ISSN 0280-5316
ISRN LUTFD2/TFRT--5839--SE

Model-based Friction Compensation

Timothée Basson
Julien Lescot

Department of Automatic Control
Lund University
June 2009

Lund University
Department of Automatic Control
Box 118
SE-221 00 Lund Sweden

Document name
MASTER THESIS

Date of issue
June 2009

Document Number
ISRN LUTFD2/TFRT--5839--SE

Author(s)
Timothée Basson and Julien Lescot

Supervisor
Anders Robertsson Automatic Control, Lund
Germain Garcia INSAT, France
Carlos Canudas-de-Wit INPG, France
Rolf Johansson Automatic Control, Lund (Examiner)

Sponsoring organization

Title and subtitle
Model-based Friction Compensation (Modellbaserad friktionskompensering)

Abstract
Friction is present in all mechanical systems and causes a wide range of problems for control. The development of model-based strategies accounting for Friction in the designed control has been a vast area of Research since these last decades. A promising Friction model which received a lot of attention these last years is the so-called LuGre model, which originates in the collaboration between the two control Research groups of Lund (Sweden) and Grenoble (France). This model is quite simple and is capable of capturing a wide range of well known friction phenomena, such as the Stribeck effect or the frictionnal lag. In particular this model is used in [Robertsson et al., 2004], where a general method for friction compensation for nonlinear systems is presented. The compensation strategy is simple: it just consists in adding to the control signal a friction estimate, computed using a LuGre model based observer. This thesis deals with the application of the theory of this article on a real experiment: the stabilization of the Furuta pendulum in the upright position. First, attention is paid so that the initial hypothesis of this article be satisfied. These hypotheses consist in finding a stabilizing control for the system when Friction is neglected, and an associated Lyapunov function verifying some properties. Then, Friction is included by following the procedure presented in the article. The friction estimate is computed according to the discretized LuGre form, presented in fFreidovich et al., 2006g, and the main result of the article is verified both in Simulation and on the real process, the simulations being carried out with Matlab-Simulink and the real experiments by using a dSPACE device. From a practical point of view, the implemented compensation scheme works perfectly in Simulation: the limit cycles originating from an uncompensated friction are totally annihilated, while for real experiments this oscillating behaviour is still remaining, but happens to be significantly reduced. From a theoretical point of view, the results of [Robertsson et al., 2004] are fully verified in Simulation, while for real experiments the presence of remaining limit cycles prevents a perfect verification of the theory.

Keywords
Friction Compensation, LuGre Model, Furuta Pendulum, Limit Cycle, dSPACE

Classification system and/or index terms (if any)

Supplementary bibliographical information

ISSN and key title
0280-5316

ISBN

Language
English

Number of pages
98

Recipient's notes

Security classification

Preface

This thesis is the final thesis (Projet de fin d'études) of our respective schools : INSA of Toulouse (Timothée), INPG (Julien). It was carried out within the Erasmus framework at the Department of Automatic Control of the Lund Institute of Technology (LTH) at Lund University.

First we would like to thank Prof. Dr. Rolf Johansson and Dr. Anders Robertsson for their supervision of and the important impulses for our thesis. Their advice and criticism for our work were a great help and indispensable for our thesis.

Thanks to them we have learnt a lot about control and working in the research area. This will be very useful during our PhD next year.

Especially we want to thank Leif Andersson for his patience, for his help with using Linux at the Control Department and for his practical advice for writing our thesis. And thank you to Eva Westin, Eva Schildt and Britt-Marie Mårtesson for their administrative help.

We would also like to thank our supervisors in our respective schools in France : Prof. Germain Garcia (INSAT) and Prof. Carlos Canudas-de-Wit (INPG).

Thanks also go to our friends at the Department of Control: Lorenz, Meike, Anne-Kathrin, Julia, Giulio, Johan, Olof, Johannes, and Martin, for their friendship and for showing us how the Nations work in Lund.

Also a special thank to our friends in Lund: Brunilde, Yudzu, Basti, Maxence, Vincent for all the good time we spent together.

I would like to thank also my family for their support and for having given me the opportunity to study abroad. I would like to thank especially my roommate, co-worker and friend Timothée for his patience with me during this semester. I hope we will stay as good friends as we were in Lund when we are in France.

And finally I want to thank Fanny for being here in Lund and for giving me the opportunity to escape from my work at the university.

Julien

My thanks also go to my friends in France: Rémy, John, Erwan, Shushu, Régis, Marie-Aude and Fanny, who were, in spite of the distance, always there for me during this year abroad.

I would also like to especially thank my roommate, "team mate", and friend Julien for his kindness, patience, and zest for life. I am sure that in France we will stay as good friends as we were in Lund!

Finally and most importantly I would like to thank my parents and my sister for their support – both material and inspirational –, advice and help when things were not going exactly as I would have expected.

Timothée

Contents

1. Introduction

1.1 Problem Description

This thesis deals with Model-based Friction compensation. The goal is to verify the theory presented in the article by [Robertsson *et al.*, 2004], where a Friction compensation method, based on the LuGre model, is derived for general nonlinear systems.

The process considered is the Furuta pendulum (see figure 1.1). It is a system with two rotational degrees of freedom, but only one, the arm angle, is actuated by a DC motor , the other being the pendulum angle which has to be stabilized in the upright position.

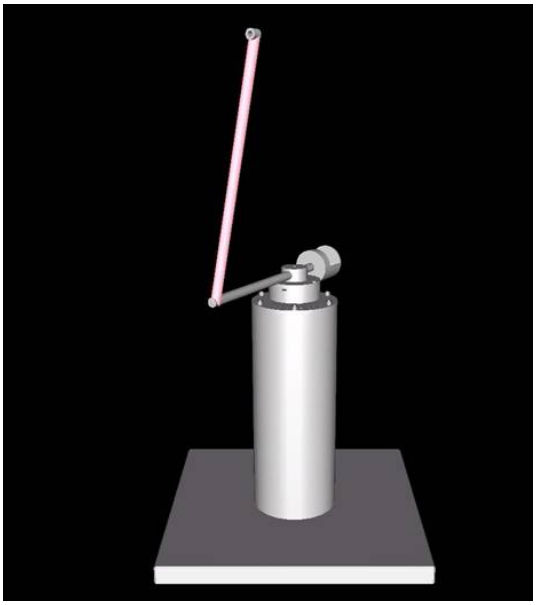


Figure 1.1 The furuta pendulum

The theory is verified in Simulation with Matlab Simulink and experimentally using a dSPACE board.

Moreover the work from a previous thesis written by Thomas Dietz [Dietz, 2006] is used as help for building the Simulink files and having a first knowledge about Friction compensation. However, as the process used here is the same as the one used in this thesis, the parameters for the plant and for the Friction Model are taken from the identification done in this thesis.

1.2 Motivations

Friction is widely spread in most of the mechanical or electro-mechanical systems. This property can be seen as an advantage or an inconvenient depending on the domain of application where friction occurs. For instance without friction it would be impossible to move with a bike, the wheels would just slip on the ground and the

bike would stay motionless. But now if the domain considered is robotic applied to medicine, the robot has to be highly accurate and not to be submitted to any kind of disturbance.

Unfortunately, as it will be explained later, one of the drawbacks of Friction is the emergence of limit cycles. For this reason, Friction compensation has to be investigated, and solving this problem could lead to very good improvements in a lot of domains. Moreover, as friction is highly non-linear the survey of such a phenomenon could allow to develop new theoretical elements about non-linear control.

1.3 Phenomena Due to Friction

One of the major phenomena arising with friction uncompensated systems is the emergence of limit cycles. Two typical limit cycles are well-known : the Hunting and the Stick-Slip motion. These two phenomena are described in the following section, the examples being picked from the PhD thesis of Henrik Olsson[Olsson, 1996].

A Simple Friction Model

In this part, each simulation is run using the following model for the friction :

$$F = \begin{cases} F_C sgn(v), & if\ v \neq 0 \\ F_e, & if\ v = 0\ and\ |F_e| < F_S \\ F_S sgn(F_e), & otherwise \end{cases} \qquad \begin{array}{lcl} F_C & : & \text{Friction force} \\ F_e & : & \text{External force} \\ F_S & : & \text{Break-away force} \end{array} \tag{1.1}$$

The model is a coulomb model extended with a breakaway force. It is not the simplest one but it allows to show phenomena due to friction. The figure 1.2 clearly describes the model :

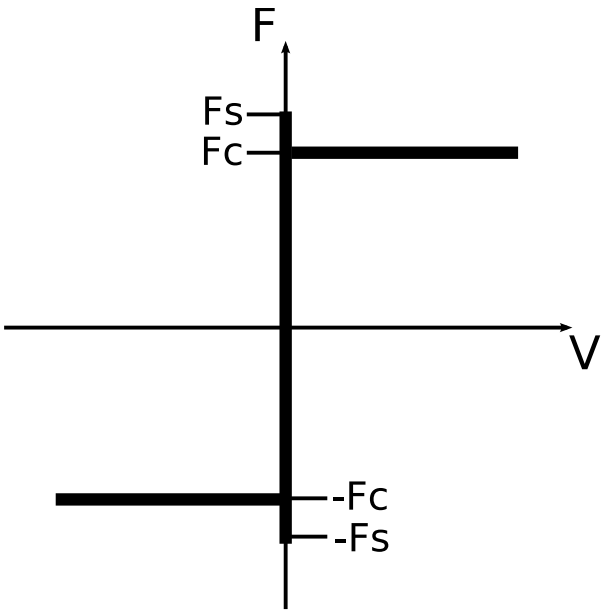


Figure 1.2 The friction model

For implementational reasons, the model is slightly modified so that the behaviour at zero velocity also occurs for a given ε. This has to be done because zero does not really exist in a computer.

$$F = \begin{cases} F_C sgn(v), & if\ |v| \geq \varepsilon \\ F_e, & if\ |v| < \varepsilon\ and\ |F_e| < F_S \\ F_S sgn(F_e), & otherwise \end{cases} \tag{1.2}$$

Hunting

Hunting is a limit-cycle around the reference value. It is due to the integral part of a position-controller which is applied to a system submitted to friction. The phenomenon is illustrated by taking the example of a mass moving along a one-dimensional axis.
The motion equation can thus write :

$$m\frac{d^2x}{dt^2} = u - F \tag{1.3}$$

where u is the control force given by :

$$u(t) = -K_v v(t) - K_p x(t) - K_i \int^t (x(\tau) - x_r(\tau))d\tau \tag{1.4}$$

and F the friction force as in 1.2

Illustrations[1]

Figure 1.3 shows the system without friction and in Figure 1.4 Friction is included.

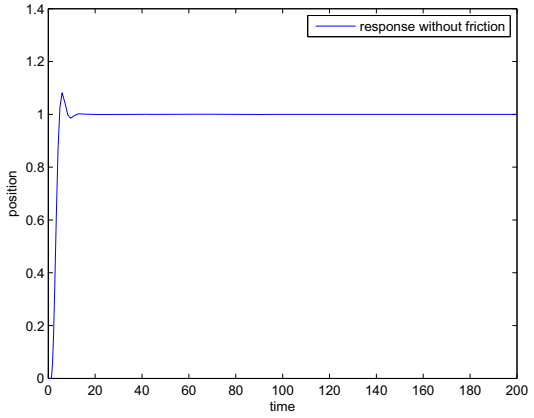


Figure 1.3 Position of the mass without friction

[1]The simulations parameters are : Fc=0.5, Fs=1, Kv=2, Kp=2, Ki=1, xr=1, and m=1. For more details please refer to [Olsson, 1996]

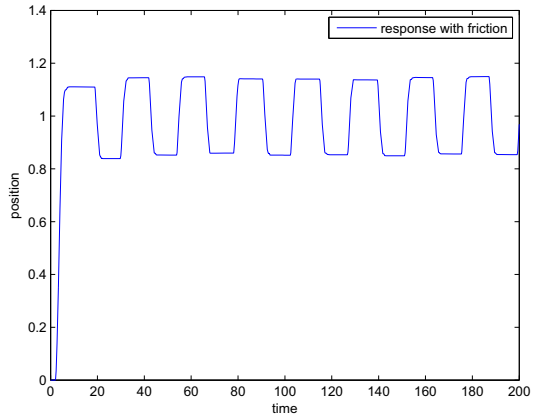


Figure 1.4 Position of the mass with friction

If the two figures are compared, clearly the friction is a significant drawback because it leads to oscillations around the reference point. If more attention is paid to figure 1.4, it can be noticed that the mass point just stays motionless besides the reference value for a while before it goes back and stays below it. This phenomenon is similar to another phenomenon called slip-stick motion.

Stick-Slip Motion

Stick-slip motion is the fact that the considered mechanical object moves by steps: a slip-step and a stick-step. The system which is used to highlight this phenomenon is a simple mass coupled with a spring. A velocity control is used, and the system can be finally represented by the following equations :

$$m\frac{d^2x}{dt^2} = k(y-x) - F \tag{1.5}$$

$$\frac{dy}{dt} = v_{ref}\,, \tag{1.6}$$

the friction force F being described by the model 1.2

Illustrations[2]

Thanks to the figures 1.5 and 1.6 the meaning of stick-slip motion becomes obvious. If there is no friction (see Figure 1.5) the mass just follows the reference and oscillates around it. But when Friction is taken into account (Figure 1.6), the mass just stays for a while at a given position (for instance from $t = 10\,s$ to $t = 15\,s$) until it moves again to try to reach the reference value (for instance from $t = 15\,s$ to $t = 17s$).

[2]the simulations parameters are : Fc=1, Fs=1.5, k=2, $v_{ref} = 0.1$, and m=1. For more details please refer to [Olsson, 1996]

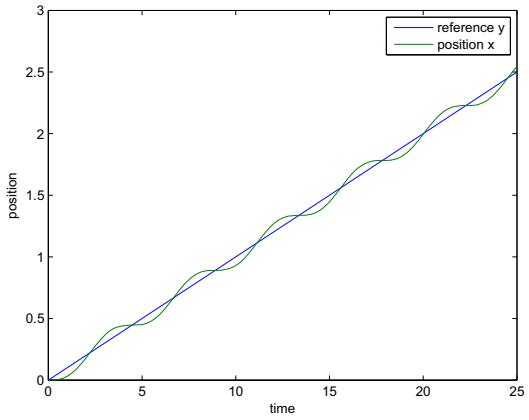


Figure 1.5 Position of the mass without friction

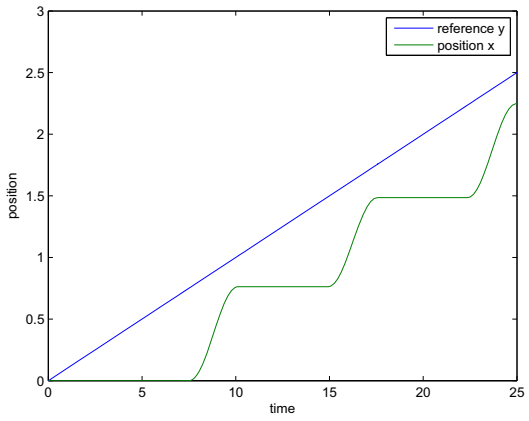


Figure 1.6 Position of the mass with friction

1.4 Friction Compensation

This section aims at giving information about Friction and its compensation.

A first part deals with the description of the most important phenomena involved in friction and presents a recently built model of friction, which will also be used throughout this thesis: the LuGre friction model.

In the second part, the results of the article [Robertsson *et al.*, 2004] – in which a method of friction compensation for general nonlinear systems is derived – are presented. Indeed, for the authors it seemed necessary to explain this article, as the goal of this thesis is to verify these results on a real experiment.

Typical Friction Behaviour and the LuGre Model

Friction typically occurs when two different surfaces are put into contact. As an example one can consider (as in the previous section) a rectangular block, lying over a flat surface:

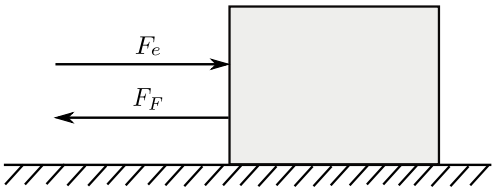


Figure 1.7 Rectangular bloc lying on a flat surface. F_F corresponds to the tangential friction force between the two surfaces, opposed when F_e, an external force, is tangentially applied to the block.

Basically, friction contacts can be encountered in two different states: “sticking” (also called presliding) and “sliding”. The transition between these two states is continuous, and the force required to pass from sticking to sliding is called the breakaway force, or the Stiction Force.

In the state of “sticking”, there is no macroscopic relative movement between the two surfaces. Given the above example, this state would correspond to the situation where the friction F_F exactly opposes the applied force F_e so that the block keeps motionless.

On the contrary, when F_e reaches the breakaway force, the friction forces are no longer able to overcome the applied force, and thus the block starts moving, leading to the sliding mode.

A recently friction model which received a lot of attention these last years is the so-called LuGre model, which was built thanks to a collaboration between the Control departments of Lund (Sweden) and Grenoble (France).

This model is capable of capturing a wide range of phenomena which have been observed while studying the friction behaviour of mechanical systems.

This section intends to give a short introduction to this model, and some of the characteristic friction behaviours covered by this model. Further annotations and expressions concerning the description of the friction which will be used throughout this thesis are introduced.

Moreover, this section does not intend to give a complete description and listing of all the friction phenomena captured by the LuGre model.

Indeed for more insight the reader is strongly recommended to refer to [Åström, 1998]. Further a really good introduction to friction modelling, and to the different friction models developed prior to the LuGre model, is given in the PhD thesis of [Olsson, 1996].

The LuGre model is a dynamical model for the description of the friction. In this model, friction is physically described as the result of the bending of surface asperities between the two surfaces in contact. These asperities are modelled as bristles which deflect like springs when a tangential force is applied (see [Åström, 1998] and [Olsson, 1996]). The LuGre model collects the behaviour of all these surface asperities and describes their accumulated behaviour through one bristle, see figure 1.8:

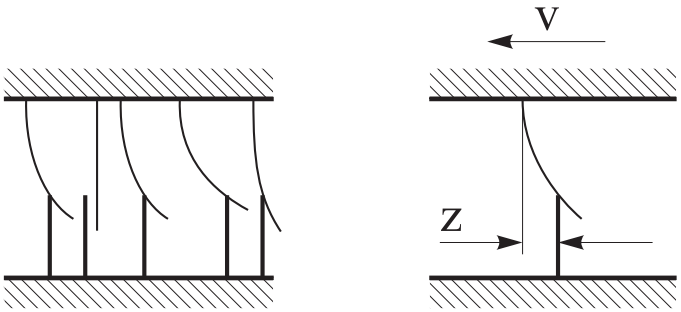


Figure 1.8 Left: schematic picture showing the modelization of the two contact surfaces asperities, by bristles, which bend under load. Right: the average deflection of the bristles of the contact surfaces is collected through one bristle

Referring to [Olsson, 1996], the LuGre model is described by the following equations:

$$\frac{dz}{dt} = v - \sigma_0 \frac{|v|}{g(v)} z \tag{1.7}$$

$$F_F = \sigma_0 z + \sigma_1(v)\frac{dz}{dt} + F_v v \tag{1.8}$$

$$g(v) = \left(F_C + (F_S - F_C)e^{-(v/v_S)^2}\right) \tag{1.9}$$

$$\sigma_1(v) = \sigma_1 e^{-(v/v_d)^2} \tag{1.10}$$

where v denotes the relative velocity of the contact surfaces and F_F is the LuGre friction force.
Referring again to [Olsson, 1996] , these equations can be interpreted as follows:

Equation (1.7) describes the behaviour of the internal state of the friction model, i.e. the average deflection of the bristles and collected through one bristle, as mentioned above. The first right hand term allows the deflection to be proportional to the integral of v , while the second term ensures that z be velocity dependant at steady state by reaching the value

$$z_{SS} = \frac{1}{\sigma_0} g(v) sgn(v)$$

for nonzero velocity v.

Equation (1.8) describes the LuGre friction force's dependence on the internal state and its derivative. The first term accounts to the friction force due to the stiffness σ_0 of the bristles, while the second one allows well-behaved transitions between sticking and sliding for small velocities v. Finally, the third term accounts for linear viscous friction which appears at high velocities.

Equation (1.9) details the function $g(v)$, which specifies the velocity dependence of the friction force F_F at steady state and for relatively low velocities. Indeed, at steady state:

$$F = F_{SS} = g(v) sgn(v) + F_v\, v$$

As it will be seen below, this function accounts for the Stribeck effect, and more specifically it can be seen as a time constant defining how quick the steady state

value z_{SS} can be reached.

Equation (1.10) describes a common parametrisation of the damping coefficient $\sigma_1(v)$. With this parametrisation, $\sigma_1(v)$ exponentially decays with the velocity. Hence it is maximal for zero velocities and therefore allows to avoid unwanted oscillations at the breakaway, while it becomes negligible at high velocities. However a simplified version of the friction model above, where $\sigma_1(v)$ is assumed to be constant, is also used frequently.

The main friction phenomena which are handled by the LuGre model are described below. For more information about these phenomena, the reader is again strongly advised to refer to [Olsson, 1996] and [Åström, 1998].

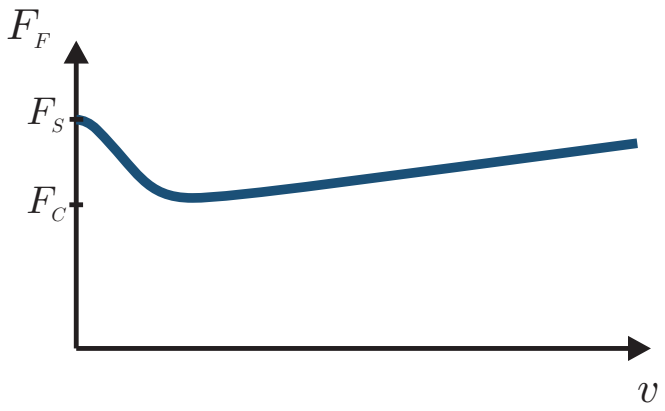


Figure 1.9 Typical behaviour of the friction force F_F in the sliding regime and in dependence on the velocity v. For low velocities the Stribeck effect governs the transition of the friction force from the Stiction force F_S to the Coulomb friction Force F_C, while at high velocities the Coulomb and the viscous friction forces dominate the behaviour.

- Stribeck effect

  The experimentally observed decrease of the friction force at low relative velocity, from the breakaway force (higher friction value at stiction) to a lower value when sliding occurs is known as the Stribeck effect (see figure 1.9). This effect is captured in the LuGre model by the function $g(v)$, which also encounters for friction value at steady state.

- Static friction and Breakaway force

  Experiments show that static friction and breakaway forces depend on the rate of increase of the external force applied, and thus velocity. Hence, friction is a dynamic process which does not only depend statically on the the force or velocity applied.

- Presliding

  Presliding or sticking occurs when the external force applied keeps below the breakaway force. Even though there is no macroscopic movement in the presliding state, it does not mean that there is no displacement between the two surfaces. Indeed, the friction surfaces can be imagined to be connected by springs, allowing a relative movement between the two surfaces.

- Frictional lag

  The frictional lag corresponds to a time delay between velocity and steady state friction force, and leads to an hysteretic behaviour of the friction force for velocity changes. Indeed, F_F is lower for decreasing velocities than for increasing ones, the hysteretic behaviour being moreover strongly dependent on the rate of the velocity change.

Summary of article [Robertsson *et al.*, 2004]

In this article, the problem of friction compensation for nonlinear systems is considered. The method presented here starts from hypothesis which are made on the system when friction is not taken into account. It is first supposed that the system dynamics without friction can be written in the following state-space representation form:

$$\begin{cases} \dot{x} &= F(x)+G(x)u \\ u &= H(x,t) \quad , \end{cases} \tag{1.11}$$

with $x \in \mathbb{R}^n$ the state vector, and $u = H(x,t) \in \mathbb{R}$ the control action which achieves uniform asymptotic stability in a compact subset $\Omega \subset \mathbb{R}^n$.

Further it is assumed that there exists a non negative function V such that :

- the zero level of V belongs to Ω, i.e:

$$V_0 = \{x \in \mathbb{R}^n \ : \ V(x,t) = 0\} \subset \Omega \tag{1.12}$$

- V satisfies the following inequality:

$$0 \leq |y(x,t)| = V(x,t) \leq V_m |x|^2 \tag{1.13}$$

- the time derivative of V along the trajectories verifies the inequality:

$$\dot{V} \leq - \begin{bmatrix} x \\ u \end{bmatrix}^T Q(t) \begin{bmatrix} x \\ u \end{bmatrix} \leq 0 \tag{1.14}$$

  with

$$Q = Q^T = \begin{bmatrix} Q_1 & Q_{12} \\ Q_{12}^T & Q_2 \end{bmatrix}, Q_2 > 0$$

Friction is then taken under consideration by considering it as an external force, acting directly on the control.The real control τ applied to the plant thus writes:

$$\tau = u - F \tag{1.15}$$

and may, as seen before, prevent the system from being stabilized. The solution proposed by the authors is to compensate this friction by modifying the nominal controller u as follows:

$$u = H(x,t) \rightarrow u_{new} = H(x,t) + \hat{F}, \tag{1.16}$$

so that finally the new "friction-compensated" real control τ writes:

$$\tau = u_{new} - F = H(x,t) + \hat{F} - F \tag{1.17}$$

$\hat{F}$ is a friction estimate, and is reconstructed by the following observer, based on the LuGre model previously defined:

$$\begin{cases} \hat{F} &= \sigma_0\hat{z}+\sigma_1\dot{\hat{z}}+F_v v \\ \dfrac{d\hat{z}}{dt} &= v-\sigma_0\dfrac{|v|}{g_{fr}(v)}\hat{z}+K \\ &= v-\varphi(v)\hat{z}+K, \hat{z}(0)=0 \end{cases} \tag{1.18}$$

The main result of [Robertsson *et al.*, 2004] and given in the theorem below is the found expression of the observer gain K of (1.18) (also called injection function) so that the new controller defined by (1.16) and compensating the Friction also achieves to perform uniform asymptotic stability for the system when Friction is present.

THEOREM 1.1
Under the previous hypothesis (1.12), (1.13), (1.14), and given the system (1.11), (1.15), and the LuGre model previously seen consider the control:

$$u = H(x,t)+\hat{F}$$

where the friction estimate $\hat{F}$ is defined by (1.18).
For:

$$\rho > 4Q_2\sigma_1\sigma_0 \tag{1.19}$$

the observer injection function chosen as :

$$K = \frac{2(H(x,t)Q_2+Q_{12}x)(\sigma_0+\sigma_1\varphi)}{(\rho-4Q_2\sigma_1\sigma_0)} \tag{1.20}$$

will guarantee that along any solution $[x(t),\ z(t),\ \hat{z}(t)]$ of the closed loop system the limit relation

$$\lim_{t\to+\infty} V(x(t)) = 0$$

holds, that is x(t) converges to the compact set V_0 defined in (1.12). □

The proof of this theorem is based on finding a value of K such that the following candidate function :

$$W = V+\frac{\rho}{2}(z-\hat{z})^2 \tag{1.21}$$

be a Lyapunov function for the system augmented by the LuGre friction model and the observer (1.18), and with the new "Friction-compensated" control (1.17).

More precisely the time derivative of W:

$$\dot{W} = \dot{V}+\rho\, e\,\dot{e}$$

can be majored:

$$\dot{W} \leq -\begin{bmatrix} x \\ H \end{bmatrix}^T Q(t) \begin{bmatrix} x \\ H \end{bmatrix} + f(K)$$

by using the property (1.14) on $\dot{V}$ and the following dynamic of the internal state error $e = z-\hat{z}$:

$$\dot{e} = -\sigma_0\frac{|v|}{g_{fr}(v)}e-K \tag{1.22}$$

Finally K is chosen such that $f(K) = aK^2+bK+c$ is made minimal and negative.

1.5 Objective of the thesis

The main goal of this thesis is to verify the results presented in [Robertsson *et al.*, 2004] on a real experiment: the Furuta pendulum. Following the procedure explained in this article, first a control stabilizing in the upright position the system without friction has to be found, as well as the associated Lyapunov function which has to verify properties (1.12), (1.13) and (1.14). Throughout this report, this function will be named as the "initial Lyapunov function", meaning that in a first time, consideration have to made on the system without friction.

Once this done it will be then verified, both in Simulation and on the real process, that, when Friction compensation is applied with the observer defined in (1.18) and the expression of K found in Theorem 1.1, the candidate function W defined in (1.21) is a Lyapunov function.

This function being built by adding a term to the initial Lyapunov function V so that it can be a Lyapunov function for the system augmented with the Lugre friction model and the LuGre observer, it will be named as the "extended Lyapunov function" throughout this report.

1.6 Outline of the report

The outline of the report will be organized as follows:

Chapter 2 introduces the hardware of the Furuta pendulum, and gives a state space model of the plant.

In Chapter 3, the different control strategies which were tried to perform the stabilization of the pendulum in its upright position and when Friction is not taken into account are presented.

Chapter 4 deals with implementational considerations. First a description of the dSPACE bench is given.Then the way friction compensation will be performed is presented, and finally the chapter ends by giving the general framework used for simulations without hardware in the loop and on the real process.

Chapter 5 presents the results of the Matlab-Simulink simulations which were carried out in order to verify the result seen in [Robertsson *et al.*, 2004].

Chapter 6 has the same purpose but here experiments are carried out on the real process.

Chapter 7 finally gives a summarize, and a conclusion to the work and analysis done throughout this thesis. Additionally, the authors also give their personal feelings about the thesis at the end of the chapter.

2. The Furuta Pendulum : Description and Modelization

2.1 Hardware Description

Figure 2.1 The Furuta pendulum

The Furuta pendulum is an inverted pendulum created by the Professor Katsuhisa Furuta. The device is very simple, it consists in a simple arm which moves thanks to a conventional DC motor. A rod with a small mass is attached at the arm and is the real pendulum of the process. The main goal which can be achieved with the Furuta pendulum is to stabilize the pendulum in its upright position. One of the particularities of the system is that the process is under-actuated, there are two degree of freedom (the angle of the arm and the angle of the pendulum) and only one actuator : the moment applied to the arm. Other inverted pendulums exist, for instance the cart-pendulum system. But a main advantage of the Furuta pendulum is that the motion of the arm is not bounded as the motion of the cart is. The limitation comes from the limited length of the rail of the cart pendulum whereas for the Furuta pendulum the arm can rotate indefinitely around its axis.

In this system the variable which is controlled is the angle of the pendulum θ. The measure of θ is given by a potentiometer whereas the measure of φ is given by

an encoder. The two velocities of these angles are reconstructed by a built-in analog filter.

2.2 Modelization and State-Space Representation

Equation of Motion

In this section, equations for the modelization and the state-space form of the Furuta pendulum are described. It is really easy to find such a modelization for instance in [Izutsu and Furuta, 2007]. But here the modelization as well as the values of the parameters of the Furuta pendulum are derived from a previous master thesis [Dietz, 2006]. Here the main result is summarized.

The main result is the matrix equation 2.1:

$$\mathbf{M}(\mathbf{q}) \begin{pmatrix} \ddot{\varphi} \\ \ddot{\theta} \end{pmatrix} + \mathbf{h}(\dot{\mathbf{q}}, \mathbf{q}) + \mathbf{G}(\mathbf{q}) = \mathbf{f} \tag{2.1}$$

with:

$$\mathbf{M}(\mathbf{q}) = \begin{bmatrix} (mL^2 + m_2L_2^2 + J_P)\sin^2(\theta) + (m+m_2)R^2 + J & -(mL + m_2L_2)R\cos(\theta) \\ -(mL + m_2L_2)R\cos(\theta) & (mL^2 + m_2L_2^2 + J_P) \end{bmatrix} \tag{2.2}$$

$$\mathbf{h}(\dot{\mathbf{q}}, \mathbf{q}) = \begin{pmatrix} 2(mL^2 + m_2L_2^2 + J_P)\dot{\varphi}\dot{\theta}\sin(\theta)\cos(\theta) + (mL + m_2L_2)R\dot{\theta}^2\sin(\theta) \\ -(mL^2 + m_2L_2^2 + J_P)\dot{\varphi}^2\sin(\theta)\cos(\theta) \end{pmatrix} \tag{2.3}$$

$$\mathbf{G}(\mathbf{q}) = \begin{pmatrix} 0 \\ -g(mL + m_2L_2)\sin(\theta) \end{pmatrix} \tag{2.4}$$

$$\mathbf{f} = \begin{pmatrix} M_M - M_F \\ -M_{FP} \end{pmatrix} \tag{2.5}$$

M_M	:	Moment of the motor
M_F	:	Friction Moment applied on the joint of the arm
M_{FP}	:	Friction moment of the joint of the pendulum

State-Space Form

Now using the equation 2.1 and applying basic arithmetic operations the following

equations can be obtained :

$$\ddot{\varphi} = \frac{1}{C_1(C_1\sin^2(\theta)+C_3+J)-C_2^2R^2\cos^2(\theta)}\cdot \tag{2.6}$$
$$\Big[C_1\big(-2C_1\dot{\varphi}\dot{\theta}\sin(\theta)\cos(\theta)-C_2R\dot{\theta}^2\sin(\theta)+M_M-M_F\big)\dots$$
$$+C_2R\cos(\theta)\big(C_1\dot{\varphi}^2\sin(\theta)\cos(\theta)+gC_2\sin(\theta)-M_{FP}\big)\Big]$$
$$\ddot{\theta} = \frac{1}{C_1(C_1\sin^2(\theta)+C_3+J)-C_2^2R^2\cos^2(\theta)}\cdot \tag{2.7}$$
$$\Big[\big(C_1\sin^2(\theta)+C_3+J\big)\big(C_1\dot{\varphi}^2\sin(\theta)\cos(\theta)+gC_2\sin(\theta)-M_{FP}\big)\dots$$
$$+C_2R\cos(\theta)\big(-2C_1\dot{\varphi}\dot{\theta}\sin(\theta)\cos(\theta)-C_2R\dot{\theta}^2\sin(\theta)+M_M-M_F\big)\Big]$$

where the constants $C_1 = mL^2+m_2L_2^2+J_P$, $C_2 = mL+m_2L_2$ and $C_3 = (m+m_2)R^2$ were introduced[1].

Now considering the following state vector :

$$X = \begin{pmatrix} \varphi \\ \dot{\varphi} \\ \theta \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix}$$

And considering equations (2.6) and (2.7) the following state-space representation can be written.

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = \frac{1}{C_1(C_1\sin^2(x_3)+C_3+J)-C_2^2R^2\cos^2(x_3)}[C_1(-2C_1\sin(x_3)\cos(x_3)x_2\cdot \\ \quad x_4-C_2Rx_4^2\sin(x_3)+M_M)+C_2R\cos(x_3)(C_1x_2^2\sin(x_3)\cos(x_3)\dots \\ \quad +gC_2\sin(x_3))] \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = \frac{1}{C_1(C_1\sin^2(x_3)+C_3+J)-C_2^2R^2\cos^2(x_3)}[(C_1\sin^2(x_3)+C_3+J)(C1x_2^2\cdot \\ \quad \sin(x_3)\cos(x_3)+gC_2\sin(x_3))+C_2R\cos(x_3)(-2C_1\sin(x_3)\cos(x_3)x_2\cdot \\ \quad x_4-C_2Rx_4^2\sin(x_3)+M_M)] \end{cases}$$

It can be written as follows :

[1] By introducing these constants a simpler parametrization for the parameters of the pendulum was introduced, since the constants collect the parameters of the rod and the mass of the pendulum, thus combining them in the equations to one rigid body. C_1 is the moment of inertia of this body with respect to the joint between pendulum and arm, while C_2 is a constant that is related to the impulse of the pendulum. C_3 is the moment of inertia of the pendulum in the upright or hanging position with respect to the axis of rotation of the motor.

$$\begin{cases} \dot{x}_1 &= & x_2 \\ \dot{x}_2 &= & \dfrac{1}{A(X)}[C_1 B(X) + D(X)E(X)] \\ \dot{x}_3 &= & x_4 \\ \dot{x}_4 &= & \dfrac{1}{A(X)}[F(X)E(X) + D(X)B(X)] \end{cases} \tag{2.8}$$

With :

$$A(X) = C_1(C_1 \sin^2(x_3) + C_3 + J) - C_2^2 R^2 \cos^2(x_3) \tag{2.9}$$

$$B_n(X) = -2C_1 \sin(x_3)\cos(x_3)x_2 x_4 - C_2 R x_4^2 \sin(x_3) \tag{2.10}$$

$$B(X) = B_n(X) + M_M \tag{2.11}$$

$$D(X) = C_2 R \cos(x_3) \tag{2.12}$$

$$E(X) = C_1 x_2^2 \sin(x_3)\cos(x_3) + gC_2 \sin(x_3) \tag{2.13}$$

$$F(X) = C_1 \sin^2(x_3) + C_3 + J \tag{2.14}$$

3. Control Strategy

As said previously, one will experimentally verify the results of [Robertsson *et al.*, 2004], and summarized in Theorem 1.1, by trying to stabilize the Furuta pendulum for

$$\left[\theta = 0, \dot{\theta} = 0, \varphi = \varphi_{ref}, \dot{\varphi} = 0\right]$$

i.e. in its upright position and for a given arm angle φ_{ref}. This chapter details the different approaches tried so that the results of [Robertsson *et al.*, 2004] can be applicable. The goal was therefore to find, first, a control

$$u = H(X,t)$$

achieving uniform asymptotic stability for the system without friction, and then a non negative function V verifying properties (1.12), (1.13) and (1.14).

3.1 Lyapunov candidate functions

The methods presented in this section are nonlinear Lyapunov based control strategies. A positive definite candidate function has to be found, as well as the control which renders its derivative at least negative semi-definite.

The first Lyapunov function tried is a basic quadratic form, while the second one is built by making consideration on the energy.

Quadratic Form

Here the candidate function takes the following form:

$$V(X) = \frac{1}{2}X^T X = \frac{1}{2}(x_1^2 + x_2^2 + x_3^2 + x_4^2) \tag{3.1}$$

The time derivative of V(x) – derived using the state space representation form of the system given in the section 2.2 page 18 – can be written as:

$$\dot{V}(X) = G(X) + H(X)M_M \text{ , with} \tag{3.2}$$

$$G(X) = x_1 x_2 + x_3 x_4 + \frac{D(X)E(X) + C_1 B_n(X)}{A(X)} x_2 + \frac{D(X)B_n(X) + F(X)E(X)}{A(X)} x_4 \tag{3.3}$$

$$H(X) = \left[\frac{C_1 x_2 + D(X)x_4}{A(X)}\right] \text{ ,} \tag{3.4}$$

and has to be negative definite. As a choice of $\dot{V}$ one chooses a quadratic form:

$$\dot{V}(X) = -X^T Q X = -q(X) \text{ , with } q(X) \leq 0 \tag{3.5}$$

The control M_M allowing to obtain such a form is then found by equaling (3.2) with (3.5):

$$M_M = \frac{-q(X) - G(X)}{H(X)} \tag{3.6}$$

Unfortunately it appears that $H(X)$ is not invertible as it cancels for $C_1x_2 = -D(X)x_4$. Moreover, expanding the expression (3.6) of M_M:

$$M_M = \frac{1}{C_1x_2 + D(X)x_4}\left[-A(X)q(X) - A(X)x_1x_2 - A(X)x_3x_4 - (D(X)E(X)\ \ldots \right.$$
$$\left. + C_1B_n(X))x_2 - (D(X)B_n(X) + F(X)E(X))x_4\right],$$

one sees that there is no possible simplification, and that M_M is indefinite when $H(X) = 0$. To avoid an infinite command at the singularity, one could set M_M to some arbitrary value, but then $\dot{V}$ would become:

$$\dot{V} = G(X) = x_4\left[-\frac{C_2}{C_1}R\cos(x_3)x_1 + x_3 + \frac{C_2^2}{C_1^2}R^2\cos^2(x_3)\sin(x_3)\cos(x_3)x_4^2\ldots\right. \quad (3.7)$$
$$\left. + g\frac{C_2}{C_1}\sin(x_3)\right] \quad (3.8)$$

and would therefore not be negative definite. An other approach had therefore to be investigated.

Energy Based Function

Here the proposed candidate function is energy based. In a first time, the mechanical energy:

$$V(X) = K + P$$

was tried, with K and P the kinetic and potential energies respectively. But the function was cancelling in the downright position ($\theta = \pi$), but not in the upright one (as the potential energy becomes maximal). So the idea was to "invert" the potential energy, so that the upright position becomes the minimum of the function. The chosen function was then:

$$V(X) = \frac{1}{2}\left(L(X) - L_0\right)^2\ ,$$

with L the Lagrangian of the system such that :

$$L(X) = K - P$$
$$L(X) = \frac{1}{2}C_1x_2^2\sin^2(x_3) + \frac{1}{2}(C_3 + J)x_2^2 - C_2Rx_2x_4\cos(x_3) + \frac{1}{2}C_1x_4^2 - gC_2\cos(x_3)\ldots$$
$$- V_{pendulum} - V_{arm0}\ ,$$

and L_0 a constant such that $V(X)$ is positive definite and cancels at the upright position:

$$L_0 = -gC_2 - V_{pendulum} - V_{arm0}$$

The time derivative of V writes:

$$\dot{V}(X) = (L(X) - L_0)\,\dot{L}(X), \quad (3.9)$$

and after some calculations, one finds the expression of $\dot{L}(X)$:

$$\dot{L}(X) = G(X) + H(X)B_n(X) + H(X)M_M\ , \quad (3.10)$$

with:

$$G(X) = C_1 x_2^2 x_4 \sin(2x_3) + 2gC_2 \sin(x_3)x_4 + C_2 R x_2 x_4^2 \sin(x_3)$$
$$H(X) = x_2$$

As before, one wants to find a control which makes $\dot{V}$ negative semi-definite. According to (3.9), a solution is therefore to set $\dot{L}(X)$ to:

$$\dot{L}(X) = -q(x)(L(X) - L_0),\ q(x) \geq 0 \tag{3.11}$$

such that $\dot{V}$ becomes:

$$\dot{V}(X) = -q(X)(L(X) - L_0)^2 \tag{3.12}$$

The control allowing this is found by equaling (3.10) with (3.11):

$$\begin{aligned} M_M &= \frac{-q(X)(L - L_0) - G(X) - H(X)B_n(X)}{H(X)} \\ &= -q(X)\left[\frac{1}{2}C_1 x_2 \sin^2(x_3) + \frac{1}{2}(C_3 + J)x_2 - C_2 R x_4 \cos(x_3)\right] \ldots \\ &+ \frac{1}{x_2}\left[-q(X)\left(\frac{1}{2}C_1 x_4^2 + gC_2(1 - \cos(x_3))\right) - 2gC_2 \sin(x_3)x_4\right] \end{aligned}$$

The problem happening here is the same as before : M_M is not defined for

$$H(X) = x_2 = 0$$

Moreover, setting M_M to an arbitrary value for $x_2 = 0$ (so that it doesn't become infinite at the singularity) would not solve the problem, as $\dot{V}$ is equal to:

$$\dot{V} = 2gC_2\left[\frac{1}{2}C_1 x_4^2 + gC_2(1 - \cos(x_3))\right]\sin(x_3)x_4$$

at the singularity, and is obviously of variable sign (depending on x_3 and x_4). Finally, one can't find a control such that $\dot{V}$ is always negative semi-definite. An other approach has therefore to be investigated.

3.2 A Simpler Approach: LQ Control

Linearization and LQR

Here, the simpler approach, consisting in linearizing our model around the upright position and applying a Linear Quadratic control on the linearization was used. Linearizing our model around the point

$$\left[\varphi = \varphi_{ref}, \dot{\varphi} = 0, \theta = 0, \dot{\theta} = 0\right]$$

gives:

$$\dot{x} = A_{lin}\, x + B_{lin}\, u \tag{3.13}$$

with:

$$A_{lin} = \frac{1}{C_1(C_3+J)-C_2^2R^2}\begin{bmatrix}0 & 1 & 0 & 0\\ 0 & 0 & C_2^2Rg & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & gC_2(C_3+J) & 0\end{bmatrix}, \quad (3.14)$$

$$B_{lin} = \frac{1}{C_1(C_3+J)-C_2^2R^2}\begin{bmatrix}0 & 0\\ C_1 & C_2R\\ 0 & 0\\ C_2R & C_3+J\end{bmatrix} \quad (3.15)$$

$$u = \begin{bmatrix}M_M+M_F\\ -M_{FP}\end{bmatrix} \quad (3.16)$$

When M_F (friction moment applied to the motor) and M_{FP} (friction moment of the pendulum joint) are not taken into account, the system becomes:

$$\dot{x} = \frac{1}{C_1(C_3+J)-C_2^2R^2}\left[\begin{pmatrix}0 & 1 & 0 & 0\\ 0 & 0 & C_2^2Rg & 0\\ 0 & 0 & 0 & 1\\ 0 & 0 & gC_2(C_3+J) & 0\end{pmatrix}x + \begin{pmatrix}0\\ C_1\\ 0\\ C_2R\end{pmatrix}M_M\right] \quad (3.17)$$

Verifying the Properties of Article [Robertsson *et al.*, 2004]

From LQ control theory (see [Alazard, 2002]) one knows that a gain K_c can be easily found such that the cost criterion is minimized and the state feedback control

$$u = -K_c\, x$$

achieves uniform asymptotic stability in a certain region Ω. Hence, the first hypothesis of Theorem 1.1 is fulfilled, with Ω being the region where the linearization "holds". The only remaining hypothesis is the existence of a non negative function V verifying properties (1.12),(1.13) and (1.14).

Let V derived from the computation of the LQR.[1] V is such that:

$$V(X) = X^T P_c X, \quad (3.18)$$

and the first two conditions (1.12) and (1.13) hold. Indeed, the zero level of V consists in only the origin point, which belongs to Ω and, as V is not time-varying, inequality (1.13) is fulfilled by choosing for example :

$$V_m = V$$

So the only remaining condition to prove is the inequality on $\dot{V}$ (1.14).

V being the closed loop Lyapunov function of the stabilized system, it verifies (see again [Alazard, 2002]:

$$(A-BK_c)^T P_c + P_c(A-BK_c) + (Q_X + K_c^T R K_c) = 0 \quad (3.19)$$

[1] For computing the LQ control, a solution of a Riccati equation has to be found and this solution leads to a Lyapunov function

and therefore its time derivative writes:

$$\dot{V}(X) = -X^T(Q_X + K_c^T R K_c)X \ ,$$

and can be rewritten as:

$$\dot{V}(X) = -\begin{bmatrix} X \\ -K_c X \end{bmatrix}^T \begin{bmatrix} Q_X & 0 \\ 0 & R \end{bmatrix} \begin{bmatrix} X \\ -K_c X \end{bmatrix} = -\begin{bmatrix} X \\ u \end{bmatrix}^T Q \begin{bmatrix} X \\ u \end{bmatrix}$$

with R being positive. Condition (1.14) is therefore fulfilled.

So friction can now be considered and then compensated by applying the results of [Robertsson *et al.*, 2004], i.e. our LQ control can be modified as follows:

$$u_{LQ} = -K_c X \ \rightarrow \ u_{new} = u_{LQ} + \hat{F}$$

where the friction estimate $\hat{F}$ is reconstructed with the LuGre observer (1.18), and K chosen as in (1.20) such that :

$$K = \frac{2\left(H(X,t)Q_2 + Q_{12}X\right)\left(\sigma_0 + \sigma_1\phi(\dot{\phi})\right)}{\rho - 4Q_2\sigma_1\sigma_0}$$

with: $H(X,t) = u_{LQ} = -K_c\,X, \quad Q_2 = R, \quad$ and $\quad Q_{12} = \mathbb{O}_4$

4. Methods for Implementation and Evaluation

4.1 Hardware Description

In order to be efficient and not to spend a lot of time in programming, a dSPACE[1] is used. This device is a complete real-time development kit. It consists in a board equipped with a Power PC PPC 750GX CPU with 1Ghz CPU clock and is mounted in an extension box in order to render its manipulation safe. Moreover this device is really easy to interface with any processes thanks to the Connection Panel which provides an easy connection to all the input/output of the dSPACE board.

One of the advantages of using a dSPACE is that the real-time task is implemented in the processor and after that the CPU is totally independent from the host computer. This allows to have the full control of the sample time without any disturbance from the OS of the host computer. It is well known that real time programming is not easy as it requires high skills in programming. Using the dSPACE device allows not to worry about all these practical aspects (interrupts, background task, concurrent programming, etc...). Indeed, the user just needs to draw a Simulink diagram, which is then automatically compiled into C-code and transferred to the CPU of the board.

Nevertheless the use of the ControlDesk[2] software is not so straightforward as the version of the dSPACE device is a little bit old[3]. The software has lots of bugs[4] and is a little bit hard to use especially the first time[5] but it is really a small trouble, it would be harder to do the same thing without using the dSPACE.

4.2 Controller Used

This section describes how the controller and the friction compensation are implemented. As seen in section 3.2 page 23 the controller is an LQR and is based on a linearized model of the pendulum at the upright position (one of the equilibrium points of the system). The friction compensation is implemented as in the article [Robertsson *et al.*, 2004], the friction estimate being just added to the control signal delivered by the LQR :

$$u = u_{LQ} + \hat{F} \tag{4.1}$$

with :

[1] dSPACE is a trademark of dSPACE GmbH

[2] Copyright 2005, dSPACE GmbH. All right reserved

[3] The dSPACE in use is the version 5.0

[4] Never installed windows XP SP3, in that case it will be impossible to modify a previous saved version of a layout

[5] A small tutorial has been realised in order to get started easily with the software, the tutorial is available in the appendix A

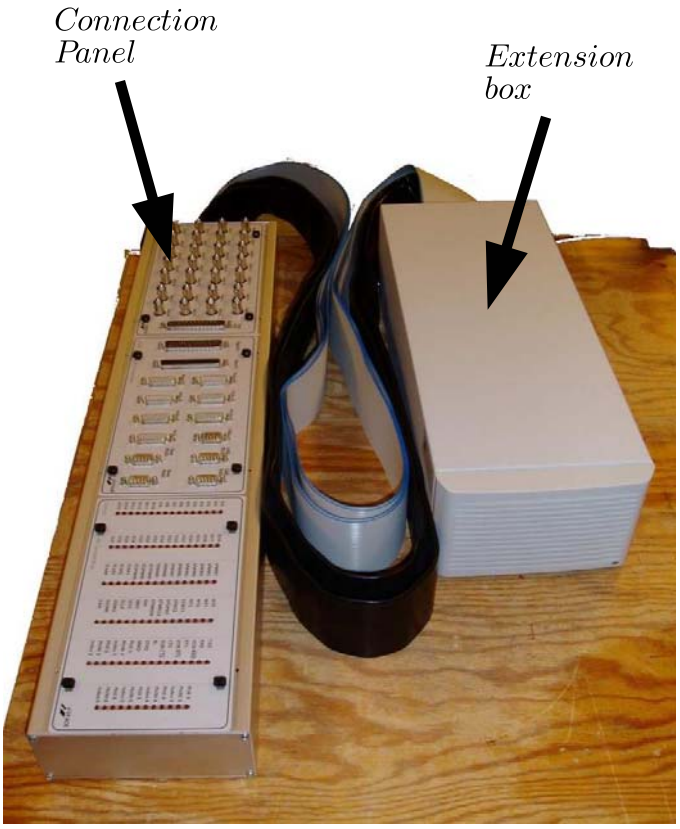


Figure 4.1 dSPACE extension box and connection panel.

u	:	Final control signal
u_{LQ}	:	Control signal given by the LQ controller
$\hat{F}$	:	Friction compensation term

As the model of the pendulum is linearized around the upright position, the LQR controller can only be applied in a neighbourhood Ω of this particular point. In order to have an estimation of the largest neighbourhood where the LQR applies, a simulation has been done. This simulation just tried lots of initial values around the upright position and saved the position for which the control lead to a stable position. This method allows to find the region of attraction of the system. But the fact is that when an initial condition did not lead to an equilibrium point Matlab just crashed. So it was hard to know if it was because the initial condition was bad or because Matlab encountered a numerical problem due for example to an inappropriate solver. Due to this problem, the solution chosen for finding the neighbourhood was to try a simple neighbourhood with simulations and with the real process. So the set Ω which is used is the following :

$$\Omega = -0.3 \leq \theta \leq 0.3 \tag{4.2}$$

It occurs that with this set the pendulum has a good behaviour. One can argue that the velocity could be taken into account, but it was hard to find a criterion to include it in the set. And as the current set works fine none other set was tried.
Now the question is how to get the pendulum around the upright position in order to be able to switch to the LQR. A straightforward idea could be to put by hand the pen-

dulum in Ω and just start the experiment. It occurs that this method did not yield to the stabilization of the system. A simple explanation could be that the initial velocity of the pendulum and the arm are equal to zero in this case. Consequently it could be too hard for the controller to maintain the pendulum near the upright position and for zero velocities by itself. So another way of doing it is to use the swinging-up method. As a lot of literature about swinging up control is already available, it seemed not necessary to describe the used swinging up strategy, which is the same as the one applied in [Åström and Furuta, 1996].

Now, thanks to this non-linear control, it is possible to reach the neighbourhood of the upright position and to switch to LQ control.
In the figure 4.2 the implementation is summarized throughout a Simulink file.

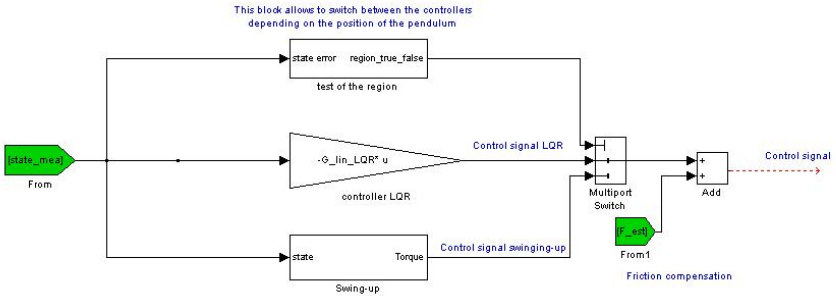


Figure 4.2 Simulink implementation of the controller

4.3 Implementation of the LuGre Model

As previously seen in section 1.4 page 9, the friction estimate $\hat{F}$ (which will be added to the initial control in order to overcome friction) is computed using the following Lugre observer (1.18):

$$\begin{cases} \hat{F} &= \sigma_0\hat{z} + \sigma_1\dot{\hat{z}} + F_{\dot{\varphi}}\dot{\varphi} \\ \dfrac{d\hat{z}}{dt} &= \dot{\varphi} - \sigma_0\dfrac{|\dot{\varphi}|}{g_{fr}(\dot{\varphi})}\hat{z} + K \end{cases} \tag{4.3}$$

In order to be implemented on the real experiment, this observer has to be discretized. As seen in [Freidovich *et al.*, 2006], a discretization of the internal state estimate $\hat{z}$ using a Forward Euler method can lead to numerical instability for high angular velocities $\dot{\varphi}$. Here a better alternative, also presented in [Freidovich *et al.*, 2006], is adopted.

First the internal state equation can be written:

$$\frac{d\hat{z}}{dt} = b(\dot{\varphi}) - a(\dot{\varphi})\,\hat{z}, \tag{4.4}$$

with:

- $a(\dot{\varphi}) = \sigma_0\dfrac{|\dot{\varphi}|}{g_{fr}(\dot{\varphi})}$.

- $b(\dot{\varphi}) = \dot{\varphi} + K$.

Then, assuming as in [Freidovich *et al.*, 2006] that the coefficients $a(\dot{\varphi})$ and $b(\dot{\varphi})$ are constant during one sample time, an analytic solution can be found for the internal friction state, and the LuGre discretized observer writes:

$$\begin{cases} \text{For } a \neq 0\text{:} \\ \quad \hat{z}_{k+1} = e^{-a_k T}\hat{z}_k + \frac{b_k}{a_k}(1 - e^{-a_k T}) \\ \text{and for } a = 0\text{:} \\ \quad \hat{z}_{k+1} = b_k T + \hat{z}_k \end{cases} \tag{4.5}$$

with:

- $a_k = \sigma_0 \frac{|\dot{\varphi}_k|}{g(\dot{\varphi}_k)}$.
- $b_k = \dot{\varphi}_k + K$.
- T: Sampling time.

Throughout this report, the observer implementing this recursion will be called the "discretized Lugre observer".

Moreover, this observer was also implemented in a slightly different form, by using as advised in the article a new state variable $z_{new} = \sigma_0\, z$:

$$\frac{d}{dt}\hat{z}_{new} = \sigma_0\dot{\varphi} - \sigma_0 \frac{|\dot{\varphi}|}{g_{fr}(\dot{\varphi})}\hat{z}_{new} + \sigma_0 K \tag{4.6}$$

$$\hat{F} = \hat{z}_{new} + \frac{\sigma_1}{\sigma_0}\hat{z}_{new} + F_v\dot{\varphi} \tag{4.7}$$

(4.8)

$$g(\dot{\varphi}) = \begin{cases} F_{c+} + (F_{s+} - F_{c+})e^{-(\dot{\varphi}/v_S)^2} & \text{for } \dot{\varphi} > 0 \\ F_{c-} + (F_{s-} - F_{c-})e^{-(\dot{\varphi}/v_S)^2} & \text{for } \dot{\varphi} < 0 \\ \frac{1}{2}\left(\lim_{\dot{\varphi}\to 0^+} g(\dot{\varphi}) + \lim_{\dot{\varphi}\to 0^-} g(\dot{\varphi})\right) & \text{for } \dot{\varphi} = 0 \end{cases} \tag{4.9}$$

$$\sigma_1(\dot{\varphi}) = \sigma_1 e^{-(\dot{\varphi}/v_d)^2} \tag{4.10}$$

The discretization is done in the same way as previously, and leads to the same recursion (4.5). But now a_k and b_k are equal to:

- $a_k = \sigma_0 \frac{|\dot{\varphi}_k|}{g(\dot{\varphi}_k)}$.
- $b_k = \sigma_0\,(\dot{\varphi}_k + K)$.

Moreover, equation (4.9) shows the implementation of the Stribeck function $g(\dot{\varphi})$, which, contrary to the original LuGre model, uses distinct parameters for positive and negative relative velocities.

An advantage with this new representation is that it is less sensitive to numerical errors, since z_{new} is of higher order of magnitude.

For this reason the friction observer will be computed following this recursion, and will be, throughout the report, named as the "discretized Lugre observer with the Freidovich parametrization".

4.4 Framework Used for the Simulations and for Experiments on the Real Process

General Remarks

For the implementation of the friction compensation, a Simulink diagram was built. This model contains input (and output) blocks to read (and write) data from (and to) the Furuta pendulum for the real experiment, whereas for the simulations without hardware in the loop these blocks are replaced with a model of the Furuta pendulum with friction included.

A special attention was paid so that this Simulink model be the same for the simulations and for experiments on the real process. Moreover both of the simulations and the experiments were done using as set point the zero reference.

For the real experiments, the control scheme and the friction compensation are designed in discrete time (with a sample time of 0.001 s) so that no solving by the dSPACE process be involved. Further the implementation of the friction compensation schemes is done using the LuGre discretized form (4.5) with the Freidovich parametrization .

For the simulations without hardware in the loop, two different approaches are considered. The "real model" of the Furuta pendulum being highly nonlinear, there is no choice for both approaches but to use a stiff solver with a variable step size ("ode23s Stiff. Mod. Rosenbrock") .

In the first approach, the compensation of the friction is implemented again following the LuGre discretized form (4.5) with the Freidovich parametrization.

The computation scheme is thus hybrid, with the model of the plant and the internal model of the friction being solved with a variable step size solver, while the friction compensation and the control scheme are solved in discrete time with a fixed sample time.

In the second approach the friction compensation is now implemented in continuous time by using directly the equations of the LuGre model. Hence everything is solved using the ode23s solver. This method may give a better friction estimate than the one computed with the first approach, because it uses the continuous LuGre form, and not a discretization.

One could argue that for these reasons this approach should be preferred. Nevertheless, designing the friction compensation and the control scheme in discrete time – as it is done in the hybrid approach – has the advantage that no continuous time solving is required by the dSPACE, so that finally one does not have to worry when the Simulink model will be used for experiments on the real process.

Simulations without hardware in the loop will thus be done using this hybrid approach, while the second approach may be used in case numerical problems would arise with the discretized LuGre observer.

Framework for the Computation of the Lyapunov Functions

The initial Lyapunov function of the system without friction, V, is computed the same way for experiments in Simulation and on the real process. As soon as the pendulum is swung up and caught by the LQ control, V is calculated according to (3.18).

The framework for the computation of

$$W = V + \frac{\rho}{2}(z-\hat{z})^2$$

the extended Lyapunov function, is much harder to obtain. Indeed, it is not hard to compute V nor to obtain $\hat{z}$, but it is impossible to have access to z, the internal state of the LuGre model.

For the simulations without hardware in the loop, it is logical to choose z as the state computed by the internal friction model used in the plant.

For the experiments on the real process, two different approaches are considered. The first one is done off line, and is based on the idea of computing z using a more accurate solver than the one used to compute $\hat{z}$. z is therefore computed afterwards by integrating the continuous LuGre equations with the same stiff solver as the one used for the friction model in the plant. This computation thus requires that the states, the Lyapunov function V and the internal friction estimate $\hat{z}$ be stored during the experiment.

The second approach allows an on line computation of W by finding a recursion for e, the internal friction state error. Recall the expression of $\dot{e}$ given in (1.22):

$$\dot{e} = -\sigma_0 \frac{|\dot{\varphi}|}{g_{fr}(\dot{\varphi})} e - K \tag{4.11}$$

The method of discretization used here relies on the same idea which was proposed in [Freidovich *et al.*, 2006] in order to discretize the dynamic of the internal friction state z.

Indeed, in a same way equation (4.11) can be rewritten in the new variable

$$e_{new} = \sigma_0 \, e$$

so that numerical errors are limited:

$$\frac{d}{dt} \, e_{new} = b(X) - a(\dot{\varphi}) \, e_{new},$$

with:

- $a(\dot{\varphi}) = \sigma_0 \dfrac{|\dot{\varphi}|}{g(\dot{\varphi})}$.
- $b(\dot{\varphi}) = -\sigma_0 \, K$.

Then, one assumes that the coefficients $a(\dot{\varphi})$ and $b(\dot{\varphi})$ are constant during one sample time, so that an analytic solution can be found. Finally, the discretization of the internal friction state error writes:

$$\begin{cases} \text{For } a \neq 0\text{:} & & \\ \quad \hat{e}_{new}^{(k+1)} & = & e^{-a_k T} e_{new}^{(k)} + \frac{b_k}{a_k}(1 - e^{-a_k T}) \\ \\ \text{and for } a = 0\text{:} & & \\ \quad e_{new}^{(k+1)} & = & b_k T + e_{new}^{(k)} \end{cases} \tag{4.12}$$

with:

- $a_k = \sigma_0 \frac{|\dot{\varphi}_k|}{g(\dot{\varphi}_k)}$.

- $b_k = -\sigma_0 K$.
- T: Sampling time.

With this recursion, W can therefore be computed while real experiments are being run. For this reason, this approach will be preferred to the off line one, and will be used to compute W for the experiments on the real process.

5. Simulation in Matlab Simulink

This chapter intends to document observations concerning the simulations which were run in Matlab Simulink in order to verify the results of [Robertsson *et al.*, 2004]

5.1 General Setup

For the simulations, the model of the friction in the plant is implemented using the continuous LuGre model equations (1.7) , (1.8), (1.9) and (1.10), and is solved with a variable step size solver. The choice of the solver has critical influence on the execution time, and the quality of the solutions because numerical problems arise with some of the built-in Simulink solvers.

Name	ode 3/4/5	ode45	ode23s	ode23s
Sample time (ms)	fixed: 1, 0.1, 0.01	variable	variable[a]	variable[b]
End of simulation [s]	crash	304.8	204.3	124.1

Table 5.1: details of solvers and related execution time for a simulation of 25*s*, performed with the first approach (see text below) and for the initial conditions [3533]

[a] without minimum step size bound
[b] with minimum step-size bound of 0.1 ms

Indeed, as shown in the above table, it was impossible to simulate the system with any fixed step size solver, even with a small sample time of 0.01 ms, while the commonly used variable step size solvers were extremely slow.

This is mostly due to the fact that friction dynamics are really fast, so that the friction model in the plant and the model of the plant form a stiff problem. Therefore the stiff solver ode23s with variable step size was chosen. Additionally, an inferior step size bound, with a relative tolerance for the numerical errors were also given, as it was observed that simulations were stuck when the arm velocity approached the zero (velocity reversal).

For friction compensation the discretization (4.5), with the parametrization made in [Freidovich *et al.*, 2006] and with a constant damping term σ_1, is used on the assumptions of perfect knowledge of the friction parameters (see Appendix C for the parameters values).

Thus, the simulation setup forms a hybrid system, with the model of the plant and the friction model in the plant being computed continuously (using the ode23s solver), while the friction compensation and the control schemes are computed in discrete time with a sample time of 1 ms.

Nevertheless, because of the fast friction dynamics, there may be some risks that the assumptions made to build the discretization be violated, so that finally the Lugre model would not be properly described by this discretization. This could thus result in an impossibility to verify, even in simulation, the results presented in [Robertsson *et al.*, 2004].

For this reason, a second approach, where the friction compensation is computed by solving the continuous LuGre equations with the same ode23s solver, was also implemented.

Of course it won't be possible to implement this approach on real experiments, because it would require that there exists an embedded stiff solver in the dSPACE, which is not the case. For this reason all the simulations will be implemented using the first approach, and if some problems arise with the aforementioned discretization, the second approach will be tested.

For the computation of the initial Lyapunov function V, the expression :

$$V = X^T P_c X$$

seen in (3.18), was used. P_c, solution of the closed loop Ricatti equation (3.19), is automatically obtained thanks to the discrete LQ design done in Matlab.

Moreover, the decrease of V can only be verified once the LQ control manages to grab the pendulum and stabilize it so that no swinging up control is involved anymore.

The initial condition was therefore set to:

$$\left[\varphi_0, \dot{\varphi}_0, \theta_0, \dot{\theta}_0\right] = X_{up} = [0, 0, 0.1, 0]$$

so that the LQ control be active from the beginning and never switch to the swinging up control.

For the computation of W, the ρ parameter value was set to $1e8$ so that condition (1.19) of Theorem 1.1 be fulfilled.

Moreover, the following setup was used:

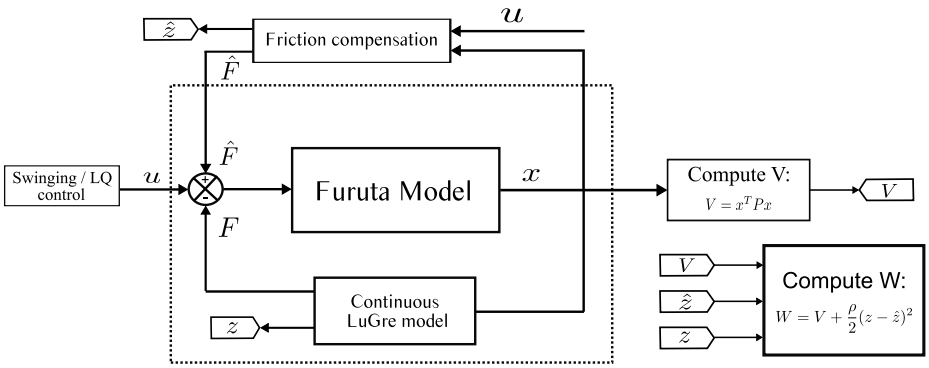


Figure 5.1 Setup used for the computation of W

Indeed, as the model of the plant and the internal friction model of the plant are solved with a variable step size solver while the friction compensation is computed according to the discretized LuGre form (4.5), the "exact" internal state z is computed by the internal friction model of the plant. Moreover, for the same reasons mentioned before, W was also computed setting to the same value X_{up} the initial condition.

Finally, the reference applied for all the simulations is the zero reference:

$$X_{ref} = [0, 0, 0, 0]$$

5.2 Simulations of the Plant Without Friction, and Computation of the Function V

This section presents the simulations which were carried out to check if the control described in section 3.2 page 23 achieves, in a region Ω, uniform asymptotic stability

for the system without friction, and if the candidate function V resulting from this control is a Lyapunov function.

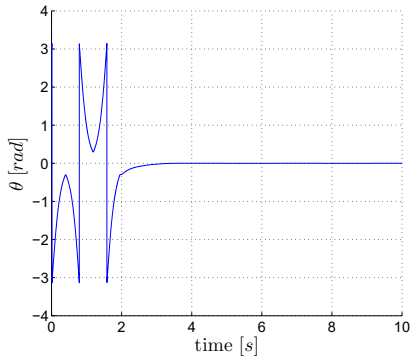


Figure 5.2 theta response for an initial condition of $[3, 5, 3, 3]$

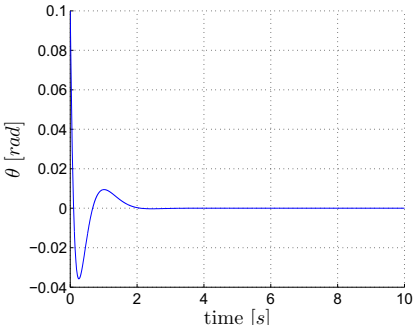
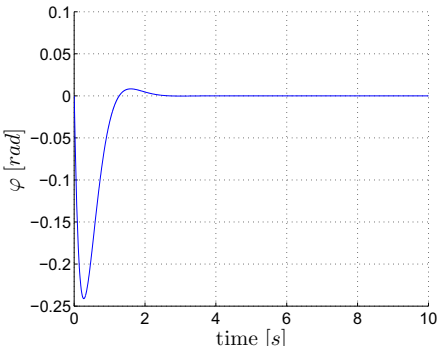


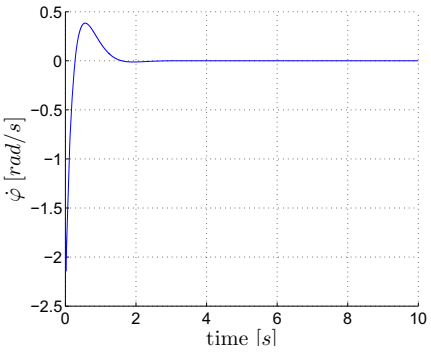
Figure 5.3 theta response for an initial condition of $[0, 0, 0.1, 0]$

Figure 5.2 shows the pendulum angle time response when initially the pendulum starts with nonzero velocities in the downright position, while in figure 5.3 the pendulum is close to the upright position.

When initially the pendulum is in the downright position, it is first brought towards the upright position thanks to the swinging-up control, and then caught by the stabilizing control, which manages to grab it and stabilize it if the velocities are not too high.

When now the pendulum is initially close to the upright position, the LQ controller manages to stabilize it from the beginning so that swinging up control never occurs.





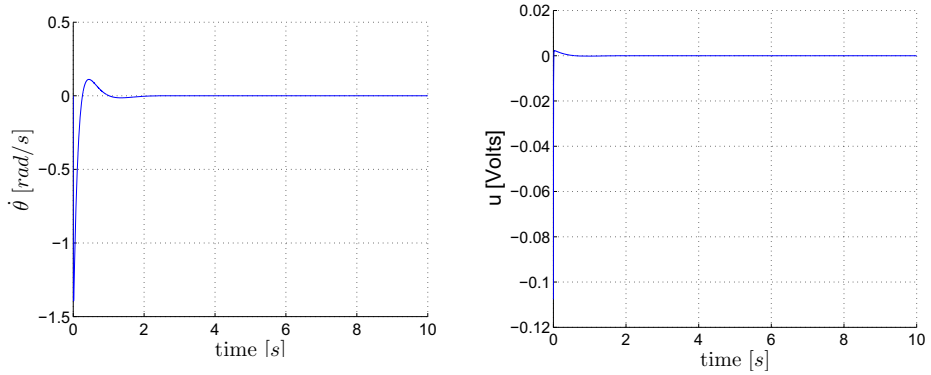


Figure 5.4 φ,$\dot{\varphi}$, $\dot{\theta}$, and the control time responses for $\left[\varphi_0,\, \dot{\varphi}_0,\, \theta_0,\, \dot{\theta}_0\right] = X_{up} = [0, 0, 0.1, 0]$

Figure 5.4 shows the time response of the other state variables, and of the control when the pendulum starts from this same initial condition X_{up}. The transients of course depend on the initial condition chosen, but globally if the initial velocities are taken to a reasonable value, the system is always stabilized in the upright position.

Finally, figure 5.5 shows as expected the decrease of the Lyapunov function V for the same initial condition.

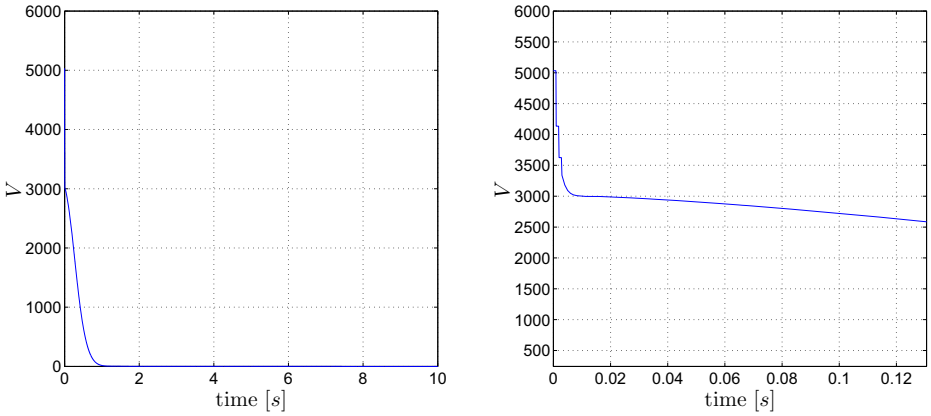


Figure 5.5 The Lyapunov function V, resulting from the applied LQ control, is decreasing. Simulations were carried for the initial condition: $\left[\varphi_0,\, \dot{\varphi}_0,\, \theta_0,\, \dot{\theta}_0\right] = X_{up} = [0, 0, 0.1, 0]$

5.3 Simulations of the Plant With Friction, and Computation of W

In this section the model of the Furuta pendulum now includes the friction model (See Figure 5.1)

Direct Effects of Friction Compensation

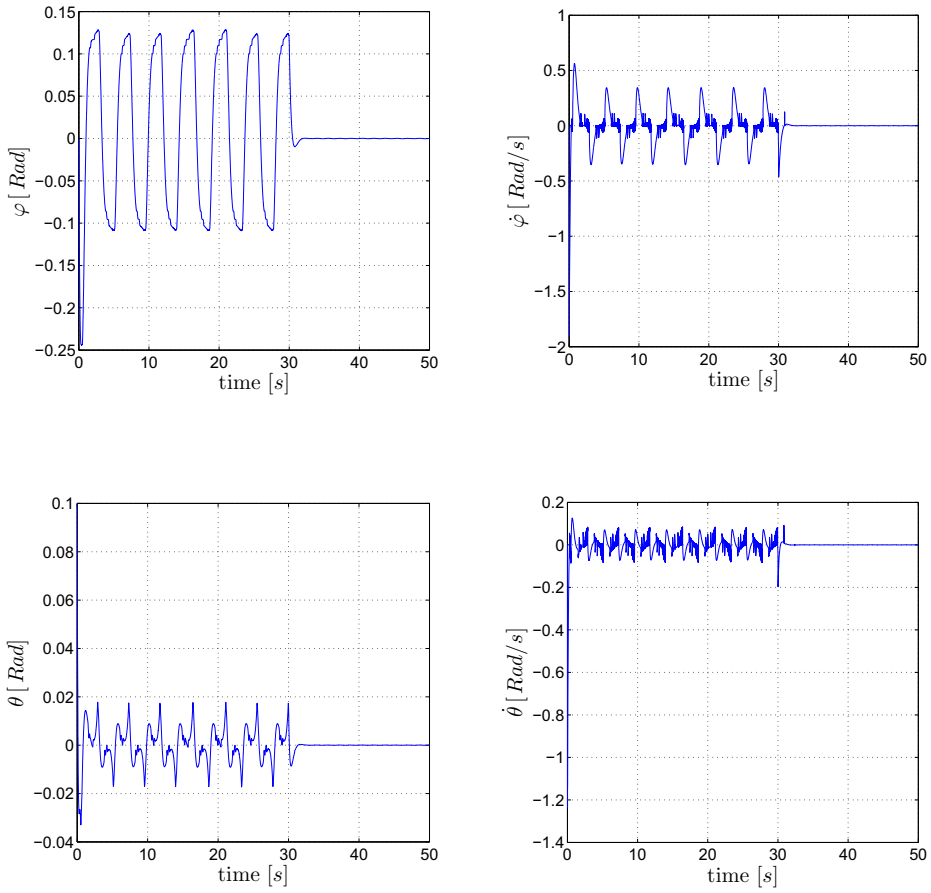


Figure 5.6 the four state variable response for $\left[\varphi_0,\, \dot{\varphi}_0,\, \theta_0,\, \dot{\theta}_0\right] = X_{up} = [0, 0, 0.1, 0]$. Friction compensation starts at $t = 30s$

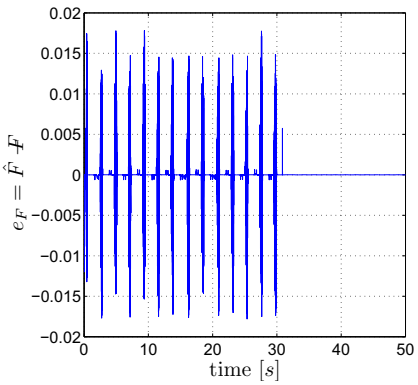


Figure 5.7 Friction error $e_k = \hat{F} - F$. The simulation setup is the same as in figure 5.6

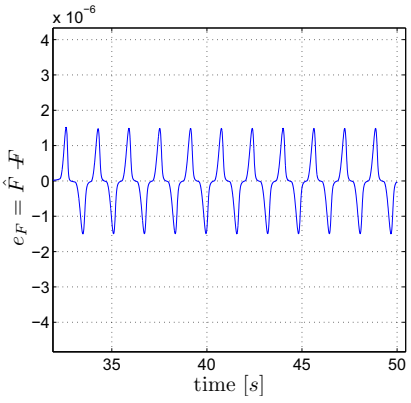


Figure 5.8 Zoom of friction error $e_k = \hat{F} - F$, from $t = 30\,s$ to $t = 50\,s$. The simulation setup is the same as in figure 5.6

Figure 5.6 shows the four state variables time response for the initial condition X_{up}. From $t = 0\,s$ to $t = 30\,s$ no friction compensation is done, and the typical limit cycle behaviour for friction uncompensated mechanical systems occurs. At t = 30s the

compensation is triggered out, and limit cycles are immediately annihilated, while the system quickly stabilizes to the given zero reference. The friction compensation scheme is therefore very efficient, as the system behaves as if no friction were acting on it once the compensation scheme has been turned on. This comes from the fact that the friction error

$$e_F = \hat{F} - F$$

keeps in a small order of magnitude as soon as friction compensation is applied, see Figures 5.7 and 5.8 . Indeed, $\hat{F}$ is really close from the "real" friction force F, because it is computed using the discretized version of the continuous LuGre model, actually used to simulate F.

Computation of W

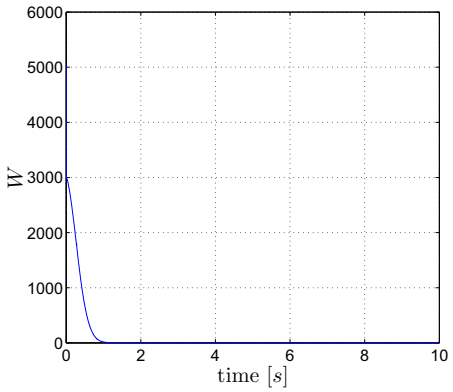


Figure 5.9 extended Lyapunov Function W

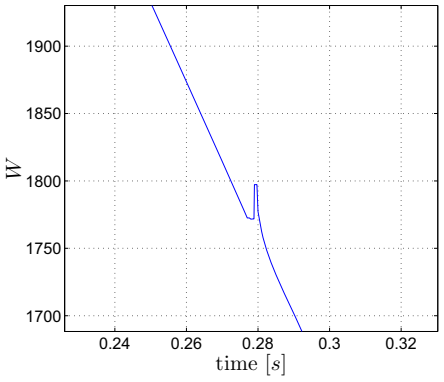


Figure 5.10 Zoom on the first peak of figure 5.9

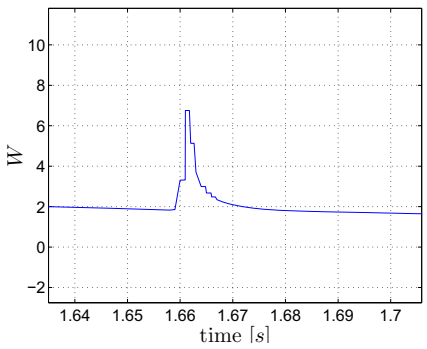


Figure 5.11 Zoom on the second peak of figure 5.9

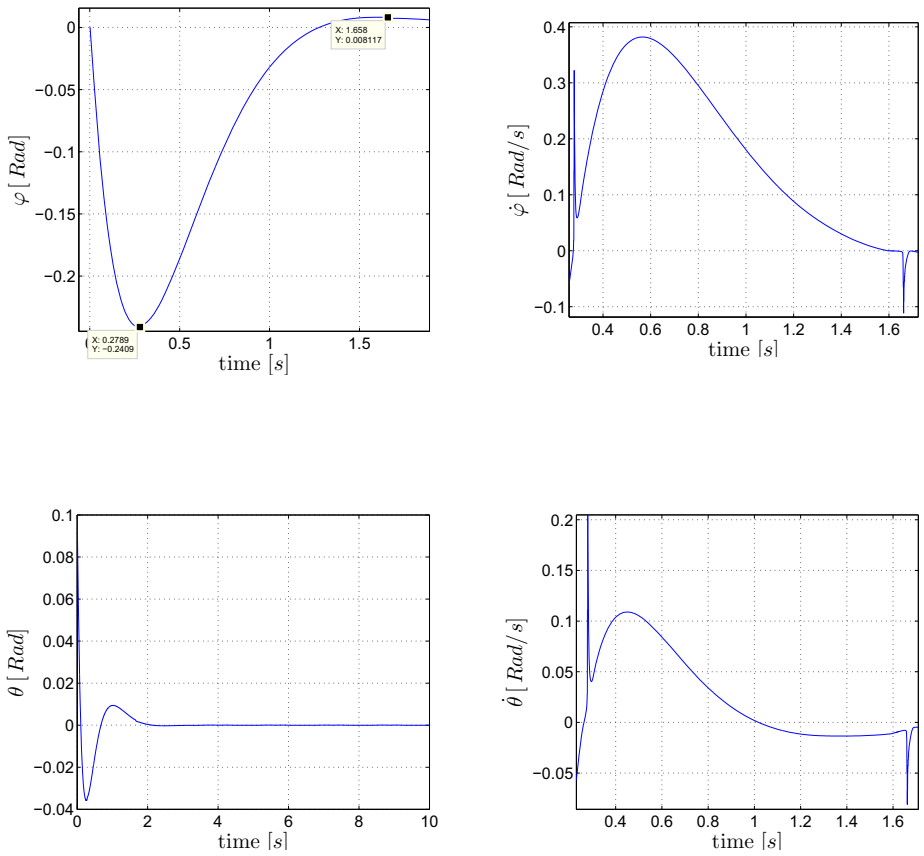


Figure 5.12 Peaking behaviour occurs for velocity reversal and is propagated through the state velocities

Figure 5.9 shows the extended Lyapunov function W when Friction compensation is applied. The obtained function is decreasing all the time, except for velocity reversals (change of the arm direction) where it presents a strange peaking behaviour, also propagated through the state velocities. See figure 5.10, 5.11 and 5.12.

This behaviour does not seem to be due to the fact that ρ was set to a wrong value.

Indeed, recall condition (1.19) of Theorem 1.1:

$$\rho > 4Q2\sigma_0\sigma_1\,,$$

which gives in our case a lower bound

$$\rho_{min} = 6.8\,e\,6$$

This condition has to be fulfilled, as it ensures that the obtained candidate function W will be decreasing.

Indeed, if ρ is chosen below this bound (see figure 5.13), oscillations appear, and W is no more decreasing, while a value above this limit roughly leads to the same results as in figure 1.21. However, a too high value of ρ would also lead to an amplification of the two previously mentioned peaks, see figure 5.14. For these reasons, ρ was therefore kept to the same value $1\,e\,8$.

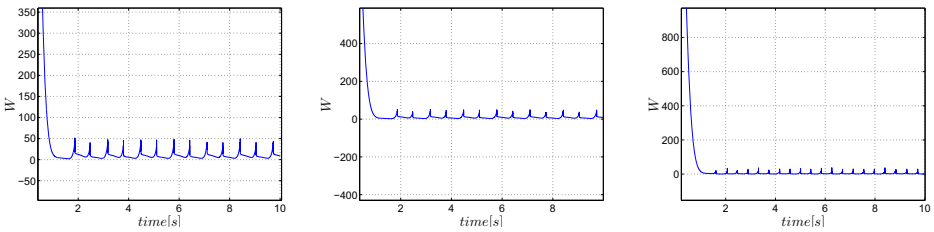


Figure 5.13 Extended Lyapunov function W for different values of ρ all below the lower bound ρ_{min}. Left: $\rho = 10$. Center: $\rho = 1000$. Right: $\rho = 1e5$

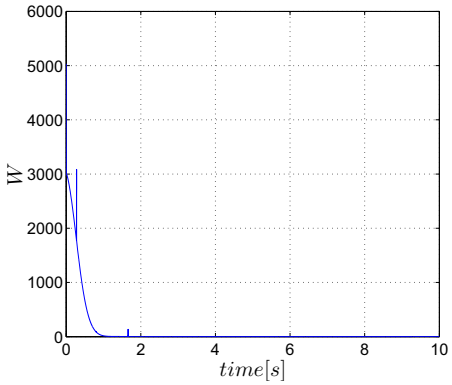


Figure 5.14 Extended Lyapunov function for a ρ value of $1e10$

Moreover, figure 5.15 suggests that the origin of this strange phenomenon relies on the computed friction estimate $\hat{F}$, which, contrary to the real friction F, takes a peak value at velocity reversals:

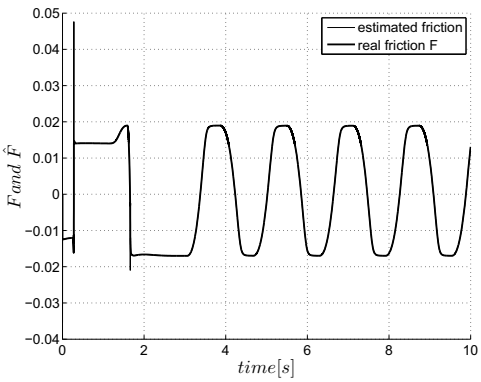


Figure 5.15 real friction F and its estimate $\hat{F}$

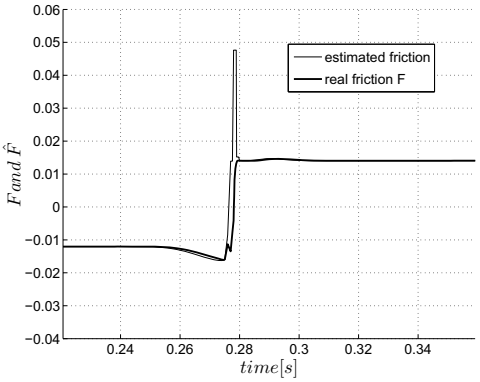


Figure 5.16 Zoom of figure 5.15 from $t = 0.22s$ to $t = 0.36s$

Therefore a problem apparently arises in the computation of $\hat{F}$ at velocity reversals, and then the discretized Lugre version (4.5) seems to be involved. This fact can be verified if now $\hat{F}$ is computed using the Lugre model equations (1.7) , (1.8), (1.9) and (1.10):

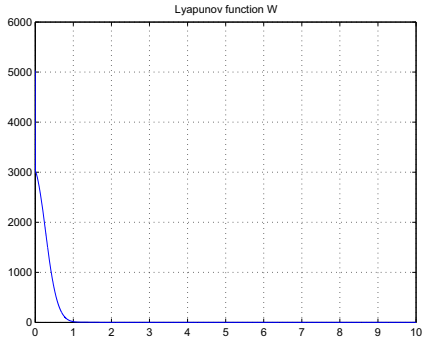


Figure 5.17 computation of W, $\hat{F}$ being computed using the "exact" continuous time LuGre model

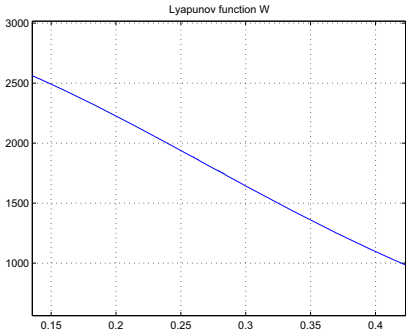


Figure 5.18 zoom of figure 5.17 where the first previous peak was located (see figure 5.10)

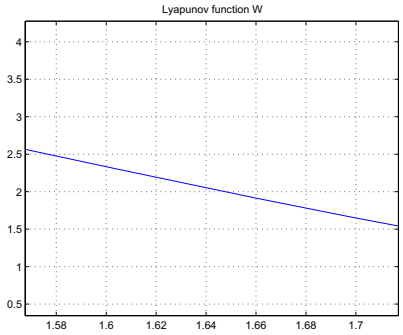


Figure 5.19 zoom of figure 5.17 where the second previous peak was located (see figure 5.11)

Indeed, with the continuous LuGre observer the peaks totally disappear, see figures 5.18 and 5.19, and W is, as expected, perfectly decreasing. Of course as said before it won't be possible to implement this setup, as no performant solver such as the one used here is available on the dSPACE. But at least this simulation is a good way to verify the results exposed in [Robertsson *et al.*, 2004], and it points out that the discretization of the LuGre model is responsible for this peaking behaviour.

A lot of time was passed on trying to identify what could possibly cause these peaks.

First, recall the continuous LuGre friction observer, with the Freidovich parametrization:

$$\begin{cases} \hat{F} &= \hat{z} + \dfrac{\sigma_1}{\sigma_0}\dot{\hat{z}} + F_{\dot{\varphi}}\dot{\varphi} \\ \dfrac{d\hat{z}}{dt} &= -\sigma_0 \dfrac{|\dot{\varphi}|}{g_{fr}(\dot{\varphi})}\hat{z} + \sigma_0(\dot{\varphi} + K) \end{cases} \tag{5.1}$$

and its discretization, done as in [Freidovich *et al.*, 2006]:

$$\begin{cases} \text{For } a \neq 0\text{:} \\ \quad \hat{z}_{k+1} = e^{-a_k T}\hat{z}_k + \frac{b_k}{a_k}(1 - e^{-a_k T}) \\ \\ \text{and for } a = 0\text{:} \\ \quad \hat{z}_{k+1} = b_k T + \hat{z}_k \end{cases} \tag{5.2}$$

with:

- $a_k = \sigma_0 \frac{|\dot{\varphi}_k|}{g(\dot{\varphi}_k)}$.

- $b_k = \sigma_0 (\dot{\varphi}_k + K)$.

Basically, two different problems arise with this discretization.

- First, the discretization can only be "perfect" if, during the chosen sample time, the coefficients a_k and b_k remain constant. As mentioned before, friction is a phenomena in which fast dynamics are involved, especially for the breakaway, and for velocity reversals. Hence, it is likely that a_k and b_k will change during the chosen sample time of 1 ms. Simulation were thus run, in which the sample time was reduced up to $1\,\mu$s. No significant improvement was observed. Instead, oscillations around the second peak appeared:

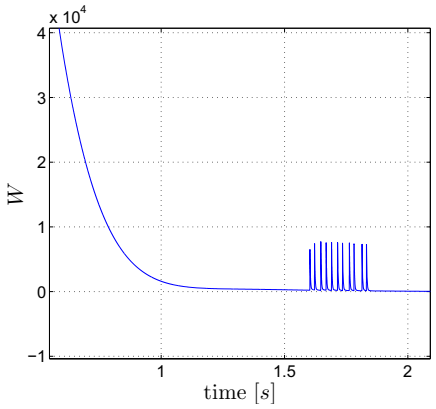


Figure 5.20 Extended Lyapunov function W computed with a sample time of 1e-6

- The fact that there are two different recursions, depending on the value of a_k. The switch between these two recursions theoretically happens for $a_k = 0$, i.e. at velocity reversals ($a_k = 0 \Rightarrow \dot{\varphi} = 0$). Of course for implementational reasons a threshold should replace this "zero switching value", otherwise with the first recursion and for a small a_k the internal state would "explode". The peaking behaviour happening for the previous simulations could thus be due to the chosen threshold of $1e-6$, which might be too low. But no significant improvements were seen with a threshold reduced to $1e-4$, while with $1e-2$ abnormal oscillations appear:

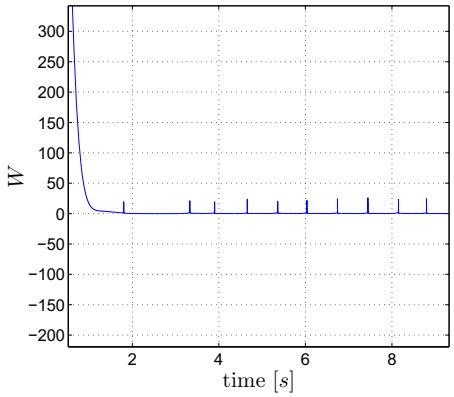


Figure 5.21 Extended Lyapunov function W computed with a sample time of 1e-3, and with a threshold of 1e-2

Hence, in spite of all the numerous simulations performed, no real way that could lead to obtain a perfectly decreasing function W with the Lugre discretization (4.5) was found. Nevertheless, the reason why these peaks appear is definitely due to the approximations which are made in this discretization.

It will thus be hard to verify the decrease of W for the real experiments, first because the friction compensation is implemented with this discretization, and then because other phenomena, such as noisy measurements are likely to happen.

6. Experiments

6.1 Introduction

In this section, the friction compensation strategy [Robertsson *et al.*, 2004] is applied to the real process. For being sure that the friction parameters are in steady state, the experiments are done after a run of at least five minutes which means that during each one the pendulum will be already in the upright position.

6.2 Reduction of the Limit Cycles

For this experiment the system is first controlled only by the LQR controller (figure 6.1 and 6.2), and in a second time the friction compensation is added to the control signal (figure 6.3 and 6.4).

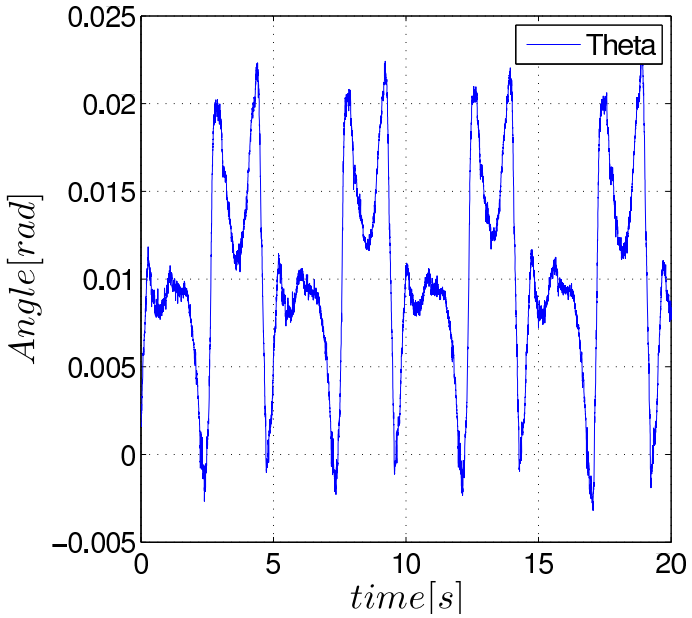


Figure 6.1 Measure of θ without friction compensation

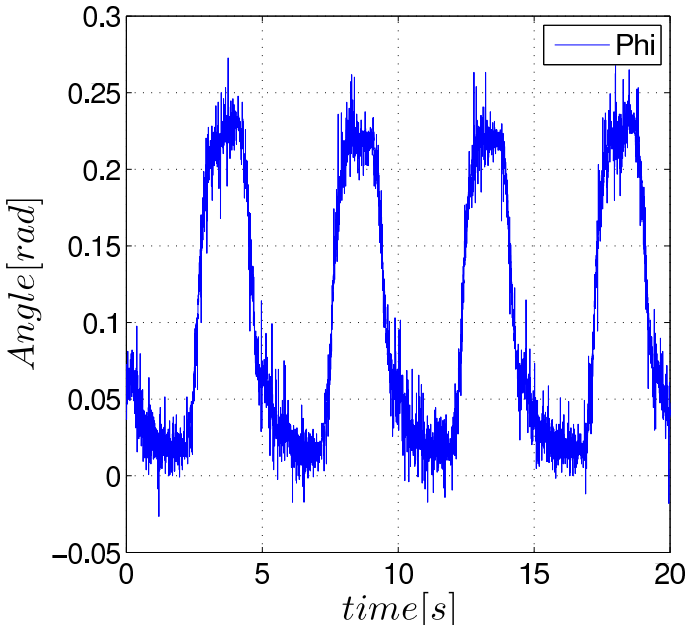


Figure 6.2 Measure of φ without friction compensation

Considering θ one can say that the system is well controlled because the deviation from the upright position is very small, but for the angle φ there is a static error and a limit cycle. The static error could be handled by an integrator and is not really the study point in this thesis. One can say that the amplitude of limit cycle is not so high, but depending on the application, it could be a problem. In order to try to reduce it, friction compensation is applied.

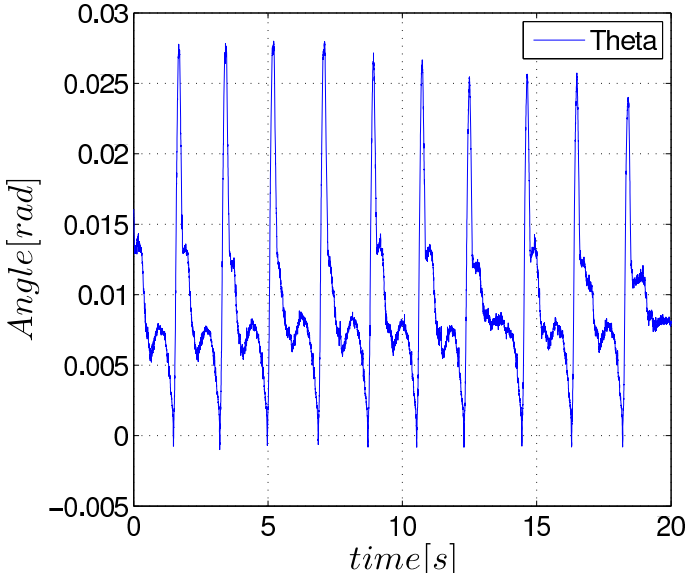


Figure 6.3 Measure of θ with friction compensation

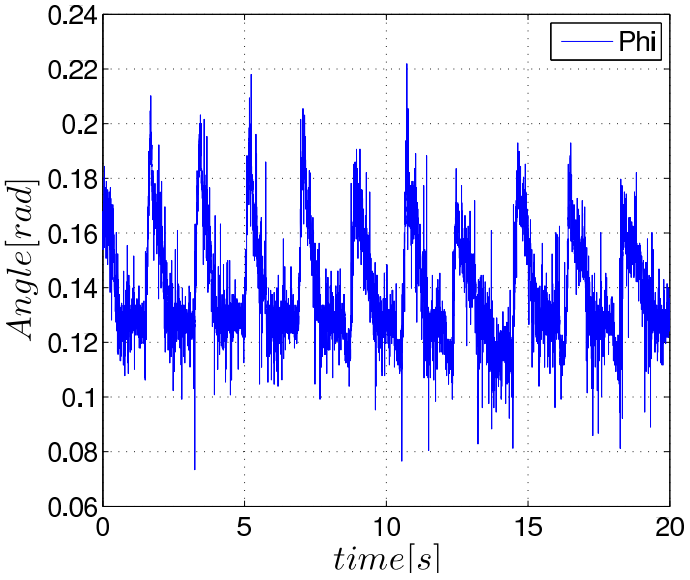


Figure 6.4 Measure of φ with friction compensation

Considering the angle φ, the result is better because the amplitude of the limit cycle is reduced but considering the angle θ the result seems to be less accurate. It is hard to compare noisy signals only by looking at it. So the covariance matrix of the error is used. The evaluation of such a matrix is done with Matlab, as it is a statistical criterion and the survey is focused on the limit cycles, the mean value is removed from the signals before performing the computation. The results are summarized in the following table.

Table 6.1 Performance of the friction compensation

	$var(\varphi - \varphi_{ref})$	$var(\theta - \theta_{ref})$
Without compensation	$6.95.10^{-3}$	$3.38.10^{-5}$
With compensation	$3.47.10^{-4}$	$2.92.10^{-5}$

As it has been assumed by looking at the figures, the result is better for φ and is almost the same for θ. The gain for φ is very good because it is a gain of a factor ten. So from this point one can state that the friction compensation is useful.

6.3 Test of the Value of ρ

In the equation (1.19 page 14) it has been seen that ρ is a parameter of the observer and its value is bounded.

$$\rho_{min} = 6.8.10^{6} \tag{6.1}$$

Therefore the goal of this section is to verify or rather to test the value of this bound. Lots of experiments have been done with different values of ρ. The value of the first cross-term of the covariance matrix (the φ term) is used to compare the performance.

Table 6.2 Influence of the value of ρ

Value of ρ	$var(\varphi - \varphi_{ref})$
Without Compensation	$4.71.10^{-3}$
10	$1.22.10^{-3}$
1000	$9.35.10^{-4}$
1.10^{5}	$2.40.10^{-2}$
1.10^{6}	$4.13.10^{-4}$
1.10^{7}	$4.31.10^{-4}$
1.10^{8}	$5.03.10^{-4}$

Obviously the compensation works for any value of ρ but the performances are not identical. The worst case is for

$$\rho = 1.10^{5}$$

in this case the limit cycles are bigger than the original ones. But clearly for any value higher than the bound for ρ (or in the same order of magnitude) the performances are constant. Therefore the bound could be interpreted in experiments as a lower limit which guarantees a good behaviour of the compensation. This result is not so surprising because ρ is here to be sure that the Lyapunov function is decreasing, but as it will be seen in the next section, in experiments it is almost impossible. Then for the authors it is not so strange to not totally follow the theory.

6.4 Lyapunov Function

In this part, W, the extented Lyapunov function of the system, is plotted. There are two representations of the Lyapunov function: one during the transient and the stabilization of the pendulum and the other in steady state, when the pendulum is already stabilized. The goal of these two experiments is quite different,the first one is to check if the function is decreasing and the second one if the function is equal to zero in steady-state.

First Experiment

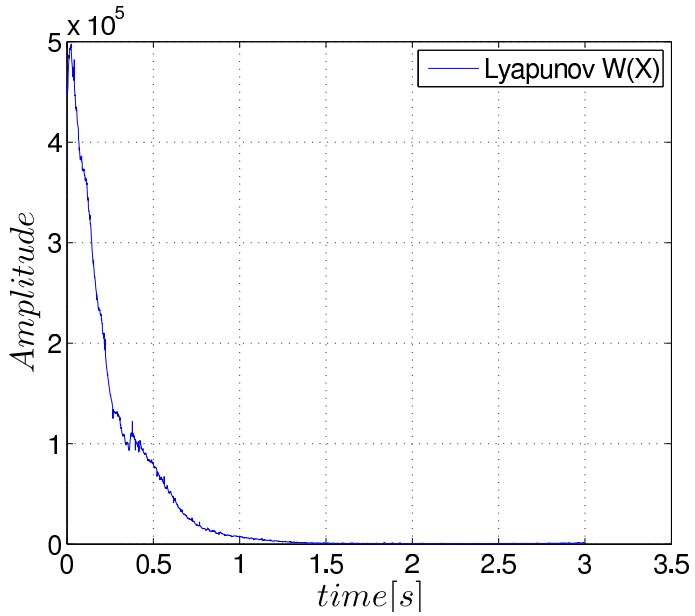


Figure 6.5 Lyapunov function W(X) during the transient

Considering figure 6.5, W is "globally" decreasing, as at the time $t = 0.35$ the function is slightly increasing. Anyway as it will be seen in what follows, one cannot expect to obtain a perfectly decreasing function, so, all in all, the obtained result is quite nice.

Second Experiment

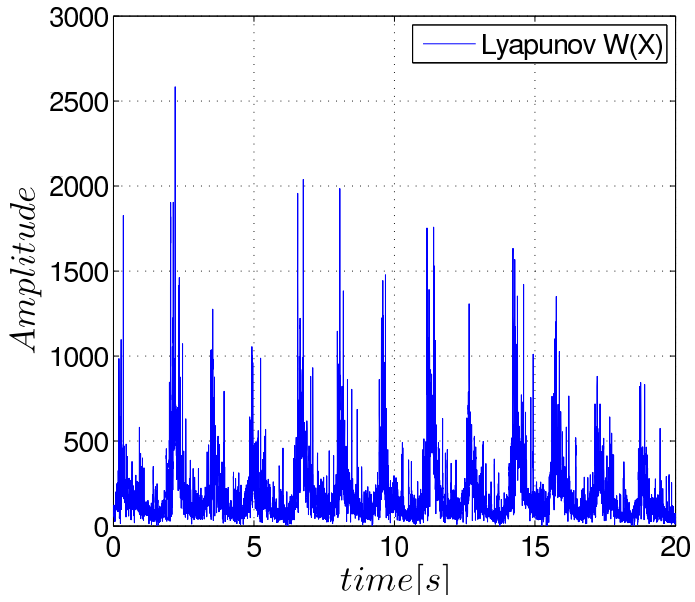


Figure 6.6 Lyapunov function W(X) in steady-state

Considering figure 6.6, one sees that W is not equal to zero in steady-state, but has some oscillations which corresponds to the limit cycles of the system.

Discussion

Referring to the first experiment, the results seem quite good because W is decreasing. However it is not strictly decreasing as it is with simulations without hardware in the loop. One can argue that the model of the process does not fit exactly the real process so it seems unrealistic to expect a perfectly decreasing function.
Nevertheless, what could be a little bit disappointing is the result for the second experiment: in steady-state W is far from being equal to zero. But the order of magnitude of $W(X)$ and of the oscillations are quite different. The first one is about 10^5 whereas the second one is 10^3 so there is a factor 100 between both of them. Therefore even if the amplitude of the oscillations seems significant it is nothing compare to the amplitude of the function $W(X)$. Then the function can be considered as positive definite and its derivative as almost negative definite.

Something could be investigated : it is the origin of such oscillations. The form of the function $W(X)$ is such that :

$$W(X) = V(X) + \rho e^2$$

So the equation has two terms, one depending on X and the other on ρe^2 which is the error of estimation of the friction force.

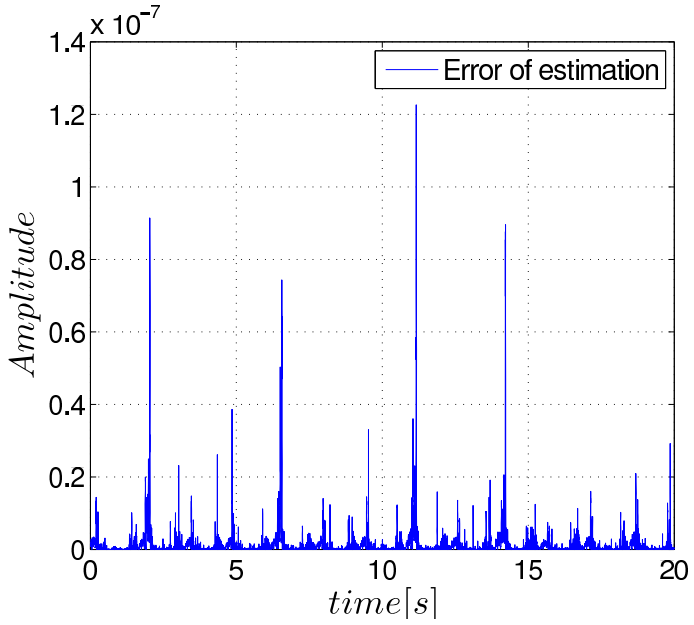


Figure 6.7 The $\rho e^2 term$ in steady-state

The figure 6.7 clearly show that the error is equal to zero in steady-state. This result was predictable as the estimation of the friction is done with an observer it is logical that the error tends to zero.
Moreover that means that the oscillations come from $V(X)$. The form of $V(X)$ is given in equation (3.18 page 24). Clearly the function depends on X, then steady state as X should tend to zero, $V(X)$ should tend to zero too. But as it is noticed in the section 6.2 even if friction compensation is applied there are still limit cycles so $V(X)$ can not be equal to zero in steady-state. And so does the function $W(X)$. An assumption could be made about the limit cycles, in simulation there are no limit-cycles using the ode23s solver, but here this kind of solver is not available for the dSPACE

device so it is not really possible to have an accurate estimation of the friction and this leads to remaining limit-cycles.
So it is clear that the link between simulation and experiment is hard to make, there are so many disturbances on reality that a mathematical tools such as Lyapunov functions is really hard to use. In this case the hypothesis for a such a function are not completely fulfilled but the authors think that the result obtained in figure 6.5 is quite sufficient to validate the theoretical part.

7. Conclusions

7.1 Conclusions About the Work

- The theory in the article has been verified in simulation and is correct, the friction compensation works well, the limit cycles are cancelled. And thanks to the specific gain for the friction observer a true Lyapunov function is obtained.
- In the same way, in experiments the effects of friction compensation have been verified. But the result is not as good as in Simulation. The limit cycles are only reduced and not cancelled. Moreover the condition on ρ is hardly verified and only one thing can be stated : the minimum value for ρ just guarantees that the compensation will work fine. But that does not mean that a lower value for ρ will not work.
- The previous conclusion about the performance in experiment leads to this other statement. A high precision compensation requires high precision estimation but this means accurate measures for the parameters which time and cost consuming. So for now this method seems to be very hard to implement in a serial product as it is unrealistic to estimate accurately the parameters for each part.
- One of the topic of the article is Lyapunov function, this theory is really powerful on the paper. Nevertheless the experiments show that it is quasi-impossible to survey a Lyapunov function in real-time because in a real process the perfect stabilization does not exist and there are always oscillations which lead to a non strictly decreasing function. So visualizing Lyapunov function on a real process is not really appropriate.
- Using simulink for simulation is not hard, but what is hard is to choose the right solver for the right model. Depending on the solver the results could be quite different or strange, so a particular attention has to be given for this choice. Otherwise it could be really easy to choose the wrong solver and to just notice it too late or at least to have lost a lot of time.
- The method used in this thesis would require an advanced solver for having good results. But for now the implementation of such algorithms is not straightforward on a traditional DSP or micro controller which is traditionally used for controlling a process.
- The files used in this thesis are well commented and clear, the authors hope that their work could be used easily by someone else.

A. General Tutorial for ControlDesk

Introduction : This tutorial is aimed at new users of the dSPACE and ControlDesk modules. The first part describes, first, the procedure to create a very simple experiment, and then how to open an existing experiment. The configuration of Simulink and ControlDesk is not explained in this part. But this will follow with the second part of the tutorial, which gives some information about the building in Simulink, and also some clues to different problems you may encounter. Please note that the purpose of this tutorial is not to replace the official ControlDesk guides. If you need further information, please thus refer to these manuals. However the authors found that these manuals were pretty dense to read, and were not able to find good tutorials. For all these reasons, they decided to write this tutorial, and hope it will be helpful for the readers.

Before doing anything you have to start the devices in a specific order :

1. Connect the process to the dSPACE I/O board.
2. Switch on the extension box.
3. Switch on the process.
4. Switch on the computer.

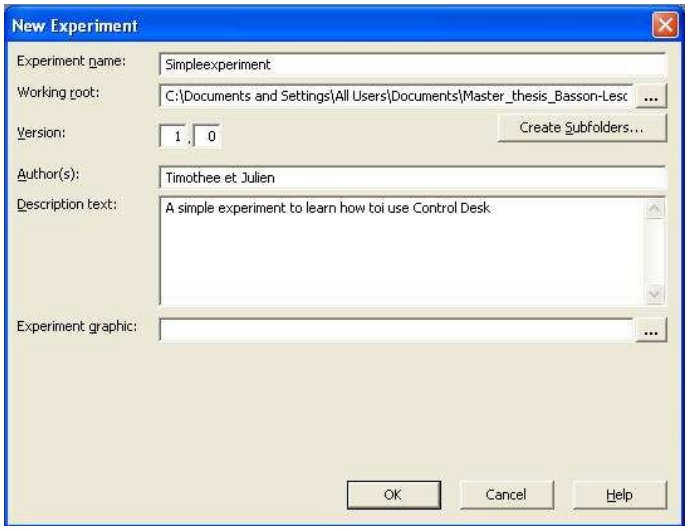
A.1 Creating / Opening an experiment

Creating a new experiment

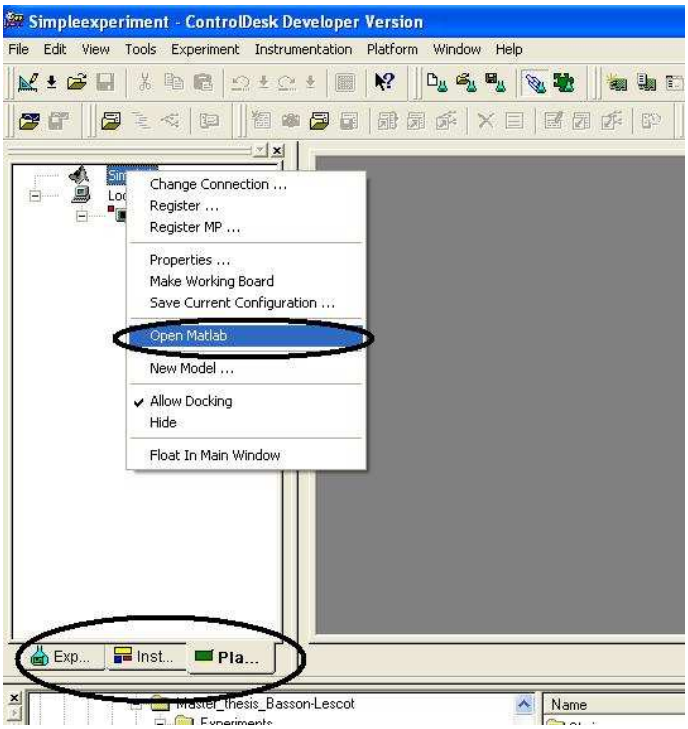
The considered experiment is very simple: it consists of sending a signal command and reading the output of the process.

1. Open ControlDesk 

2. You have to click on : File ⇒ New Experiment. A Dialog box appears, at least you have to give a name and a location for your experiment.

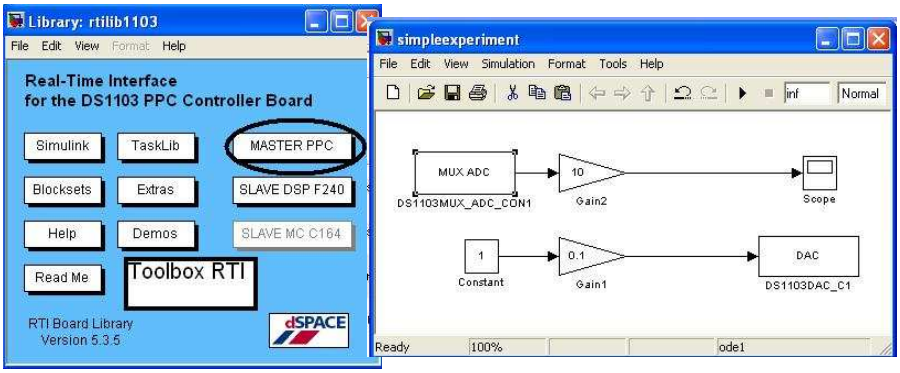


3. Launch Matlab inside ControlDesk, by clicking on the tab platform and right-clicking on the Matlab icon. The icon light ups.



4. After you have to create a new Simulink model. This is achieved by right-clicking on the Matlab icon and clicking on new model . The Matlab icon is now changed with the Simulink icon.

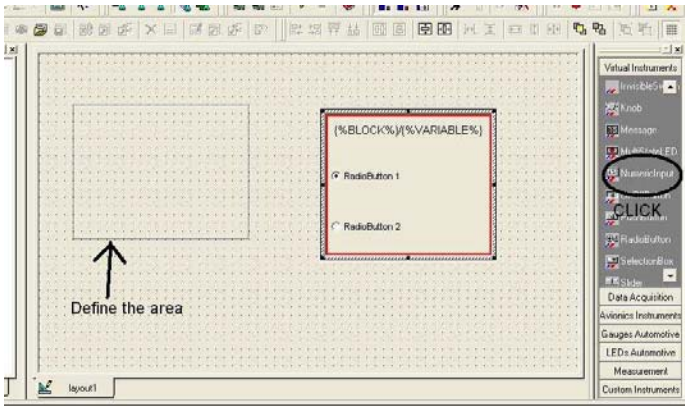
   Now you have to draw your model. In our case we need an ADC and an DAC. The toolbox where the devices for the dSPACE are located can be reached by typing rti in the Matlab prompt. The configuration of such devices is simple: you just have to double-click on the device and choose the channel. Moreover for the DAC you can choose its termination value. This is useful because if you set 0 for example when you will click on stop the process will stop, If you don't do that the permanent value on the DAC will be the last value before clicking on stop. Which means that even if the experiment stops, on the computer the device will keep running.



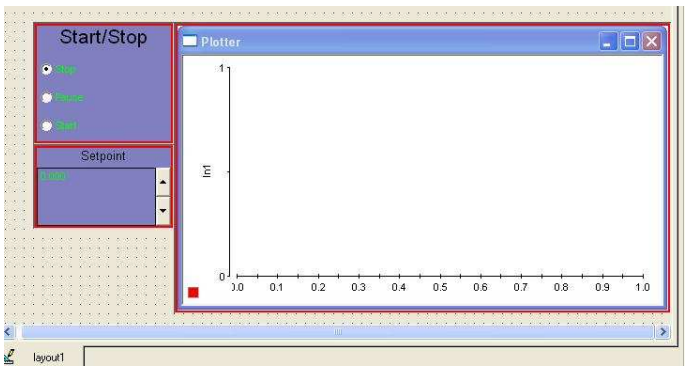
   The ADC and DAC are in the category MASTER PPC. You have to insert the two gains as shown above because you have a factor ten between the dSPACE and Matlab.(See Documentation)

5. The next step is the GUI. First you have to create a new layout. You just have to click on the layout icon as below:

6. After you have to add elements to your layout. To do so click on the desired element and after draw on the layout the area for your element.

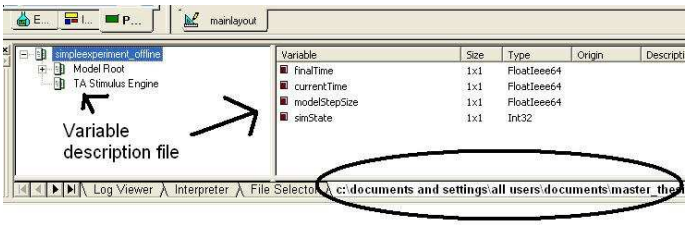


In our GUI we have three radio buttons, a plotter array (Category Data Acquisition), and a numerical input. The Start/Stop block allows us to start and stop the real time task. The plotter element plots the output and finally the last one changes the set-point.

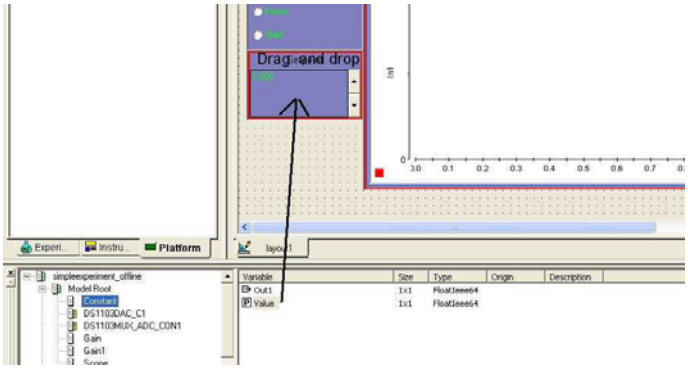


For now each element has a red border, which means that the elements are not associated with a variable of the Simulink model.

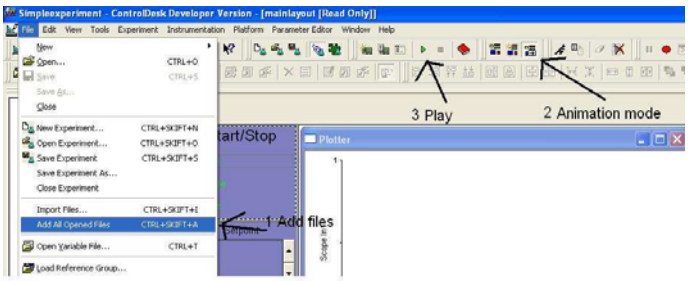
7. The next step is to bind Simulink and the GUI. In order to do it, you have to generate the Variable description file. This is achieved by right clicking on the Simulink icon and after by clicking on Build Variable Description file . After that, a new tab appears on the bottom of control desk.



After it is very easy you just have to drag and drop the variables which are useful for your GUI. Here we bind simState to the buttons, the constant to the numerical input and the scope to the plotter. The variable simState is very important because it monitors the behaviour of the real-time task of the dSPACE. It is required to put this variable to STOP when the experiment end (see page 432 on the Experiment Guide).You can highlight the variables which are associated to the GUI by right clicking on the GUI and clicking on highlight variables.



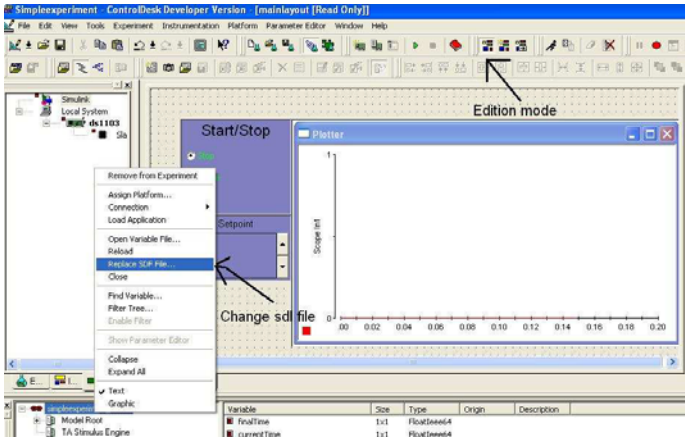
8. Now when the GUI is ready it is a good idea to test it with Simulink. First you have to add your file to the experiment, switch for the animation mode and press play or start in your GUI. **But before doing anything, be sure that the Simulink icon is highlight on the tab platform.**



Now you should be able to start and stop the simulation from your Gui, change the value of the set-point and view the plot, here as there is no model the plotter

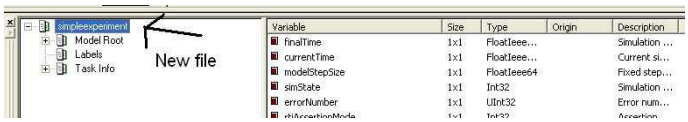
should be zero all the time. If your GUI seems to work you can now download the program to the CPU of the dSPACE.

9. **Here you have to be very cautious ! Indeed when Simulink builds the model and downloads it to the CPU the program starts immediately and you won't be able to control it with the GUI right now. So you have to be sure that the set-point is set to zero in order to avoid uncontrolled motion.** So when the set-point is set to zero press Ctrl-B in Simulink to build the model.
10. When the model is built you have to enable the GUI for the real experiments, in order to do it, you have to change the sdf file. This is achieved by right clicking on the sdf file which is offline and click on replace SDF file. But this is possible only if you switch to the edition mode.

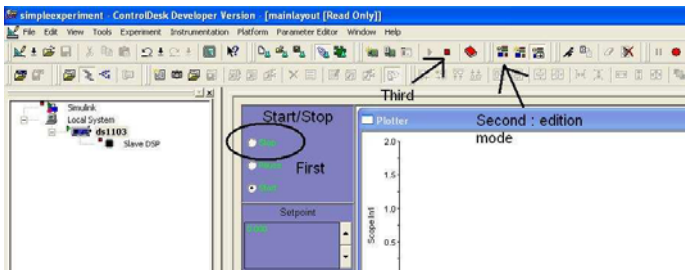


When the dialog box appears you have to choose the file which does not contain the keyword offline, in our example we choose the file simple experiment.sdf.

Now the bottom of control desk should have changed.



11. The last step is to switch to the animation mode. Now you can control the process with your GUI.
12. If you want to stop your experimentation you have to click on stop on your interface, switch to the edition mode and click on stop on control desk.

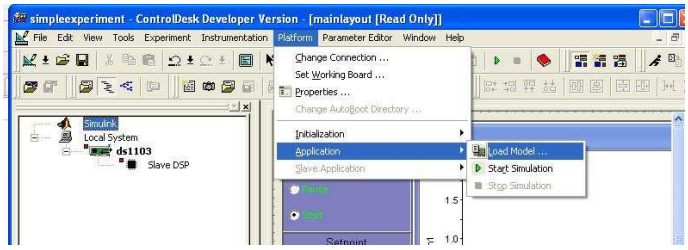


The green arrow disappears and a red square appears on the ds1103 platform tab.

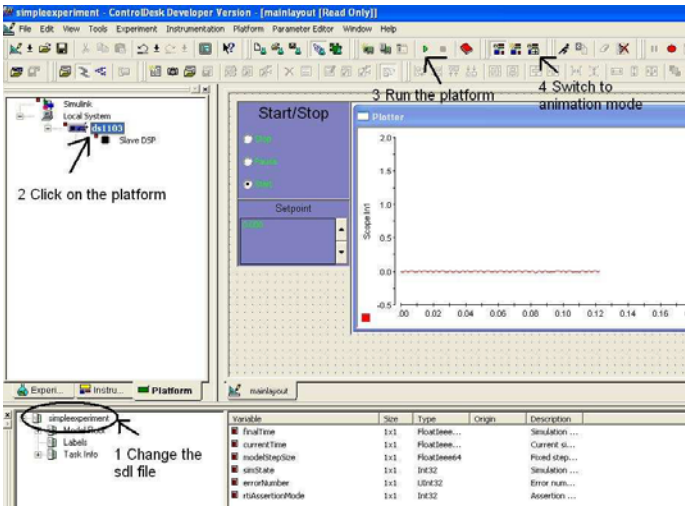
13. Now before closing ControlDesk you have to shut down Matlab, this has to be done inside ControlDesk. You just have to right click on the Matlab icon and click on Close Matlab. Now you can close ControlDesk and shutdown **in this order**: the computer, the process and the extension box.

Opening an existing experiment

1. Switch on and launch control desk as describe above. Click on : File ⇒ Open experiment.
2. Now the Simulink model has to be loaded, click on Platform ⇒ Application ⇒ Load Model. Be Careful you have to change the type of the file because the default type is .sdl



3. After loading the Simulink file you have to build it by clicking on Incremental Build or Ctrl-B. Then have to change first the SDL file and switch to the animation mode. Now you should have the control of the process using your GUI.

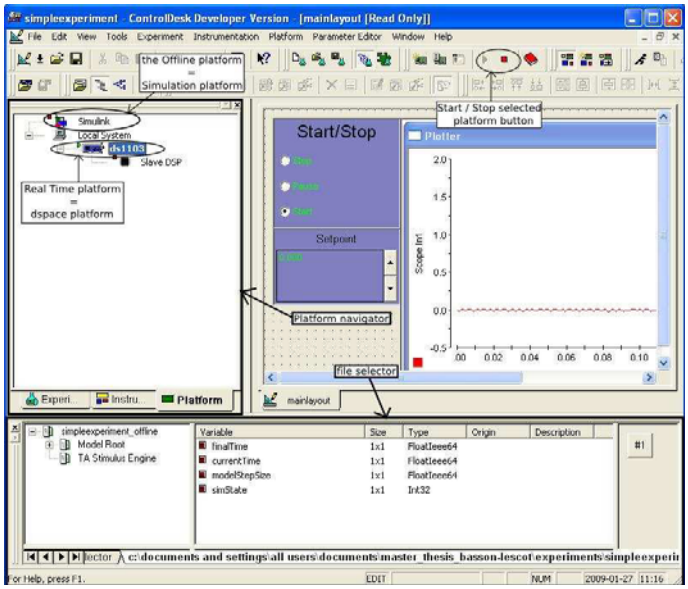


A.2 Information you should know and troubleshooting

This section is organized around different questions, which are either a need of information or a problem encountered. Of course, this part is extremely light in comparison of the dSPACE Experiment guide, and you should therefore refer to it if you need deeper explanations.

1. How does the link between Simulink and the ControlDesk is made ? How does the Build procedure works ?

   In a case of a single processor, Control desk has two distinct platforms:



   (a) The Simulation platform : this is an Offline platform , in the sense that if you use this platform, no code will run on the CPU.

   (b) The local system : this is the real time platform, i.e. the CPU or more generally the dSPACE board on which code will be transferred. As seen above, we have in our case a ds1103.

   As explained in the tutorial, when you start an experiment, you will first create (or open) a model. Then, once the model is achieved you can run it on the two different platforms :

   - Either you want to run your Simulink model on the real process ⇒ **REAL TIME PLATFORM**
   - Or you want to simulate it offline ⇒ **OFFLINE PLATFORM.**

   In any situation (either in Simulation or on your real process), you will want to control the simulation, so you must have access to the parameters of your Simulink file (stop time, simState variable, parameters of the controller you implemented...). This is done in Control desk by creating a layout in which you will be able to link your Simulink variables with the layout. To do that, ControlDesk uses an .sdf file (variable description file). One, <your_model>-_off_line.sdf contains all the variables associated with the Offline platform, while an other one, <your_model>.sdf contains all the ones associated with the Real Time platform. Each sdf file will be associated to one (or multiple) defined layout(s), on which links between instrument panels and the sdf variables will be performed so that one will be able to tune the variables of the considered platform.

   To summarize:

- Access to the variables of the Offline platform is done through the sdf file <your_model>off_line.sdf and modifications of these are made by linking them to a defined layout (see Part 1 of the tutorial).
- Access to the variables of the Real Time platform is done through the sdf file <your_model>.sdf , and modifications of these are made by linking them to a defined layout . But if you have already linked the variable for the offline platform, you should not have to link again the variable for the Real Time platform, this is done automatically.

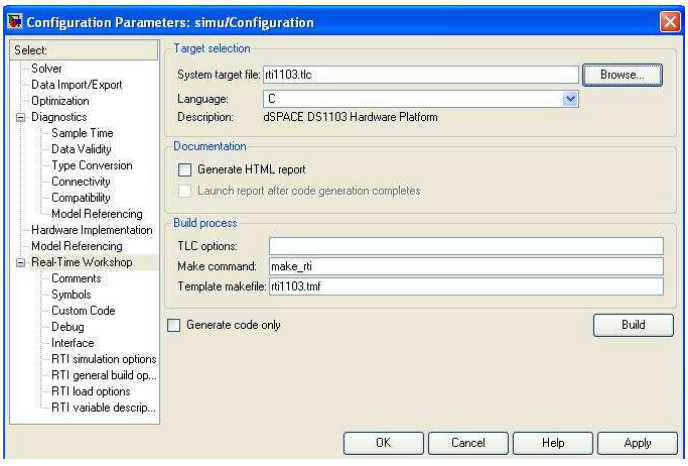
For more description on how to create a layout, and how to add instruments on the layout, please refer to the ControlDesk experiment guide.

2. But, How can these .sdf files be generated ?

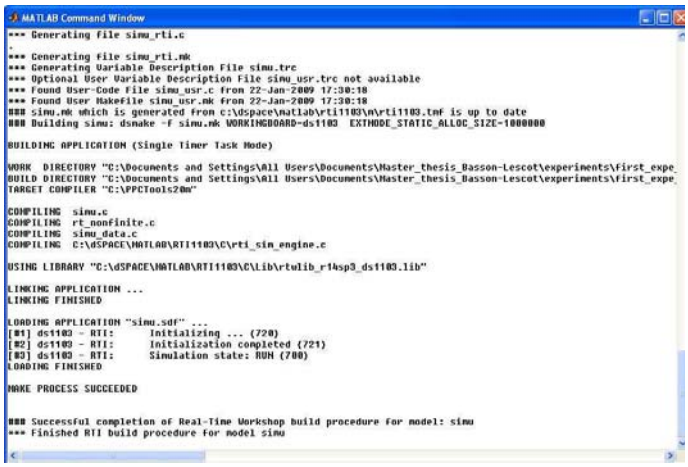
   When opening an existing project, the files <your_model>_off_line.sdf and <your_model>.sdf are already created. Then, as this section deals with how to generate sdf files, we will suppose that one Simulink file <your_model.mdl> had just been created (has been modified), so that finally, one need to (re)generate these files (<your_model>.sdf is generated when the .mdl is built).

   (a) In this case, as control algorithms are implemented in C-code on the CPU, you will have to translate Simulink code into C-code. Hopefully this is automatically done thanks to the RTW compiler, which, from the .mdl file (containing all the Simulink variables and parameters value pairs describing the Simulink model) will generate the C-code and directly transfer it into the CPU.

   By clicking on the Build button in the tab: Simulation-> Configure Parameters :

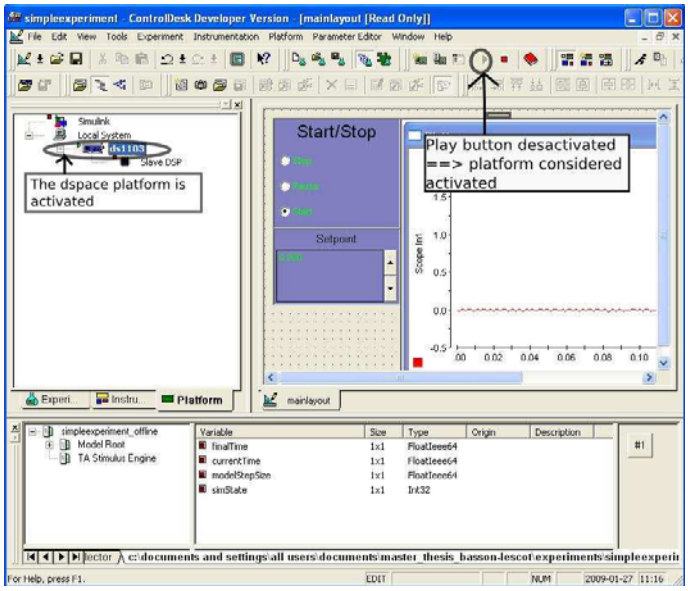


   The building procedure will run (see the bottom figure). First the c-file <your_model>-rti.c which will be transferred to the CPU is generated. Then a trc file <your_model>.trc is created : this is the variable description file providing for the ControlDesk information on the available Simulink variables and how they are linked. Then the building application starts with the usual compiling and linking steps. Finally, the file <your_model> .sdf is also created, just before the initialisation of the real time platform, and the C-code execution.



Finally, if everything works fine, the message MAKE PROCESS SUCCEEDED will appear, implying that building was successful Now the C-code has been transferred, and if the controller is in RUN mode, then the code will be executed immediately.

This description of the Build procedure was crucial for us. Using Simulink, one can think that the algorithm will be simulated on the off line platform, but in fact, by clicking on the build button, C-code will be generated and will automatically run on the CPU. Hence, as soon as the building procedure ends, the dSPACE platform is launched. This is shown with the play button, taking the place of the pause button, on the dSPACE platform tab :



(b) Running <your_model>.mdl on the Offline platform : Running your mdl file on the Offline platform is pretty simple. Now again, let us suppose that your Simulink file <your_model>.mdl is new , (or has been modified). Therefore, you will need to (re)generate the sdf file for the Offline Simulation : <your_model>_Off_line.sdf. As seen in part 1, this is

simply done by right clicking on the Offline platform (the one with the Simulink icon) and by selecting Build variable files.(see part 1). This will rebuild the <your_model>_off_line.sdf file.

3. What if everything is built, but I want to pass from one platform to an other ?

   A platform is launched / stopped by first, selecting it into the platform tab of the platform selector. and then by clicking on the Start / Stop Platform button(figure 3c).Then, passing from one platform to an other is simply done (see figure below) by first, selecting the platform you want to stop (if there is one). Then stop it using the Start/ Stop Platform button. After, do the same with the platform you want to start : select it, and after click on the start button of the Start stop button. Note Also that **it is not because you have changed the .sdf file that the associated platform is selected.** To put in a nutshell, if for example you want that your simulation be run in real time, this is not because you selected the <your_model>.sdf file and then clicked on the start button that it will work ! If unfortunately the selected platform is the offline one, then by clicking to the start platform button you will start the simulation on the offline platform.

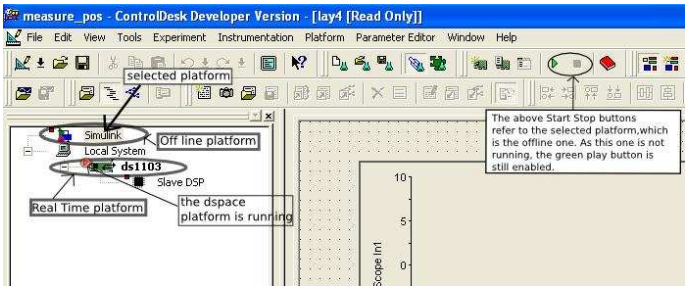
   So the right way to do is:

   (a) open the sdf file of the platform you want to launch.

   (b) select the desired platform on the tab

   (c) click on the start / stop platform button 

4. When I click on the stop button, the process is still running, Why ? Again, this problem is due to the fact that the Start / Stop button is associated with the platform which is currently selected.



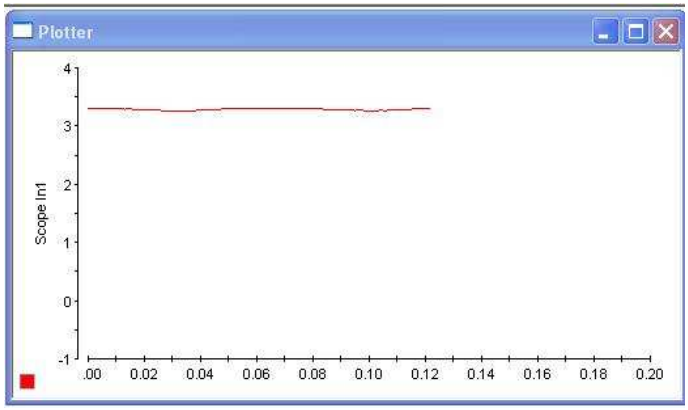
5. When the simulation runs on the Real time platform, and I change some values of my variables using the layout, it doesn't happen anything.

   This is because you did not load the right .sdf file. A layout is linked with a certain sdf file, so if the wrong sdf file is loaded, then changing the parameters in the layout won't have any impact ! For the real time platform, you need to load the <your_model>.sdf while for the offline platform the file <your_model>_off_line.sdf .

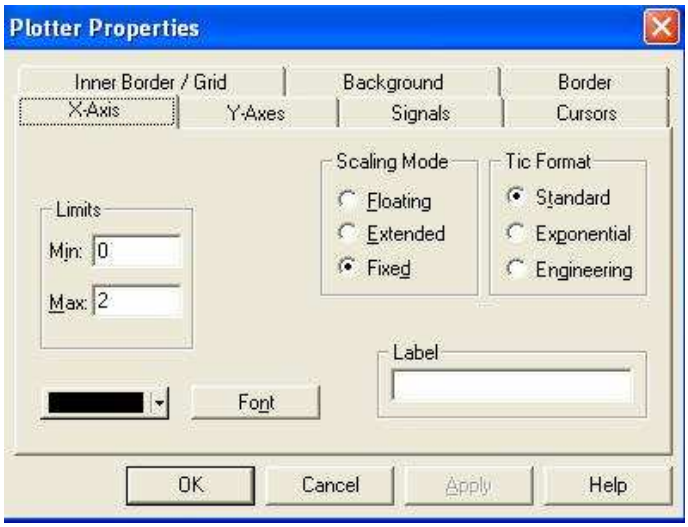
6. In the layout associated with the <your_simu>.sdf file, I put a radio button, linked with the SimState variable. Why when I press the Stop mode button on my radio button the stop button is not enabled in the ControlDesk toolbar ?

This is because the simState is a variable which sets the RUN mode for the CPU. The CPU can either be in the RUN, PAUSE or STOP mode. But these modes represent the states of the real time task which is running on the CPU. So you can be on the Stop mode for the real time task whereas the CPU is still ON, but in this case the CPU do nothing at all.

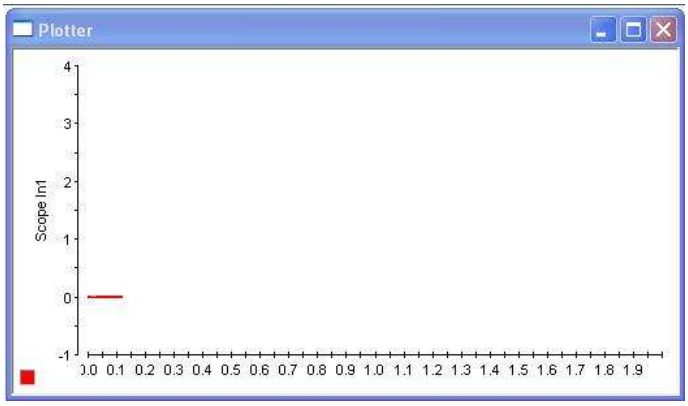
7. Plotters and capture settings, how to use them? When you use a plotter on a layout, you have first the default plotter:



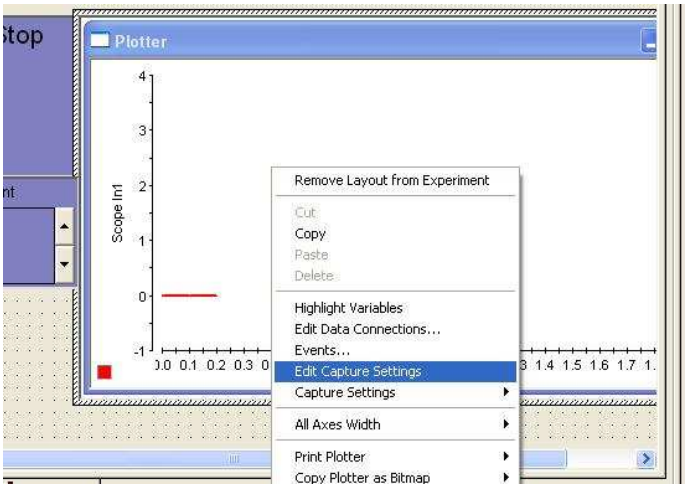
(a) The X scale is too small, how should I change it? Change the setting of the X-axis by right clicking on the plotter and click on axis properties. You will obtain the same dialogue box as below :



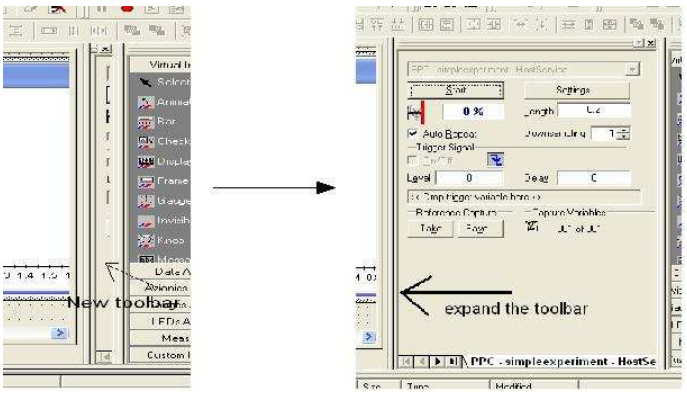
(b) It does not help! I still have a problem:



In fact even if the scale in X is bigger, the signal still has the same range as before. So you have to change the capture settings. this can be done by right clicking on the plotter and click on edit capture settings.



Nothing seems to happen after clicking but if we look at the right side of the software a new toolbar has appeared.



Now you can change the length of the capture, here by default the length is set to 0.2. With this dialogue box you can also save the data. Please refer to the manual for more information.

A. Starting the Experiment

In this appendix is described the procedure for starting the experiment for the Furuta pendulum. But before it is highly recommended to read the "General Tutorial for ControlDesk".

A.1 Setting-Up the Experiment

1. **Before doing anything, just be sure that each device is switched off and disconnected from the power supply (the dSPACE, the host computer and the Furuta pendulum**

2. Connect the pendulum with the extension box of the dSPACE, the connections are summarized in the table A.1. Just do not forget to plug the ground between the devices.

Table A.1 Connection between the Furuta pendulum and the dSPACE

From the Furuta	to the dSPACE
Arm angle	ADCH1
Arm velocity	ADCH2
Pendulum angle 360	ADCH5
Pendulum angle top	ADCH7
Pendulum velocity 360	ADCH6
Pendulum velocity top	ADCH8
In 1 (Motor voltage)	DACH1

3. Connect the devices to their power supply.

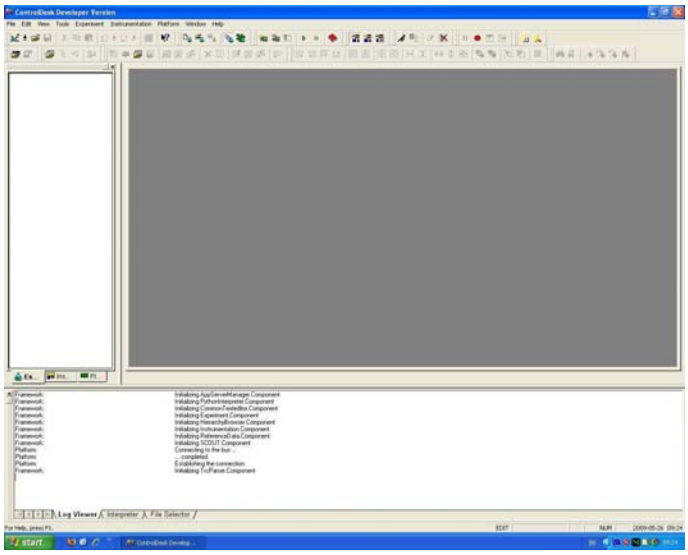
4. Switch on the dSPACE board (switch at the back of extension box).

5. Switch on the Furuta pendulum.

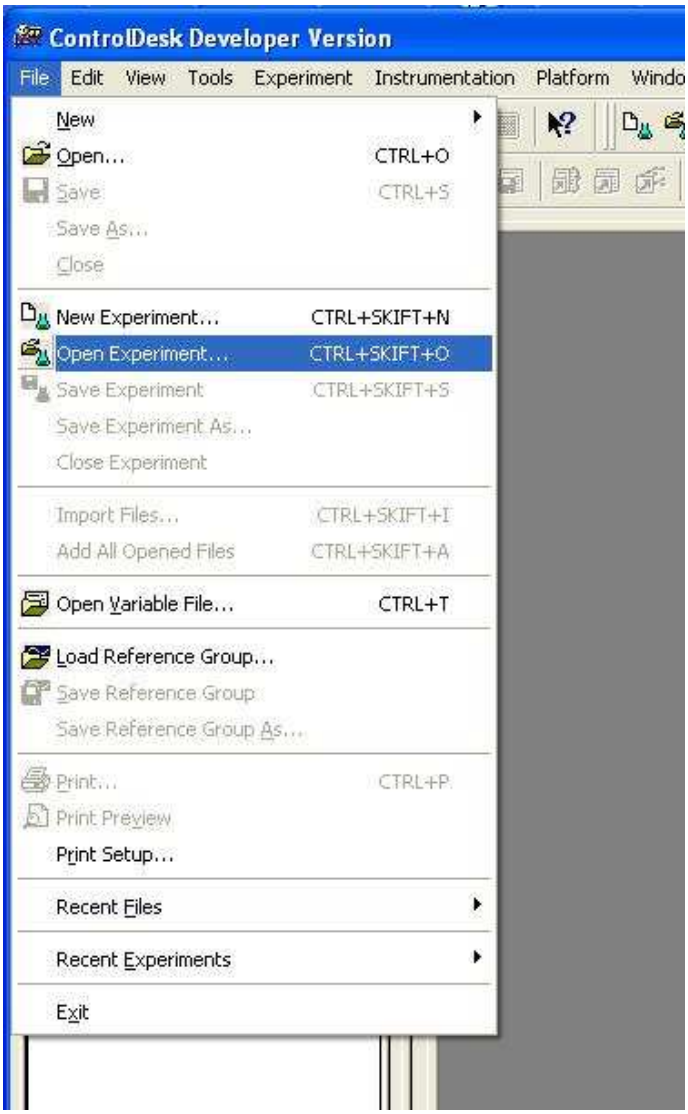
6. Make sure that the green dongle for ControlDesk is plugged at the back of the host computer and switch on the computer (login: dspace password: dspace).

A.2 Open the Experiment

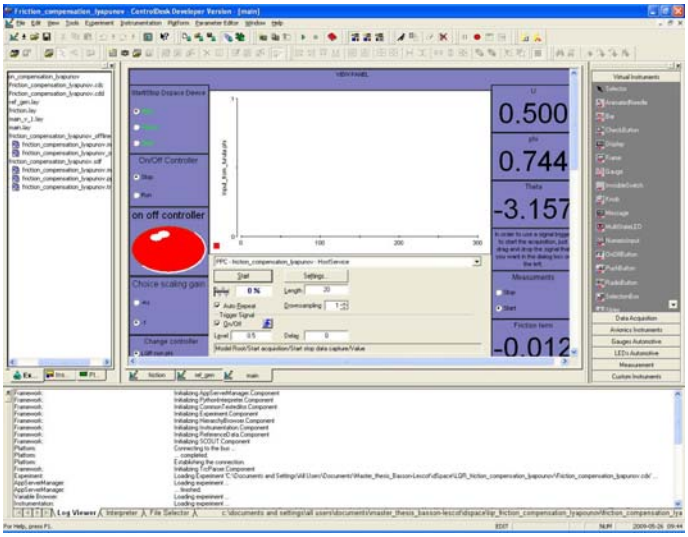
1. Open ControlDesk 



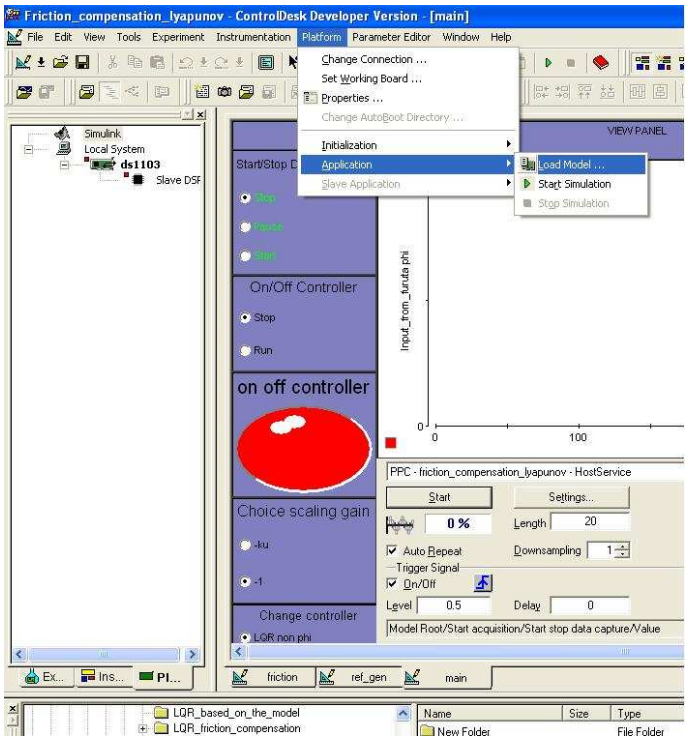
2. Now just open the existing experiment for the Furuta, the file is located in at the following path : root_directory\dSPACE \LQR_friction_compensation-_lyapunov \Friction_compensation_lyapunov.cdx



You should obtain the following :



3. Click on the menu Platform ⇒ Application ⇒ Load Model ...



And open the model : friction_compensation_lyapunov.mdl, after that Matlab, Simulink and the Editor of Matlab should start.

A.3 Start the Experiment

Description of the GUI

Here is a description of the GUI which is used for controlling the Furuta pendulum. There are three tabs, one for the main control, one for the reference generator and finally one for the friction compensation.

Main tab Here is a list of each component of the main tab :

1. SimState Buttons (number 1 in figure A.1)
   These radio buttons allow to change the state of the real-time task in the CPU. It is possible to start, stop or pause the task. Which is good to do is to always stop the task before closing the experiment in order to avoid uncontrolled behaviour from the Furuta.
2. On/Off Controller (number 2 in figure A.1)
   These two radio buttons allow to enable or disable the controller, if Off is checked, the DC motor receives a zero signal as control signal. In the other case the regular control signal is applied. Moreover the led changes its colour : Red for Off and Green for On.
3. Choice of scaling gain (number 3 in figure A.1)
   There are two gains for the control signal which can be chosen. These gains

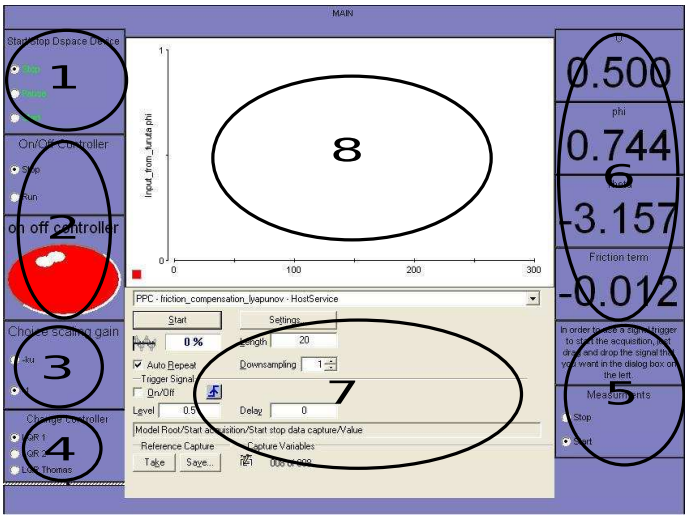


Figure A.1 The Main Tab

are just before the DAC block in Simulink: the gain -1 corresponds to just an inversion to obtain a logical behaviour (positive control implies positive angle) and the gain $-ku$ corresponds to the gain of the DC motor to convert Torque in Voltage.

4. Change controller (number 4 in figure A.1)
   These radio buttons allow to choose the LQ-controller three controllers are available (see table A.2).

Table A.2 Description of the LQR

LQ1	An LQR which allows to do easily the swinging up
LQ2	An LQR without any constraints on the angle ϕ
LQR Thomas	The LQR which was used by Thomas Dietz, but be sure that the scaling gain is put on $-ku$. Moreover it is easier to switch the controller when the pendulum is already in upright position

5. Measurements (number 5 in figure A.1)
   This is an extra feature which allow the user to create a signal to do a trigger for the acquisition of the data.
6. Display (number 6 in figure A.1)
   There are simple displays for reading the value of U, φ, θ and the Friction term.
7. Plotter settings (number 7 in figure A.1)
   This window controls the parameters of the plotters, it can be reach by other ways, but even with reading the Experiment Guide it's hard to figure out how to make the window appears, so the window is directly integrated into the GUI.
8. Plotter (number 8 in figure A.1)
   This is the plotter where normally φ is plotted. But the user can add plotters as many as he wants in order to plot other variables.

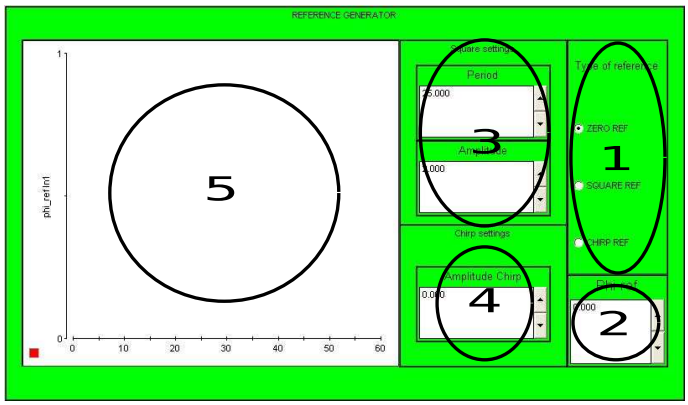


Figure A.2 The Reference Generator

The Reference Generator

1. Type of reference (number 1 in figure A.2)
   These radio buttons allow to choose the type of reference that the user wants, there are three different signals, a constant one, a square one and a chirp signal.

2. Phi ref (number 2 in figure A.2)
   This numerical input allows to change reference value for φ

3. Square settings (number 3 in figure A.2)
   These two numerical inputs change the amplitude and the period of the square signal.

4. Chirp settings (number 4 in figure A.2)
   Here it is possible to change the amplitude of the chirp signal.

5. Plotter (number 5 in figure A.2)
   With this plotter it is possible to visualize the reference signal.

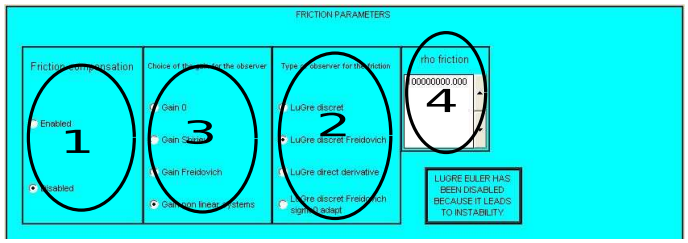


Figure A.3 The Friction Parameters Tab

Friction Parameters

1. Friction compensation (number 1 in figure A.3) These radio buttons control if the friction compensation is enabled or not.

2. Type of observer for the friction (number 2 in figure A.3)
   here it is possible for the user to choose between different type of observer, there is no big difference between observers so during the experiment for instance the authors have chosen the LuGre discret Freidovich.

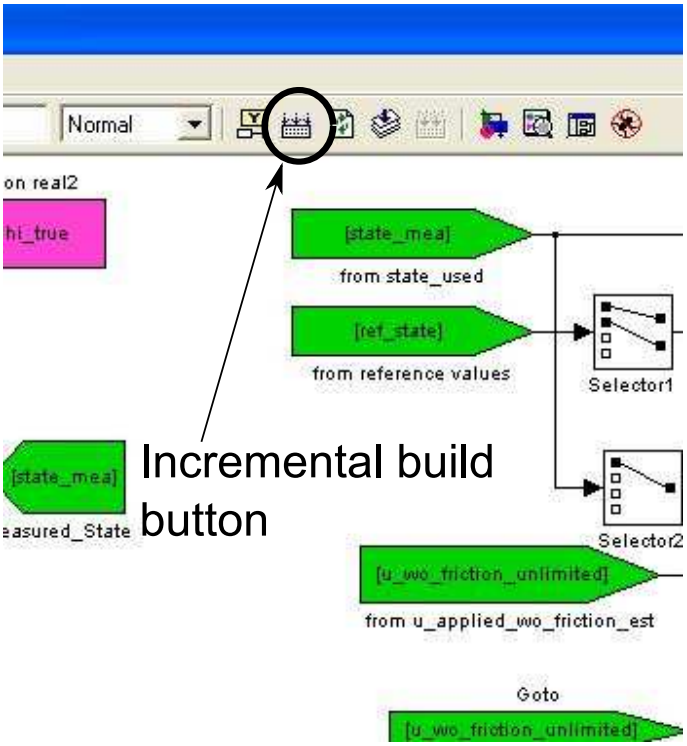
3. Choice of the gain for the observer (number 3 in figure A.3)
   As observers are used, they have a gain of adaptation of the error, here it is possible for the user to choose the gain for the observer. The gain called :"Gain for non linear systems" is the gain of the article.

4. rho friction (number 4 in figure A.3)
   Here it is possible to change the value of the parameter rho which steps in the computation of the gain of the observer.

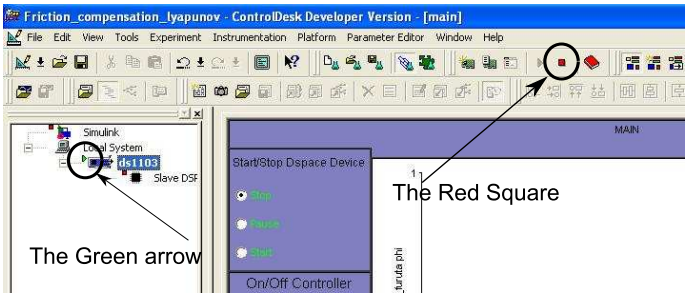
Launching the experiment

1. Make sure the emergency button is disabled

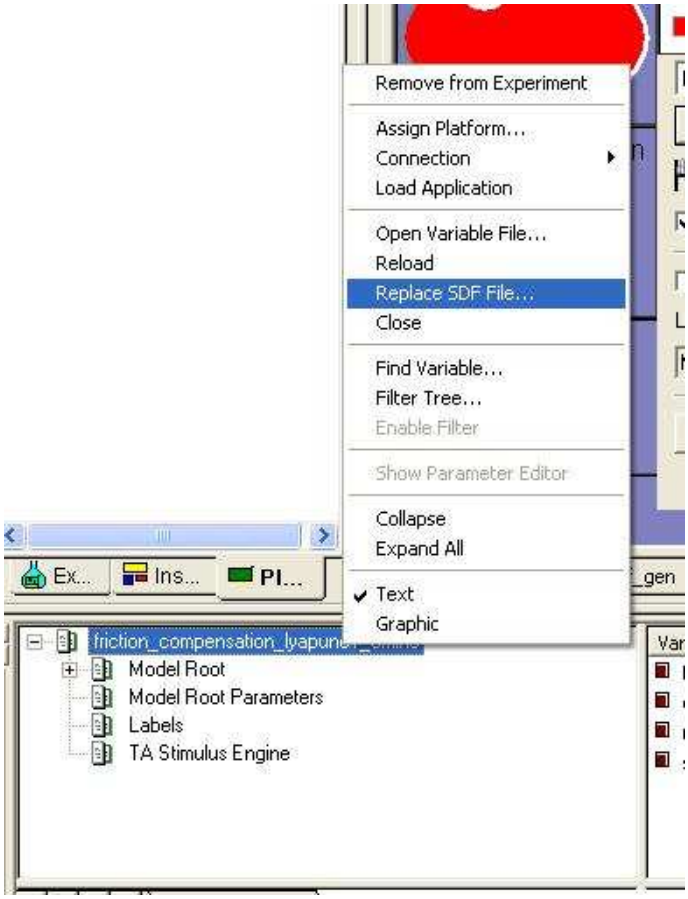
2. Now the Simulink file has to be compiled and transferred into the dSPACE board. To do so, the user just have to click on "incremental build" in Simulink.



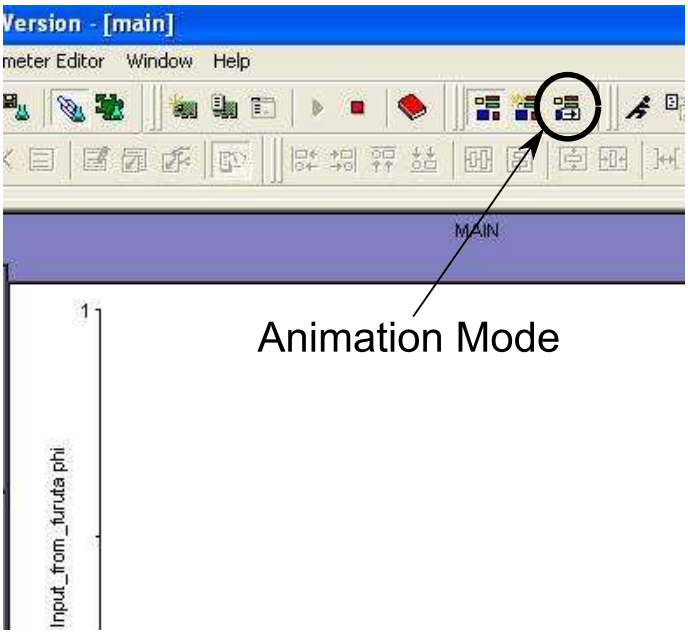
3. After that, as it was mentioned in the General tutorial, a green arrow should appear in ControlDesk in the navigator and a red square appears to in the toolbar.



4. The last thing to do is to run the GUI. The SDF file has to be replaced by the file : “friction_compensation_lyapunov.sdf” instead of “friction_compensation_-lyapunov.sdf_offline”.



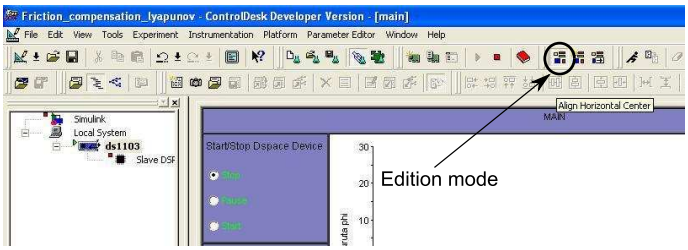
5. Finally the next step is to click on Animation Mode :



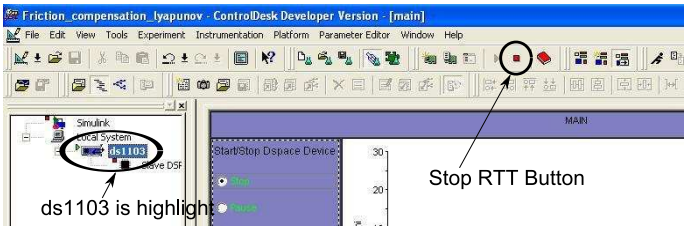
Now you should be in control of the Furuta pendulum throughout the GUI.

A.4 Stopping the Experiment

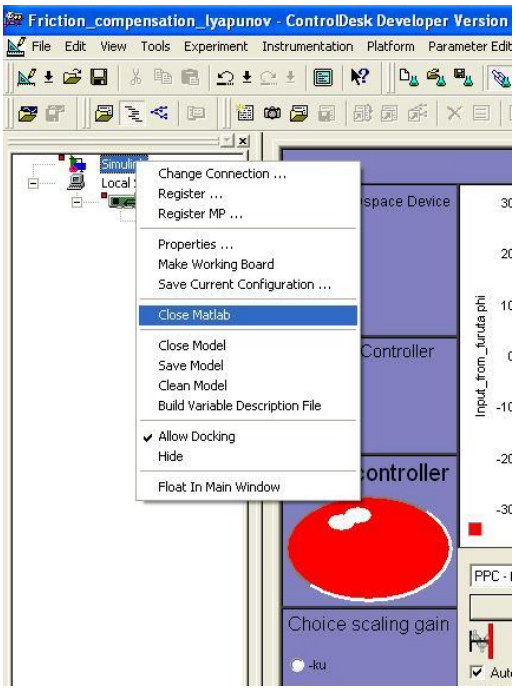
1. Put the Controller Off using the GUI.
2. Put the variable SimState on the state : “Stop” using the GUI.
3. Put the GUI in the edition mode.



4. Stop the real-time task (RTT) by clicking on the red square, but be sure that “ds1103” is highlight before doing it.



5. Close Matlab, but inside ControlDesk with the dedicated menu, this menu can be reached using the platform navigator.



A.5 Remarks and Troubleshootings

1. The Plotter
   It may happen that the plotter does not work at all, in this case the authors recommend to start the experiment but with Matlab instead of the ds1103 as platform. After that, the plotter should work fine for the Matlab experiment but also for the ds1103 experiment.

2. The GUI
   It may happen that the GUI is not linked with the variables of the experiment, in that case the user should bind the variables manually according to the table A.3.

Table A.3 Recap for binding the variables and the GUI

Marks	Associate variable
The Main tab (see figure A.1 for the marks)	
1	simState
2	Output2furuta → choice control (for the button and the light)
3	Output2furuta → choice scaling
4	Model Root → change controller
5	Start acquisition → Start stop data capture
6	Model Root → control signal : for U Input_from_furuta → phi : for φ Input_from_furuta → theta : for θ Model Root → Friction compensation : for Friction term
7	No variable
8	Input_from_furuta → phi
The Reference Generator (see figure A.2 for the marks)	
1	Model Root → choice ref
2	Ref_Gen → Phi_ref_value
3	Ref_Gen → SQUARE REF → period_square Ref_Gen → SQUARE REF → amplitude_square
4	Ref_Gen → CHIRP_REF → Amplitude_chirp
5	Ref_Gen → phi_ref
Friction Parameters (see figure A.3 for the marks)	
1	Model Root → Friction_switch
2	friction compensation → switch_friction_estimator
3	friction compensation → switch_gain_friction_est
4	friction compensation → Observer Gain Shiriaev Mat → Observer Gain Shiriaev Mat parameters → rho_NL

3. Starting Simulink
   In the experiment there is no need for the user to launch any scripts to initialize the variables. But this is done anyway, each time Simulink is started, the script startup.m is executed and the editor is launched. This is done thanks to the Callback function “PreloadFcn” (See page 392 in the Experiment Guide). The script startup.m just executes each script which is required for the experiment.

4. Swinging-up
   It may happen that the swinging-up does not work the first time, in that case just restart the pendulum using the little black button and try it again.

B. Setup of the Matlab-Simulink simulations

B.1 Introduction

This Appendix aims at giving information about the final Simulink file which was built in Matlab-Simulink in order to verify in simulations the results seen in [Robertsson *et al.*, 2004].

Of course plenty of Simulink files were carried out before this final Simulation. These simulations can be found at the following path:

Simulink/, each

one consisting in one directory, containing a *readMe* text which fully describes the purpose of the test considered, the m-files and the Simulink model used. Therefore for the authors it did not seem necessary to comment further these test simulations.

The Simulink experiment used for the final simulations is located at :

Simulink/main-Simulation

As said before, it was built in such a way that few modifications have to be done when this file will be used for real experiments. Therefore, no S-functions nor Matlab-functions were used (except for the model of the process, which of course will be replaced by inputs / outputs of the Furuta pendulum for real experiments).

This explains why sometimes the operations done in Simulink, and especially for the friction compensation subsystem, look complicated and ugly. Indeed, this is specially the case for the friction compensation subsystem, while the other ones remain quite understandable. For this reason, the explanation will be organized as follows:

- In a first part, a list and description of the different Simulink subsystems will be given, while a table will summarize the nomenclature of the m-files used for this Simulation.
- The last part will be entirely dedicated to the description of the Friction compensation block, and to the different Friction models implemented.

B.2 General description

The list below briefly describes the different subsystems, found in the Simulink file:

- **Ref_Gen** : block which generates the Reference signal. Three different reference signals, acting on φ, are available : constant, step, or chirp signal. For more details, see the *readMe* file.
- **Swing_Up** : block which computes the control, seen in [Åström and Furuta, 1996], for the swinging up:

$$u = -k_{swing} \cdot \ sign\left[\dot{\theta} cos\theta (cos\theta - 1 + \frac{\dot{\theta}^2}{2\omega_0^2}\right]$$

- **Calc_V** : computation of the initial Lyapunov function V:

$$V = X^T P_C X$$

- **Furuta_With_Friction**: Model of the process, including a S-function implementing the state representation given in, and an internal model of the friction to compute the "real" friction force F_F. The friction model used here is implemented using equations (1.7)(1.8),(1.9), and (1.10) of the continuous LuGre model.
- **Friction_Compensation**: Block implementing different friction observers for the computation of the friction force and internal friction state estimates $\hat{F}$ and $\hat{z}$. These observers will be described in the next section.
- **Calc_W**: Simple Computation of the Extended Lyapunov function W:

$$W = V + \frac{\rho}{2}(z - \hat{z})^2$$

- **Compute_ek**: Computation of the internal state error :

$$e_k = z_k - \hat{z}_k \ ,$$

  using equation (4.12) (page 31)

In the following table the m-files used for the Simulation are given:

init_Friction_Params	defines all the friction observers parameters
init_Model_Params	defines the Furuta model parameters
init_RefGenerator	sets the ref_Gen parameters
init_Simulation	sets the Simulation variables, such as the simulation time, or the inferior bound for the ode23s solver for example
LQR_Swinging_Params	sets the LQ and Swinging parameters
FurutaModel	S-function implementing the model of the process
runMe	Run file for the Simulation

B.3 description of the Friction_Compensation block

This section intends to document the different friction observers which were implemented for Friction compensation purpose. These observers are all based on the LuGre model, the only differences between them being :

- If the LuGre model is discretized or not, and how the discretization is performed.
- If the model is or isn't written in the new variable (Freidovich parametrization):

$$z_{new} = \sigma_0 z$$

Of course from all these observers only one – the one leading to the best results – was used for the simulations both in Simulink and on the real process. Nevertheless, for the authors it seemed important to give a description of each one.

- *LuGre_friction_Observer_Continuous* (Observer 1) : only observer which is implemented in Continuous time, using the continuous LuGre observer, seen in (1.18), section 1.4. For this observer, $\dot{z}$ is therefore integrated using the ode23 s stiff solver. Moreover, this observer is the one used in the Simulation part, when it was noticed that W had a strange peaking behaviour for a friction compensation implemented using a discrete observer.
- *LuGre_friction_Observer_discrete* (Observer 2) : "discretized Lugre observer" i.e. observer implemented using the discretized version (4.5). z is the "normal" state variable (therefore no Freidovich parametrization is done). Moreover,in the expression of F, $\dot{z}$ is integrated using the explicit forward Euler method.
- *LuGre_friction_Observer_discrete_parametrization_Freidovich* (Observer 3): "discretized Lugre observer with the Freidovich parametrization". Now the discretized form with the Freidovich parametrization is used. Again, $\hat{F}$ is calculated using the Forward Euler method.
- *LuGre_friction_Observer_Derivative_direct* (Observer 4): Same observer as Observer 3 but this time, in the expression of $\hat{F}$, $\hat{z}$ is not discretized anymore. Instead, the expression, seen in (4.4) is directly used.
- *LuGre_friction_Observer_discrete_parametrization_Freidovich*(Observer 5) : Same observer as observer 3 but now the damping term σ_1 is not constant, but is velocity depending. See equation (4.10)
- *LuGre_friction_Observer_discrete_Euler_Parametrization_ Freidovich* (Observer 6): now $\hat{z}$ is not discretized as it is done in [Freidovich *et al.*, 2006], but instead, using the simple Forward Euler method. Again, the discretization is expressed in the Freidovich variable z_{new}

C. Parameter Values

This section documents the parameters used for the different simulations performed. The bold police style indicates that the considered parameters were used for the experiments described throughout the thesis, while a normal police style refers to parameters of other Friction models (the Coulomb and Dahl models), which were not described in the report.

C.1 Explanation of Parameters

Table C.1: Explanation of parameters

	Parameter	*Explanation*
Simulation parameters	**T_sample_discrete**	Fundamental sampling time that is used during the execution of the system.
	t_step_min	inferior bound for the variable step size used by the ode23s solver.
	t_step_max	superior bound for the variable step size used by the ode23s solver.
	k_u	Output factor for the conversion of the moment computed by the controller into control voltage that is applied to the system.
System parameters	**m_a_est**	Mass of the arm of the pendulum.
	l_a_est	Length of the arm of the Furuta pendulum.
	l_a_inert_est	Distance of the cog of the arm from its axis of rotation.
	J_m_est	Moment of inertia of the motor shaft around its axis of rotation.
	m_2_est	Mass of the pendulum rod.
	m_p_est	Mass of the weight attached to the top of the pendulum.
	l_2_est	Distance of the cog of the pendulum rod from the joint connecting pendulum and arm.
	l_p_est	Distance of the weight attached to the top of the pendulum and the joint connecting pendulum and arm.
Friction Com pensation Parameters	**F_c_p_est_p_f**	Coulomb friction of the arm for positive velocities (see F_c in equation (1.9)).
	F_c_p_est_n_f	Coulomb friction of the arm for negative velocities (see F_c in equation (1.9)).
	F_s_p_est_p_f	Stiction force of the arm for transfer from stiction to friction in positive turning direction (see F_s in equation (1.9)).

F_s_p_est_n_f	Stiction force of the arm for transfer from stiction to friction in negative turning direction (see F_s in equation (1.9)).
v_s_est_f	Stribeck velocity (see v_s in equation (1.9)).
sig_0_est_f	Stiffness of the LuGre model (see σ_0 equations (1.7) to (1.10)).
sig_1_est_f	"Damping" of the LuGre model (see σ_1 in equations (1.7) to (1.10)).
v_d_est_f	Parameter that controls the speed of decrease of the damping rate with increasing velocity for the LuGre model with velocity dependent damping (see v_d in equation (1.10)).
F_v_est_p_f	Viscous friction parameter for positive velocities (see F_v in equation (1.8)).
F_v_est_n_f	Viscous friction parameter for negative velocities (see F_v in equation (1.8)).
F_c_est_dahl	Coulomb friction for the Dahl model.
sig_0_est_dahl	Stiffness of the Dahl model.
exp_dahl	Exponent used in the Dahl model.
thres_coul_vel	Half width of the dead-zone for velocities in the static coulomb map, to prevent excessive switching of the friction estimate (Velocities that have a smaller absolute value than this threshold are assumed to be zero).
thres_coul_cont	Half width of the dead-zone for the control signal in the static coulomb map, to prevent excessive switching of the friction estimate (Control signals that have a smaller absolute value than this threshold are assumed to be zero).
thres_rem_sing	Half width of the dead-zone where the the coefficient a_k, involved in the discretized Lugre form (4.5), is considered "small" enough so that the recursion for the case $a_k = 0$ can be applied.
rho_fricobs_shir	ρ parameter of the Shiriaev Observer gain K. see [Shiriaev *et al.*, 2003].
rho_fricobs_frei	ρ parameter of the Freidovich Observer gain K. For more details, see [Freidovich *et al.*, 2006].
rho_NL	ρ parameter of the injection function K seen in [Robertsson *et al.*, 2004], and in Theorem 1.1 of section 1.4.
$\mathbf{Q_{12}}$	Non diagonal term of the matrix Q encountered in inequality (1.14), section 1.4.

	Q_{22}	Diagonal term of the matrix Q encountered in inequality (1.14), section 1.4
Controller parameters	**R_l**	Punishment for control effort for discrete time LQ-controller design.
	Q_l	Punishment for control error for discrete time LQ-controller design.
Ref_ Gen parameters	**chosen_ref_signal**	constant defining the type Reference given. 1: Constant φ-ref; 2: Step φ-ref; 3 = chirp φ-ref (sinus with time-varying frequency).
Const. φ-ref	**phi_ref**	Value of the φ reference
Step φ-ref	**period_square**	Half period of the Square reference
	amplitude_square	High level of the Square reference
Chirp φ-ref	**A_chirp**	Amplitude of chirp signal
	c_1_chirp	First chirp constant
	c_2_chirp	Second chirp constant

C.2 Basic Parameter Setup

Table C.2: Parameters values

	Parameter	*Value*	*Unit*
Simulation parameters	**T_sample_discrete**	1	$[ms]$
	t_step_min	0.1	$[ms]$
	t_step_max	2	$[ms]$
	k_u	10.96	$[V/Nm]$
System parameters	**m_a_est**	0.165	$[kg]$
	l_a_est	0.254	$[m]$
	l_a_inert_est	0.044	$[kg]$
	J_m_est	$0.381 \cdot 10^{-4}$	$[kg\ m^2]$
	m_2_est	0.02	$[kg]$
	l_p_est	0.421	$[m]$
	l_2_est	=l_p_est$/2 = 0.2105$	$[m]$
	m_p_est	0.015	$[kg]$
Friction Com pensation Parameters	**F_c_p_est_p_f**	0.014	$[Nm]$
	F_c_p_est_n_f	0.012	$[Nm]$
	F_s_p_est_p_f	0.019	$[Nm]$
	F_s_p_est_n_f	0.017	$[Nm]$
	v_s_est_f	0.04	$[m/s]$

	sig_0_est_f	$3.4 \cdot 10^2$	$[Nm/m]$
	sig_1_est_f	1	$[Nm\ s/m]$
	v_d_est_f	0.006	$[m/s]$
	F_v_est_p_f	0.00023	$[Nm\ s/m]$
	F_v_est_n_f	0.00022	$[Nm\ s/m]$
	F_c_est_dahl	=F_c_p_est_p_f	$[Nm]$
	sig_0_est_dahl	=sig_0_est_f	$[Nm/m]$
	exp_dahl	1	$[-]$
	thres_coul_vel	$1 \cdot 10^{-4}$	$[m/s]$
	thres_coul_cont	$1 \cdot 10^{-4}$	$[Nm]$
	thres_rem_sing	$1 \cdot 10^{-6}$	$[-]$
	rho_fricobs_shir	$1 \cdot 10^{8}$	$[-]$
	rho_fricobs_frei	$1 \cdot 10^{5}$	$[1/s]$
	rho_NL	$1 \cdot 10^{8}$	$[1/s]$
Controller parameters	**R_l**	5000	$[-]$
	Q_l	$\begin{pmatrix} 1/0.1^2 & 0 & 0 & 0 \\ 0 & 1/1^2 & 0 & 0 \\ 0 & 0 & 1/0.3^2 & 0 \\ 0 & 0 & 0 & 1/2^2 \end{pmatrix}$	$[-]$
Ref_ Gen parameters	**chosen_ref_signal**	1	$[-]$
Const. φ-ref	**phi_ref**	0	$[Rad]$
Step φ-ref	**period_square**	25	$[s]$
	amplitude_square	2	$[Rad]$
Chirp φ-ref	**A_chirp**	12	$[Rad]$
	c_1_chirp	$2 \cdot 10^{-3}$	$[-]$
	c_2_chirp	0	$[-]$

D. Article : Friction Compensation for Non Linear Systems Based on the LuGre Model

FRICTION COMPENSATION FOR NONLINEAR SYSTEMS BASED ON THE LUGRE MODEL

A. Robertsson * **A. Shiriaev** ** **R. Johansson** *

* *Dept. of Automatic Control, Lund Institute of Technology*
Lund University, PO Box 118, SE-221 00 Lund, Sweden.
{Anders.Robertsson|Rolf.Johansson}@control.lth.se

** *Dept. of Applied Physics and Electronics*
Umeå University, SE-901 87 Umeå, Sweden
anton.shiriaev@tfe.umu.se

Abstract: Friction may be a major obstacle for stability and control performance. Based on a nominal control law, which stabilizes a nonlinear dynamical system to an equilibrium or to a desired limit cycle motion and which is derived without taking the friction disturbance into account, a method for compensating the impact of the friction is presented and stability analysis for the closed loop system is presented. The result can be interpreted as a nonlinear observer-based solution for the LuGre friction model with new design for the nonlinear injection function. An illustrative example is given for the Furuta pendulum.

1. INTRODUCTION

This paper is devoted to the problem of friction compensation in nonlinear control systems. It is assumed that the system dynamics without friction is described by the equations

$$\begin{aligned} \frac{d}{dt}x &= F(x) + G(x)u \\ u &= H(x,\, t) \end{aligned} \tag{1}$$

where a state vector $x \in R^n$, a control action $u \in R^1$, and F, G, H are smooth vector fields of appropriate dimensions.

While, in addition, it is assumed that the controller $u = H(x,\, t)$ for the closed loop system (1) is designed to achieve uniform asymptotic stability of a compact subset $\Omega \subset R^n$ and does so successfully. Furthermore, it is assumed that there exists a nonnegative function $V(x,\, t)$ such that:

- the zero-level set of V belongs to Ω, i. e.,
$$V_0 = \{x \in R^n : V(x,\, t) = 0\} \subset \Omega \tag{2}$$
- the function V satisfies the inequalities
$$0 \le |y(x,t)|^2 = V(x,\, t) \le V_m |x|^2, \quad \forall t \tag{3}$$
where $y(x,t)$ may be viewed as a regulated output
- the time derivative of V evaluated along any solution $x(t)$ of the closed loop system (1) satisfies the following relations
$$\begin{aligned} \frac{dV}{dt} &= \frac{d}{dt}V\,(x(t),\, t) = \kappa_0(x,\, u,\, t) \\ &= \kappa_0(x,\, H(x,t),\, t) \le -\varepsilon V\,(x(t),\, t) \end{aligned} \tag{4}$$
for some $\varepsilon > 0$
- the function κ_0 in (4) is such that for any *matched* disturbance d, we have
$$\begin{aligned} &\kappa_0 = \kappa_0(x,\, H(x,t) + d,\, t) = \\ &\quad = \kappa_0(x,\, H(x,t),\, t) + \kappa_1(x,\, t)d + \kappa_2(x,\, t)d^2 \end{aligned}$$

As a special case of this form we consider

$$\frac{d}{dt}\,V_{|_{u=H}} \le -\begin{bmatrix} x \\ H \end{bmatrix}^T Q(t) \begin{bmatrix} x \\ H \end{bmatrix} \le 0, \tag{5}$$

$$Q = Q^T = \begin{bmatrix} Q_1 & Q_{12} \\ Q_{12}^T & Q_2 \end{bmatrix}, \quad Q_2 > 0 \tag{6}$$

for the *particular case* of $u = H(x,\, t)$.

In (Shiriaev *et al.*, 2003) the case for passive systems was considered.

As one can expect, a presence of friction in the actuator, that is when the 'true' control action applied to the system (1) has the form

$$\tau = u - F, \tag{7}$$

may prevent from the stabilization of the set (2) and disregard the expected behavior of the closed loop system.

The problem of modeling the friction F is a vast area of research, and many different models for this phenomenon are available in the literature. In this paper it will be assumed that the friction is modeled based on the LuGre model (Canudas de Wit *et al.*, 1995). The LuGre friction model is a first-order dynamical system of the form

$$\begin{aligned} F &= \sigma_0 z + \sigma_1 \dot{z} + \alpha_2 v \\ \frac{d}{dt} z &= v - \sigma_0 \frac{|v|}{g_{fr}(v)} z \end{aligned} \tag{8}$$

Here z is the internal state of the model; v corresponds to the relative velocity between two surfaces in contact, and it is assumed that it is measurable. The function $g_{fr}(v)$ has usually the form

$$g_{fr}(v) = \alpha_0 + \alpha_1 e^{-v^2/v_0^2}, \quad \alpha_0 > 0, \quad \alpha_1 > 0.$$

In the further development, we will not take into consideration the exact form of g_{fr}, while only the property that $g_{fr}(v)$ is positive and strictly separated from 0, will be used.

The main contribution of the paper is the modification of the nominal controller $u = H(x,t)$ to a new one that enables to guarantee asymptotic stability of the set (2). In the course of doing this, none of the properties of the model (8) have been used except the fact that its state is bounded, and the dynamical system (8) itself is time-invariant and has a very particular structure. It is worth mentioning another approach to deal with the closely related problem of position tracking developed for the mechanical systems in (Canudas de Wit and Kelly, 1997).

2. MAIN RESULT

To compensate for the friction it is suggested to modify the nominal controller of (1) as follows

$$u_{nom} = H(x,\, t) \quad \rightarrow \quad u = H(x,\, t) + \hat{F}$$

where the friction estimate $\hat{F}$ is reconstructed by an observer

$$\begin{aligned} \hat{F} &= \sigma_0 \hat{z} + \sigma_1 \dot{\hat{z}} + \alpha_2 v \\ \frac{d\hat{z}}{dt} &= v - \sigma_0 \frac{|v|}{g_{fr}(v)} \hat{z} + K \\ &\widehat{=}\; v - \varphi(v)\hat{z} + K, \qquad \hat{z}(0) \;\; = 0 \end{aligned} \tag{9}$$

Here the observer gain K (i. e., the injection function) could be seen as a variable to be defined. In the proof of Theorem 1 an expression for the function K is derived and presented.

Theorem 1. Given the system (1), (7), (8), consider the controller

$$u = H(x,\, t) + \hat{F} \tag{10}$$

where the friction estimate $\hat{F}$ is defined by (9).

For $\rho > 4\, Q_2\, \sigma_1\, \sigma_0$ the observer injection function chosen as

$$K = \frac{2(H(x,\, t) Q_2 + Q_{12} x)(\sigma_0 + \sigma_1 \varphi)}{(\rho - 4 Q_2 \sigma_1 \sigma_0)} \tag{11}$$

will guarantee that along any solution

$$\left[x(t),\, z(t),\, \hat{z}(t)\right]$$

of the closed loop system the limit relation

$$\lim_{t \to +\infty} V(x(t)) = 0 \tag{12}$$

holds, that is, $x(t)$ converges to the compact set V_0 defined in (2). ■

Proof. The dynamics of the error e between the 'true' internal friction state z and its estimate $\hat{z}$, that is

$$e = z - \hat{z}$$

satisfies the equation

$$\dot{e} = -\sigma_0 \frac{|v|}{g_{fr}(v)} e - K \tag{13}$$

Let us consider a Lyapunov function candidate for the system augmented by the friction model (8) and the observer (9) as

$$W = V + \frac{\rho}{2} e^2 \tag{14}$$

Then, the time derivative is

$$\begin{aligned} &\frac{d}{dt} W = \dot{V} + \rho\, e\, \dot{e} \;=\; \dot{V} + \rho\, e(-\sigma_0 \frac{|v|}{g_{fr}(v)} e - K) \\ &\leq - \begin{bmatrix} x \\ (H + \hat{F}) - F \end{bmatrix}^T Q(t) \begin{bmatrix} x \\ (H + \hat{F}) - F \end{bmatrix} \\ &+ \rho\, e(-\sigma_0 \frac{|v|}{g_{fr}(v)} e - K) \\ &= - \begin{bmatrix} x \\ H \end{bmatrix}^T Q(t) \begin{bmatrix} x \\ H \end{bmatrix} - 1 \cdot \alpha e^2 + \beta e + \gamma \end{aligned} \tag{15}$$

where

$$\begin{aligned} \alpha &= \rho\varphi + Q_2(\sigma_0 - \sigma_1\varphi)^2 \\ \beta &= 2xQ_{12}\sigma_0 - 2xQ_{12}\sigma_1\varphi - \rho K + 2HQ_2\sigma_0 \\ &\quad - 2HQ_2\sigma_1\varphi + 2Q_2\sigma_1 K\sigma_0 - 2Q_2\sigma_1^2 K\varphi \\ \gamma &= -2Q_{12}x\sigma_1 K - 2HQ_2\sigma_1 K - Q_2\sigma_1^2 K^2 \end{aligned} \tag{16}$$

$$\varphi = \sigma_0 \frac{|v|}{g_{fr}(v)} > 0$$

By completion of squares, it holds that

$$-\alpha e^2 + \beta e = -\left(\sqrt{\alpha}e - \frac{\beta}{2\sqrt{\alpha}}\right)^2 + \frac{\beta^2}{4\alpha}$$

Thus

$$\frac{d}{dt}W \leq -\begin{bmatrix} x \\ H \end{bmatrix}^T Q(t) \begin{bmatrix} x \\ H \end{bmatrix} - \left(\sqrt{\alpha}e - \frac{\beta}{2\sqrt{\alpha}}\right)^2 + \frac{\beta^2}{4\alpha} + \gamma \tag{17}$$

As the first two terms of Eq.(17) are negative (semi-)definite, it suffices to check the sign of

$$\frac{\beta^2}{4\alpha} + \gamma \tag{18}$$

These two terms can be written as a quadratic polynomial in the undetermined function K.

For simplicity we show the calculations for the particular case of $Q_{12} = 0$ and further down show the result for $Q_{12} \neq 0$.

$$\frac{\beta^2}{4\alpha} + \gamma = \frac{aK^2 + bK + c}{f} \tag{19}$$

where

$$\begin{aligned} a &= -\rho(-\rho + 4Q_2\sigma_1\sigma_0) \\ b &= -4\rho H Q_2(\sigma_0 + \sigma_1\varphi) < 0 \\ c &= \left(2HQ_2(\sigma_0^2 - \sigma_1\varphi)\right)^2 > 0 \\ f &= 4((\sigma_0 - \sigma_1\varphi)^2 Q_2 + \rho\varphi) > 0 \end{aligned} \tag{20}$$

For a large enough value of ρ , a will be positive. As known, the minimum of the polynomial

$$aK^2 + bK + c$$

with $a > 0$ is achieved at

$$K_{min} = -\frac{b}{2a} = \frac{-2\,Q_2\,(\sigma_0 + \sigma_1\,\varphi)}{(-\rho + 4\,Q_2\,\sigma_1\,\sigma_0)}H \tag{21}$$

and equals to

$$\min_K \left\{aK^2 + bK + c\right\} = -\frac{4\,Q_2^2\,H^2\,\sigma_1\,\sigma_0}{(\rho - 4\,Q_2\,\sigma_1\,\sigma_0)} \tag{22}$$

The minimum is negative provided that

$$\rho - 4\,Q_2\,\sigma_1\,\sigma_0 > 0$$

which is the same condition for a being positive in (20), and will be satisfied for a large enough value ρ.

For the case of $Q_{12} \neq 0$, we get

$$K_{Q_{12}\neq 0} = \frac{2(Q_2H + Q_{12}x)(\sigma_0 + \sigma_1\varphi)}{(\rho - 4Q_2\sigma_1\sigma_0)} \tag{23}$$

and the minimum is

$$-4\sigma_0\sigma_1\frac{(Q_2H + Q_{12}x)^2}{(\rho - 4Q_2\sigma_1\sigma_0)} < 0 \text{ for } \rho > 4Q_2\sigma_1\sigma_0$$

Therefore, if the observer gain K is chosen as in (23), or the same as in (11), then the time derivative of the function W along the solution of the closed loop system looks as

$$\frac{d}{dt}W \leq -\begin{bmatrix} x \\ H \end{bmatrix}^T Q(t) \begin{bmatrix} x \\ H \end{bmatrix} - (\sqrt{\alpha}e - \frac{\beta}{2\sqrt{\alpha}})^2 - \frac{4\,(Q_2H + Q_{12}x)^2\,\sigma_1\,\sigma_0}{(\rho - 4\,Q_2\,\sigma_1\,\sigma_0)} \leq 0 \tag{24}$$

To analyze the behavior of the closed loop system solutions based on the differential relation (24), it is worth to mention one property of the LuGre friction model (8)

Lemma 1. For any initial condition z_0, the solution $z(t)$ of the system (8) is bounded. ■

Equipped with this fact one can readily prove the feedback controller (10) asymptotically stabilize the set

$$V_0 = \left\{x : V(x) = 0\right\}$$

Indeed, choose any initial condition x_0, z_0, $\hat{z}_0$ and consider the corresponding solution

$$[x(t),\, z(t),\, \hat{z}(t)] = [x(t, x_0),\, z(t, z_0),\, \hat{z}(t, \hat{z}_0)]$$

of the closed loop system (1), (8), (10), (9) with the observer gain (11). Along this solution the nonnegative function W satisfies relation (24), and is non-increasing.

Integrating the differential relation (24) from 0 to T, one gets

$$\begin{aligned} 0 &\geq W(x(T), e(T), T) - W(x(0), e(0), 0) \\ &= V(x(T), T) - V(x(0), 0) \\ &+ \frac{\rho}{2}(z(T) - \hat{z}(T))^2 - \frac{\rho}{2}(z(0) - \hat{z}(0))^2 \\ &\leq -\varepsilon \int_0^T V(x(t), t)dt - \int_0^T (\sqrt{\alpha}e(t) - \frac{\beta}{2\sqrt{\alpha}})^2 dt \\ &- \int_0^T \frac{4\,(Q_2H + Q_{12}x)^2\,\sigma_1\,\sigma_0}{(\rho - 4\,Q_2\,\sigma_1\,\sigma_0)}dt \end{aligned} \tag{25}$$

This inequality together with (3) and a radially unbounded V implies

- $x(t)$ is bounded;
- $(z(t) - \hat{z}(t))$ is bounded, therefore $\hat{z}(t)$ is bounded due to boundedness of $z(t)$ coming from Lemma 1;
- $$\int_0^{+\infty} V(x(t), t)dt = \int_0^{+\infty} |y(x(t), t)|^2 dt < +\infty$$

  The right-hand side of the closed loop system is bounded in some neighborhood of the compact set Ω, then Barbalat's lemma helps us to conclude that $y(x(t), t)$ converges to zero as time t tends to infinity.

3. EXAMPLE: STABILIZATION OF PERIODIC ORBITS IN FURUTA PENDULUM

3.1 *Generation and Exponential Stabilization of Periodic Regimes for the Furuta Pendulum when Friction is absent*

In this section we consider a rotational pendulum (Fig. 1), named after Professor K. Furuta, rendering it to oscillate around its downward equilibrium. The equation of motion for the Furuta pendulum is

$$M(\theta)\begin{bmatrix}\ddot{\phi}\\ \ddot{\theta}\end{bmatrix}+C(\theta,\dot{\theta},\dot{\phi})\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\end{bmatrix}+G(\theta)=\begin{bmatrix}\tau_\phi\\ 0\end{bmatrix} \quad (26)$$

where ϕ is the angle of the arm; θ is the angle of the pendulum; τ_ϕ is a control torque that could be applied to the arm. The matrices M, C, G are

$$\begin{aligned} M &= M(\theta) = \begin{bmatrix}\alpha+\beta\sin^2\theta & \gamma\cos\theta\\ \gamma\cos\theta & \beta\end{bmatrix}\\ C &= C(\theta,\dot{\theta},\dot{\phi})\\ &= \begin{bmatrix}\beta\dot{\theta}\sin\theta\cos\theta & (\beta\dot{\phi}\cos\theta-\gamma\dot{\theta})\sin\theta\\ -\beta\dot{\phi}\sin\theta\cos\theta & 0\end{bmatrix}\\ G &= G(\theta) = \begin{bmatrix}0\\ -\delta\sin\theta\end{bmatrix} \end{aligned} \quad (27)$$

For the system in Fig. 1, the physical parameters are

$$\begin{aligned} &\alpha = 0.00354,\ \beta = 0.00384,\\ &\gamma = 0.00258,\ \delta = 0.103 \end{aligned} \quad (28)$$

For details on the modeling and other contributions to the friction compensation for the Furuta pendulum, see (Gäfvert *et al.*, 1999), (Shiriaev and de Wit, 2004) and references therein.

The following five-step procedure describes details of constructing the feedback controller:

1) Choice of Virtual Holonomic Constraint and Properties of Virtual Limit System. To generate oscillations of the Furuta pendulum, we have chosen a particular *virtual holonomic constraint*

$$\phi = a \quad (29)$$

where a is a given constant angle of the arm. If a feedback controller successfully stabilizes the constraint (29), then remaining dynamics of (26) is covered by the equation

$$\beta\,\ddot{\theta}-\delta\,\sin\theta = 0. \quad (30)$$

The system (30) describes the motions of a one-degree of freedom pendulum, its phase portrait around its downward equilibrium and the corresponding solutions versus time are shown in Figure 2. It is well known that the system (30)

Fig. 1. Furuta pendulum, Dept. of Automatic Control, Lund University.

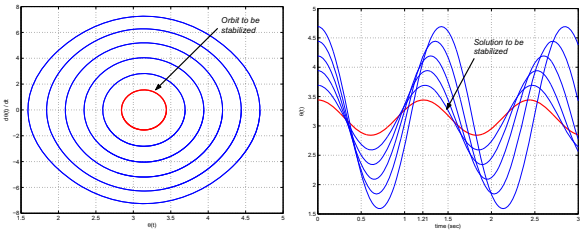


Fig. 2. *(Left)* The phase portrait of the *virtual limit system* (30). *(Right)* The solutions of the *virtual limit system* (30) versus time.

has the first integral (the total energy E of the mathematical pendulum)

$$E(\theta,\dot{\theta}) = \frac{\beta}{2}\,\dot{\theta}^2+\delta\,\cos\theta, \quad (31)$$

while its time derivative satisfies the passivity relation, i. e., along any solution of (30)

$$\frac{d}{dt}\,E = \dot{\theta}\cdot f, \quad (32)$$

where f is an external (control) torque.

2) Choice of Orbit. Figure 2 shows in red the orbit we have chosen to stabilize and the corresponding solution $\theta_\gamma(t)$ of the system (30) versus time. It can be is seen that the chosen solution has the period $T \approx 1.21$ sec.

3) Partial Feedback Linearization of (26). Introduce the regulated output y as

$$y = \phi - a,$$

then straightforward calculations show that an equivalent (partly linear) form of (26) looks as

$$\beta\,\ddot{\theta}-\delta\,\sin\theta = \beta\dot{y}^2\sin\theta\cos\theta-\gamma\cos\theta v \quad (33)$$

$$\ddot{y} = v \quad (34)$$

where the new control input v relates to the true control input τ_ϕ as follows

$$\tau_\phi = \frac{v-A(\theta,\dot{\theta},\dot{y})}{B(\theta)} \quad (35)$$

with

$$B = \frac{\beta}{\alpha\beta - \gamma^2 + [\beta^2 + \gamma^2]\sin^2\theta} \quad (36)$$

$$A = \frac{B}{\beta}\{\beta\left(-2\beta\dot{y}\dot{\theta}\sin\theta\cos\theta + \gamma\,\dot{\theta}^2\,\sin\theta\right) - \gamma\cos\theta\left(\beta\,\dot{y}^2\cos\theta\sin\theta + \delta\,\sin\theta\right)\} \quad (37)$$

Based on the passivity relation (32), one can rewrite the equations (33)–(34) into a new (but not equivalent) form

$$\dot{E} = \dot{\theta}\left(\beta\cdot\dot{y}^2\cdot\sin\theta\cdot\cos\theta - \gamma\cdot\cos\theta\cdot v\right) \quad (38)$$

$$\ddot{y} = v \quad (39)$$

4) Linear Auxiliary System and Its Controllability Consider the equations (38)–(39) and formally substitute on the right-hand sides the periodic solution

$$\theta = \theta_\gamma(t),\ \dot{\theta} = \dot{\theta}_\gamma(t),\ y = 0,\ \dot{y} = 0 \quad (40)$$

which we have chosen to stabilize. Then we have got a linear periodic system

$$\dot{E} = \underbrace{-\dot{\theta}_\gamma(t)\cdot\gamma\cdot\cos\theta_\gamma(t)}_{\rho(t)}\cdot v, \qquad \ddot{y} = v \quad (41)$$

One could rewrite Eq. (41) in state-space form

$$\frac{d}{dt}\begin{bmatrix} E \\ y \\ \dot{y}\end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0\end{bmatrix}}_{A}\begin{bmatrix} E \\ y \\ \dot{y}\end{bmatrix} + \underbrace{\begin{bmatrix} \rho(t) \\ 0 \\ 1\end{bmatrix}}_{b(t)} v \quad (42)$$

It has been checked that the system (42) is completely controllable over the period $T \approx 1.21$ sec. The controllability opens the way for constructing stabilizing controller for the linear system (42). Given a 3×3 matrix $G = G^T > 0$ and a scalar $\Gamma > 0$, consider the LQR problem with the following minimization criterion

$$\arg\ \min_v \jmath = \arg\ \min_v \int_0^{+\infty}\left\{\xi(t)^T G\xi + \Gamma v(t)\right\}dt$$

with $\xi = [E,\ y,\ \dot{y}]^T$. As known, to solve the LQR-problem, one needs to find a stabilizing solution $R(t)$ of the Lur'e-Riccati equation

$$\dot{R}(t) = -A(t)^T R(t) - R(t)A(t) - G + R(t)\,b(t)\Gamma^{-1}b(t)^T R(t) \quad (43)$$

Then, the stabilizing feedback controller for (42) is

$$v = -\frac{b(t)^T R(t)\xi}{\Gamma} = -\left[\frac{\rho(t)}{\Gamma}, 0, \frac{1}{\Gamma}\right]R(t)\begin{bmatrix} E \\ y \\ \dot{y}\end{bmatrix} \quad (44)$$

Following this way for the system (42), one might get a controller with unacceptable *slow* rate of convergence. To improve convergence, one could try to shift the eigenvalues of the closed loop system monodromy matrix within a disc of the complex plane with radius smaller than 1. This could be done, for example, by solving a linear quadratic optimization problem for the modified system

$$\frac{d}{dt}\begin{bmatrix} E \\ y \\ \dot{y}\end{bmatrix} = \{A + \varepsilon I_3\}\begin{bmatrix} E \\ y \\ \dot{y}\end{bmatrix} + \begin{bmatrix} \rho(t) \\ 0 \\ 1\end{bmatrix} v \quad (45)$$

In contrast to Eq. (42), we have added to the linear part of the system a destabilizing term εI_3 with $\varepsilon > 0$. In the simulations the numerical solution $R(t)$ of the Riccati equation (43) has been found when

$$G = I_3, \quad \Gamma = 0.1, \quad \varepsilon = 0.2.$$

5) Final Form of Controller and Simulation of Closed Loop System. The final form of the stabilizing controller for the pendulum system (33)–(34) looks as (35) with

$$v = -\Gamma^{-1}\left[-\dot{\theta}\gamma\cos\theta,\ 0,\ 1\right]^T R(t)\begin{bmatrix} \tilde{E} \\ y \\ \dot{y}\end{bmatrix}, \quad (46)$$

$$\tilde{E} = \underbrace{(\frac{\beta}{2}\,\dot{\theta}^2 + \delta\,\cos\theta)}_{E(\theta,\dot{\theta})} - \underbrace{(\frac{\beta}{2}\,\dot{\theta}_\gamma^2(0) + \delta\,\cos\theta_\gamma(0))}_{E(\theta_\gamma(0),\dot{\theta}_\gamma(0))}$$

$$y = \phi - a, \quad \dot{y} = \dot{\phi}$$

3.2 *Orbital stabilization in Presence of Friction*

In case when the friction is present in the loop we can utilize the main result of the paper to recover the desired performance. To apply Theorem 1, one needs to transform the Furuta pendulum (26) and the feedback controller (35), (46) into the form of the closed loop system (1).

It could be readily done, if one assumes that the function $F(x)$ representing the right-hand side of (1) for the Furuta pendulum is composed of its dynamics together with the feedback transformation (35), that is, the control variable u in (1) is now the variable v in (35). Such a feedback transformed system has the cycle, shown in red on Figure 2, as a solution. In this case, the controller $u = H(x,t)$ in (1) is defined by (46), while the Lyapunov function V in (2)–(3) is

$$V(\theta,\dot{\theta},\phi,\dot{\phi},t) = x^{\,T}R(t)x \quad (47)$$

$$\text{where } x = \begin{bmatrix} E(\theta,\dot{\theta}) - E(\theta_0,\dot{\theta}_0) \\ \phi \\ \dot{\phi}\end{bmatrix} \quad (48)$$

with $[\theta_0,\dot{\theta}_0]$ belonging to the cycle and $R(t) = R(t+T)$, $T = 1.21$ sec. being a stabilizing solution of the Lur'e-Riccati equation (43). One could check that it satisfies (4).

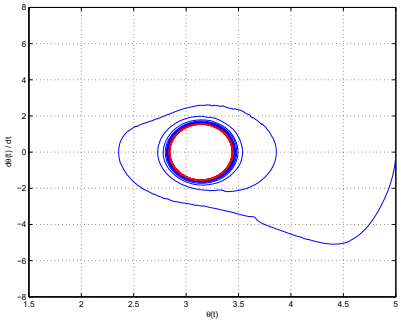


Fig. 3. The behavior of $[\theta(t), \dot{\theta}(t)]$ in the closed loop system with the feedback controller (35) when both friction and friction compensation are present. Here the orbit chosen to be stabilized is shown in red.

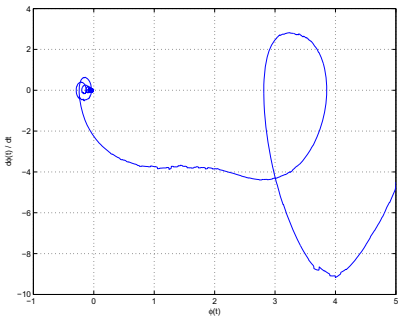


Fig. 4. The behavior of the arm $[\phi(t), \dot{\phi}(t)]$ in the closed loop system with the feedback controller (35) when both friction and friction compensation are present.

Figures 3–6 show the behavior of the closed loop system when both friction, friction compensation and white noise in all the measurements are added. Here the LuGre model parameters are chosen as

$$\alpha_0 = 0.21, \quad \alpha_1 = 0.022, \quad \alpha_2 = 0$$
$$\sigma_0 = 80, \quad \sigma_1 = 1.5, \quad v_0 = 0.1$$

while the value of control parameter ρ is 12000. As we can seen in Figure 6, the friction observer will produce a very good friction estimate after the initial transient.

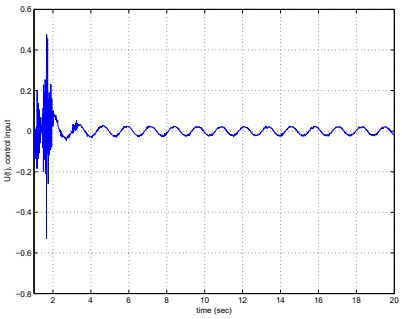


Fig. 5. The control signal u in the closed loop system with the active friction compensation.

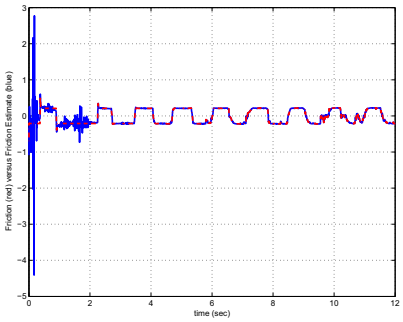


Fig. 6. The value of friction (red) and friction estimate (blue) in the closed loop system.

4. CONCLUSIONS

This paper deals with the problem of friction compensation in nonlinear control systems. It is assumed that the friction has a particular structure, the so called LuGre model, and it is also assumed that a nominal controller has been derived for stabilizing the system to a certain equilibrium or to a desired limit cycle motion, specified—e. g., via *virtual holonomic constraints*, without taking the friction disturbance into account. The main result of the paper suggests a friction observer and modification of the control in such a way that the overall closed loop system preserves the asymptotic stability of the desired attractive set that was originally obtain by the controller for the frictionless case. Finally, the design of a stabilizing (nominal) control law used together with the proposed friction compensation method has been evaluated in a simulation study for the Furuta pendulum.

5. REFERENCES

Canudas de Wit, C. and R. Kelly (1997). Passivity based control design for robots with dynamical friction. In: *Proceedings of the 5th IASTED International Conference*. Mexico. pp. 84–88.

Canudas de Wit, C., H. Olsson, K.J. Åström and P. Lischinsky (1995). A new model for control of systems with friction. *IEEE Transactions on Automatic Control* **40**, 419–425.

Gäfvert, M., J. Svensson and K.J. Åström (1999). Friction and friction compensation in the Furuta pendulum. In: *Proceedings of the 5th European Control Conference*. Karlsruhe, Germany. CM-4:3.

Shiriaev, A., A. Robertsson and R. Johansson (2003). Friction compensation for passive systems based on the LuGre model. In: *Proc. 2nd IFAC Workshop on Lagrangian and Hamiltonian Methods for Nonlinear Control,*. Seville, Spain. pp. 183–188.

Shiriaev, A. and C. Canudas de Wit (2004). Virtual constraints: a tool for orbital stabilization of nonlinear systems. submitted to *NOLCOS'2004*.

E. Bibliography

Alazard, D. (2002): *La regulation LQ/LQG, notes de cours SUPAERO.*

Åström, K. J. (1998): "Control of systems with friction." In *The Fourth International Conference on Motion and Vibration Control, MoViC*, pp. 25–32. Zurich, Switzerland.

Åström, K. J. and K. Furuta (1996): "Swinging up a pendulum by energy control." In *IFAC'96, Preprints 13th World Congress of IFAC*, vol. E, pp. 37–42. San Francisco, California.

Dietz, T. (2006): "Model-based friction compensation for the Furuta pendelum using the LuGre Model," (Modellbaserad friktionskompensering för furuta-pendel med hjälp av lugre-modellen). Master's Thesis ISRN LUTFD2/TFRT--5773--SE. Department of Automatic Control, Lund University, Sweden.

Freidovich, L., A. Robertsson, A. Shiriaev, and R. Johansson (2006): "Friction compensation based on LuGre model." In *Proceedings of the 45th IEEE Conference on Decision and Control.* San Diego, CA, USA.

Izutsu, M. and K. Furuta (2007): "Design of a model following stabilizer to an artificial gravity control model." *SICE, 2007 Annual Conference*, Sept., pp. 1252–1257.

Olsson, H. (1996): *Control Systems with Friction.* PhD thesis ISRN LUTFD2/TFRT--1045--SE, Department of Automatic Control, Lund University, Sweden.

Robertsson, A., A. Shiriaev, and R. Johansson (2004): "Friction compensation for nonlinear systems based on the LuGre model." In *Preprints 2004 6th IFAC-Symposium on Nonlinear Control Systems (NOLCOS 2004).* Stuttgart, Germany.

Shiriaev, A., A. Robertsson, and R. Johansson (2003): "Friction compensation for passive systems based on the LuGre model." In *Proc. 2nd IFAC Workshop on Lagrangian and Hamiltonian Methods for Nonlinear Control,*, pp. 183–188. Seville, Spain.