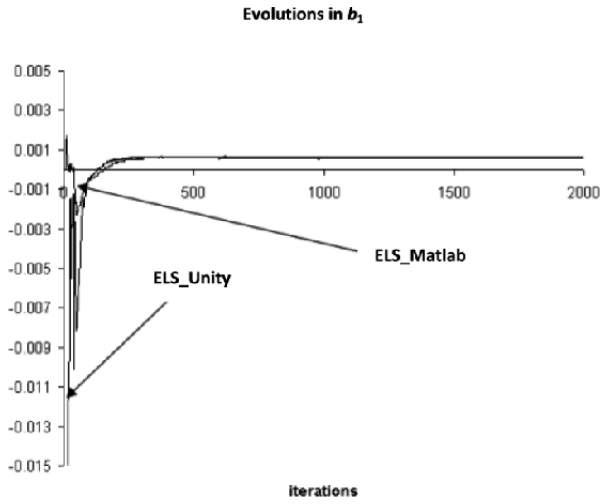


Figure 5.41. *Pressure in the ALICE TPC detector – ELS identification, evolution of b_1*

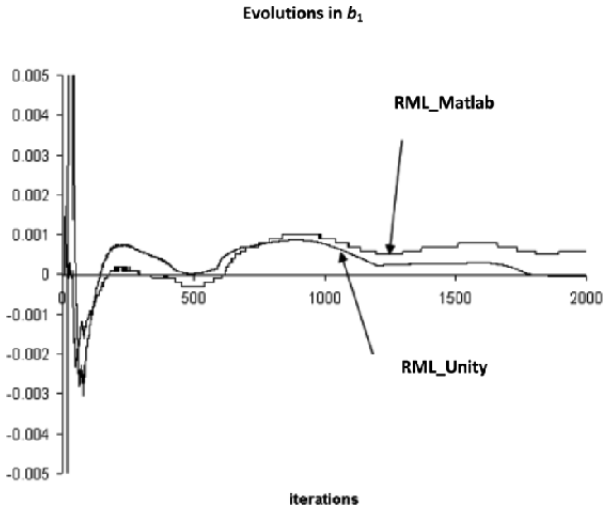


Figure 5.42. *Pressure in the ALICE TPC detector – RML identification, evolution of b_1*

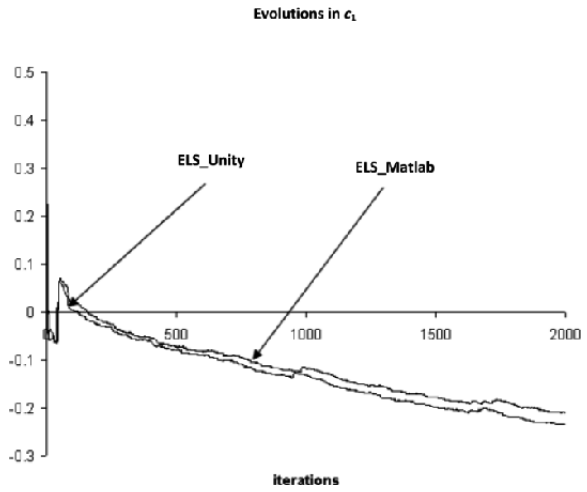


Figure 5.43. *Pressure in the ALICE TPC detector – ELS identification, evolution of c_1*

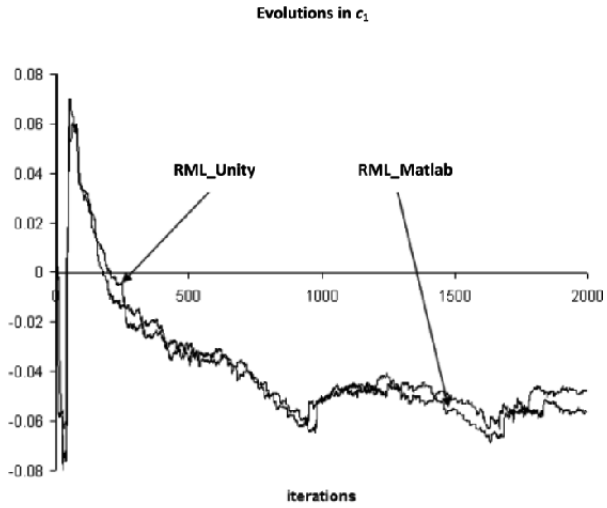


Figure 5.44. *Pressure in the ALICE TPC detector – RML identification, evolution of c_1*

The above curves demonstrate a number of points:

– the evolutions in b_1 and c_1 follow trends similar to those obtained using Matlab;

– the ELS method exhibits rapid convergence toward a stable parameter b_1;

– the RML method appears to show relative convergence for the parameter c_1.

5.2.4.3.4. Final choice

In conclusion, on the basis of the studies described above, the choice was made to use a coefficient b_1 identified by the ELS method and a coefficient c_1 identified by the RML method. The relevant model is obtained by using the combined ELS + RML method. This enabled us to draw up a model whose expression is:

$$y_{ELS+RML}(t) = \left(\frac{0.000567q^{-1}}{1-q^{-1}} q^{-8} \right) u(t) + \left(\frac{1-0.05q^{-1}}{1-q^{-1}} \right) e(t) \qquad [5.19]$$

5.3. Conclusion

The process of identification/modeling is an essential stage in the successful synthesis of regulator loops for gas systems. The GCS project uses Schneider PLCs. The use of function blocks in the PLC enables us to introduce advanced reusable algorithms. In that vein, we have developed a set of DFB components for online identification and modeling. In spite of certain limitations inherent to the technology of the PLC, the identifications are relevant. Thanks to a concrete example from an onsite system (i.e. the pressure in the ALICE TPC detector), we note that the identification/modeling is ensured to yield results similar to those obtained by the simulation in Matlab.

5.4. Bibliography

[BOR 01] BORNE P., BEN ABDENNOUR R., KSOURI M., *et al*., *Identification et commande numérique des procédés industriels*, Technip, Paris, 2001.

[CAB 01] CABARET S., COPPIER H., RACHID A., *et al*., "GPC and PFC with a Programmable Logic Controller", *MMAR*, Szczecin, Poland, August 2007.

[CAB 02] CABARET S., "From identifying and modeling to control: the MultiController, an advanced control loop strategy for cryogenic systems at CERN", *EUROSIM*, Ljubljana, Slovenia, September 2007.

[CAB 03] CABARET S., COPPIER H., RACHID A., *et al*., "MultiController: an object programming approach to introduce advanced control algorithms for the GCS large scale project", *ICALEPCS*, Knoxville, United States, October 2007.

[LAN 02] LANDAU I.D., *Commande des systèmes – conception, identification et mise en œuvre*, Hermès, Paris, 2002.

[MEN 81] MENAHEM M., *Le correcteur à simulation – Description et utilisation du modèle CZ*, Contrôle Bailey, 1981.

Chapter 6

Application to Vehicles

6.1. Introduction

The applications of automation engineering in the domain of transport, and particularly road transport, have for decades occupied an important place, and numerous laboratories, researchers and industrial players are involved in this field.

This chapter focuses on two types of vehicle: hydraulic diggers and cars.

First of all, we shall present the excavator-loader that is the subject of our discussion. Our aim is to research and design an onboard control/command to assist the drivers of excavator-loaders during the phases of guiding the movements of the tools, in urban and suburban settings. The goal is not to design entirely autonomous machines – i.e. completely automated machines which, with no human intervention, would be capable of carrying out the cycles of perception, planning and execution necessary for the realization of a variety of tasks. Rather our objective is to present a few designs for posts to guide this type of building machinery:

– so that the actions of steering become more intuitive from the point of view of the drivers;

– to decrease the time needed to train the drivers;

– to improve the information fed to the drivers during the performance of the tasks.

Then, we shall turn our attention to the dynamics of the vehicle, considering different control laws to assist car driving. Firstly, the modeling of the vehicle

---

Chapter written by Elie KAFROUNI and Mohammed CHADLI.

dynamics is touched upon, with the discussion encompassing a number of different models widely used in the literature (model with two degrees of freedom, four degrees of freedom, etc.) in both linear and nonlinear forms. These models are then validated by an industrial simulator and then experimentally. The final section is given over to robust control of the vehicle's dynamics with an uncertain multi-model representation. Different control laws are applied: the first is reconstructed state feedback control; the second is an H∞-type law and the third control law is an observer-based robust control.

6.2. Hydraulic excavator-loader

Hydraulic excavator-loaders are building machines designed to efficiently carry out tasks in diverse, non-structured environments and on very different types of terrain as well, in varying weather conditions. An excavator-loader comprises a moving platform (the chassis) which is moved by caterpillar tracks or wheels. The chassis supports a directable turret or driver's cab, to which is connected an articulated mechanism (an arm), at the end of which a working tool can be attached. The combination of the chassis, turret and arm is comparable to a mobile manipulator. With a standard excavator-loader, the arm is formed of three rotoid joints with parallel axes. The tool being used can be changed, so the same machine can be used to perform different tasks.

The activities of excavator-loaders can be characterized as fairly repetitive sequences of elementary tasks, the realization of which has to be adapted to the changing properties of the environments. The types of task performed by these machines are very diverse: mass excavation of ground, precision landscaping, digging of different types of trenches (sloped, curved, multi-level, etc.), digging of channels, leveling of ground, movement of objects, movement of the platform from one place to another, gripping and unloading of materials, alongside other more complex tasks.

With the exception of certain automated devices to protect the thermal motor (the single onboard source of energy) and the hydraulic distribution circuit, these machines do not include any devices to help or substitute the driver in performing the tasks which he may be asked to carry out. Driving support for these machines could also open the door to the possibility of improving yield, productivity and safety, as well as reducing the fatigue of the drivers.

The aim of this chapter, as stated above, is not to design entirely autonomous machines: i.e. machines able to carry out the cycles of perception, planning and execution necessary for the realization of a variety of tasks, completely independently of human intervention. The set goal of this work was to examine the

possibilities for use of techniques in robotics in order to modify the current design of command posts for this type of building machinery:

– so that the actions of steering become more intuitive from the drivers' point of view;

– to decrease the time needed to train the drivers;

– to improve the information fed to the drivers during the performance of the tasks.

In other words, the objective was to research and design an onboard control/command system to help excavator-loader drivers as they guide the movements of the tools, in urban and suburban environments.

6.2.1. *Conventional manual piloting*

All hydraulic excavator-loaders used on construction sites are controlled manually by a driver. The guidance of the movements of the platform, turret and the different parts of the arm is based on individually controlling the speed of each hydraulic actuator (piston or motor). This piloting system is made up of an interface comprising levers, joysticks and/or pedals. A degree of freedom of a joystick enables the driver to control the apertures of the orifices for pressurized oil to pass through. The flowrate of oil feeding an actuator generally depends on the aperture and the difference in pressure upstream and down of the orifice in question, which alters the speed of motion of a part of the arm (see Figure 6.1).

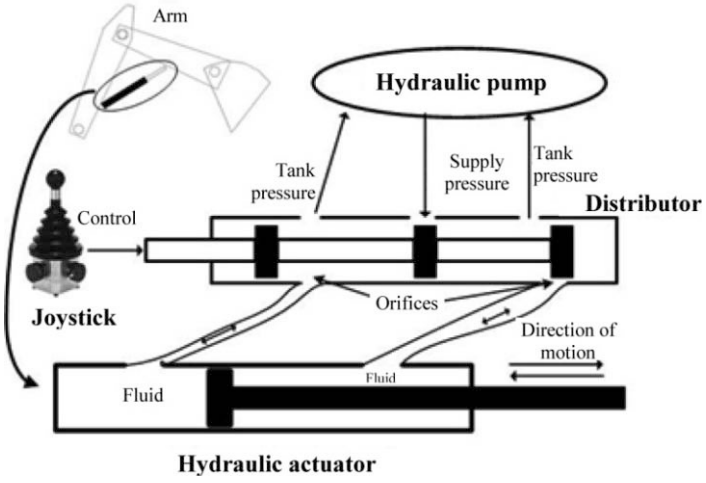


Figure 6.1. *Principle of movement of a part of the arm*

6.3. Principle of movement of a part of the arm

6.3.1. *Role of the drivers*

Typically, at any one time, the driver has to make the decision to alter the speeds of the platform, turret and the parts of the arm in order to guide the movements of the tool to perform the task at hand. Thus, his visual perception of the "instantaneous" situation of the tool leads him to "constantly" modify it on the basis of the goal being sought and his view of the environment. In cases where the driver does not have a clear view of the environment, one or more people present on the construction site provide him with secondary information which helps him to make decisions in order to accomplish his goal.

The current *modus operandi* requires the drivers to have experience, dexterity and an aptitude for visually assessing the position or speed of the tool. In this mode of control during the realization of a task, the drivers must have mentally established the link between the maneuvers of the joysticks and the directions of movement of the tools. It is also helpful to note that this link depends not only on the correspondence between the positions of the joysticks and the speeds of the actuators (flowrate of the hydraulic fluid), but also on the specific design properties of the hydraulic circuit controlling the actuators.

In addition, there is no standardization of the piloting interfaces. When the drivers have to change machines, they also have to adapt to new functionalities of the interface. The technology used to guide the various actuators from the levers and the pedals is primarily hydraulic, but the circuits for distribution of pressurized fluid are not standardized.

For instance, when leveling, the driver performs three sequential main phases of guidance of the movements of a tool (see Figure 6.2). With the conventional devices for manual control of the speed of the pistons:

– phase 1: after having correctly positioned the stick in relation to the boom, the driver lowers the tool until it touches the ground: simultaneous guidance of the descent of the boom and preparation of the "angle of attack" of the tool;

– phase 2: the driver coordinates the movements of the stick and the boom to create "horizontal" movement in the desired direction, whilst maintaining the tool at the same height and in the same orientation;

– phase 3: simultaneous control of the ascent of the boom and rotation of the tool in order to free it before transferring it to an unloading site.

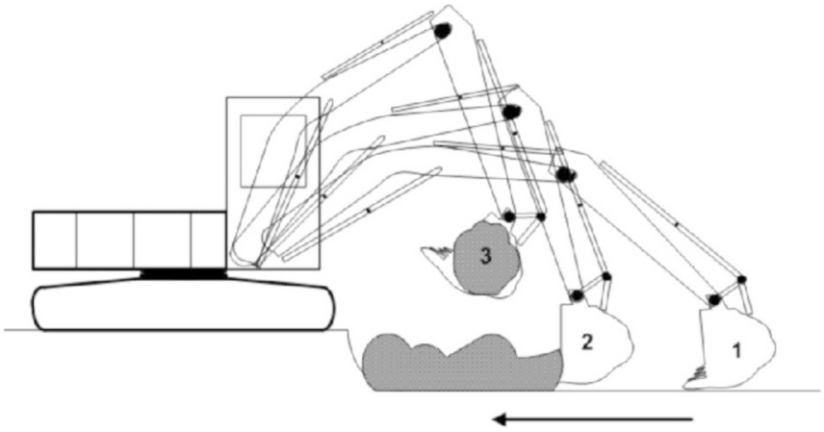


Figure 6.2. *Performance of the task of leveling*

6.3.2. *Objectives*

The general objectives were defined in collaboration with the company MECALAC, who design and market machinery specifically built for public service works. These machines are designed to be multifunctional, better adapted than the conventional excavator-loaders to perform public service works in urban or suburban spaces. In comparison to the conventional excavator-loaders, the peculiarity of these machines is that their articulated arm has five main parts instead of the conventional three. The main specific points of these machines are (see Figure 6.3):

– an articulated chassis, with the turret connected to the front of the chassis;

– a solid engine block at the rear of the chassis;

– an articulated arm, connected to the turret but somewhat removed from it;

– the arm or carrier is made of five main parts, connected two by two by rotoid (pivoting) joints whose axes are not all parallel.

These particular design points cause inexperienced drivers difficulty in guiding the movements of the tools, and in detecting in time the possibility of collision between any part of the arm and the housing protecting the engine.

In order to facilitate the drivers' work and minimize the number of maneuvers they need to carry out, researchers have developed and put forward support systems. Most of these employ the technique of resolved motion-rate control [WHI 60] to design new guidance systems based on direct control of the rate of motion of the

tool in relation to one or more particular systems of coordinates, chosen on the basis of the tasks needing to be performed. Thereby, the drivers no longer need to worry about coordinating the movements of the actuators.

Figure 6.3. *General view of a wheeled MECALAC 12MXT machine*

In order to make the job easier for the drivers of such machines (whether they are novices or seasoned professionals), the idea was to design an interface enabling them to pilot the movements of the tools by directly setting:

– the velocity vector (direction and modulus) of translational movement of the tool holder;

– the velocity vector of rotational movement of the tool holder.

With resolved motion-rate control [WHI 60], the driver sets:

– phase 1: the direction of the velocity vector of translational movement of the tool and, gradually, the modulus of that vector and that of its speed of rotation in relation to a marker linked to the turret;

– phase 2: after making contact with the ground, the direction of the speed of translational movement is altered, and the motion imposed on the tool becomes horizontal;

– phase 3: modification of the speed of rotation of the tool, enabling it to be withdrawn, combined with a modification of the direction of its translational velocity, and the motion of the tool becomes vertical.

This new interface should relieve drivers of the task of coordinating the movements of each part of the arm and thus allow them to concentrate solely on the movement to give to the tool holder, and therefore the tools, in accordance with their own visual perception of the execution of the tasks.

Thus, the aim is to design and develop an onboard control system whose overall functional structure is illustrated in Figure 6.4. In other words, we aim to control the rates of motion of the hydraulic actuators in a coordinated manner, in order to make movements with the tool holder, imposed by the drivers.

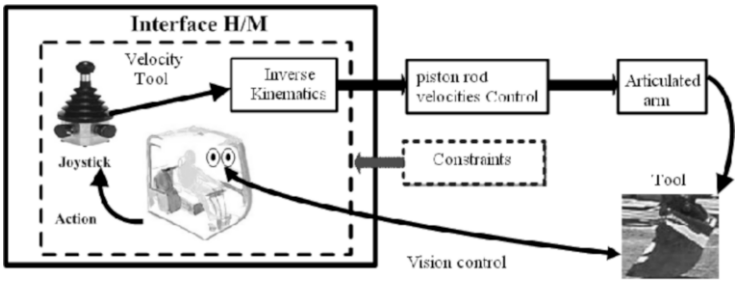


Figure 6.4. *General block diagram of the new system*

With our proposed solution, in order to perform the task of leveling as described above, the driver imposes (see Figure 6.5): descent (stage 1), horizontal motion (stage 2), rotation (stage 3) and re-ascent (stage 4).

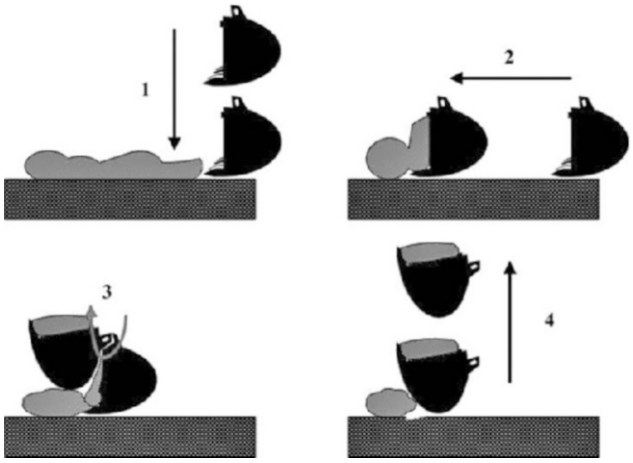


Figure 6.5. *Performance of leveling with the proposed method. The arrows indicate the directions of movement of the scoop during the various phases*

This goal involves carrying out two independent types of work: one relating to the "driver/machine" interface system; the other to the development of algorithms to control the speeds of the piston stems.

In relation to the current control posts, this requires significant modifications to be made in terms of the conventional interface between the driver, the joysticks and the pedals. Then, the instructions given by the driver regarding speed need to be converted into instructions for the speed of the piston stems, with account taken of the design constraints, both geometric (minimum and maximum extensions of the pistons) and hydraulic (the speeds of the pistons depend on the distribution of the maximum available power).

A number of modes of "exploitation" of the configurations of the mechanical arm can be envisaged, depending on the tasks or the phases of the tasks being performed. Indeed:

– the arm can be used like a conventional arm – i.e. with three out of the five parts moving;

– the arm can also be used as a "coplanar" arm, with a degree of redundancy equal to 1 – i.e. with four out of the five parts moving;

– the tool holder and the part of the arm to which it is attached can be moved in a different plane to that which is defined by the first two parts. All combinations of movements are possible;

– the machine can be used as a conventional loader or to loader/transport palettes.

Thus, the objective is to implement resolved motion-rate control in the case of the arms on MECALAC machines:

– to define reference coordinate systems that can be used to express the components of the kinematic torsor of the tool holder;

– to transform these components into speed commands that can be obeyed by the hydraulic actuators:

- to take account of the geometric limits,

- to take account of the limits imposed at these speeds by the hydraulics;

– to put forward one or more interface devices, with several degrees of freedom, to be manipulated by the drivers in order to modify the speed commands given to the tool holder, online (in real time). Each of these degrees of freedom must be put in correspondence with one of the components of the kinematic torsor.

6.3.3. *Functional specification of the interface*

In order to integrate a piece of driving support software enabling the drivers to guide the movements of the tool holder, based on the principle of resolved motion-rate control, it is very important to specify the main functions of a new interface to guide the movements of the tool holder in the workload of the articulated arm. This approach requires us to automatically implement the coordinated control of the movements of the pistons.

The piloting interface and the driver are in the turret. It is therefore natural to uncouple the guidance of the rotation of the turret from the guidance of the movements of the tool holder.

The movements of the tool holder are guided by two independent devices:

– one controls the rotation of the turret;

– the other controls the movements of the tool holder in relation to the turret (and thus in relation to the driver).

At all times, the drivers must be able to alter the direction and intensity of the translational movement and rotational speed of the tool holder. The guidance orders given by the driver need to be converted into control signals for the speeds of the connections of the pistons.

To begin with, we shall describe the geometry of the arm, the direct transformation between the articulation speeds and the torque of the velocity torsor, as well as the specifications of the MECALAC 12MXT machine. Then, after giving a description of the current pilot interface, we shall give a description of the functional specifications of a new one.

6.3.3.1. *Mechanical arms of MECALAC machines*

Unlike conventional articulated mechanisms, comprising three main parts, which are fitted to most excavator-loaders, the mechanism bearing the tools of MECALAC machines has five main parts (see Figure 6.6):

– a boom;

– an after-boom;

– a part called the “swing”;

– a stick;

– a tool holder.

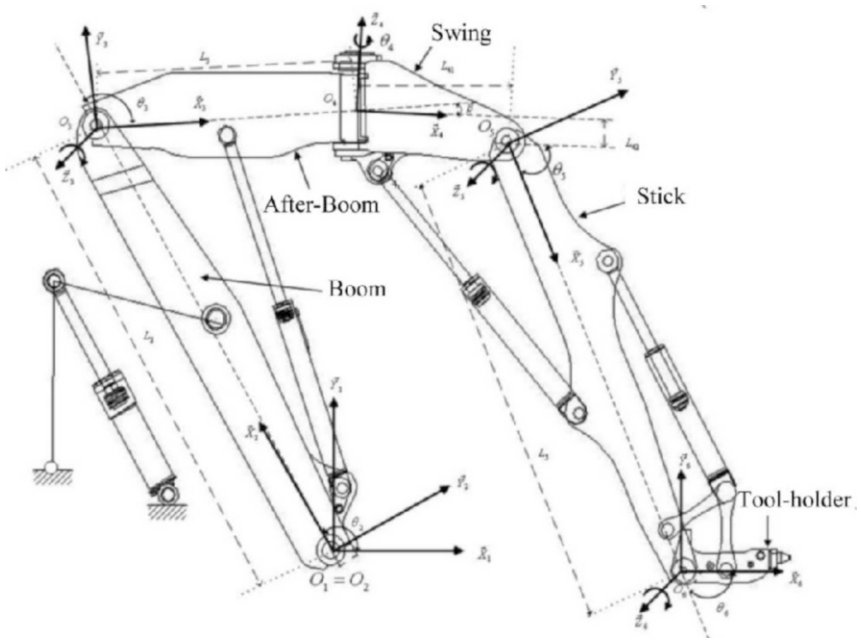


Figure 6.6. *Representation of the arm and the labels attached to the five main parts*

These five parts are linked by pivot joints, forming a simple chain. The axes of rotation of the boom and after-boom are parallel and orthogonal to the axis of rotation of the turret. The axis of rotation of the swing is almost orthogonal to the axis of rotation of the after-boom, while the axes of rotation of the stick and tool holder are both orthogonal to the axis of the swing and are parallel to one another. Each of these joints is moved by way of a closed mechanical chain including a hydraulic piston.

The cockpit (i.e. the turret) is set laterally in relation to the arm. The whole ensemble is connected to a platform by a pivot joint (with a vertical axis) which facilitates up to 360° rotation. This design, whereby the engine block is disconnected from the turret and the arm, reduces the space taken up by a machine without reducing the working area needed to carry out roadside works (see Figure 6.7a), but means there is an added possibility of collision between part of the arm and the chassis, the engine or the wheels.

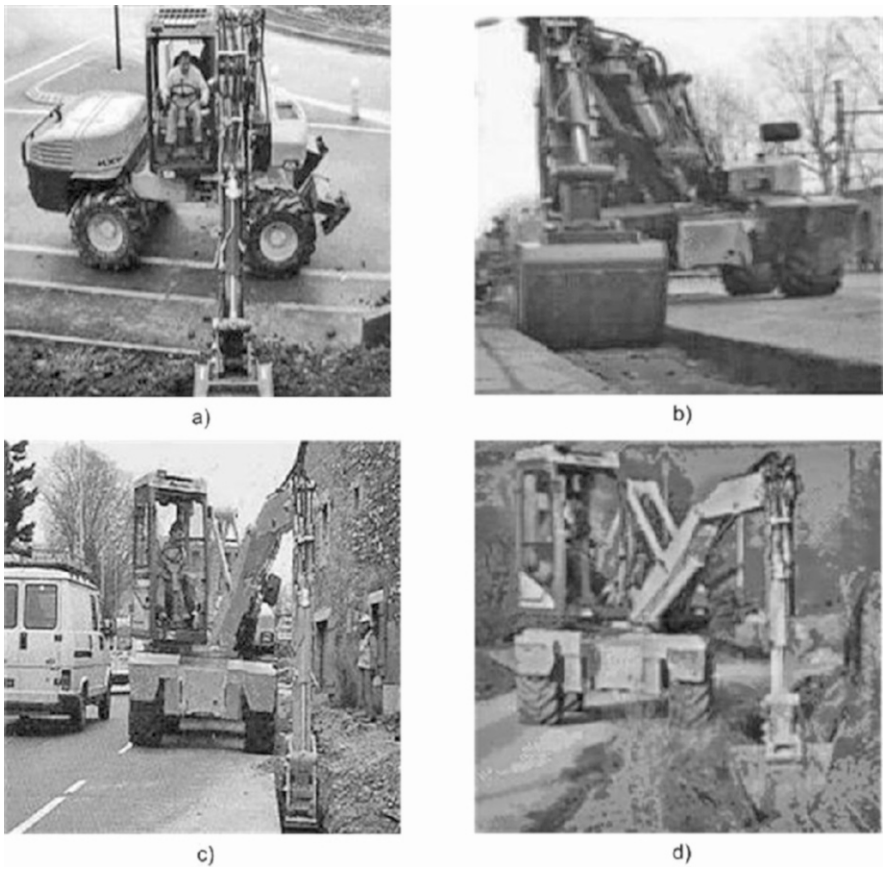


Figure 6.7. *Illustration of some working situations for the 12MXT excavator-loader: (a) view of the turret independent of the engine; (b) leveling; (c) trench digging; (d) leveling*

6.3.3.1.1. Geometric description of the arrangement of the parts

Beginning with the cockpit, which we consider as a point of reference, and drawing inspiration from the notation conventions advanced by Denavit-Hartenberg [CRA 86], a coordinate system is attached to each of the five main parts (see Figure 6.6). Using these coordinate systems, a set of five articular variables $(\theta_2, \theta_3, \theta_4, \theta_5, \theta_6,)$ is defined, with each of these defining the joint position of a part of the arm in relation to the previous one (see Figure 6.6). This way of working enables us to define the general expression of the geometric transformation which takes us from the referential framework $(\mathbb{R}_{i-1})$ to $(\mathbb{R}_i)$, in the form of a

4×4 matrix. Given that all the joints are rotary joints with a single degree of freedom, we get:

$$^{i-1}T_i = \begin{bmatrix} ^{i-1}R_i(\theta_i) & ^{i-1}P_i \\ 0 & 1 \end{bmatrix} \quad [6.1]$$

where:

– $^{i-1}P_i$ = vector of O_i in relation to $(\mathbb{R}_{i-1})$;

– $^{i-1}R_i(\theta_i)$ = orientation matrix of $(\mathbb{R}_i)$ in relation to $(\mathbb{R}_{i-1})$.

The referential framework $\{\mathbb{R}_1 \equiv (O_1; \vec{X}_1, \vec{Y}_1, \vec{Z}_1)\}$ or coordinate system attached to the cockpit is chosen such that:

– its origin is identical to that of the framework $\{\mathbb{R}_2 \equiv (O_2; \vec{X}_2, \vec{Y}_2, \vec{Z}_2)\}$ attached to the boom;

– its unitary direction $\vec{Z}_1$ is identical to $\vec{Z}_2$, located on the axis of the turret/boom joint.

In other words, using the homogeneous transformation matrices and the properties of their products [CRA 86; DOM 88], the geometric transformation from $\{\mathbb{R}_1\}$ to $\{\mathbb{R}_2\}$ is represented by:

$$^1T_2(\theta_2) = Rot(Z, \theta_2) \quad [6.2]$$

Continuing in this manner until we reach the after-boom, the situation (position and orientation) of $\{\mathbb{R}_3 \equiv (O_3; \vec{X}_3, \vec{Y}_3, \vec{Z}_3)\}$ in relation to $\{\mathbb{R}_1\}$ is given by:

$$^1T_3(\theta_2, \theta_3) = Rot(Z, \theta_2) Tr(X, L_2) Rot(Z, \theta_3) \quad [6.3]$$

In this expression, L_2 is the distance between the two pivot joints present on the boom.

The direction of the axis of the pivot joint which connects the swing to the after-boom is not located in a parallel plane to that defined by $(\vec{Y}_3, \vec{Z}_3)$. The geometric

transformation from $\{\mathbb{R}_3\}$ to $\{\mathbb{R}_4 \equiv (O_4; \vec{X}_4, \vec{Y}_4, \vec{Z}_4)\}$, attached to the swing, is such that:

$$^3T_4(\theta_4) = Rot(X, -\pi/2)Rot(Y, \beta)Tr(X, L_3)Rot(Z, \theta_4) \quad [6.4]$$

where the angle β is constant $(3.5°)$.

Finally, the geometric transformation from $\{\mathbb{R}_4\}$ to $\{\mathbb{R}_6\}$, the referential attached to the tool holder, is such that:

$$^4T_6 = Tr(X, L_{41})Tr(Z, L_{42})Rot(X, \pi/2)Rot(Z, \theta_5)Tr(X, L_5)Rot(Z, \theta_6) \quad [6.5]$$

In relation to the referential $\{\mathbb{R}_1\}$, the coordinates of the position of the origin of the referential linked to the tool holder are such that:

$$^1P_{6x} = l_2C_2 + l_3C_{23} + l_{41}C_4C_{23\beta} - l_{42}S_{23\beta} + l_5C_{23\beta}C_4C_5 - l_5S_{23\beta}S_5$$
$$^1P_{6y} = l_2S_2 + l_3S_{23} + l_{41}C_4S_{23\beta} + l_{42}C_{23\beta} + l_5S_{23\beta}C_4C_5 + l_5C_{23\beta}S_5$$
$$^1P_{6z} = l_{41}S_4 - l_5S_4C_5$$

6.3.3.1.2. Direct kinematic model

The kinematic model of the arm completes the geometric modeling by establishing the relations between the articular velocities and the velocity torsor $(\vec{V}_6(O_6), \vec{\Omega}_6)$ of the tool holder in relation to a fixed referential. The first component of the torsor denotes the translational velocity vector of the origin of the referential attached to the tool holder in relation to the origin of the referential attached to the turret. The second denotes the rotational velocity vector of this part in relation to the reference framework. In other words, we need to establish the transformation from the articular velocity vector $\dot{\underline{\theta}} \in ^5$ to the vector $\underline{V} \in \mathbb{R}^6$ of the six components of velocity of the tool holder:

$$\dot{\underline{\theta}}^T = [\dot{\theta}_2 \quad \dot{\theta}_3 \quad \dot{\theta}_4 \quad \dot{\theta}_5 \quad \dot{\theta}_6]$$
$$\underline{V}^T = [V_X \quad V_Y \quad V_Z \quad \Omega_X \quad \Omega_Y \quad \Omega_Z] \quad [6.6]$$
$$\underline{V} = f(\underline{\theta}, \dot{\underline{\theta}})$$

The method adopted to construct this model is that which is known, in the literature, as the iterative method [CRA 86]. Beginning with the part chosen as a

reference point – here the turret – the method consists of applying the theorem of composition of the velocity vectors, from one part to another:

$$\begin{aligned}\vec{\Omega}_{i+1} &= \vec{\Omega}_i + \vec{\Omega}_{i+1/i} \\ \vec{V}(O_{i+1}) &= \vec{V}(O_i) + \vec{\Omega}_i \wedge \overrightarrow{O_i O}_{i+1}\end{aligned} \quad [6.7]$$

In these expressions, $\vec{\Omega}_{i+1/i}$ denotes the rotational velocity vector of the part (C_{i+1}) in relation to the part (C_i).

The rotational velocity vector of the part, $\vec{\Omega}_i$, can be represented by a vector ${}^{i}\omega_i \in \mathbb{R}^{3\times 1}$ which expresses the three components of $\vec{\Omega}_i$ in the referential attached to the part (C_i), considered on the basis of the representation of $\vec{\Omega}_{i-1}$ in the same framework:

$${}^{i}\omega_i = {}^{i}R_{i-1}(\theta_i)\,{}^{i-1}\omega_{i-1} + \dot{\theta}_i\,{}^{i}Z_i \quad [6.8]$$

In this expression, ${}^{i}R_{i-1}(\theta_i)$ denotes the rotational matrix which defines the orientation of the referential linked to the part (C_{i-1}) in relation to the linked to the part (C_i).

By setting:

$$\phi_2 = \theta_2\,,\ \phi_3 = \phi_2 + \theta_3\,,\ \phi_4 = [\phi_3 - \beta \quad \theta_4],\ \phi_5 = [\phi_3 \quad \theta_4 \quad \theta_5],\ \phi_6 = [\phi_3 \quad \theta_4 \quad \theta_5 + \theta_6] \quad [6.9]$$

and taking account of the geometric properties of the arrangement of the parts of the arm in question (see section 6.3.3.1.1) and the two expressions from equation [6.8], the rotational velocity vector of the tool holder in relation to the reference point gives us the following equation:

$${}^{1}\Omega_6 = \left[{}^{1}R_2(\phi_2)\dot{\theta}_2 + {}^{1}R_3(\phi_3)\dot{\theta}_3 + {}^{1}R_4(\phi_4)\dot{\theta}_4 + {}^{1}R_5(\phi_5)\dot{\theta}_5 + {}^{1}R_6(\phi_6)\dot{\theta}_6\right]\begin{bmatrix}0\\0\\1\end{bmatrix} \quad [6.10]$$

Thus, if we use simplified notations such as $C_{56} = \cos\left(\theta_5 + \theta_6\right)$, $s4 = \sin(\theta_4)$, we get:

$$ {}^{1}\Omega_{6} = \begin{bmatrix} -\left(\dot{\theta}_5 + \dot{\theta}_6\right) S_4 C_{56} - S_{56}\dot{\theta}_4 \\ \left(\dot{\theta}_5 + \dot{\theta}_6\right) S_4 S_{56} + C_{56}\dot{\theta}_4 \\ \dot{\theta}_2 + \dot{\theta}_3 + \left(\dot{\theta}_5 + \dot{\theta}_6\right) C_4 \end{bmatrix} \qquad [6.11] $$

Similarly, each translational velocity vector ${}^{1}\vec{V}_i\left(O_i\right)$ can be represented by the vector ${}^{i}v_i \in \mathbb{R}^{3\times 1}$, which expresses its components in the referential linked to the part (C_i) in question. Thus, we obtain:

$$ {}^{i}v_i = {}^{i}R_{i-1}\left(\theta_i\right)\left[{}^{i-1}v_{i-1} + {}^{i-1}\omega_{i-1} \times {}^{i-1}P_i \right] \qquad [6.12] $$

In these two expressions, ${}^{i-1}P_i$ denotes the vector of position of the origin O_i in relation to the origin O_{i-1}, expressed in the referential linked to the part (C_{i-1}).

Using the geometric properties of the arrangement of the parts of the arm and the expressions from equations [6.11] and [6.12], denoting the three unitary directions as $X, Y, Z \in \mathbb{R}^{3\times 1}$, we obtain:

$$ \begin{aligned} {}^{1}v_6 = {}^{1}R_6\, {}^{6}v_6 = {} & \dot{\theta}_2\left[EX + \left({}^{1}R_2\left(\phi_2\right) L_2 + A \right) Y + BZ \right] + \dot{\theta}_3 \left[EX + AY + BZ \right] \\ & + \dot{\theta}_4\left[{}^{1}R_4\left(\phi_4\right) L_{41} Y - {}^{1}R_5\left(\phi_5\right) L_5 C_5 Z \right] + \dot{\theta}_5 L_5\, {}^{1}R_5\left(\phi_5\right) Y \end{aligned} \qquad [6.13] $$

where:

$$ X = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T, \quad Y = \begin{bmatrix} 0 & 1 & 0 \end{bmatrix}^T, \quad Z = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T, $$

and

$$ \begin{aligned} A &= {}^{1}R_3\left(\phi_3\right) L_3 + L_{42}\, {}^{1}R_4\left(\phi_4\right) S_4 + {}^{1}R_5\left(\phi_5\right) L_5 \left(C_4 + 1\right) \\ B &= \left(L_{41} C_4 - {}^{1}R_5\left(\phi_5\right) L_5 S_4 S_5 \right) \\ E &= -\left(L_{42}\, {}^{1}R_4\left(\phi_4\right) C_4 \right) \end{aligned} $$

6.3.3.1.3. Specifications of the arms on MECALAC machines

In comparison to the standard arms with three coplanar joints, this design offers a greater dexterity of the tool holder in the execution of certain tasks such as that illustrated in Figure 6.7.

For example, the specifications of the kinematic design of the turret/arm mean the machines are able to dig trenches along a wall or a sidewalk without having to alter the orientation of the turret: see Figure 6.7 (b, c, d). The swing enables the stick and the tool holder to be moved laterally without the necessity of rotating the turret. However, the piston used to alter the position of the swing is not a working piston – i.e. normally, this joint is positioned when all the other hydraulic actuators are at rest. Thus, the drivers assign a constant value to the swing depending on the phase of the task needing to be done. This depends on the dexterity and experience of the drivers.

Depending on whether or not the positioning of the swing in relation to the after-boom is null (see Figure 6.8), we define two categories of configurations of the tool holder:

$-\theta_4 = 0\ rad$, the five parts are all aligned in the same plane (P1);

$-\theta_4 \neq 0\ rad$, the boom and the after-boom are aligned in the plane P1, the swing, stick and tool holder are aligned in a plane P2 such that the line intersecting with P1 is defined by the direction $\vec{Z}_4$.

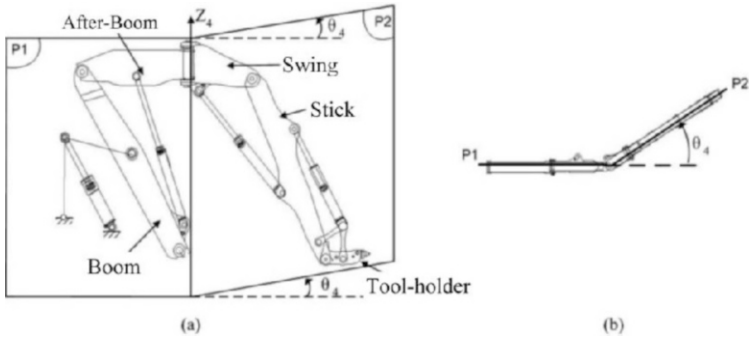


Figure 6.8. *Kinematics of the arm of the MECALAC 12MXT machine. (a) Head-on view; (b) Bird's-eye view*

Thus, depending on the articular position of the swing, two modes of piloting of the arm can be considered in the new interface: null swing and non-null swing.

6.3.3.2. *Current cockpit*

The current cockpit for 12MXT machines (see Figure 6.9) comprises two joysticks, a steering wheel and three pedals.

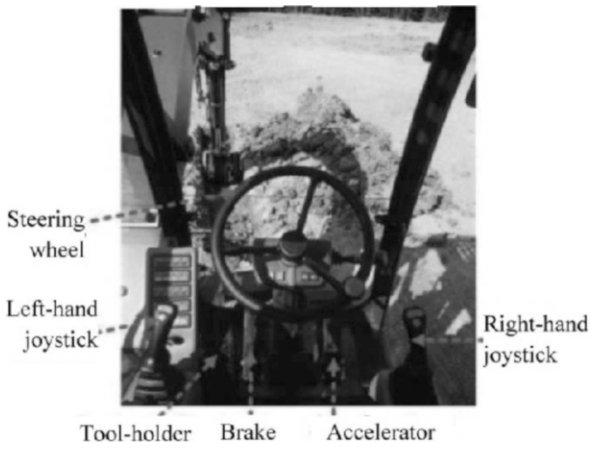


Figure 6.9. *Interior of the driver's cabin in the MECALAC 12MXT excavator-loader*

These piloting devices have very particular functions. The steering wheel controls the direction of movement of the platform. The pedal on the left controls the opening and closing of the device to attach the tools to the tool holder. The other two pedals are the accelerator and brake for the platform.

6.3.3.2.1. Roles of the joysticks

The joystick on the driver's left has two rotational degrees of freedom. It also has two buttons. It serves to control the position of the swing, and the movements of the turret or stick (see Figure 6.9):

– any deflection from the neutral position, to the right or to the left, will respectively cause a clockwise or anticlockwise (right or left) rotation of the turret by way of a hydraulic system;

– forward or backward deflections of the joystick respectively control the extension or retraction of the piston of the stick;

– two buttons are used to control, by incrementation, the extension or retraction of one of the two pistons of the swing: when one of the pistons is extended, the other is retracted.

The joystick on the driver's right has a button which, depending on whether it is depressed or not, enables him to control either the movements of the boom or the tool holder, i.e. the movements of the after-boom or the tool holder (see Figure 6.10):

– in both cases, a deflection to the right extends the tool holder piston, while a deflection to the left retracts it;

– if the button is not depressed, a forward deflection retracts the boom piston, whereas a backward extends it;

– depression of this button combined with a forward deflection causes the after-boom piston to extend, and conversely, a backward deflection with the button depressed causes it to retract.

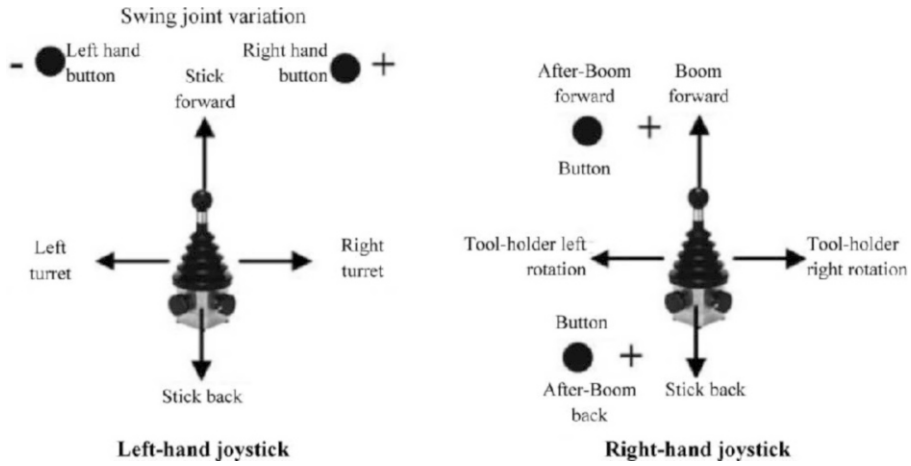


Figure 6.10. *Role of the joysticks in the current cockpit*

6.3.3.2.2. Piloting difficulties

The two joysticks are type "*X*" (see Figure 6.11): it is impossible to maneuver them simultaneously in both directions. Hence, at any one time the driver can only simultaneously control:

– the turret with the left joystick and one – and only one – of the following pistons with the right joystick: tool holder, after-boom or boom;

– the stick with the left joystick and one – and only one – of the following pistons with the right joystick: boom, after-boom or tool holder.

These basic functionalities mean the drivers are not able to control the boom, after-boom and tool holder pistons simultaneously. Furthermore, the orientation of the turret cannot be altered at the same time as the position of the stick. The directions of deflection of the degrees of freedom of these joysticks do not correspond to the directions of movement of the tool holder in relation to the driver.

For these reasons, the cockpit also has a "button box" which enables the driver to select a particular mode of operation amongst a number of others, with all of them predefined at the design stage. For instance, in "excavator-loader" mode, only the boom, stick and tool holder pistons are controlled, and therefore the machine is driven like a standard excavator-loader; in "MECALAC" mode, the driver individually controls each of the pistons on the machine.

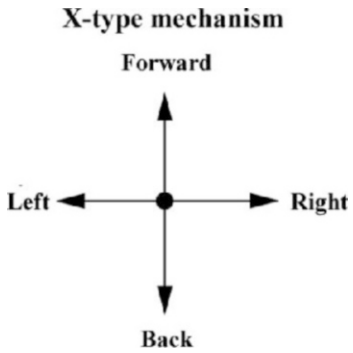


Figure 6.11. *Type of joysticks in the current cockpit*

Sometimes, when the drivers position the swing ($\theta_4 \neq 0$) to carry out the task of leveling, they move the chassis whilst keeping the different parts of the arm in a fixed position. The difficulty of carrying out this task lies in the fact that the drivers need to correctly position the chassis in relation to the task.

With regard to such machines, inexperienced drivers need several months' training before acquiring the experience necessary to control the machinery. According to Pépin [PEP 05], the possibilities for guiding the movements of the mechanical parts pose a problem of mental representation:

– often, beginner drivers make mistakes regarding the position of the controls in the cabin;

– during the execution of a task, an experienced driver does not wonder about the position of the controls – he knows where they are. However, he will find it difficult to orally report the actions he has undertaken once the task has been completed.

The turret, which is connected to the chassis, is – from a hydraulic point of view – controlled independently of the pistons, by way of a hydraulic system. The boom piston, mounted on a parallelogram, is located on the left of the turret. This arrangement may cause a problem for the drivers. Indeed, when the cabin swivels quickly and greatly to the left, their perception of the location of the arm in relation to the engine casing may be blocked by the presence of the boom piston. It is

therefore possible that part of the arm will collide with this casing. The same is true in terms of collisions between the arm and the chassis or the wheels.

6.3.3.3. *Functional specifications of a new cockpit for the movements of the tool holder*

The main aim of the new cockpit is to give drivers a way of directly controlling the movements of the tool holder without having to coordinate the velocities of the piston stems. In order to realize this objective, we need to define – depending on the configuration in which a driver sets the swing – the degrees of freedom of the tool holder and the directions of maneuvering of the new interfacing device which will be able to control these degrees of freedom. In addition, we need to consider that with the new cockpit, the drivers can, at all times, control the movements of the turret and the arm with the current method of control – i.e. individual control of each hydraulic actuator.

In the new cockpit, it is possible to uncouple the control of the movements of the turret from that of the movements of the tool holder. When presenting the structure of MECALAC arms, we cited two possible modes of control depending on the position of the swing in relation to the after-boom, i.e. depending on whether or not the angle θ_4 is null.

First of all, we shall define the degrees of freedom of the arm for each mode, specifying whether it is possible that these modes will be accepted by the drivers. Later on, an example of the device is put forward for the new interface for controlling the turret and tool holder.

6.3.3.3.1. Degrees of freedom of the tool holder when the swing is null

When the swing is null ($\theta_4 = 0$), all the parts of the arm are aligned in the same plane $(P1 \equiv P2)$. In these configurations, the arm can be used in one of two modes: "standard" and "MECALAC":

– standard mode: the articular position of the after-boom in relation to the boom is constant. The three parts involved in the operation of piloting are: the boom, the stick and the tool holder. In this mode, the arm has a configuration which is similar to that of a standard excavator-loader;

– MECALAC mode: in this case, four parts are involved: the boom, the after-boom, the stick and the tool holder.

In both these modes, the tool holder has three degrees of freedom in relation to any fixed point in the plane P1: two translational degrees of freedom (horizontal and vertical motion) and one degree of freedom of orientation in relation to any axis

perpendicular to P1. The model of geometric description of the arrangement of the parts, combined with the direct kinematic model put forward above, enables us to express these degrees of freedom as a function of the articular variables defined therein. The reference framework chosen is $\{\mathbb{R}_1\}$ (see Figure 6.12).

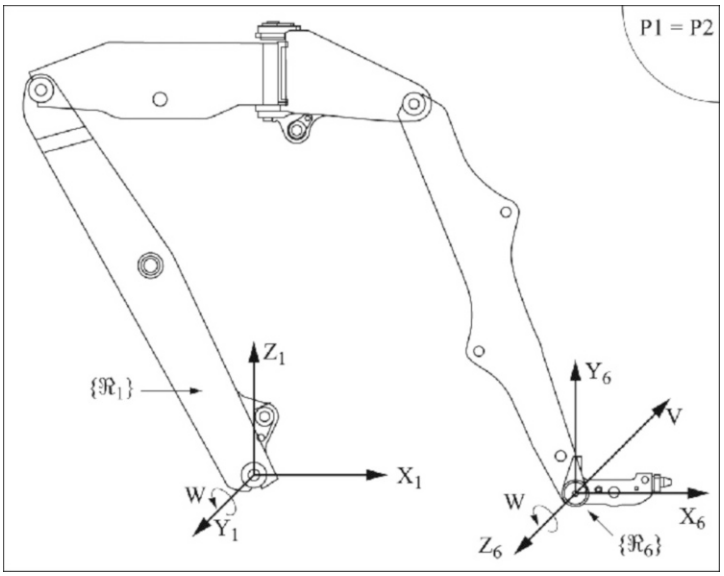


Figure 6.12. *Choice of reference framework when the swing is null*

In "MECALAC1" mode and in relation to this framework, the orientation of the framework $\{\mathbb{R}_6\}$, connected to the tool holder, results from a rotation of axis Y_1 parallel to Z_6, such that: ${}^1R_6(\theta) = R\left(Y, \theta_2 + \theta_3 + \theta_5 + \theta_6\right)$. Thus we define the degree of freedom of rotational velocity of the tool holder in relation to the framework $\{\mathbb{R}_1\}$. The expression given by equation [6.11] becomes:

$$ {}^1\Omega_6 = \omega Z \qquad [6.14] $$

where $\omega = \dot{\theta}_2 + \dot{\theta}_3 + \dot{\theta}_5 + \dot{\theta}_6$ and $Z = [0 \quad 0 \quad 1]^T$.

Similarly, if we set:

$$ \varphi_2 = \theta_2 \,,\ \varphi_3 = \varphi_2 + \theta_3 \,,\ \varphi_4 = \varphi_3 - \beta \,,\ \varphi_5 = \varphi_4 + \theta_5 \,,\ \varphi_6 = \varphi_5 + \theta_6 \qquad [6.15] $$

the expression given by [6.13], in "MECALAC1" mode, gives us the vector of velocity of the translational movement of the origin of the framework linked to the tool holder:

$$\begin{aligned} {}^1v_6 = {} & \dot{\theta}_2\left[L_{42}\,{}^1R_4(\varphi_4)X + \left({}^1R_2(\varphi_2)L_2 + {}^1R_3L_3 + 2{}^1R_5(\varphi_5)L_5\right)Y + L_{41}Z\right] \\ & + \dot{\theta}_3\left[-L_{42}\,{}^1R_4(\varphi_4)X + \left({}^1R_3(\varphi_3)L_3 + 2{}^1R_5(\varphi_5)L_5\right)Y + L_{41}Z\right] + \dot{\theta}_5L_5\,{}^1R_5(\varphi_5)Y \end{aligned} \quad [6.16]$$

where $X = [1 \quad 0 \quad 0]^T$, $X = [0 \quad 1 \quad 0]^T$ and $Z = [0 \quad 0 \quad 1]^T$.

From this latter expression [6.16], we get:

$$\begin{bmatrix} V_X \\ V_Y \end{bmatrix} = J(\theta_2, \theta_3, \theta_5)\begin{bmatrix} \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_5 \end{bmatrix} \quad [6.17]$$

$$J(\theta_2, \theta_3, \theta_5) = \left[J_1 \;\vdots\; J_2 \;\vdots\; J_3\right] : J_k \in R^{2\times 1}$$

Also, we have:

$$\begin{aligned} & J_3 = L_5\begin{bmatrix} -\sin\varphi_5 \\ \cos\varphi_5 \end{bmatrix}; J_2 = J_3 + p_{5x}\begin{bmatrix} -\sin\varphi_3 \\ \cos\varphi_3 \end{bmatrix} - p_{5y}\begin{bmatrix} \cos\varphi_3 \\ \sin\varphi_3 \end{bmatrix}; \\ & J_1 = J_2 + L_2\begin{bmatrix} -\sin\varphi_2 \\ \cos\varphi_2 \end{bmatrix} \end{aligned} \quad [6.18]$$

where $p_{5x} = L_3 + L_{41}\cos(\beta) + L_{42}\sin(\beta)$ and $p_{5y} = L_{42}\cos(\beta) - L_{41}\sin(\beta)$.

PROPERTY OF THE TRANSFORMATION MATRIX.– In "MECALAC" mode, the transformation $J(\theta_2, \theta_3, \theta_5)$, commonly referred to as the "Jacobian" of the arm, is – in the case of MECALAC arms – characterized by a 2×3 rectangular matrix: $J(\underline{\theta}) \in R^{2\times 3}$. It is this matrix which needs to be established in order to obtain the direct kinematic model. Because the translational velocity vector has two degrees of freedom which depend on three articular variables, the arm is redundant in this mode.

From the expressions in equation [6.18], using the matrix rank theory, we deduce that:

$$rank\left[J_1 \mid J_2 \mid J_3\right] = rank\left\{L_5\begin{bmatrix}-\sin\varphi_5\\ \cos\varphi_5\end{bmatrix}, L_2\begin{bmatrix}-\sin\varphi_2\\ \cos\varphi_2\end{bmatrix}, p_{5x}\begin{bmatrix}-\sin\varphi_3\\ \cos\varphi_3\end{bmatrix} - p_{5y}\begin{bmatrix}\cos\varphi_3\\ \sin\varphi_3\end{bmatrix}\right\} \quad [6.19]$$

From the right-hand side of this equality, we can extract three 2nd-order determinants and we obtain:

$$\begin{aligned}\Delta_1 &= -L_2L_5\sin(\varphi_5-\varphi_2) = -L_2L_5\sin(\theta_3+\theta_5-\beta)\\ \Delta_2 &= L_2\left[p_{5x}\sin(\theta_3)+p_{5y}\cos(\theta_3)\right]\\ \Delta_3 &= L_5\left[p_{5x}\sin(\theta_5-\beta)-p_{5y}\cos(\theta_5-\beta)\right]\end{aligned} \quad [6.20]$$

In view of the domain of definition of the variables θ_3 and θ_5, and of the value of β = 0.06rad, it is easy to show that the determinants Δ_2 and Δ_3 never take a value of 0. Indeed:

$$\Delta_2 = 0 \Rightarrow \theta_3 = \text{atan}\,2(-p_{5y}, p_{5x}) = 0.03 \text{ rad} \notin [-2,2\,;-0,5]\,.$$

Similarly:

$$\Delta_3 = 0 \Rightarrow \theta_5 = \beta + \text{atan}\,2(p_{5y}, p_{5x}) = 0.03 \text{ rad} \notin [-1,9\,;-0,6]\,.$$

In standard mode, the transformation matrix $J(\theta_2,\theta_5) = \left[J_1 \quad J_2\right]$ is a square 2×2 matrix: $J(\underline{\theta}) \in R^{2\times 2}$.

CONCLUSION.– In summary, for all the configurations in the workload, the tool holder has two translational degrees of freedom in relation to the turret. The direct kinematic model gives us:

– the two components of the velocity of translational movement of the tool holder in relation to the turret, which are functions of the first three articular velocities;

– the rotational velocity of the tool holder ω rad/s, which is equal to the sum of the four articular velocities.

6.3.3.3.2. Degrees of freedom of the tool holder when the swing is non-null

When the articular variable θ_4 is a non-null constant, the swing, stick and tool holder are in a plane P2 which forms a constant angle with the plane P1, which contains the boom and the after-boom (see Figure 6.13). Remember that this mode of use is only used by drivers for specific tasks – e.g. for performing leveling near to a wall.

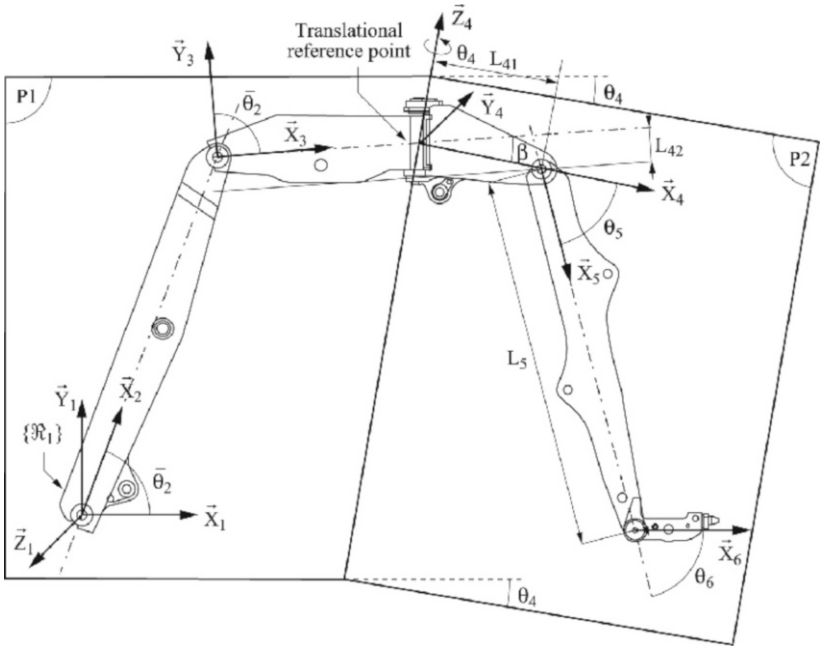


Figure 6.13. *Configuration of non-null swing*

Thus, at the moment when the driver manually sets the orientation of the swing in relation to the after-boom, the task and the driver are in two different planes, which makes it difficult to carry out the desired task by using resolved motion-rate control.

Thus, when the drivers position the swing, i.e. $\left(\theta_4 = \text{constant}, \dot{\theta}_4 = 0\right)$, the rotational velocity vector given by equation [6.11] becomes:

$$ {}^{1}\Omega_6 = \begin{bmatrix} -(\dot{\theta}_5 + \dot{\theta}_6) S_4 C_{56} \\ (\dot{\theta}_5 + \dot{\theta}_6) S_4 S_{56} \\ \dot{\theta}_2 + \dot{\theta}_3 + (\dot{\theta}_5 + \dot{\theta}_6) C_4 \end{bmatrix} \qquad [6.21] $$

Similarly, the translational velocity vector ${}^{1}\vec{V}_6$ from equation [6.13] becomes:

$$ {}^{1}v_6 = \begin{bmatrix} {}^{1}v_{6x} \\ {}^{1}v_{6y} \\ {}^{1}v_{6z} \end{bmatrix} == \begin{bmatrix} -(L_2 S_2)\dot{\theta}_2 - A_1(\dot{\theta}_2 + \dot{\theta}_3) - B_1\dot{\theta}_5 \\ (L_2 C_2)\dot{\theta}_2 + A_2(\dot{\theta}_2 + \dot{\theta}_3) + B_2\dot{\theta}_5 \\ (L_5 C_4 S_5)\dot{\theta}_5 \end{bmatrix} \qquad [6.22] $$

where:

$$ A_1 = \left(L_3 S_{23} + L_{42} C_{23\beta} + L_{41} C_4 S_{23\beta} + L_5 C_{23\beta} S_5 + L_5 S_{23\beta} C_4 C_5\right) $$
$$ A_2 = \left(L_3 C_{23} - L_{42} S_{23\beta} - L_{41} C_4 C_{23\beta} - L_5 S_{23\beta} S_5 + L_5 C_{23\beta} C_4 C_5\right) $$
$$ B_1 = L_5\left(C_{23\beta} C_4 S_5 + S_{23\beta} C_5\right) $$
$$ B_2 = L_5\left(C_{23\beta} C_5 - S_{23\beta} C_4 S_5\right) $$

From this expression, we obtain:

$$ \begin{bmatrix} {}^{1}V_{6x} \\ {}^{1}V_{6y} \\ {}^{1}V_{6z} \end{bmatrix} = J(\theta_2, \theta_3, \theta_5) \begin{bmatrix} \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_5 \end{bmatrix} \qquad [6.23] $$
$$ J(\theta_2, \theta_3, \theta_5) = [J_1 \mid J_2 \mid J_3] : J_k \in \mathbb{R}^{3\times 1} $$

where we have:

$$ J_3 = \begin{bmatrix} -B_1 \\ B_2 \\ L_5 C_4 S_5 \end{bmatrix}; \quad J_2 = \begin{bmatrix} -A_1 \\ A_2 \\ 0 \end{bmatrix}; \quad J_1 = J_2 + \begin{bmatrix} -L_2 S_2 \\ L_2 C_2 \\ 0 \end{bmatrix} \qquad [6.24] $$

From the expressions in equation [6.24], we can calculate the determinant:

$$ \Delta = L_2 L_5 C_4 S_5 \left(A_1 C_2 - A_2 S_2\right) \qquad [6.25] $$

In view of the domain of definition of the articular variable θ_5, this determinant is null if:

$$(A_1C_2 - A_2S_2) = 0 \Rightarrow L_3S_3 + (L_{42} + L_{41}C_4 + L_5S_5)C_{3\beta} + L_5C_4C_5S_{3\beta} = 0$$

In addition, when $\theta_4 \neq 0$, we can see from equations [6.21] and [6.22] that the six components of the translational velocity vector and rotational velocity vector are given by a system of six equations with four unknowns. Hence, these six equations are dependent. Thus, on the basis of the velocity vector of the tool holder, it is near impossible to obtain the desired articular velocities of the parts of the arm without imposing constraints on its movements.

For instance, the "Easy Drive" option used by drivers to keep the orientation of the tool holder constant in relation to the reference framework (preserving the angle of incidence of the tools) enables us to write:

$$^1\Omega_6 = 0 \Rightarrow \begin{cases} (i) & \dot{\theta}_5 + \dot{\theta}_6 = 0 \\ (ii) & \dot{\theta}_2 + \dot{\theta}_3 = 0 \end{cases} \qquad [6.26]$$

With these two constraints, the translational velocity vector from equation [6.22] becomes:

$$^1V_6 = \begin{bmatrix} ^1V_{6x} \\ ^1V_{6y} \\ ^1V_{6z} \end{bmatrix} = \underbrace{\begin{bmatrix} -L_2S_2 & -B_1 \\ L_2C_2 & B_2 \\ 0 & L_5C_4S_5 \end{bmatrix}}_{J} \begin{bmatrix} \dot{\theta}_2 \\ \dot{\theta}_5 \end{bmatrix} \qquad [6.27]$$

From the 3×3 matrix J, we can extract three 2^{nd}-order determinants:

$$\begin{bmatrix} ^1V_{6y} \\ ^1V_{6Z} \end{bmatrix} \Rightarrow |\Delta_1| = L_2L_5C_2C_4S_5 \neq 0 \qquad \forall\ \theta_2, \theta_5;$$

$$\begin{bmatrix} ^1V_{6x} \\ ^1V_{6z} \end{bmatrix} \Rightarrow |\Delta_2| = -L_2L_5S_2C_4S_5.$$

There is a configuration ($\theta_2 = \frac{\Pi}{2}$) where $|\Delta_2| = 0$;

$$\begin{bmatrix} {}^1V_{6x} \\ {}^1V_{6y} \end{bmatrix} \Rightarrow |\Delta_3| = L_2 L_5 \left(C_4 C_{3\beta} S_5 + S_{3\beta} C_5 \right) \neq 0 \qquad \forall\ \theta_3, \theta_3, \theta_5, \beta.$$

On the basis of the above, the drivers cannot impose the velocity vector of the tool holder. Thus, in using the "Easy Drive" option, the drivers can:

– simultaneously control the boom and the stick;

– have the boom and stick immobile, but control the translational motion of the platform.

In conclusion, when the configuration of the swing is non-null, it is preferable for the control of the tool holder to be done by individual control of the different parts of the arm, the chassis and the turret – i.e. the current method of control.

6.3.3.3.3. Conclusion

On the basis of the above, we propose two new modes of control: "standard" mode and "MECALAC" mode when the value given to θ_4 is null. With these two modes, the drivers impose set points for the velocity of the tool holder (both translational and rotational motion) in relation to a reference framework. Hence, an articulated mechanical device with three degrees of freedom is sufficient. Each of these degrees of freedom corresponds to a component of velocity. It should be noted that each deflection of one degree of freedom corresponds to a component of the velocity vector of the tool holder. Furthermore, we have to add to the new cockpit: one degree of freedom for the movements of the turret and one degree of freedom for the movements of the swing. When the position of the swing is non-null, the tool holder will be controlled by individual commands to the hydraulic actuators of the different parts of the arm.

6.3.3.4. *Possible realizations*

A number of mechanical "devices" can be put forward to control the movements of the turret, the swing and the tool holder in the new cockpit. All these devices need to take account of the fact that the drivers can, at any time, switch to standard guidance of the tool holder. Hereafter in this section, we describe two possible choices: joysticks and an articulated mechanism.

6.3.3.4.1. Joysticks

The first possible choice is to use two joysticks (see Figure 6.14): one with two degrees of freedom (right-hand joystick) and the other with a single degree of freedom and an axis of rotation of the handle (left-hand joystick). Also, when the drivers choose standard mode (fixed position of the after-boom), we advocate adding a degree of freedom to the left-hand joystick to directly control the after-boom piston. Thus, the drivers can position this part before choosing standard mode to execute the task. Both joysticks have buttons for added functionalities in the new interface.

The right-hand joystick (i.e. on the driver's right hand), can be used to control the two components of the translational velocity vector of the tool holder, independently or in combination ("O"-type mechanism):

– deflection forward or backward from the neutral position respectively controls the "horizontal" translational motion forward or backward of the tool holder;

– deflection to the right or left of the neutral position respectively controls the "vertical" translational motion upward or downward of the tool holder.

In addition, this joystick has two buttons enabling drivers to choose one of the three modes of operation offered.

The second joystick (on the driver's left hand) controls the rotation of the turret, the movements of the after-boom piston and the orientation of the tool holder ("T"-type mechanism):

– any deflection to the right or left of the neutral position respectively causes the turret to rotate to the right or left, by way of a hydraulic mechanism;

– deflection to the fore or rear of the neutral position directly controls the extension or retraction of the after-boom piston, by way of a hydraulic mechanism;

– rotation of the handle to the right or the left controls the orientation of the tool holder in both directions.

In addition, this joystick has three buttons. The first two enable the drivers to directly control the swing piston, while the third serves to select the mode of control.

Although the solution put forward here enables the driver to impose the desired direction and modulus of the velocity vector of the tool holder and its orientation, this possible solution has a drawback: there is no correspondence between the vertical movement (up and down) of the tool holder and the manipulation of the joystick by the drivers (right and left).

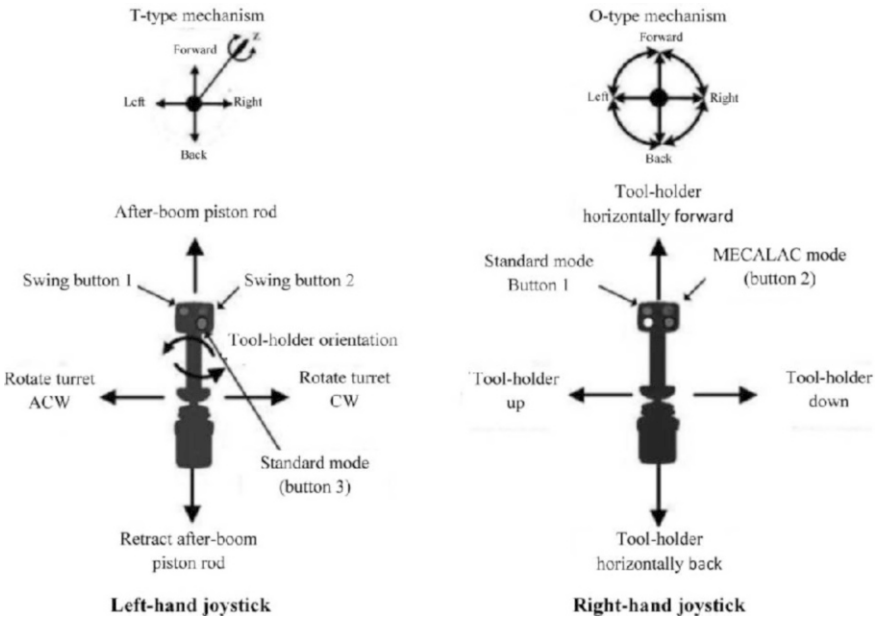


Figure 6.14. *Functions of the new joysticks*

The advantage to using joysticks is that the driver has no difficulty in switching from standard mode to the new mode and vice versa.

In order to offer drivers the possibility of changing, at any time, to standard mode, a PLC has to be added to the outputs from the joysticks. Thus, depending on the control mode selected by the drivers, this PLC sends the commands from the joysticks to two different places:

– current control mode: the commands control the movements of each hydraulic actuator. The deflection of each degree of freedom of the joysticks controls the aperture of the pressurized hydraulic fluid orifice in the chambers of these actuators;

– new control modes: the commands are sent directly to an onboard computer (or calculator) where they are used by the software to determine the articular velocities of the actuators.

6.3.3.4.2. Articulated mechanism

The second possible solution that we propose is to use an articulated mechanism which, by its arrangement, defines the three components of the velocity vector of the tool holder (two translational and one rotational) to ensure correspondence between

the movement of the tool holder and the manipulation of that mechanism by the right hand of the driver for the new control modes, “standard” and “MECALAC1” when the articular variable of the swing is null.

In these two modes, the movements of this mechanism by the driver’s hand take place in a plane parallel to the plane containing the arm. The proposed mechanism has the same structure as the joysticks. It has memory springs on the joints which return the joystick to neutral configuration ($\vec{V} = 0, w = 0$), when the driver releases his hand. The articulated mechanism comprises a base, two parts of equal length and a tool attached to the end of the second part (see Figure 6.15). The axes of rotation are parallel ($\vec{Z}_2 \parallel \vec{Z}_3$).

The two parts are equipped with potentiometers mounted on the joints. Thus, when the driver moves the mechanism with his hand, the deflection of each part from its neutral position enables us to predict the two components of the translational velocity of the tool holder ($\vec{V}$). Similarly, the orientation of the tool determines the rotation of the tool holder in both directions (w).

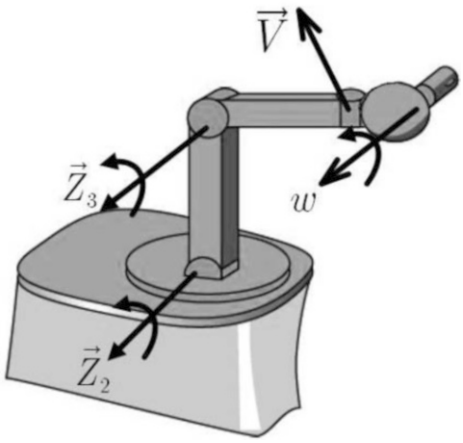


Figure 6.15. *Articulated mechanism*

In conclusion, this articulated structure offers drivers a correspondence between the movements of the tool holder and the manipulation of this structure in all possible configurations of the arm. However, it exhibits the following drawbacks:

– during the course of repetitive manipulation of this mechanism, it is obvious that the concept of "fatigue" comes into play, which will influence the drivers' actions;

– consequently, the drivers find it difficult to impose the "horizontal" or "vertical" direction of the tool holder and maintain it for a certain length of time;

– this mechanism cannot be used to control the arm with the current method if the drivers wish to switch back. Hence, they find themselves with two interfaces in the cabin.

6.3.3.4.3. Conclusion

Above, we have put forward two possible solutions for the interface. Each of them has as many advantages as disadvantages. It is difficult to find a solution which is simple to implement on a machine and which at the same time exhibits correspondence between the movements of the tool holder and the driver's hand. Hence, the realization of an interface requires a complete and in-depth study – both mechanical and ergonomic – which will not be performed within the limitations of the current work.

6.3.3.5. *Calibration of the joysticks and constraints*

In order to implement this strategy on the machine, the joysticks chosen must be robust and must satisfy the design requirements. All industrial joysticks, of course, have the capability to perform potentiometric measurements. Each axis of freedom of the handle corresponds directly to a fine-tuned potentiometer, which is then used in a voltage divider where its value conveys information about the inclination of the handle of the joystick. Thus, the output from the joystick is proportional to the deflection of the control unit from its central position. Then, depending on the control mode chosen, the data gleaned from the joysticks are used to control the orifices of the chambers of the pistons or are fed into the onboard computer after passing through an analog-digital converter.

In the latter case, the issue arises of calibration between the positions of the potentiometers installed in the joysticks and the components of the velocity vector of the tool holder. We need to know the maximum value given by the maximum deflection of the handle of the joystick (in radian) and the maximum possible value of one of the components of the velocity vector of the tool holder (in m/s). Thus, we have to choose a coefficient to determine the command imposed in the tool holder – a coefficient which is by no means easy to find. Even if we do manage to find the ideal coefficient, is the velocity imposed on the tool holder feasible (in view of the maximum power of the hydraulic pump)? What happens if the pistons are near to their uppermost or lowermost limits?

The issue of calibration can be simplified if, at the level of the functionalities, it is accepted that the direction of the translational velocity vector is more important than its modulus. In other words, at all times, the configuration given to the joysticks sets the desired direction of the translational velocity vector, while the modulus initially deduced from the configuration of the potentiometers may be diminished if:

– the courses of the hydraulic pistons involved in the movements of the tool holder are too close to their uppermost or lowermost limits. Also, we need to ensure that when any of the piston stems reaches its limit, it will not do so with any velocity, regardless of the actions being performed;

– the total flowrate required of the pump by all the hydraulic actuators is greater than the pump's maximum capacity.

Thus, the directions of the velocity of translational and rotational motion are guaranteed, but the modulus of these velocities will be adapted in order to deal with numerous constraints.

6.3.3.6. *Other functions*

The "resolved motion-rate control" technique requires the use of articular position sensors mounted on each part of the arm. As these sensors provide information in real time, we can use the direct geometric model to work out the position of every point of the moving arm in relation to a fixed point on the chassis. Based on these data, it is possible to add new functionalities to support the drivers.

In addition, in order for these functionalities to be effective and more operational, it is necessary to add a graphic and tactile interface which can be placed in the driver's cabin on the machine. This display system (touchscreen) enables drivers to monitor in detail and in real time all the operations that they are carrying out, and offers them an overall view of the position of the machine in relation to the task – as well as, of course, any information that is useful for the smooth progression of that task. Below, we describe three functionalities: limitation of the working area, height indicator and the display of how far from the machine the workspace for the task is. Furthermore, this display monitor uses touchscreen technology, so the drivers can select the necessary parameters.

6.3.3.6.1. Limitation of the working area

A particular point about the MECALAC machine is that it is designed for civil engineering work in urban areas, which means that the drivers will not always have a clear view of the place where they are performing a task (e.g. digging). Thus, it is

necessary to develop "computerized defense mechanisms" to protect the arm and the machine from critical situations and limit the working area.

Thus, it is convenient to avoid all possible collisions between any part of the arm and the engine casing or any other part of the chassis. By integrating realistic representations of the volumes occupied by the parts in question, it is possible to predict potential collisions. We have to use the same approach as that used above to deal with the first two constraints: i.e. preservation of the directions and velocity vectors imposed by the drivers, but decrease of their modulus depending on the predicted dangers of collision.

In addition, we could suggest to the drivers, for instance, that they limit the working area with their tool at the start of a task, so that during the execution, the arm will not touch the obstacles already defined by the tool.

Finally, because MECALAC machines operate in urban areas, it is advantageous to limit the movements of the arm so as not to touch the electrical cables located 4.5 m underground. To do so, we can use the direct geometric model.

6.3.3.6.2. Digging indicator

Another idea to be taken into account in terms of the driver/machine interface is an assistance device to indicate the height during the realization of digging task. The advantage of this device is to help the drivers to dig a trench with the same depth during the movements of the machine.

To begin with, the driver attaches a tool to the tool holder. Then, he drives to the dig site (see Figure 6.16). At that time, the arm has an initial configuration $\theta_0 = \left[\theta_{02} \quad \theta_{03} \quad \theta_{05} \quad \theta_{06}\right]$, and the software determines the initial position of the tool holder, *Po*, in relation to the reference framework:

$$\begin{aligned} X_0 &= l_2C_2 + l_3C_{23} + l_{41}C_4C_{23\beta} - l_{42}S_{23\beta} + l_5C_{23\beta}C_4C_5 - l_5S_{23\beta}S_5 \\ Y_0 &= l_2S_2 + l_3S_{23} + l_{41}C_4S_{23\beta} + l_{42}C_{23\beta} + l_5S_{23\beta}C_4C_5 + l_5C_{23\beta}S_5 \end{aligned} \quad [6.28]$$

Next, the driver uses the touchscreen interface to enter the desired height (h_d) and begins to dig. As this task advances, the software calculates the Cartesian position (X , Y) where the height, *h*, is:

$$h = Y - Y_0 \quad [6.29]$$

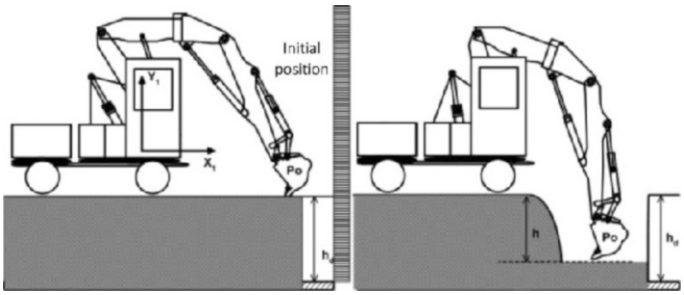


Figure 6.16. *Regulation of height for the realization of a task of digging on flat ground*

When the ground is not flat, an inclinometer on the platform can be used to feed back the angle of inclination α with the horizontal (see Figure 6.17). Thus, the computer determines the initial position of the tool holder, *Po*, in relation to the reference framework $\left\{\mathbb{R}_0 \equiv \left(O;\vec{X}_0,\vec{Y}_0\right)\right\}$:

$$X_0 = l_2 C_{\alpha 2} + l_3 C_{\alpha 23} + l_{41} C_4 C_{\alpha 23\beta} - l_{42} S_{\alpha 23\beta} + l_5 C_{\alpha 23\beta} C_4 C_5 - l_5 S_{\alpha 23\beta} S_5$$
$$Y_0 = l_2 S_{\alpha 2} + l_3 S_{\alpha 23} + l_{41} C_4 S_{\alpha 23\beta} + l_{42} C_{\alpha 23\beta} + l_5 S_{\alpha 23\beta} C_4 C_5 + l_5 C_{\alpha 23\beta} S_5$$

where:

$$C_{\alpha 2} = \cos\left(\alpha + \theta_2\right), C_{\alpha 23} = \cos\left(\alpha + \theta_2 + \theta_3\right), C_{\alpha 23\beta} = \cos\left(\alpha + \theta_2 + \theta_3 + \beta\right);$$

$$S_{\alpha 2} = \sin\left(\alpha + \theta_2\right), S_{\alpha 23} = \sin\left(\alpha + \theta_2 + \theta_3\right), S_{\alpha 23\beta} = \sin\left(\alpha + \theta_2 + \theta_3 + \beta\right).$$

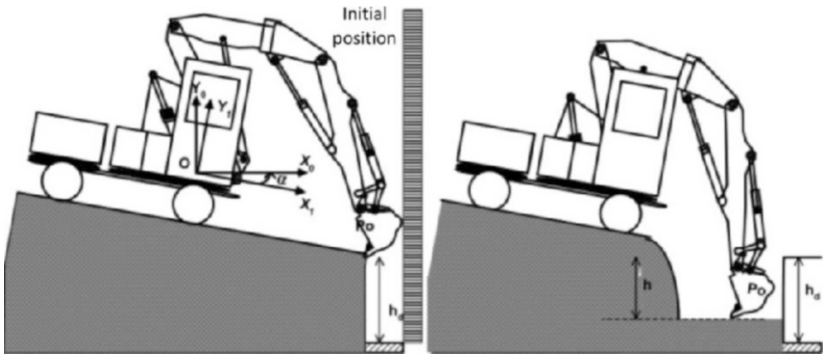


Figure 6.17. *Regulation of height for the realization of a task of digging on uneven ground*

The height thus determined is sent to the display unit to keep the driver informed about the advancement of the task. When the driver goes beyond the desired height, an alarm is sounded and the movement of the arm is locked, so long as the driver does not change the direction of movement of the scoop.

6.3.3.6.3. Display of the MECALAC machine and cycle of advancement of the task

The display in real time of a "miniature" representation of the machine and the advancement of the task, in relation to a reference point set by the tool at the beginning of a task, is necessary when part of the arm and the work area are not visible to the drivers. This animation offers a possibility for support which enables them to take the appropriate decision about the velocity vector of the tool holder. In addition, information about the limitations of the joints, the total flowrate required and the possibility of a collision will constantly be displayed to the drivers in order to alert them and help them to make decisions regarding the modification of the velocity of the tool holder.

6.3.3.7. *Conclusion*

The principle of the new system for controlling the movements of the tools, presented above, enables the driver to alter components of velocity of the tool holder in relation to the reference framework connected to the turret, depending on the chosen mode. Remember that in "standard" and "MECALAC1" modes, all the parts of the arm are aligned, the swing is immobile, and the tool holder has three velocity components.

In practice, the driver's actions with the control system, i.e. the joysticks, will be converted into electrical values which, sampled regularly, will in turn be converted into velocities for the variables of configuration of the arm. This section is given over to the determination of these conversions, with the following hypotheses:

– the data needing to be converted are the components of the kinematic torsor $(\vec{V}, \vec{\Omega})$ of the tool holder;

– the variables of configuration are the respective angular positions of the main parts in relation to one another – i.e. the articular variables previously defined: $\theta_2, \theta_3, \theta_5$ and θ_6.

When the swing is aligned with the after-boom, only the option "MECALAC", for which the arm exhibits redundancy (four articular velocities for three independent components of the kinematic torsor) will be considered. Indeed, the "standard" option, for which the arm has no redundancy, is merely a particular case of the former. Based on the direct transformation between the articular velocities and the three components of the torsor described in the previous section, the general

approach known as the "projected gradient" method will be used to express the four articular velocities on the basis of the three components of velocity of the tool holder.

We shall then go on to demonstrate, by way of simulations, that it is necessary to take account of the fact that the driver can give commands of velocity of the tool holder which cause the configuration variables to reach one or the other of their limits with non-null velocities.

In order to avoid these problems, we propose that when the values of the articular variables are approaching their limits, a procedure be added which decreases the modulus of the velocity vector of the tool holder controlled by the joysticks, whilst preserving its direction.

Also, because the power of the hydraulic pump on the MECALAC machine is limited, it is possible that the power required by the hydraulic actuators of the parts of the arm will surpass the maximum power. Thus, another procedure is added with the aim of avoiding this problem whilst conserving the direction of velocity of the tool holder.

6.3.4. *Limit of articular position and velocities*

Having defined the new functionalities, we now need to engineer and implement them. The following sections detail all the techniques used to calculate the velocities of the piston stems on the basis of the commands for the velocity vector of the tool holder given by the new interface and the different inverse kinematics. In addition, these velocities need to take account of all the constraints, such as the limits of the piston stems, the limits of their velocities and the risk of collision.

6.3.4.1. *Inverse kinematics in MECALAC mode*

Remember that in this mode, the tool holder has three degrees of freedom. The direct kinematic model gives us the three components of the velocity of the tool holder in relation to the turret: two for translational motion and one for rotation, such that:

$$\begin{bmatrix} V_X \\ V_Y \end{bmatrix} = J\left(\theta_2, \theta_3, \theta_5\right) \begin{bmatrix} \dot{\theta}_2 \\ \dot{\theta}_3 \\ \dot{\theta}_5 \end{bmatrix} \qquad [6.30]$$

$$\omega = \dot{\theta}_2 + \dot{\theta}_3 + \dot{\theta}_5 + \dot{\theta}_6$$

The "Jacobian" matrix, in this mode, is rectangular, measuring 3×4: $J(\underline{\theta}) \in R^{3\times4}$. As this Jacobian is not invertible, the construction of the inverse kinematic model involves choosing, out of all the possible vectors $\underline{\dot{\theta}} \in R^3$ which yield the same vector $\underline{V} \in R^3$, the one which has "the most interesting" properties.

In view of these hypotheses, at all times during sampling, the problem lies in calculating a solution to the following system of equations:

$$\text{(a)} \quad J\left(\theta_2(kT), \theta_3(kT), \theta_5(kT)\right) \begin{bmatrix} \dot{\theta}_2(kT) \\ \dot{\theta}_3(kT) \\ \dot{\theta}_5(kT) \end{bmatrix} = \begin{bmatrix} V_{Xd}(kT) \\ V_{Yd}(kT) \end{bmatrix}$$

$$\text{(b)} \quad \dot{\theta}_2(kT) + \dot{\theta}_3(kT) + \dot{\theta}_5(kT) + \dot{\theta}_6(kT) = \omega_d(kT)$$

The method adopted consists of solving part (a) first, followed by part (b). In the interests of ease of notation, and because the calculation has to be performed each time sampling takes place, we shall posit:

$$\mathrm{V}_d = \begin{bmatrix} V_{Xd}(kT) \\ V_{Yd}(kT) \end{bmatrix} \qquad [6.31]$$

$$\Psi = [\theta_3, \theta_5]^T \; ; \; \Phi = [\theta_2, \theta_3, \theta_5]^T$$

Solving part (a) involves determining a particular solution $\dot{\Phi}_d$ amongst all the solutions $\dot{\Phi}$ such that $J(\Phi)\dot{\Phi} = \mathrm{V}_d$. Indeed, as the rank of the matrix $J(\Phi)$ is equal to 2 in all configurations, the system $J(\Phi)\dot{\Phi} = \mathrm{V}_d$ has an infinite number of solutions. Once a solution has been determined, part (b) will give us: $\dot{\theta}_{6d} = \omega_d - \dot{\theta}_{2d} + \dot{\theta}_{3d} + \dot{\theta}_{5d}$.

6.3.4.1.1. The "projected gradient" method

For part (a), the resolution method known as the "projected gradient method", put forward by Liégeois [LIE 77], consists of taking the least-norm solution $\dot{\Phi}_{\min}$ and adding to it the projection of the gradient of a cost function $H(\Phi)$ on the kernel of the application represented by $J(\Phi)$. The least-norm solution can be expressed with the generalized right-hand inverse of the matrix $J(\Phi)$:

$$\begin{aligned}\dot{\Phi}_{\min} &= J^{+}(\Phi)\mathrm{V}_{d} : J^{+}(\Phi) = J^{T}(\Phi)\left[J(\Phi)J^{T}(\Phi)\right]^{-1} \\ \dot{\Phi}_{d} &= \dot{\Phi}_{\min} - \alpha\left[I_{3\times 3} - J^{+}(\Phi)J(\Phi)\right]\nabla_{\Phi}(H(\Phi))\end{aligned} \qquad [6.32]$$

Indeed, all the right-hand pseudo-inverse matrices are such that $JJ^{+} = I$. Hence, the second term in the solution is a particular solution to the homogeneous system $J(\Phi)\dot{\Phi} = 0 \in R^{2}$. In other words, the solution $\dot{\Phi}_{d}$ obtained consists of automatically seeking out the "best" configuration to follow the current configuration whilst minimizing the cost function. A number of cost functions have been put forward so that the solutions obtained for the articular velocities only yield acceptable configurations for the arm – i.e. configurations which are located within the working area of the tool holder.

Of these cost functions, that initially put forward by Liégeois [LIE 77] is, in our case, such that:

$$i = 2,3,5 : H\left(\Phi\right) = \frac{1}{3}\sum_{i}\left[\frac{\theta_{i} - a_{i}}{a_{i} - \theta_{i\,\max}}\right]^{2} ; a_{i} = \frac{\theta_{i\max} + \theta_{i\min}}{2} \qquad [6.33]$$

The weight associated with each of the quadratic terms is the same, regardless of the deviation $\theta_{i} - a_{i}$. The cost function proposed by [SIN 95] enables us, by choosing a positive factor ϕ, to attach a greater weight to those articular variables which are near to one of their limits:

$$i = 2,3,5 : H(\Phi) = \sum_{i}\cosh\left[\phi\frac{\theta_{i} - a_{i}}{\theta_{i\max} - \theta_{i\min}}\right] \qquad [6.34]$$

In all cases, it should be noted that the choice of the cost function and that of the scalar α are two factors which influence the values of the articular velocities.

In order to implement this method, each time a sample is taken, on the basis of values of the three components of the torsor of velocities $V_{Xd}, V_{Yd}, \omega_{d}$ and the values of the three articular variables $\theta_{2}, \theta_{3}, \theta_{5}$, we need to calculate:

– the values of the 6 terms in the pseudo-inverse matrix based on the values of the articular variables;

– the values of the 9 terms of the product of the pseudo-inverse matrix by the matrix $J\left(\Phi\right)$;

– the three terms of the gradient of the cost function.

These calculations have to be performed on the basis of the analytical expressions of the 6 + 9 terms in question. Hence, this method is unwieldy and difficult to implement. The main advantage to it lies in the fact that the movements which will be imposed on the four pistons in question (boom, after-boom, stick and tool holder) will be continuous movements.

6.3.4.1.2. Procedure of implementation

The procedure of implementation of the above method is directly inspired by [ZGH 90]. In accordance with the direct kinematic model, it is always possible to express part (a) of the system of equations in the following form:

$$\begin{aligned} &J(\boldsymbol{\Phi})\dot{\boldsymbol{\Phi}} = \mathrm{V}_d : J(\boldsymbol{\Phi}) = \left[J_1(\boldsymbol{\Phi}) \;\vdots\; J_0(\boldsymbol{\Phi}) \right] \\ &J_1(\boldsymbol{\Phi}) \in R^{2\times 1} \; ; J_0(\boldsymbol{\Phi}) \in R^{2\times 2} \\ &J_1(\boldsymbol{\Phi})\dot{\theta}_2 + J_0(\boldsymbol{\Phi})\dot{\Psi} = \mathrm{V}_d \end{aligned} \qquad [6.35]$$

The inverse of the square matrix $J_0(\boldsymbol{\Phi}) = [J_2(\boldsymbol{\Phi}), J_3(\boldsymbol{\Phi})]$ always exists.

Any and every solution $\dot{\boldsymbol{\Phi}}$ to part (a) can be expressed as the sum of a particular solution $\dot{\boldsymbol{\Phi}}_p$ and a solution $\dot{\boldsymbol{\Phi}}_h$ of the homogeneous system associated therewith. We can easily show that the general solution to the homogeneous system is such that:

$$\dot{\boldsymbol{\Phi}}_h = \begin{bmatrix} 1 \\ -J_0^{-1} J_1 \end{bmatrix} k = \dot{\varphi}_h k \quad \forall k \in R \qquad [6.36]$$

The kernel of $J(\boldsymbol{\Phi})$, of dimension 1, is engendered by the vector $\dot{\varphi}_h$. A particular solution is:

$$\dot{\boldsymbol{\Phi}}_p = \begin{bmatrix} 0 \\ J_0^{-1} V_d \end{bmatrix} \qquad [6.37]$$

The least-norm solution $\dot{\boldsymbol{\Phi}}_{\min}$ is obtained by looking for the value of the scalar *k* such that:

$$\frac{\partial}{\partial k}\left(\left\| \dot{\boldsymbol{\Phi}}_p + k\dot{\varphi}_h \right\|^2 \right) = 2\left(\dot{\boldsymbol{\Phi}}_p + k\dot{\varphi}_h \right)^T \dot{\varphi}_h = 0 \qquad [6.38]$$

Thus, using the unit vector $u_h = \dot{\varphi}_h / \|\dot{\varphi}_h\|$, we get:

$$\dot{\Phi}_{\min} = \dot{\Phi}_p - \frac{\dot{\Phi}_p^T \dot{\varphi}_h}{\|\dot{\varphi}_h\|^2} = \dot{\Phi}_p - \left(\dot{\Phi}_p^T u_h\right) u_h \qquad [6.39]$$

The term $\left(I - J^+ J\right)\nabla H(\Phi)$ is the projection of the gradient onto the kernel of $J(\Phi)$. Thus it is part of the kernel, which means we can write:

$$\left(I - J^+ J\right)\nabla H(\Phi) = \frac{\nabla H^T(\Phi)\dot{\varphi}_h}{\|\dot{\varphi}_h\|^2} = \left(\nabla H^T(\Phi) u_h\right) u_h$$

Finally, the solution $\dot{\Phi}_d$ is obtained by calculating the value of the following expression:

$$\dot{\Phi}_d = \dot{\Phi}_p - \left(\left(\dot{\Phi}_p + \alpha \nabla H\right)^T u_h\right) u_h \qquad [6.40]$$

At each moment of sampling, the desired articular velocities, calculated using equation [6.40], require calculations to be done: offline calculations, such as the number of constants in the analytical expressions, and then online calculations, such as the number of calls to trigonometric functions, additions, multiplications and other types of operations which are summarized in Table 6.1. With this method, the calculations are simple to implement with a real-time calculator.

Standard mode $(\theta_3 = \text{constant}, \dot{\theta}_3 = 0)$ is a particular solution equal to the general solution of the system [6.40], with $u_h = 0$, $\alpha = 0$ and $J_0(\Phi) = [J_1(\theta_2, \theta_5), J_3(\theta_5)]$. Thus, the articular velocity vector is obtained by calculating the value of the following expression:

$$\dot{\Phi}_d = \begin{bmatrix} \dot{\theta}_2 \\ \dot{\theta}_5 \end{bmatrix} = \dot{\Phi}_p = J_0^{-1} V_d \qquad [6.41]$$

6.3.4.2. *Simulations*

In this section, the inverse kinematic model of the MECALAC 12MXT excavator-loader is tested by way of digital simulations, using a simulation software package (Matlab®, Simulink) with the aim of testing the proposed inverse kinematic model. Figure 6.18 illustrates the different stages necessary to obtain the desired

articular velocities for all the parts of the arm on the basis of the velocity vector of the tool holder set by the joysticks. The desired articular velocities determined are sent to an axle control system, causing movement in all the parts of the arm. Yet, as we are not using a dynamic model of the arm, the articular positions of all the parts are obtained by integration of the desired articular velocities, taking account of the articular limitations ($\theta_{i\max}, \theta_{i\min}$) and the initial configuration, θ_{i0}, of each part of the arm. All the calculations are carried out with every period of sampling T_e.

Elements	Online				Offline	
	Addition	Product	Trigon.	Root	Addition	Product
$J_o^{-1}(\Psi)$	2	11	4	0	0	0
$\dot{\Phi}_p$	2	4	0	0	0	0
u_h	5	8	0	2	0	0
∇H	3	6	3	0	9	9
$\dot{\Phi}_d$	11	9	0	0	0	0
$\dot{\theta}_{6d}$	3	0	0	0	0	0
Total	26	38	7	2	9	9

Table 6.1. *Numbers and types of operation to calculate the desired articular velocities*

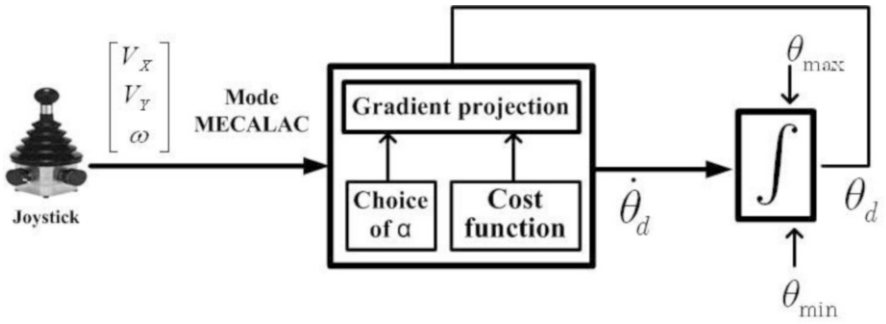


Figure 6.18. *Functional diagram of the simulation of the new system by Simulink*

Simulation of the inverse kinematic model in MECALAC mode involves, firstly, demonstrating the advantage to using the projected gradient method for the realization of a particular task when one or more articular variables are close to their limits. Then, a number of simulations are carried out in order to choose one of the two cost functions and the value of α. For the duration of these simulations: the

turret is immobile and the initial orientation of the tool holder is conserved ($w = 0$), and the Cartesian velocity $\underline{V}$ is kept constant for a sufficient time T for one of the articular variables to reach its limit.

6.3.4.2.1. Advantage of the projected gradient method

Remember that the projected gradient method enables us to reduce the articular velocities of those parts which are closest to the limits of their position (both uppermost and lowermost), by choosing a value of $\alpha \neq 0$. In order to demonstrate this advantage, we carry out simulations for two values of α $(\alpha = 0, \alpha \neq 0)$ and for two initial configurations such that:

– the first initial configuration of the arm, $\theta_0 = [1.9 \quad -1.3 \quad -0.9 \quad 0.78]$, is chosen with a single articular variable, for the boom, which is close to its uppermost limit of position. The second initial configuration of the arm, $\theta_0 = [1.9 \quad -1.3 \quad -1.6 \quad 0.78]$, is chosen with two articular variables – those for the boom and the stick – close to their limit positions;

– the Cartesian velocity required for the tool holder is chosen so as to bring the boom towards its limit $\left(\underline{V} = [-0.3 \quad 0] \text{ rad/s}\right)$.

The graphs respectively illustrate the evolution of certain articular variables in their respective spaces of definition, and their desired velocities (see Figure 6.19) for the first configuration. For $\alpha = 0$, we note that the articular position of the boom reaches its limit in 4 s, whereas for $\alpha \neq 0$, the boom reaches its limit after 6 s. the increase in the time before the boom reaches its limit position gives rise to a gain in the distance of the tool holder, which demonstrates the advantage to the projected gradient method. In addition, when $\alpha \neq 0$, we note that the articular variable of the boom is the most heavily penalized (that is to say, its articular velocity is reduced) and the movement of the tool holder is mainly due to the movement of the stick.

For the second configuration, where $\alpha = 0$, the boom reaches its uppermost limit position after 2.8 s, whereas when $\alpha \neq 0$, the stick reaches its lowermost limit position after 2.8 s. In both cases, the use of the projected gradient method plays no part in increasing the distance of the tool holder (see Figure 6.20).

6.3.4.2.2. Choosing the value of α and the cost function

After multiple simulations, we note that the second cost function offers more advantages than the first function, in spite of the fact that we need to handle two parameters: α and ϕ. For the same desired Cartesian velocity of the tool holder, if we look at the duration of the simulation, we note that there is a gain in the Cartesian

distance of the tool holder with the second function ($\alpha = 3$ and $\phi = 2$) in relation to the first function ($\alpha = 20$). However, if we increase the value of ϕ, we observe oscillations in the desired articular velocities.

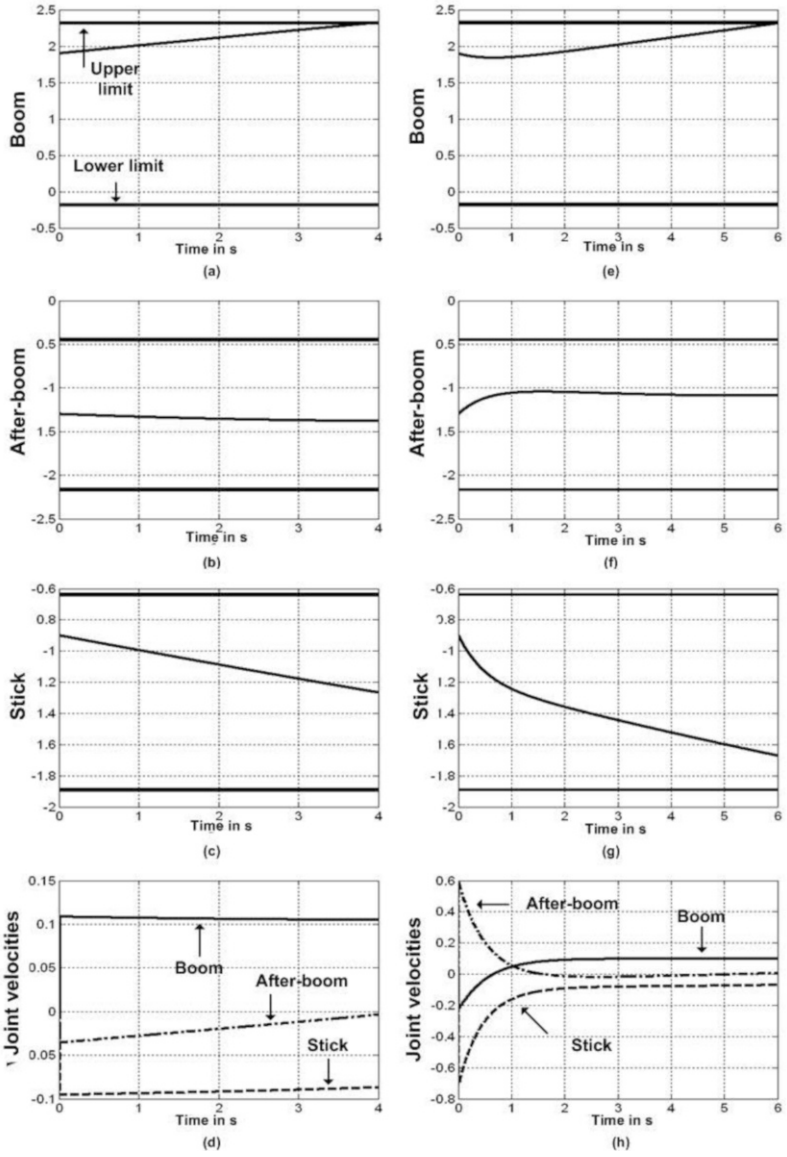


Figure 6.19. *Simulation in MECALAC1 mode when the boom is near to its uppermost limit position for* $\alpha = 0$ *(a, b, c, d) and* $\alpha \neq 0$ *(e, f, g, h)*

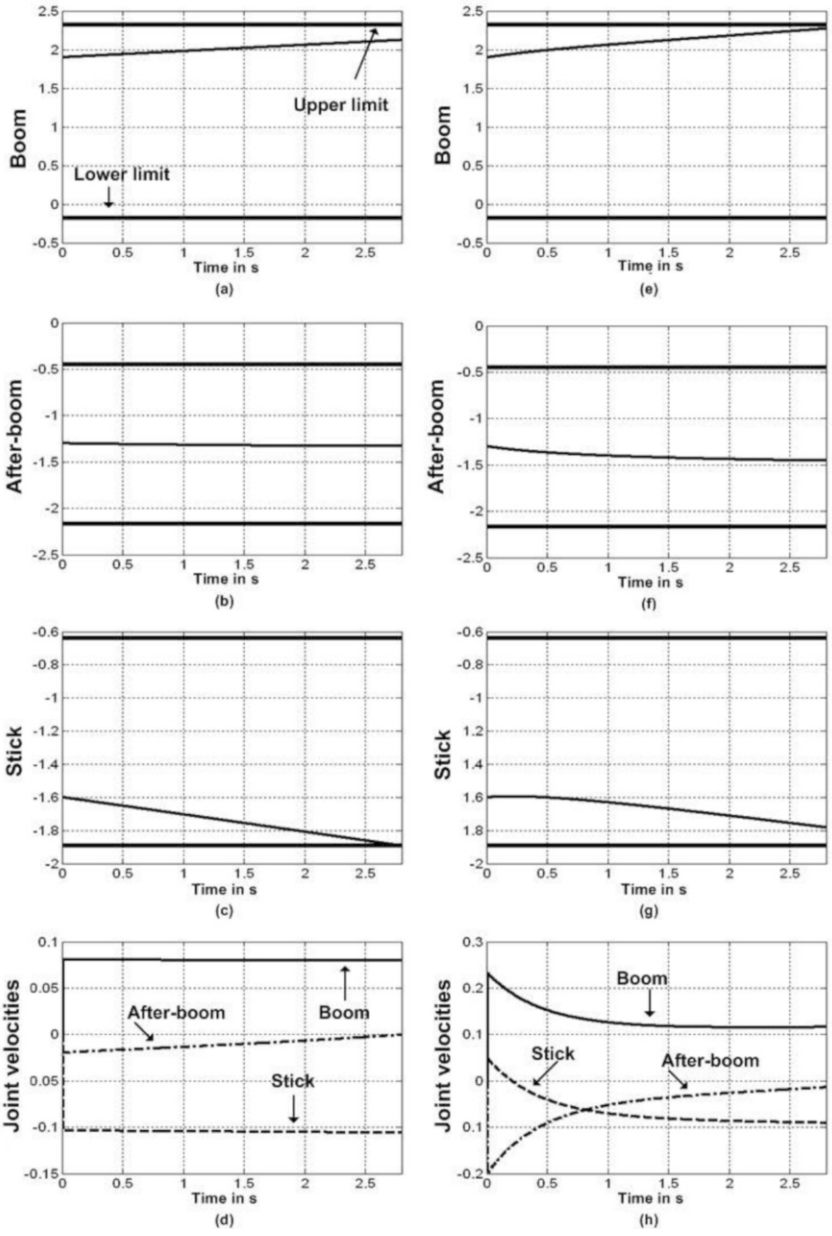


Figure 6.20. *Simulation in MECALAC1 mode when the boom and stick are near to their limit positions for* $\alpha = 0$ *(a, b, c, d) and* $\alpha \neq 0$ *(e, f, g, h)*

6.3.4.2.3. Limitation of the method

In order to study the limitation of the method, we impose a Cartesian velocity of the tool holder $\underline{V} = [0.2 \quad 0.3]$ m/s for a time of $T = 10$ s which is sufficient for one or more of the articular variables to reach their limit positions. Then we impose another velocity vector, $\underline{V} = [-0.3 \quad -0.2]$ m/s for $T = 22$ s, for the parts to move away from their limit positions. Figure 6.20 illustrates the evolution of an extract of the articular variables in their definition spaces, the desired articular velocities and the desired and actual direction of the tool holder.

Even if we reduce their velocities, the stick and the after-boom approach their uppermost limit positions, but when the stick reaches that point $(T = 10 \text{ s})$, the direction of the velocity of the tool holder is not preserved (see Figure 6.21(f)). When the velocity changes direction, the articular variables change until the boom reaches its lowermost limit position $(T = 30 \text{ s})$. This leads to the loss of direction of the desired velocity of the tool holder. Thus, when one of the parts reaches its uppermost or lowermost limit position, the other parts continue to move, causing the loss of the direction imposed by the drivers. Furthermore, at the moment when it reaches its absolute limit, we observe a sudden stop of the part in question, which decreases the lifecycle of the pistons.

6.3.4.2.4. Conclusion

The conversion of the commands for the velocity of the tool holder into articular velocities for the parts of the arm is "immediate" as long as the tool holder is in the accessible domain, i.e. as long as the parts of the arm are not on the boundary of their domains of definition. Thus, at these boundaries, the inverse kinematic model poses the following problems:

– the parts of the arm reach their uppermost or lowermost limit with non-null velocity, which causes them to come to a sudden stop;

– consequently, the direction of the velocity of the tool holder, imposed by the drivers with the joysticks, is not preserved.

The next section describes a procedure we advocate to avoid these types of problems.

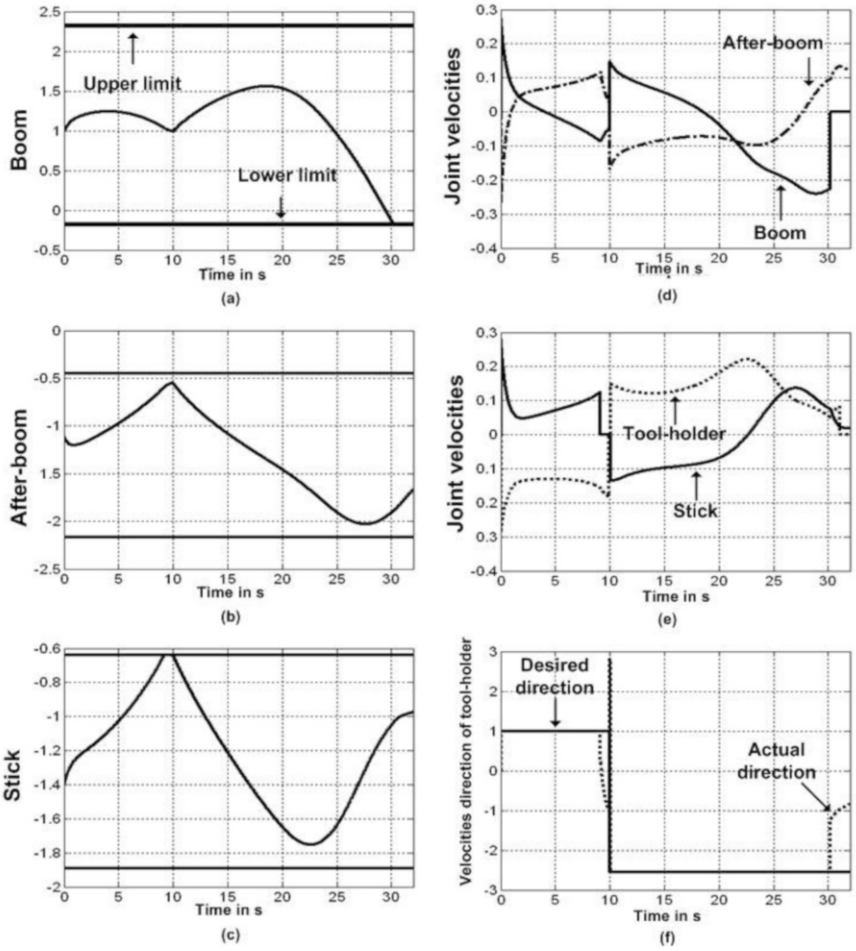


Figure 6.21. *Simulation in MECALAC1 mode to illustrate the limitation of the projected gradient method*

6.3.5. *Articular limits*

On the basis of the above, the articular velocities of the parts of the arm determined on the basis of the velocity of the tool holder and the inverse kinematic model pose a problem when the articular variables are on the boundary of their domains of definition, when the direction of the velocity of the tool holder imposed by the drivers is not assured. In order to avoid this problem, we propose to add a

procedure to the transformation between the desired velocity vector of the tool holder and the articular velocity vector.

The aim of this procedure is to make sure that when one of the articular variables reaches an area near to one of its limits, only the modulus of the velocity vector of the tool holder is decreased, so that as long as the velocity is not modified by the driver, its modulus reaches a zero value when the variable in question reaches its absolute limit. Such a procedure is intended to prevent any part reaching the absolute limit of its articulations with a non-null velocity.

6.3.5.1. *Modification of the articular velocity vector*

Remember that the coordinates of the torsor of velocities of the tool holder $\underline{V} \in R^3$, set by the joysticks at each moment of sampling ($t = kT$), comprise a d'un Cartesian velocity vector Vd and a velocity of rotation of the tool holder ωd, such that:

$$\begin{aligned} &\underline{V}^{\mathrm{T}}(kT) = \left[V_d(kT) \quad \omega_d(kT)\right] \\ &V_d(kT) = \left[V_x(kT) \quad V_y(kT)\right] = \|V\| U(kT) \\ &U = \left[\frac{V_x(kT)}{\|V\|} \quad \frac{V_y(kT)}{\|V\|}\right], \|V\| = \sqrt{(V_x)^2 + (V_y)^2} \end{aligned} \qquad [6.42]$$

where U is the unit direction of the Cartesian velocity vector and $\|V\|$ its modulus. Therefore, in order to alter the commands for the velocity of the tool holder without altering its direction, we need only adjust the modulus. The modified command for velocity of the tool holder becomes:

$$V_c(kT) = \lambda V_d(kT) \text{ where } 0 \le \lambda \le 1 \qquad [6.43]$$

because the vector of articular velocities of the parts of the arm is such that:

$$\dot{\theta}_d(kT) = J^{-1} V_d(kT) \qquad [6.44]$$

Hence, using equation [6.44], the vector of the modified articular velocities becomes:

$$\dot{\theta}_c(kT) = \lambda \dot{\theta}_d(kT) \qquad [6.45]$$

Consequently, we need only determine the value of λ, online, in order to modify the commands for the articular velocities whilst maintaining the direction of the velocity vector for the tool holder imposed by the drivers in carrying out a particular task. In other words, the value of λ must be equal to 1 when there is no modification, and as the joint approaches its absolute limit, it decreases progressively to arrive at that limit with a value equal to 0.

6.3.5.2. *Description of the procedure*

In order to know whether, for a given command of the velocity vector of the tool holder, one of the parts is approaching its limit, each time a sample is taken ($t = kT$), a prediction of the new configuration of the tool holder has to be calculated for each articular variable, using a linear integration with one step in advance. Thus, for each part, this prediction depends on the current articular position and the articular velocity determined by way of the inverse kinematic model:

$$\hat{\theta}_i(k+1) = \theta_i(k) + T\dot{\theta}_{id}(k) : i = 2,3,5,6 \qquad [6.46]$$

Let $\Theta_i = [\theta_{i\min} \quad \theta_{i\max}]$ represent the domain of definition of each of the articular variables (see Figure 6.22). In order to progressively decrease the articular velocity as the limit approaches, the domain of definition is divided into three parts: $\Theta_i = \Theta_{LIi} \cup \bar{\Theta}_i \cup \Theta_{LSi}$, such that the length Θ_{LIi} and Θ_{LSi} are two identical braking zones on either side of each limitation. This length of braking zone is defined by using a single parameter ρ such that:

$$\Theta_{LIi} = \Theta_{LSi} = \rho(\theta_{i\max} - \theta_{i\min}) \in \mathbb{R} \qquad 0 \prec \rho \prec \frac{1}{2} \qquad [6.47]$$

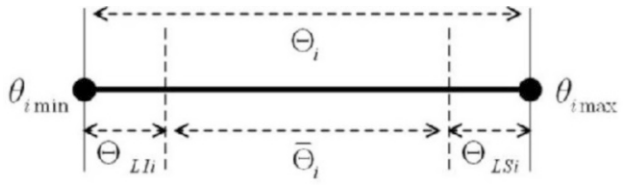


Figure 6.22. *Partition of the definition domain of a joint variable*

Thus, for each articular variable and depending on the prediction of the position $\hat{\theta}_i(k+1)$ in one of the three intervals, a unique value is attributed to λ_i such that if:

$-\ \hat{\theta}_i(k+1) \in \bar{\Theta}_i \Rightarrow \lambda_i = 1;$

$-\ \hat{\theta}_i(k+1) \geq \Theta_{i\max} \Rightarrow \lambda_i = 0;$

$-\ \hat{\theta}_i(k+1) \leq \Theta_{i\min} \Rightarrow \lambda_i = 0;$

$-\ \hat{\theta}_i(k+1) \in \bar{\Theta}_{LSi} \Rightarrow 0 \prec \lambda_i \prec 1;$

$-\ \hat{\theta}_i(k+1) \in \bar{\Theta}_{LIi} \Rightarrow 0 \prec \lambda_i \prec 1.$

Then, for all the parts of the arm, we obtain:

$$\lambda = \min\{\lambda_i\} \qquad [6.48]$$

Based on the above, the values of the articular velocities depend on the two variables ρ and λ. If the desired articular velocity enables us to move away from the limit point when the prediction $\hat{\theta}_i(k+1)$ is in the braking zone, tests on the sign of the articular velocity are added to the procedure (see Figure 6.23) so that the articular velocity is not altered.

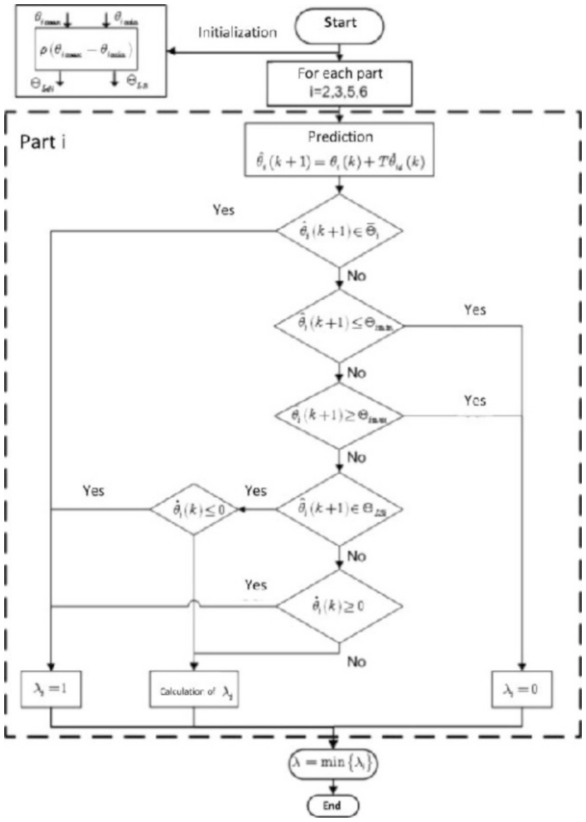


Figure 6.23. *Joint limit procedure*

6.3.5.3. *Algorithm*

The problem which arises for the calculation of λ_i is in the braking zone. What is the correct value for λ_i? Is this value unique? Which is the right method to use to

solve this problem? In order to answer these questions, and because we know that λ_i will decrease progressively to eventually arrive at a value of zero, two approaches to the calculation are tested.

APPROACH 1.– The first approach is to calculate, each time a sample is taken, the slope λ_i of the line between the prediction position of the articular variable and the variation of the braking zone (see Figure 6.24). Thus, λ_i is:

$$\lambda_i = \frac{\left[\theta_{im\max} - \hat{\theta}_i(K+1)\right] \quad ou \quad \left[\hat{\theta}_i(K+1) - \theta_{i\min}\right]}{\rho\left(\theta_{i\max} - \theta_{i\min}\right)} \qquad [6.49]$$

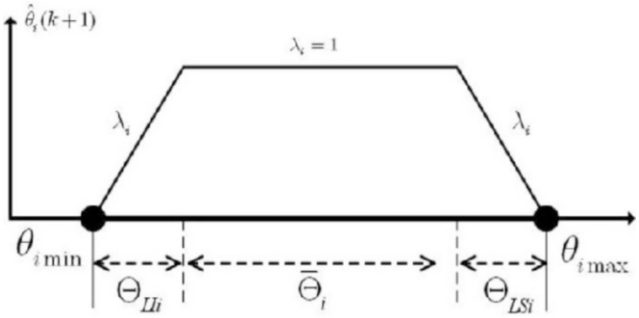


Figure 6.24. *Approach 1*

This approach is simple to carry out in real time: it ensures that the part will arrive with a null velocity at the limit points when $\hat{\theta}_i(k+1) = \theta_{i\max}$ or $\hat{\theta}_i(k+1) = \theta_{i\min}$, but depending on the values chosen for the ρ, it may lead to sharp decelerations.

APPROACH 2.– The second approach is based on the following remark: if the driver persists in imposing a velocity vector on the tool holder in the same direction, with the same module, then it is possible to predict the date after kT at which a particular articular variable will reach its limit value ($\theta_{i\max}$ or $\theta_{i\min}$) with a non-null velocity. Indeed, by way of a simple linear interpolation, we obtain:

$$\theta_{i\max}(kT+\delta) = \theta_i(k) + \delta\dot{\theta}_{id}(k) \qquad [6.50]$$

Note that it is always possible to "round off" δ to a whole number of sampling periods.

This approach would involve defining a time interval $\Delta > \delta$ and a third-order interpolation polynomial between the dates kT and $kT + (\Delta = nT)$ such that:

$$\begin{aligned} &\theta_{i\max}(kT+\delta) = \theta_i(k) + \Delta\dot{\theta}_{ic}(k) \Rightarrow \dot{\theta}_{ic}(k) < \dot{\theta}_{id}(k) \\ &t = kT : \tilde{\theta}_i(t) = \theta_i(k) \ , \ \dot{\tilde{\theta}}_i(t) = \dot{\theta}_{id}(k) \\ &t = kT + \Delta : \tilde{\theta}_i(t) = \theta_{i\max}, \ \dot{\tilde{\theta}}_i(t) = 0 \end{aligned} \quad [6.51]$$

where the third-order polynomial is of the form:

$$\tilde{\theta}_i(t) = a_3 t^3 + a_2 t^2 + a_1 t + a_0 \quad [6.52]$$

The analytical expressions of the coefficients of this polynomial can easily be obtained. This enables us to predict what the value of $\tilde{\theta}_i(k+1)$ would be, and thus to obtain:

$$\dot{\theta}_{ic}(k) = \frac{\tilde{\theta}_i(k+1) - \theta_i(k)}{T} \quad [6.53]$$

This leads us to define:

$$\lambda_i = \frac{\dot{\theta}_{ic}(k)}{\dot{\theta}_{id}(k)} \quad [6.54]$$

6.3.5.4. *Simulations and analysis of the method*

The implementation of the procedure in the simulation software involves adding the algorithm for calculating λ between the output of the vector of the desired articular velocity and the integrator (see Figure 6.25). Then, using the same parameters for the simulation (see Chapter 3), simulations are performed with the aim of choosing the better of the two approaches proposed when the articular variables are approaching their limits. With the first approach, the effect of varying ρ is tested, whereas with the second method, the effect of varying Δ is tested. During the simulations of each approach, the Cartesian velocity of the tool holder is $\underline{V}_1 = [0.4 \quad 0.3]$ m/s for $T_1 = 15$ s and then $\underline{V}_2 = [-0.3 \quad -0.4]$ m/s for $T_2 = 25$ s, with a value of $\alpha = 20$.

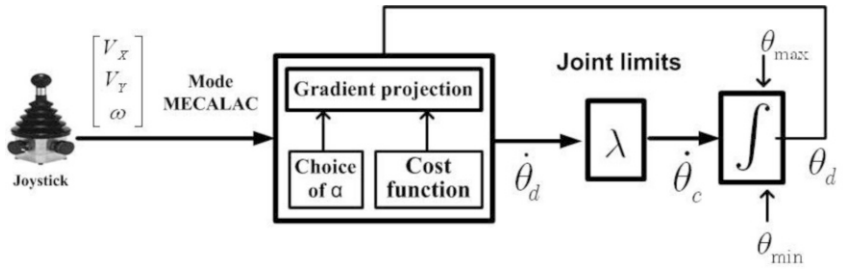


Figure 6.25. *Functional diagram of the simulation of the new system with the articular limits algorithm, performed on SIMULINK*

APPROACH 1.– With this approach, we examine the effect of varying ρ and its effectiveness. The graphs illustrate the evolution of the four articular variables in their definition spaces, their desired articular velocities, their desired modified velocities, the desired direction and current direction of the tool holder and the variation of λ for two different values of ρ (see Figures 6.26 and 6.27).

For both values, if we apply the command $\underline{V}_1$, the simulations show that when the after-boom and the stick enter the braking zone, the articular velocities of all the parts of the arm begin to slow down, until the parts in question arrive at their limits with a null velocity (see Figure 6.27(b)-(f)). Furthermore, the value of λ progressively decreases, reaching 0 when the modified articular velocities are at rest. When the direction of the velocity of the tool holder changes, $\underline{V}_2$, then the desired articular velocities are restored until the boom enters the braking zone, or all the articular velocities begin to slow down, reaching 0 when the boom reaches its limit (see Figure 6.27 (b)-(f)). The desired and current direction of the velocity of the tool holder remain constant throughout the duration of the simulation.

In addition, the simulations show that the variation of ρ plays a significant part in slowing down the articular velocities. The more the value of ρ increases, the less drastic the deceleration. However, this increase enlarges the braking zone ($\Theta_{LIi}, \Theta_{LSi}$) and decreases $\bar{\Theta}_i$ when the velocities are not modified.

APPROACH 2.– With this approach, the same simulations are repeated with the aim of studying the effect of the variation of Δ and the efficacy of the approach for the proposed system. The braking zone is set in stone ($\rho = 0.15$) (see Figures 6.28 and 6.29).

When the two parts (the after-boom and the stick) enter the braking zone, and for both values of Δ, all the desired articular velocities begin to decelerate, but as they

approach their limits, all the parts stop abruptly. Even with a variation in Δ, the modified articular velocities decrease less quickly if Δ increases, but there is no assurance that the absolute limit will be attained with a null velocity (see Figure 6.29(d)-(h)).

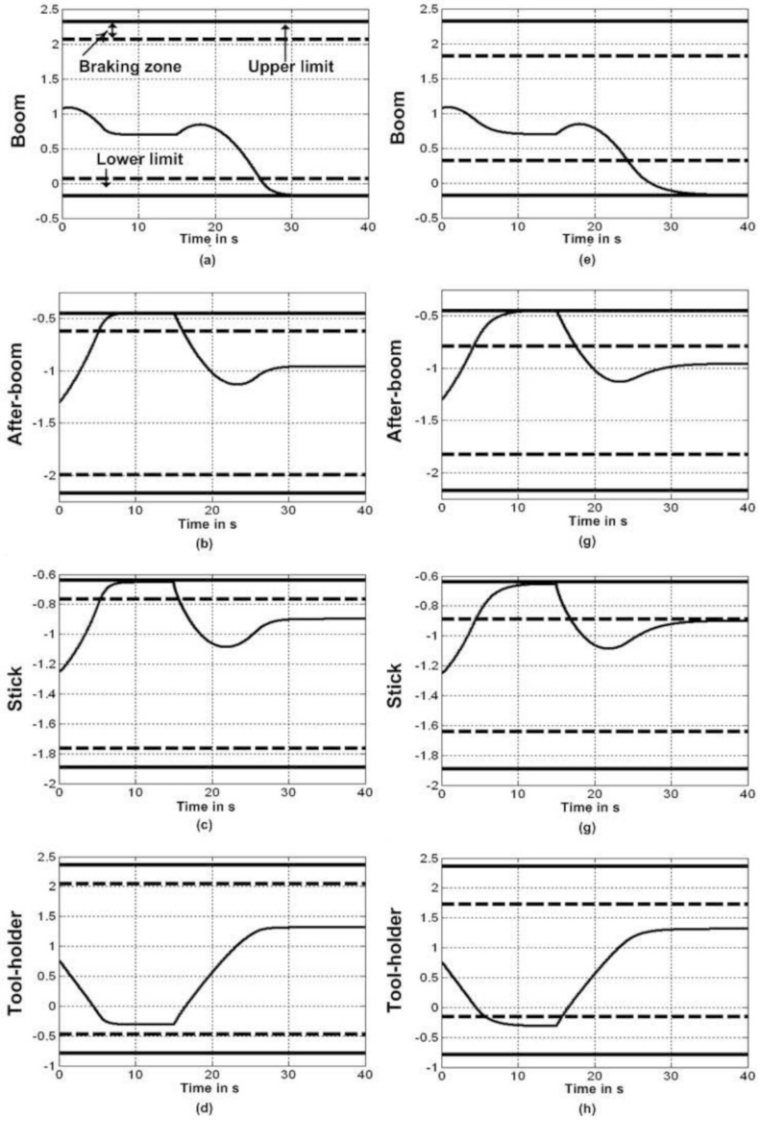


Figure 6.26. *Joint positions of the arm parts:*
$\rho = 0.1$ *(a)-(b)-(c)-(d);* $\rho = 0.2$ *(e)-(f)-(g)-(h)*

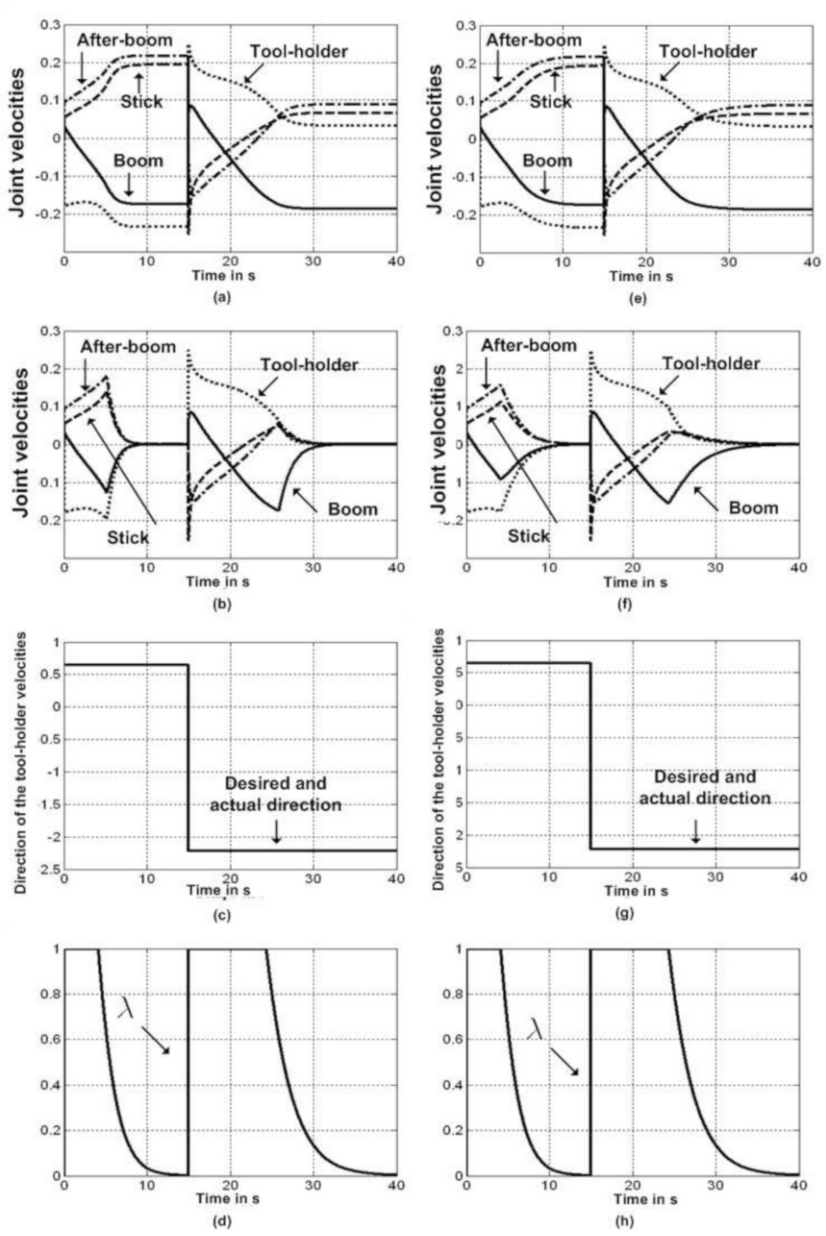


Figure 6.27. *Desired joint velocities: (a)* $\rho = 0.1$ *, (e)* $\rho = 0.2$ *; modified joint velocities: (b)* $\rho = 0.1$ *; (f)* $\rho = 0.2$ *; velocity direction of the tool-holder: (c)* $\rho = 0.1$ *, (g)* $\rho = 0.2$ *; variation of* λ *: (d)* $\rho = 0.1$ *, (h)* $\rho = 0.2$

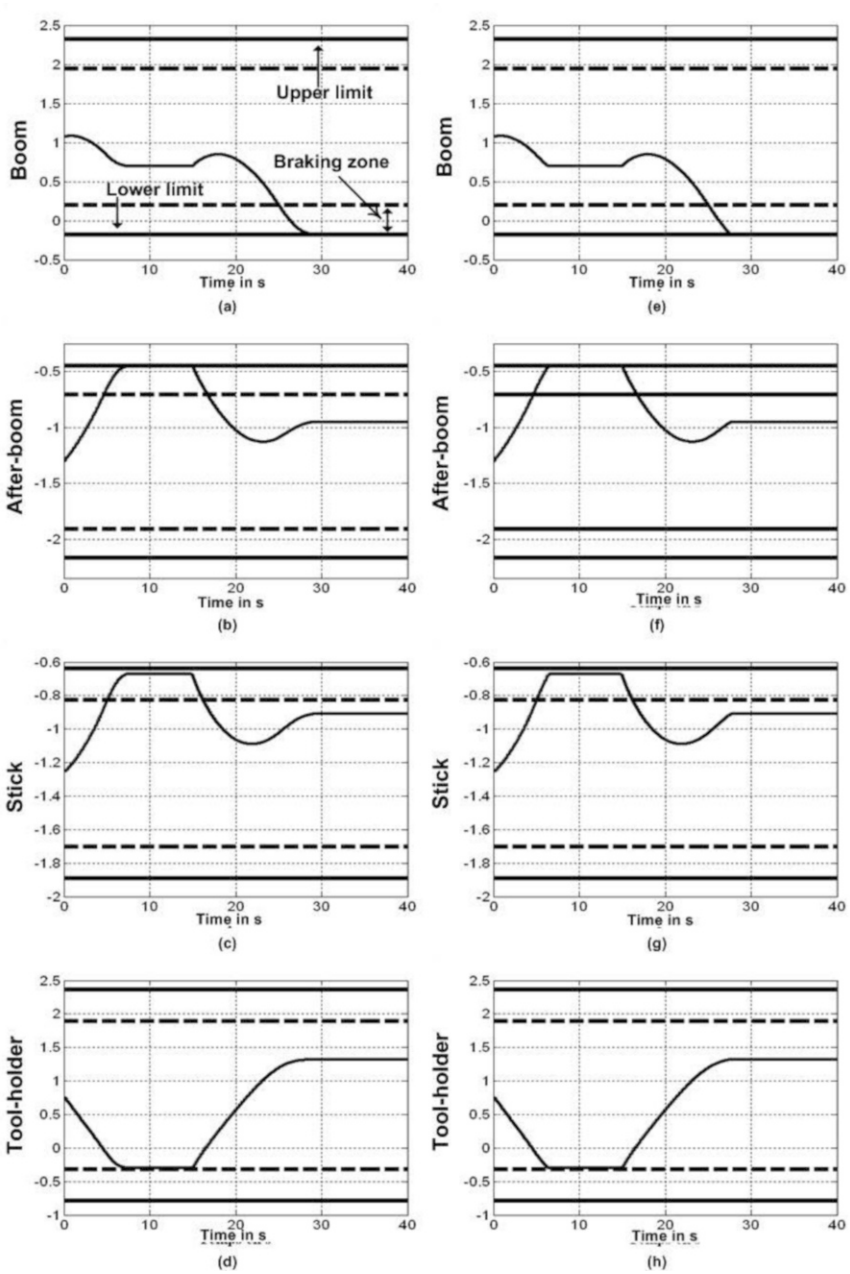


Figure 6.28. *Joint positions of the arm parts:* $\Delta = 10$ *(a)-(b)-(c)-(d);* $\Delta = 30$ *(e)-(f)-(g)-(h)*

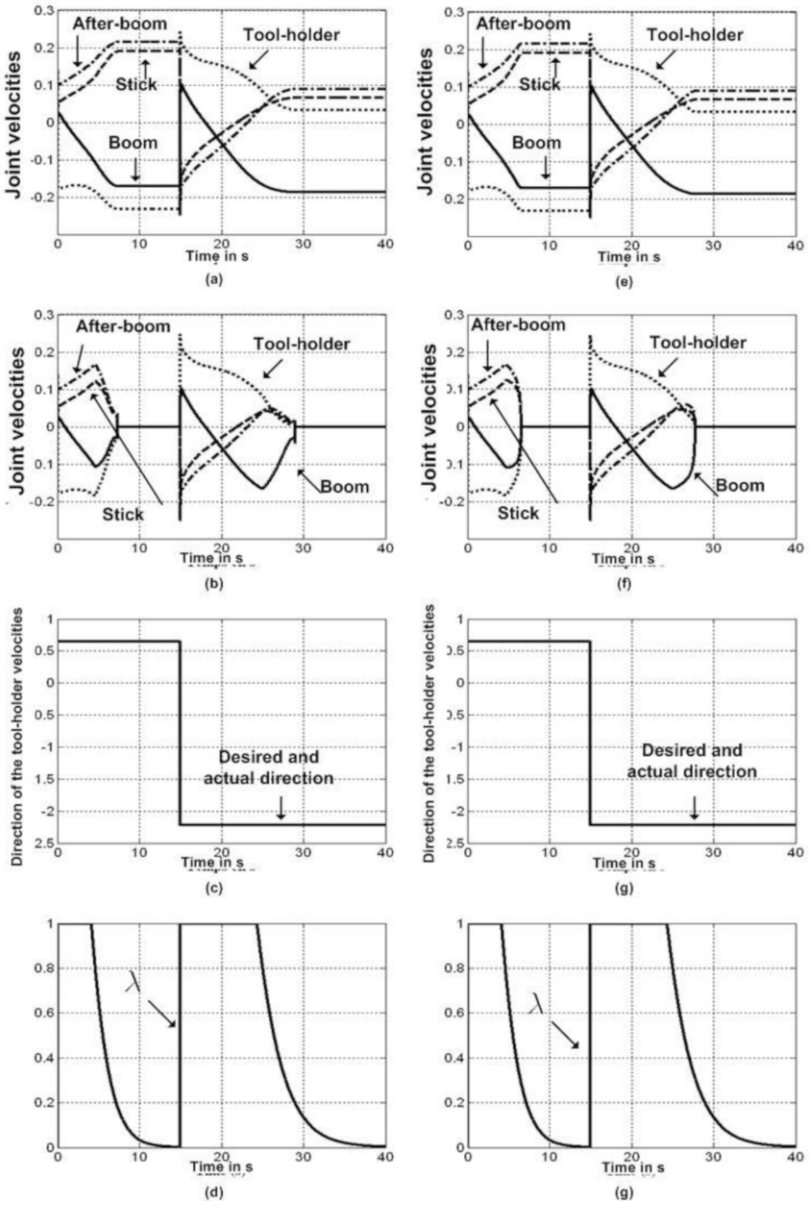


Figure 6.29. *Desired joint velocities: (a)* $\Delta = 10$ *, (e)* $\Delta = 30$ *; modified desired joint velocities: (b)* $\Delta = 10$ *, (f)* $\Delta = 30$ *; velocity direction of the tool-holder: (c)* $\Delta = 10$ *, (g)* $\Delta = 30$ *; variation of* λ *: (d)* $\Delta = 10$ *, (g)* $\Delta = 30$

6.3.5.4.1. Conclusion

The addition of the procedure is necessary so that each part of the arm will reach its limit position with a null velocity. This procedure decreases the modulus of the velocity vector of the tool holder whilst preserving its direction. On the basis of the simulations, we can state that:

– with the first approach, the commands for the desired articular velocities decrease progressively, and the part reaches its limit with a null velocity;

– with the second approach, the commands for the desired articular velocities decrease progressively, but after a certain period of time, they stop abruptly, and this depends on the value of Δ. Hence, there is no guarantee that the part will reach its limit with a null velocity.

Thus, the first approach is the only one which guarantees the part will reach its limit with a null velocity, even if this sometimes involves sharp deceleration. This deceleration depends on the value chosen for ρ.

6.3.6. *Limits of the articular velocities*

Each part of the arm is moved with a linear hydraulic actuator (a piston). The power required to move these pistons is derived from an engine, feeding two hydraulic pumps. The first pump provides the flowrate required to move the turret and operate the brakes and the swing. The second pump provides the flowrate required by the rest of the arm.

With the proposed system, we need to be vigilant with regard to the total flowrate required by all the pistons of the second pump. That is, when a certain number of pistons are engaged, the velocities demanded of the pistons are not feasible. This is due to the total sum of the flowrates required by the pistons exceeding the flowrate that the pump can supply. In this case, the pump reduces its production in a random manner. This type of pump is called a *load-sensing* pump [FAI 87; RYH 91]. Consequently, when the flowrate required exceeds the pump's maximum capacity, the power requirement is reduced in order to correspond to the maximum power that the pump can provide.

In order to avoid this problem, a procedure is added, ensuring that when the required flowrate is greater than the maximum flowrate, all the flowrates will be penalized by the same coefficient, with the aim of preserving the direction of velocity of the tool holder. Thus, the modified total flowrate becomes:

$$Q_{\text{modified}} = \gamma Q_d \qquad [6.55]$$

In the wake of the modification of the flowrate with each piston, the command for the articular velocity of each articulation needs to be modified. The next part of this section is devoted to the realization of this procedure, which requires us to first determine the velocity of each piston on the basis the desired articular velocity, $\dot{\theta}_{id}$. Then, the required flowrate in each piston is predicted.

6.3.6.1. *Relations between the configuration of the arm and the "lengths" of the pistons*

The length of the piston (the straight segment between the two points of attachment to the arm parts) determines the angle of rotation of each joint, and consequently the position and orientation of the tool holder in the space.

Numerous researchers have developed analytical relations, drawing a connection between the lengths of the hydraulic pistons and the articular variables of each joint by geometrical reasoning. The next step is to obtain the velocities of the pistons with a simple derivation of these relations in relation to time (time-derivation) and in relation to the articular variables [HEM 92; TAK 94].

Thus, similarly, all the analytical models to calculate the lengths of the pistons on the basis of the articular variables of the arm are obtained. For instance, the length of the boom piston is:

$$BD = L_F = \sqrt{\|AB\|^2 + \|AD\|^2 - 2\|AB\|\|AD\|\cos(\alpha + \beta)} \qquad [6.56]$$

where $\|BC\|$, $\|AB\|$ and $\|DC\|$ are constant lengths, while $\|AC\|$ and $\|AD\|$ are lengths calculated as a function of θ_2. Also, the two angles α and β are:

$$\alpha = \cos^{-1}\left(\frac{\|AB\|^2 + \|AC\|^2 - \|BC\|^2}{2\|AB\|\|AC\|}\right) \text{ and } \beta = \cos^{-1}\left(\frac{\|AD\|^2 + \|AC\|^2 - \|DC\|^2}{2\|AD\|\|AC\|}\right).$$

Note that these analytical relations are difficult to implement in a real-time system because they necessitate a great many operations and therefore take a significant amount of time to calculate. Thus, their time derivatives are difficult to obtain analytically, and it impossible to then implement them in a real-time system.

To remedy this problem, we use the experimental values, measured on the arm of the MECALAC machine and which represent the lengths of the pistons as a function of the articular variables. The graphs in Figure 6.30 illustrate the length of each piston as a function of the articular variables measured experimentally (solid line) and calculated by analytical relations (dotted line). We note that the measurements

taken on the arm correspond quite closely with the analytical relations obtained by geometric reasoning.

The method involves interpolating the experimental values to create a polynomial function with n degrees to predict the length of the piston on the basis of the articular variable θ_i :

$$L_{i\_piston} = f(\theta_i) = a_n {\theta_i}^n + a_{n-1} {\theta_i}^{n-1} + a_{n-2} {\theta_i}^{n-2} + \ldots\ldots + a_2 {\theta_i}^2 + a_1 \theta_i + a_0 \quad [6.57]$$

The velocity of the piston is therefore the time-derivative of this expression:

$$\dot{L}_{i\_piston} = \frac{df(\theta_i)}{d\theta_i}\frac{d\theta_i}{dt} = \left(n a_n {\theta_i}^{n-1} + (n-1) a_{n-1} {\theta_i}^{n-2} + \ldots\ldots + 2a_2 \theta_i + a_1\right)\dot{\theta}_i \quad [6.58]$$

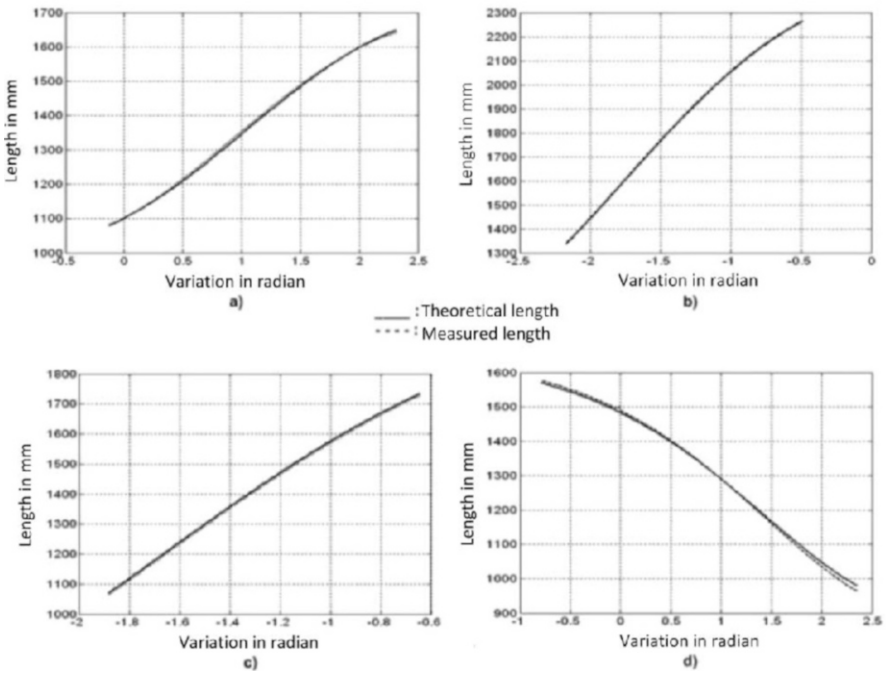


Figure 6.30. *Theoretical and experiment representations of the piston lengths of the MECALAC 12MXT: (a) boom, (b) after-boom, (c) stick, (d) tool-holder*

Table 6.2 shows the degree and all the coefficients of the interpolation polynomial for all the pistons. The graphs in Figure 6.31 illustrate the length of each piston measured experimentally and that predicted by the polynomial function [6.57] on the basis of the articular variables. We observe that the predicted values are very close to the measured values. Thus, the lengths of the pistons and their velocities will be predicted using these polynomial functions. For real-time calculations, these functions are easier to calculate than the analytical models.

	Degree	a_4	a_3	a_2	a_1	a_0
Boom	**3**	**0**	**-32.7**	**102.6**	**174.4**	**1101.9**
After-boom	**3**	**0**	**-67.795**	**268.532**	**294.3517**	**857.2847**
Stick	**3**	**0**	**58.6**	**-311.8**	**-50.5**	**1880.2**
Tool-holder	**4**	**5.8**	**-5.3**	**-49.6**	**-143.8**	**1483.3**

Table 6.2. *Definition domain of the machine joint variables*

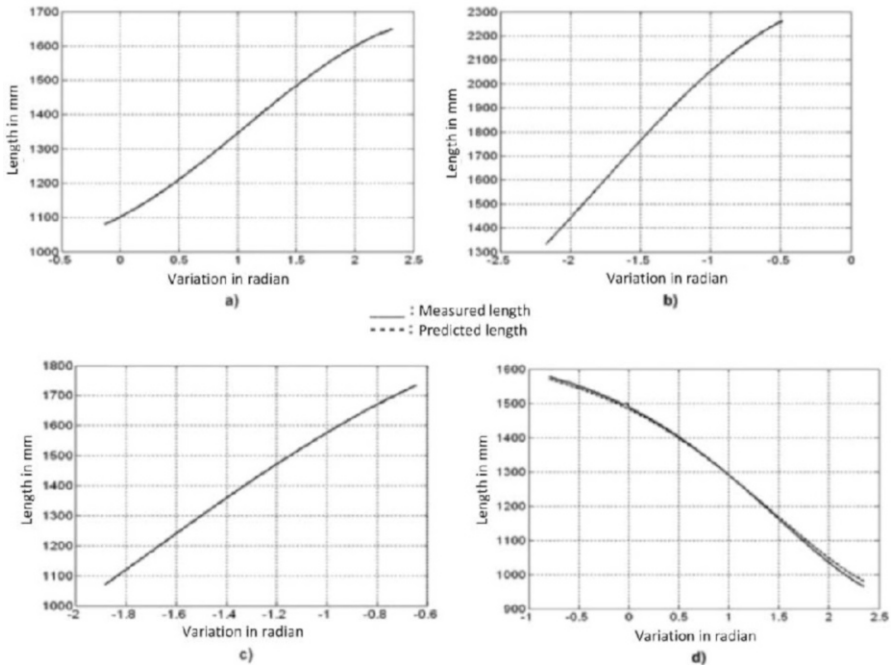


Figure 6.31. *Theoretical and experiment representations of the piston lengths of the MECALAC 12MXT: (a) boom, (b) after-boom, (c) stick, (d) tool-holder*

6.3.6.2. *Estimation of the total required flowrate*

Each linear hydraulic piston used to move a part C_i of the arm is composed of two elements: a fixed element attached to the previous part C_{i-1} and a moving element (the stem of the piston) attached to part C_i. These two elements are connected to the pump and to a tank by pipes. To move the piston stem at a particular desired velocity, the pump provides a flowrate Q_i of a fluid which enters one or the other of these two elements, depending on the desired direction of motion of the stem (see Figure 6.32). Thus, this flowrate exerts a pressure P_i, causing the stem to move in the direction sought.

In general, the flowrate necessary for the desired movement of the piston stem depends on the surface of the piston and on the desired velocity of the piston [FAI 87]. However, because the pipes connect the piston to the different elements of the hydraulic system (servo-valve, pump, tank, etc.) where the hydraulic fluid circulates, the flowrate will be reduced when this fluid is compressed. This reduction is proportional to the volume in which the fluid runs and to the degree of change in pressure. Thus, the flowrate Q_i is:

$$Q_i = \begin{cases} Q_{i1} = A_{i1}\dot{L}i_{\_piston} + \dfrac{V_{i1}}{\beta}\dot{P}_{i1} & \text{if } \dot{L}i_{\_piston} \succ 0 \\ Q_{i1} = A_{i2}\dot{L}i_{\_piston} - \dfrac{V_{i2}}{\beta}\dot{P}_{i2} & \text{if } \dot{L}i_{\_piston} \prec 0 \end{cases} \quad [6.59]$$

where A_{i1} and A_{i2} are, respectively, the surfaces of the two chambers (left and right) of each piston in the arm (see Table 6.3), β denotes the bulk modulus of the fluid, and V_{i1} and V_{i2} are, respectively, the volumes of fluid in the left and right chambers, increased by the dead volumes of the supply pipes [FAI 87; MER 76].

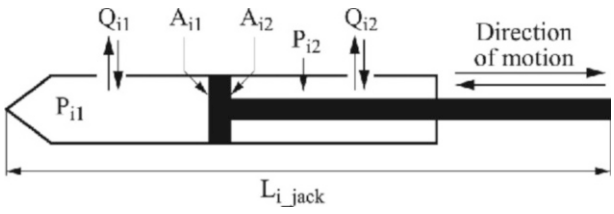


Figure 6.32. *Standard linear hydraulic piston*

	A_{i1} **in cm²**	A_{i2} **in cm²**
Boom	**220**	**94**
After-boom	**50**	**22**
Stick	**52**	**23**
Tool-holder	**40**	**18**

Table 6.3. *Surface estimation of the two chambers of pistons*

In order to obtain an estimation of the required flowrate without measuring the pressure in the two chambers, expression [6.59] is reduced to:

$$Q_i \simeq \begin{cases} Q_{i1} \simeq A_{i1}\dot{L}_{i\_piston} & \text{if } \dot{L}_{i\_piston} \succ 0 \\ Q_{i1} \simeq A_{i2}\dot{L}_{i\_piston} & \text{if } \dot{L}_{i\_piston} \prec 0 \end{cases} \quad [6.60]$$

Thus, the total flowrate required of the pump is the sum of all the flowrates required by each piston:

$$Q_d = Q_2 + Q_3 + Q_5 + Q_6 \quad [6.61]$$

6.3.6.3. *Procedure*

In order to solve the problem of the maximum power of the hydraulic pump, at each sampling time, the total flowrate required by the pistons on the arm (Q_d) is compared to the maximum flowrate that the pump is capable of delivering ($Q_{\max}$) where two cases arise: $Q_d \prec Q_{\max}$ and $Q_d \succ Q_{\max}$ (see Figure 6.33).

Following this alteration of the flowrate, it is easy to prove that the modified velocity of the piston and the modified articular velocity become:

$$\dot{L}_{i\_\text{piston}\_\text{modif}} = \gamma \dot{L}_{i\_\text{piston}} \Leftrightarrow \dot{\theta}_{ci} = \gamma \dot{\theta}_{di} : 0 \le \gamma \prec 1 \quad [6.62]$$

6.3.6.4. *Simulations and analysis of the method*

This procedure was implemented in the simulation software after the articular limit procedure. The choice to carry it out at this point was an arbitrary one, because the tests on the articular limits and the limit of the pump will be carried out at every sampling step, so the order in which these procedures take place is not important. The simulation parameters are identical to the previous simulations (see section 6.3.4.2). During the simulations of this procedure, the Cartesian velocity

of the tool holder, $\underline{V_1} = [0.4 \quad 0.3]$ m/s , is kept constant for a duration of $T_1 = 15$ s. Then it is altered: $\underline{V_2} = [-0.3 \quad -0.4]$ m/s , for a duration of $T_2 = 25$ s, with $\alpha = 3$.

With the first velocity vector, $\underline{V_1}$, the simulations demonstrate that when all the parts are moving (see Figure 6.34), the flowrate required to ensure the movement of all these parts is greater than the pump's maximum capacity. Thus, the articular velocities are all penalized by the same coefficient in order to reduce the velocities whilst maintaining the same direction (see Figure 6.34 (h)).

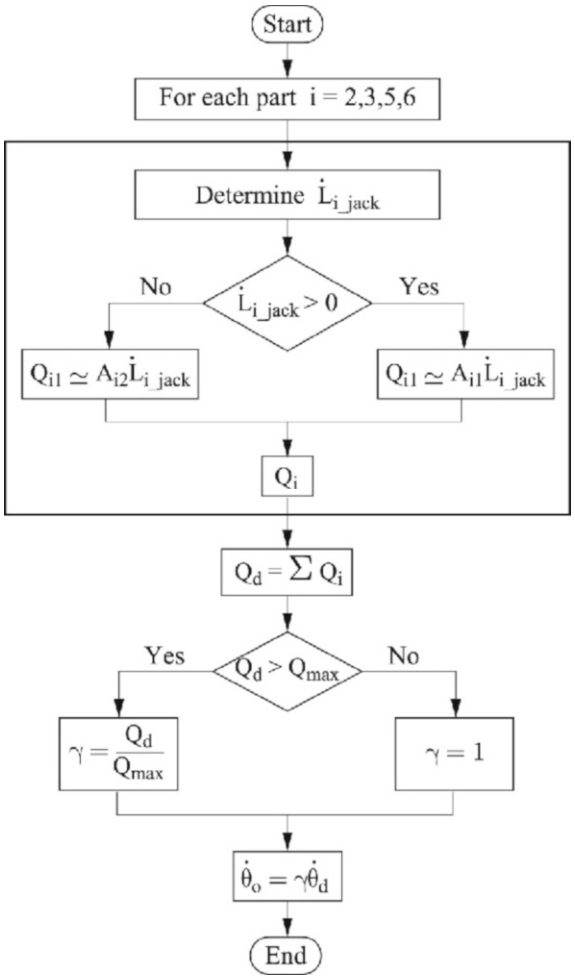


Figure 6.33. *Procedure algorithm of the flow ate modification required of the pump*

Similarly, when the velocity vector of the tool holder changes direction, $\underline{V}_2$, the articular velocities are penalized so that the maximum capacity of the pump is not surpassed.

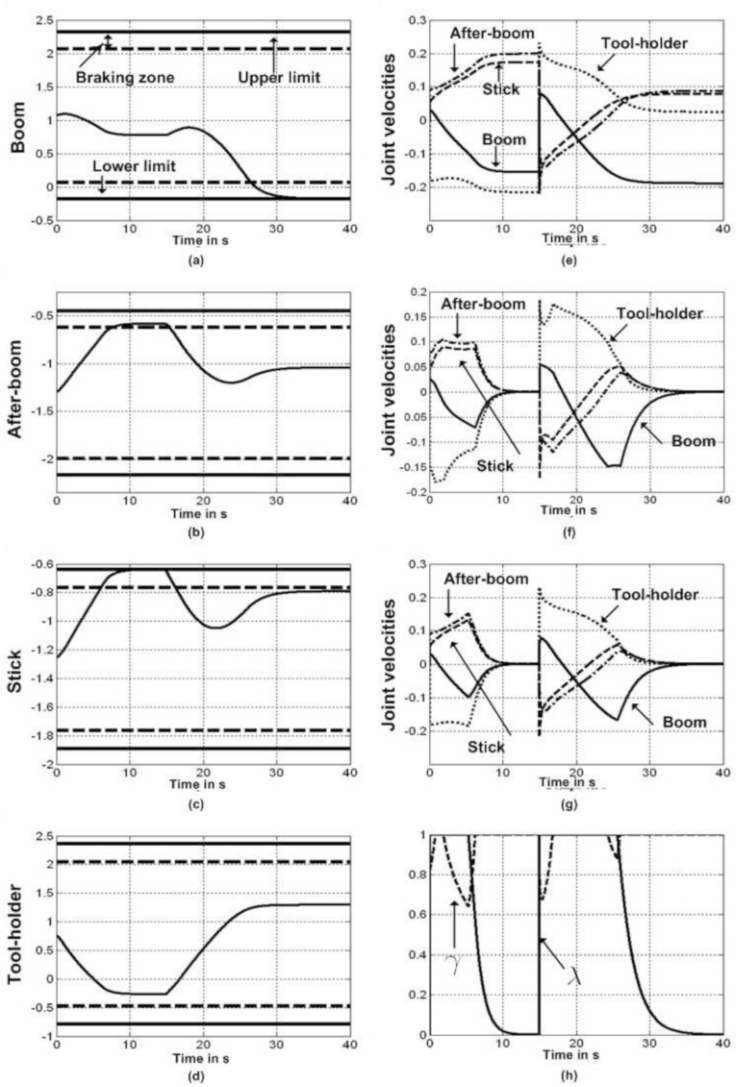


Figure 6.34. *Joint positions: (a)-(b)-(c)-(d); desired joint velocities: (e); desired joint velocities on exiting the block "power imit": (f); desired joint velocities on exiting the block "joint limits": (g); evolution of λ and γ: (h)*

6.3.7. *3D simulation*

In the context of simulation of the proposed system, it is interesting to test and verify all the algorithms developed by setting the machine in motion, in a virtual environment. In order to accomplish this task, we need to couple the model developed in SIMULINK with a three-dimensional (3D) animation. The *Virtual Reality Toolbox* associated with SIMULINK is able, during simulation of the proposed system, to recover specific signals which will serve to animate the 3D model of the MECALAC machine. We need three elements in order to adapt these signals to the virtual model:

– SIMULINK blocks to couple the model created in SIMULINK with the virtual reality environment created with the software "VR-Builder";

– VR-Builder to create 3D environments;

– VR-Viewer to view the 3D animation.

6.3.7.1. *Modeling of the machine in 3D*

The 3D modeling of the machine is performed in VRML (Virtual Reality Modeling Language). The model is obtained either by developing code or by creating it with the suite "V-Realm Builder". This package enables users to draw objects in 3D and then generate the VRML code that corresponds to those objects. Each object created has a set of parameters that can be modified during the simulation, such as the position, dimension and spatial orientation in relation to a reference framework of the scene.

One of the tools available in this builder is a tree-type structure, which means we can use the "parent/child" technique during the simulation and the assembly of the moving parts (see Figure 6.35). For instance, if the 3D model of any mechanical arm comprises two parts (C_1 and C_2), then C_2 is the "child" of the "parent" part C_1. If C_1 is moving, C_2 is too; yet if C_2 is moving, C_1 remains immobile. In view of the specifications of this software, the 3D model of the MECALAC machine is made up of two parts:

– fixed part: chassis, wheels, engine;

– moving part: the turret, the parts of the arm and the hydraulic pistons.

With account taken of the "parent/child" technique, the 3D model of the MECALAC machine is generated by using the geometric data of all the elements (see Figure 6.36).

The most difficult aspect of modeling the machine relates to the pistons, which are not all identical. Each piston is made up of several parts, each one being located on a different level of the tree structure, where all the attachment points of all the parts are

rotoid joints. For example, the piston of the stick comprises two parts – the fixed part is attached to the swing and the moving part to the stick. In the tree diagram, the fixed part is on the same level as the swing (the parent) and the moving part is on the same level as the stick (the child of the swing). The same method is used to model the after-boom piston.

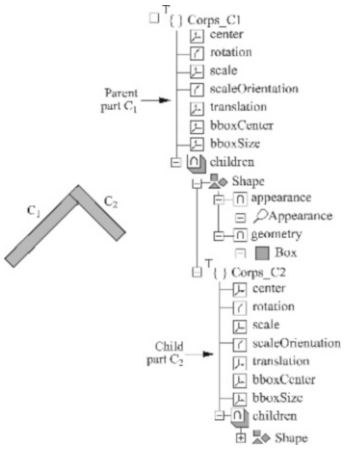


Figure 6.35. *Example of a model in VRML Builder of a mechanical arm comprising two parts*

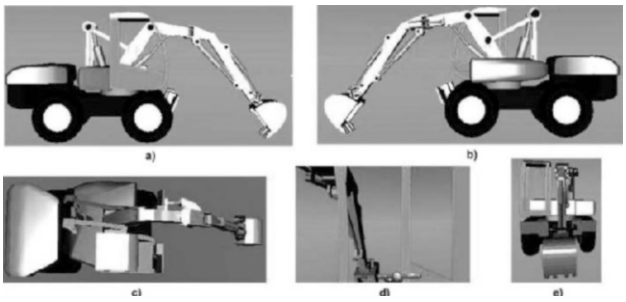


Figure 6.36. *3D model of the MECALAC machine: (a) right-hand view; (b) left-hand view; (c) bird's-eye view; (d) view from the cabin; (e) head-on view*

Unlike the previous two pistons, the tool holder piston has two additional parts: a lever and a piston. The lever, along with the fixed part of the piston and the stick, makes up the "stick" group. The piston, along with the moving part and the tool holder, belongs to the "tool holder" group, which is the child of the "stick" group.

Unlike all the others, the boom actuator is in the form of a diamond, which comprises the piston, the lever and a piston. This group is attached to the body of the

boom by way of the piston, while the other parts of the piston are attached to the chassis. Along with the chassis, they form a subset which is the "parent" of the subset formed by the piston and the boom.

6.3.7.2. *Animation*

In order to animate the model of the machine created, we have to link it to the model of the proposed system, by way of a block from Virtual Reality Toolbox. This block requires us to indicate firstly the address of the VRML file from "VR-Builder" and secondly the parameters of animation of each element of the machine (see Figure 6.37).

The animation of each element requires the angle of rotation that the element has to perform. The value of this angle is gleaned from the output of the simulation program at each sampling time. The articular position of each part of the arm resulting from the desired articular velocity is sent to the 3D model to position that part within the space. In order to position each piston in the space, we have to position all the parts of that piston.

Thus, on the basis of the desired articular position of the arm part, the length of the piston is determined. Then, the articular position of each part of the piston in relation to the part to which it is attached is determined using the theoretical equations. All these data are sent to the virtual environment to animate the model by indicating which part needs to move. As all these parts have rotoid joints, we need only put a checkmark for the "rotation" of each part (see Figure 6.37).

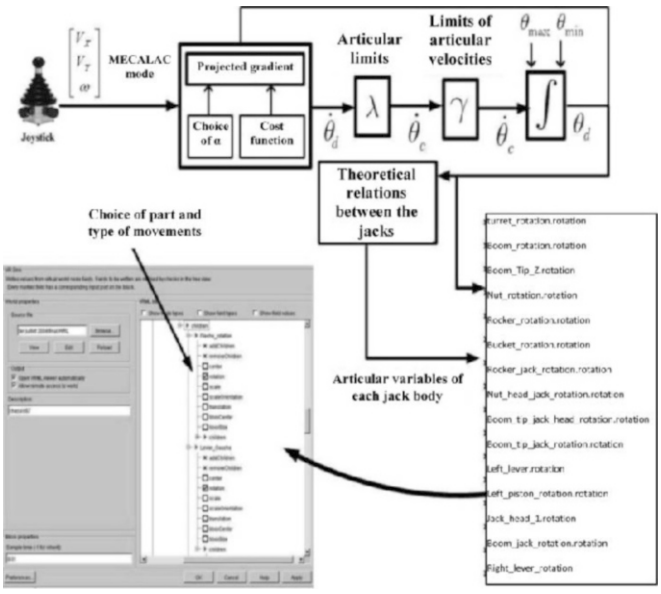


Figure 6.37. *Precision block for animating the parts*

6.3.7.3. *Result of the 3D simulation*

The 3D model of the MECALAC machine was implemented in the simulation suite. At each sampling time and during the simulation for a desired Cartesian velocity of the tool holder, a window is opened automatically and the arm of the machine moves in the direction chosen by the driver, in accordance with the view chosen by the user (see Figure 6.38 (a), (b), (c)). Furthermore, we added a second simulation window to inform the driver about the situation of the simulation. In this window, the extent of each articular variable is illustrated. Therefore, the user is able to view online the evolution of these articular variables, and alert messages are displayed when the articular variables are in the braking zones or if there is a limitation of the hydraulic pump (see Figure 6.38 (d), (e), (f)).

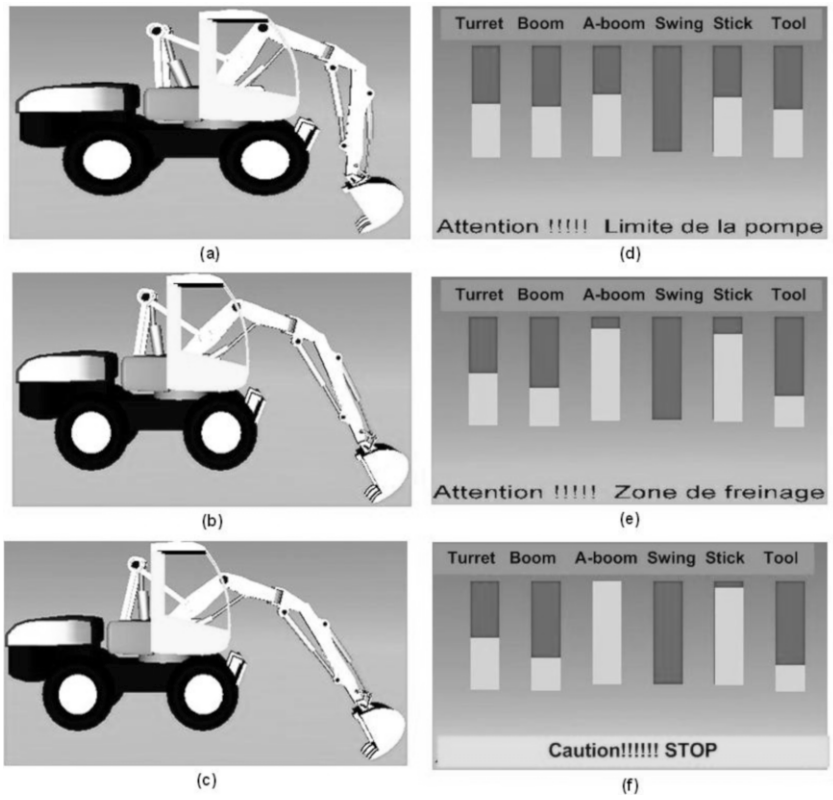


Figure 6.38. *3D simulation of the MECALAC machine*

6.3.8. *Onboard computer architecture*

The new functionalities to assist in the driving of the MECALAC machine, put forward above, require a real-time computer architecture to be designed for an onboard control-command system based on a communication bus. This system needs to have a real-time kernel to execute numerous algorithms and assure information exchanges between the different pieces of equipment with as fast a run time as possible.

6.3.8.1. *Overview of other projects*

There are many onboard systems designed for experimental building machinery. The majority of them use industrial computers connected to a fieldbus to carry out the control-command of the machine.

Ha *et al.* propose an onboard system made up of M200 digital controllers communicating with an industrial computer via a CAN (controller area network) at 250 Kb/s [HA 02]. The data acquisition and control algorithms are developed in C++ on the operating system Windows NT. The sampling frequency chosen for the data acquisition is 10 ms.

Another project is LUCIE. This is an excavator-loader with three onboard PC104 computers communicating *via* a CAN bus. Each computer is responsible for an independent task. The first is devoted to the driving, the second to GPS localization and the last takes care of specific obstacle detection by means of a laser sensor [SEW 88; SEW 96; SEW 00].

6.3.8.2. *Structure of the system*

The implementation of the interface system on a MECALAC machine requires an onboard computer, a touchscreen, sensors and joysticks. The onboard system put forward must perform the following tasks in this order (see Figure 6.39):

– gather the input data and channel them to the onboard computer. These inputs are:

- five articular position sensors, used to obtain the absolute angular position for the turret, boom, after-boom, stick and tool holder,

- two joysticks to obtain the three components of the velocity vector of the tool holder,

- digital input used for acquisition of logical data from the push-buttons on the joysticks;

– execute the interface algorithms on an industrial computer (resolved motion-rate control, height indicator, limitation of the work space, etc.);

– send the results of the algorithms on the computer to the appropriate outputs. These outputs are:

- a microbot: capable, on the one hand, of controlling the switch between the old and new method of piloting of the tool holder and, on the other hand, providing these outputs with chopper-type command signals (known as pulse width modulation or PWM) to control an electro-valve. This valve serves to regulate the distributor of each hydraulic actuator for each part of the arm,

- digital output capable of controlling low-powered pre-actuators: contactors, relays, etc.

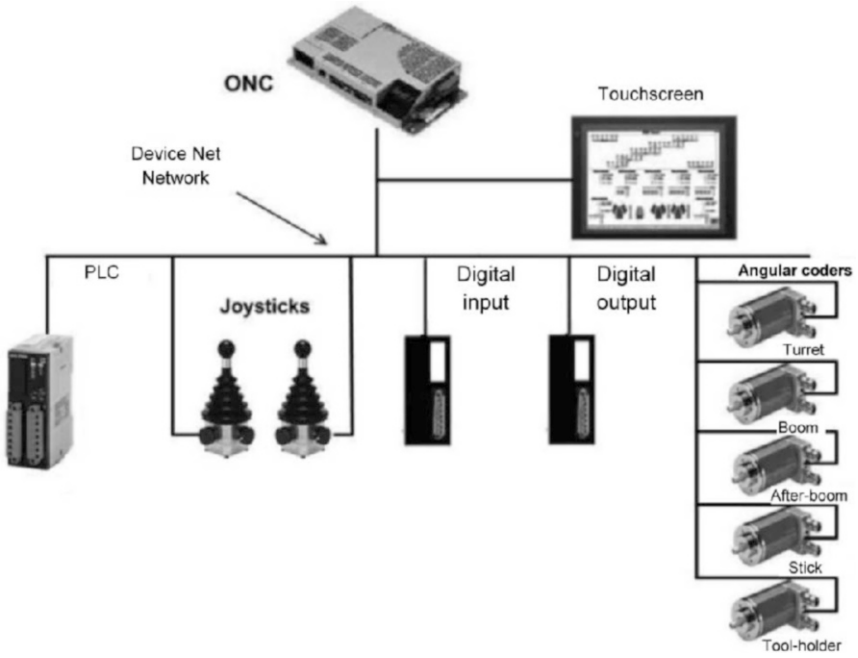


Figure 6.39. *Architecture of the onboard system*

The onboard industrial computer used is an ONC (open network controller). It has various communication ports, such as Ethernet and DeviceNet. The ONC presents the following advantages:

– it supports QNX 4.25, a real-time, multi-task operating system, capable of executing programs developed in C++;

– CPU: 133 MHz;

– it includes DeviceNet connectivity as standard, and therefore establishes a distributed connection between all the deported elements (sensors, pistons, etc.);

– it has no hard disk, and can be connected to any DeviceNet component without a single line of coding. The system will have the reliability of the DeviceNet network;

– it is able to deport all sorts of inputs/outputs (On/Off, analog, PWM);

– it includes *FinsGateway QNX* – a software platform for communication. Devices with FinsGateway can communicate transparently. This means we can easily add other modules by way of networking technologies: Ethernet, series, DeviceNet, Controller Link and SYSMAC.

The ONC offers all the necessary characteristics for the system in question. It is able to compute all the control algorithms and manage all communication between the equipment on the network.

All the modules seen above are interconnected by a CAN bus, so they can communicate with one another. They communicate and coordinate their actions by passing messages over a network, which facilitates flexible data sharing. The use of this network enables us to easily add new modules into the network.

DeviceNet uses the CAN communication protocol [ODA 04]. It is essentially used in onboard and automated systems. It is an open, multi-bit network which combines commands and data exchanges on the same level of online control. This network is fast and robust, particularly for non-voluminous data exchange. The method chosen for bus access is the master–slave arrangement. The master is the onboard computer and the slaves are the rest of the modules.

The graphic display unit is an OMRON touchscreen, linked to the ONC via a standard Ethernet port. It is able to display fixed or modifiable information. It is programmable, meaning that it can load and execute a simple application. It is capable, firstly, of reading the memory of the ONC for the data it needs for its program and, also, of writing to the memory with the data entered by the driver on the touchscreen.

6.3.8.3. *Testing array*

For obvious reasons of cost and space, the system proposed cannot be developed on a full-scale machine. However, it is crucial to validate the algorithms and the duration of the communications between the modules in conditions similar to real-life situations in order to be able to install the system on the real machine.

A testing array was created for the purpose of quick prototyping of the proposed system, to carry out studies on the implementation of the algorithms and perform tests in

order to check that the proposed system is appropriate for the application (see Figure 6.40). These tests will determine the optimum capacities of the proposed system.

In order to simulate the movement of the arm of the excavator-loader and the action of the hydraulic pistons, the testing array includes an induction motor driven by a variator which enables us to vary the position of the angular sensor.

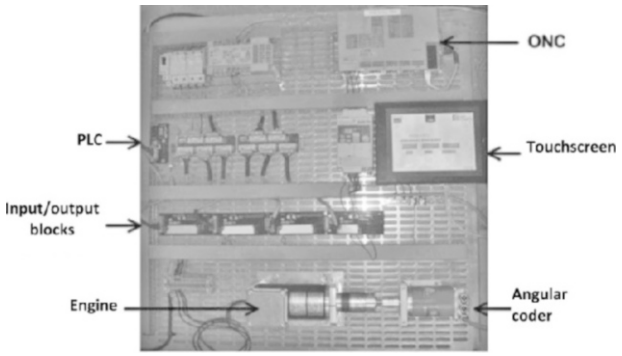


Figure 6.40. *Testing array*

6.3.8.4. *Communication calculation time*

The power of the ONC's CPU is distributed between the programming of all the algorithms and the management of the communications. In order to work out the speed of the system, it is necessary to calculate:

– the two maximum communication times for the input data to be sent to the ONC (T_{com1}) on the one hand, and for the data to be sent from the ONC to the outputs (T_{com2}), on the other;

– the runtime for the algorithms (T_{algo}).

Thus, the total cycle time for the structure is:

$$T_{\text{cycle}} = T_{\text{coml}} + T_{\text{alg}} + T_{\text{com2}} \qquad [6.63]$$

To begin with, the devices in the testing array are configured so as to obtain the two maximum times, both theoretically and experimentally. Numerous tests are carried out on the testing array in order to study the influence of the DeviceNet slaves. These tests show that the two times depend on the type of slaves connected. Experimentally, the two maximum times are:

$$T_{com} = T_{com1} + T_{com2} = (14 \pm 0,5)\, ms$$

Because the experimental values are similar to those yielded by the theory, the maximum communication time of the onboard system, installed on the MECALAC machine where other modules are connected to the DeviceNet network, can be predicted:

$$T_{com} = T_{com1} + T_{com2} = (18 \pm 0.5)\, ms$$

6.3.9. *Conclusion*

The work presented in this section of the chapter focused on a particular form of driving support for building machinery, and more specifically on the study and realization of the human–machine interface for guiding the movements of the tools on hydraulic excavator-loaders – particularly "MECALAC" excavator-loaders with six degrees of freedom.

Firstly, all the characteristics of the MECALAC loader and its geometric and kinematic models were presented. We then went on to outline a number of the main functionalities of a new device to support the driver, and put forward a computer architecture. The new support device aims to transform the commands given to the tool by the driver into articular velocities. This conversion is done by the inverse kinematic model, using a projected gradient approach.

We also demonstrated two algorithmic procedures which take account, on the one hand, of the problem related to the parts of the arm reaching their uppermost or lowermost limit with non-null velocity, and on the other hand, the problem of the power required by the hydraulic actuators of the parts of the arm surpassing the maximum power.

6.4. Automobiles

6.4.1. *Models of automobiles*

In the existing body of literature, we find a number of different models, of varying degrees of completeness and complexity. In this section, we present those models which are most widely used, and show how to validate them using the simulation software "CarSim". To begin with, the bicycle model with two degrees of freedom, corresponding to the lateral and yaw movements, is presented. Then, a model with four degrees of freedom, which takes account of the roll movement, is advanced. We also present a model with 11 degrees of freedom, which is able to describe the lateral and longitudinal dynamics, the dynamics of yaw, of roll, of the wheels and of active braking.

6.4.1.1. *Modeling elements*

The movement of the vehicle is defined by a set of translational and rotational movements. We can enumerate six main movements (see Figure 6.41): the translation movements (lateral, longitudinal and vertical movement) and rotational movements (motion of roll around the *x* axis, movement of yaw around the *z* axis and movement of pitch around the *y* axis, felt when braking, for instance).

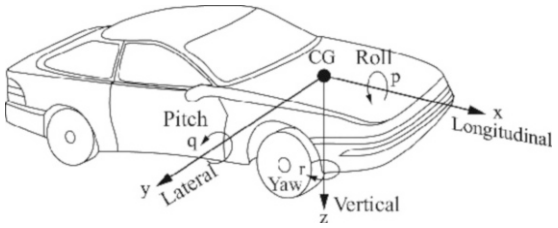


Figure 6.41. *The different movements of a car*

In order to be better able to model the vehicle's behavior, it is important to understand these different movements, main elements and forces of contact between the tires and the road.

6.4.1.1.1. Introduction to the lateral dynamics

The various subsets, such as the steering wheel, steering column, steer wheels and contact between the tires and the road, affect the vehicle's lateral dynamics (see Figure 6.42). By the action of the driver on the steering wheel, these elements enable the vehicle to move laterally, to move during a turn and carry out changes of lane.

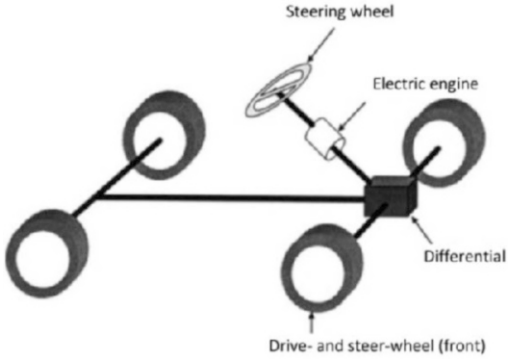


Figure 6.42. *The parts making up the lateral mode*

6.4.1.1.2. Longitudinal slip and tire slip angle

Longitudinal slip of the wheel

The longitudinal slip ratio λ for the wheel is obtained by way of the following expression:

$$\lambda = \frac{w_v(t) - w(t)}{w_v(t)} \text{ if } w_v(t) > w(t) \text{ (braking)}$$

$$\lambda = \frac{w(t) - w_v(t)}{w_v(t)} \text{ if } w(t) < w_v(t) \text{ (acceleration)} \quad [6.64]$$

where $w_v(t) = \frac{v(t)}{R_w}$ is the angular velocity of the vehicle (rad/s), $w(t)$, the angular velocity of the wheel and R_w, the wheel radius. It describes the standardized difference between the angular velocity of the vehicle and the angular velocity of the wheel. Thus, depending on the value assumed by λ, we can distinguish two situations:

– $\lambda = 0$, the wheel is free: no longitudinal force is exerted on the wheel;

– $\lambda = 1$, the wheel is locked ($w(t) = 0$).

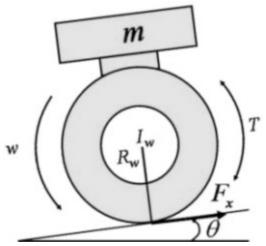


Figure 6.43. *Longitudinal wheel slip*

Wheel slip angle

The slip is the change in the vehicle's trajectory due to the transversal deformation that the tires undergo when they are subjected to the action of a lateral force. In practice, this force may have various origins: side-wind, centrifugal force in a turn, slope on the road or indeed inclination of the wheel (wheel camber).

6.4.1.1.3. Tire forces

The vehicle's dynamics depend, largely, on the dynamics of the tires. One of the most widely used models by tire manufacturers and car makers is that put forward

by Pacejka [PAC 88]. This empirical model is able to express the longitudinal, transversal and forces of self-alignment for different conditions of contact between the tire and the ground (dry, wet, icy ground, etc.) and for different vertical loads. Its general form is given by:

$$y(x) = D \sin[C \arctan\{Bx - E\,(Bx - \arctan(Bx))\}] \qquad [6.65]$$

$$Y(X) = y(x) + Sv$$

$$X = X + Sh$$

Y and X correspond respectively either to the longitudinal force and the longitudinal slip or to the later force and the slip angle. The different parameters and their meanings that are involved in this formula are:

– *C:* shape factor (used to adjust the shape of the curve);

– *D*: peak value (maximum of the adhesion curve);

– *B:* damping factor (original slope);

– *E*: curvature factor;

– *Sh*: used to introduce a horizontal shift in the curvature in relation to the origin;

– *Sv*: for a vertical shift.

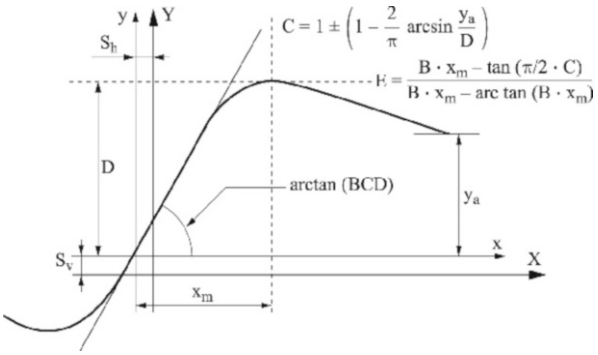


Figure 6.44. *Characteristic curve of formula [6 .65]*

6.4.1.2. *Slip/yaw model*

The slip/yaw model is a model that is widely used to represent the vehicle's lateral movement. The model, also referred to as the bicycle model with two degrees

of freedom corresponding to the lateral and yaw movements, considers the vehicle to be a rigid body with a single wheel for each train (see Figure 6.45). Indeed, the resulting wheel has a steering angle equivalent to the steering angle of the two wheels, and a slip value obtained by projection of the slip of both wheels. Consequently, neither the camber nor the change of direction induced are considered by this model; this assumes that the phenomenon of roll is absent.

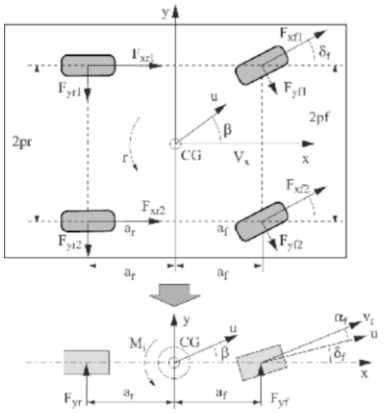


Figure 6.45. *Bicycle model*

MODELING HYPOTHESES.– In order to obtain this model, a certain number of hypotheses are considered:

– constant longitudinal velocity (v_x = *const*);

– no roll or pitching movements;

– small steering angle (δ_f);

– small yaw angle between the vehicle and the ground.

The equations governing the lateral dynamics can be written, taking account of the fundamental principle of dynamics applied at point *G*:

– lateral movement:

$$m(\dot{v}_y + v_x r) = (F_{xf1} + F_{xf2})\sin(\delta_f) + (F_{yf1} + F_{yf2})\cos(\delta_f) + (F_{yr1} + F_{yr2}) \quad [6.66]$$

– yaw movement:

$$I_z\dot{r} = a_f(F_{xf1} + F_{xf2})\sin(\delta_f) + a_f(F_{yf1} + F_{yf2})\cos(\delta_f) - a_r(F_{yr1} + F_{yr2}) + \frac{d_f}{2}(F_{xf2} - F_{xf1})\cos(\delta_f) + \frac{d_r}{2}(F_{xr2} - F_{xr1}). \quad [6.67]$$

The hypotheses considered enable us to carry out the following simplifications:

$$Cos(\delta_f) = 1,\ sin(\delta_f) = \delta_f \quad [6.68]$$

$$F_{xf1} = F_{xf2} = F_{xf,}\ F_{xr1} = F_{xr2} = F_{xr} \quad [6.69]$$

$$F_{yf1} = F_{yf2} = F_{yf}, F_{yr1} = F_{yr2} = F_{yr} \quad [6.70]$$

Equations [6.66] and [6.67] become:

$$mv_x\left(\dot{\beta} + r\right) = 2F_{yf} + 2F_{yr} \quad [6.71]$$

$$I_z\dot{r} = 2a_fF_{yf} - 2a_rF_{yr} \quad [6.72]$$

where:

$$F_{yr} = 2C_r\alpha_r \text{ and } F_{yf} = 2C_f\alpha_f \quad [6.73]$$

$$\alpha_f = \delta_f - \beta - \frac{a_f}{v_x}r \quad \text{(front slip angle)} \quad [6.74]$$

$$\alpha_r = -\beta + \frac{a_r}{v_x}r \quad \text{(rear slip angle)} \quad [6.75]$$

Thus, we can deduce the following linear model:

$$\dot{x}(t) = Ax(t) + B\delta_f(t) \quad [6.76]$$

with:

$$x = \begin{bmatrix}\beta\\ r\end{bmatrix}, A = \begin{bmatrix}a_{11} & a_{12}\\ a_{21} & a_{22}\end{bmatrix}, B = \begin{bmatrix}b_1\\ b_2\end{bmatrix}$$

$$a_{11} = -\frac{2C_f + 2C_r}{mv_x} \quad a_{12} = -1 - \frac{2C_fa_f - 2C_ra_r}{mv_x{}^2},\ a_{21} = -\frac{2C_fa_f - 2C_ra_r}{I_z},$$

$$a_{22} = -\frac{2C_fa_f{}^2 + 2C_ra_r{}^2}{v_x\, I_z}, b_1 = \frac{2C_f}{mVx}, b_2 = \frac{2a_fC_f}{I_z}$$

6.4.1.3. *Model with four degrees of freedom*

In comparison to the bicycle model, this model takes account of the movements of roll (see Figure 6.46). It comprises four degrees of freedom: the lateral velocity v_y, the yaw velocity r, the roll angle $\emptyset$ and the roll velocity p.

MODELING HYPOTHESES.–

– Vertical and pitch dynamics not taken into account.

– Mass of the vehicle, m_v, comprising three masses: the sprung mass m_s, the front unsprung mass m_{uf} and the rear unsprung mass m_{ur}.

– Fixed axis for the vehicle's roll.

– Constant longitudinal velocity.

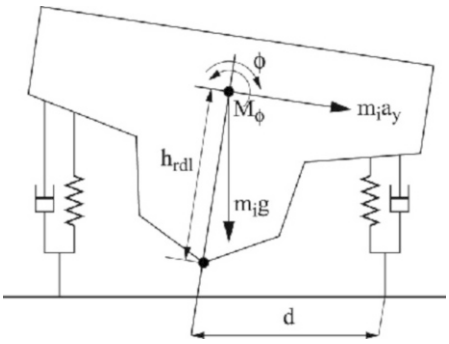


Figure 6.46. *Model for the roll dynamics*

Considering the slip angle (β) at the center of gravity and the yaw angle (Ψ) to be sufficiently small, and the longitudinal velocity (v_x) to be equal to the vehicle's velocity (v), the expression of the lateral acceleration can be written as follows:

$$a_y = \dot{v_y} + \dot{v_x}\Psi + v_x r \qquad [6.77]$$

Also, with small steering angles, the slip angle β can be calculated as follows:

$$\beta = tan^{-1}(\frac{v_y}{v_x}) \approx \frac{v_y}{v_x} \qquad [6.78]$$

The expressions of the slip angles are given by:

$$\alpha_f = \delta_f - \beta - \frac{a_f}{v_x} r + R_f \emptyset \qquad [6.79]$$

$$\alpha_r = \delta_r - \beta + \frac{a_r}{v_x} r + R_r \emptyset \qquad [6.80]$$

where R_f are R_r are constants linked to the effect of roll on the vehicle. δ_f and δ_r are the steering angles for the front and rear wheels.

If we take the above simplifications into account, we obtain the following dynamic equations:

– lateral dynamics:

$$\begin{aligned} m_v\dot{v}_y &= -m_v r v_x - m_s h_s \dot{p} + F_{yf1} + F_{yf2} + F_{yr1} + F_{yr2} = -m_v r v_x \\ &\quad -m_s h_s \dot{p} - (2C_f + 2C_r)\frac{v_y}{v_x} + \left(-2C_f \frac{a_f}{v_x} + 2C_r \frac{a_r}{v_x}\right) r \\ &\quad +2C_f \delta_f + 2C_r \delta_r + (2C_f R_f + 2C_r R_r)\varnothing \end{aligned} \quad [6.81]$$

– yaw dynamics:

$$\begin{aligned} J_{zz}\dot{r} - J_{xz}\dot{p} &= (F_{yf1} + F_{yf2})a_f \\ &\quad -(F_{yr1} + F_{yr2})a_r + (F_{xr2} - F_{xr1})\frac{d_r}{2} + (F_{xf2} - F_{xf1})\frac{d_f}{2} \\ &= (-2C_f a_f + 2C_r a_r)\frac{v_y}{v_x} - (2\frac{c_f a_{f^2}}{v_x} + 2\frac{c_r a_{r^2}}{v_x})r \\ &\quad +2C_f a_f \delta_f - 2C_r a_r \delta_r + (2C_f a_f R_f - 2C_r a_r R_r)\varnothing \end{aligned} \quad [6.82]$$

with $F_{xf1} = F_{xf2} = F_{xr1} = F_{xr2} = 0$;

– roll dynamics:

$$\begin{aligned} &J_{xx}\dot{p} - J_{xz}\dot{r} - C_{\varnothing}\varnothing + m_s g h_s \sin(\varnothing) + m_s a_y h_s \cos(\varnothing) \\ &= -C_{\varnothing} p - (K_{\varnothing} - m_s g h_s)\varnothing + m_s h_s \dot{v}_y + m_s h_s v_x r \\ &\dot{\varnothing} = p. \end{aligned} \quad [6.83]$$

Finally, the model with four degrees of freedom can be expressed as a function of the state vector $[v_y, r, p, \emptyset]^T$.

6.4.1.4. *Reduced nonlinear model of the vehicle*

In order to get as close a reflection of the vehicle's actual behavior as possible, we can consider a model containing the lateral dynamics, longitudinal dynamics, yaw dynamics, roll dynamics, the dynamics of all four wheels and the dynamics of the active steering mechanism (see Figure 6.47). This gives us a reduced nonlinear model [MES 08; YOU 10].

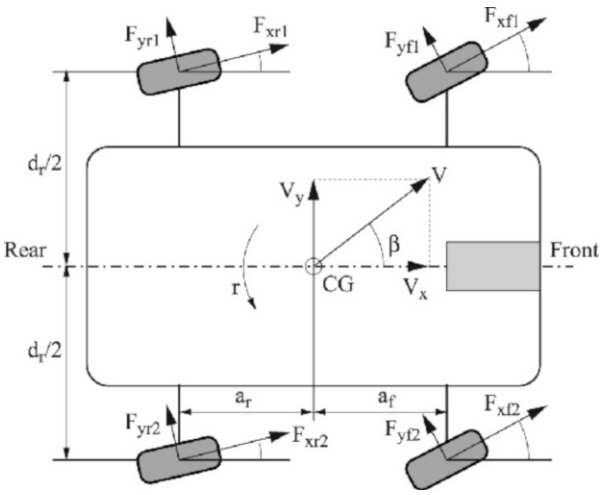


Figure 6.47. *Model of the four-wheel drive vehicle*

If we discount the aerodynamic forces, the vertical motion and the pitching movement of the vehicle, we get:

$$\begin{aligned} m_v(\dot{v}_x - rv_y) &= \cos(\delta_f)(F_{xf1} + F_{xf2}) \\ &+ \cos(\delta_r)(F_{xr1} + F_{xr2}) \\ &- \sin(\delta_f)(F_{yf1} + F_{yf2}) - \sin(\delta_r)(F_{yr1} + F_{yr2}) \end{aligned} \quad [6.84]$$

$$\begin{aligned} m_v(\dot{v}_y - rv_x) + m_s h_s \dot{p} &= \sin(\delta_f)(F_{xf1} + F_{xf2}) \\ &+ \sin(\delta_r)(F_{xr1} + F_{xr2}) + (F_{yf1} + F_{yf2}) \\ &+ \cos(\delta_r)(F_{yr1} + F_{yr2}) \end{aligned} \quad [6.85]$$

$$\begin{aligned} \dot{r} - J_{xz}\dot{p} = {} & \left(F_{yf1} + F_{yf2}\right)a_f \cos\left(\delta_f\right) \\ & - \left(F_{yr1} + F_{yr2}\right)a_r \cos(\delta_r) + \left(F_{xf1} + F_{xf2}\right)a_f \sin\left(\delta_f\right) \\ & - (F_{xr1} + F_{xr2})a_r \sin(\delta_r) + (F_{xr2} \\ & - F_{xr1}) \cos(\delta_r)\frac{d_r}{2} + (F_{xf2} - F_{xf1}) \cos\left(\delta_f\right)\frac{d_f}{2} + \\ & \left(F_{yf2} - F_{yf1}\right) \sin\left(\delta_f\right)\frac{d_f}{2} + (F_{yr2} - F_{yr1}) \sin(\delta_r)\frac{d_r}{2} \end{aligned} \quad [6.86]$$

$$J_{xx}\dot{p} - J_{xz}\dot{r} = -(C_{\emptyset f} + C_{\emptyset r})\, p - (K_{\emptyset f} + K_{\emptyset r})\emptyset + m_s g h_s \sin(\emptyset) + m_s a_y h_s \cos(\emptyset) \quad [6.87]$$

$$J_{r1}\dot{w}_{f1} = C_{m1} - C_{f1} - r_{eff1}F_{xf1},$$

$$J_{r2}\dot{w}_{f2} = C_{m2} - C_{f2} - r_{eff2}F_{xf2},$$

$$J_{r3}\dot{w}_{r1} = C_{m3} - C_{f3} - r_{eff3}F_{xr1},$$

$$J_{r4}\dot{w}_{r2} = C_{m4} - C_{f4} - r_{eff4}F_{xr2}, \quad [6.88]$$

$$\dot{\delta}_f = -\frac{K_{sf}}{C_{sf}}\delta_f + \frac{K_{sf}}{C_{sf}}\delta_{sf} + \delta_c,\ \dot{\delta}_r = -\frac{K_{sr}}{C_{sr}}\delta_r + \frac{K_{sr}}{C_{sr}}\delta_{sr} \quad [6.89]$$

6.4.1.5. *Model with eleven degrees of freedom*

On the basis of the reduced nonlinear model set out above, we can obtain a model of the vehicle with 11 degrees of freedom: the lateral velocity v_y, the yaw velocity r, the yaw angle ψ, the roll velocity p and the roll angle $\emptyset$ represent five degrees of freedom relating to the car's chassis. The angles of direction of the wheels (δ_f and δ_r) and the rotational velocities of the four wheels w_{f1}, w_{f2}, w_{r1} and w_{r2}, bring the number of degrees of freedom up to 11.

$$J_{zz}\dot{r} - J_{xz}\dot{p} = a_1 v_y + a_2 r + a_3\delta_f + a_4\delta_r + a_5\emptyset + a_6(w_4 - w_3) + a_7(w_2 - w_1) \quad [6.90]$$

$$J_{xx}\dot{p} - J_{xz}\dot{r} = -C_{\emptyset}p + a_8\emptyset + a_9\dot{v_y} + a_{10}r + a_{17}\psi \quad [6.91]$$

$$\dot{v_y} = -rv_x + a_{11}\dot{p} + a_{12}v_y + a_{21}r + a_{13}\delta_f + a_{14}\delta_r + a_{15}\emptyset \quad [6.92]$$

$$\dot{\emptyset} = p \quad [6.93]$$

$$J_t\dot{w}_i = -\tau_{bi} - \frac{C_{\sigma f}{r_{eff}}^2}{v_x}w_i + r_{eff}C_{\sigma f},\ i = 1,2,3,4 \quad [6.94]$$

$$\dot{\delta}_f = -\frac{K_{sf}}{C_{sf}}\delta_f + \frac{K_{sf}}{C_{sf}}\delta_{sf} + \delta_c \quad [6.95]$$

$$\dot{\delta}_r = -\frac{K_{sr}}{C_{sr}}\delta_r + \frac{K_{sr}}{C_{sr}}\delta_{sr} \quad [6.96]$$

This gives us the following model:

$$\begin{bmatrix} \dot{v}_y \\ \dot{r} \\ \dot{\psi} \\ \dot{p} \\ \dot{\phi} \\ \dot{\delta}_f \\ \dot{\delta}_r \\ \dot{w}_1 \\ \dot{w}_2 \\ \dot{w}_3 \\ \dot{w}_4 \end{bmatrix} = \begin{bmatrix} \dfrac{1}{a_{11} - \dfrac{a_{11}^2}{a_{16}} J_{zz} a_9} \left(A_3 + \dfrac{a_{11}}{a_{18}} \left(J_{xz} A_1 + J_{zz} A_2 \right) \right) - \dfrac{A_3}{a_{11}} \\ -A_2 - \dfrac{a_9 - \dfrac{J_{xx}}{a_{11}}}{1 - \dfrac{a_{11}}{a_{16}} J_{zz} a_9} \left(A_3 + \dfrac{a_{11}}{a_{18}} \left(J_{xz} A_1 + J_{zz} A_2 \right) \right) + J_{xx} \dfrac{A_3}{a_{11}} \\ A_{10} \\ \dfrac{1}{a_{11} - \dfrac{a_{11}^2}{a_{16}} J_{zz} a_9} \left(A_3 + \dfrac{a_{11}}{a_{18}} \left(J_{xz} A_1 + J_{zz} A_2 \right) \right) - \dfrac{A_3}{a_{11}} \\ A_{11} \\ A_{12} \\ A_{13} \\ A_{14} \\ A_{15} \\ A_{16} \\ A_{17} \end{bmatrix} \begin{bmatrix} v_y \\ r \\ \psi \\ p \\ \phi \\ \delta_f \\ \delta_r \\ w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix} +$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ \dfrac{-k_{sf}}{c_{sf}} & 0 & 0 & 0 & 0 & 0 \\ 0 & \dfrac{-k_{sr}}{c_{sr}} & 0 & 0 & 0 & 0 \\ 0 & 0 & \dfrac{-1}{J_t} & 0 & 0 & 0 \\ 0 & 0 & 0 & \dfrac{-1}{J_t} & 0 & 0 \\ 0 & 0 & 0 & 0 & \dfrac{-1}{J_t} & 0 \\ 0 & 0 & 0 & 0 & 0 & \dfrac{-1}{J_t} \end{bmatrix} \begin{bmatrix} \delta_{sf} \\ \delta_{sr} \\ \tau_1 \\ \tau_2 \\ \tau_3 \\ \tau_4 \end{bmatrix} \quad [6.97]$$

where:

$$A_{10} = [\,0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\,]$$

$$A_{11} = [\,0\ 0\ 0\ 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\,]$$

$$A_{12} = [0\ 0\ 0\ 0\ 0\ -\frac{K_{sf}}{C_{sf}}\ 0\ 0\ 0\ 0\ 0]$$

$$A_{13} = \left[\,0\ 0\ 0\ 0\ 0\ 0\ -\frac{K_{sr}}{C_{sr}}\ 0\ 0\ 0\ 0\,\right]$$

$$A_{14} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ -\frac{{r_{eff}}^2 C_{\sigma f}}{v_x J_t}\ 0\ 0\ 0\,]$$

$$A_{15} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -\frac{{r_{eff}}^2 C_{\sigma f}}{v_x J_t}\ 0\ 0\,]$$

$$A_{16} = \left[\,0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -\frac{{r_{eff}}^2 C_{\sigma f}}{v_x J_t}\ 0\,\right]$$

$$A_{17} = [0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ -\frac{{r_{eff}}^2 C_{\sigma f}}{v_x J_t}\,]$$

Taking certain hypotheses into account, in this section, we have presented a number of different models of vehicles: a model with two degrees of freedom, a model with four degrees of freedom, a reduced nonlinear model and a model with 11 degrees of freedom. In the next section, these models are validated by the software package CarSim.

6.4.2. *Validation of the vehicle models*

We shall begin by presenting the simulation tool "CarSim", which will be used to validate the different models cited above. Then, we present the results of the simulations. Finally, we compare the model with two degrees of freedom to an experimental test carried out on a track with a Citroën C4 equipped with an inertial measurement unit.

6.4.2.1. *Introduction to the tool CarSim 7.1*

CarSim is a professional software suite developed by the Mechanical Simulation Corporation, used to simulate the dynamics of cars. Thanks to this tool, we can virtually recreate different driving situations and test the vehicle's behavior and its response to different maneuvers (changing lane, slalom, acceleration, slope, etc.).

CarSim comprises five main parts (integrated software modules) as shown in Figure 6.48.

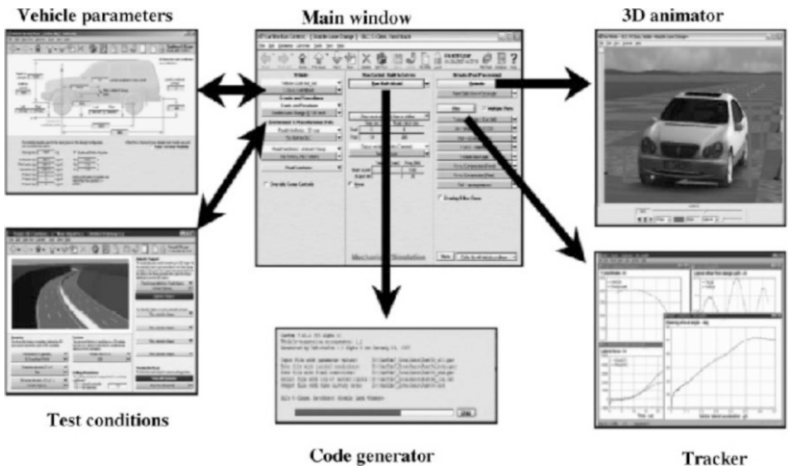


Figure 6.48. *The five main components of CarSim*

– *Vehicle parameters*: this block enables us to define various physical parameters of the vehicle (dimensions, engine, tires, bodywork, etc.) and mathematical models representing the forces of contact between the tires and the ground and suspension exerted on the vehicle.

– *Test conditions*: in this block, the user can create his own customized circuit (choice of maneuvers, road conditions, aerodynamic forces, etc.).

– *Code generator*: this block generates a block diagram that can be used by a number of different mathematical computation tools (Matlab®/Simulink, LabView, dSpace, etc.).

– *Animator*: once the program has been compiled, this block enables us to watch the maneuver on a 3D video representation.

– *Tracker*: for each test, this block can record and track up to 700 variables.

6.4.2.2. *Choice of vehicle in CarSim*

In order to be able to validate the proposed models, we need to define a vehicle model in CarSim. For this purpose, we proceed as follows:

– choice of the dimensions of the vehicle: the simulation vehicle has to have the same parameters as the computed model. Thus, in the "Sprung Mass" window (see Figure 6.49), we can choose the dimensions of the vehicle: the sprung mass m,

the distances between the vehicle's center of gravity and, respectively, the front train a_f and the rear train a_r, the moments of inertia I_{xx}, I_{yy} and I_{zz} and the distance h_{roll} between the center of gravity and the roll axis;

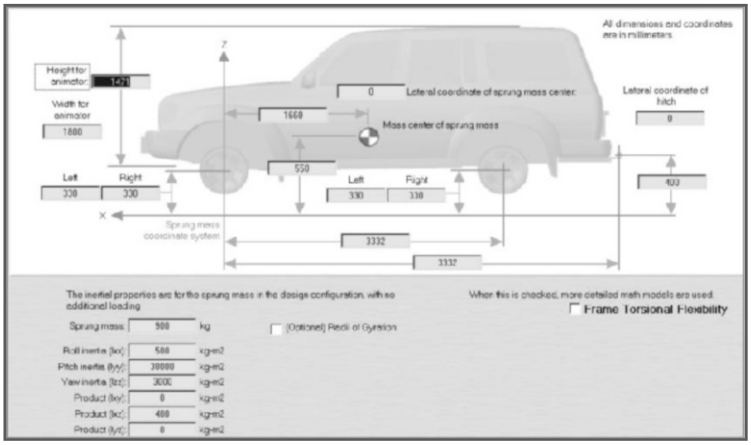


Figure 6.49. *Choice of dimensions of the vehicle*

– choice of tires: the tire is a highly important component in the behavior of a road vehicle. The modeling error and the precision of movement of the system depend, to a large extent, on this choice. In the "tire" window (see Figure 6.50), we can choose the parameters for the tires: tire radius, maximum force applied to the tires and the models of the forces of contact between the tire and the ground;

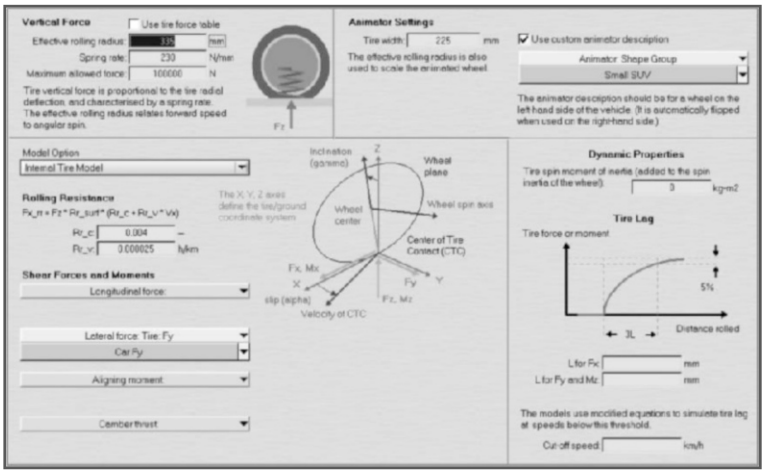


Figure 6.50. *Choice of parameters for the tires*

– definition of the forces of contact between the tires and the ground: CarSim offers a number of possibilities for defining the forces of contact between the tires and the ground. The model which is most widely used is Pacejka's "magic formula". Thus, we can define our forces by using the parameters that characterize this model (see Figure 6.51) or directly generating the characteristic curve by introducing the coordinates into a table (see Figure 6.52);

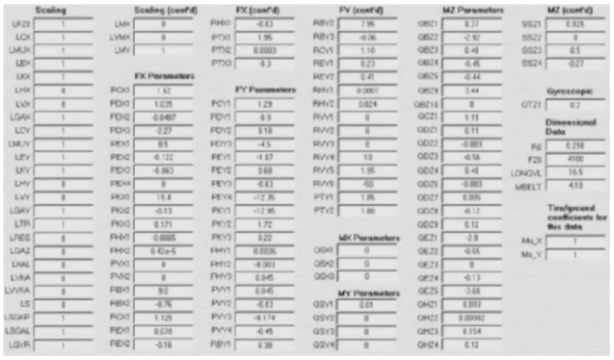


Figure 6.51. *Parameters for Pacejka's Magic Formula 5.2 (MF5.2)*

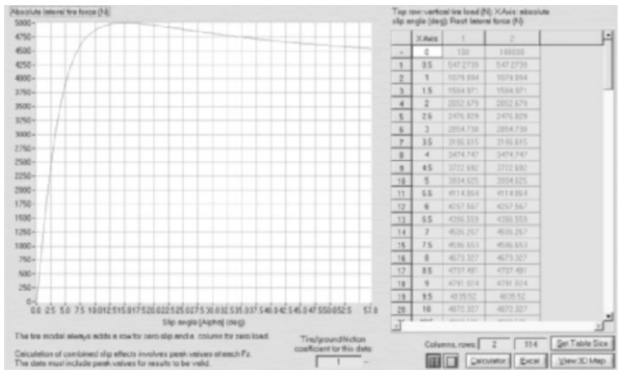


Figure 6.52. *Characteristic curve of the lateral force in CarSim*

– choice of the type of suspension: we use the word "suspension" to denote all the mechanical elements which connect the wheels to the body (the main structure of a vehicle). With CarSim, we can choose several different types of suspension. In our case, we use a simple independent suspension (see Figure 6.53) in which we define the unsprung mass, the wheel radius, the distance between the front and rear wheels, the vertical stiffness coefficient of the spring and the vertical stiffness coefficient of the shock absorber.

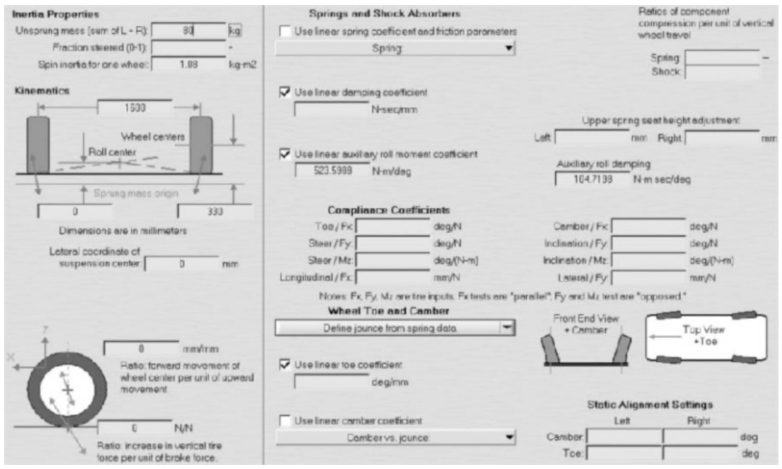


Figure 6.53. *Window for defining the parameters for the independent suspension*

6.4.2.3. *Validation of the linear model*

The validation method used consists of evaluating the vehicle's behavior with CarSim when given various maneuvers to perform and comparing the results with the theoretical model computed previously.

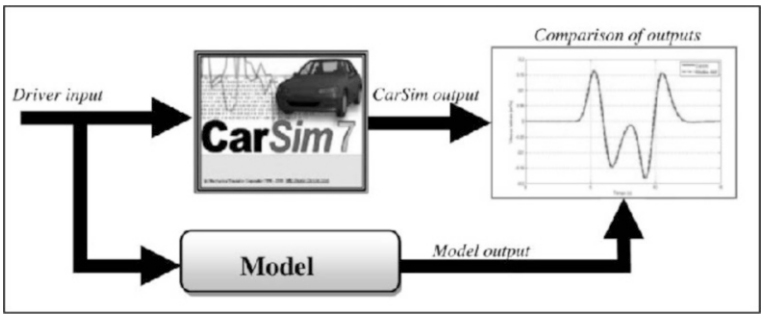


Figure 6.54. *Validation with CarSim*

In order to validate the models with 2 d.o.f, 4 d.o.f and 11 d.o.f, we can use a slalom test, a chicane test or a double lane-change test.

6.4.2.3.1. Double lane-change test

This test simulates an obstacle avoidance scenario by a double lane-change (see Figure 6.55). The vehicle has to drive between cones which form the desired

trajectory. Figures 6.56 and 6.57 respectively show the path taken by the vehicle during the maneuver and the shape of the curve for the input (steering angle).

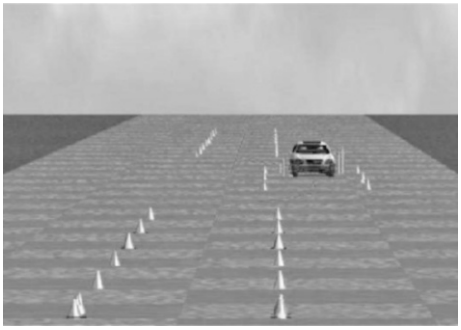


Figure 6.55. *Double lane-change test in CarSim*

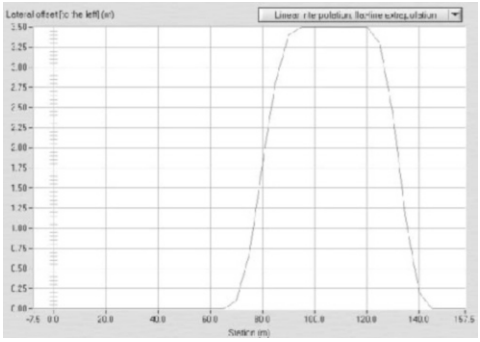


Figure 6.56. *Trajectory of the vehicle (chicane)*

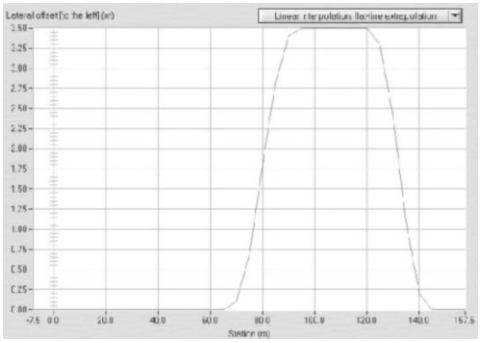


Figure 6.57. *Steering angle*

Simulation results: model with 4 d.o.f

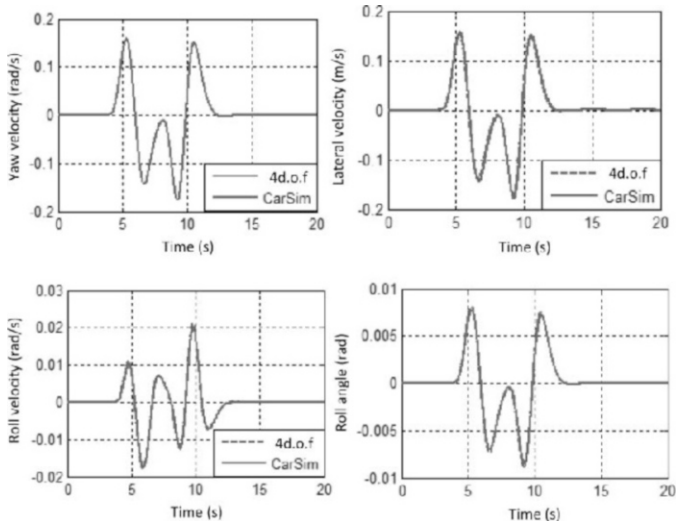


Figure 6.58. *Comparison of the outputs of the model with 4 d.o.f (chicane)*

We can see that the model's outputs do indeed correspond closely to the outputs of the model generated by CarSim (see Figure 6.58).

6.4.2.3.2. Slalom test

This test involves following a sinusoidal trajectory (see Figure 6.59). Figures 6.60 and 6.61 respectively show the path taken by the vehicle during this maneuver and the shape of the curve for the input (steering angle).

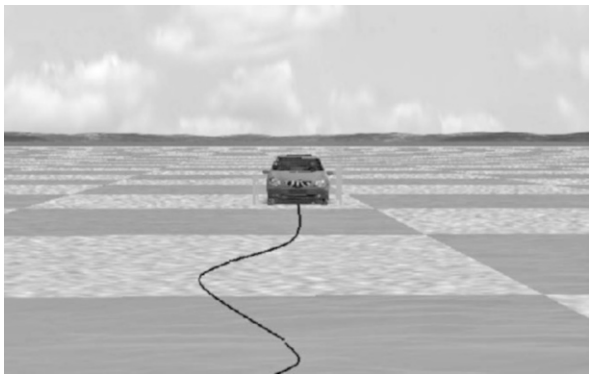


Figure 6.59. *Slalom test in CarSim*

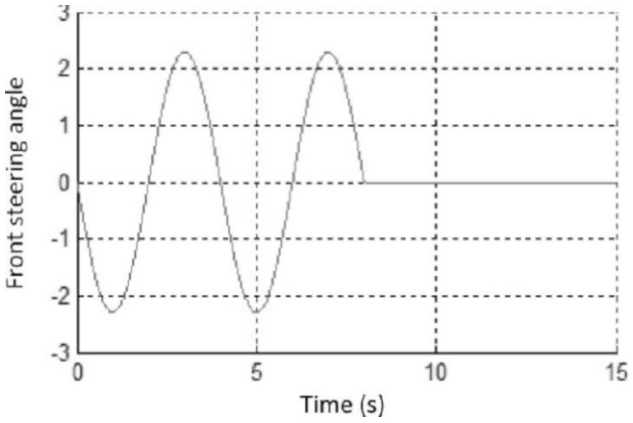


Figure 6.60. *Steering angle (slalom)*

Simulation results: model with 4 d.o.f

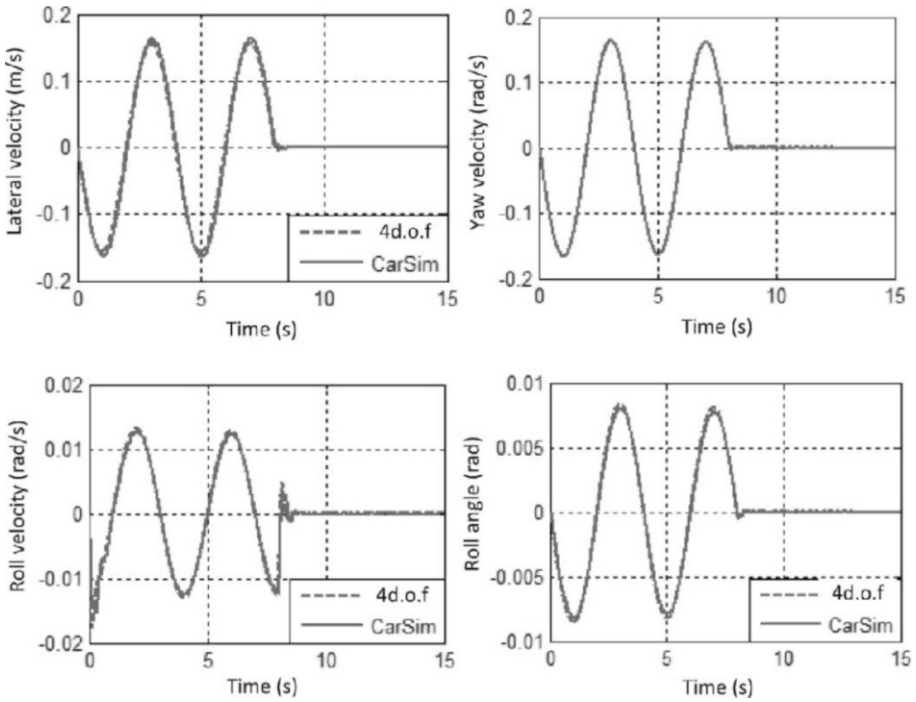


Figure 6.61. *Comparison of the outputs of the model with 4 d.o.f (slalom)*

Simulation results: model with 11 d.o.f.

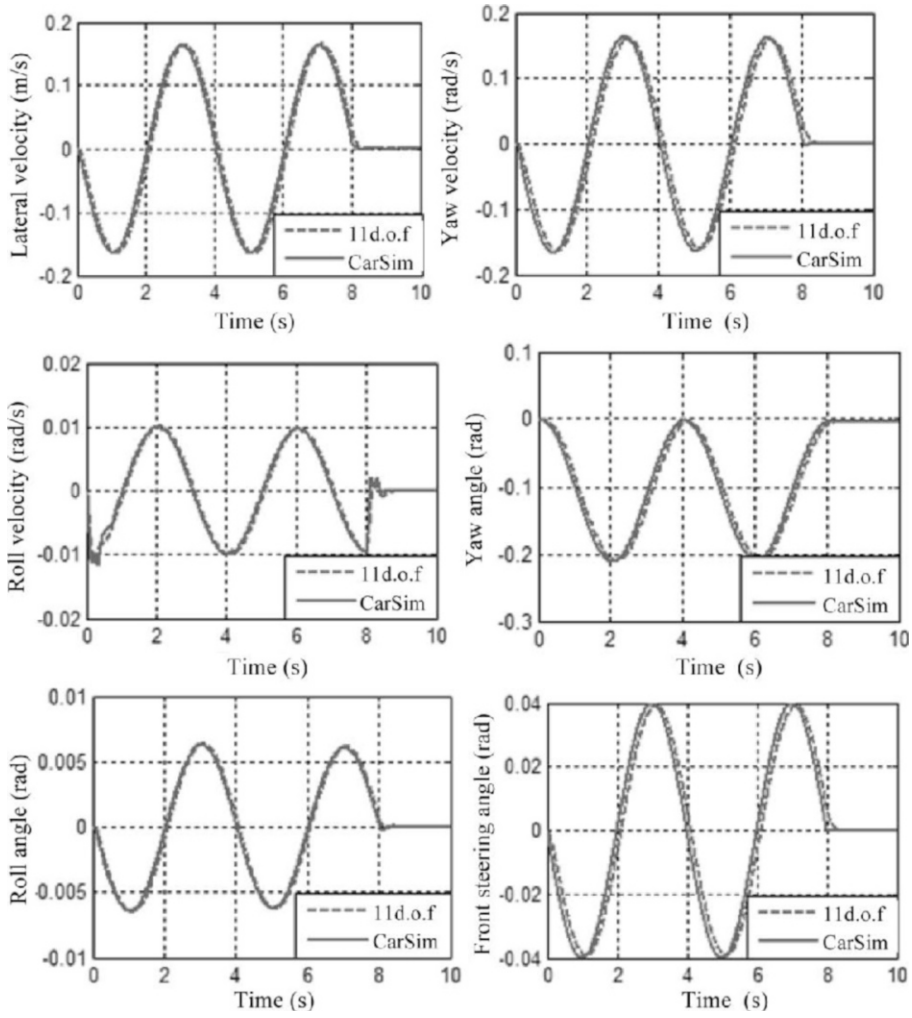


Figure 6.62. *Comparison of the outputs of the model with 11 d.o.f (slalom)*

Figures 6.58, 6.61 and 6.62 demonstrate the quality of the modeling carried out for the three models presented.

6.4.2.4. *Validation of the reduced nonlinear model of the vehicle*

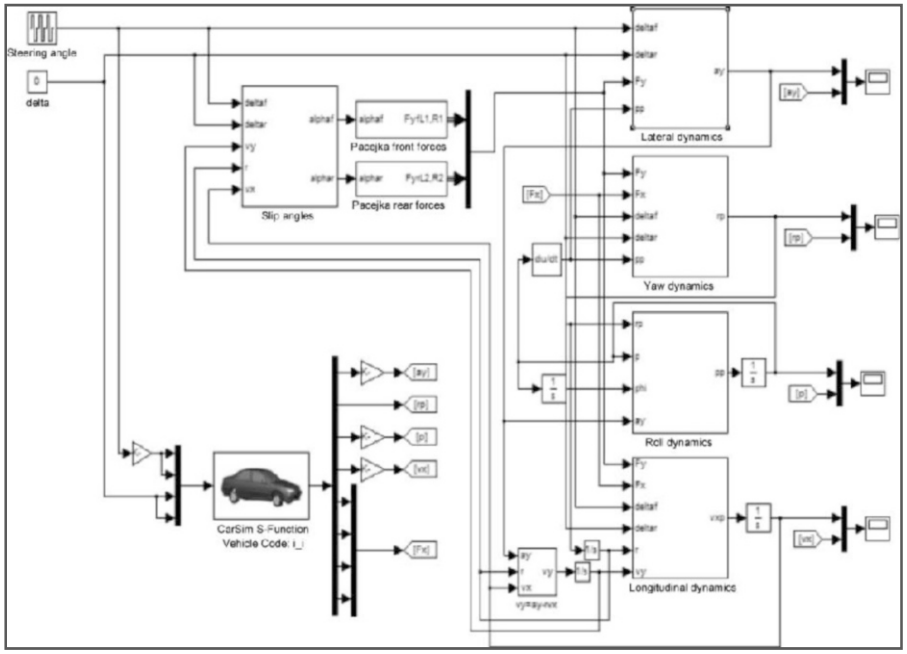


Figure 6.63. *Comparison of the reduced nonlinear model and CarSim in Simulink*

We have simulated the different dynamics of the vehicle (lateral dynamics, roll dynamics, yaw dynamics and longitudinal dynamics) using the equations from the reduced nonlinear model of the vehicle presented above (see Figure 6.63). The lateral forces are defined using Pacejka's model. Figure 6.64 shows the results of the comparison of the output from the reduced nonlinear model against CarSim.

6.4.2.4.1. *Simulation results*

6.4.2.4.2. Experimental validation

The test was carried out on a Citroën C4 vehicle equipped with an inertia measurement unit (RT5200). This instrument measures all the dynamics, without simplified hypotheses, for any type of vehicle and in any orientations (see Figure 6.65). It is an inertial navigation system with six axes: three accelerations and three angular velocities. The RT5200 can easily be installed, along with a GPS antenna which helps the unit gather even more data and improves its precision. The device is installed inside the car and the GPS antenna has to be installed directly

above the inertia measurement unit (see Figure 6.66). The unit measures the following data:

– the longitudinal velocity v_x;

– the lateral velocity v_y;

– the yaw velocity r;

– the slip angle α;

– the roll angle Ø;

– the roll velocity p.

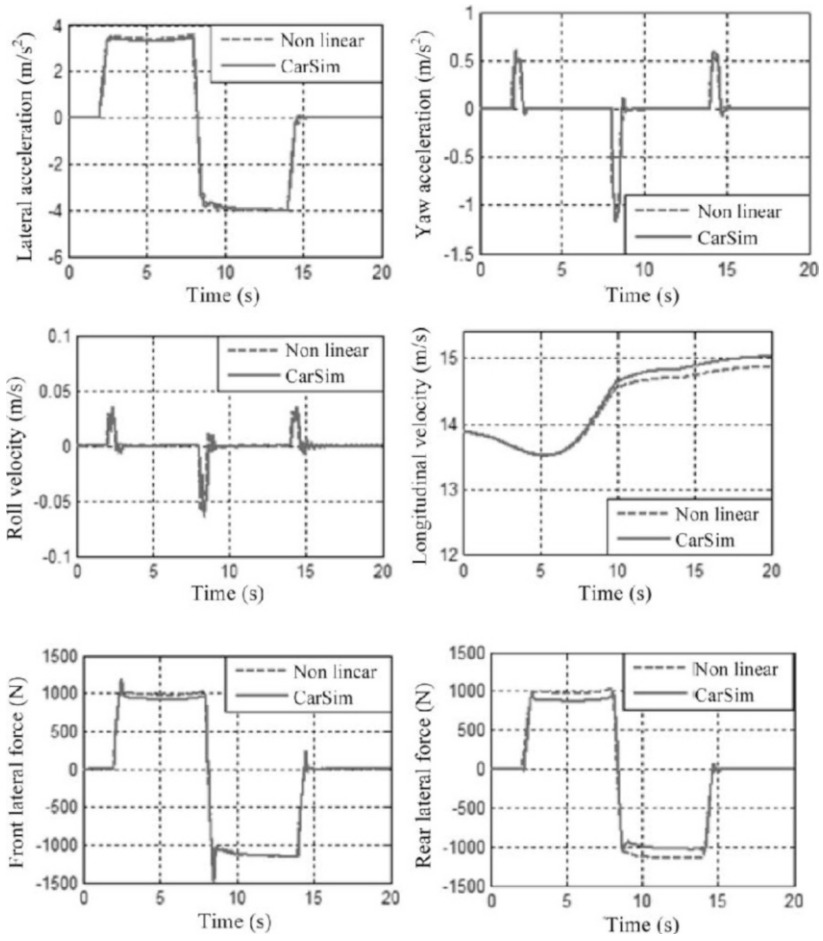


Figure 6.64. *Comparison with the output from the reduced nonlinear model of the vehicle*

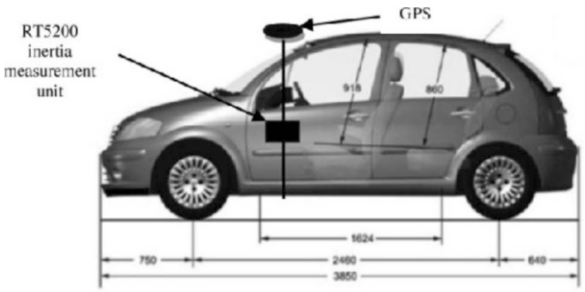


Figure 6.65. *Installation of the inertia measurement unit*

Figure 6.66. *RT5200 inertia measurement unit and GPS antenna*

Data acquisition is performed by a diagnostics instrument connected to a laptop computer running a diagnostic software suite. The experimental validation was performed on a test track 100 m long and 5 m wide.

Figure 6.67 shows the results of the comparison between the output from the model with 2 d.o.f and the data collected by the RT2500 during a test similar to a slalom test.

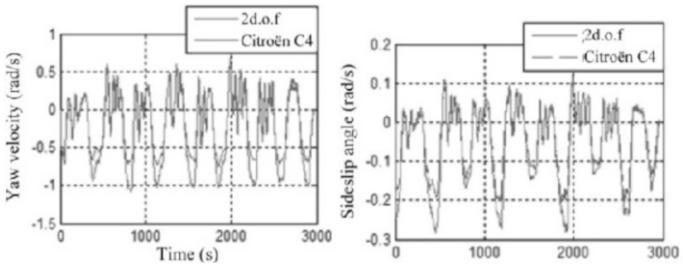


Figure 6.67. *Comparison between the output from the model with 2 d.o.f and the data measured in the vehicle itself*

We can see that the output from the 2 d.o.f model closely follows that measured by the inertia measurement unit.

The above sections have been devoted to the modeling of cars and the validation of the models, first with the program CarSim and then by experimentation. The next section is given over to the control of the vehicle's dynamics, using different control laws and different models.

6.4.3. *Robust control of a vehicle's dynamics*

A vehicle is a complex system, subject to many uncertainties stemming from many different factors (velocity, tire grip, etc.) and subject to many disturbances (such as side winds, variation in mass, etc.) which may negatively impact on or destabilize its behavior. In that sense, robust control laws are absolutely crucial [CHA 06, CHA 12, TAN 00]. In the past few decades, a nonlinear approach referred to as the multi-model approach has been developed. Indeed, to begin with, we are going to present the nonlinear method for approximating the vehicle's dynamics by a multi-model comprising two local models. This multi-model is then validated using CarSim. Then, control laws are applied, to regulate the dynamics of the vehicle. The first such law is based on reconstructed state feedback. The second is an H_∞ law that minimizes the effect of outside disturbances. The third control law is an observer-based robust control which takes account of the parametric uncertainties.

6.4.3.1. *Multi-model representation of the lateral dynamics of the vehicle*

The modeling of the dynamics of the vehicle and formulation of the different control laws are based on multi-model representation, also referred to as Takagi-Sugeno (TS) fuzzy representation [TAK 85], as presented in Chapter 3. This approach has been widely used in recent decades for many applications [CHA 02; GUE 06; TAN 01].

The nonlinearities of the model of the vehicle stem primarily from the forces of contact between the tire and the ground. The idea is to approximate the two lateral forces (front and rear) with a multi-model using two local models. In general, the lateral forces of contact between the tires and the ground are considered to be proportional to the slip angle. This approximation is valid only with a small slip angle (see Figure 6.68).

The interval of evolution of the slip angle is divided into two regions: a region which corresponds to a small slip angle (slip angle $< \alpha_{opt}$) and another which corresponds to a large slip angle.

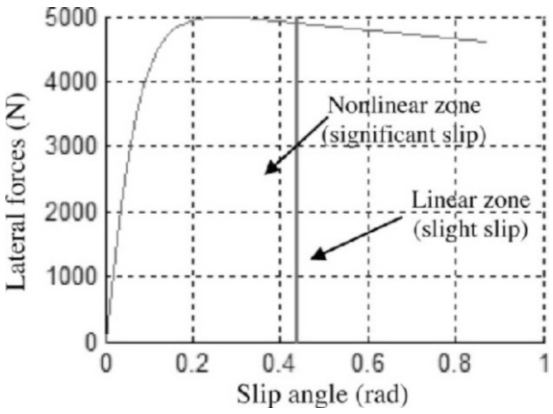


Figure 6.68. *Variation of the lateral forces as a function of the slip angle*

Figure 6.69 shows the dependency of the stiffness functions $\frac{\partial F_{f,r}}{\partial \alpha_{f,r}}$ on the road grip μ (a parameter which characterizes the state of the road surface). The approximation of the lateral forces is obtained by:

$$F_{yf} = h_1(|\alpha_f|)C_{f1}\alpha_f + h_2(|\alpha_f|)C_{f2}\alpha_f \quad [6.98]$$

$$F_{yr} = h_1(|\alpha_r|)C_{r1}\alpha_r + h_2(|\alpha_r|)C_{r2}\alpha_r \quad [6.99]$$

with $C_{fi} = \frac{\partial F_{yfi}}{\partial \alpha_{fi}}$ and $C_{ri} = \frac{\partial F_{yri}}{\partial \alpha_{ri}}$ respectively representing the stiffness coefficients of the front and rear wheels. The functions $h_1(|\alpha(t)_f|)$ and $h_2(|\alpha(t)_f|)$ represent the activation functions with $h_1(.) + h_2(.) = 1$ and $h_1(.) > 0$, $h_2(.) > 0$. Their expressions are given by:

$$h_i(|\alpha(t)_f|) = \frac{w_i(|\alpha(t)_f|)}{\sum_{i=1}^{2} w_i(|\alpha(t)_f|)}, i = 1,2 \quad [6.100]$$

where

$$w_i(|\alpha_f|) = \frac{1}{(1 + \left|\frac{|\alpha_f| - c_i}{a_i}\right|)^{2b_i}} \quad [6.101]$$

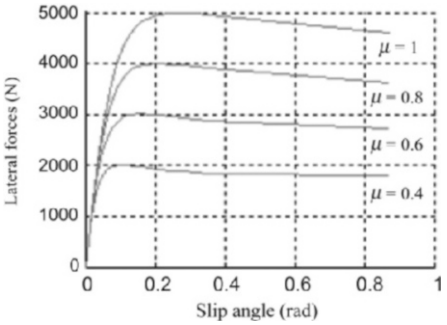


Figure 6.69. *Variation of the lateral forces as a function of the slip angle and of the grip of the tires*

In order to estimate the parameters C_{fi} and C_{ri} and the activation functions $h_i(|\alpha(t)_f|)$ for a given friction coefficient, the Levenberg–Marquardt optimization algorithm (LMA) is used in combination with the least-squares method. With a friction coefficient of $\mu = 0.2$, we get:

$$C_{f1} = 359980,\ C_{f2} = -88030,\ C_{r1} = 359980,\ C_{r2} = -88030$$

$$a_1 = 0.0251 \quad b_1 = 2.0140 \quad c_1 = \text{-}0.0202$$

$$a_2 = 0.0347 \quad b_2 = 1.9929 \quad c_2 = \text{-}0.0209$$

As Figure 6.70 shows, the multi-model representation of the tire forces in equations [6.98] and [6.99] is a good approximation of the tire forces obtained by CarSim.

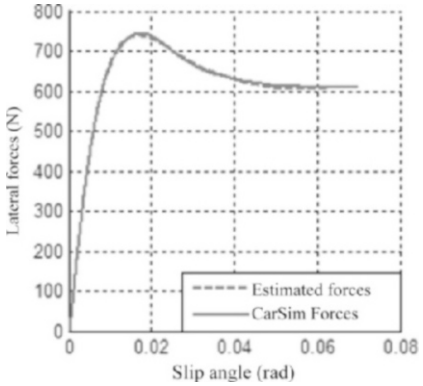


Figure 6.70. *Comparison between the forces obtained by CarSim and the estimated forces*

6.4.3.2. *Validation of the multi-model*

By replacing these forces with their expressions [6.98]–[6.99] in the nonlinear model [6.71]–[6.72] of the vehicle's lateral dynamics, and taking account of [6.74]–[6.75], we obtain the following multi-model:

$$\dot{x}(t) = \sum_{i=1}^{2} h_i\left(\left|\alpha(t)_f\right|\right)\left(A_i x(t) + B_{fi}\delta_f(t) + B_{ri}\delta_r(t)\right)$$

$$y(t) = Cx(t) \qquad [6.102]$$

with:

$$x = \begin{pmatrix}\beta \\ r\end{pmatrix}, \quad A_i = \begin{pmatrix} -\dfrac{2C_{fi} + 2C_{ri}}{mv_x} & -1 - \dfrac{2C_{fi}a_{fi} - 2C_{ri}a_{ri}}{mv_x{}^2} \\ -\dfrac{2C_{fi}a_f - 2C_{ri}a_r}{I_z} & -\dfrac{2C_{fi}a_f{}^2 + 2C_{ri}a_r{}^2}{v_x I_z} \end{pmatrix}$$

$$B_{fi} = \begin{pmatrix} \dfrac{2C_{fi}}{mVx} \\ \dfrac{2a_f C_{fi}}{I_z} \end{pmatrix}, B_{ri} = \begin{pmatrix} \dfrac{2C_{ri}}{mVx} \\ -\dfrac{2a_r C_{ri}}{I_z} \end{pmatrix}, C = \begin{pmatrix} 0 & 1 \end{pmatrix}$$

In order to validate the multi-model [6.102] with CarSim, we apply a chicane-type input $\delta_f(t)$.

Figure 6.71 shows how closely the multi-model representation approximates the nonlinear model. Simulations with significant slip angles demonstrate the quality of the multi-model approximation proposed here in comparison to the linear model.

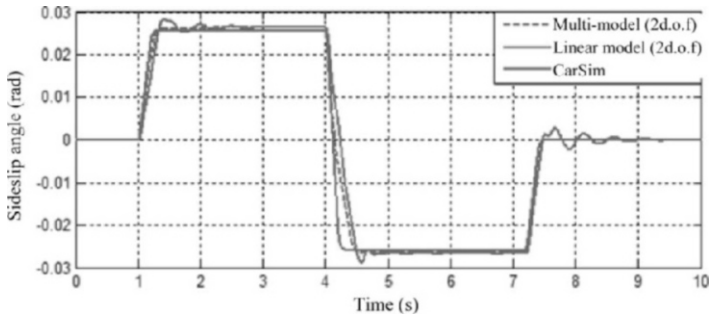


Figure 6.71. *Comparison of the sideslip angle for the three models: linear, multi-model and CarSim*

6.4.3.3. *Reconstructed state feedback control*

Consider a nonlinear dynamic system represented by a multi-model, comprising M local models, described by the following equations:

$$\dot{x}(t) = \sum_{i=1}^{M} \mu_i\big(z(t)\big)(A_i x(t) + B_i u(t)) \qquad [6.103]$$

$$y(t) = Cx(t)$$

where $x(t) \in \Re^n$ is the state vector, $u(t) \in \Re^m$ is the input vector and $y(t) \in \Re^p$ represents the output vector. The matrices A_i, B_i, and C_i are of appropriate dimensions. $\mu_i\big(z(t)\big)$ are the activation functions of the local models and $z(t)$ represents the decision variables vector, which is dependent upon measurable variables.

The expression of the multi-observer is as follows:

$$\dot{\hat{x}}(t) = \sum_{i=1}^{M} \mu_i\big(z(t)\big)(A_i \hat{x}(t) + B_i u(t) + L_i(y(t) - \hat{y}(t)) \qquad [6.104]$$

$$\hat{y}(t) = \sum_{i=1}^{M} \mu_i\big(z(t)\big) C_i \hat{x}(t)$$

where $\hat{x}(t)$ represents the state vector as estimated by the multi-observer, $\hat{y}(t)$ is the estimated output vector and $L_i \in \Re^{n \times p}$ are the gains of the observer to be calculated.

The control law used is of the form:

$$u = -\sum_{i=1}^{M} \mu_i\big(z(t)\big) K_i \hat{x}(t) \qquad [6.105]$$

6.4.3.3.1. Application of the model with 2 d.o.f of the vehicle

Consider the following multi-model of the vehicle:

$$\dot{x}(t) = \sum_{i=1}^{2} h_i\big(|\alpha(t)_f|\big)\left(A_i x(t) + B_{fi}\delta_f(t) + BM_z(t)\right)$$

$$y(t) = Cx(t) \qquad [6.106]$$

where:

$$x = \begin{pmatrix}\beta \\ r\end{pmatrix}, \qquad A_i \begin{pmatrix} -\dfrac{2C_{fi} + 2C_{ri}}{mv_x} & -1 - \dfrac{2C_{fi}a_{fi} - 2C_{ri}a_{ri}}{mv_x{}^2} \\ -\dfrac{2C_{fi}a_f - 2C_{ri}a_r}{I_z} & -\dfrac{2C_{fi}a_f{}^2 + 2C_{ri}a_r{}^2}{v_x I_z} \end{pmatrix}$$

$$B_{fi} = \begin{pmatrix} \frac{2C_{fi}}{mVx} \\ \frac{2a_f C_{fi}}{I_z} \end{pmatrix}, B = \begin{pmatrix} 0 \\ \frac{1}{I_z} \end{pmatrix}, C = (\ 0\ \ 1\ ) \quad [6.107]$$

The expression of the multi-observer is as follows:

$$\dot{\hat{x}}(t) = \sum_{i=1}^{2} h_i\left(\left|\alpha(t)_f\right|\right)(A_i\hat{x}(t) + B_{fi}\delta_f(t) + BM_z(t) + L_i(y(t) - \hat{y}(t))$$

$$\hat{y}(t) = C\,\hat{x}(t) \quad [6.108]$$

The expression of the control law in question is as follows:

$$u = -\sum_{i=1}^{2} h_i\left(\left|\alpha(t)_f\right|\right)K_i\hat{x}(t) \quad [6.109]$$

The control input is the moment M_z around the vehicle's center of gravity. The steering angle of the rear wheels is considered to be zero ($\delta_r(t) = 0$) and we suppose that the measurement of the sideslip angle is not available.

The augmented system is written as:

$$\begin{pmatrix} \dot{x}(t) \\ \dot{\tilde{x}}(t) \end{pmatrix} = \sum_{i=1}^{2} \sum_{j=1}^{2} \mu_i(z(t))\mu_j(z(t)) \begin{pmatrix} A_i - B_iK_j & B_iK_j \\ 0 & A_i - L_iC_j \end{pmatrix} \begin{pmatrix} x(t) \\ \tilde{x}(t) \end{pmatrix} \quad [6.110]$$

with $\dot{\tilde{x}}(t) = \dot{x}(t) - \dot{\hat{x}}(t)$ being the dynamics of the state estimation error. The stability of the model [6.110] is ensured by the following result.

THEOREM 6.1.– If there are symmetrical and positive definite matrices P_1 and P_2, and matrices K_i and L_i which satisfy $\forall(i, j) \in I_M{}^2, i < j$:

$$L(G_{ii}, P_1) < 0 \quad [6.111]$$

$$L(G_{ij}, P_1) \leq 0$$

$$L(\pounds_{ii}, P_2) < 0 \quad [6.112]$$

$$L(\pounds_{ij}, P_2) \leq 0$$

where:

$$G_{ij} = A_i - B_iK_j, \quad \pounds_{ii} = A_i - L_iC_j$$

$$L(G_{ij}, P_1) = \left(\frac{G_{ij} + G_{ji}}{2}\right)^T P_1 + P_1\left(\frac{G_{ij} + G_{ji}}{2}\right), L(\pounds_{ij}, P_2) = \left(\frac{\pounds_{ij} + \pounds_{ji}}{2}\right)^T P_2 + P_2\left(\frac{\pounds_{ij} + \pounds_{ji}}{2}\right)$$

then the augmented system [6.110] is asymptotically stable via the controller [6.109].

A version of this result in linear form is given in Chapter 3 – *Control tools*.

6.4.3.3.2. Simulation results

Solving conditions [6.111] and [6.112] using the LMI toolbox in Matlab® gives the following gains:

$$K_1 = [43870 \;\; 33821], \qquad K_2[42376 \;\; -5932]$$

$$L_1 = [27.1155 \;\; -25.2996]^T, \;\; L_2 = [-1.1301 \;\; 0.4450]^T \qquad [6.113]$$

We apply a chicane-type steering angle to the front wheels (see Figure 6.72), at a velocity of $v_x = 20$ m/s.

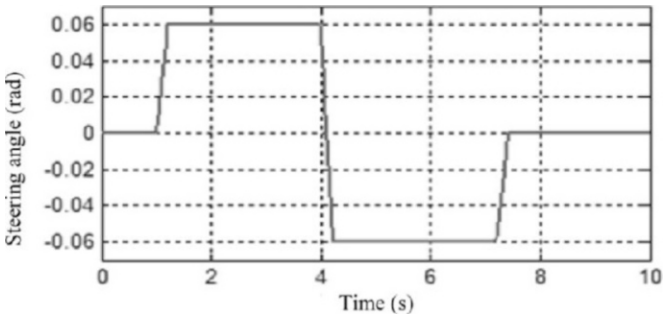


Figure 6.72. *Steering angle of the front wheels (rad)*

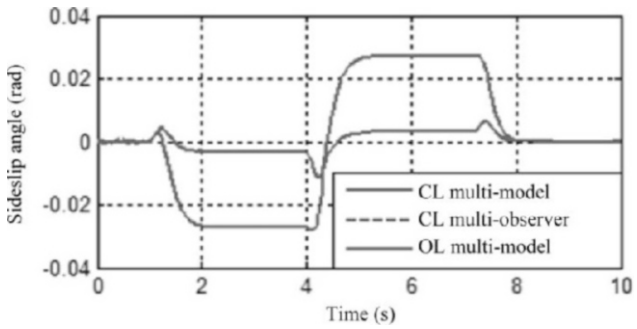


Figure 6.73. *Evolution of the sideslip angle in open- and closed-loop regimes*

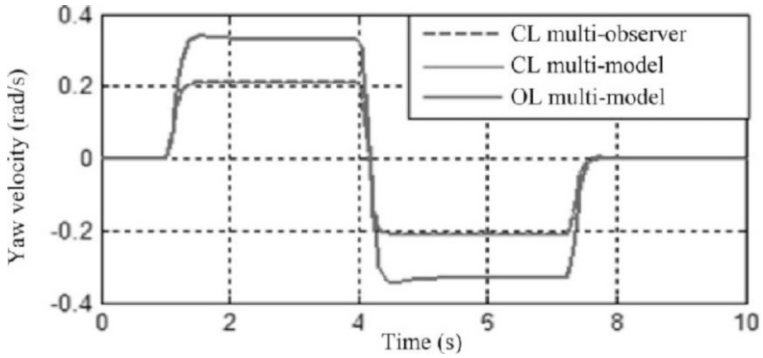


Figure 6.74. *Evolution of the yaw velocity in open- and closed-loop regimes*

We can see in Figures 6.73 and 6.74 that the outputs from the closed-loop multi-model are smaller than those from the open-loop system (the aim is to minimize the sideslip angle in order to guarantee the stability of the system). Figure 6.73 also demonstrates the good estimation of the sideslip angle.

6.4.3.3.3. Simulation with CarSim

We used the multi-observer [6.108] to estimate the sideslip angle on the basis of the measured yaw velocity. Although the multi-model [6.106] was validated by CarSim, Figures 6.75 and 6.76 illustrate the validation of the open-loop multi-model, and Figure 6.76 shows a poor estimation of the sideslip angle for a velocity of $v_x = 8.33$ m/s. This poor estimation is due to noising of the measurements and to external disturbances from CarSim. In order to remedy this problem, an observer-based robust control law H_∞, which takes external disturbances into account, is used.

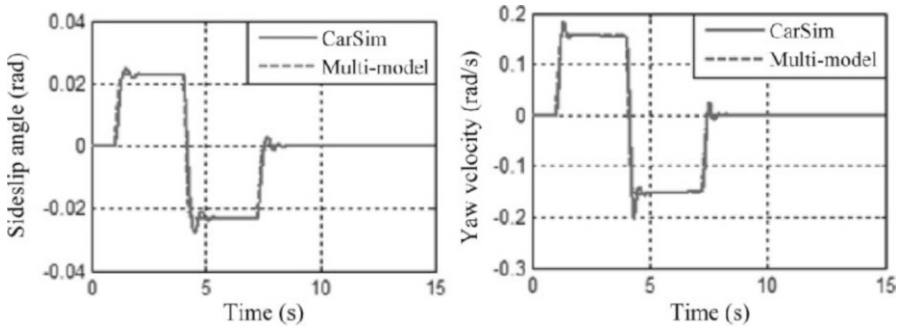


Figure 6.75. *Sideslip angle and yaw velocity*

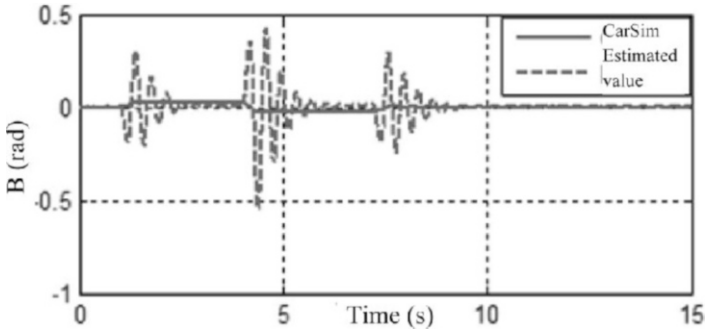


Figure 6.76. *Sideslip angle in CarSim and its estimated value*

6.4.3.4. *Multi-observer-based robust H_∞ control law*

Consider the general case of a nonlinear system described by an uncertain multi-model subject to external disturbances, defined as follows:

$$\begin{aligned}
\dot{x}(t) &= \sum_{i=1}^{M} \mu_i(\xi)\left(\left(A_i + \Delta A_i\right)x(t) + \left(B_{2i} + \Delta B_{2i}\right)u(t) + B_{1i}w(t)\right) \\
z(t) &= \sum_{i=1}^{M} \mu_i(\xi)\left(C_{1i}x(t) + D_{1i}u(t)\right) \\
y(t) &= \sum_{i=1}^{M} \mu_i(\xi)\left(\left(C_{2i} + \Delta C_{2i}\right)x(t) + D_{2i}w(t)\right)
\end{aligned} \qquad [6.114]$$

ΔA_i, ΔB_{2i} and ΔC_{2i} represent the parametric uncertainty matrices of appropriate dimensions. We suppose that they can be expressed as follows:

$$\Delta A_i = D_{Ai}F_i(t)E_{Ai} \quad \Delta B_i = D_{Bi}F_i(t)E_{Bi} \quad \Delta C_i = D_{Ci}F_i(t)E_{Ci} \qquad [6.115]$$

where $E_{Ai}, E_{Bi}, E_{Ci}, D_{Ai}, D_{Bi}$ are constant matrices of appropriate dimensions and the uncertain matrix $F_i(t)$, bounded as follows:

$$F(t)_i^T F(t)_i < I \quad i = 1, \ldots, M \qquad [6.116]$$

The multi-observer in question is of the following form:

$$\begin{aligned}\dot{\hat{x}}(t) &= \sum_{i=1}^{M} \mu_i(\xi)(A_i\hat{x}(t) + B_{1i}w(t) + B_{2i}u(t) - L_i(y(t) - \hat{y}(t))) \\ \hat{y}(t) &= \sum_{i=1}^{M} \mu_i(\xi)(C_{2i}\hat{x}(t) + D_{2i}w(t))\end{aligned} \quad [6.117]$$

The observer-based multi-controller is:

$$u(t) = \sum_{i=1}^{M} \mu_i(\xi)K_i\hat{x}(t) \quad [6.118]$$

The objective is to determine the gains of the multi-controller K_i and the gains of the multi-observer L_i, $i = 1\ldots, M$ so that system [6.114] is asymptotically stable in the presence of external disturbances $w(t)$.

THEOREM 6.2.– [OUD 07] If there are matrices $Z > 0$, $Y > 0$, M_i, J_i and Q_{ij} with $Q_{ij}^T = Q_{ji}$ and positive scalars $\alpha > 0, \varepsilon_{kij}, k = 2, \ldots, 5$ and $i, j = 1, \ldots, M$, such that the following LMIs are satisfied:

$$\begin{pmatrix} \Theta_{ii} & \Lambda_{ii} \\ * & \Psi_{ii} \end{pmatrix} < Q_{ii} \quad i = 1, \ldots, M \quad [6.119]$$

$$\begin{pmatrix} \Theta_{ij}+\Theta_{ji} & \Lambda_{ij}+\Lambda_{ji} \\ * & \Psi_{ij}+\Psi_{ji} \end{pmatrix} \leq Q_{ij} + Q_{ji} \quad i < j \quad [6.120]$$

$$\begin{pmatrix} Q_{11} & \cdots & Q_{1M} & V_{1k}^T \\ \vdots & \ddots & \vdots & \vdots \\ Q_{M1} & \cdots & Q_{MM} & V_{rk}^T \\ V_{1k} & \cdots & V_{Mk} & -I \end{pmatrix} < 0, \quad k = 1, 2, \ldots, M \quad [6.121]$$

where:

$$V_{ik} = \begin{pmatrix} C_{1i}Z + D_{1i}M_k & D_{1i}M_k \end{pmatrix} \quad [6.122]$$

and Θ_{ij}, Λ_{ij}, Ψ_{ij} are defined in equations [6.123], [6.124] and [6.125] respectively, then the stability of multi-model [6.114] with the a performance H_∞ is guaranteed for an attenuation γ via the controller [6.118].

$$\Theta_{ii} = \begin{pmatrix} \Gamma & ZE_{Ai}^T & M_j^T E_{Bi}^T & ZD_{Bi} & ZE_{Ai}^T & M_j^T E_{Bi}^T & ZE_c^T \\ E_{Ai}Z & -\varepsilon_{1ii}I & 0 & 0 & 0 & 0 & 0 \\ E_{Bi}M_j & 0 & -\varepsilon_{2ij}^{-1}I & 0 & 0 & 0 & 0 \\ D_{Bi}^T Z & 0 & 0 & -\varepsilon_{2ij} & 0 & 0 & 0 \\ E_{Ai}Z & 0 & 0 & 0 & -\varepsilon_{4ii}^{-1}I & 0 & 0 \\ E_{Bi}M_j & 0 & 0 & 0 & 0 & -\varepsilon_{3ij}^{-1}I & 0 \\ E_c Z & 0 & 0 & 0 & 0 & 0 & -\varepsilon_{5ii}^{-1}I \end{pmatrix} \quad [6.123]$$

$$\Psi_{ij} = \begin{pmatrix} -2\alpha Z & M_j^T E_{Bi}^T & M_j^T E_{Bi}^T & \alpha I & 0 & 0 & 0 & 0 \\ E_{Bi}M_j & -\varepsilon_{3ij}^{-1} & 0 & 0 & 0 & 0 & 0 & 0 \\ E_{Bi}M_j & 0 & -\varepsilon_{6ij} & 0 & 0 & 0 & 0 & 0 \\ \alpha I & 0 & 0 & A_i^T Y + C_{2j}^T J_i^T + YA_i + J_i C_{2j} & YD_{Bi} & YD_{Ai} & YD_{Bi} & J_i D_c \\ 0 & 0 & 0 & D_{Bi}^T Y & -\varepsilon_{3ij}I & 0 & 0 & 0 \\ 0 & 0 & 0 & D_{Ai}^T Y & 0 & -\varepsilon_{4ii}I & 0 & 0 \\ 0 & 0 & 0 & D_{Bi}^T Y & 0 & 0 & -\varepsilon_{3ij}I & 0 \\ 0 & 0 & 0 & D_c^T J_i^T & 0 & 0 & 0 & -\varepsilon_{5ii}I \end{pmatrix} \quad [6.124]$$

$$\Lambda_{ii} = \begin{pmatrix} B_{2i}M_j & 0 \\ 0 & 0 \end{pmatrix} \quad [6.125]$$

$$\begin{aligned} \Gamma = & \left(ZA_i^T + M_j^T B_i^T \right) + \left(ZA_i^T + M_j^T B_i^T \right)^T \\ & + \gamma^{-2} B_{1i} B_{1i}^T + \varepsilon_{1ii} D_{Ai} D_{Ai}^T + \varepsilon_{6ij} D_{Bi} D_{Bi}^T \end{aligned} \quad [6.126]$$

The gains of the controller and of the observer are given by:

$$K_i = M_i Z^{-1},\ L_i = Y^{-1} J_i \quad [6.127]$$

6.4.3.4.1. Application to the model of the vehicle with 2d.o.f

Consider the following multi-model vehicle:

$$\begin{aligned}\dot{x}(t) &= \sum_{i=1}^{2} h_i\left(\left|\alpha_f\right|\right)\left(\left(A_i+\Delta A_i\right)x(t)+\left(Bfi+\Delta_{Bfi}\right)\delta_f(t)+BM_z(t)\right)\\ z(t) &= \sum_{i=1}^{2} h_i\left(\left|\alpha_f\right|\right)C_{1i}x(t)\\ y(t) &= \sum_{i=1}^{2} h_i\left(\left|\alpha_f\right|\right)\left(\left(C_{2i}+\Delta C_{2i}\right)x(t)+\left(D_i+\Delta D_i\right)\delta_f(t)\right)\end{aligned} \quad [6.128]$$

with:

$$x(t) = \begin{bmatrix} v(t) & r(t)\end{bmatrix}^T \quad \Delta C_{2i}(t) = D_{Ci}F_i(t)E_{Ci}$$

where $a_y(t) = \dot{v}(t) + u(t)r(t)$ is the lateral acceleration and *z(t)* represents the output needing to be controlled.

$$A_i = \begin{bmatrix} -2\dfrac{C_{fi}+C_{ri}}{mV_x} & -2\dfrac{C_{fi}a_f - C_{ri}a_r}{mV_x} - V_x \\ -2\dfrac{C_{fi}a_f - C_{ri}a_r}{I_Z V_x} & -2\dfrac{C_{fi}a_f{}^2 + C_{ri}a_r{}^2}{I_Z V_x}\end{bmatrix}, Bfi = \begin{bmatrix} 2\dfrac{C_{fi}}{m} \\ 2\dfrac{a_f C_{fi}}{I_Z}\end{bmatrix}, B = \begin{bmatrix} 0 \\ \dfrac{1}{I_Z}\end{bmatrix} \quad [6.129]$$

$$D_i = \begin{bmatrix} 2\dfrac{C_{fi}}{m} \\ 0\end{bmatrix}, C_{1i} = \begin{bmatrix}1 & 0\end{bmatrix}, C_{2i} = \begin{bmatrix} -2\dfrac{C_{fi}+C_{ri}}{mV_x} & -2\dfrac{C_{fi}a_f - C_{ri}a_r}{mV_x} \\ 0 & 1\end{bmatrix} \quad [6.130]$$

We assume that the parametric uncertainties $\Delta A_i, \Delta B_{fi}, \Delta C_{2i}$ and ΔD_i can be written in the following form:

$$\Delta A_i(t) = D_{Ai}F_i(t)E_{Ai}, \Delta B_{fi}(t) = D_{Bfi}F_i(t)E_{Bfi} \quad [6.131]$$

$$\Delta D_i(t) = D_{Di}F_i(t)E_{Di}, \Delta C_{2i}(t) = D_{Ci}F_i(t)E_{Ci} \quad [6.132]$$

with:

$$D_{Ai} = D_{Bfi} = D_{Ci} = D_{Di} = \begin{bmatrix} 0.1 & 1 \\ 1 & 0.1 \end{bmatrix}$$

$$E_{Ai} = \begin{bmatrix} -2\dfrac{C_{fi}+C_{ri}}{mV_x} & -2\dfrac{C_{fi}a_f - C_{ri}a_r}{mV_x} \\ -2\dfrac{C_{fi}a_f - C_{ri}a_r}{I_Z V_x} & -2\dfrac{C_{fi}a_f{}^2 + C_{ri}a_r{}^2}{I_Z V_x} \end{bmatrix},\ E_{Bfi} = \begin{bmatrix} 2\dfrac{C_{fi}}{m} \\ 2\dfrac{a_f C_{fi}}{I_Z} \end{bmatrix} \quad [6.133]$$

$$E_{Ci} = \begin{bmatrix} -2\dfrac{C_{fi}+C_{ri}}{mV_x} & -2\dfrac{C_{fi}a_f - C_{ri}a_r}{mV_x} \\ 0 & 1 \end{bmatrix},\ E_{Di} = \begin{bmatrix} 2\dfrac{C_{fi}}{m} \\ 0 \end{bmatrix} \quad [6.134]$$

The expression of the control law is:

$$M_Z(t) = -\sum_{i=1}^{2} h_i\left(\left|\alpha(t)_f\right|\right) K_i \hat{x}(t) \quad [6.135]$$

The structure of the multi-observer is based on the structure of the model of the vehicle [6.128]:

$$\begin{aligned} \dot{\hat{x}}(t) &= \sum_{i=1}^{2} h_i(\alpha_f)(A_i\hat{x}(t) + BM_Z(t) + B_{fi}\delta_f(t) + B_w w(t) - L_i(y(t) - \hat{y}(t))) \\ \hat{y}(t) &= \sum_{i=1}^{2} h_i(\alpha_f)(C_{2i}\hat{x}(t) + D_i\delta_f(t) + D_w w(t)) \end{aligned} \quad [6.136]$$

6.4.3.4.2. Simulation results

By solving the LMIs [6.119] and [6.125] in theorem 6.2, with $\varepsilon_{k**} = 1$ (k =2,3,4,5), and $\gamma = 0.95$, we get:

– for $v_x = 8.33$ m/s:

$$\mathrm{K}_1 = 10^3\begin{bmatrix} 0.1390 & -3.8413 \end{bmatrix} \quad \mathrm{K}_2 = 10^3\begin{bmatrix} 1.1926 & -5.5092 \end{bmatrix}$$

$$L_1 = \begin{bmatrix} -1.0340 & 0.4806 \\ -2.7110 & 9.6939 \end{bmatrix} \qquad L_2 = \begin{bmatrix} -0.1064 & -0.2017 \\ -3.9439 & -4.3620 \end{bmatrix} \quad [6.137]$$

– for $v_x = 20$ m/s:

$$K_1 = 10^3 [-0.0797 \quad -5.5306] \quad K_2 = 10^3 [0.4507 \quad -5.5281]$$

$$L_1 = \begin{bmatrix} 1.4342 & 0.0461 \\ -0.4658 & 0.0743 \end{bmatrix} \quad L_2 = \begin{bmatrix} 2.0142 & 0.0870 \\ -0.5806 & -0.0424 \end{bmatrix} \qquad [6.138]$$

The steering angle of the wheels is given by Figure 6.72. The evolution of the two state variables (lateral velocity and yaw velocity) is shown in Figures 6.77 and 6.78.

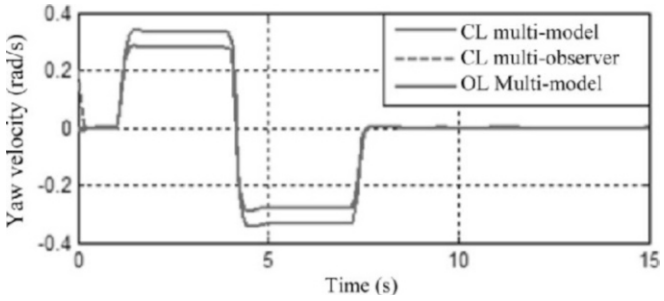


Figure 6.77. *Evolution of the lateral velocity in an open- and closed-loop system at* $v_x = 20$ *m/s*

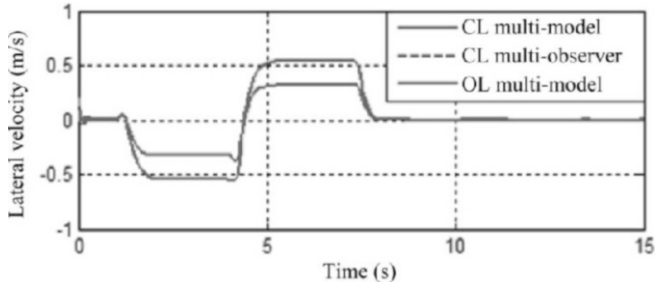


Figure 6.78. *Evolution of the yaw velocity in an open- and closed-loop system at* $v_x = 20$ *m/s*

6.4.3.4.3. Simulation with CarSim

The two trajectories traced in Figure 6.79 show the lateral velocity estimated by the observer and the yaw velocity in an open-loop system at $v_x = 20$ m/s.

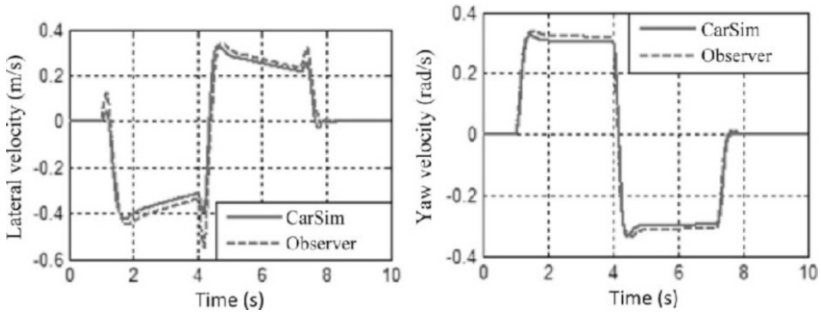


Figure 6.79. *Estimation of the lateral velocity and the yaw velocity*

We observe the quality of estimation of the lateral velocity and the yaw velocity. Quite unlike the conditions from theorem 6.1, the conditions from theorem 6.2, which take account of the parametric uncertainties and of the external disturbances, have enabled us to better estimate these variables.

The control using the moment M_Z in CarSim is generated by acting on the brake torque of each wheel, as shown by Figure 6.80.

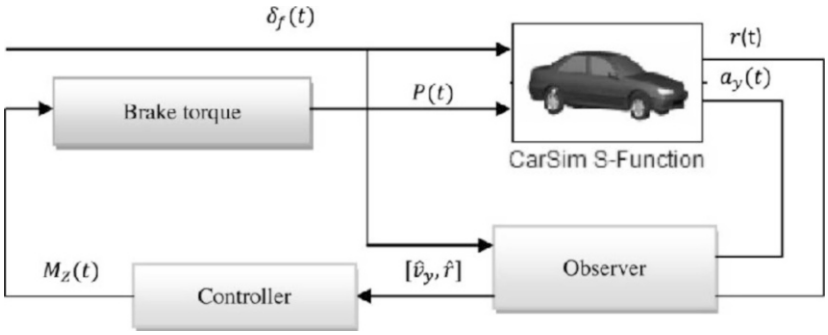


Figure 6.80. *Structure of control with the moment M_Z in CarSim*

Using control law [6.118] in CarSim, the estimation of the lateral velocity and the yaw velocity is shown in Figure 6.81.

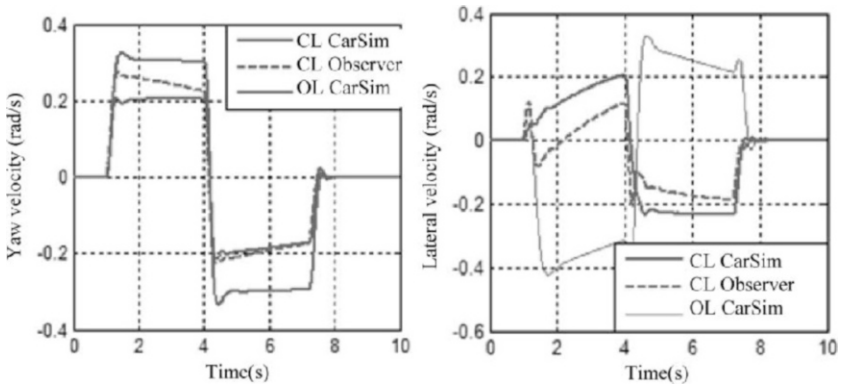


Figure 6.81. *Evolution of the state variables with and without control in CarSim*

Note that the controller [6.118] outlined in this section ensures the vehicle's performances in terms of stability remain good in spite of the noising in the measurements and the slight variation of the longitudinal velocity during the maneuver.

6.4.3.5. *Observer-based multi-model robust control*

Consider the following uncertain multi-model of the vehicle with 4 d.o.f:

$$\begin{aligned} \dot{x}(t) &= \sum_{i=1}^{2} \mu_i(\left|\alpha_f(t)\right|)\left[(A_{i0}+\Delta A_i)x(t)+(B_{i0}+\Delta B_i)\delta_f(t)\right] \\ y(t) &= \sum_{i=1}^{2} \mu_i(\left|\alpha_f(t)\right|)\left[(C_{i0}+\Delta C_i)x(t)+(D_{i0}+\Delta D_i)\delta_f(t)\right] \end{aligned} \quad [6.139]$$

with:

$$x=\begin{pmatrix} v_y \\ \dot{\psi} \\ \phi \\ \dot{\phi} \end{pmatrix}, A_{i0}=\begin{pmatrix} -2\dfrac{(C_{fi0}+C_{ri0})}{m.v_x} & 2\dfrac{(C_{fi0}a_f-C_{ri0}a_r)}{m.v_x}-v_x & 0 & 0 \\ -2\dfrac{(C_{ri0}a_r+C_{fi0}a_f)}{I_z.v_x} & -2\dfrac{(C_{fi0}a_f^2-C_{ri0}a_r^2)}{I_z.v_x} & 0 & 0 \\ 0 & 0 & 0 & 1 \\ -\dfrac{2m_s h_{roll}(C_{fi0}+C_{ri0})}{I_x.m.v_x} & \dfrac{2m_s h_{roll}(C_{ri0}a_r-C_{fi0}a_f)}{I_x.m.v_x} & \dfrac{m_s g h_{roll}-K_{roll}}{I_x} & \dfrac{C_{roll}}{I_x} \end{pmatrix}$$

$$B_{i0}=\begin{pmatrix}\frac{2C_{fi0}}{m}\\ \frac{2a_fC_{fi0}}{I_z}\\ 0\\ \frac{2m_sh_{roll}C_{f0i}}{I_xm}\end{pmatrix},C_{i0}=\begin{bmatrix}-2\frac{(C_{fi0}+C_{ri0})}{m.v_x} & 2\frac{(C_{fi0}a_f-C_{ri0}a_r)}{m.v_x} & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\end{bmatrix} \quad [6.140]$$

$$D_{i0}=\begin{pmatrix}\frac{2C_{fi0}}{m}\\ 0\\ 0\end{pmatrix},F_i=\begin{pmatrix}f_i & 0\\ 0 & f_i\end{pmatrix},R_i=\begin{pmatrix}d_i & 0\\ 0 & d_i\end{pmatrix},\Delta A_i=R_iF_iE_{1i},\Delta B_{fi}=R_iF_iE_{f2i} \quad [6.141]$$

$$E_{f2i}=\begin{pmatrix}\frac{2C_{fi0}}{m}\\ \frac{2a_fC_{fi0}}{I_z}\\ 0\\ \frac{2m_sh_{roll}C_{f0i}}{I_xm}\end{pmatrix}$$

$$E_{1i}=\begin{pmatrix}-2\frac{(C_{fi0}+C_{ri0})}{m.v_x} & 2\frac{(C_{fi0}a_f-C_{ri0}a_r)}{m.v_x} & 0 & 0\\ -2\frac{(C_{ri0}a_r+C_{fi0}a_f)}{I_z.v_x} & -2\frac{(C_{fi0}a_f^2-C_{ri0}a_r^2)}{I_z.v_x} & 0 & 0\\ 0 & 0 & 0 & 1\\ -\frac{2m_sh_{roll}(C_{fi0}+C_{ri0})}{I_x.m.v_x} & \frac{2m_sh_{roll}(C_{ri0}a_r-C_{fi0}a_f)}{I_x.m.v_x} & \frac{m_sgh_{roll}-K_{roll}}{I_x} & \frac{C_{roll}}{I_x}\end{pmatrix}$$

[6.142]

The functions $\mu_i(|\alpha_f|)$ are the activation functions. $x(t)$ is the state vector, $\delta_f(t)$ is the input vector (steering angle), $y(t)$ is the output vector of the system and $\Delta A_i, \Delta B_i, \Delta C_i$ and ΔD_i represent the parametric uncertainties of the model.

We consider that the roll angle ϕ is not measurable. The structure of the observer is based on the structure of the model of the vehicle [6.139]:

$$\begin{aligned}\dot{\hat{x}} &= \sum_{i=1}^{2} \mu_i \left(\left|\alpha_f\right|\right)\left[A_i\hat{x} + B_i\delta_f + G_i\left(y - \hat{y}\right)\right] \\ \hat{y} &= \sum_{i=1}^{2} \mu_i \left(\left|\alpha_f\right|\right)\left[C_i\hat{x} + D_i\delta_f\right]\end{aligned} \qquad [6.143]$$

The control law considered is of the form:

$$\delta_f = -\sum_{i=1}^{2} \mu_i \left(\left|\alpha_f\right|\right) K_i\hat{x} \qquad [6.144]$$

The control strategy that we use is shown in Figure 6.82. In a normal situation, it is the driver who controls the vehicle and when the estimated roll angle approaches a threshold value (risk indicator), the controller acts so that this limit will never be surpassed.

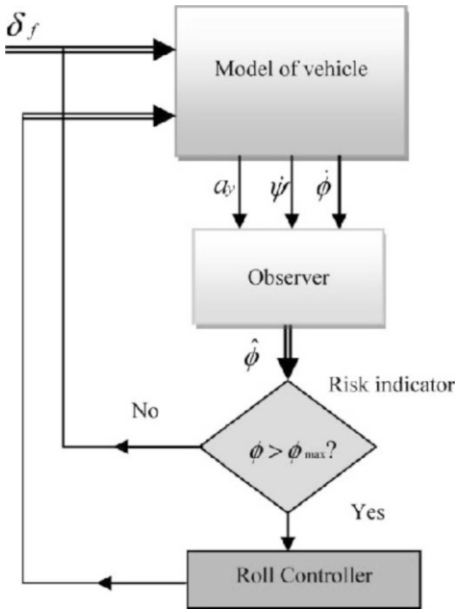


Figure 6.82. *Strategy for controlling the roll angle*

The dynamics of the state estimation error are expressed thus:

$$\dot{e}(t) = \dot{x}(t) - \dot{\hat{x}}(t) \quad [6.145]$$

On the basis of systems [6.139] and [6.142], we obtain the following equations:

$$\begin{aligned}\dot{x}(t) = &\sum_{i=1}^{2}\sum_{j=1}^{2}\mu_i\left(\left|\alpha_f\right|\right)\mu_j\left(\left|\alpha_f\right|\right)\left[\left(A_i - B_iK_j\right)x + B_{ri}K_je + B_{fi}\delta_f\right] \\ &+ \sum_{i=1}^{2}\sum_{j=1}^{2}\mu_i\left(\left|\alpha_f\right|\right)\mu_j\left(\left|\alpha_f\right|\right)\left[\left(\Delta A_i - \Delta B_{ri}K_j\right)x + \Delta B_{ri}K_jx\right]\end{aligned} \quad [6.146]$$

$$\dot{\hat{x}} = \sum_{i=1}^{2}\sum_{j=1}^{2}\mu_i\left(\left|\alpha_f\right|\right)\mu_j\left(\left|\alpha_f\right|\right)\left(A_i - B_{ri}K_j\right)\hat{x} + \sum_{i=1}^{2}\sum_{j=1}^{2}\mu_i\left(\left|\alpha_f\right|\right)\mu_j\left(\left|\alpha_f\right|\right)G_iC_je \quad [6.147]$$

$$\dot{e} = \sum_{i=1}^{2}\sum_{j=1}^{2}\mu_i\mu_j\left(A_i - G_iC_j + \Delta B_{ri}K_j\right)e + \sum_{i=1}^{2}\sum_{j=1}^{2}\mu_i\mu_j\left(\Delta A_i - \Delta B_{ri}K_j\right)x \quad [6.148]$$

The stabilization of the uncertain multi-model [6.139], with reconstruction of the state by the observer [6.143], is obtained by the conditions given in [CHA 06].

6.4.3.5.1. Simulation results

In the context of this work, we consider both the nonlinearities relating to the forces of contact, the variations of the tire grip on the road and the unavailability of the measurement of the roll angle of the vehicle. The steering angle of the front wheels is indicated in Figure 6.83. In this maneuver, we cause a dangerous situation at time $t = 14$ s in order to test and validate the observer and the controller advocated here.

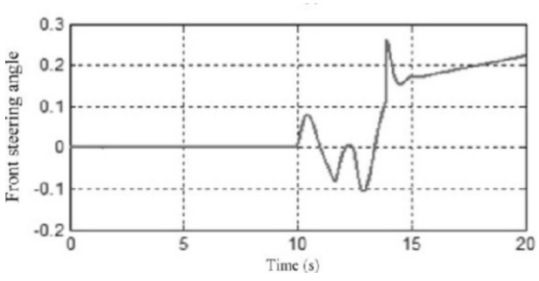


Figure 6.83. *Steering angle of the front wheels*

By solving the conditions obtained in [CHA 06] using the LMI toolbox in Matlab®, we get:

$$K_1 = \begin{bmatrix} -0.1219 & 0.7831 & 0.3233 & 0.0271 \end{bmatrix},$$
$$K_2 = \begin{bmatrix} -0.0346 & 0.7867 & 0.2737 & 0.0232 \end{bmatrix}$$

$$G1 = \begin{bmatrix} -2.9996 & 31.0080 & 4.5860 \\ 4.5600 & 76.3180 & 18.5189 \\ 0.7881 & 77.3784 & -309.3142 \\ 2.4548 & -7.7978 & 49.3901 \end{bmatrix} \quad G2 = \begin{bmatrix} -7.2614 & -14.0371 & 4.2722 \\ 0.2679 & 31.4561 & 16.1053 \\ -0.5252 & 3.9349 & -312.4993 \\ 2.5512 & -0.6617 & 49.7290 \end{bmatrix}$$

The simulation results are shown in Figures 6.84 and 6.85.

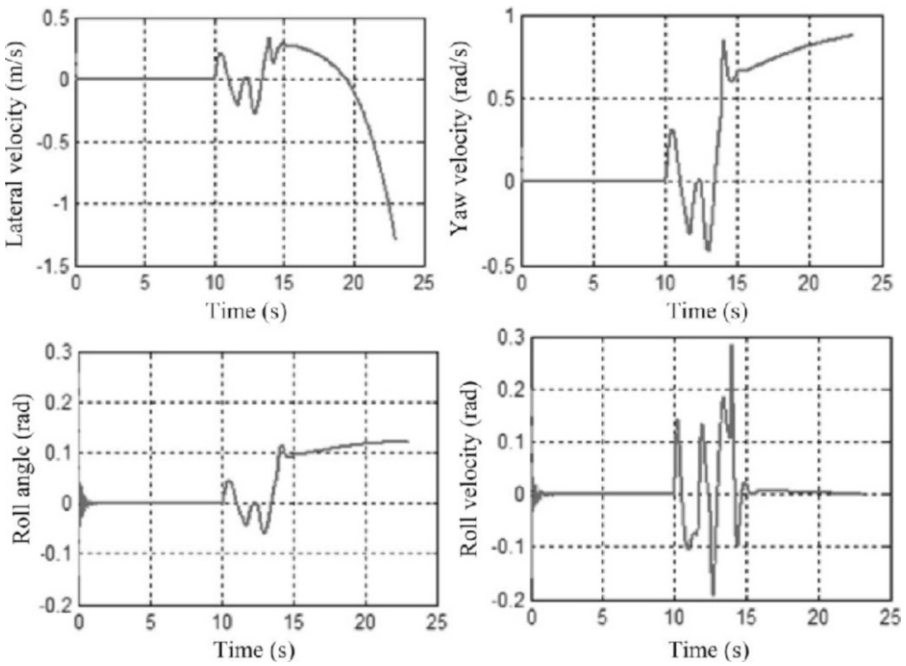


Figure 6.84. *Estimation of the state variables for the multi-model with 4 d.o.f in an open-loop system*

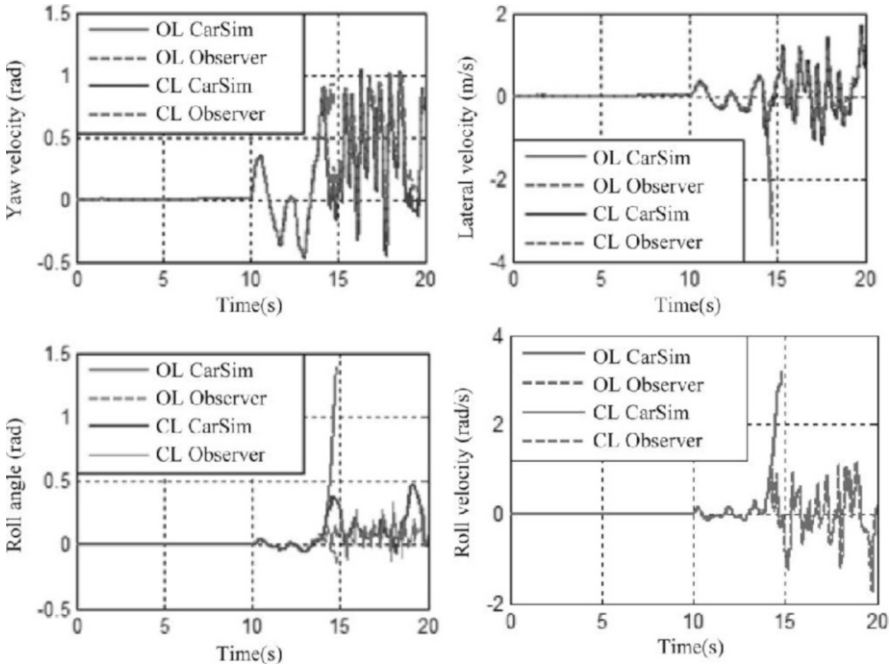


Figure 6.85. *Evolution of the state variables with and without control in CarSim*

These simulation results show that the controller [6.144] guarantees the stability of the vehicle after the driver makes a dangerous maneuver.

6.4.4. *Conclusion*

In this chapter, we have validated a number of models (linear and multi-model) of vehicles using CarSim. Then, we applied robust control laws to improve the vehicle's performances in terms of stability and maneuverability.

To begin with, we approximated the lateral forces using multi-model expressions. This approximation enabled us to obtain an uncertain multi-model which represents the lateral dynamics of the vehicle throughout the entire range of variation of the slip angle α_f. This multi-model was then validated using the vehicle simulator "CarSim". The quality of this modeling in relation to the linear model has also been demonstrated. Next, we presented three control laws to control the dynamics of the vehicle. The first control law uses reconstructed state feedback. The objective of the second control law with H_∞ performance was to overcome the

disturbances and take account of the parametric uncertainties of the system. The third is an observer-based multi-model robust control law. This is applied to control the roll dynamics of the vehicle in case of dangerous *maneuvers*.

6.5. Bibliography

[BOY 94] BOYD S., EL GHAOUI L., FERON E., *et al.*, "Linear matrix inequalities in system and control theory", *SIAM*, Philadelphia, 1994.

[CHA 00] CHADLI M., MAQUIN D., RAGOT J., "Relaxed stability conditions for the T-S fuzzy systems", *IEEE International Conference on Systems, Man, and Cybernetics*, vol. 5, pp. 3514–3519, 2000.

[CHA 02a] CHADLI M., MAQUIN D., RAGOT J., "Output stabilization in multiple model approach", *International Conference on Control Applications*, vol. 2, pp. 1315–1320, 2002.

[CHA 02b] CHADLI M., Stabilité et stabilisation des multimodèles, Thesis, INPL-CRAN, Nancy, December 2002.

[CHA 06a] CHADLI M., ELHAJJAJI A., "Observer-based robust fuzzy control of nonlinear systems with parametric uncertainties-comment on", *Fuzzy Sets and Systems Journal*, vol. 157, no. 9, pp. 1276–1281, 2006.

[CHA 06b] CHADLI M., "On the stability analysis of uncertain fuzzy models", *International Journal of Fuzzy Systems*, vol. 8, no. 4, pp. 224–231, December 2006.

[CHA 12a] CHADLI M., BORNE P., *Multimodèles en Automatique : outils avancés d'analyses et de synthèse*, Hermès, Paris, 2012.

[CHA 12b] CHADLI M., BORNE P., *Multiple models Approach in Automation: Takagi-Sugeno Fuzzy Systems*, ISTE Ltd, London and John Wiley & Sons, London, 2013.

[CHA 12c] CHADLI M., GUERRA T.M., "LMI Solution for Robust Static Output Feedback Control of Takagi-Sugeno Fuzzy Models", *IEEE Transactions on Fuzzy Systems*, vol. 20, no. 6, 2012.

[CHA 12d] CHADLI M., KARIMI H.R., "Robust observer design for unknown inputs Takagi-Sugeno models", *IEEE Transactions on Fuzzy Systems*, 2012.

[CRA 86] CRAIG J.J., *Introduction to Robotics Mechanics and Control*, Addison-Wesley, 2nd edition, Boston, 1986.

[DAH 11] DAHMANI H., Système embarqué d'évaluation de la dynamique du véhicule et d'aide à la conduit, Thesis, University of Picardie Jules Verne, Amiens, 2011.

[DOM 88] DOMBRE E., KHALIL W., *Modélisation et commande des robots*, Hermès, Paris, 1988.

[FAI 87] FAISANDIER J., *Mécanismes oléo-hydrauliques*, Bordas, Paris, 1987.

[GUE 04] GUERRA T.M., VERMEIREN L., "LMI-based relaxed nonquadratic stabilization conditions for nonlinear systems in the Takagi–Sugeno's form", *Automatica*, vol. 40, no. 5, pp. 823–829, 2004.

[HA 02] HA Q., SANTOS M., NGUYEN Q., *et al.*, "Robotic excavation in construction automation", *IEEE Robotics & Automation Magazine*, 2002.

[HEM 92] HEMAMI A., DANESHMEND L., "Force analysis for automation of the loading operation in an L-H-D loader", *Proceedings of the IEEE Conference on Robotics and Automation*, Nice, France, May 1992.

[KOI 92] KOIVO A., "Controlling an intelligent excavator for autonomous digging in difficult ground", *Proceedings the 9th International Symposium on Automation and Construction*, Tokyo, Japan, June 1992.

[LIE 77] LIEGEOIS A., "Automatic supervisory control of the configuration and behavior of multibody mechanisms", *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 7, no. 12, pp. 868–871, 1977.

[MER 76] MERRIT E., *Hydraulic Control Systems*, John Wiley, New York, 1976.

[MES 08] MESSOUSSI W., Contribution à la commande robuste de la dynamique du véhicule, Thesis, University of Picardie Jules Verne, Amiens, 2008.

[ODA 04] ODVA, Technologie de réseau : ODVA, l'organisation qui supporte des technologies de réseau construites sur DeviceNet, available at http://www.odva.org/, 2004.

[OUD 07] OUDGHIRI M., Synthèse de systèmes d'assistance à la conduite robuste et tolérants aux fautes, Thesis, University of Picardie Jules Verne, Amiens, 2008.

[PAC 88] PACEJKA H.B., Modelling of the pneumatic tyre and its impact on vehicle dynamic behavior, Technical Report i72B, University of Technology, Delft, Netherlands, 1988.

[PEP 05] PEPIN T., Interfacer la commande d'une pelleteuse et d'un logiciel de simulation de conduite, technical report, Compiegne University of Technology France, June 2005.

[RYH 91] RYHMING L., *Dynamique des fluides*, Presses polytechniques et universitaires romandes, Lausanne, 1991.

[SEW 88] SEWARD D., BRADLEY D., BRASSERIE R., "The development of research models for automatic excavation", *Proccedings of the 5th International Symposium on Automation and Robotics in Construction*, Tokyo, Japan, 1988.

[SEW 96] SEWARD D., MARGRAVE F., SOMMERVILLE I., *et al.*, "LUCIE the robot excavator – design for system safety", *Proceedings IEEE International Conference on Robotics and Automation*, Minneapolis, United States, May 1996.

[SEW 00] SEWARD D., PACE C., MORREY R., *et al.*, "Safety analysis of autonomous excavator functionality", *Proceedings of the International Conference on Reliability Engineering and System Safety, on Robotics and Automation*, Minneapolis, United States, 2000.

[SIN 95] SINGH N., ZGHAL H., SEPHERI N., *et al.*, "Coordinated-motion control of heavy-duty industrial machines with redundancy", *Robotica*, vol. 13, pp. 623–633, 1995.

[TAN 98] TANAKA K., IKEDA T., HE Y.Y., "Fuzzy regulators and fuzzy observers: relaxed stability conditions and LMI-based design", *IEEE Transactions Fuzzy Systems*, vol. 6, no. 1, pp. 250–256, 1998.

[TAN 01] TANAKA K., HORI T., WANG H.O., "A fuzzy Lyapunov approach to fuzzy control system design", *IEEE American Control Conference*, vol. 6, pp. 4790–4795, 2001.

[TAK 85] TAKAGI M., SUGENO M., "Fuzzy identification of systems and its application to modelling and control", *IEEE Transactions on Systems Man and Cybernetics*, vol. 15, no. 1, pp. 116–132, 1985.

[TAK 94] TAKAHASHI H., TSUKAMOTO Y., KAMATA H., *et al.*, "Model analysis on the bucket control for automatic shoveling", *Proceedings 16th International Conference on Computers and Industrial Engineering*, March 1994.

[WHI 69] WHITNEY D.E., "Resolved-motion rate control of manipulators and human prosthesis", *IEEE Transaction on Man-Machine Systems and Control*, vol. 10, pp. 47–53, 1969.

[ZGH 90] ZGHAL H., DUBEY R., EULER J., "An efficient gradient projection optimization for manipulators with multiple degrees of redundancy", *Proceedings IEEE International Conference on Robotics and Automation*, Cincinnati, United States, pp. 1006–1011, 1990.

Chapter 7

Real-time Implementation

7.1. Implementation of algorithms on real-time targets around distributed architectures

7.1.1. *Introduction*

Once we have chosen the control law to use to guide the process, we need to implement that control law with its model (depending on the type of control) on a real-time target. Thus, we need to describe the type of real-time targets we encounter in the industry. The first, and most widely used, is a PLC-SCADA structure [CAB 07], arranged into one or several communications networks (Ethernet, CAN, RTIE: real time industrial ethernet). This structure enables us to act on the actuators and receive data from the sensors about the process.

The second structure is a variant for mobiles or vehicles, i.e. onboard embedded systems. It is based on a decentralized architecture around a CAN fieldbus. This structure can be designed using an industrial PC or a Rabbit micro-controller.

The main problems arise around two kinds of issue: the development of algorithms with an object-oriented approach from the very beginning of their design within a framework, and the reliability of the distributed architecture in terms of the control loop (networked control system).

In this seventh chapter on the *implementation of algorithms on real-time targets around distributed architectures*, we shall first give an introduction to *object-oriented programming (OOP) in the case of a framework* wherein, once we have

---

Chapter written by Marco PEZZETTI and Hervé COPPIER.

recapped exactly what a programmable logic controller is, the UNICOS system at CERN is discussed in detail. We then go on to present the *multi-controller object* and the tools associated therewith, for implementing the control laws desired and chosen by the user for his process. Finally, we describe a *distributed architecture for control: the excavator-loader testing array* in the case of a vehicle (onboard system for a building site machine) with its reliability and rapidity tests carried out on a specific array.

7.1.2. *Object-oriented programming in the case of a framework*

7.1.2.1. *The programmable logic controller*

Programmable logic controllers (PLCs) relate to the control of physical processes by generating control signals and acquiring data directly about the process. The hardware architecture usually comprises an input/output interface, a processor, memories and communication modules. The PLC is very often used in conjunction with human machine interface (HMI) supervision and external tools. Figure 7.1 simply summarizes the operation of a PLC in an automated installation.

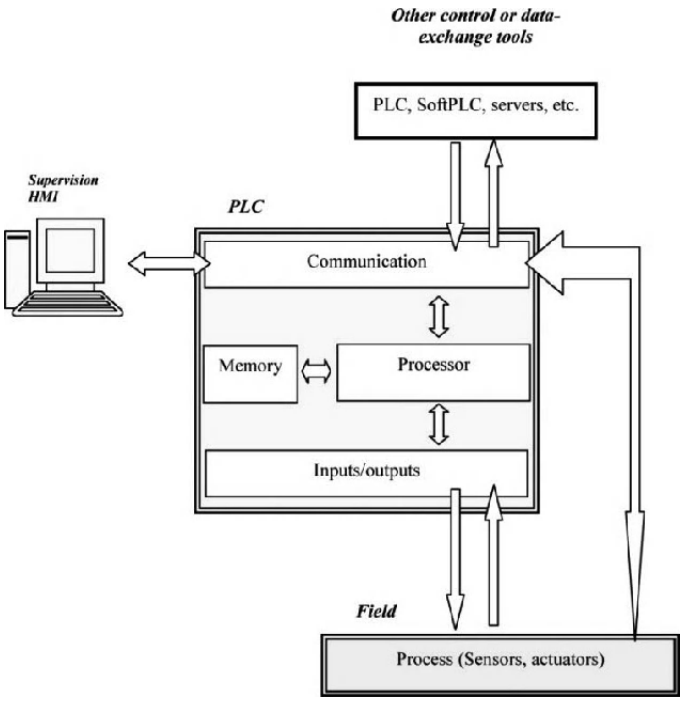


Figure 7.1. *A PLC in an automated system*

7.1.2.1.1. Implementation in line with the norm IEC 61131-3

Work has been done on the implementation of algorithms on PLCs in line with IEC norm 61131, relating to the specification and architecture of programming languages. This norm is divided into numerous sub-parts, including the norm IEC 61131-3, about the languages themselves and their typical constituent elements.

The five standard languages for logic controllers described by this norm are:

– the textual language *instruction list* (IL);

– the textual language *structured text* (ST);

– the graphic language *ladder diagrams* (LD);

– the graphic language *functional block diagrams* (FBD);

– the sequential language Grafcet SFC (*sequential functional chart*).

These languages share certain programming properties, which means that any implementation in one of the languages can be "used" in any of the other four. This strong characteristic of the norm makes possible numerous development strategies to deliver similar functions. The interest in and choice of one language over another is at the developer's discretion.

In order to guarantee their compatibility, the five languages have common elements, which – according to [FRA 01] – are:

– the types of data and variables (elementary or structured);

– the program organization units (POUs) which communicate by way of input/output interface variables or global variables;

– sequential function charts.

The projects of implementation of the algorithms on PLCs presented here are in line with the norm IEC 61131-3. The "objects" (POUs) were developed using the language ST. The tests, for their part, use the language FBD, which facilitates quick visualization of the variables and simple forcing of certain values. Most of the results were obtained with a Premium PLC made by Schneider, using the development software Unity V2.1.

7.1.2.1.2. Example of a control system

The control system for the GCS project uses industrial solutions. This arbitrary choice means that in designing the command/control system, we are able to focus on its implementation and development. The control systems use industrial programmable logic controllers from Schneider Electric (TSX Premium). The

associated SCADA, provided by ETM, is a PVSS II. The communication between these two modules uses the Modbus TCP protocol. The interfacing between the field and the PLC is organized by deported modules made by WAGO. The input/output modules (local or deported) from Schneider are not used, and communication between the WAGO and TSX Premium modules runs through a Profibus communication module at the level of the PLC's chassis (see Figure 5.23).

7.1.2.2. *The UNICOS system at CERN*

Automated mechanisms have a crucially important role to play in the operation of the accelerator, and also in the experiments. Whether a question of cryogenics or vacuums, the control systems are omnipresent, from the moment construction begins to the very end of the detector's life. In order to standardize them, CERN used an existing idea as the basis for developing a series of generic objects common to all the types of installations, which enables us to represent each process and considerably simplify the control of a large number of inputs/outputs. Thus, from one domain to another, the standard to be applied remains the same, which greatly improves the efficiency of the programmers as operators.

UNICOS (UNIfied COntrol System) is a standard which was put in place as part of the control/command project for the cryogenic system of the LHC. It is based on an object-oriented architecture on three levels:

– the supervision layer, comprising the SCADA, the supervision center and the operating workstations (OWSs);

– the control layer, including the CPUs of the PLCs and the different engineering workstations (EWSs) used to program the PLCs;

– the field layer, which forms the link between the different inputs/outputs and the control system.

At present, the SCADA in the supervision layer uses the software PVSS® (ETM-Siemens). The automated layer is made up entirely of Schneider PLCs (either Quantum or Premium) and the programming used UNITY-PRO®. In a second phase, Siemens machines have been added to the standard thanks to the development of S7 UNICOS – an adaptation of the UNICOS objects for Step 7®. The different pieces of information from the supervision layer are stored on a data server (DS) which communicates using Ethernet TCP/IP technology with the PLC to update the values displayed on the different reports and with the operating workstations.

There are two types of workstations. Some include the supervision software and exchange information with the data server. New workstations use a Terminal

Service to connect remotely and therefore do not need to have the SCADA installed. The Terminal Service is a sort of "secure" gateway between the *technical network* and CERN's publicly-available network, known as GPN (*general purpose network*), which enables the operators to assume control of the supervision via a connection made through the GPN.

Today, new possibilities for architectures and control are presenting themselves constantly. The *Control Equipment* experts are constantly concerned with new aspects for the control systems improvement aiming to put in place a reliable and effective system capable of feeding information to the operators and cryogenic engineers about the state of the installations. It is involved on three detectors located on the perimeter of the LHC: CMS, ATLAS and TOTEM, and on many experiments such as CAST – a solar telescope project – MERIT and detectors based in the northern zone (e.g. NA62) and the western zone, collecting data from the collisions caused by the SPS (super proton synchrotron).

Only the most recent experiments have the benefit of the UNICOS strategy, with the others still operating on the basis of obsolete distributed control systems (DCS). With the aim of extending the standardization, many singular control systems will be migrated in the coming years.

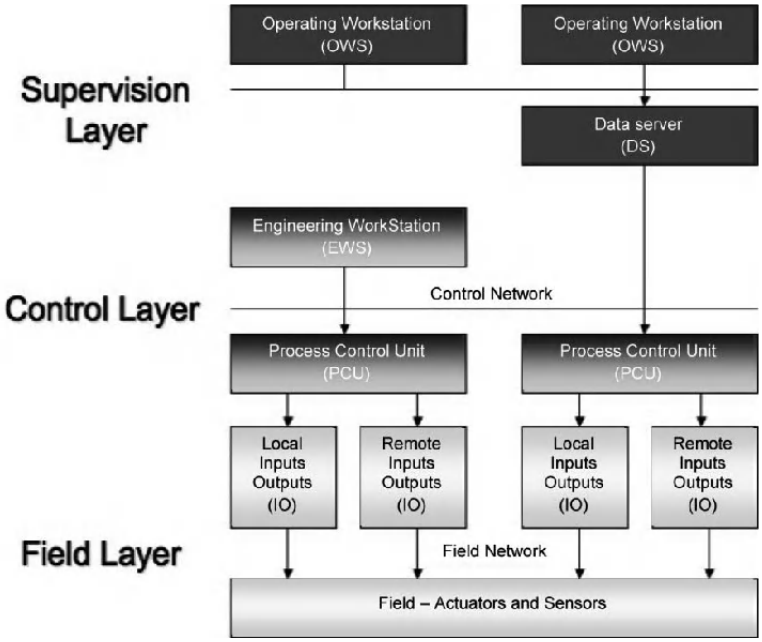


Figure 7.2. *Communication structure – the three UNICOS layers*

7.1.2.2.1. UNICOS objects

The programming standard is based on subdivision of the process into modules and objects which represent the whole. Thus, analysis has led to the creation of three types of object or module which are embedded in parallel in the PLC and the supervision system. By way of the "middleware", which manages the memory space and the communication of a PLC and the architecture imposed by UNICOS, every object involved via the project under the engineering workstation (EWS) will be linked to an entity in the supervision system that is directly available. The requirements of projects such as those currently in motion have led CERN to develop control-command systems which are extremely comprehensive and rich with numerous peculiarities. The UNICOS objects are the result of this approach, and integrate the experience gained from previous projects. Thus, numerous functionalities have been developed, and the system is regularly updated.

The logic objects are distributed into three types of interconnected objects:

– IO: Input/Output Objects:

– DO and AO: digital outputs (on/off) and analog outputs;

– DI and AI: digital inputs and analog inputs;

• FO: *Field Objects*:

– lLocal: equipment controlled locally only;

– OnOff: binary state equipment (on/off);

– Analog: analog equipment;

– Anadig: process with analog input and output in the form of MLI impulses.

• CO: *Control Objects*:

– Controller: regulating object – PID;

– Alarm: object to validate or deactivate alarms and interlocks;

– Process Control: object used to control other objects in the CO or FO layers.

7.1.2.2.2. Inter-object relations

The IO objects, to some degree, represent a link between the field equipment and the control system. The signals sent or received by the system are transmitted through these objects, which have a physical address and are linked to a space on an input/output card on the PLC. The AI and DI signals represent feedback on the state of the instruments and, more generally, the sensors. The DO and AO objects send signals which will be directly adapted to the process. The Field Objects typically

represent the whole of the controlled instrumentation system: valves, motors, compressors, etc. – all the actuators in the installation. The Control Objects are at a higher level, which means they are able to control other objects. A Control Object may control a number of other Control Objects, each of which is in control of one or more Field Objects.

It is possible for a Field Object to send a signal to an AO which is not suitable for the machine connected to it. In this case, the signal is put through a computing IO which transforms the transmitted signal into a signal that can be interpreted and executed by the actuator connected in the field. The opposite situation is also possible, so that if a sensor sends back data which are not interpretable, those data are put through a computing object which conditions them.

There are four modes for controlling objects. In *Forced* mode, the supervising operators can manipulate the inputs to an object from a Operating Workstation, and manually force a value without the control/command system being able to go back into Auto mode by way of an AuAuMoR (*auto auto mode request*, whereby control can be taken back from manual to automatic mode). *Auto* mode enables the object to be piloted by a process control object (PCO). In *Manual* mode, the object is controlled by the operator from the OWS. Finally, *Local Drive* mode means that the object is controlled (or driven) locally in the field.

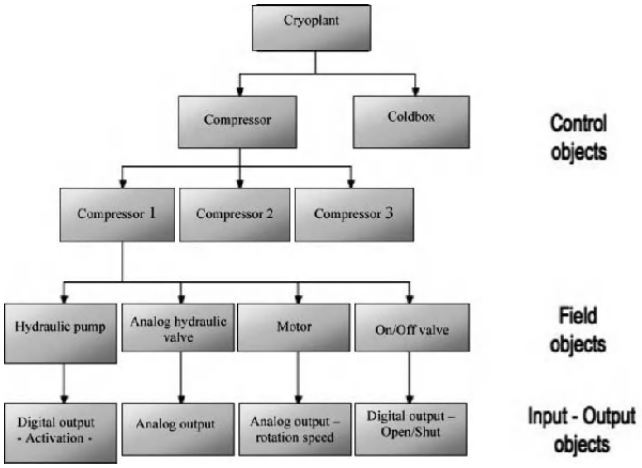


Figure 7.3. *Control architecture – the three types of object*

7.1.2.2.3. I/O objects

The IO objects form the link between the process and the representative equipment in the control system. These objects can work with digital values (Digital

Input/Output) or analog values (Analog Input/Output). If an error in input/output is detected on a card on the PLC, these objects are put in IOError mode, and inform any objects that try to connect to them of this. As we can see in Figure 7.4, there are different ways of approaching an object depending on its nature:

– Process Input: information reflecting the status of an object or, in this particular case, sent from the sensor;

– Auto Request: an order sent from another object;

– Manual Request: information modified by a supervising operator from an OWS;

– Parameters: information modified at the engineering workstation (EWS).

Similarly, an object can produce different outputs:

– Status: the object sends a status report to the data server (DS) for the purposes of supervision, or to other objects;

– Order: the object sends an order via Auto Request to other objects or to the process in the field.

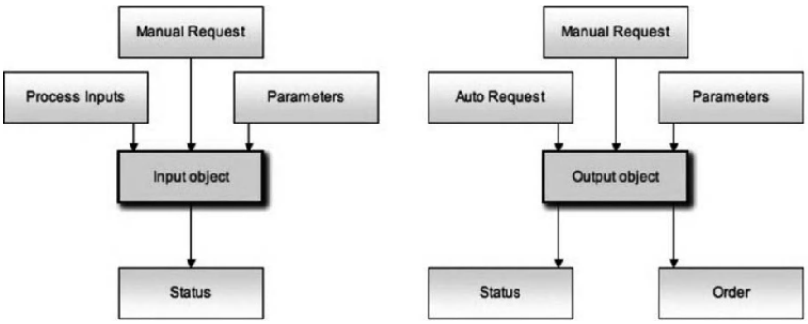


Figure 7.4. *Architecture of access to the IO objects*

7.1.2.2.4. Field Objects

As we have seen previously, there are four types of Field Object: Local, OnOff, Analog and Anadig. In structural terms, the instances all have a common basis (Standard Logic Device) and a part specific to each object depending on its function (Interlock Logic), which will be programmed by the developer. Interlock Logic corresponds to the logic of interruption of the objects. It can place the object in a safety position (*Stop Interlock*) or prevent it from starting up (*Start Interlock*) from other objects. The field objects are controlled by a PCO or a PID controller, and send orders to the Analog and Digital Outputs.

In the case of field objects, the Process Inputs correspond to statuses reported by other objects, on the basis of which the Interlock block formulates its logic. As shown by the diagram in Figure 7.5, the Interlock Logic acts directly on the process and can interrupt or block it.

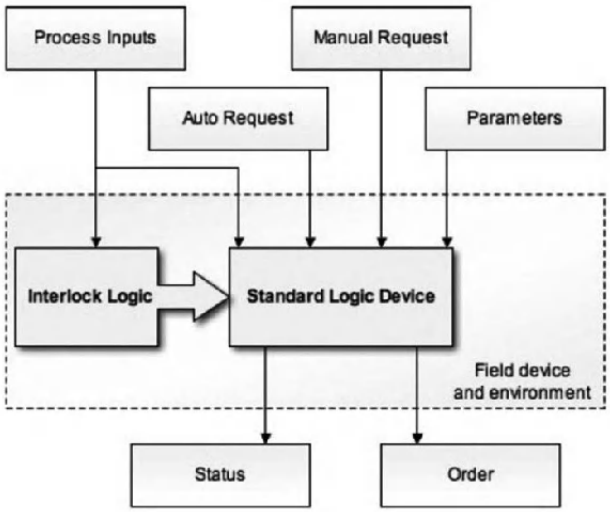


Figure 7.5. *Architecture of the field objects*

Order outputs are transmitted via Auto Request from the AO/DO objects, and the status is sent to the DS for supervision purposes.

7.1.2.2.5. Control Objects

The *Control Objects* are in charge of the whole of the installation. On the basis of the status of the objects or orders received, they act on the field objects or on other *control objects*. The *Controller Objects* contain a regulation algorithm to control different objects. In general, they represent a conventional PID. They can control up to four objects as well as two objects in *Split Range*.

The *Alarm Objects* do not give orders, but are able to transmit an alarm signal to the supervision center. This signal means that the process is not behaving as it normally does – e.g. when an *Interlock* is in force (*Start* or *Stop Interlock*). It helps determine the cause of that interlock out of a number of possible causes.

Process Control Objects are modules which are able to generate orders and send them to multiple field objects or to multiple other process control objects, homogeneously.

These modules contain a common part and logic blocks which are programmable on a case-by-case basis, depending on the application. The block *Interlock Logic* includes the programming of temporary interruptions (*Temporary Interlock*) or definitive stops (*Stop Interlock*) and start preventions (*Start Interlock*). The block *Configuration Logic* determines that states of the objects. Finally, the block *Specific Process Device Logic* contains the control algorithms specific to the process. It is this part which controls all the dependent objects. Integrated in the form of a GrafCet, sequential executions coordinate between the different dependent objects and determine the orders of start/stop.

Objects other than the input/output objects can only be controlled by a single PCO. Only the master PCOs (at the top of the hierarchy, see Figure 7.6) are worked by the operator.

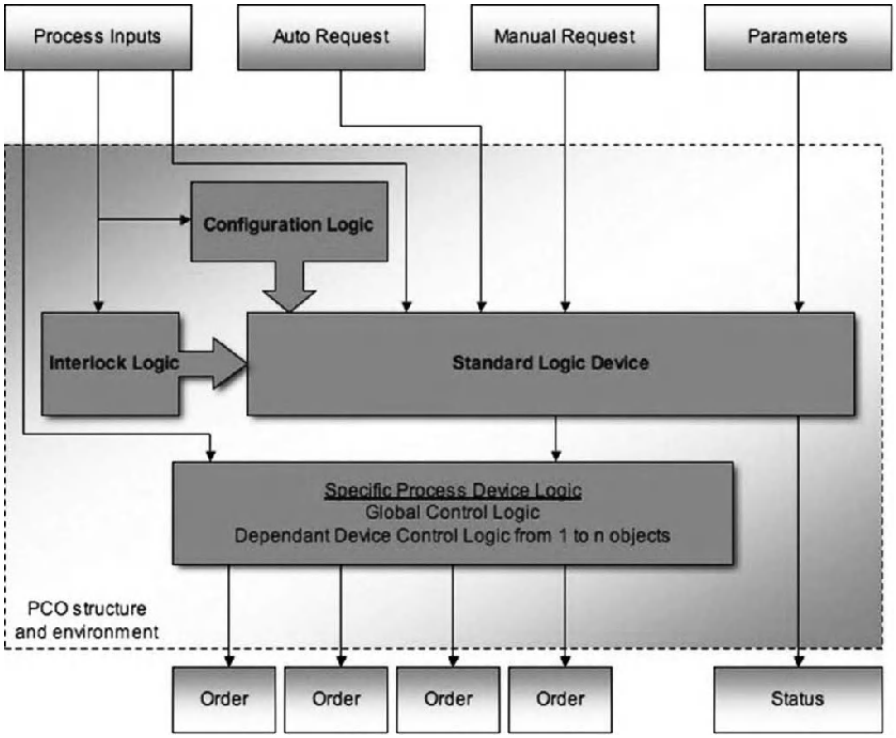


Figure 7.6. *Architecture of Process Control Objects*

The preliminary work during the development of the control system is of capital importance, because we need to clearly define the list of PCOs, their hierarchy and the dependencies of the field objects associated therewith.

7.1.3. *MultiController*

MultiController is an object developed as part of the CERN/GCS project to introduce advanced algorithms to regulate the installations of the gas control system (GCS). MultiController is an additional object with embedded advanced control functionality which was produced for the CERN/GCS framework. It can be used in the process of automatic generation of applications by way of the "Model Driven Approach" explained in [CAB 07; OBJ 01; OBJ 03; THO 05]. It is a concrete application of the introduction of advanced controls into a large-scale automation engineering project, with new methodologies stemming from industrial computing. MultiController ensures the integration of the algorithms by establishing a virtual object link between the SCADA, the PLC and the installation.

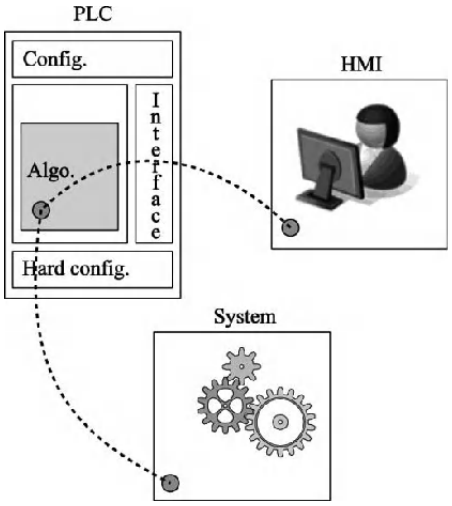


Figure 7.7. *MultiController: a means of introducing advanced algorithms by establishing the link between the various organs of industrial automation engineering*

7.1.3.1. *Principles and functions of MultiController*

The realization of MultiController is based on operational principles which are consistent with the advanced control algorithms and which take account of the technological constraints of the project methodologies.

In advance of the creation of the object, an overall reflection was necessary in the interests of client satisfaction. In conjunction with the actors involved, the requirements were translated and the design of the object established as a result. The declared objective was to provide functionalities which are a compromise between expertise and ease of use.

7.1.3.1.1. Client requirements

As it is to be integrated into the UNICOS framework, MultiController has to respect certain specific requirements. These are listed below:

– the object adheres as closely as possible to the UNICOS standard. This means that the formats, functions and interfaces have to correspond as closely as possible with what is already defined in the other basic UNICOS objects;

– the output and the set value can be limited;

– an internal mechanism can provide a set value which increases with a specified gradient;

– the object exhibits "tracking" behavior, as defined in UNICOS. When the system is in regulation mode, the tracking function can interrupt the regulation by applying a predetermined command (conditions and values supplied by the client);

– MultiController can perform scaling of the measured value, the controller output and the set value.

7.1.3.1.2. Design choice

A model of MultiController was put forward which would best satisfy the needs of the programmers, the experts and the users, and fundamental design choices were put in place. These design choices offer ease of use and considerable advantages to all those involved in the development and use of the object:

– MultiController uses the notion of "modes" as defined in UNICOS. These modes are subdivided as follows: automatic mode, manual mode and forced manual mode;

– MultiController employs the notion of "behavior" for the regulatory operation and positioning of the object;

– the management of modes and behaviors is different. We can work in regulation mode or in position in the three UNICOS modes;

– MultiController has a PID in order to satisfy all the users;

– the object offers many advanced control algorithms in a single entity;

– it is possible to add other controllers without changing the interface, thanks to a generic DFB structure. This possibility facilitates a substantial gain in terms of time and transparency;

– there is a "recipe" mechanism whereby we can save the relevant parameters of the regulation so as to reuse them if need be. This operation is performed from the HMI console (PVSS II);

– the parameters of the object may be gleaned from an implementation with or without a recipe.

7.1.3.1.3. Functionalities

About the advanced regulations in the object

Using MultiController, there is the possibility of selecting the regulation algorithm used by the controller. The choice can be made in the PLC's program or via the dedicated interface in the HMI. Thus, the operation has an appreciable degree of freedom as regards the putting in place and change of an advanced regulation algorithm.

The construction of the object is done with a unique DFB interface in the PLC, with MultiController imbued with all the regulations developed for the Schneider PLC. There is the choice of a structure of generic parameters (called "Param" in the object) in order to facilitate a monolithic approach to the object. The way in which the parameters of the structure are dealt with depends on the selection of the regulation. Thus, the same parameter can be used in different ways depending on which algorithm is active in the controller. Consequently, the structure of MultiController means that further advanced commands can be added without changing the DFB interface. The addition of algorithms is thus more effective in the process of development and evolution of the object.

About the modes and behaviors in MultiController

MultiController works with three operating modes (UNICOS standard):

– automatic mode, in which the program has control over the object;

– manual mode, in which the user (via the HMI) takes control of the object. It is also possible for the system to switch from manual to automatic mode by way of an AuAuMor request (loss of manual mode);

– forced manual mode, in which the user assumes total control of the object (forced mode can never be exited without express intervention by the user).

MultiController has two different behaviors: *regulation* and *positioning*. The operation of the modes with that of the behaviors is fundamentally different. Regulation means that the object works with a type of regulation. Positioning is a direct affectation of the controller output when regulation is no longer desired. This is similar to the dichotomy between open-loop and closed-loop function. Figure 7.9, below, gives an overview of the behaviors and their interactions with the modes.

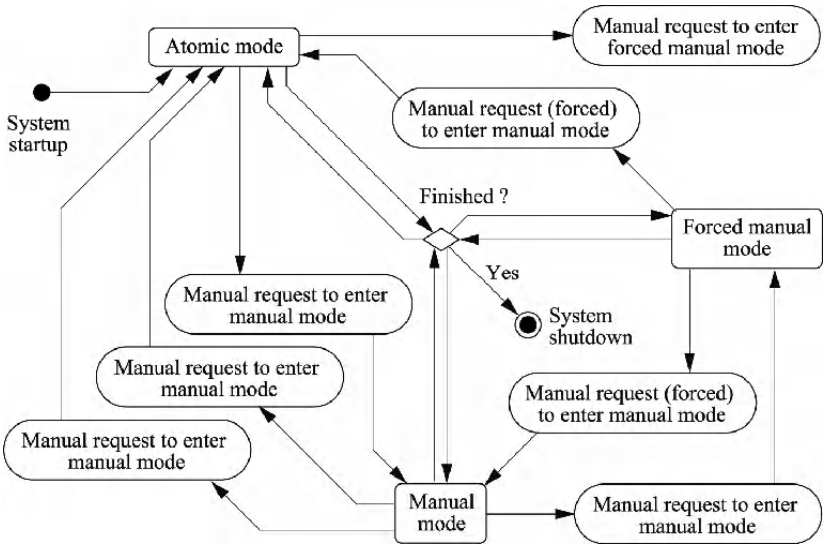


Figure 7.8. *The modes in MultiController and their operation*

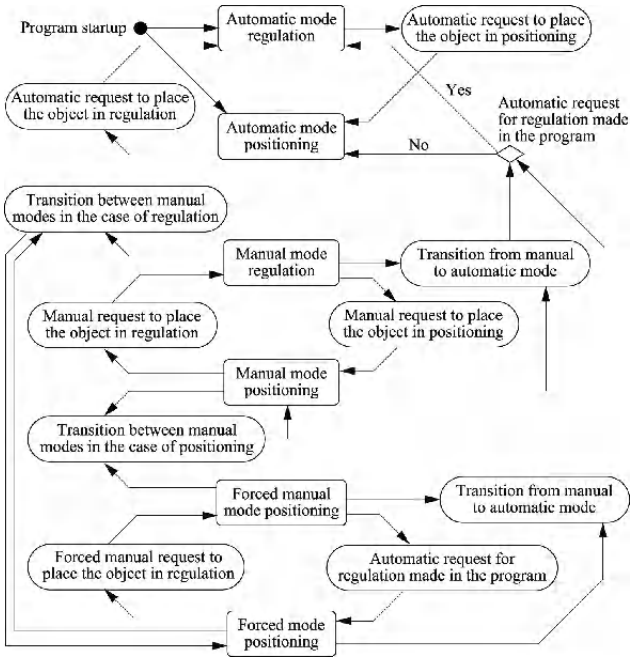


Figure 7.9. *The behaviors in MultiController and their operation*

A change of mode does not mean a change of behavior, except in the case of specific relative transitions (e.g. in automatic mode, the automatic requests and parameters are considered as a matter of priority).

About the tracking function

Tracking (Figure 7.10) is a particular function of MultiController, developed in the wake of the UNICOS standard. It is a technique which enables the program to take control of the controller output during regulation, regardless of which mode is active. Thus, in very specific conditions, it is possible to "track" the object being controlled and thus prevent restarts outside of the zone of regulatory function.

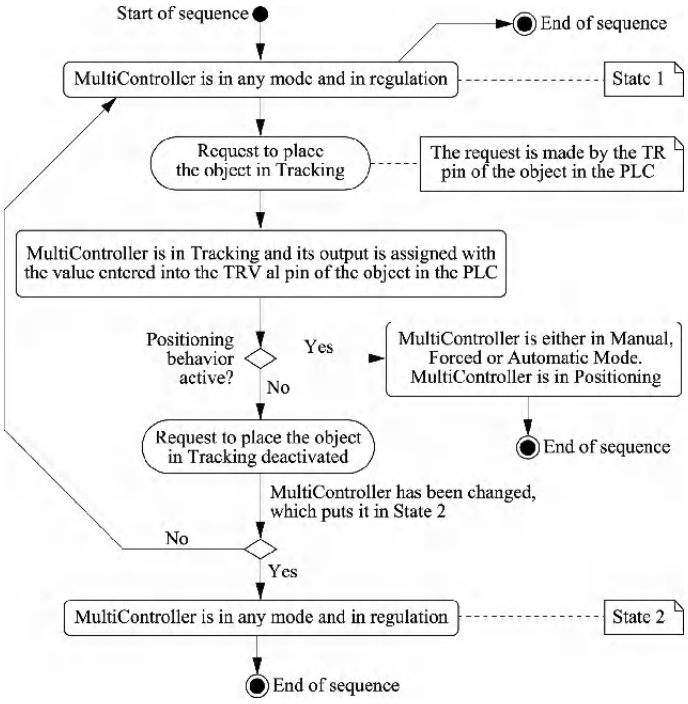


Figure 7.10. *The tracking function in MultiController*

7.1.3.1.4. Design of MultiController

Structure and organization of the PLC code in the derived functional block (DFB)

The internal structure of the PLC code (Figure 7.11) introduced is divided in such a way as to favor ease of development. The basic idea of the internal

organization is to create "zones" dedicated to each function. The division is as follows:

– a zone for the putting in place of the modes, behaviors and limitations and determination of the status and values of the object;

– a zone which assigns the variables to the proper regulation;

– zones which are reserved for advanced algorithms;

– a zone which takes care of the assignment of the outputs to the proper regulation;

– a final zone for the assignment of certain registries which could not be done in the previous zones.

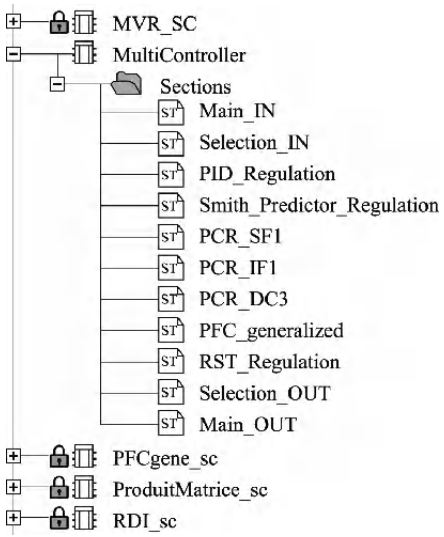


Figure 7.11. *Organization of the Unity code in MultiController for Schneider PLCs*

The second advantageous characteristic of the PLC code of MultiController lies in its capacity to encapsulate other objects. Because of this highly interesting function, the regulation algorithms developed previously are encapsulated in MultiController. Certain protected objects made by Schneider (PCR library) have also been added.

The principle is simple, and is based on the reuse in a DFB of other previously-developed DFBs. Thus, an advanced control object can be made "outside of" MultiController but can be called and used by it.

This principle of encapsulation is an effective solution for working with "pure regulation" codes without affecting the functions of the object. In addition, this enables us to lock certain complex commands in the source code in order to avoid any unwanted manipulations of these commands.

Figure 7.12 shows MultiController with the regulatory DFBs encapsulated in the structure of the object.

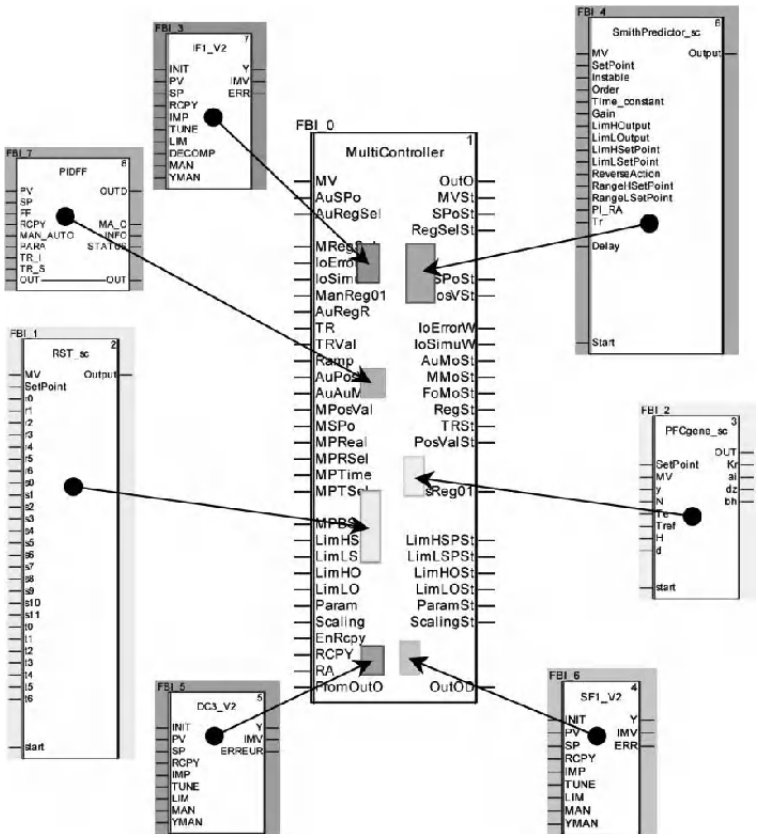


Figure 7.12. *Principle of encapsulation of regulatory algorithms in MultiController in Unity*

MultiController has the following algorithms:

– PID – Classic Unity library;

– Smith Predictor for stable systems (first- and second-order, double pole) and integrator systems – the SmithPredictor_sc object developed;

– predictive functional control for first-order systems – the SF1_v2 object from Schneider's PCR library;

– predictive functional control for second- and third-order systems – the DC3_v2 object from Schneider's PCR library;

– predictive functional control for integrator systems – the IF1_v2 object from Schneider's PCR library;

– generalized predictive functional control – the PFCgene_sc object developed;

– RST controller – the RST_sc object developed (this object can also be used with the external GPC_sc object for GPC control).

The user interface associated with MultiController in the PVSS SCADA satisfies a number of objectives:

– to give the user an interface with the same principles as those behind the objects in the UNICOS framework. Thus, MultiController is not perceived as a complex function but rather as an additional UNICOS object;

– to enable the gas process engineer to use regulation algorithms other than the PID – to choose them, try them out and quantify their effectiveness;

– to give simple mechanisms for use of the regulations during operation;

– to ensure a notion of "testing" and "tuning" thanks to recipes and the ability to save them;

– to familiarize the users of the object with the regulatory techniques and the methodologies for using them.

The PVSS views of the HMI representation of MultiController are of three orders, as shown in Figure 7.13.

The main view (Status) relates to the general state of the object with its modes and behaviors. It shows the state of the set values and the measured values as well as the limits. Finally, it offers a visualization of the active regulation with the corresponding parameters. For instance, in Figure 7.13, we can see that the regulation type selected is a Smith predictor. The automatic set point is zero and the manual set point is 67. The object is in manual mode and positioning behavior. The regulatory parameters are 5.1 seconds for the time constant, a time latency of 2.56 seconds, a gain of 2.1 and a first-order system with a reverse action for the regulator.

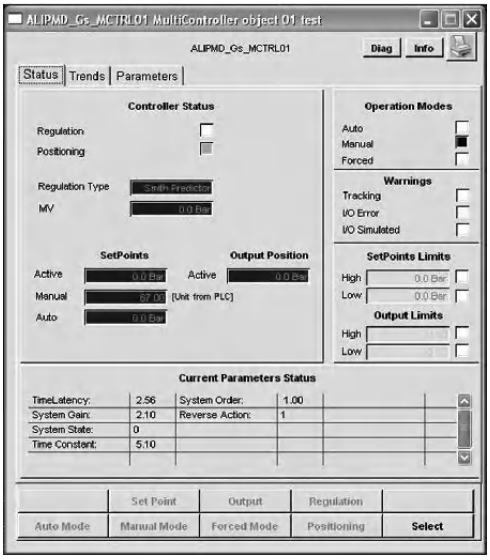


Figure 7.13. *Main view (Status) of the interface of MultiController in the PVSS SCADA*

Figure 7.14 shows the PVSS view of the trend curves. This tab enables us to see the measured value, the set point and the control of the regulator. This view is a UNICOS standard developed with the classic PVSS functions. The trend curves can be configured in terms of time, scale and view.

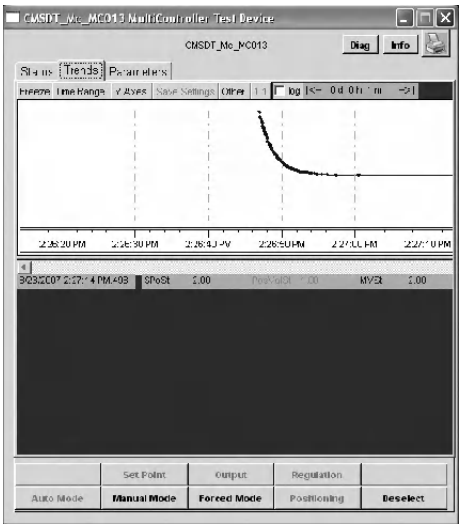


Figure 7.14. *View of the trend curves in the interface of MultiController in the PVSS SCADA*

The final view (Figure 7.15) represents the regulations available in the object. Each tab (the "Smith Predictor" tab in the example below) can be used to select the desired regulation and to change the online parameters (only in manual mode).

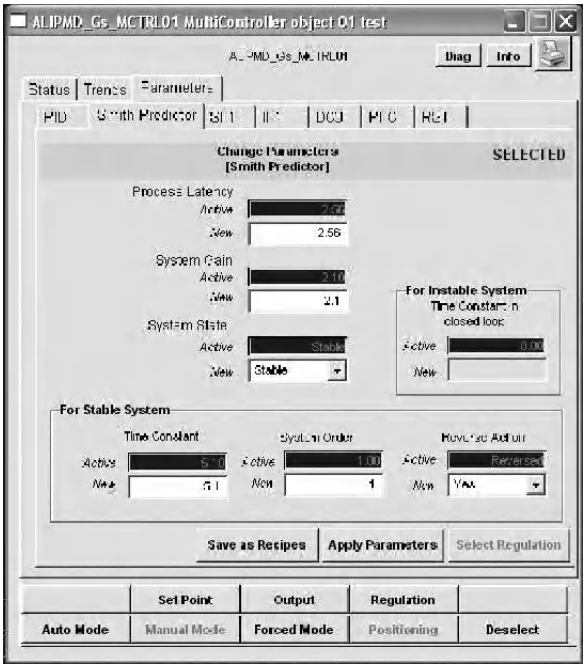


Figure 7.15. *View of the parameters pane in the interface of MultiController in the PVSS SCADA*

Each regulation tab has a function for saving the manual parameters as a recipe (see the "*Save as Recipe*" button in Figure 7.14). The recipes created can then be loading into MultiController to be applied in automatic mode.

7.1.3.2. *Use of MultiController in the PLC*

In this part, the use of MultiController in the PLC Premium with the Unity software is discussed.

The FBD language is a simple tool to instantiate a MultiController. It is possible, therefore, to track the changes in the variables online, view the data path and gain a global view of the regulation process.

FBD is characterized by graphing programming. The instances are directly identifiable and the relations between them are clearly visible.

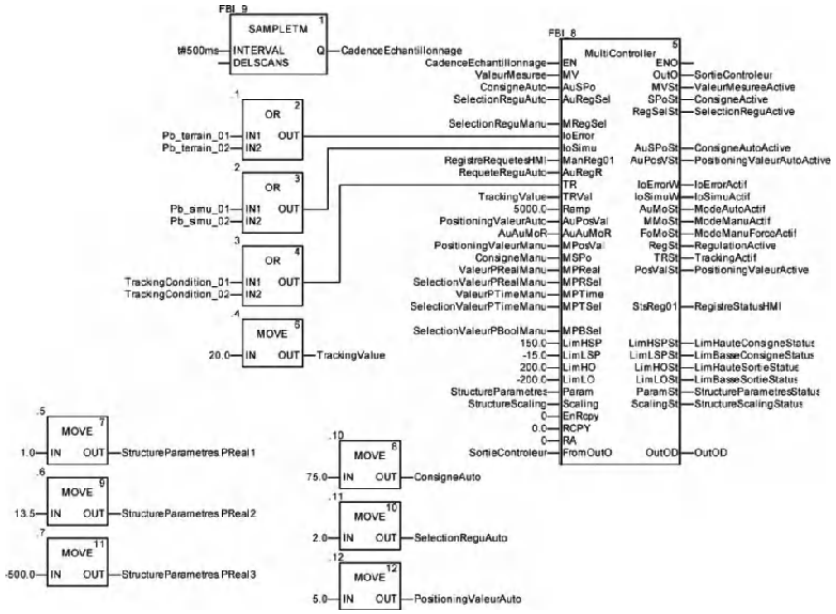


Figure 7.16. *Simple programming in FBD (Unity) of a regulation with MultiController*

The example of FBD programming presented in Figure 7.16 demonstrates the interactions and settings. Because of this graphic programming method, it is easier to read the online diagnostics. Indeed, when the PLC is connected to the program's Unity console, the values are animated in real time.

There may be limitations to this programming. When the strategies become more complex (startups, transitions between two states, etc.), it is more difficult to implement them by using functional blocks. Structured language can then take up the baton to facilitate implementation.

7.1.3.2.1. Evolved applications of MultiController

Today, MultiController is operational. Users are allowed to work with nearly eight different regulatory algorithms.

The automation engineering toolbox in the Schneider PLC can be used to create solutions which are more elaborate than control loops. This offers us a glimpse of the possibility of additional functions for MultiController users. For the moment, the DFB objects for modeling/identification are not integrated as such into the GCS framework. They are not represented in the HMI and are therefore not "controllable" in that sense. However, they can be used on occasion with MultiController in the PLC.

The following example shows a use of MultiController to put in place an adaptive regulation with GPC.

GPC adaptive regulation

Consider the simulated system $H(p)$ to be corrected:

$$H(p) = \frac{2}{1+10p}$$

The adaptive control employed by MultiController uses the object GPC_sc (generalized predictive control) and the object MCR_sc (recursive least squares) for online determination of the model.

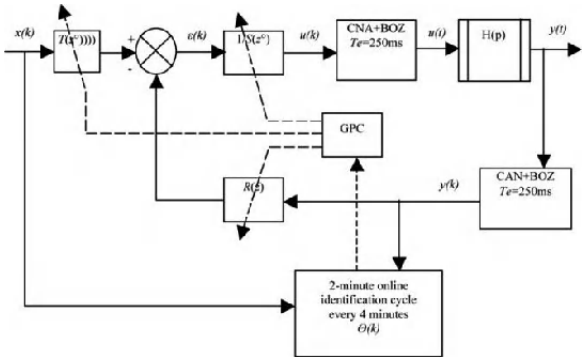


Figure 7.17. *Example of the use of MultiController: adaptive control in the Schneider PLC – Principle*

The principle of adaptive control illustrated in this example is simple:

– MultiController is used with the RST controller. The object GPC_sc is directly connected to the structure of the parameters. Adaptation is taken care of by the object MCR_sc, which performs online identification of the model of the process needing to be corrected. The block SBPAfix (PRBS) is used to generate sufficient value steps to facilitate appropriate identification. For reasons of convergence and initialization of the recursive least squares algorithm in the block, identification is carried out cyclically every four minutes for two minutes;

– the parameters of generalized predictive control are:

- $N_1 = 1$, as there is no latency,
- $N_2 = 5$ so as to have a sufficient horizon,
- $N_u = 1$ (default value),

- λ ~0.013, which is the optimal online computation,

- $\lambda_1 = 0.999$ and $\lambda_2 = 1$ so as to have a slight forgetting factor and a decreasing gain,

- $F_0 = 10.0$ so as to have an identification which converges as desired.

Figure 7.18 shows the code run on the Schneider PLC for this adaptive control.

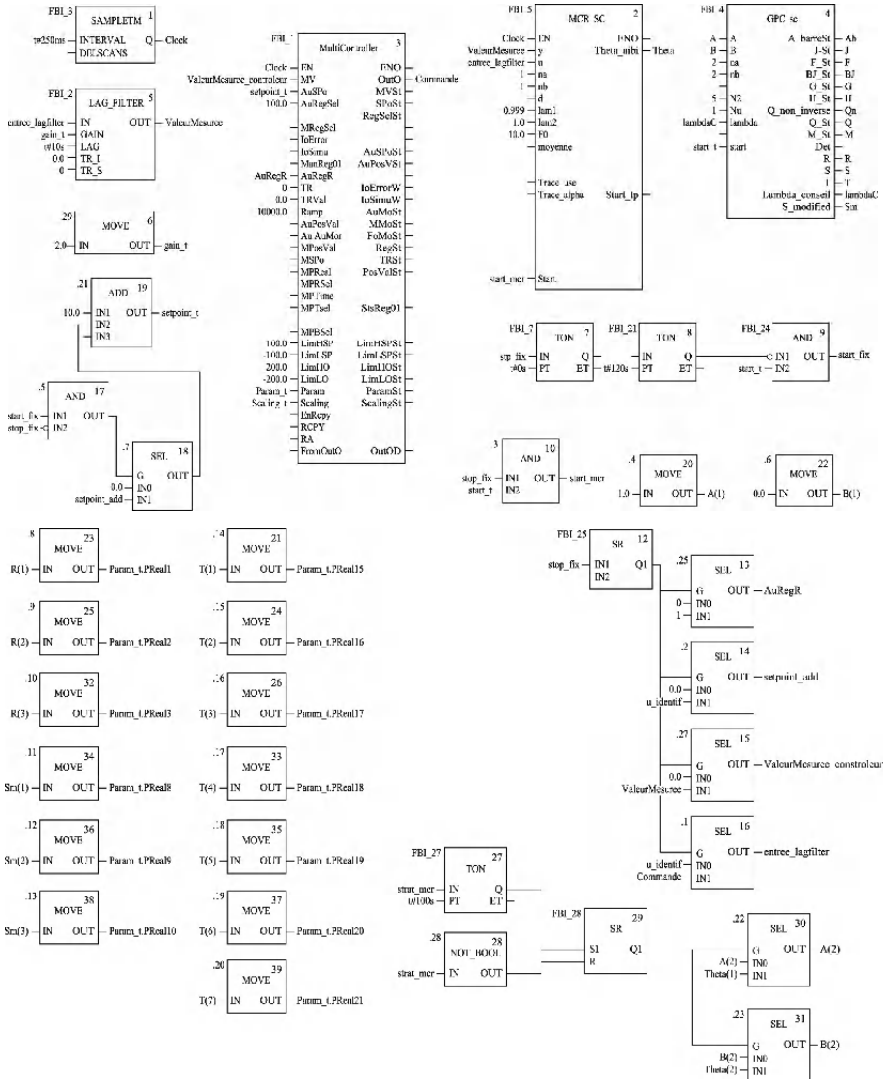


Figure 7.18. *Use of MultiController: adaptive control in the Schneider PLC – PLC program*

Upon startup, the system performs a two-minute cycle of parametric identification before starting regulation (the block SBPAfix_sc generates value steps in an open-loop regime). When the regulation process is begun, adaptive control takes place in a closed-loop regime every four minutes. The principle is that the control system works on the set value to produce sufficient variations (5% variation with the same block SBPAfix_sc).

DFB programming is more complex. However, it does enable us to visualize the variables of the online system more clearly. In this example, we catch a glimpse of the various links between MultiController and the blocks in the automation engineering toolbox provided with the Schneider PLC.

The crux of the difficulty of the program put in place is to organize the functional blocks properly. The code needs to be sufficiently well-spaced out, with few direct connections. Conversely, the blocks have to be subtly organized so the screen shows the pertinent connected values at the same time. This adaptive control is an example of this issue.

We obtain the results shown in Figure 7.19 on the output and the control.

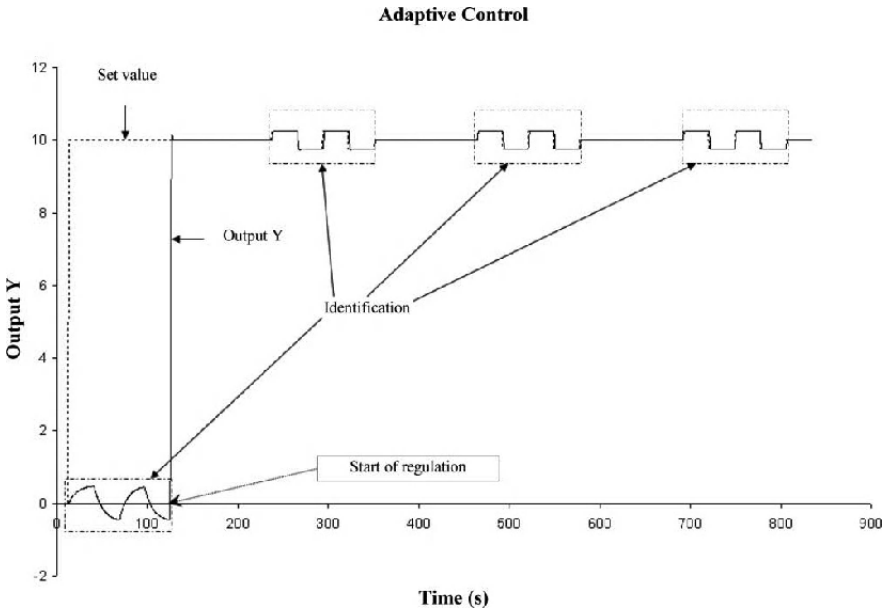


Figure 7.19. *Example of the use of MultiController: adaptive control in the Schneider PLC – system output*

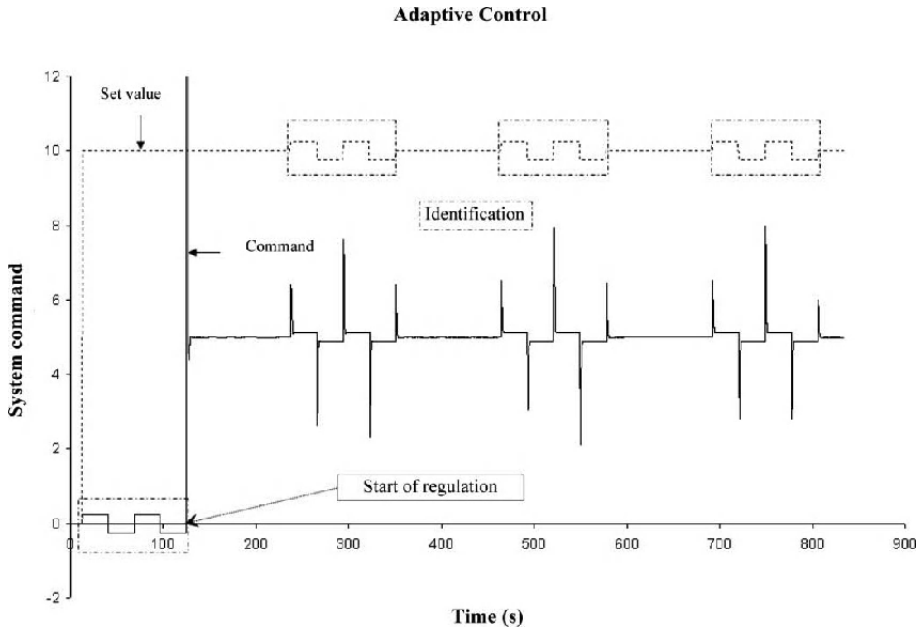


Figure 7.20. *Example of the use of MultiController: adaptive control in the Schneider PLC – system command*

Figures 7.19 and 7.20 show that the regulator output ensures adequate correction after the initial identification phase. Indeed, the parameters set by the GPC algorithm appear to offer good correction of the system. Finally, the successive identifications are performed perfectly in a closed-loop regime. This example illustrates the future possibilities offered to the users of the control systems on the GCS project. Integration into the framework and development guided by the models in the automation engineering toolbox provided with the Schneider PLC would likely be the next step. Such work would perfectly complement the advances made with MultiController.

7.2. A distributed architecture for control (rapidity/reliability): excavator-loader testing array

7.2.1. *Objectives of the testing array*

This study is part of the project “driving support and critical situation detection for smart building machines”. It consists of designing the real-time computer architecture for the onboard control/command system based on a communication

bus. This study is performed in collaboration with the company OMRON (industrial automation engineering, components, sensors and PLCs).

The main characteristics of the project are:

– distributed architecture built around sensors, actuators, control joysticks and the human–machine interface;

– the time to access the measurements needs to be guaranteed at around 30 ms;

– the computation needs to be less than 10 ms;

– reliability of the data communicated on the bus.

7.2.2. *Presentation of the onboard computer platform*

The aim of this platform is to have available a testing array in order to implement the equations relating to the operation of the excavator-loader.

On this testing array, we find various components:

– ONC: this is an industrial computer upon which we can implement the various equations. This computer functions on the QNX operating system. Thanks to this operating system, we can run the programs in real time:

- it includes DeviceNet connectivity as standard and therefore establishes a distributed connection between all our deported elements (sensors, jacks, etc.),

- it has no hard disc and can be linked to any OMRON or DeviceNet component without a single line of programming. The system will have the reliability of the network DeviceNet,

- it is able to deport all kinds of inputs/outputs (On/Off, analog, PWM),

- it supports QNX 4.25, a real-time multitask operating system, capable of executing programs in C++;

– logical input block: this block serves to acquire logical information from push-buttons, contacts, etc.;

– analog input: this block enables us to measure variables with continuous voltage or current;

– logical outputs: this block is able to control low-power pre-actuators: contactors, relays, etc.;

– angular encoder (TRS): this is capable of providing the absolute angular position;

– PLC: the PLC is capable of providing PWM chopper control signals on its outputs in order to control a solenoid valve. This serves to regulate the pressure flowrate in the stems of the jacks for each joint in the articulated arm.

All the components seen above are linked by a CAN bus in order to communicate. The communication access method is a master (ONC)/slave (E/S block, TRS, etc.) type arrangement. In addition, we have added an induction motor piloted by a variator in order to simulate the variable rotation of the TRS angular sensor.

The general structure of the platform is presented in Figure 7.21.

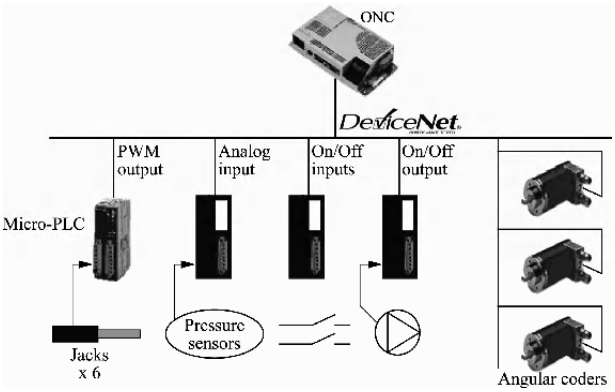


Figure 7.21. *Structure of the onboard system*

Figure 7.22. *Presentation of the testing array*

7.2.3. *Examination of the rapidity of the onboard computer structure*

7.2.3.1. *General configuration of the DeviceNet network*

The parameters to be regulated are as follows:

– *access method*: this is the method which resolves the conflict that arises when several modules are simultaneously sending information to the bus. The method chosen is a master-slave where the ONC is the master and all the other modules are slaves. The master is responsible for all the transmissions. This means that no input module will be able to send information unless the ONC has previously requested it. Thus, no output module is able to request information from the ONC, but rather it is the ONC which decides to supply the other modules with that information. A slave cannot send information to another slave directly, but has to go through the master;

– *transmission rate*: all the modules have to use the same transmission rate in order for the network to function correctly. In the system proposed here, the transmission rate is 500 Kb/s, which is the highest rate of which the modules are capable;

– *node number*: each module used in the network has its own node number which identifies it within the network (see Table 7.1);

– all the node numbers have to be different.

The analog and digital input/output blocks are of type DRT2. This type of slave module automatically detects the transmission rate of the bus. Hereafter, the angular sensor and the PLC will also be configured at 500 Kb/s.

Node	**Module**	**Size of input** Words	**Size of output** Words
1	PLC	8	8
2	Analog input	4	-
3	On/Off input	1	-
4	On/Off output	-	1
5	Angular encoder	2	1
6	Analog output	-	2
TOTAL		**15**	**12**

Table 7.1. *Size of the datapackets from the slaves in DeviceNet*

7.2.3.2. *ONC's communication method over the DeviceNet network*

The ONC communicates with all the other modules over the DeviceNet network by the following method.

Remote I/O communications (see Figure 7.21) associate the input/output data with their event memory and read/write to that memory. The areas of memory associated with the modules are regularly refreshed. Meanwhile, the ONC sends data to each output module and receives data from each input module. Thus, the ONC has to communicate with all the modules before beginning another communication.

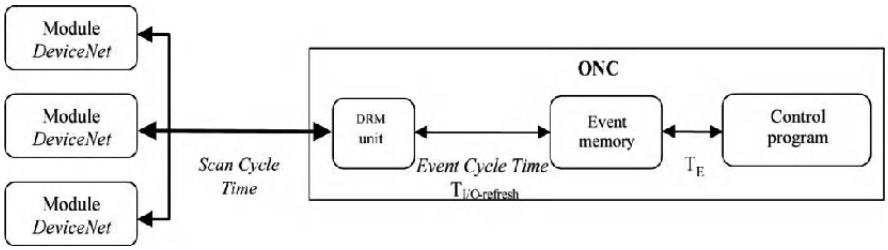


Figure 7.23. *Remote I/O communications*

7.2.3.3. *Configuration of the DRM unit in the ONC*

The ONC has to be configured to optimize the management of the DeviceNet communications. Thus, we need to configure its DRM unit, the software component of the ONC which acts as a communication unit and which enables us to connect to DeviceNet.

The DRM unit provides us with two services: DRM0 and DRM1. These two services are capable of working as two modules independently of DeviceNet. We use only DRM0. The parameters of its configuration screen are as follows:

– parameters:

- transmission rate: 500 Kb/s,

- *Master*: ONC with scanlist;

– timers: the timers are asynchronous and control communications between the modules in Figure 7.21.

Scan cycle time (SCT): the period during white the DRM unit sends output data and receives all the input data from the slaves. This communication lasts for a variable TRM time. The minimum value of the SCT is limited by the TRM time. In

this case, the DRM unit would communicate incessantly with the slave modules. We can read the value used in the memory area *DeviceNet status area* in the ONC.

Event cycle time (ECT): the period during which the input and output information is copied from the DRM unit to the event memory. This operation lasts for a fixed time period $T_{I/Orefresh}$. We can specify the ECT to a precision of 1 ms, and reducing its value decreases the global yield of the ONC. In our application, the computing power of the CPU in the ONC is distributed between computing control algorithms and managing communications. Hence, if we decrease the ECT, the maximum time to compute the control algorithms will increase.

Scan Mode: the *ScanList file* mode is chosen, otherwise the value of the ECT is automatically set too high (50 ms).

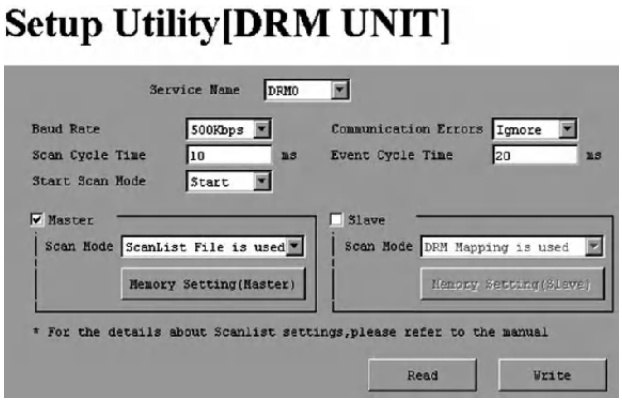


Figure 7.24. *Configuration of the ONC: DRM0 unit*

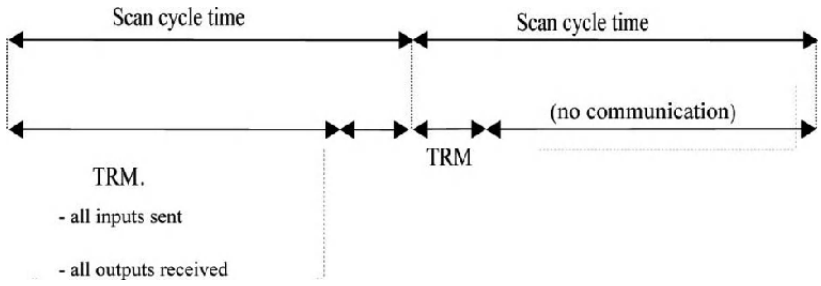


Figure 7.25. *Relation between scan cycle time and TRM*

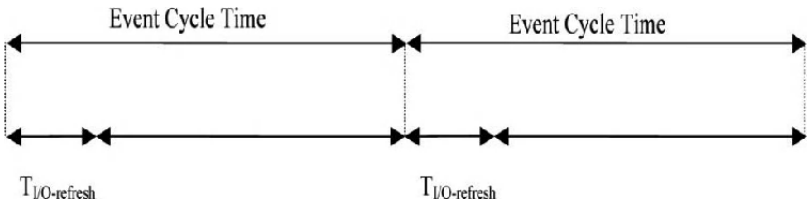


Figure 7.26. *Relation between event cycle time and* $T_{I/O\ refresh}$

7.2.3.4. *Communication cycle time (TRM)*

In order to discover the speed of the system, it is necessary to exactly calculate the maximum communication time (TRM_{max}), i.e. the maximum time necessary for the data in the DRM unit to be sent to each output slave and for the data to be received from each input slave.

In this section, we calculate the theoretical TRM and the maximum TRM experimentally for the testing array. By comparing these figures we will be able to validate the theoretical equations of the TRM. Then, the theoretical TRM of the final system on board the excavator-loader is calculated.

Module	**Time (ms)**
PLC	0.914
Analog input	0.298
On/Off input	0.202
On/off output	0.322
Angular encoder	0.386
Analog output	0.354
HiDensity	3.5
ExplicitMessage	0
COSCyclic	0
(0.01*N) + 1	1.06
Typical TRM	**7.036**

Table 7.2. *Theoretical TRM of the testing array*

In addition, we can calculate $t_{I/O\_refresh}$ theoretically with the following equation:

$$t_{I/O\_refresh} = 0.7 + (0.001 * (\text{Words IN} + \text{Words OUT})) \text{ [ms]} \quad [7.1]$$

Using Table 7.2:

$$t_{I/O\_refresh} = 0.727 \text{ ms} \quad [7.2]$$

7.2.3.5. *Experimental TRM*

We carry out communication between the ONC and the various peripheral devices equipped with a DeviceNet interface by using the functions developed in C language. We read and write the content of the event memory addresses assigned to the slaves. These functions of the ONC of communication with the event memory are the optimized version of the functions initially developed by OMRON. The greatest single modification that has been made is to initialize the type of memory used only once.

The ONC is able to read the maximum TRM with a memory area *DeviceNet Status Area* which is found in the event memory. In order to read it, we can use the same functions as before, accessing the address configured in the menu "*DRM0 mapping*": CIO 1500. The offset which holds the maximum value of the TRM in the DeviceNet Status Area is 88. Hence, we read the memory address CIO 1588 to obtain the *experimental TRM of the testing array*, which is 7 ± 0.5 ms. This value is in line with the theoretical value (see Table 7.3).

In order to compare the theoretical and experimental values, numerous tests are carried out, where we have disconnected a number of the DeviceNet slaves (see Table 7.4). These tests demonstrated the influence of each module on the overall communication time, and this confirmed that the TRM depends on the number and type of slaves connected. Furthermore, we note that half of the TRM is used in communication with the PLC.

In conclusion, the experimental maximum values of the TRM are similar to the theoretical maximum values, which means we can use the theoretical formulae to evaluate the maximum communication time:

$$TRM_{\text{typical}} \approx TRM_{\max} \quad [7.3]$$

Connected nodes						TRM (ms)	
1	**2**	**3**	**4**	**5**	**6**	**Theoretical** (typical)	**Experimental** (maximum) ± 0.5 ms
				✓	✓	1.760	2
	✓			✓	✓	2.068	2
	✓	✓		✓	✓	2.280	2
	✓	✓	✓	✓	✓	2.612	2
✓				✓	✓	6.184	5
✓			✓	✓	✓	6.516	6
✓		✓	✓	✓	✓	6.728	6
✓	✓	✓	✓		✓	6.640	6
✓		✓	✓		✓	6.332	6
✓			✓		✓	6.120	5
✓			✓			5.756	5
✓						5.424	5

Table 7.3. *Communication time depending on the number of connected modules*

Scan cycle time	***Event cycle time***	***Tchan***	**Number of repeated sample ($T_{e\ mem}$ = ECT)**
10	20	[19.75; 22.25]	21
0	20	[19.75; 22.25]	21
15	20	[19.75; 22.25]	21
15	30	[29.75; 32.25]	32
5	10	[10.75; 12.25]	11
11	22	[21.75; 24.25]	24

Table 7.4. *Repeated samples*

7.2.3.6. *TRM of the final onboard system*

The communication time of the onboard system which will be installed on the excavator-loader will be different, because other modules will be connected to DeviceNet. In total, there will be 11 slaves – namely:

– 1 digital input module;

– 2 modules of four analog inputs, as there are six pressure sensors;

– 6 angular sensors;

– 1 PLC;

– 1 digital output module.

With these DeviceNet modules connected to the ONC, the final maximum TRM is:

$$TRM_{\text{final\_max}} \approx TRM_{\text{final\_typical}} = 8.960 \text{ ms} \quad [7.4]$$

The maximum TRM found enables us to calculate the minimum sampling frequency.

7.2.3.7. *Minimum sampling frequency*

In this section, the minimum sampling frequency ($T_{e\ min}$) for the proposed onboard system is examined with the aim of choosing the best in view of all the constraints to which the system is subject. This time will be the sampling frequency of the control algorithm.

Firstly, the theoretical $T_{e\ min}$ is calculated, and then its value is verified experimentally. Finally, we find the optimum T_e and deduce from it the margin of computation time for the algorithms.

7.2.3.7.1. *Theoretical* $T_{e\ min}$

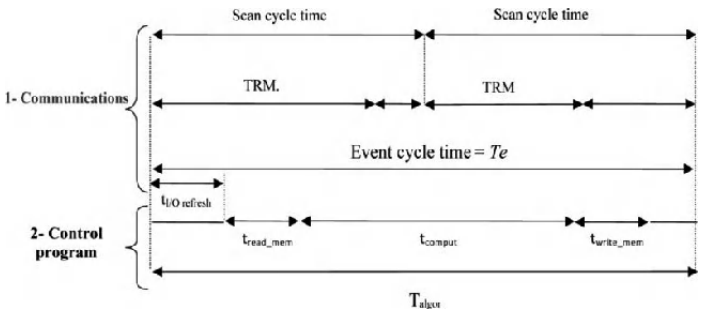


Figure 7.27. *Chronogram of the ONC – DeviceNet protocol*

As the two timers SCT and ECT are asynchronous, the ECT must be a multiple of the SCT in order to synchronize. The chosen value for the ECT is double that of the SCT. In the case, the ECT is the maximum refresh time for a cycle, which must be the minimum sampling frequency:

$$ECT = 2 \cdot SCT \quad \Rightarrow \quad T_{e\min} = ECT_{\min} \qquad [7.5]$$

As we saw previously, the SCT is linked to the TRM by:

$$SCT_{\min} = TRM_{\max} \qquad [7.6]$$

There are two processes which are executed in parallel and independently, which we have to synchronize in order for the information to be transmitted as quickly as possible. These two processes are:

– DeviceNet communication, which transmits data between all the connected slaves and the event memory of the ONC;

– the control program which is executed in the central processing unit of the ONC. It receives and sends the information to the event memory.

The control program must periodically carry out the following tasks:

– wait until the input data arrive in the event memory ($t_{I/O\ refresh}$);

– read the input data from the event memory (t_{read_mem});

– compute the control algorithms (t_{comput});

– write the results into the event memory (t_{write_mem});

– wait for a safety period (δ), to ensure that the results have been written into the memory before being sent to the output modules, before beginning the next cycle.

Hence, the period of execution of the control algorithms (T_{algor}) must be:

$$T_{\text{algor}} > t_{I/Orefresh} + t_{read\_mem} + t_{comput} + t_{write\_mem} \qquad [7.7]$$

By synchronizing the control program with the communications:

$$T_{\text{algor}} = ECT = T_e \qquad [7.8]$$

$T_{e\ min}$ is limited by the communication time or by the computation time for the algorithms:

$$T_{e\min} = \max(2 \cdot SCT_{\min}, T_{\text{algor}\min}) \quad [7.9]$$

In the previous sections, we saw that the TRM_{max} is less than 10 ms, both for the testing platform and for the final system. Hence, in order to ensure that conditions [7.7] and [7.8] are satisfied, we choose SCT = 10 ms and ECT = 20 ms.

Thus, the theoretical sampling frequency is:

$$T_{e\ \text{theoretical}} = 20 \text{ ms} \quad [7.10]$$

Experimental $T_{e\ min}$

We are going to verify $T_{e\_theoretical}$ experimentally. We describe the tests carried out, the measuring errors committed and the implementation of synchronization.

The test involves constantly reading the event memory to detect any change in the data assigned to an input. This change takes place more rapidly than communication does. Theoretically speaking, the information in the memory should be refreshed with a constant period ECT. We read the position of the motor at maximum speed, which varies every 0.251 ms.

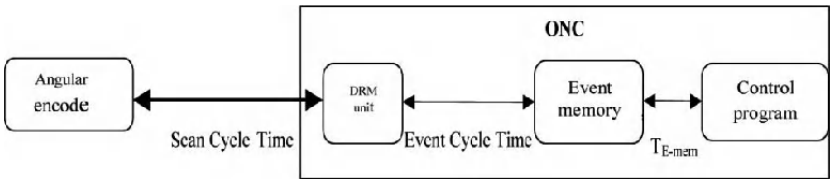


Figure 7.28. *Experimental sampling frequency*

7.2.4. *Results*

Experimentally, for the configuration SCT = 10 ms and ECT = 20 ms, if we sample every 21 ms, one sample will be lost after 200 samples. Hence, the sampling frequency of the cycle of the optimized control program found experimentally is:

$$T_{e\ \text{experimental}} = 21 \text{ ms} \quad [7.11]$$

We shall use the T_e found experimentally, rather than the $T_{e\ \text{theoretical}}$. In order to find this value, we assumed that the memory refresh frequency was exactly the same as the value of the configuration parameter ECT, but experimentally we observed

that this is not the case. This T_e found experimentally is the minimum permissible by the communication operations. We shall calculate the maximum time taken to compute the algorithms which does not entail an increase in this T_e. For this reason, the period of execution of the algorithms (T_{algor}) must be equal to T_e:

$$T_{algor} = T_{e\ experimental} = 21 \text{ ms} \quad [7.12]$$

The condition to obtain the minimum T_{algor}, limited by the algorithm computation time, is given by (0.16). As the program is not synchronized with the communications, we do not need to wait for a new piece of data to enter the memory ($t_{I/O\ refresh}$). The read and write time for the event memory is very slight (less than 1 ms). Hence, by using [7.11] and [7.12], we get:

$$t_{comput\ max} = 20 \text{ ms} \quad [7.13]$$

This is the maximum time taken to compute the algorithms on the excavator-loader, which enables us to use the T_e limited by the communications (21 ms).

7.2.4.1. *Jack stem speed control algorithm*

7.2.4.1.1. Implementation and optimization

We express the implementation of the algorithm and the optimizations carried out in this section.

The control law is defined in continuous time. In order to implement it on the onboard computer, we need to discretize it. Using the delay differential discretization law, we obtain:

$$\tau[n] = \bar{M}\left[\ddot{q}_{c\_pump}[n] + K_1\dot{e}[n] + K_2 e[n] + K_3 T_e \sum_{k=0}^{n} e[k]\right] \quad [7.14]$$

It is this equation which is implemented in the onboard structure. In addition, we need to calculate the necessary variables on the basis of the variables obtained in the previous algorithms:

– $q[n]$: the vector of measurements of the angular positions of the axles of the excavator-loader's mechanical arm (4×1);

– $\dot{q}_{c\_pump}[n]$: the vector of the desired speeds of the axles, considering the limits of the joints and of the pump (4×1).

Figure 7.29 is the organigram of the algorithm implemented.

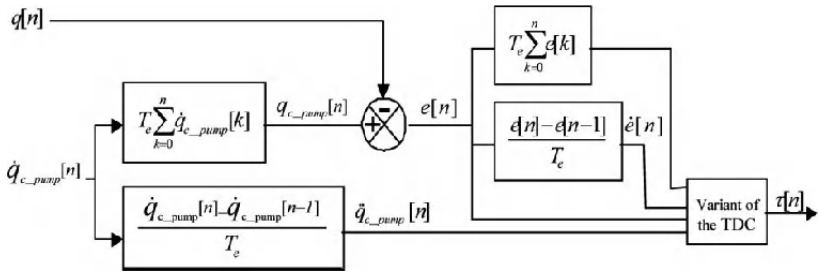


Figure 7.29. *Organigram of the control algorithm*

In order to minimize the execution time of the control algorithm, we made the following optimizations:

– to consider the matrix constants as scalar constants, as they are multiples of the identity matrix;

– to remove the loops. In order to do so, we developed specific functions for the size of the vectors used (4×1). As this algorithm has no loops, its asymptotic complexity is of order $O(1)$.

7.2.4.1.2. Computation time

The power of the CPU in the ONC is divided between our control program and the communications. That is to say, the managing of the communications slows down our control program, and therefore the execution time is not constant. This time increases when we decrease the value of the configuration parameter ECT, as the frequency of the communications increases.

We not only implemented the fourth algorithm (TDC), but also optimized the three algorithms which had already been programmed (computation of the possible set points). Table 7.5 shows the values of the minimum and maximum execution time for a cycle of four algorithms in the excavator-loader as a series (t_{comput}), depending on the value of the ECT.

ECT	$\mathbf{t_{comput}} \pm 0.25$ ms	
	Min.	**Max.**
30	1	1.5
20	1	***2***
10	1	2.5
1	2.5	6.5

Table 7.5. *Complete execution cycle time*

From the above, we can see that the optimum value for the ECT is 20 ms. Also, in order to use a sampling frequency of 21 ms, we have a time margin of 20 ms to compute the algorithms. We can see from Table 7.5 that this condition is largely fulfilled. Hence, we can use the sampling frequency of 21 ms and even implement new algorithms without needing to increase this frequency.

Figure 7.30 shows the structural diagram of the program, including its different functions.

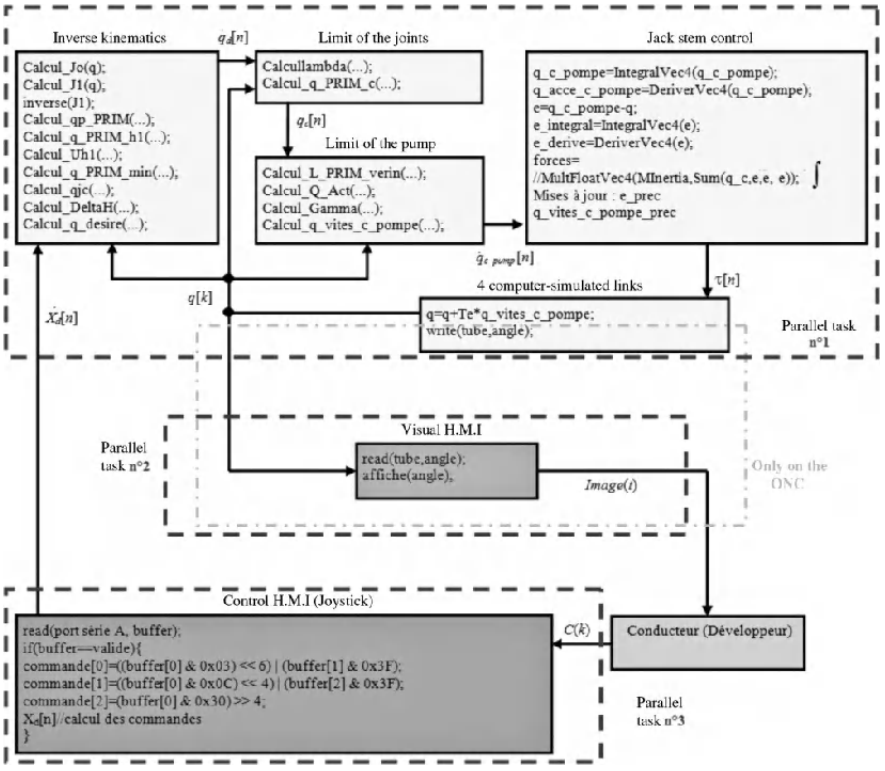


Figure 7.30. *Structural diagram of the program*

7.3. Conclusion

The MultiController thus created is a program which synthesizes all the required regulation algorithms for the GCS project. It considers the new development strategies in order to fully integrate itself in the context. This "object" also uses

some of the expertise from CERN (UNICOS framework) to distribute all the functions of the algorithms to the users. Incorporated into the new framework of the GCS project, MultiController is able to systematize the implementation of advanced control algorithms for all of the 23 gas installations. Finally, the MultiController provides users with a standard of use by way of "object processing" in the PLC and in the SCADA supervision system. This involves eight advanced control algorithms which are available to the designer.

In the context of the excavator-loader array, the four algorithms for automated control of the excavator-loader have been programmed. The fourth of these algorithms, which is a variant of the TDC control law, has been optimized.

After programming and optimization, the maximum execution cycle time for the four algorithms is 2 ± 0.25 ms depending on the optimal configuration found. This computation is extremely fast, so the sampling frequency is limited by the communications rather than by the computation time of the algorithms.

We conclude that the period for a cycle of the onboard system is 21 ms. Also, we can add new algorithms without increasing this time, until the algorithm computation time is greater than 20 ms.

7.4. Bibliography

[CAB 07] CABARET S., COPPIER H., RACHID A., *et al.*, "Framework and model driven approaches for the 23 gas control systems of the LHC experiments at CERN: practices for a SCADA-PLC industrial based solution", *IMSM07*, Buenos Aires, Argentina, February 2007.

[FRA 01] JIMENEZ FRAUSTRO F.F., Conception sûre des automatismes industriels: modélisation synchrone de langages d'automates programmables de la norme CEI-61131-3, Doctoral thesis, University of Rennes 1, France, November 2001.

[OBJ 01] Object Management Group Staff, A Proposal for an MDA Foundation Model – An ORMSC White Paper V00-02, 5 April 2001.

[OBJ 03] Object Management Group Staff and Contributors, MDA Guide Version 1.0.1, 12 June 2003.

[THO 05] THOMAS G., *et al.*, "LHC GCS: A Model-Driven approach for the automatic PLC and SCADA code generation", *ICALEPCS'2005*, Geneva, Switzerland, October 2005.

General Conclusion

The aim of this book is to demonstrate how best to integrate advanced automation engineering, "one of the major engineering sciences", into industrial reality, by use of some concrete examples of complex systems. Of these systems, we can cite the cryogenic systems or physical gas systems at CERN, the thermal or stone-producing systems for applications developed at Schneider Electric, for mechanical-hydraulic systems for a manufacturer of excavator-loaders and automobile vehicles. These complex systems, requiring the optimization and control of physical parameters, are driven by real-time distributed structures such as PLCs or onboard systems.

The different studies described in the chapters of this book reveal a number of useful recommendations in the design of the control-command for real systems on real-time targets in a distributed architecture environment.

To begin with, we have given a review of the modeling and linear control tools, such as PID regulators, the Smith Predictor, RST and generalized predictive control (GPC). Various examples have also been put forward to illustrate these regulation techniques. In the case of nonlinear systems described by multi-models, we have put forward conditions to enable engineers to synthesize various control laws. Firstly, LMI conditions for the synthesis of state feedback control were determined. Then, controls based on reconstructed state feedback and static state feedback were examined. These controls were presented in the form of LMIs that can be easily solved by existing numerical tools. Illustrative examples were given.

Then, we looked at some concrete examples of complex systems.

– *The LHC at CERN* uses the largest cryogenic installation in the world, employing helium as a coolant. This installation is controlled by industrial Programmable Logic Controllers (PLCs). The aims of this study are to present the issue of the models and control systems applied to certain cryogenic installations used at CERN and to exploit the experience accumulated over the course of more than 10 years during the construction, development and instauration of the cryogenic installations of the LHC. One of the most important points is the mathematical modeling of the nonlinear physical phenomena involved in the cryogenic processes. In large-scale installations such as those presented here, the possibilities for experimentation are greatly limited both by the costs and the risks involved in creating the installations. Nevertheless, on an exceptional occasion, it was possible to carry out an experimental campaign in a nitrogen heat exchanger in order to identify the process being controlled.

It should also be mentioned that, in the context described above, i.e. models and control applied on a massive scale to nonlinear systems, the practical application of traditional (linear) techniques may pose problems. Thus, we use new advanced modeling and control techniques to optimize the management of the installations, with the aim of enhancing the performance of the systems being controlled and thus reduce the undesirable effects which could, in the long term, increase operating costs and affect the availability of the system.

The main results relate to the controls applied on a large scale to cryogenic installations. Indeed, in the context of the standard identification and control techniques, this manuscript offers solutions for parametric identification and design of the control system for the nitrogen heat exchanger used by the ATLAS experiment. A summary of the technical and industrial approaches is given, with particular regard to the implementation of the mathematical results in industrial controllers by way of PLC objects made by Schneider. It should be mentioned that the heat exchanger is not usually available for the purposes of experimentation, and that the work carried out was only made possible by an exceptional authorization granted to us.

– *In the case of mechatronic systems*, the work presented herein focused on a particular form of driving support for building machines, and in particular on the study and realization of a human–machine interface to control the movements of hydraulic digging tools with six degrees of freedom. Firstly, geometric and kinematic models were presented. Next, a number of the main functions for a new driving support device were outlined and a computer architecture put forward. The new driving support device aims to transform the commands given by the driver to the tool into articulation speeds. This conversion is done by way of an inverse kinematic model using a “projected gradient” method.

In the context of automobiles, multi-controllers were applied with a view to improving stability and safety. The Takagi-Sugeno multi-model techniques were applied using LMI-type numerical tools.

Various algorithms for automated control of the excavator-loader were also programmed by integrated real-time constraints (the maximum execution cycle time for the algorithms, the sampling frequency, the communication time, etc.).

Because of these theoretical tools, for modeling and control, and real-world examples from industry, this book will be a good support to engineers and engineering students. It will give the readers access to both theoretical and practical tools to create their own control-command strategies for different mechatronic systems.

List of Authors

Sébastien CABARET
General Electric
Belfort
France

Mohammed CHADLI
UPJV-MIS
Amiens
France

Hervé COPPIER
ESIEE
Amiens
France

Elie KAFROUNI
CNAM
Amiens
France

Marco PEZZETTI
CERN
Geneva
Switzerland

Index

M

N, O, P

R

S

T

U, V, Y