Ecole Nationale Supérieure d'Electrotechnique, d'Electronique, d'Hydraulique et d'Informatique de Toulouse
Filière Informatique et Mathématiques Appliquées

Robustness to jitter in real time system

Nicolas Andreff

June 1994

Travail de fin d'études effectué au département d'Automatique
de l'Ecole Polytechnique de Lund (Lunds Tekniska Högskola), Suède.

Abstract

Ce travail concerne le contrôle d'un moteur en présence de retards.

Abstract

This work considers control of a motor in presence of delays. Design equations are derived based on polynomial design. The design depends on the designer's choices, i.e. desired closed-loop behaviour, expected delay, observer poles and sampling period.

The main part of this work studies the influence of these choices on the stability of the system : stability domains in terms of delays are computed, a gain scheduling controller is simulated and a trade off between robustness to delays and load disturbance cancellation is found when placing the observer poles.

Contents

1 Introduction

In computer control the sampling time is an important notion. As a computer is a discrete system, the signals coming from (resp. going to) a continuous system have to be sampled (resp. digitalized). This is done at determined instants via sensors (resp. actuators). Usually, the sampling is done periodically, and then we talk of sampling period or sampling interval.

The problem we deal with in this work is called *jitter*. What does it mean ? Well, there can be some time varying phenomena in the computer. A good illustration of such processes is the example of a buffer in a distributed architecture. The queuing in the buffer is such that if an information comes in (from a sensor, for instance), it can not be decided when it will be treated. This will create a delay between the sensor and the computer. There are many other sources to time varying delays between the sensors and the computers and similarly between the computer and the actuators.

The problem with these time varying delays is that they deteriorate the behaviour of the closed loop system (response to a control signal, stability). Our goal is to investigate the influence of jitter and to try to compensate for these delays. To achieve our task, we will control a DC-motor and we will collect all the delays in a unique one between the motor and the controller.

This report contains 10 parts including introduction and conclusion. In section 2, the formulas used to design the various controllers are derived. In section 3, we present how the simulations were done and stochastic models for the delays. Sections 4–9 contain the results of these simulations. Specifically, section 4 deals with the behaviour of the uncompensated controller. In section 5 are introduced the first compensated controllers. Then, section 6 repeats the simulations of section 4 for a controller which compensates for a constant delay. Section 7 presents a gain scheduling controller. Robustness is discussed in section 8. Finally, in section 9, we investigate the influence of the sampling period on the sensitivity to delays.

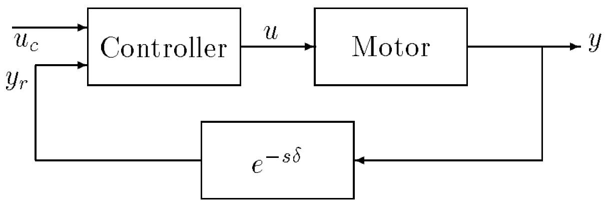

Figure 1: Block diagram of the system we study

2 System characteristics

In this section, we define the system to be studied. The system consists of a motor to be controlled by a controller in presence of delays (see Figure 1).
First, consider the case when there is no delay, $\delta = 0$.

2.1 In absence of delay

We here define the characteristics of the motor and give the equations needed to design an RST controller to control it.

2.1.1 Motor

The motor has a continuous time transfer function of the form :

$$G(s) = \frac{K}{s(1 + sT_c)} \tag{1}$$

where K and T_c are constant.
The continuous-time state-space representation of the motor can be written as :

$$\left\{ \begin{array}{l} \frac{dx}{dt}(t) = Ax(t) + Bu(t) \\ y(t) = Cx(t) \end{array} \right. \tag{2}$$

where A, B and C are defined by :

$$A = \begin{bmatrix} 0 & 1 \\ 0 & -\frac{1}{T_c} \end{bmatrix}, \; B = \begin{bmatrix} 0 \\ \frac{K}{T_c} \end{bmatrix}, \; C = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{3}$$

As a discrete-time RST controller will be designed, the pulse-transfer function has to be determined. Discretizing the system with a constant period h, we obtain the following results.

$$\left\{ \begin{array}{l} x(kh + h) = \Phi x(kh) + \Gamma u(kh) \\ y(kh) = Cx(kh) \end{array} \right. \tag{4}$$

where Φ and Γ are defined by :

$$\Phi = \begin{bmatrix} 1 & (1-a)T_c \\ 0 & a \end{bmatrix}, \; \Gamma = \begin{bmatrix} h - (1-a)T_c \\ 1 - a \end{bmatrix} \text{ with } a = e^{-\frac{h}{T_c}} \tag{5}$$

Hence, the pulse-transfer function is given by :

$$H(z) = \frac{B(z)}{A(z)} = \frac{K\left[(h-(1-a)T_c)\,z + ((1-a)T_c - ah)\right]}{(z-1)(z-a)} \tag{6}$$

2.1.2 RST design

We will now determine the controller via an RST design. It means that the control law can be written in the polynomial form (Åström-Wittenmark(1990), chapter 10) :

$$R(q)u(k) = T(q)u_c(k) - S(q)y(k) \tag{7}$$

As the controller is chosen to handle the load disturbances, the polynomial R must be of the form $R = (z-1)R_1$, i.e. there is an integrator in the controller.
The desired closed-loop pulse-transfer function is then defined :

$$H_m(z) = \frac{B_m(z)}{A_m(z)} = \frac{1+p_1+p_2}{B(1)} \frac{B(z)}{z^2+p_1 z+p_2} \tag{8}$$

Using the same notations as in Åström-Wittenmark(1990), we determine the following equations to be solved.

$$deg A_o \geq 2 \tag{9}$$

$$\left\{ \begin{array}{l} T = A_o B'_m \\ (z-1)AR_1 + BS = A_o A_m \end{array} \right. \tag{10}$$

where A_o can be interpreted as an observer polynomial, which is a stable polynomial chosen by the designer.

2.2 In presence of delays

When there are delays between the sampler and the controller, the dynamics of the system change. The pulse-transfer function has to be recomputed. Then the controller will be redesigned, taking into account the modifications.

2.2.1 Motor and delay

This section presents the dynamics of the sub-system containing the motor and the delay element (i.e. everything but the controller).

Consider the case of a delay δ of the form :

$$\delta = dh + \tau \qquad d \geq 0\ ,\ 0 \leq \tau < h$$

where h is the sampling period.

Note y_r the delayed output signal. (y_r is the signal received by the controller instead of the motor output signal)

Then, using a new state x_d which corresponds to the delayed state x (see (4)), the discrete-time state-space representation of the sub-system can be written as :

$$\left\{ \begin{array}{l} x_d(kh+h) = \Phi x_d(kh) + \Gamma_0 u(kh - dh + h) + \Gamma_1 u(kh - dh) \\ y_r(kh) = C x_d(kh) \end{array} \right. \tag{11}$$

where Φ is the same as in (5) and Γ_0, Γ_1 are defined as :

$$\begin{array}{l} \Gamma_0 = \left[\begin{array}{c} \gamma_{01} \\ \gamma_{02} \end{array} \right] = \left[\begin{array}{c} K\left[h - \tau - T_c\left(1 - ae^{\frac{\tau}{T_c}}\right)\right] \\ K\left(1 - ae^{\frac{\tau}{T_c}}\right) \end{array} \right] \\ \\ \Gamma_1 = \left[\begin{array}{c} \gamma_{11} \\ \gamma_{12} \end{array} \right] = \left[\begin{array}{c} K\left[\tau + aT_c\left(1 - e^{\frac{\tau}{T_c}}\right)\right] \\ -Ka\left(1 - e^{\frac{\tau}{T_c}}\right) \end{array} \right] \end{array} \tag{12}$$

Hence, we obtain the pulse-transfer operator between the control signal and the delayed output signal y_r :

$$\begin{aligned} H_\delta(z) &= \tfrac{B_\delta(z)}{A_\delta(z)} \\ &= \tfrac{\gamma_{01} z^2 + (\gamma_{11} - a\gamma_{01} + T_c(1-a)\gamma_{02})z + (-a\gamma_{11} + T_c(1-a)\gamma_{12})}{z^{d+1}(z-1)(z-a)} \end{aligned} \tag{13}$$

Remark : If $\tau = 0$, there is a common factor z in A_δ and B_δ. B_δ is actually equal to zB.

2.2.2 RST design

We would like to keep the same observer and the same closed-loop pulse-transfer function as in the no-delay case. But, Theorem 10.2 in Åström-Wittenmark(1990), requires the following condition to be fullfilled :

$$degA_m - degB_m \geq degA_\delta - degB_\delta$$

This can not be done, because

$$degA_m - degB_m = 1$$

and

$$degA_\delta - degB_\delta = d + 1.$$

The degree of A_m must therefore be increased by d. The closed-loop pulse-transfer operator is then chosen of the form :

$$H_{md}(z) = \frac{B_{md}(z)}{A_{md}(z)} = \frac{B_m(z)}{z^d A_m(z)} \tag{14}$$

The second condition of Theorem 10.2 gives thus :

$$degA_{od} \geq d + 4 \quad if\ \tau \neq 0 \tag{15}$$

$$degA_{od} \geq d + 3 \quad if\ \tau = 0 \tag{16}$$

As the polynomials R, S and T should be adapted to any unknown delay, the new observer is taken as :

$$A_{od} = A_o A_\tau \tag{17}$$

where both A_o and A_τ are stable polynomials.
Relations (15) and (16) become then

$$deg A_\tau \geq d+2 \quad if\ \tau \neq 0 \tag{18}$$

$$deg A_\tau \geq d+1 \quad if\ \tau = 0. \tag{19}$$

Equations (6) and (13) give

$$A_\delta = z^{d+1} A. \tag{20}$$

With this, the input-output relationship for the closed-loop system is

$$y = \frac{z^{d+1} BT}{z^{d+1} AR + B_\delta S} u_c. \tag{21}$$

Notice that this is the discrete-time pulse transfer function from u_c to y.
The following relationship is thus obtained.

$$\frac{z^{d+1} BT}{z^{d+1} AR + B_\delta S} = \frac{B_{md}}{A_{md}} = \frac{B_{md} A_o A_\tau}{A_{md} A_o A_\tau}. \tag{22}$$

Using

$$B_{md} = B_m = B B_m^{'},$$

and equating the numerators (resp. denominators) in (22), we get

$$z^{d+1} T = B_m^{'} A_o A_\tau \tag{23}$$

and

$$z^{d+1} AR + B_\delta S = A_{md} A_o A_\tau. \tag{24}$$

As A_τ has a variable degree according to the delay, the easiest is to choose it as

$$A_\tau = z^n$$

where n is minimal and satisfies (18)–(19). From (14),(20),(23),(24) and the latter, as well as from the fact that the controller contains an integrator, we finally derive the system of equations which determines the polynomials R, S and T.

$$\begin{cases} T = B_m^{'} A_o z \\ z^{d+1} A(z-1) R_1 + B_\delta S = A_m A_o z^{2d+2} \end{cases} \quad \text{if } \ \tau \neq 0 \tag{25}$$

Furthermore, the remark about (13) solves the case of a delay equal to a multiple of the sampling period :

$$\begin{cases} T = B_m^{'} A_o \\ z^d A(z-1) R_1 + BS = A_m A_o z^{2d} \end{cases} \quad \text{if } \ \tau = 0 \tag{26}$$

3 Simulations

This section first describes the simulation that have been made, then defines the system parameters, and finally, models the stochastic delays.

3.1 Organization of the simulations

The simulation part begins by a study of the behaviour of the system in presence of (constant or stochastic) delays and without any attempt to cancel them.

Then section 2.2.2 was used to design an RST-controller which could treat the problem of a constant delay. It was used in the different cases. First, we verified that our design was correct by testing several constant delays. Then, using a controller which was designed for a constant delay equal to 10% of the sampling interval, constant and stochastic delays were introduced into the system.

As the design method works for any delay, why not use it to design a controller which will adapt itself to the actual delay ? This means that the design method will not be used once and for all, but at every sampling interval. This seems particularly useful if the delays can be measured or if there is a method to estimate their value (e.g. periodic delays). In the case of the simulation, the delays are generated and therefore known by the program. They can then be considered as measured. The efficiency of the design can thus be tested.

Later, robustness methods have been used. By varying the observer poles, the sensitivity of the system to the delays can be modified. This characteristic was used on both the controller without delay compensation and the controller which compensates for a delay equal to 10% of the sampling interval.

Finally, the influence of the sampling interval on the sensitivity to the delays was investigated.

3.2 Parameters

Below is a list of the various parameters that define the system. The numerical values are those we used in the first part of this study.

T_c time constant of the motor : 0.5 s,

K gain factor of the motor : 1000,

h sampling period : 0.01 s,

p_1, p_2 coefficients of A_m : are defined from ω and ξ (see below),

ω natural frequency : 190 rad/s ($\simeq$ 30Hz),

ξ relative damping : 0.707.

Furthermore, we chose to have the observer poles at the origin.

3.3 Modelling the stochastic delays

As we will study the case of stochastic delays, we have to determine how to model the stochastic variations. We used two models.

The first one is of the form :

$$\mathcal{M}_1 \ : \ \delta(t) = 2\bar{\delta} rect(t)$$

where $rect(t)$ is the realization at time t of a uniformly distributed stochastic variable in the interval [0,1]. Notice that $\delta(t)$ takes its values in $[0, 2\bar{\delta}]$ and that its mean value is $\bar{\delta}$. Notice also that the delay is changing every sampling period. This model can represent the case when we know that the delay is bounded but we don't know which value it actually takes.

We can also model the delays in the following way :

$$\mathcal{M}_2 \ : \ \delta(t) = \bar{\delta} + \alpha\bar{\delta}(2rect(t) - 1)$$

where $\bar{\delta}$ is the mean value of the delay and α is a parameter that gives the amplitude around the mean value as a factor of the mean value. In this case, δ takes its values in $[\bar{\delta} - \alpha\bar{\delta}, \bar{\delta} + \alpha\bar{\delta}]$. Notice again that the delay is changing every sampling period. This model can represent the case when we know the value of the delay with some uncertainty (ex: $0.001s \pm 10\%$).

4 Controller without delay compensation

Using section 2.1.2, an RST controller has been designed to control the motor. This was done leaving aside the delays. In the following, this controller will be called the *ordinary controller*. Figure 4 shows the step response of the system without any delay in this case.

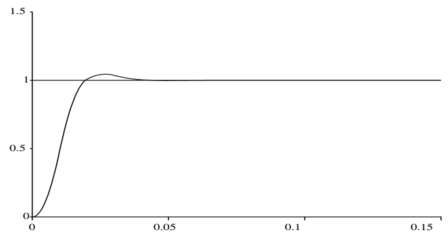
Figure 4 : Step response of the system without any delay

4.1 Case of a constant delay

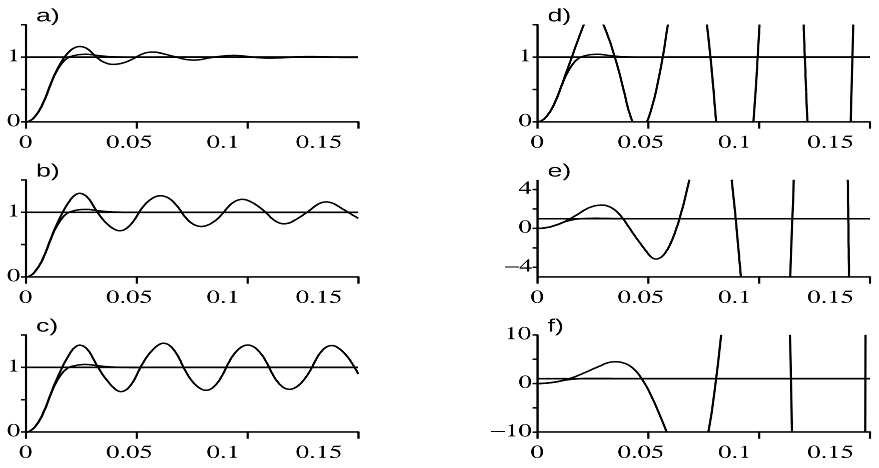
Figure 4.1 : Simulation of the ordinary controller
in the case of a constant delay δ
a) δ=0.05h, **b)** δ=0.1h, **c)** δ=0.12h, **d)** δ=0.25h, **e)** δ=0.5h, **f)** δ=h.

The results of the simulation of the ordinary controller in presence of constant delays are shown in Figure 4.1. They show that the system is stable for delays lower than 0.12h. This corresponds to the theoretical bounds to the stability domain computed via Matlab. Those are :

- Lower bound : 0 s,
- Upper bound : 0.0013 s.

The upper bound corresponds to 0.13h. In the rest of the report, we will consider delays as percentages of the sampling period. This an abuse of language because there is no constant ratio between the stability bounds and the sampling interval. Though, as we will work for a while with a constant sampling interval, it seems to be easier to interpret the delays in term of percentage of the sampling interval than in terms of absolute time.

4.2 Case of a stochastic delay

In the stochastic case, it is difficult to determine any analytical stability bound. However, the mean value of the delay can be used to get an idea of the system behaviour.

4.2.1 Model $\mathcal{M}_1$

A delay following model $\mathcal{M}_1$ has been generated with several mean values and introduced into the system (Figure 4.2.1). In each case, five realizations of the stochastic simulation are shown.

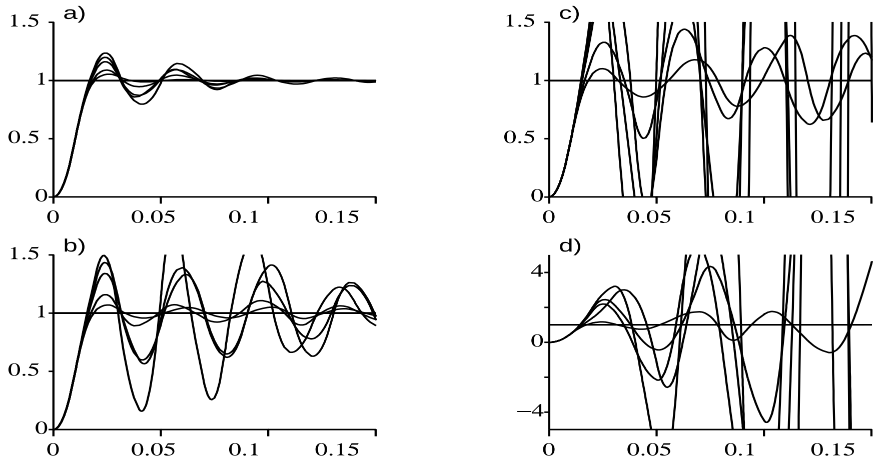

Figure 4.2.1 : Simulation of the ordinary controller
in the case of a stochastic delay following model $\mathcal{M}_1$.
a) $\bar{\delta}$=0.05h, **b)** $\bar{\delta}$=0.12h, **c)** $\bar{\delta}$=0.25h, **d)** $\bar{\delta}$=0.5h.

When the system is well inside the stability domain, the system has a behaviour close to the one it would have in presence of a constant delay equal to the mean value (case **a)**).

But if the system is to close to the stability bound, there can be some trouble (case **b)**). The system may loose its stability. Actually it does not loose its stability on a statistical point of view : after an infinite time, it will converge to the reference value. But we will consider it unstable because, meanwhile, it will have taken unacceptably high values; or because the statistical infinite time which can be finite from the physical point of vue is *really* too long.

On the opposite, the system can gain stability in the instable area (cases **c)** and **d)**). And in this case, the system is really stable. Indeed, if the delays remain long enough below the stability bound, the output signal will come close enough to the reference value to be unsensitive to the delays.

Conclusion : The mean value of the delay gives an idea of what the step response of the system would be. But one must be careful with it : the maximal reachable delay must not be too far over the upper stability bound.

4.2.2 Model $\mathcal{M}_2$

Model $\mathcal{M}_2$ contains the opportunity to change the maximal value of the delay without changing its mean value. This allows to see what happens when the mean value is fixed (Figure 4.2.2). Here also, five realizations of the stochastic simulation are shown.

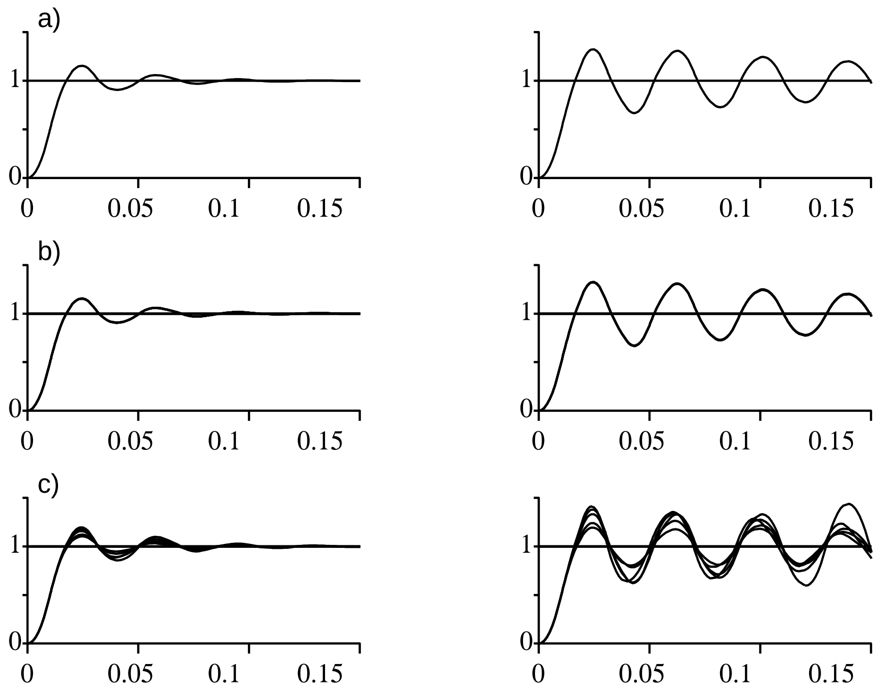

Figure 4.2.2 : Simulation of the ordinary controller
in the case of a stochastic delay following model $\mathcal{M}_2$.
when $\bar{\delta}$=0.05h (left) and $\bar{\delta}$=0.12h (right) in the following cases :
a) α=0, **b)** α=10%, **c)** α=50%.

As expected the mean value gives the overall behaviour of the system. The amplitude makes the signal oscillate more and more around the ground signal as the the parameter α grows (i.e. as the maximal value grows) and can even make the system unstable.
This can be related to the previous model. Indeed, model $\mathcal{M}_1$ is a special case of model $\mathcal{M}_2$. It is the case when the parameter α is maximal and therefore, the maximal value is twice the mean value. That explains that with the same mean value, model $\mathcal{M}_1$ is less stable than model $\mathcal{M}_2$.

Conclusion : To get an idea of the behaviour of the system, one must think both in terms of mean value and maximal value. The first one gives an idea of the overall behaviour and the value of the second compared to the stability bound determines the stability characteristics.

5 Controllers with compensation

5.1 Testing our design

In this section, the results from section 2.2.2 are used to design an RST controller which is able to cancel a constant delay. The controller is determined before every simulation and is kept constant over the simulation. The polynomials R, S, and T can be computed algebraically via Maple if the delay to be cancelled is less than the sampling period. The solution can then be translated from Maple to Simnon. If the delay is greater than one sampling period, the system to solve grows in order and it becomes too complicated with Maple. The best thing to do should then be to compute numerically the polynomials via Matlab and modify manually the control law in Simnon.

In Figure 5.1 can the results of several designs be found.

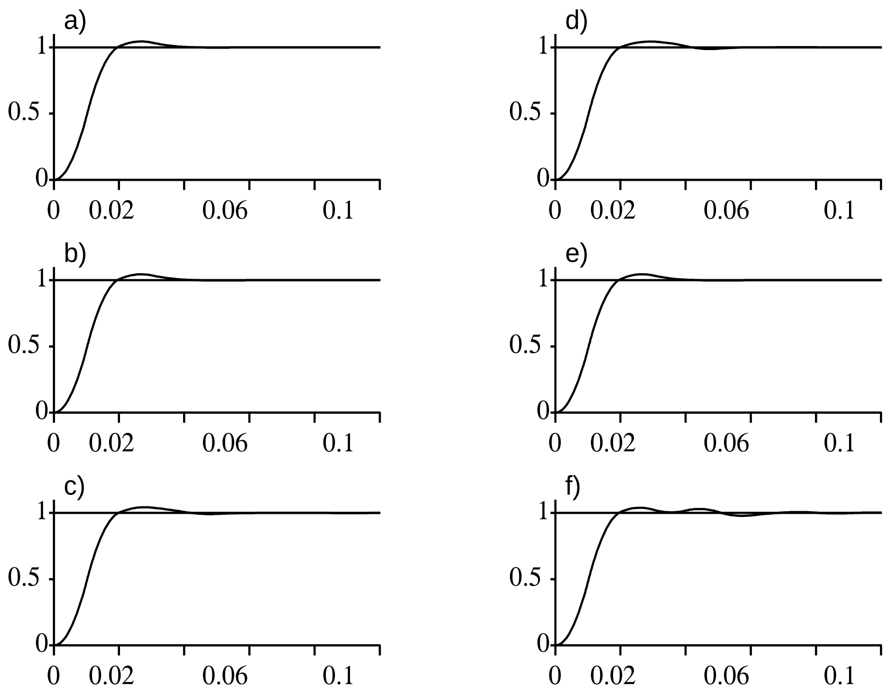

Figure 5.1 : Simulation of several modified controllers
in the case of an actual delay equal to the expected one.
a) δ=0.05h, **b)** δ=0.1h, **c)** δ=0.25h, **d)** δ=0.5h, **e)** δ=0.99h, **f)** δ=1.5h.

The five first cases use the algebraic solution obtained from Maple. The last one uses coefficients that have been computed via Matlab. As Matlab and Simnon do not have the same numerical precision, it explains why the step response does not look "nice".

The conclusion of this is that our design cancels pretty well the constant delay which it is supposed to cancel (sic).

5.2 Stability domain

The stability bounds of several controllers with compensation of a constant delay δ are given as percentages of the sampling interval in the next table :

δ	Lower Bound	Upper Bound	Width
$0.05h$	0	18	18
$0.1h$	4	24	20
$0.25h$	18	40	22
$0.5h$	31	66	35
$0.75h$	69	81	12
h	97	103	6
$1.2h$	116	125	9
$1.5h$	145	159	14
$1.7h$	166	175	9
$2h$	199	202	3
$5h$	500	501	1

Note: the precision of the computation is 1% of the sampling period.

The stability domain moves up along the real axes as δ grows. It is always more or less centered in δ. The width of the domain has a noticeable behaviour : it seems to follow a damped wave with local minima around the integer multiples of the sampling period and local maxima around the middle of each of the intervals between these multiples.
The explanation of the damping lies certainly in a degradation of the computation when the delays increase : the values of the coefficients grows, the signals reach higher values, etc.
The problem of the wave seems harder to explain. It may come from the existence of a zero located in the origin for delays equal to a multiple of the sampling period and only then.

Conclusion : We created controllers that cancel an expected constant delay, however long it can be (in principle). Yet, as the delay to cancel grows, the stability domain decreases.

6 Modified controller

In this section, we will use the controller which is designed for the expected delay $\tau = 0.1h$.
It will be called the *modified controller*. And it will be used to compare controllers with delay compensation to the ordinary one.

6.1 Case of a constant delay

The theoretical bounds to the stability domain (computed via Matlab) are :

- Lower bound: 4% of h,
- Upper bound: 24% of h .

Simnon simulations (Figure 6.1) verify these results.

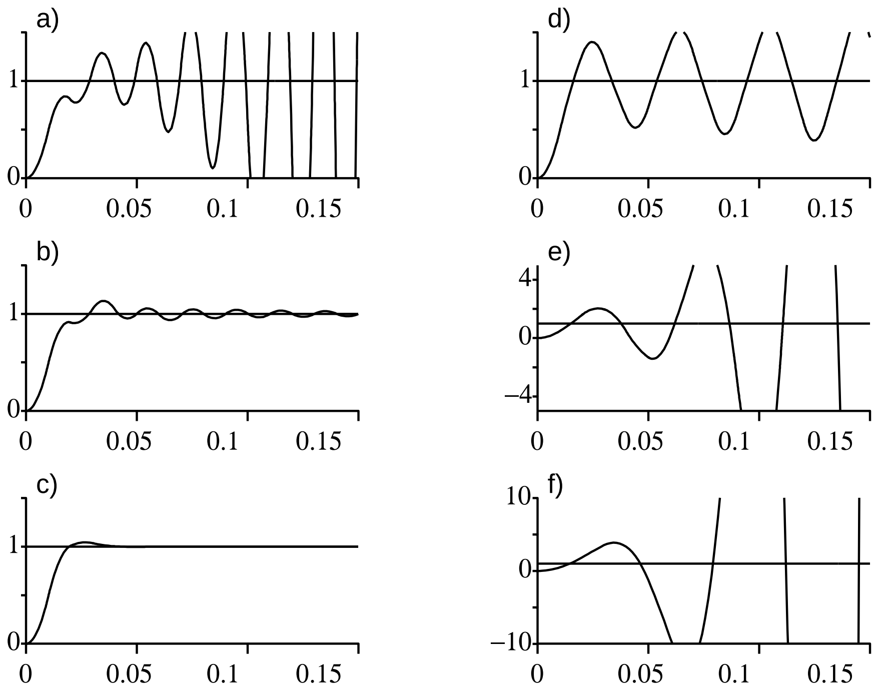

Figure 6.1 : Simulation of the modified controller for an expected delay equal to $0.1h$ in the case of a constant delay δ
a) δ=0h, **b)** δ=0.05h, **c)** δ=0.1h, **d)** δ=0.25h, **e)** δ=0.5h, **f)** δ=h.

6.2 Case of a stochastic delay

6.2.1 Model $\mathcal{M}_1$

Figure 6.2.1 shows a simulation of the system when the delays are modelled by $\mathcal{M}_1$.

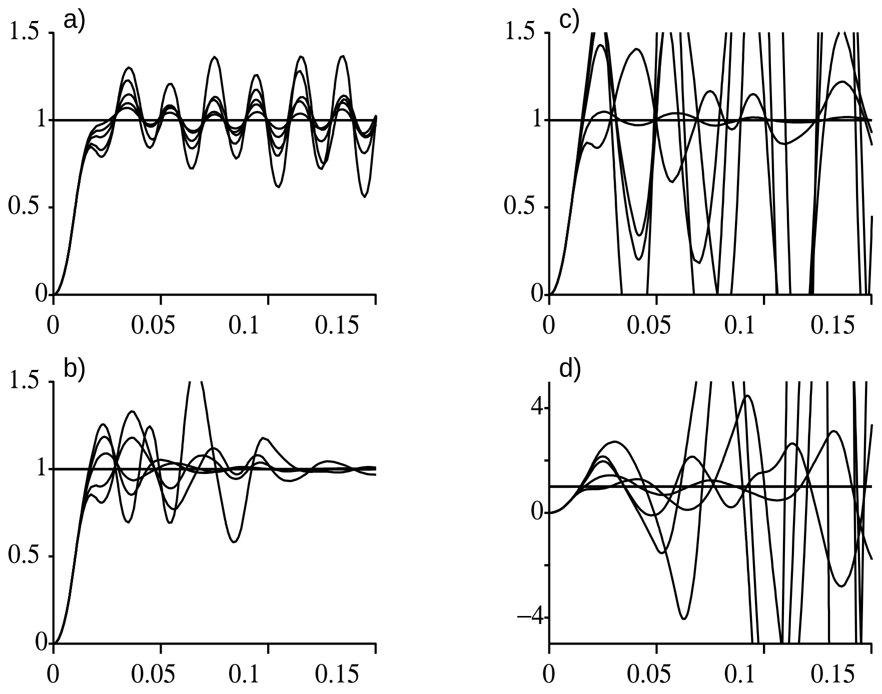

Figure 6.2.1 : Simulation of the modified controller for an expected delay equal to $0.1h$ in the case of a stochastic delay following model $\mathcal{M}_1$.
a) $\bar{\delta}$=$0.05h$, **b)** $\bar{\delta}$=$0.12h$, **c)** $\bar{\delta}$=$0.25h$, **d)** $\bar{\delta}$=$0.5h$.

The modified controller has a behaviour similar to the one of the ordinary controller. There are some differences though.

The first one concerns the area where the controller best compensates the delays : the modified controller is efficient on a larger range of delays. However, delays with a low mean value but still inside the stability range of a fixed delay can create instability (similarly to delays with a mean value close to the upper bound). Both of those two phenomena depends on the fact that the stability domain in the constant case has moved : it has enlarged and does not cover any more the area close to the origin.

6.2.2 Model $\mathcal{M}_2$

If the delays follow model $\mathcal{M}_2$, then the system has the behaviour shown in Figure 6.2.2.

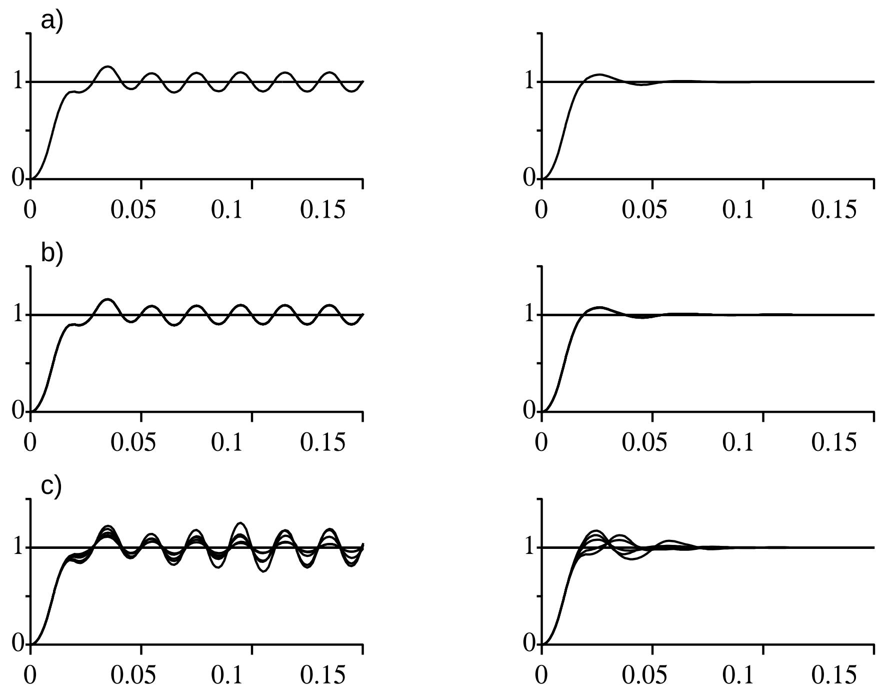

Figure 6.2.2 : Simulation of the modified controller for an expected delay equal to $0.1h$ in the case of a stochastic delay following model $\mathcal{M}_2$. when $\bar{\delta}$=$0.05h$ (left) and $\bar{\delta}$=$0.12h$ (right) in the following cases : **a)** α=0, **b)** α=10%, **c)** α=50%.

Here again, the modified controller behaves as the ordinary one. But if we compare the response of the ordinary controller when $\bar{\delta} = 0.05h$ and the response of the modified controller when $\bar{\delta} = 0.12h$ (the mean values are both larger but still close to the value used in the design), the modified controller seems to be a bit more sensitive to the amplitude around the mean value.

Conclusion : A modified controller compensates in an acceptable manner for delays that are close enough to the expected delay. But a trade-off between larger stability domain and increased risk of unstable low delays has to be found. A way to avoid the trade-off should be to try to decide if the delay at time t is under the lower stability bound and then use an ordinary controller or even better to use a controller that will compensate for this precise delay. This leads us to the idea of a controller that will adapt itself to the current delay.

7 Controller based on gain scheduling

In this section, the modified controller is left aside. Instead a controller based on gain scheduling will be studied. It uses a Simnon translation of the formulas we obtained from Maple. It is therefore limited to the cancellation of delays less than one sampling period.

To test this controller, stochastic delays following model $\mathcal{M}_2$ have been used. Figure 7 shows the simulations that have been done.

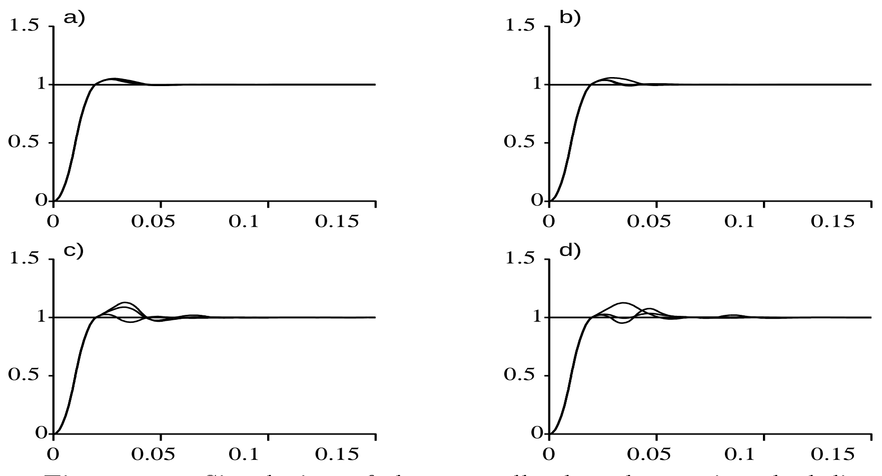

Figure 7.a : Simulation of the controller based on gain scheduling
In each case, 3 simulations of a stochastic delay
following model $\mathcal{M}_2$ with $\alpha = 0.1$
a) $\bar{\delta} = 0.05h$ **b)** $\bar{\delta} = 0.1h$ **c)** $\bar{\delta} = 0.25h$ **d)** $\bar{\delta} = 0.5h$

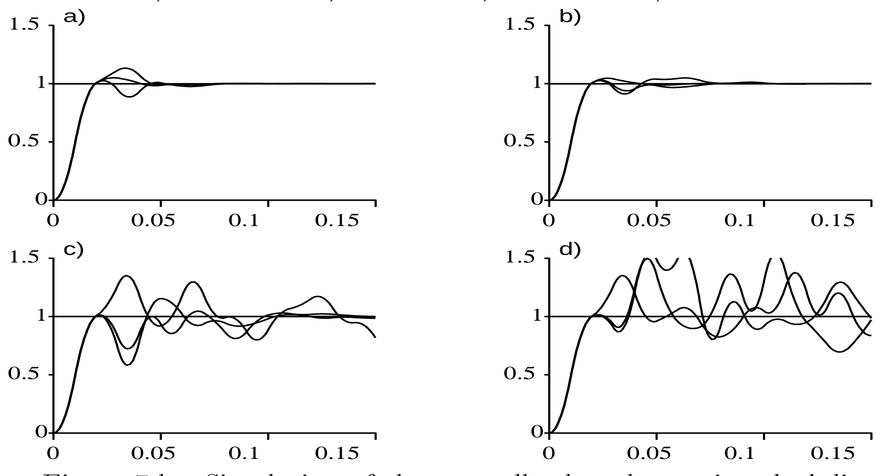

Figure 7.b : Simulation of the controller based on gain scheduling
In each case, 3 simulations of a stochastic delay
following model $\mathcal{M}_2$ with $\alpha = 0.5$
a) $\bar{\delta} = 0.05h$ **b)** $\bar{\delta} = 0.1h$ **c)** $\bar{\delta} = 0.25h$ **d)** $\bar{\delta} = 0.5h$

The behaviour is not exactly what was expected. It was supposed to be perfect. In fact, it cannot work so well because, although it is the same system order, as long as the delay is less than a sampling period, the stored values do not correspond to the values that the control law

is supposed to use. Indeed, when the control law is determined at a certain time kh, according to delay $\delta(kh)$, the system is supposed as having run before this time kh with this delay $\delta(kh)$; and, therefore, as having gone through certain states. However, as the delays at previous times were different, the states the system went through are also different.
Though, for short delays, the step response is better than the one of the ordinary controller or the modified controller (see Figure 4.2.2 and Figure 6.2.2).

When the delay is short, the system remains stable even if it has a rather bad step response, but if the delay grows then it can become unstable (see Figure 7.b, case **d**)). Actually, the simulation should last longer to see if it is really unstable, but the behaviour (and my knowledge on this subject) is bad enough to make us abandon the idea of the controller based on gain scheduling for large delays.

Conclusion : The controller based on gain scheduling may be a solution in case of short delays (less than 20% of the sampling period in our case), as long as we accept strange step responses.

8 Robustness

8.1 Robustness for the ordinary controller

The robustness properties of the ordinary controller will be discussed in this section. We will therefore vary the observer poles during the design. Actually, it will not result in the same controller as the one designed in section 4, but as the design is also done leaving aside any delay, we will consider the controllers of this section as one unique ordinary controller with variable poles.

8.1.1 Stability domain and observer poles

For an easier discussion, and even though this has no link to a physical representation, we chose to have our observer poles real and double. The observer polynomial can thus be written as $A_o(q) = (q - a_o)^2$. Figure 8.1.1 shows the stability domain as function of the observer pole a_o.

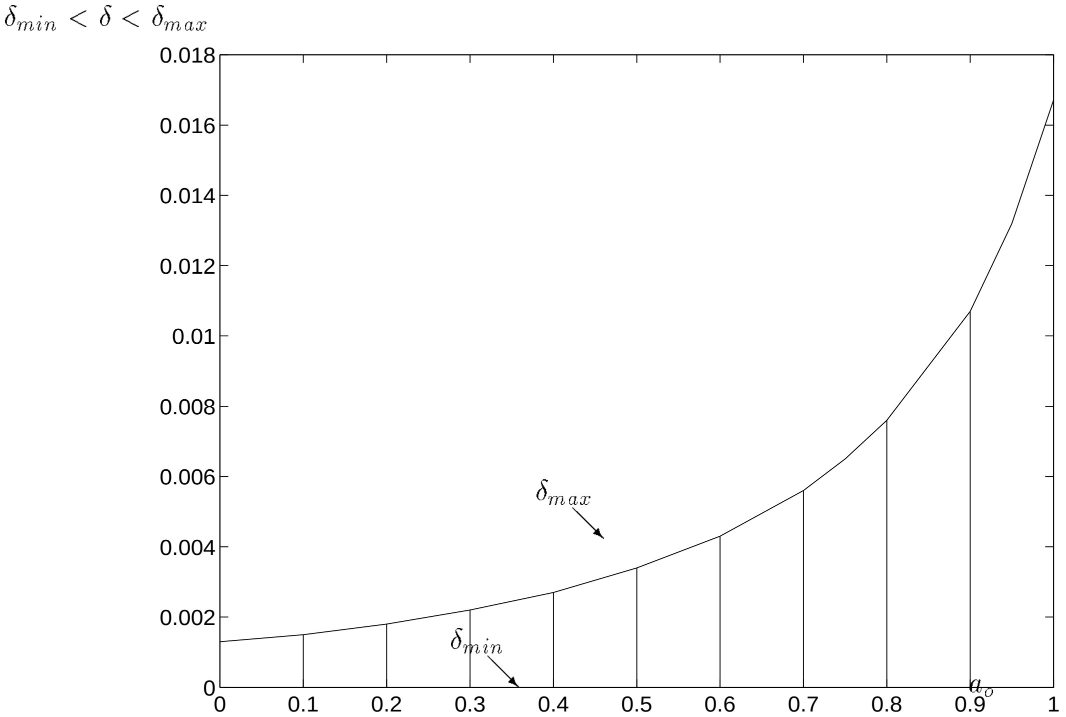

Figure 8.1.1 : Stability domain of the system
as function of the double real observer pole
Note: we diplayed the bounds in seconds

The stability domain is increasing with the value of the observer poles. It is therefore attractive from a stability point of view to place the observer poles as close as possible to 1.

8.1.2 Observer poles and load disturbances

The problem when placing the observer poles, is that as a_o increases, the observer becomes slower. It means that the load disturbances will not be detected as fast as desired. Figure 8.1.2 shows this phenomenon.

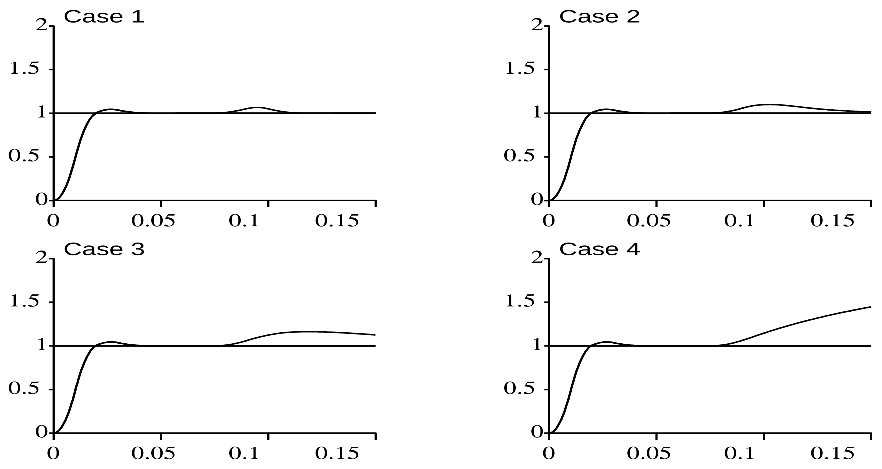

Figure 8.1.2 : Simulation of the ordinary controller with load disturbances occuring at t=0.075 and for different observer pole placements. a_o is equal to : **a)** 0, **b)** 0.5, **c)** 0.75, **d)** 0.95.

Conclusion : The observer poles should be taken lower than 0.75 if we want the integrator contained by the controller to be of some use.

8.1.3 Observer poles and step response

Figure 8.1.3 shows the step response of the system with the ordinary controller for some observer pole placements.

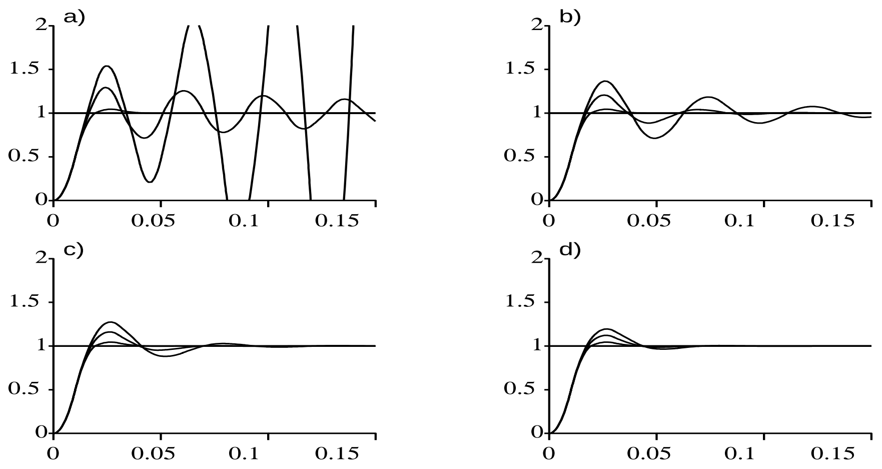

Figure 8.1.3 : Step response of the ordinary controller with different observers. a_o is equal to : **a)** 0, **b)** 0.5, **c)** 0.75, **d)** 0.95. 3 simulations have been done : $\delta = 0h, 0.1h, 0.2h$.

Here are the stability domains in each of these cases :

Pole	Stability Range (in % of h)
0	0–13
0.5	0–34
0.75	0–65
0.95	0–132

The system has not only a larger stability domain for slower observers but also an improved step response. Of course, the slowest observer gives, as expected, the smallest sensitivity to delays; but, according to previous section, it should not be used unless we are sure to be in a disturbanceless case.

Conclusion : The observer poles influence the stability of the system but also its ability to cancel load disturbances. As these two objectives are contradictory, a trade-off has to be found. A reasonable choice would be in our case to choose the poles between 0.5 and 0.75.

8.2 Robustness for the modified controller

Consider again the controller designed to compensate for a constant delay $\tau = 0.1h$.

8.2.1 Stability domain

As for the ordinary controller, we chose to have a double real observer pole a_o. Figure 8.2.1 shows the stability bounds as functions of the observer pole.

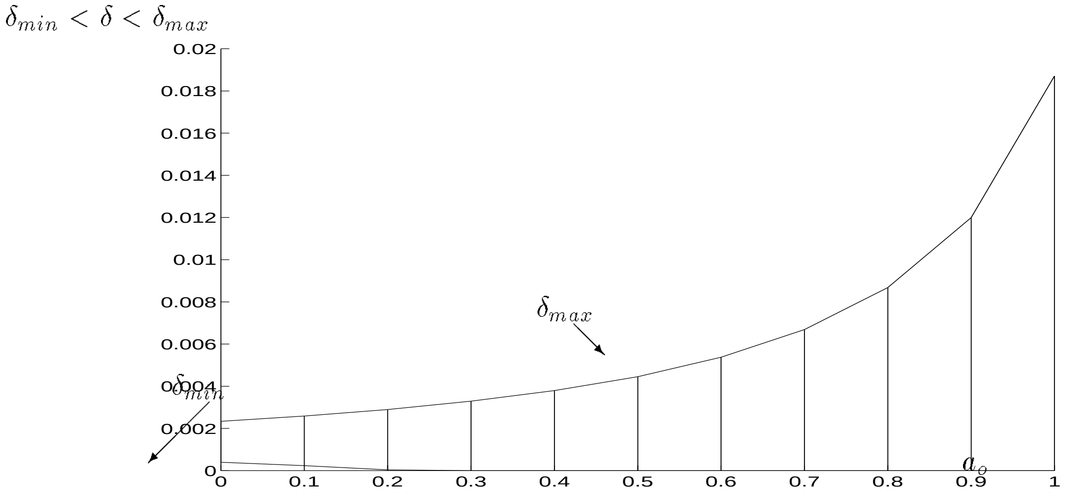

Figure 8.2.1 : Stability domain of the system as function of the double real observer pole

Note: we diplayed the bounds in seconds

Moving the poles eliminates the unstable interval around 0 we had with the deadbeat solution. This will be of a big help : the delays are now allowed to be over-estimated if the observer poles are placed far enough from the origin.

The poles should be given here a higher value than 0.25. It is noticeable that this is not a too restrictive condition in term of load disturbance cancellation.

8.2.2 Step response

A quick look on the step response (Figure 8.2.2) confirms the improved response of the system when the observer becomes slower and the elimination of the unstable short delays.

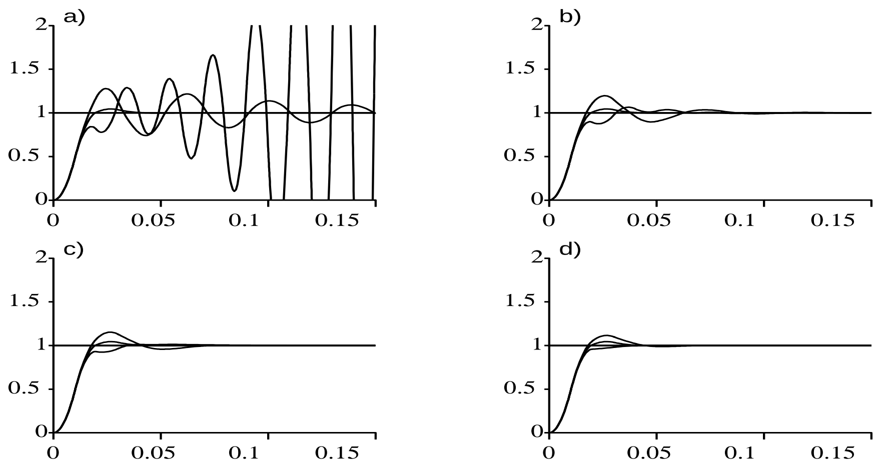

Figure 8.2.2 : Step response of the modified controller with different observers. The poles are located in: **a)** 0, **b)** 0.5, **c)** 0.75, **d)** 0.95. 3 simulations have been done : $\delta = 0h, 0.1h, 0.2h$.

Here are the stability domains in each of these cases :

Pole	Stability Range (in % of h)
0	4–24
0.5	0–45
0.75	0–76
0.95	0–148

Conclusion : The main thing to be noticed is that the modified controller lost its drawback : it can now be considered as much better than the ordinary one since the stability range is larger. But sensitivity to delays and load disturbance cancellation still have to be balanced.

8.3 Robustness and delay compensation

If the delay is known well enough (i.e. its mean value is known), then the following requirements can be made. The controller has to cancel a constant delay τ as close as possible to the mean value $\bar{\delta}$ of the real delay, to have its observer pole a_o as close as possible to the origin and to have the largest achievable stability range $[\delta_{min}(\tau, a_o), \delta_{max}(\tau, a_o)]$. This problem can be considered as a minimization problem of the form :

$$\min_{\tau, a_o} J(\tau, a_o) = Q_1(\tau - \bar{\delta})^2 + Q_2 a_o^2 - Q_3 \left[\delta_{max}(\tau, a_o) - \delta_{min}(\tau, a_o)\right]^2 \tag{27}$$

where Q_1, Q_2 and Q_3 are non negative weights.

On the opposite, if the delay has an unknown mean value, two ways can be followed. The first one is to estimate the mean value and then to use the previous paragraph. The other one is to determine a controller which will compensate for a delay as long as possible while keeping a good protection against the load disturbances.

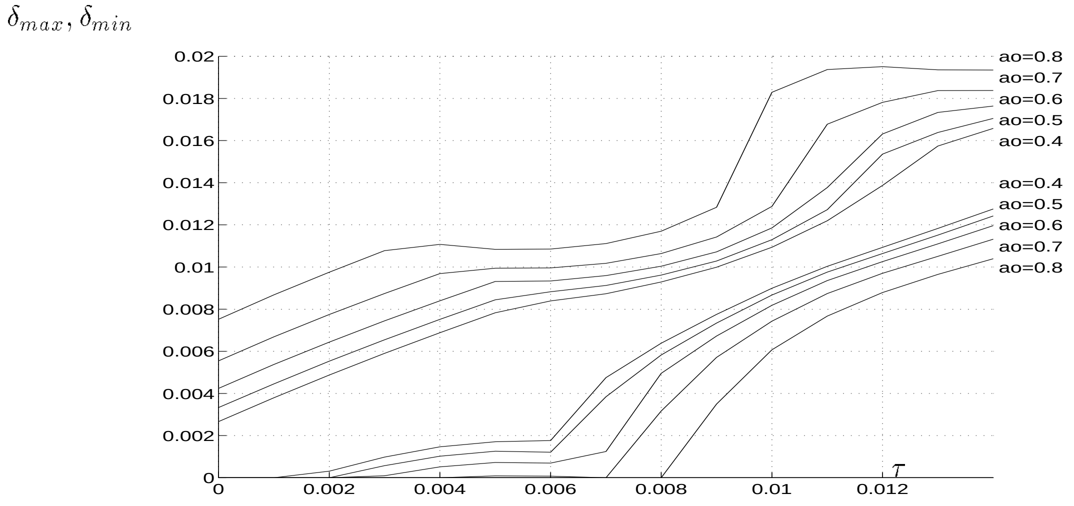

Figure 8.3.a : Stability bounds as function of the delay used in the design for several observers a_o.

Without any calculation and only refering to Figure 8.3.a, it is tempting to choose a controller designed to cancel a delay close to 0.007s (70% of the present sampling period) with both the observer poles around 0.7 since it compensates for all the delays that are shorter than the sampling period. However, this can only be done if the only requirement is to have a large stability domain. And if we do not need to have a good step response for this controller does not clear the step very well (Figure 8.3.b).

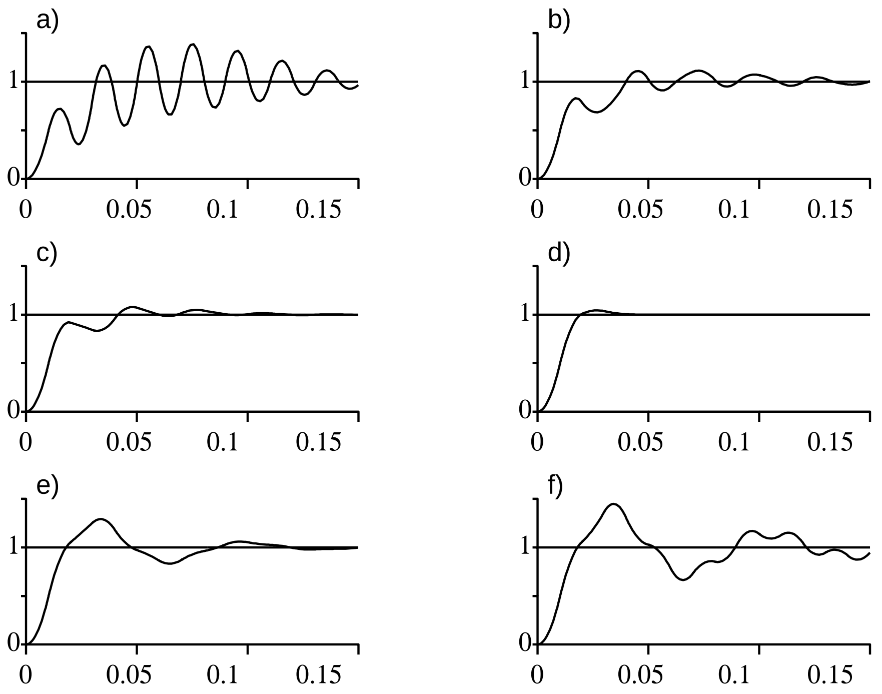

Figure 8.3.b : Step response of the controller with :
$\tau = 0.007$s, $a_o = 0.7$ and a constant delay δ equal to:
a) 0, b) $0.3h$, c) $0.5h$, d) $0{,}7h$, e) $0.9h$ and f) h.

Conclusion : We have to add a constraint to the problem above, which is to get a *nice* step response (this can be quantified by an adequate norm). And that can become rather difficult and perhaps not satisfying enough.

9 Changing the sampling period

As delays are not dependent of the sampling period, this one may have some influence on the sensitivity of the system to delays. This will be discussed in this section.

9.1 Stability domain

The sampling period was initially chosen as $h = 0.01s$. The wiseness of this choice can be decided by computing the stability bounds for the uncompensated system when the sampling period belongs to an interval $[h/10, 10.h]$ (Figure 9.1). The stability bound is defined as the largest delay for which the closed loop system is stable when the controller is designed for a delay $\tau = 0$ and when the sampling interval is h. The observer pole is $a_o = 0$.

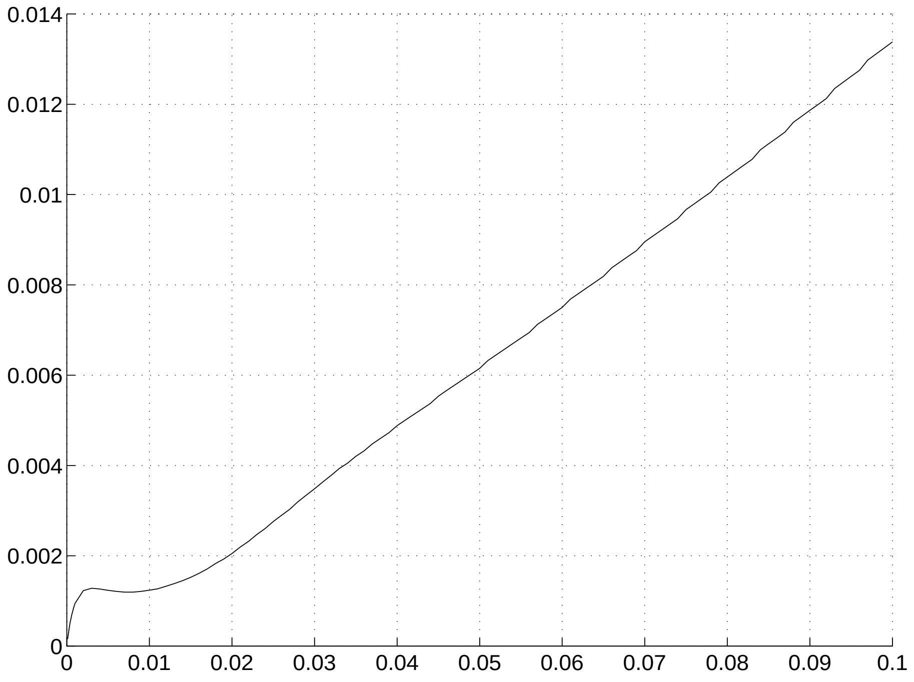

Figure 9.1 : Stability bounds of the uncompensated system in function of the sampling period.

Note : The axes are scaled in seconds.

It appears that the stability range (in terms of delays) grows with the sampling interval. Thus, our choice may have been unaccurate from the design point of view but it is quite good from the educative one.

9.2 Step response

Anyhow, the size of the stability range is not the only thing we should use to choose our sampling interval. Otherwise, we would choose an infinite period which would be completely unsensitive to delays but uncontrollable !

We must therefore consider the step response of the system (Figure 9.2).

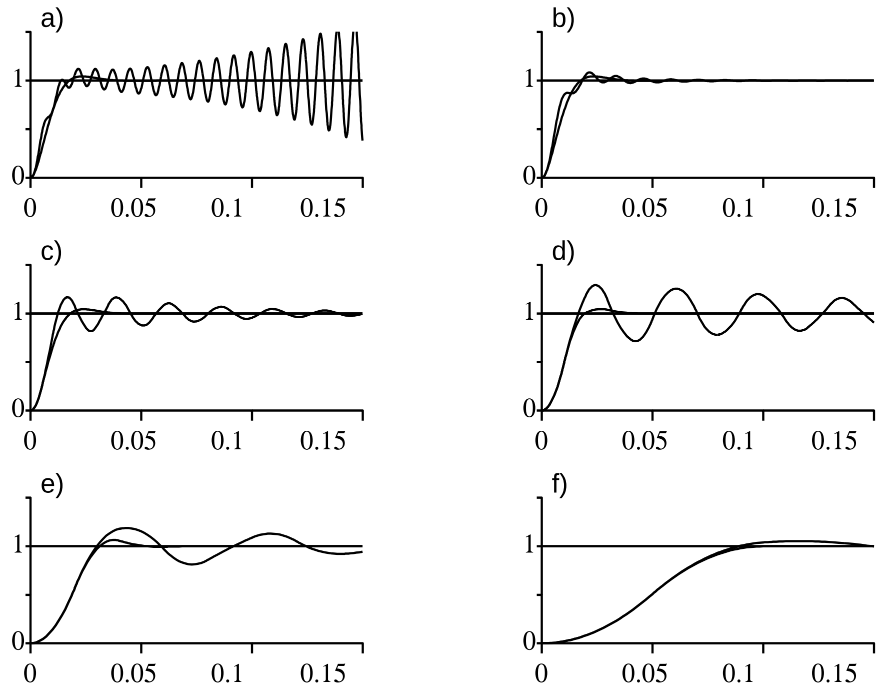

Figure 9.2 : Step response of the uncompensated system with and without a delay δ=0.001s when sampled at :
a) 0.001s, **b)** 0.002s, **c)** 0.005s, **d)** 0.01s, **e)** 0.02s, **f)** 0.05s.

These simulations confirm that a too large sampling interval is to avoid because then the closed loop is different from the one we desire (cases **e)** and **f)**). On the opposite, a too short sampling period makes the system too sensitive to delays (case **a)**. Moreover, the existence of a local maximum in the previous graph (Figure 9.1) appears also here : compare the first four cases.

Conclusion : A good choice of the sampling period when taking into account delay compensation and desired closed loop behaviour would be here 0.002 which corresponds to the usual rule of thumb $\omega h \approx 0.2 - 0.5$.

10 Conclusion

In section 2, the equations to solve in order to design a discrete-time RST controller have been derived. The stability to delays of the uncompensated controller has been studied in section 4. By designing a controller which cancels a constant delay, we have increased the stability domain (sections 5 and 6). Meanwhile this domain has been shifted and short delays have become unstable. In order to try to avoid this, a gain scheduling controller has been designed (section 7). As this was not satisfying, the robustness of the system to delays has been increased by moving the observer poles. That lead to a trade-off between robustness to delays and disturbance cancellation (section 8). Finally, studying the influence of the sampling period on the robustness to delays, we found out that the usual rule of thumb for choosing the sampling interval gives the best results.

Even if the design for compensation has been done for any delay, the results in this report concern mainly delays that are lower than the sampling period. If they become larger, then the problem of the *missing sample* can appear. Solving it should be a continuation to this work.

11 References

Åström,K.J and B.Wittenmark (1990): Computer Controlled Systems : Theory and design, Prentice Hall, Englewood Cliffs, N.J.