Titre: Title:	Développement d'un module de réglage automatique du régulateur PID Dual Loop
Auteurs: Authors:	Ernesto Cornieles, & Christophe Bougeret
Date:	1997
Type:	Rapport / Report
Référence: Citation:	Cornieles, E., & Bougeret, C. (1997). Développement d'un module de réglage automatique du régulateur PID Dual Loop. (Rapport technique n° EPM-RT-97-24). https://publications.polymtl.ca/9625/

Institution:	École Polytechnique de Montréal
Numéro de rapport: Report number:	EPM-RT-97-24
URL officiel: Official URL:	
Mention légale: Legal notice:	

Département de Génie Électrique
et de Génie Informatique

Section Automatique

DÉVELOPPEMENT D'UN MODULE DE RÉGLAGE AUTOMATIQUE DU RÉGULATEUR PID DUAL LOOP

Par

Ernesto Cornieles
Christophe Bougeret

projet dirigé par

Professeur Romano M. DeSantis
Professeur Jules O'Shea

École polytechnique de Montréal
Août 1997

Dépôt légal, Août 1997
Bibliothèque nationale du Québec
Bibliothèque nationale du Canada

Développement d'un module de réglage automatique
du régulateur PID Dual Loop

Ernesto Cornieles
Christophe Bougeret
Projet dirigé par Professeurs Romano DeSantis et
Jules O'Shea
Département de Génie Électrique et
Génie Informatique, Section Automatique

Pour se procurer une copie de ce document, s'adresser au:

Service des Éditions
École Polytechnique de Montréal
Case Postale 6079, Succursale Centre-Ville
Montréal (Québec) H3C 3A7
Téléphone: (514) 340-4711 ext. 4473
Télécopie: (514) 340-3734

Compter 0,10$ par page et ajouter 3,00$ pour la couverture, les frais de poste et la manutention. Régler en dollars canadiens par chèque ou mandat-poste au nom de l'École Polytechnique de Montréal.

Nous n'honorerons que les commandes accompagnées d'un paiement, sauf s'il y a eu entente préalable dans le cas d'établissements d'enseignement, de sociétés ou d'organismes canadiens.

DÉVELOPPEMENT D'UN MODULE DE RÉGLAGE AUTOMATIQUE DU RÉGULATEUR PID DUAL LOOP

Ernesto Cornieles, Christophe Bougeret

Département de génie électrique et de génie informatique
Section Automatique
Ecole Polytechnique de Montréal

SOMMAIRE

Ce rapport concerne la conception et le développement d'un module d'adaptation, permettant d'automatiser la procédure de réglage des gains des régulateurs PID Dual Loop. Le système asservi est constitué par un réservoir d'eau dont il faut régler le niveau et la température. L'objectif consistait à approfondir les connaissances que nous avons du PID Dual Loop, notamment de vérifier son intérêt dans le cas d'un système dont la fonction de transfert n'est pas connue. Les résultats obtenus sont satisfaisants et le module de réglage a permis de trouver un jeu de gains acceptable dans tous les essais effectués.

REMERCIEMENTS

Nous souhaitons remercier les professeurs Romano DeSantis et Jules O'Shea pour leur accueil et pour l'aide qu'ils nous ont apportée dans la réalisation de ce projet.

Un grand merci également aux professeurs Jean-Luc Jeanneau et Wisama Khalil qui ont favorisé cette coopération entre les laboratoires d'automatique de l'Ecole Centrale de Nantes et de l'Ecole Polytechnique de Montréal.

Nous aimerions aussi exprimer notre reconnaissance à l'ensemble du personnel de l'École Polytechnique de Montréal, et en particulier aux techniciens de la section Automatique pour leur aide précieuse.

TABLE DES MATIERES

LISTE DES SYMBOLES

α	= retard (en secondes)
α_1, α_2	= coefficients du dénominateur de la fonction de transfert après une itération
α_{10}, α_{20}	= coefficients du dénominateur de la fonction de transfert avant une itération
α_r	= rapport des commandes en tension pour le réglage des debits d'eau froide et d'eau chaude (sans unité)
β_1, β_2	= coefficients du prédécoupleur pour la boucle de température
$\Gamma(s)$	= fonction de transformation de la robustesse
μ	= coefficient de multiplication de la pulsation naturelle
μ_k	= coefficient de multiplication de l'amortissement
ξ	= amortissement après une itération
ξ_0	= amortissement avant une itération
ω_n	= pulsation naturelle après une itération
ω_{n0}	= pulsation naturelle avant une itération
a1, a2	= coefficients du dénominateur de l'approximation du second ordre
A1, A2	= noms génériques des gains de la boucle externe du régulateur PID Dual Loop
A11, A12	= gains de la boucle externe du régulateur de niveau
A21, A22	= gains de la boucle externe du régulateur de température
ampl	= amplitude de premier dépassement désirée à l'issue de la première étape de réglage
F_1	= fonction de transfert consigne/mesure après une itération
F_2	= fonction de transfert perturbation/mesure après une itération

F_{10}	= fonction de transfert consigne/mesure avant une itération
F_{20}	= fonction de transfert perturbation/mesure avant une itération
G_1, G_2	= fonctions de transfert particulières
k_1, k_2	= gains proportionnel et dérivatif après une itération
k_{10}, k_{20}	= gains proportionnel et dérivatif avant une itération
K1, K2, K3, K4	= noms génériques des gains de la boucle interne du régulateur PID Dual Loop
K11, K12, K13, K14	= gains de la boucle interne du régulateur de niveau
K21, K22, K23, K24	= gains de la boucle interne du régulateur de température
overshoot	= amplitude réelle de premier dépassement
pv1, pv2	= signaux d'évaluation du niveau et de la température fournis par le système au régulateur (en V)
P_S	= consigne
P_V	= mesure
P_t	= perturbation ajoutée volontairement sur la commande pour tester la robustesse
ref1, ref2	= consignes de niveau et de température (en V)
tfpa	= temps réel de premier dépassement de la réponse
tpd	= temps de premier dépassement désiré à l'issue de la première étape de réglage
U1, U2	= commandes en sortie de régulateur (en V)
u1, u2	= signaux de commandes des vannes d'eau chaude et d'eau froide fournis par le régulateur du système (en V)
u11, u12	= commandes partielles des vannes d'eau chaude et d'eau froide issues du régulateur de niveau (en V)
u21, u22	= commandes partielles des vannes d'eau chaude et d'eau froide issues du régulateur de température (en V)

LISTE DES FIGURES

LISTE DES TABLEAUX

INTRODUCTION

L'objectif de ce rapport est double:
- il s'agit d'abord de réaliser un logiciel permettant le réglage automatique (auto-tuning) des gains des régulateurs PID Dual Loop contrôlant le niveau et la température dans le réservoir d'eau présenté à la référence [Cornieles 2]. Ce logiciel permettra de régler les gains de l'un ou l'autre des régulateurs (contrôle du niveau ou de la température) ou bien des deux régulateurs (contrôle du niveau et de la température).
- les résultats expérimentaux obtenus permettront ensuite de dire dans quelle mesure les propriétés intéressantes du Dual Loop (possibilité de régler indépendamment la dynamique et la robustesse du système dans le cas d'un système du premier ordre ou du second ordre avec pôle nul [DeSantis 1]) sont conservées dans le cas d'un système de dynamique inconnue présentant un retard, des saturations et perturbations externes du banc d'essais (variation de la température d'eau d'alimentation du réservoir, de la pression d'air, etc). Les expériences ont été faites au printemps.

1.-Rappel des structures du banc d'essai et du régulateur PID Dual Loop

1.1.-Banc d'essai

Les commandes sont les tensions U1 et U2 qui permettent par l'intermédiaire d'un prédécoupleur de varier les ouvertures des vannes pneumatiques d'eau chaude et d'eau froide arrivant dans le réservoir. Les mesures sont les tensions pv1 et pv2 fournies par les capteurs et correspondent au niveau et à la température de l'eau sortant du réservoir. En temps normal (sans le module de calcul automatique des gains), le banc d'essai a la structure décrite à la figure 1:

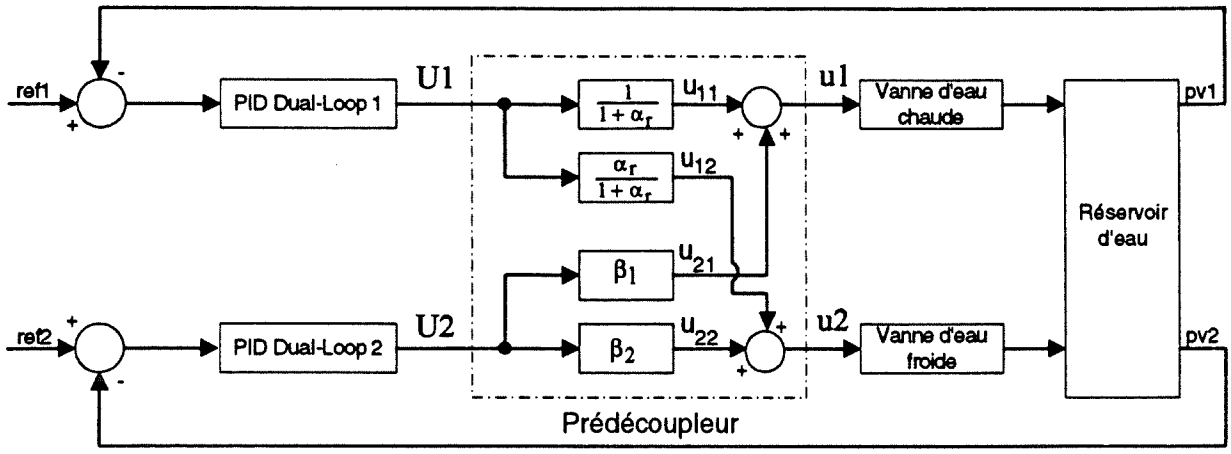


Figure 1: Structure générale de l'asservissement

1.2.- PID Dual Loop

Les régulateurs sont réalisés de façon numérique sur un PC. Ce sont des PID Dual Loop [DeSantis 1] (cf figure 2) dont la loi de commande est de la forme:

$$U(s) = K_1(P_S(s) - P_V(s)) - sK_2P_V(s) + K_4P_S(s) + K_3\left\{\frac{P_S(s) - P_V(s)}{s} - A_1P_V(s) - A_2sP_V(s)\right\}$$

A_1, A_2, K_1, K_2, K_3, K_4 sont des gains réglables mais pour toutes les expériences présentées dans ce rapport $K_4 = 0$.

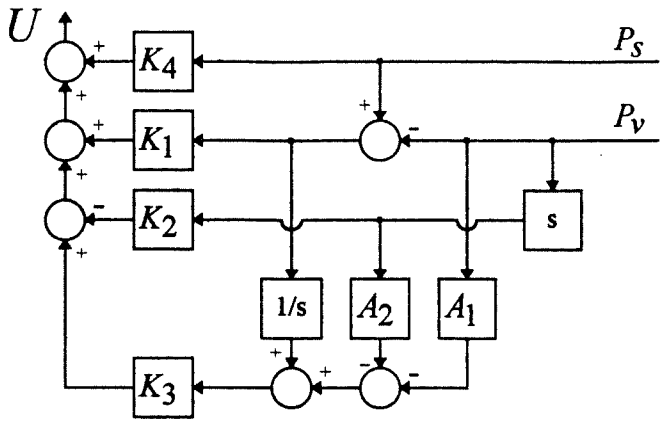


Figure 2: Structure du PID Dual Loop

1.3.-Deux propriétés caractéristiques du PID Dual Loop

D'après [DeSantis 1], l'intérêt du Dual Loop réside dans la possibilité de régler indépendemment la dynamique et la robustesse du système asservi pour des fonctions G du type

$$G_1(s) = \frac{K_M}{1+s\tau_p} \qquad \text{ou du type} \qquad G_2(s) = \frac{K_M}{(1+s\tau_p)s}$$

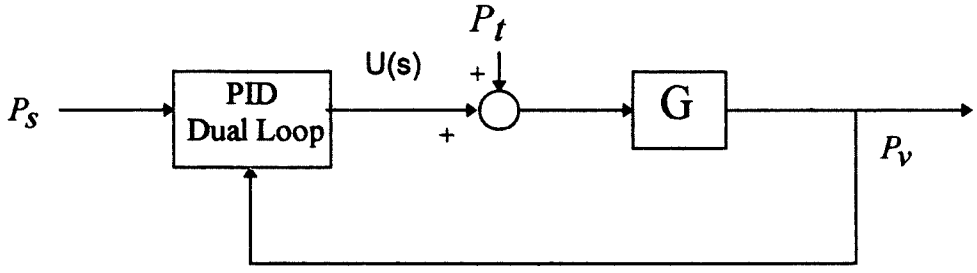


Figure 3: Schéma-bloc d'un asservissement

Ainsi, si P_t est une perturbation ajoutée sur la commande et si $F_{10}(s) = \frac{P_v(s)}{P_s(s)}$ et $F_{20}(s) = \frac{P_v(s)}{P_t(s)}$ sont les fonctions de transfert lorsqu'on n'utilise pas la boucle externe (A_1, A_2 et K_3 nuls) alors, en conservant les mêmes gains K_1,K_2 et K_4 et en ajoutant les gains A_1, A_2 et K_3 adéquats, on obtient:

- $F_1(s) = F_{10}(s)$ (fonction de transfert entrée/sortie conservée) (1)

$$\bullet\ F_2(s) = \Gamma(s)F_{20}(s) \text{ avec } \begin{cases} \Gamma(s) = \dfrac{s}{s + \dfrac{K_3 K_M}{1 + K_1 K_M}} & \text{pour } G_1(s) \\ \text{et} & \\ \Gamma(s) = \dfrac{s}{s + \dfrac{K_3}{K_1}} & \text{pour } G_2(s) \end{cases} \qquad (2)$$

donc en théorie il faut augmenter K_3 le plus possible jusqu'à obtenir la robustesse souhaitée face à la perturbation P_l.
Le réglage qui va être réalisé sur les régulateurs de niveau et de température sera fait selon la méthode préconisée par [DeSantis 1] pour les fonctions de transfert G_1 et G_2 évoquées plus haut:
- réglage deK_1 et de K_2 de façon à obtenir une dynamique satisfaisante
- introduction des gains A_1, A_2 et augmentation de K_3 jusqu'à obtenir la robustesse souhaitée.
Ensuite il sera possible de vérifier si la dynamique obtenue avec les seuls gains K_1 et K_2 est maintenue lorsqu'on introduit A_1, A_2 et K_3 (cf équation (1)) et dans quelle mesure on peut augmenter K_3 sans détériorer la réponse.

2.-Procédure de réglage automatique des gains

Un module d'adaptation a été ajouté à la structure présentée à la figure 1. La stratégie de réglage utilisée par ce module est décrite dans les paragraphes suivants. De plus, un bloc de retard a été ajouté pour simuler les retards rencontrés dans les installations industrielles où les tuyaux sont beaucoup plus longs.

2.1.-Cas monovariable

Chacune des boucles d'asservissement (niveau ou température) peut maintenant être vue de la façon suivante:

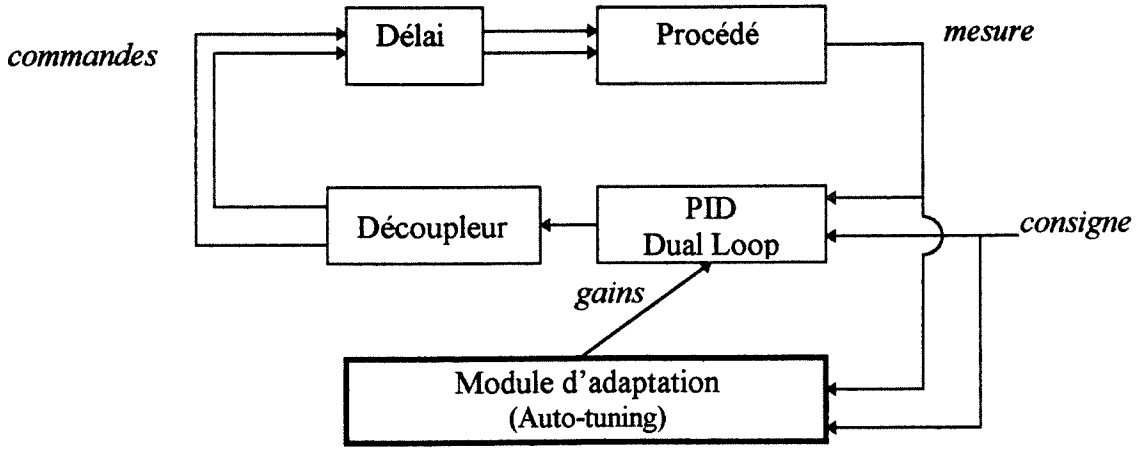


Figure 4: Schéma bloc de l'auto-tuning (monovariable)

Dans ce paragraphe, K_1, K_2, K_3, A_1 et A_2 correspondent respectivement à K11, K12, K13, A11, A12 et K21, K22, K23, A21, A22 selon que l'on considère la boucle de niveau ou la boucle de température.

Le réglage automatique se compose de deux étapes détaillées dans les deux paragraphes suivants. Quand la procédure de réglage est terminée, le programme affiche les gains suggérés du régulateur.

2.1.1.-Première étape du réglage

La première étape consiste à trouver un réglage satisfaisant pour K_1 et K_2. Pendant cette étape les seuls gains non nuls sont K_1 et K_2 (appelés k_{10} et k_{20} avant une itération et k_1 et k_2 après cette itération).
Au début du programme l'utilisateur indique l'amplitude *ampl* et le temps *tpd* maximaux de premier dépassement qu'il aimerait voir respectés à l'issue de la première étape, c'est-à-dire lorsqu'on règle le régulateur avec les deux gains K_1 et K_2 trouvés (A_1, A_2 et K_3 nuls).
La première étape s'arrête lorsque ces critères sont satisfaits ou lorsque le programme détecte des oscillations excessives dans la réponse.
Pendant cette étape, le programme procède à plusieurs essais en modifiant à chaque fois les réglages de K_1 et K_2 selon la stratégie décrite ci-dessous.

Quand K_3=0, K_4=0, A_1=0, A_2=0, la figure 2 est équivalente à la figure 5:

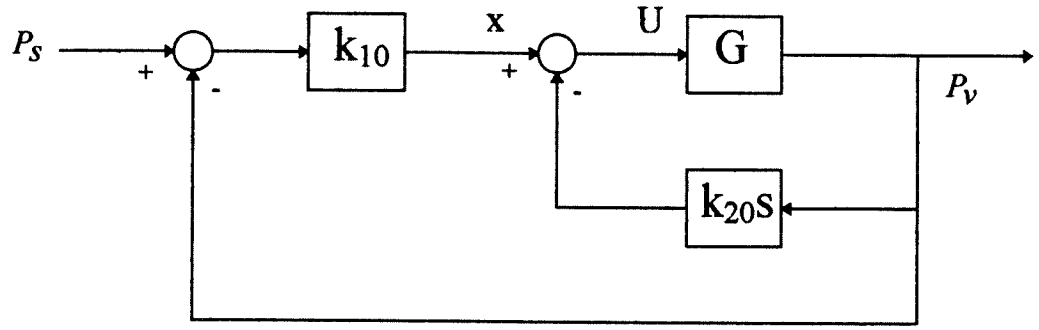


Figure 5: Schéma-bloc de l'asservissement pendant la première étape de réglage

On remarque que cette structure correspond à la loi de commande d'un régulateur PD quand elle est implantée sous la forme série avec une action dérivée uniquement sur la mesure pour éviter les écarts brusques de commande.

Cette structure permet d'établir les équations suivantes:

$$\frac{P_v}{x} = \frac{G}{1 + Gk_{20}s} \qquad \text{et} \qquad x = (P_s - P_v)k_{10}$$

En combinant ces équations, il vient

$$\frac{P_v}{P_s} = \frac{Gk_{10}}{1 + Gk_{10} + Gk_{20}s}$$

Supposons que le processus peut être modélisé par une fonction de transfert du second ordre:

$$G = \frac{K}{a_2 s^2 + a_1 s + 1} \tag{3}$$

alors $$\frac{P_v}{P_s} = \frac{1}{\frac{a_2 s^2}{Kk_{10}} + \left(\frac{a_1 + Kk_{20}}{Kk_{10}}\right)s + \frac{1 + Kk_{10}}{Kk_{10}}}$$

En considérant que $$1 + Kk_{10} \approx Kk_{10} \tag{4}$$
il s'ensuit que la fonction de transfert du système en boucle fermée est

$$\frac{P_v}{P_s} \approx \frac{1}{\alpha_{20} s^2 + \alpha_{10} s + 1} = \frac{1}{\frac{1}{\omega_{n0}^2} s^2 + \frac{2\xi}{\omega_{n0}} s + 1}$$

avec: $$\alpha_{20} = \frac{a_2}{Kk_{10}} \tag{5}$$

$$\alpha_{10} = \frac{a_1 + Kk_{20}}{Kk_{10}} \tag{6}$$

$$\omega_{n0} = \frac{1}{\sqrt{\alpha_{20}}} \tag{7}$$

$$\xi_0 = \frac{\alpha_{10}\omega_{n0}}{2} \tag{8}$$

Appelons ω_n, ξ, k_1 and k_2 les nouvelles valeurs des variables ω_{n0}, ξ_0, k_{10} and k_{20} après une première itération.
Supposons que l'on désire calculer k1 et k2 de façon à obtenir:

$$\omega_n = \mu\omega_{n0} \tag{9}$$

$$\xi = \mu_k \xi_0 \tag{10}$$

D'après (7), pour satisfaire l'équation (9), il est nécessaire de diviser α_{20} par μ^2

c'est-à-dire : $\alpha_2 = \frac{\alpha_{20}}{\mu^2}$ ou encore $\frac{a_2}{Kk_1} = \frac{1}{\mu^2}\frac{a_2}{Kk_{10}}$ (cf équation (5))

ce qui implique $\boxed{k1 = \mu^2 k_{10} = \frac{\alpha_{20}}{\alpha_2} k_{10}}$ (11)

D'autre part, d'après (8), la condition (10) est équivalente à la condition

$$\alpha_1 = \frac{2\mu_k \xi}{\omega_n} \tag{12}$$

soit encore (d'après (6)): $\frac{a_1 + Kk_2}{Kk_1} = \frac{2\mu_k \xi}{\omega_n}$ d'où $k_2 = \frac{2\mu_k \xi}{\omega_n} k_1 - \frac{a_1}{K}$

ce que l'on peut aussi écrire: $k_2 = \left(\frac{Kk_{10}\alpha_{10}}{K} - k_{10}\alpha_{10} \right) - \frac{a_1}{K} + \frac{2\mu_k \xi}{\omega_n} k_1$

$$k_2 = \frac{Kk_{10}\alpha_{10} - a_1}{K} + \frac{\alpha_{20}}{\alpha_2} k_{10} \frac{2\mu_k \xi}{\omega_n} - k_{10}\alpha_{10}$$

Finalement, d'après (5) et (12), on obtient

$$\boxed{k_2 = k_{20} + k_1\alpha_1 - k_{10}\alpha_{10}} \tag{13}$$

En supposant que les hypothèses (3) et (4) sont valables, on dispose donc des lois d'évolution des gains K_1 et K_2 (équations (11) et (13)).
En pratique le premier test réalisé pendant la première étape utilise seulement un gain proportionnel (initialisé par l'utilisateur) et après chaque test, le programme de réglage automatique analyse la réponse obtenue.

- Si cette réponse présente des oscillations importantes, la première étape s'arrête et les gains K_1 et K_2 retenus pour la seconde étape (réglage de K_3) sont ceux de l'avant-dernier test effectué.
- Sinon le sous-programme newgain.m calcule le temps *tfpa* et l'amplitude *overshoot* du premier dépassement ainsi que la pulsation naturelle et l'amortissement de l'approximation du second ordre de la réponse obtenue.
  - si les critères concernant le premier dépassement sont satisfaits, le programme passe à la seconde étape de réglage et les gains K_1 et K_2 retenus sont ceux du dernier test.
  - sinon le programme reste dans la première étape de réglage et calcule de nouveaux gains K_1 et K_2 de façon à avoir au test suivant:

- $\omega_n = \mu\omega_{n0}$ si *tfpa* > *tpd*
- $\omega_n = \omega_{n0}$ si *tfpa* < *tpd* (ce qui provoque k_1=k_{10})

- $\xi = \mu_k \xi_0$ si *overshoot* > *ampl*
- $\xi = \xi_0$ si *overshoot* < *ampl*

Après chaque test, le système retourne aux conditions initiales d'équilibre et modifie éventuellement les offsets sur les commandes des vannes afin de pouvoir lancer une nouvelle expérience avec un système stable à t=0 (sous-programme raz.m).
Sauf dans le cas où l'on ne régule que le niveau, la remise à zéro du système commence par une phase basée sur des commandes tout ou rien (vidage du réservoir jusqu'à un certain point puis remplissage avec de l'eau froide jusqu'au moment où on atteint les conditions initiales).
Dans la dernière partie de la remise à zéro, selon le choix de régulation, on utilise un simple régulateur proportionnel à gain constant et on accélère la remise à zéro en n'introduisant pas le retard habituel sur la commande.

2.1.2.-Seconde étape du réglage

Cette seconde étape consiste à trouver la plus grande valeur acceptable pour K_3. On attribue maintenant à A1 et A2 les paramètres $\alpha_1$et α_2 correspondant à la meilleure réponse obtenue pendant la première étape du réglage. Pendant toute cette étape les paramètres K1, K2, A1 et A2 sont constants.

Cette démarche s'inspire du résultat suivant:
Nous avons approximé la réponse du système asservi par un régulateur proportionnel dérivé (K_1 et K_2).

Or, d'après [DeSantis1], dans le cas d'un système de fonction de transfert $G_2(s) = \dfrac{K_M}{(1+s\tau_p)s}$, la fonction de transfert du système asservi par un régulateur proportionnel dérivé (correspondant à la figure 5) est une fonction de deuxième ordre:

$$F_{10}(s) = \frac{P_V(s)}{P_S(s)} = \frac{1}{1+2\dfrac{\xi}{\omega_n}s+\dfrac{1}{{\omega_n}^2}s^2}$$

et la fonction de transfert perturbation/sortie (perturbation ajoutée sur la commande) est

$$F_{20}(s) = \frac{P_V(s)}{P_t(s)}$$

Si l'on pose $\alpha_1 = 2\dfrac{\xi}{\omega_n}$ et $\alpha_2 = \dfrac{1}{{\omega_n}^2}$ alors

- la fonction de transfert du système asservi par le PID Dual Loop est $F_1(s) = F_{10}(s)$

- la fonction de transfert perturbation/sortie devient $F_2(s) = \dfrac{s}{s+\dfrac{K_3}{K_1}}F_{20}(s)$

Dans la deuxième étape du réglage on cherche à augmenter K_3 autant que possible afin d'améliorer la robustesse du système.

Nous nous sommes rapidement rendus compte que les équations (1) et (2) n'étaient pas strictement vérifiées, ce qui paraît normal, compte tenu des approximations effectuées. Ainsi, le dépassement augmente quand K_3 augmente et, lorsque K_3 atteint un certain seuil critique, le système se met à osciller. Par conséquent, on demande à l'utilisateur au début du programme d'indiquer l'amplitude de premier dépassement qu'il aimerait voir respectée à l'issue de la seconde phase (l'instant de ce premier dépassement devant être voisin de celui obtenu à la fin de la première phase, c'est-à-dire lorsqu'on règle les gains K_1 et K_2).
Pendant la seconde étape, le programme procède à plusieurs essais en augmentant à chaque fois la valeur de K3. Cette étape s'arrête lorsque l'amplitude maximale désirée est atteinte ou lorsque le programme détecte des oscillations excessives dans la réponse.

2.2.-Cas multivariable

Le schéma-bloc de la figure 4 est ici transformé de la façon suivante:

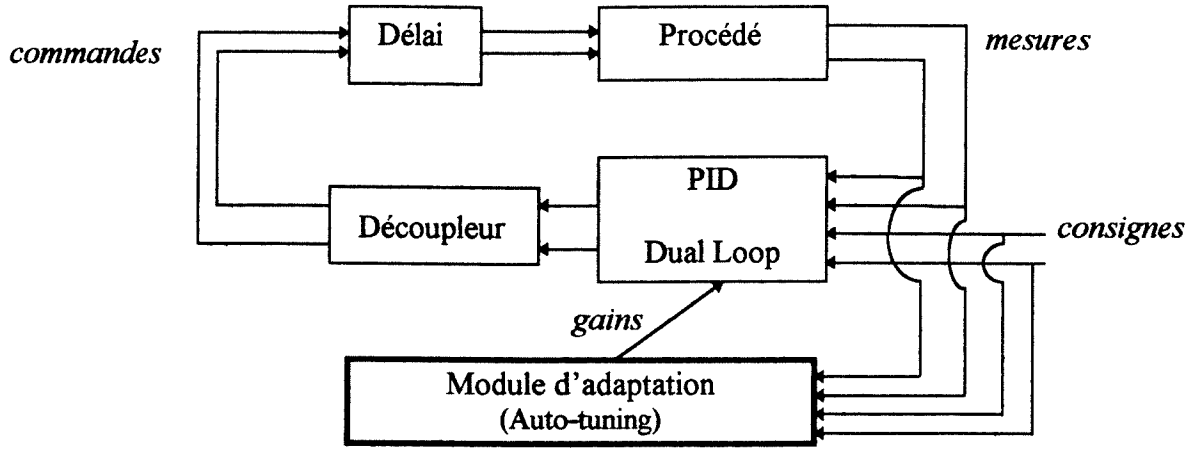


Figure 6: Schéma bloc de l'auto-tuning (multivariable)

Le principe de réglage est le même que dans le cas monovariable sauf que:
- la fin de la première étape est conditionnée par 6 critères au lieu de 3 (critères concernant les oscillations ainsi que l'amplitude et le temps de premier dépassement de la réponse en niveau et de la réponse en température).
- dans la seconde étape du réglage, on commence par ajouter progressivement un nombre égal d'incréments incK13 et incK23 (la valeur de chacun ayant été déterminé au préalable) à K13 et K23 jusqu'au moment où la sortie de l'un des deux est atteinte puis on essaie d'augmenter indépendemment K13 et K23 jusqu'à l'invalidation des critères d'oscillations et d'amplitudes de premier dépassement.

3.-Organigramme de l'algorithme de réglage automatique

L'organigramme correspondant aux méthodes décrites à la section précédente est dans l'annexe 1. L'ensemble régulateurs-modules d'adaptation est un programme développé avec le progiciel MATLAB. Ce programme (pintpri2.m) fait appel à plusieurs sous-programmes (pintauto.m, raz.m, reglak3.m, newgain.m, ordre.m, rech_osc.m) dont le listing est présenté en annexe 2.

4.-Résultats expérimentaux

On présente maintenant les tests réalisés sur le réservoir avec le programme de recherche automatique de gains pintpri2.m et raz.m (avec contrôleur proportionnel) pour la remise à zero. Seront également présentés les tests de robustesse effectués avec différentes valeurs de K_3. Ces tests (réalisés avec le logiciel pint_eds.m développé antérieurement; [Cornieles-Bougeret 3]) permettront de vérifier que l'augmentation de K_3 permet d'augmenter la robustesse sans détériorer la réponse. Les tests ont été faits avec un retard de 30s, sauf le test de la robustesse pour la température qui a éte fait avec un retard de 20s.

4.1.-Niveau

Objectifs: Il s'agit de vérifier que le programme de réglage automatique est capable de donner un jeu de gains convenables pour l'asservissement de niveau uniquement.

Modalités: On ajuste les débits initiaux d'eau chaude et d'eau froide pour obtenir l'équilibre du système (niveau constant). Dans l'interface utilisateur, on choisit K11=0.7 (valeur initiale du gain proportionnel), un retard de 30s, un temps de premier dépassement désiré de 140 s, une amplitude relative de premier dépassement de 10% pour la première étape du réglage et une amplitude relative de premier dépassement de 30% pour la seconde étape du réglage. On présente ici deux expériences réalisées avec ces conditions mais avec des incréments incK13 différents: 0.003 pour la première et 0.001 pour la seconde. Chaque test dure 750s et chaque remise à zéro dure 1000s.

Résultats:

•Première expérience: incK13=0.003

On présente ici les résultats successifs des 7 tests effectués automatiquement par le programme pintpri2.m

n°	K11	K12	K13	A11	A12	tpdn (s)	depn	oscil
1	0.70	0	0	0	0	128	0.28	-
2	0.70	3.90	0	0	0	136	0.22	-
3	0.70	8.56	0	0	0	165	0.07	-
4	0.85	12.2	0	0	0	74.5	0.03	-
5	0.85	12.2	0.003	37.3	347	101	0.27	-
6	0.85	12.2	0.006	37.3	347	94	0.29	-
7	0.85	12.2	0.009	37.3	347	91	0.47	-

Tableau 1: Résultats de la première expérience concernant le niveau

On remarque que la première étape du réglage correspond aux 4 premiers tests car les critères de premier dépassement sont respectés au quatrième test (74.5 s<140 s et 3%<10%). La seconde étape s'arrête au septième test car l'amplitude du premier dépassement y est supérieure à l'amplitude désirée (47%>30%). On remarque que l'erreur en régime permanent diminue avec l'incrément de K13. Les gains définitifs trouvés correspondent donc au sixième test:

K11=0.85 K12=12.2 K13=0.006 A11=37.3 A12=347

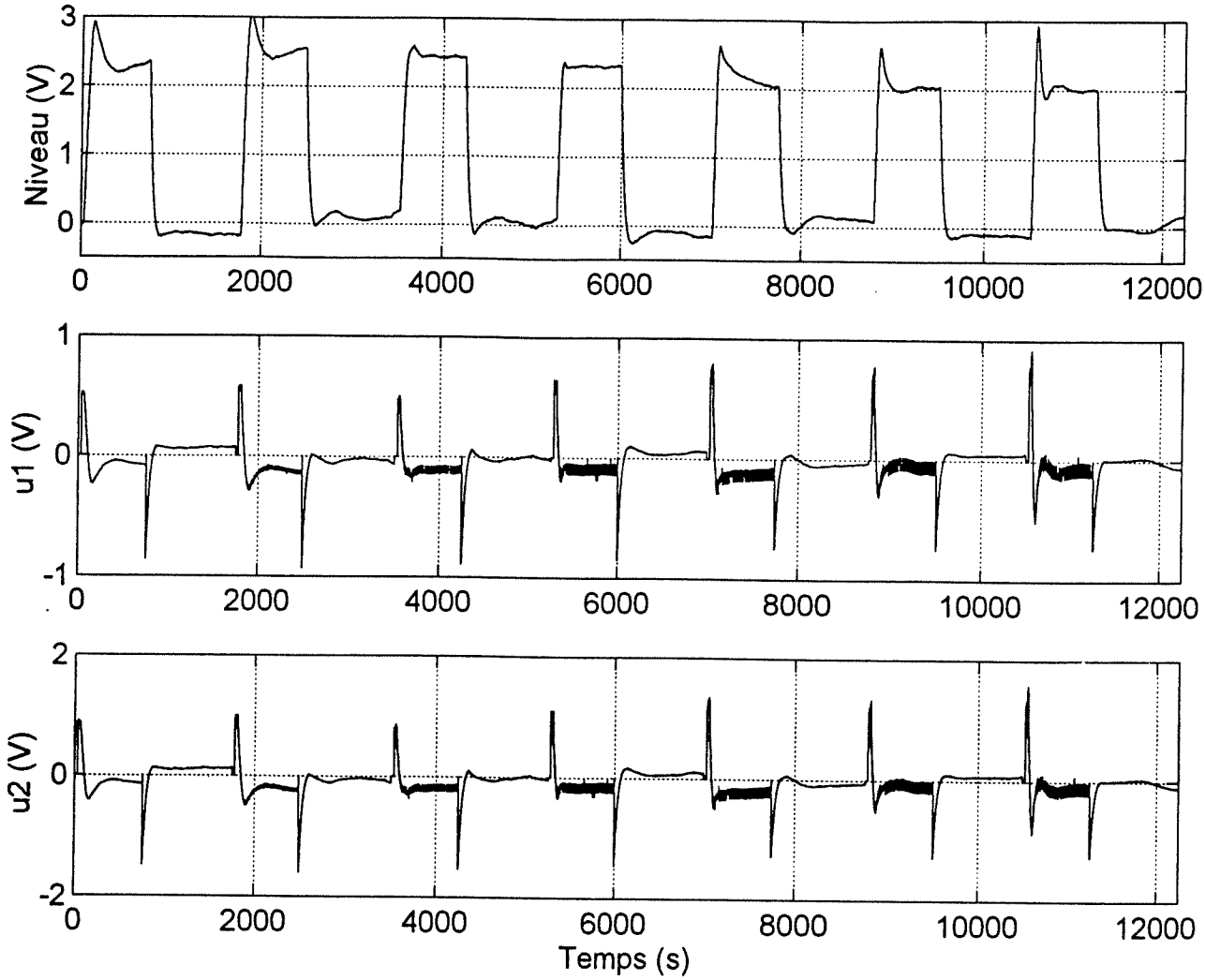


Figure 7: Résultat global du réglage automatique (niveau et commandes) des 7 tests

Sur la figure ci-dessous sont superposées les réponses du système aux 4 tests de la première étape de réglage (réglage de K11 et de K12).

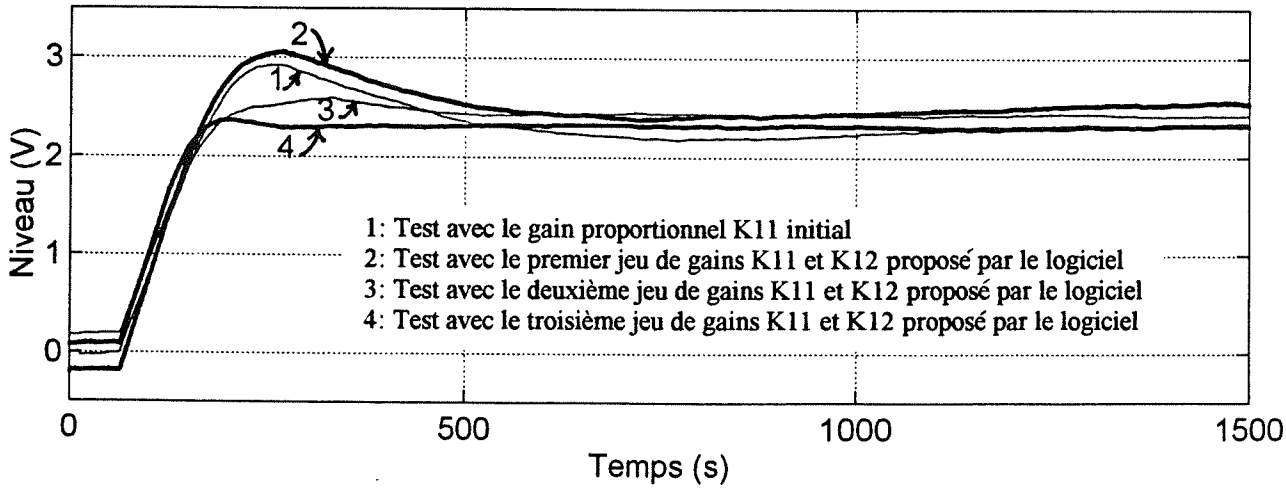


Figure 8 : Évolution de la réponse lors de la première étape du réglage (niveau)

•Test de la robustesse

Pour tester la robustesse du système, on ajoute un échelon de perturbation de 1 volt sur la commande de la vanne d'eau froide à t=500 s (quand le régime stationnaire est établi). On présente ici le comportement du niveau du réservoir lorsqu'il est asservi par deux jeux de gains différents. Le premier jeu de gains utilisé est celui trouvé par l'auto-tuning. Le second jeu est le même que le premier sauf que le gain K13 a été diminué jusqu'à incK13 (K13=0.003)

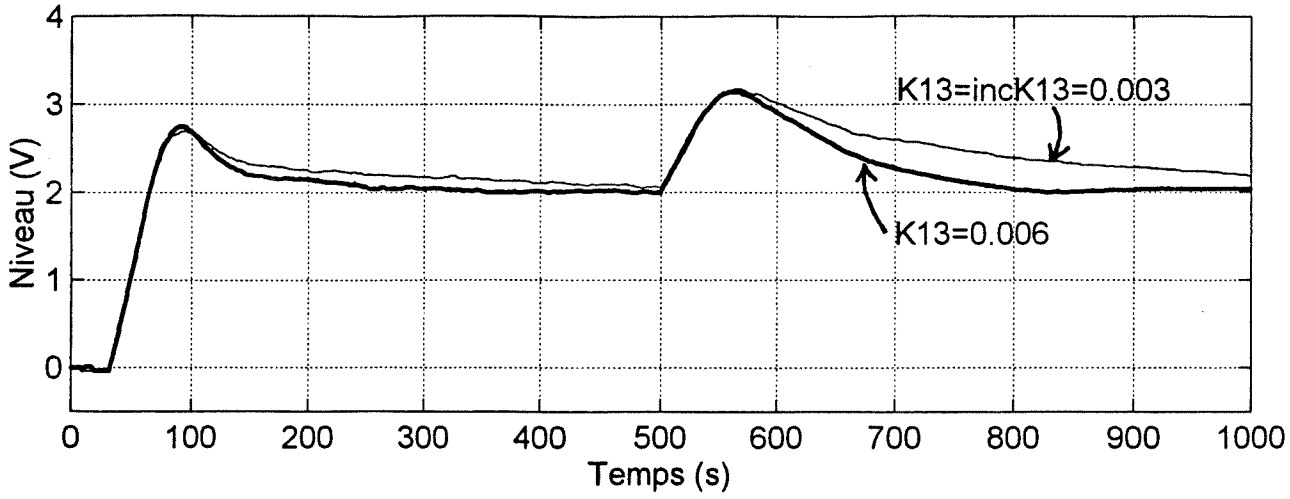


Figure 9: Évolution de la robustesse avec l'augmentation de K13 (niveau)

•Analyse des résultats

À la figure 7 on constate que les gains K11 et K12 sont modifiés progressivement lors de la première étape de réglage de façon à satisfaire les critères fixés par l'utilisateur. Bien sûr, ces critères sont raisonnables et s'inspirent des résultats obtenus avec un réglage manuel [Cornieles-Bougeret 3]. En effet nous avons pu vérifier expérimentalement que des critères trop sévères sur le temps du premier dépassement mènent à des jeux de gains qui provoquent des oscillations dans la réponse.
En ce qui concerne le réglage de K13, on constate au tableau 1 que plus K13 est élevé, plus l'amplitude de premier dépassement est grande. Cette constatation montre que les approximations (3) et (4) ont des limites. Malgré tout, le temps de premier dépassement est assez bien conservé et ces approximations permettent d'obtenir une méthode de réglage qui donne un jeu de gains satisfaisant.
D'autre part, la figure 9 montre que plus K13 est élevé , plus le système est robuste, ce qui va bien dans le sens de l'équation (2).

Puisque la seconde étape de réglage ne comprend ici que 3 tests, une deuxième expérience sur le niveau va être menée avec les mêmes conditions logicielles initiales mais avec un incrément incK13 plus petit. Cela permettra peut-être d'affiner le réglage de K13.

•Deuxième expérience: incK13=0.001

On présente ici les résultats successifs des 15 tests effectués automatiquement par le programme pintpri2.m.

n°	K11	K12	K13	A11	A12	tpdu (s)	depu	oscil
1	0.7	0	0	0	0	155	0.35	-
2	0.85	6.62	0	0	0	113	0.31	-
3	0.85	10.6	0	0	0	104	0.13	-
4	0.85	15.7	0	0	0	74	0.01	-
5	0.85	15.7	0.001	38.5	371	71	0.03	-
6	0.85	15.7	0.002	38.5	371	94	0.08	-
7	0.85	15.7	0.003	38.5	371	86	0.11	-
8	0.85	15.7	0.004	38.5	371	91	0.16	-
9	0.85	15.7	0.005	38.5	371	81	0.16	-
10	0.85	15.7	0.006	38.5	371	80	0.23	-
11	0.85	15.7	0.007	38.5	371	80	0.21	-
12	0.85	15.7	0.008	38.5	371	77	0.25	-
13	0.85	15.7	0.009	38.5	371	78	0.25	-
14	0.85	15.7	0.010	38.5	371	75	0.29	-
15	0.85	15.7	0.011	38.5	371	74	0.35	-

Tableau 2: Résultats de la deuxième expérience concernant le niveau

On remarque que la première étape du réglage correspond aux 4 premiers tests car les critères de premier dépassement sont respectés au quatrième test (74 s<140 s et 1%<10%).
La seconde étape s'arrête au 15ème test car l'amplitude du premier dépassement y est supérieure à l'amplitude désirée (35%>30%). Les gains définitifs trouvés correspondent donc au quatorzième test. Ce sont:
K11=0.85 K12=15.7 K13=0.01 A11=38.5 A12=371

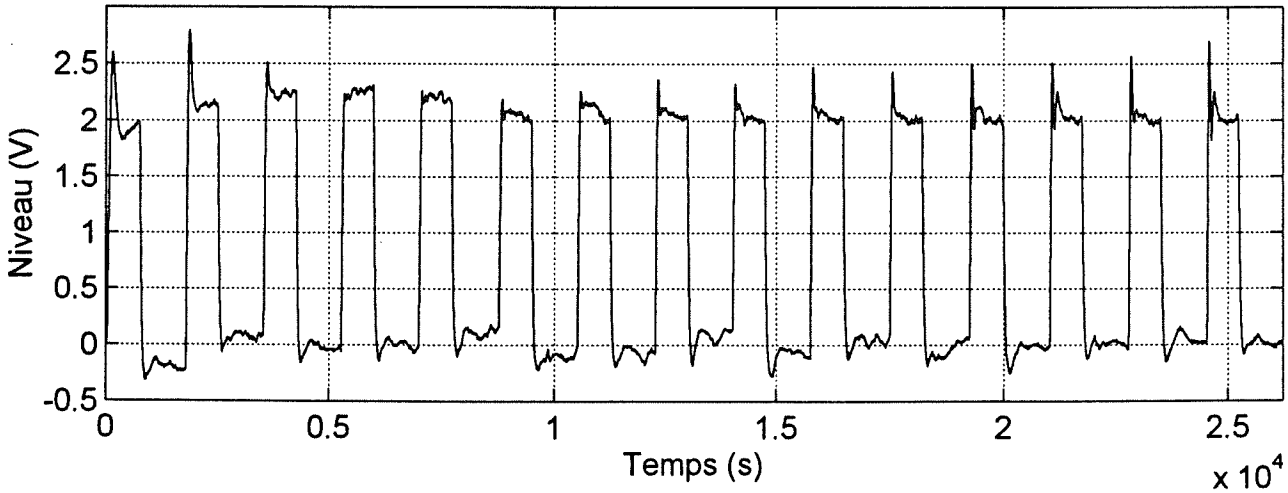


Figure 10: Résultat global du réglage automatique (niveau) des 15 tests

À la figure 11 sont superposées les réponses du système aux 4 tests de l'étape de réglage de K11 et K12.

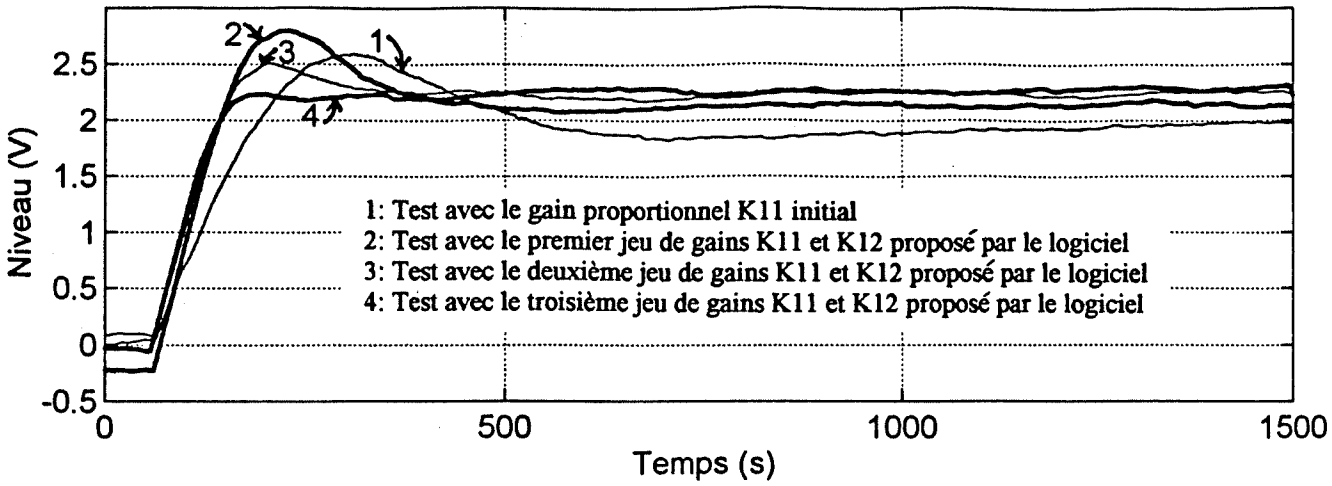


Figure 11: Évolution de la réponse lors de la première étape du réglage (niveau)

Remarque: La figure 10 et 11 font voir que le programme raz.m remet le niveau à zéro après chaque test. Il le fait au moyen d'un compensateur proportionnel qui à un gain quelconque.

•Test de la robustessee

La perturbation utilisée pour ce test est la même que celle utilisée pour la première expérience (cf p21). Le premier jeu de gains utilisé est celui trouvé par l'auto-tuning. Le second jeu est le même que le premier sauf que le gain K13 a été diminué jusqu'à incK13 (K13=0.001). L'erreur en régime permanent diminue avec l'incrément de K13.

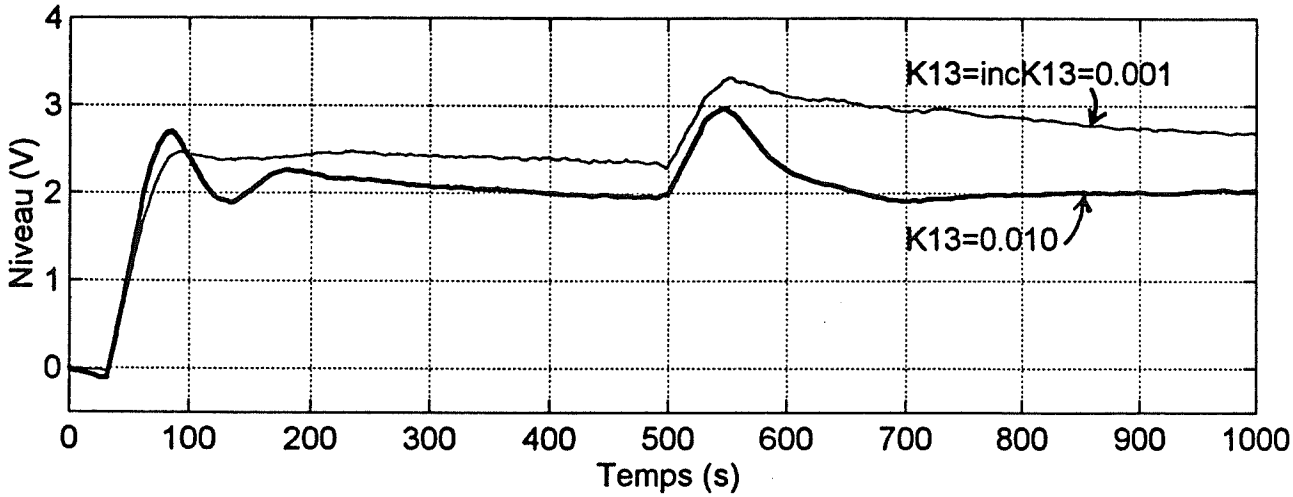


Figure 12: Évolution de la robustesse avec l'augmentation de K13 (niveau)

•Analyse des résultats

L'analyse est globalement la même que pour la première expérience sur le niveau. On se rend compte qu'avec les mêmes conditions logicielles initiales on obtient un réglage différent, notamment un gain K13 plus élevé. Cela est bien sûr dû à une température et à une pression légèrement différente pour chaque alimentation en eau (expérience faite pendant la nuit). L'influence de K13 sur la robustesse du système est encore plus flagrante à la figure 12 où le rapport entre les gains K13 testés est encore plus grand qu'à la figure 9 (rapport 10 contre rapport 2).

4.2.-Température

Objectifs: Il s'agit de vérifier que le programme de réglage automatique est capable de donner un jeu de gains convenables pour l'asservissement de température.

Modalités: On ajuste les débits initiaux d'eau chaude et d'eau froide pour obtenir l'équilibre du système (niveau constant à 0 volt et température à 2 volts). Dans l'interface utilisateur, on choisit K21=2 (valeur initiale du gain proportionnel), un temps de premier dépassement désiré de 200 s, une amplitude relative de premier dépassement de 10% pour la première étape du réglage, une amplitude relative de premier dépassement de 20% pour la seconde étape du réglage et un incrément incK23 égal à 0.01. Chaque test dure 1500s et chaque remise à zéro dure 1250s.

Résultats:

On présente ici les résultats successifs des 8 tests effectués automatiquement par le programme pintpri2.m.

n°	K21	K22	K23	A21	A22	tpdt (s)	depl	oscil
1	2	0	0	55	3072	201	0.17	-
2	2.42	35	0	75	1416	151	0.05	-
3	2.42	35	0.01	75	1416	187	0.01	-
4	2.42	35	0.02	75	1416	351	0.06	-
5	2.42	35	0.03	75	1416	287	0.11	-
6	2.42	35	0.04	75	1416	168	0.13	-
7	2.42	35	0.05	75	1416	203	0.11	-
8	2.42	35	0.06	75	1416	200	0.13	oui

Tableau 3: Résultats de l'expérience concernant la température

On remarque que la première étape du réglage correspond aux 2 premiers tests car les critères de premier dépassement sont respectés au deuxième test (151s<200 s et 5%<10%).
La seconde étape s'arrête au huitième test car le programme détecte la présence des oscillations.
Les gains définitifs trouvés correspondent donc au septième test.
Ce sont:

K21=2.42 K22=35 K23=0.05 A21=75 A22=1416

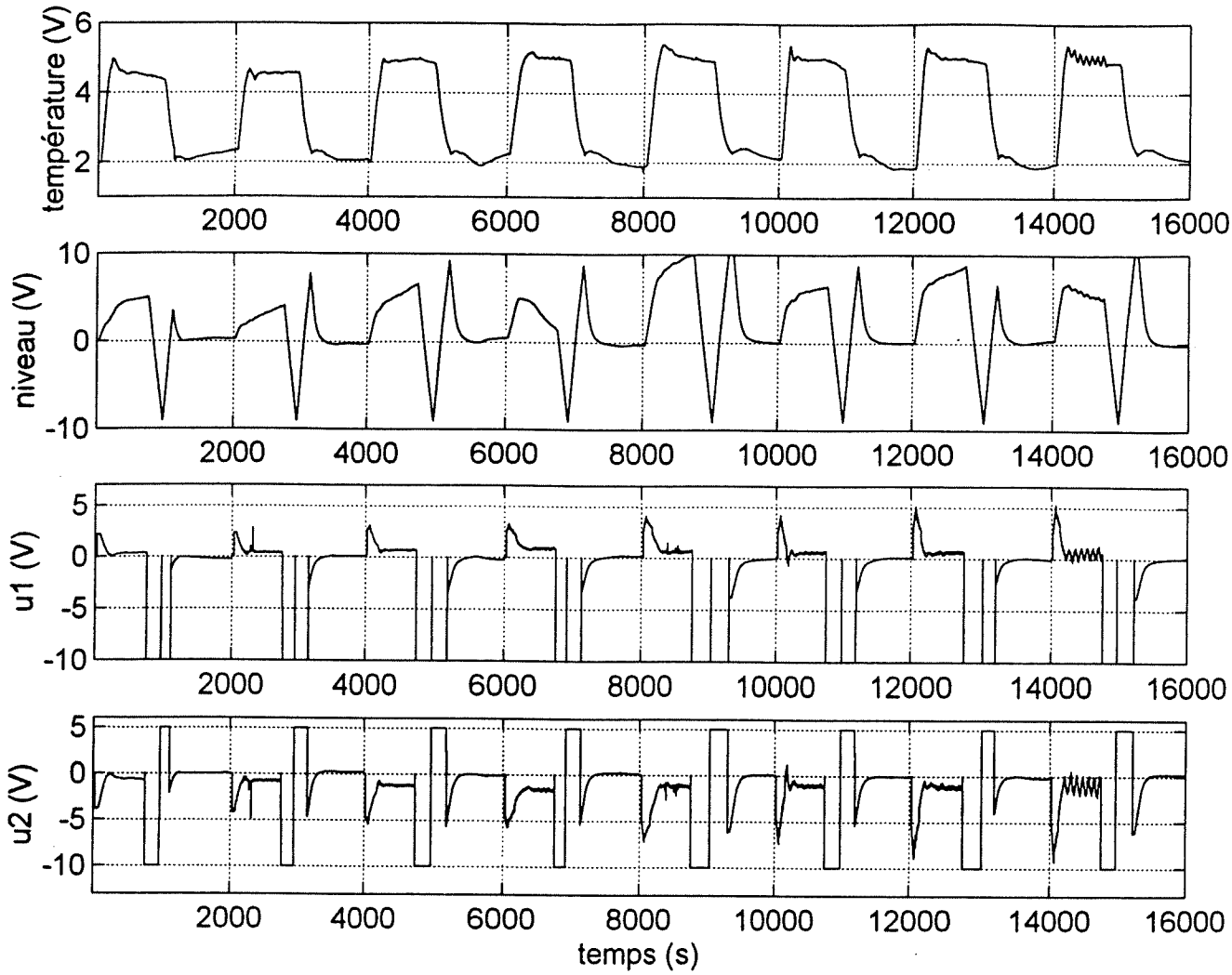


Figure 13: Résultat global du réglage automatique (température et commandes)

Sur la figure ci-dessous sont superposées les réponses du système aux 2 tests de l'étape de réglage de K21 et K22.

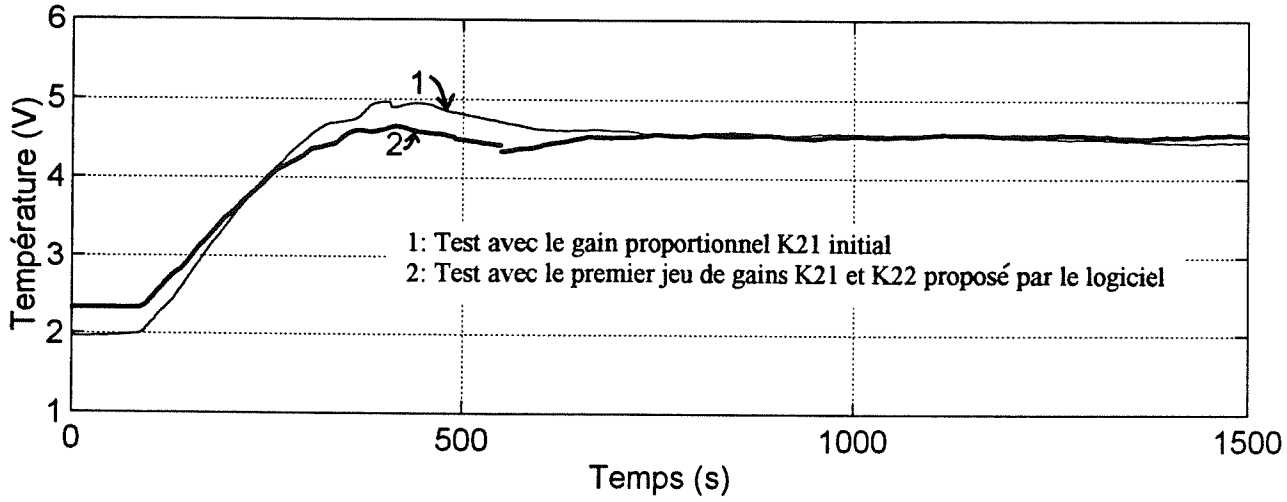


Figure 14: Évolution de la réponse lors de la première étape du réglage (température)

•Test de la robustesse

On présente le comportement de la température face à un échelon de perturbation de 1.92 volt sur la commande de la vanne d'eau froide et de -1.08 volt sur la commande de la vanne d'eau chaude. Les valeurs numériques de cette perturbation ont été calculées en fonction des gains des vannes pneumatiques afin de ne pas trop perturber le niveau de l'eau dans le réservoir.

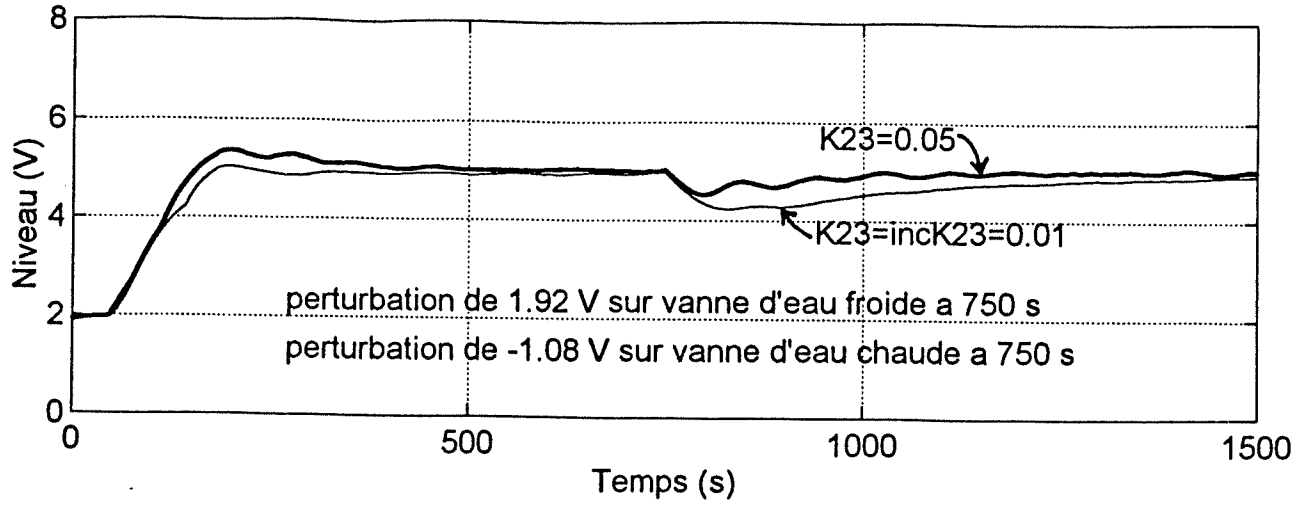


Figure 15: Évolution de la robustesse avec l'augmentation de K23 (température)

•Analyse des résultats

La première étape de réglage n'est pas très longue (seulement 2 tests). Peut-être aurait-on dû être un peu plus sévère sur les critères de premier dépassement mais les critères choisis semblaient déjà meilleurs que les résultats obtenus avec un réglage manuel du Dual Loop (cf [Cornieles-Bougeret 3]). En tout cas, on vérifie que le programme analyse bien la réponse du système au premier test (système asservi par un simple régulateur proportionnel).
À propos du gain proportionnel du premier test, signalons qu'il ne faut pas initialiser le gain proportionnel K21 avec une valeur trop grande car sinon le programme calcule des gains K21 et K22 qui provoquent des oscillations dans la réponse dès les premières itérations. Cela paraît normal puisque le programme est incapable de diminuer la valeur de K21 lors d'une itération (cf équation (11)).
En ce qui concerne le réglage de K23, on constate au tableau 3 que l'introduction de K23 a plutôt tendance à faire augmenter le temps et l'amplitude du premier dépassement. Mais surtout on s'aperçoit qu'une valeur trop grande pour K23 provoque des oscillations dans la réponse. Cette constatation rejoint celle qui a été faite pour les expériences sur le niveau, à savoir que les approximations (3) et (4) ne sont légitimes que pour des valeurs suffisamment faibles.
Malgré tout, ces approximations permettent d'obtenir une méthode de réglage qui donne un jeu de gains satisfaisant.
D'autre part, la figure 15 montre que plus K23 est élevé, plus le système rejoint rapidement le régime stationnaire après l'introduction des échelons de perturbation. Cette constatation va bien dans le sens de l'équation (2).

4.3.-Niveau et température

Objectifs: Il s'agit de vérifier que le programme de réglage automatique est capable de donner un jeu de gains convenables pour l'asservissement de niveau et de température.

Modalités: On ajuste les débits initiaux d'eau chaude et d'eau froide pour obtenir l'équilibre du système (niveau constant à 0 volt et température à 2 volts). Dans l'interface utilisateur, on choisit K11=0.7 et K21=1.5 (valeurs initiales des gains proportionnels pour le niveau et la température), un temps de premier dépassement désiré pour le niveau de 180 s, une amplitude relative de premier dépassement de 20% pour la première étape du réglage, une amplitude relative de premier dépassement de 40% pour la seconde étape du réglage et un incrément incK13 de 0.001. Pour la température un temps de premier dépassement désiré de 180 s, une amplitude relative de premier dépassement de 10% pour la première étape du réglage, une amplitude relative de premier dépassement de 20% pour la seconde étape du réglage et un incrément incK23 de 0.003. La durée de chaque test est de 1000s et la durée de remise à zéro de 1250s.

Résultats:

On présente ici les résultats successifs des 10 tests effectués automatiquement par le programme pintpri2.m.

n°	K11	K12	K13	A11	A12	K21	K22	K23	A21	A22	tpdn (s)	depn	tpdt (s)	depl	oscil
1	0.7	0	0	0	0	1.5	0	0	0	0	163	0.30	216	0.21	-
2	0.7	4.84	0	0	0	1.82	26.1	0	0	0	186	0.33	179	0.15	-
3	0.85	13.1	0	0	0	1.82	44.5	0	0	0	203	0.15	141	0.04	-
4	1.02	18.0	0	0	0	1.82	44.5	0	0	0	162	0.06	145	0.01	-
5	1.02	18.0	0.001	50.8	1455	1.82	44.5	0.003	72.3	1305	216	0.22	172	0.01	-
6	1.02	18.0	0.002	50.8	1455	1.82	44.5	0.006	72.3	1305	188	0.20	187	0.01	-
7	1.02	18.0	0.003	50.8	1455	1.82	44.5	0.009	72.3	1305	277	0.14	260	0.08	-
8	1.02	18.0	0.004	50.8	1455	1.82	44.5	0.012	72.3	1305	-	-	204	0.04	niveau
9	1.02	18.0	0.004	50.8	1455	1.82	44.5	0.009	72.3	1305	-	-	242	0.04	niveau
10	1.02	18.0	0.003	50.8	1455	1.82	44.5	0.012	72.3	1305	-	-	414	0.12	niveau

Tableau 4: Résultats de l'expérience concernant le niveau et la température

On remarque que la première étape du réglage correspond aux 4 premiers tests car les critères de premier dépassement pour le niveau sont respectés au quatrième test (162s<180 s et 6%<20%) et aussi pour la température (145s<180s et 1%<10%). Du test 5 jusqu'au test 8, le programme augmente les valeurs de K13 et de K23. Suite aux oscillations rencontrées pendant le test 8, le programme essaie d'augmenter K13 sans augmenter K23 puis inversement (essais 9 et 10) mais la réponse en niveau présente des oscillations lors de ces deux essais. Les gains définitifs trouvés correspondent donc au septième test. Ce sont:

K11=1.02 K12=18 K13=0.003 A11=50.8 A12=1455 (pour le niveau)

K21=1.82 K22=44.5 K23=0.009 A21=72.3 A22=1305 (pour la température)

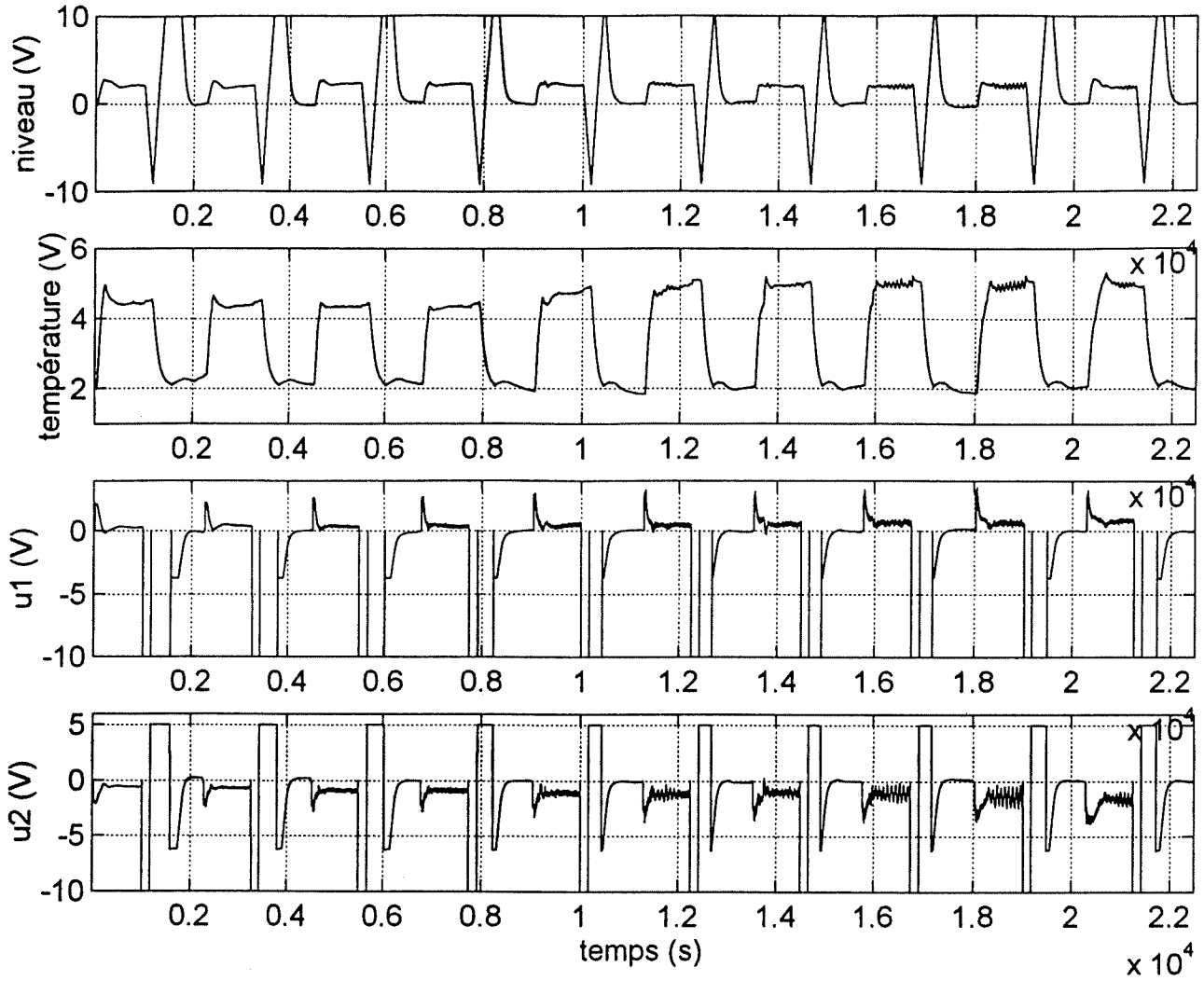


Figure 16: Résultat global du réglage automatique (commandes, niveau et température)

À la figure 17 et à la figure 18 sont superposées les réponses en niveau et en température lors des 4 tests de l'étape de réglage de K11, K12, K21 et K22.

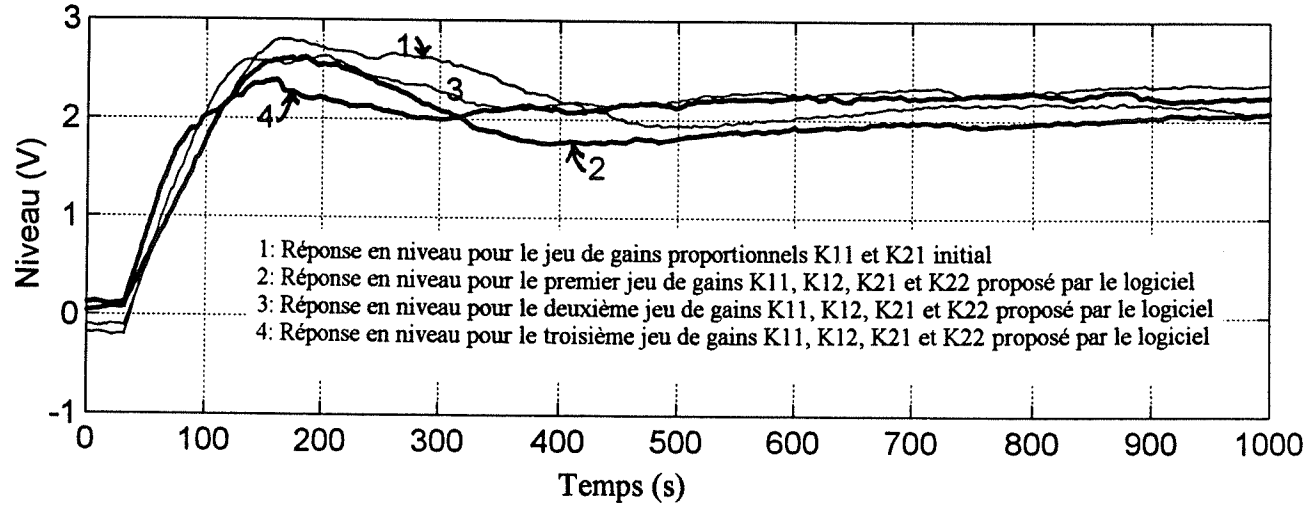


Figure 17: Évolution de la réponse en niveau lors de la première étape du réglage

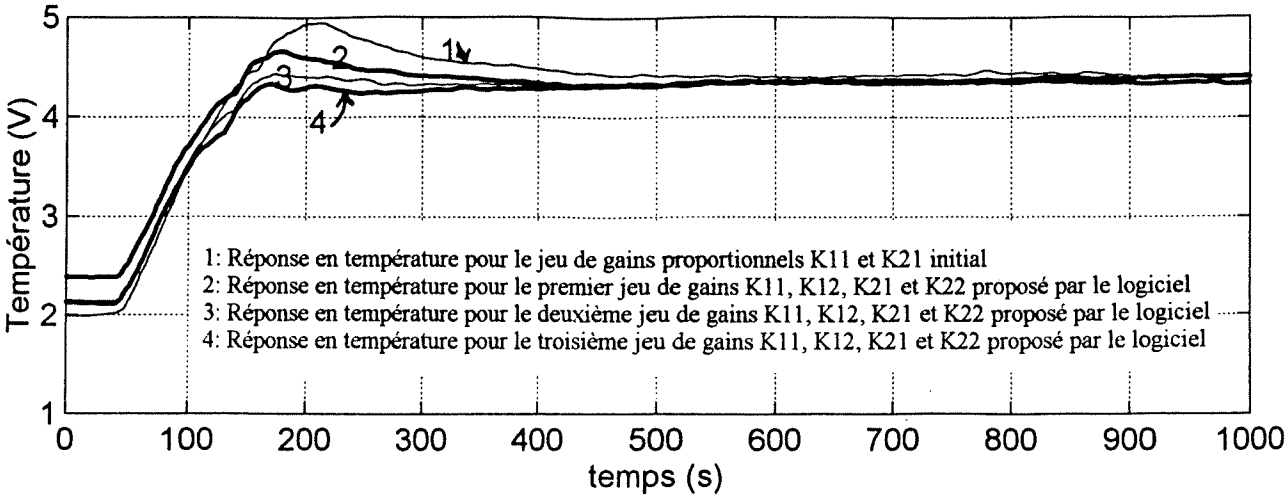


Figure 18: Évolution de la réponse en température lors de la première étape du réglage

•Test de la robustesse

On présente le comportement du niveau et de la température face à une perturbation de 1 volt ajoutée sur la commande de la vanne d'eau froide à t=1000 s.

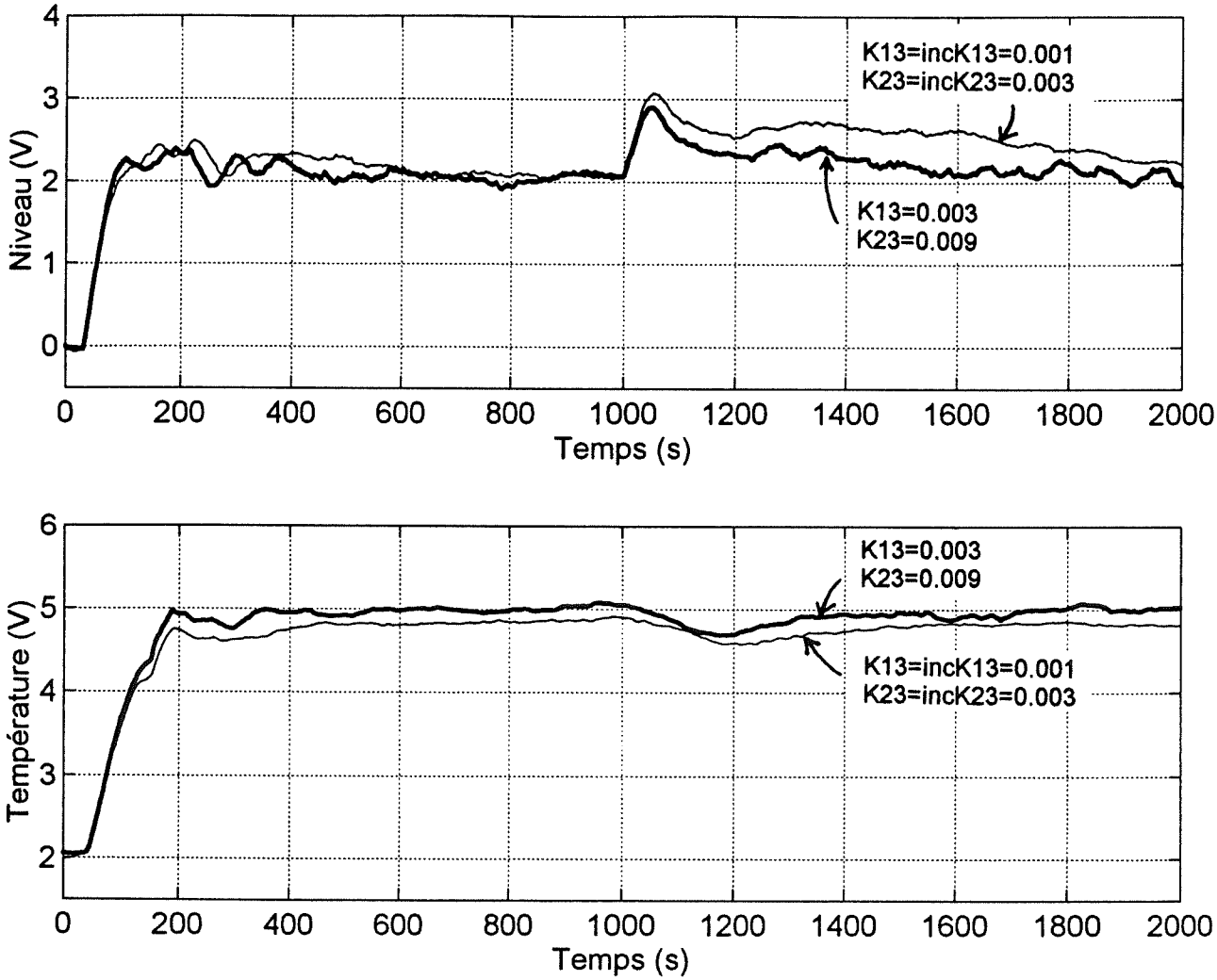


Figure 19: Évolution de la robustesse avec l'augmentation de K13 et K23 (niveau et température)

•Analyse des résultats

Le résultat obtenu en réglant automatiquement et simultanément les régulateurs des deux boucles est meilleur que le résultat obtenu en utilisant simultanément les réglages trouvés séparément et manuellement pour chacune des deux boucles (cf [Cornieles-Bougeret 3]). De même, si l'on compare le résultat du réglage simultané des deux boucles avec le résultat obtenu en utilisant les réglages trouvés pour chaque boucle (figure 20, avec retard et perturbation au milieu de l'expérience), on se rend compte que là encore le réglage multivariable est supérieur.
D'après le tableau 4, il semble plus facile de régler la boucle de température. Cela signifie que l'asservissement de température perturbe plus le niveau que l'asservissement de niveau ne perturbe la température. Cela semble normal car les proportions $\frac{1}{1+\alpha_r}$ et $\frac{\alpha_r}{1+\alpha_r}$ de la répartition (cf figure 1) sont telles que l'eau ajoutée (ou retranchée) est environ à 5 volts (en termes de mesure de température), ce qui aide la boucle de température.
Comme le montre la figure 19, plus les valeurs de K13 et K23 sont élevées, plus le système converge rapidement vers les consignes de niveau et de température en cas de suivi de consigne ou de régulation

5.- Difficultés techniques

Diverses difficultés techniques ont été rencontrées au cours de ce projet notamment au niveau de l'ordinateur dont la mémoire était insuffisante. Le PC a donc dû être changé afin de pouvoir enregistrer et afficher des expériences de plusieurs heures. On a utilisé un Pentium 166 Mhz avec 32 Mo de mémoire vive.
D'autre part, un filtre du premier ordre avec une constante de temps de 20s a été ajouté sur la mesure de la température. Cela a permis de "lisser" la réponse fournie par le thermocouple et d'introduire un détecteur d'oscillations fonctionnant correctement. La détermination des nombreuses constantes présentes dans les sous-programmes (coefficients μ et μ_k, taux maximal d'ondulation, seuil de détection d'un maximum local, temps de remise à zéro, critère de régime stationnaire atteint...) a également été assez difficile et a demandé beaucoup de tests (notamment nocturnes, pour gagner du temps). En effet le système n'est jamais le même car:
- la pression d'alimentation en eau varie au cours de la journée (hausse de pression le midi par exemple)
- la température de l'eau chaude varie aussi (cycle à 120 degrés Fahrenheit de 8 heures à 16 heures et à 97 degrés Fahrenheit la nuit, cycle différent en fin de semaine, pannes de chauffage...)
- la température de l'eau froide a augmenté au printemps puis, bizarrement, est redescendue pendant les jours plus chauds, probablement à cause d'un système de réfrigération. Mais pendant l'été la température de l'eau froide a remontée.
Enfin, la longueur des expériences a obligé à respecter certains horaires pour lancer les expériences car il est important que la dynamique du système ne varie pas trop au cours de la phase de réglage automatique.

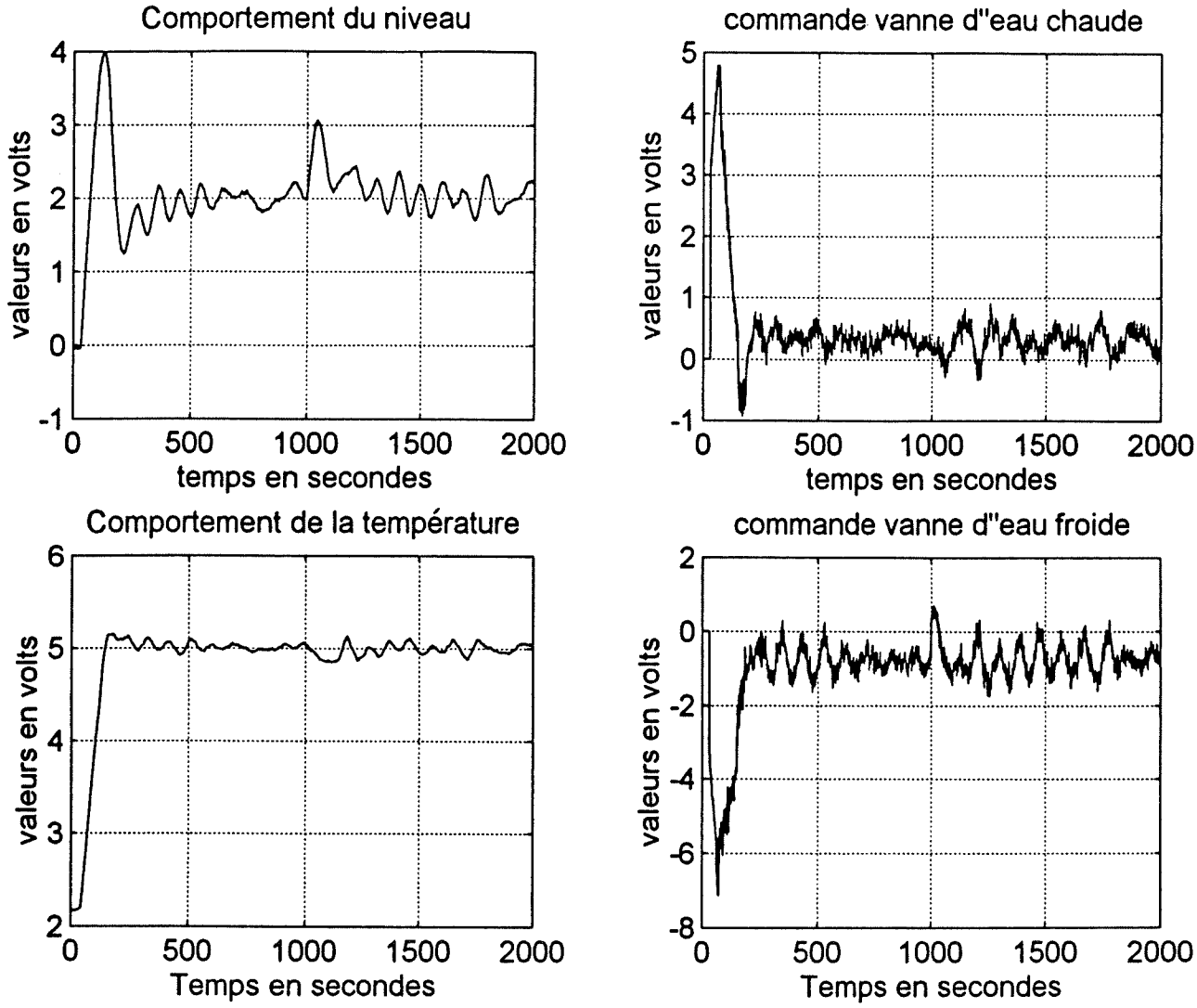


Figure 20: Comportement du système lorsque K11=0.85, K12=12.2, K13=0.006, A11=37.3 A12=347, K21=2.42, K22=35, K23=0.05, A21=75, A22=1416 (avec retard de 30s et une perturbation d'un volt sur la commande d'eau froide)

Conclusion

Le système de réglage automatique des gains d'un PID Dual Loop que nous avons mis au point semble bien répondre à la fonction qui lui incombe. Tout comme les autres systèmes de réglage proposés par la littérature, notre système nécessite la détermination préalable d'un certain nombre de paramètres tels que le pas d'incrément des gains, l'amplitude du premier dépassement et le facteur d'amortissement. Cette détermination peut être effectuée en fonction de la classe de systèmes à asservir ainsi que des spécifications d'opérations requises. Dans son application à l'asservissement de niveau et de température d'un réservoir d'alimentation, notre système a permis un réglage de gains satisfaisant dans un grand nombre d'essais. Ces gains ont fait que le comportement du niveau et de la température convergent rapidement et d'une façon stable vers les consignes. Cela malgré, la diversité des conditions de fonctionnement du banc d'essai (variation de la pression d'air, de la température de l'eau d'alimentation, variation du débit de sortie du réservoir et la présence de perturbations électriques).

BIBLIOGRAPHIE

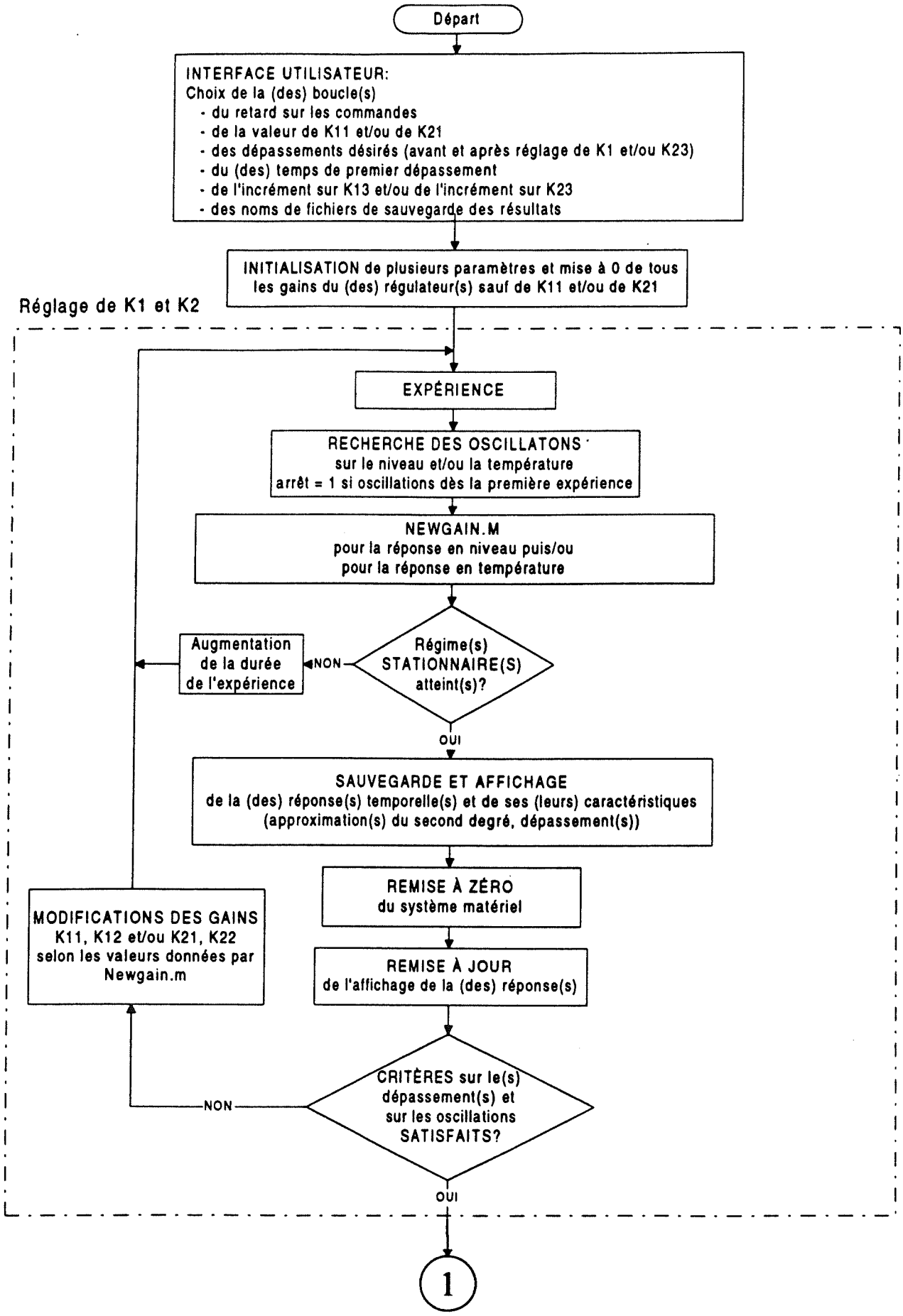
[1] Romano M. DeSantis, **A Novel PID Configuration for A Speed and Position Control,** Transactions of the ASME, vol 116, pp.542-549, September 1994.

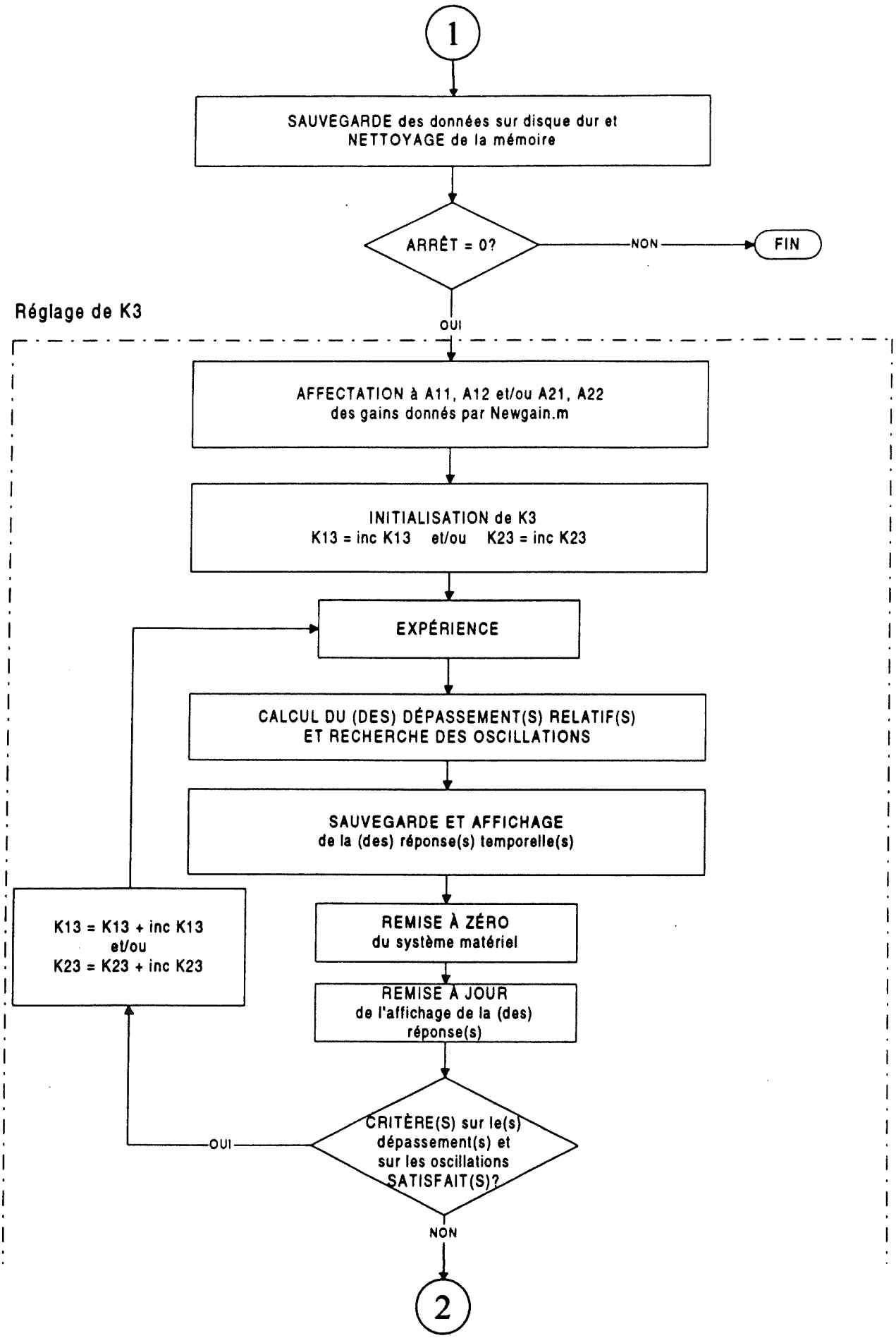
[2] Ernesto Cornieles et Geoffrey Mavel, **Asservissement en Niveau et Température d'un Réservoir d'Eau avec un Contrôleur PID Dual Loop,** École Polytechnique de Montréal, Génie Électrique, Automatique, rapport interne, Février 1997.

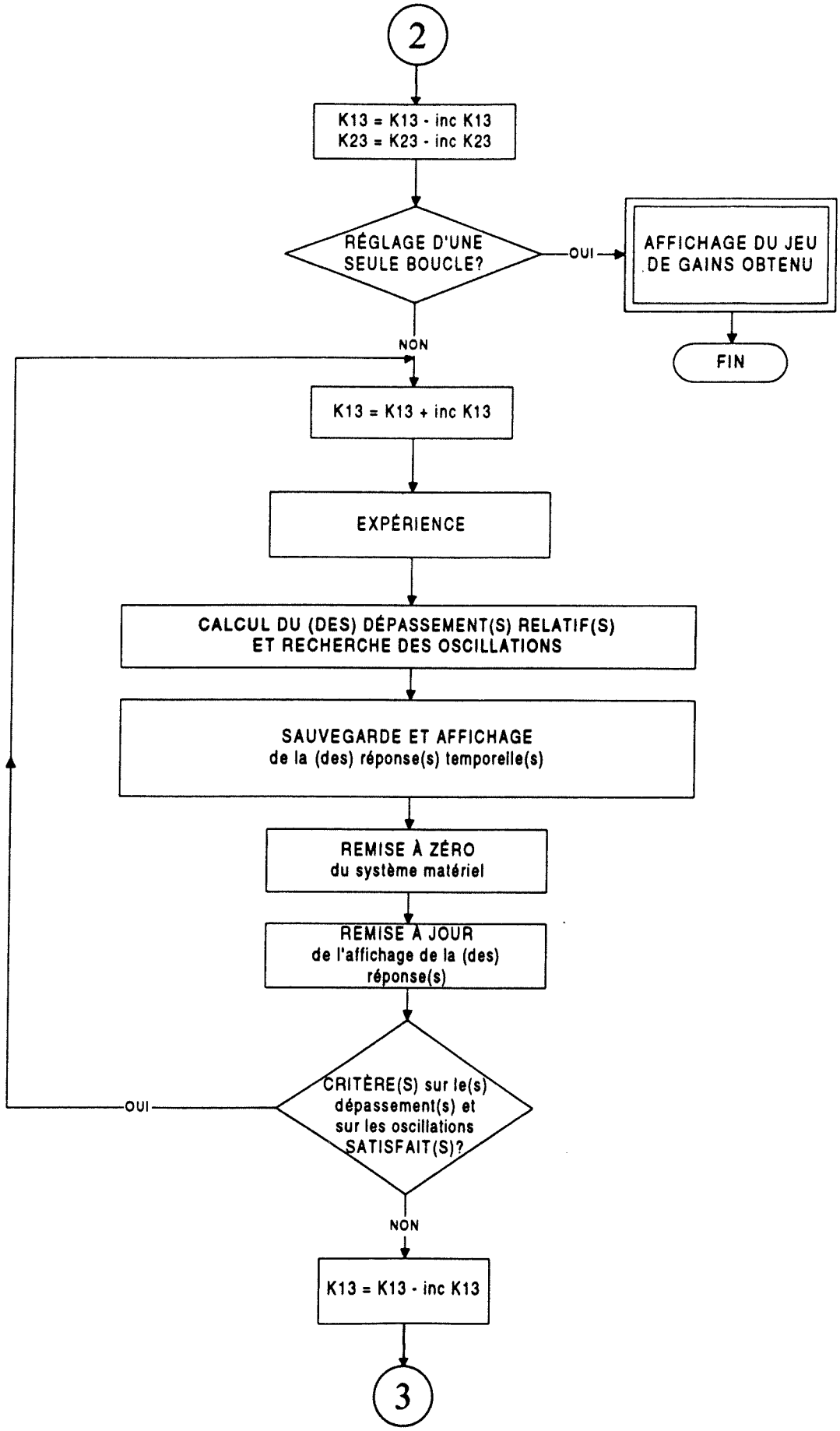
[3] Ernesto Cornieles et Christophe Bougeret, **Comparaison Expérimentale de Différentes Techniques de Réglage du Régulateur PID et PID Dual Loop,** École Polytechnique de Montréal, Génie Électrique, Automatique, rapport officiel, EPM/RT-97/19, Juin 1997.

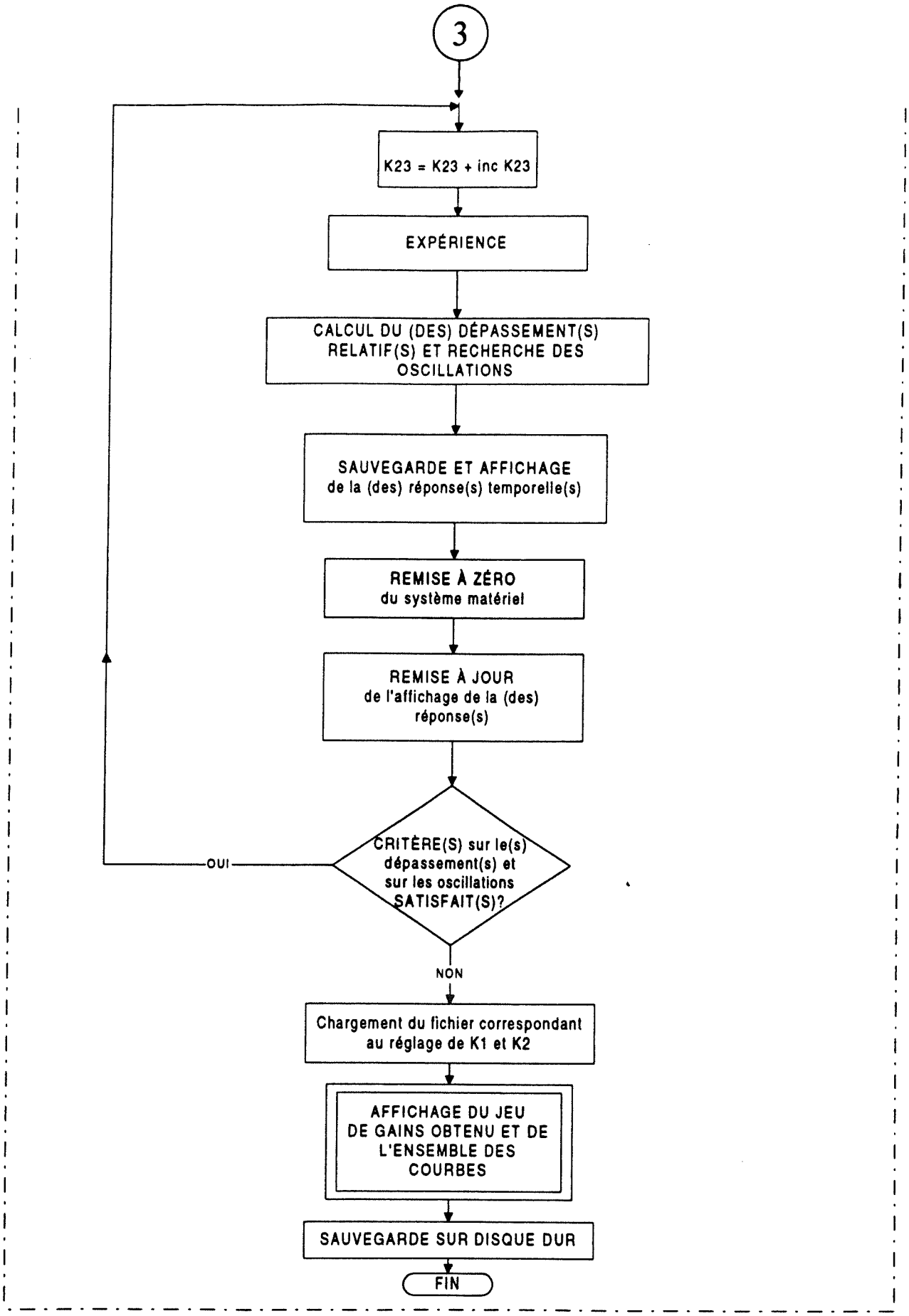
Annexe 1 :

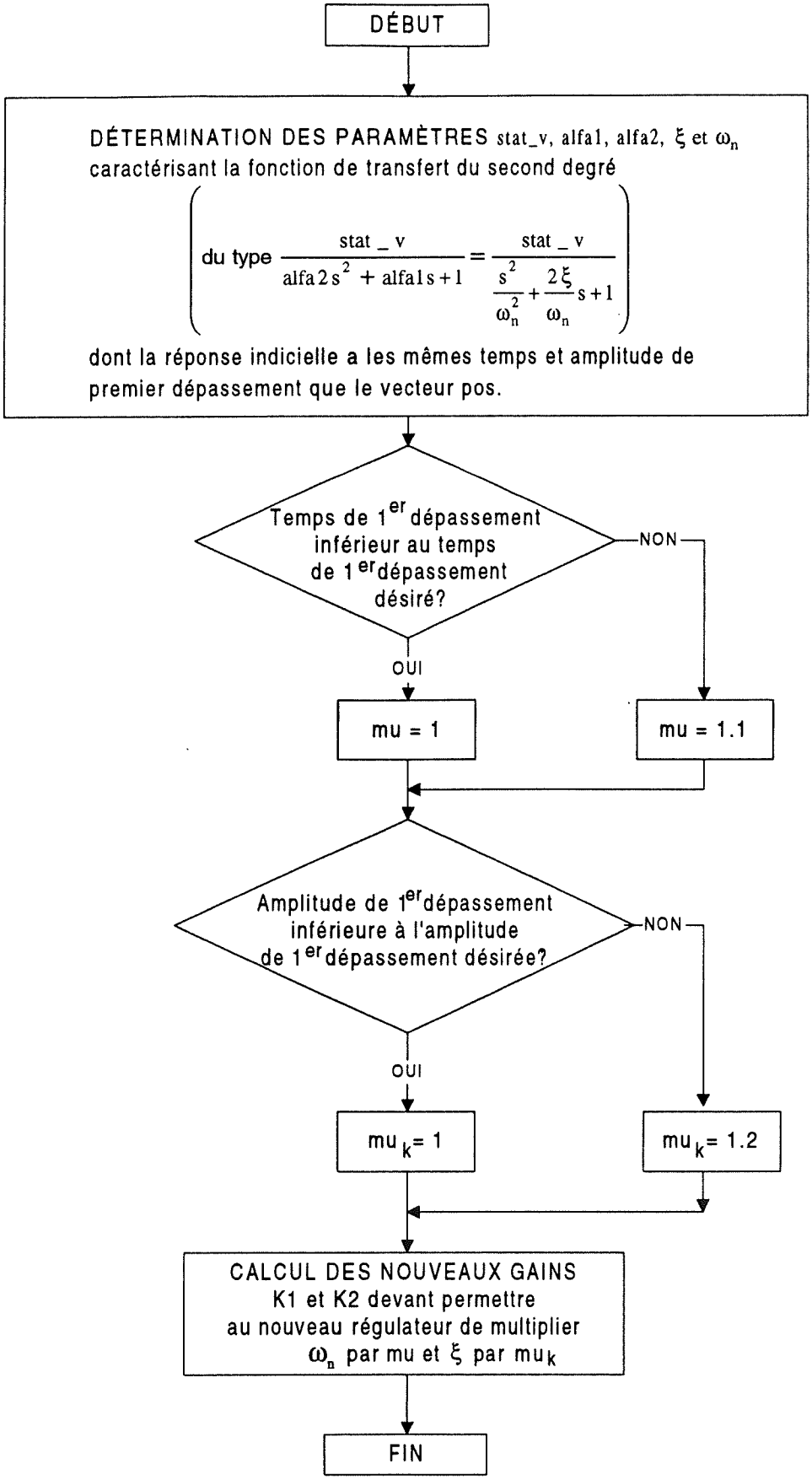
Organigramme de la boucle de réglage automatique


Départ
INTERFACE UTILISATEUR:
Choix de la (des) boucle(s)
- du retard sur les commandes
- de la valeur de K11 et/ou de K21
- des dépassements désirés (avant et après réglage de K1 et/ou K23)
- du (des) temps de premier dépassement
- de l'incrément sur K13 et/ou de l'incrément sur K23
- des noms de fichiers de sauvegarde des résultats
INITIALISATION de plusieurs paramètres et mise à 0 de tous les gains du (des) régulateur(s) sauf de K11 et/ou de K21
Réglage de K1 et K2
EXPÉRIENCE
RECHERCHE DES OSCILLATONS
sur le niveau et/ou la température
arrêt = 1 si oscillations dès la première expérience
NEWGAIN.M
pour la réponse en niveau puis/ou
pour la réponse en température
Régime(s) STATIONNAIRE(S) atteint(s)?
NON
Augmentation de la durée de l'expérience
OUI
SAUVEGARDE ET AFFICHAGE
de la (des) réponse(s) temporelle(s) et de ses (leurs) caractéristiques
(approximation(s) du second degré, dépassement(s))
REMISE À ZÉRO
du système matériel
REMISE À JOUR
de l'affichage de la (des) réponse(s)
CRITÈRES sur le(s) dépassement(s) et sur les oscillations SATISFAITS?
NON
MODIFICATIONS DES GAINS
K11, K12 et/ou K21, K22
selon les valeurs données par
Newgain.m
OUI
1


1
SAUVEGARDE des données sur disque dur et
NETTOYAGE de la mémoire
ARRÊT = 0?
NON
FIN
Réglage de K3
OUI
AFFECTATION à A11, A12 et/ou A21, A22
des gains donnés par Newgain.m
INITIALISATION de K3
K13 = inc K13 et/ou K23 = inc K23
EXPÉRIENCE
CALCUL DU (DES) DÉPASSEMENT(S) RELATIF(S)
ET RECHERCHE DES OSCILLATIONS
SAUVEGARDE ET AFFICHAGE
de la (des) réponse(s) temporelle(s)
K13 = K13 + inc K13
et/ou
K23 = K23 + inc K23
REMISE À ZÉRO
du système matériel
REMISE À JOUR
de l'affichage de la (des)
réponse(s)
CRITÈRE(S) sur le(s)
dépassement(s) et
sur les oscillations
SATISFAIT(S)?
OUI
NON
2


2
K13 = K13 - inc K13
K23 = K23 - inc K23
RÉGLAGE D'UNE SEULE BOUCLE?
OUI
AFFICHAGE DU JEU DE GAINS OBTENU
FIN
NON
K13 = K13 + inc K13
EXPÉRIENCE
CALCUL DU (DES) DÉPASSEMENT(S) RELATIF(S) ET RECHERCHE DES OSCILLATIONS
SAUVEGARDE ET AFFICHAGE de la (des) réponse(s) temporelle(s)
REMISE À ZÉRO du système matériel
REMISE À JOUR de l'affichage de la (des) réponse(s)
CRITÈRE(S) sur le(s) dépassement(s) et sur les oscillations SATISFAIT(S)?
OUI
NON
K13 = K13 - inc K13
3


3
K23 = K23 + inc K23
EXPÉRIENCE
CALCUL DU (DES) DÉPASSEMENT(S) RELATIF(S) ET RECHERCHE DES OSCILLATIONS
SAUVEGARDE ET AFFICHAGE de la (des) réponse(s) temporelle(s)
REMISE À ZÉRO du système matériel
REMISE À JOUR de l'affichage de la (des) réponse(s)
CRITÈRE(S) sur le(s) dépassement(s) et sur les oscillations SATISFAIT(S)?
OUI
NON
Chargement du fichier correspondant au réglage de K1 et K2
AFFICHAGE DU JEU DE GAINS OBTENU ET DE L'ENSEMBLE DES COURBES
SAUVEGARDE SUR DISQUE DUR
FIN



Organigramme du programme newgain.m

Annexe 2 :

Listings des programmes permettant l'auto-tuning

```
%*************************************************************
% Ce programme écrit avec Matlab 4.2c permet de déterminer
% automatiquement (auto-tuning) les gains du (des) régulateurs
% Dual Loop du réservoir en fonction des critères sur la réponse
% désirés par l'utilisateur (temps et amplitude du premier
% dépassement). Ce programme a été utilisé avec un pentium 166
% muni de 32 Mo de RAM et une carte d'entrées/sorties Labmaster
% fait appel aux sous-programmes:
% -pintauto.m (stratégie de contrôle),
% -newgain.m (identification de la réponse)
% -raz.m (remise à zéro)
% -rech_osc.m (recherche d'oscillations).
%*************************************************************


fprintf('*****************************************************\n');
fprintf('   Réglage automatique des gains du PID Dual Loop.  \n');
fprintf('      Asservissement de niveau et température.      \n');
fprintf('         Version: 12/06/97. par: CB/EC/RMDS         \n');
fprintf('*****************************************************\n');

clear

%=================================================================
%                      INTERFACE UTILISATEUR
%=================================================================

fprintf('Réglage des gains de la boucle de: \n');
fprintf('1.-Niveau \n');
fprintf('2.-Température \n');
fprintf('3.-Niveau et Température \n');
F=input('');

fprintf('\n Retard sur la commande? \n');
fprintf('0.-non \n');
fprintf('1.-oui \n');
ret=input('');

if ret==1,
        fprintf('\n Valeur du retard (en s) ? \n');
        retard=2*input('');
end;

if ((F==1)|(F==3)),
        fprintf('\n ************************************ \n');
        fprintf(' NIVEAU:\n');
        fprintf('\nValeur initiale de K11?\n');
        K11=input('');
        fprintf('\n Instant de 1er dépassement (s)? \n');
        instn=input('');
        fprintf('\n Amplitude relative du 1er dépassement:\n');
        fprintf('-après le réglage de k1 et k2 ?:\n');
```

```
        ampln12=input('');
        fprintf('\n-après le réglage de k3?:\n');
        ampln3=input('');
        fprintf('\n Incrément sur K13? \n');
        incK13=input('');
end;

if ((F==2)|(F==3)),
        fprintf('\n ************************************ \n');
        fprintf(' TEMPÉRATURE:\n');
        fprintf(' \n Instant de 1er dépassement (s)? \n');
        instt=input('');
        fprintf('\nValeur initiale de K21?\n');
        K21=input('');
        fprintf(' \n Amplitude relative du 1er dépassement:\n');
        fprintf('-après le réglage de k1 et k2 ?:\n');
        amplt12=input('');
        fprintf('\n-après le réglage de k3?:\n');
        amplt3=input('');
        fprintf('\n Incrément sur K23? \n');
        incK23=input('');
end;

fprintf('\nnom des fichiers de sauvegarde:\n');
fprintf('(exemple: ''c:\\matlab\\reserve\\toto'')\n');
fprintf('\n-pour la première phase du réglage?\n');
nom0=input('');
fprintf('\n-pour la seconde phase du réglage?\n');
nom=input('');
fprintf('\nQuand vous êtes prêt\n');
fprintf('et que le système est stationnaire, tapez return\n');
keyboard
fprintf('================================================\n');

%=================================================================
%                          INITIALISATION
%=================================================================

z=1;              % numéro de l'expérience
retour=0;         % utilisé dans le sous programme RAZ.m
w1=0;             % offsets sur les commandes des vannes
w2=0;

if F==1,
        K12 = 0;
        K13 = 0;
        K14 = 0;
        A11 = 0;
        A12 = 0;
        K21 = 0;
        K22 = 0;
        K23 = 0;
```

```
        K24 = 0;
        A21 = 0;
        A22 = 0;

        tfpan=1000;
        instt=1;
        tfpat=0;
        amplt12=1000;
        amplt3=1000;
end

if F==2,
        K11 = 0;
        K12 = 0;
        K13 = 0;
        K14 = 0;
        A11 = 0;
        A12 = 0;
        K22 = 0;
        K23 = 0;
        K24 = 0;
        A21 = 0;
        A22 = 0;

        tfpat=1000;
        instn=1;
        tfpan=0;
        ampln12=1000;
        ampln3=1000;
end

if F==3,
        K12 = 0;
        K13 = 0;
        K14 = 0;
        A11 = 0;
        A12 = 0;
        K22 = 0;
        K23 = 0;
        K24 = 0;
        A21 = 0;
        A22 = 0;

        tfpan=1000;
        tfpat=1000;

end;

%================================================================
%            réglage de k1, k2, alpha1 et alpha2
%================================================================
```

```
overshootn=100;
overshoott=100;
osciln=0;
oscilt=0;

while (((tfpan>instn)|(tfpat>instt)|(overshootn>ampln12)|(overshoott>am
plt12))&(osciln==0)&(oscilt==0)),

        nb_pts=1500;
        if F==3
                nb_pts=2000;
        end
        ref1=2;
        ref2=5;

        fprintf('\nEXPÉRIENCE %g.\n',z)

        if ((F==1)|(F==3))
                gain1n(z)=K11;
                gain2n(z)=K12;
                fprintf('K11= %g.\t',K11);
                fprintf('K12= %g.\n',K12);
        end

        if ((F==2)|(F==3))
                gain1t(z)=K21;
                gain2t(z)=K22;
                fprintf('K21= %g.\t',K21);
                fprintf('K22= %g.\n',K22);
        end

        pintauto;
%-----------------------------------------------------------------
%Recherche des oscillations
%-----------------------------------------------------------------
        if ((F==1)|(F==3)),
                pos=pv1;
                rech_osc;
                osciln=oscil;
                if osciln==1,
                        fprintf('sur le niveau\n');
                end;
        end;

        if ((F==2)|(F==3)),
                pos=pv2-pv2(1);
                rech_osc;
                oscilt=oscil;
                if oscil==1,
                        fprintf('sur la température\n');
                end;
        end;
```

```
%-----------------------------------------------------------------

        arret=0;
        if (((osciln==1)|(oscilt==1))&(z==1)),
                fprintf('Oscillations dès la 1ère expérience\n');
                fprintf('donc arrêt du programme après la RAZ\n');
                fprintf('Essayez de nouveaux gains initiaux\n');
                arret=1;
        end

%-----------------------------------------------------------------

        if ((osciln==0)&(oscilt==0)),
                tech=T;
%-----------------------------------------------------------------
% BOUCLE DE NIVEAU
%-----------------------------------------------------------------

                if ((F==1)|(F==3)),

                        k10n=K11;% gains utilisés pour le réglage de k3
                        k20n=K12;

                        pos=pv1;% initialisation des paramètres
                        k1=K11; % pour utiliser le
                        k2=K12; % sous-programme newgain.m
                        Fs=0;
                        if F==3,
                                Fs=3;
                                F=1;
                        end;

                        newgain;

%-----------------------------------------------------------------
% on refait l'expérience sur une plus longue durée
%si le régime stationnaire n'a pas été atteint
%-----------------------------------------------------------------

                        nbre=nb_pts;
                        nbre1=nb_pts;
                        while srnawrt==1;
                                fprintf('expérience niveau trop courte'
);
                                raz;
                                nbre=nbre+500;
                                nb_pts=nbre;
                                pintauto;
                                pos=pv1;
                                newgain
                        end
```

```
                        nb_pts=nbre1;

%-------------------------------------------------------------------
% on redonne à F sa valeur initiale
%-------------------------------------------------------------------
                        if Fs==3,
                                F=Fs;
                                Fs=0;
                        end;
%-------------------------------------------------------------------
%Sauvegarde et affichage des données
%-------------------------------------------------------------------
                        k1n=k1;          % on sauvegarde les paramètres
                        k2n=k2;          % concernant le niveau
                        alfa_10n=alfa_10;
                        alfa_20n=alfa_20;

                        fprintf('\nIdentification de la réponse en nive
au \n');

                        a1n(z)=alfa_10n;
                        a2n(z)=alfa_20n;
                        depassementn(z)=overshoot;
                        tpdn(z)=tfpa;

                        overshootn=overshoot;
                        tfpan=tfpa;
                        fprintf('A11= %g.\t',alfa_10n);
                        fprintf('A12= %g.\n',alfa_20n);
                        fprintf('depassement= %g.\n',overshootn);
                        fprintf('Tps de 1er dépassement= %g.\n',tfpan);
                end;

%-------------------------------------------------------------------
% BOUCLE DE TEMPÉRATURE
%-------------------------------------------------------------------

                if ((F==2)|(F==3)),


                        k10t=K21;% gains utilisés pour le réglage de k3
                        k20t=K22;

                        pos=pv2-pv2(1); % initialisation des paramètres
                        k1=K21;          % pour utiliser
                        k2=K22;          % le sous-programme newgain.m
                        Fs=0;
                        if F==3,
                                Fs=3;
                                F=2;
                        end;
```

```
                        newgain;
%-------------------------------------------------------------------
% on refait l'expérience sur une plus longue durée
% si le régime stationnaire n'a pas été atteint
%-------------------------------------------------------------------

                        nbre=nb_pts;
                        nbre1=nb_pts;
                        while srnawrt==1;
                                fprintf('expérience température trop co
urte');
                                raz;
                                nbre=nbre+500;
                                nb_pts=nbre;
                                pintauto;
                                pos=pv2-pv2(1);
                                newgain
                        end
                        nb_pts=nbre1;

%-------------------------------------------------------------------
% on redonne à F sa valeur initiale
%-------------------------------------------------------------------
                        if Fs==3,
                                F=Fs;
                                Fs=0;
                        end;
%-------------------------------------------------------------------
% Sauvegarde et affichage des données
%-------------------------------------------------------------------
                        k1t=k1; % on sauvegarde les paramètres
                        k2t=k2; % concernant la température
                        alfa_10t=alfa_10;
                        alfa_20t=alfa_20;

                        fprintf('\nIdentification de la réponse en temp
érature \n');
                        a1t(z)=alfa_10t;
                        a2t(z)=alfa_20t;
                        depassementt(z)=overshoot;
                        tpdt(z)=tfpa;

                        overshoott=overshoot;
                        tfpat=tfpa;

                        fprintf('A21= %g.\t',alfa_10t);
                        fprintf('A22= %g.\n',alfa_20t);
                        fprintf('dépassement= %g.\n',overshoott);
                        fprintf('Tps de 1er dépassement= %g.\n',tfpat);
                end;
        end;
```

```
%=================================================================
%               AFFICHAGE DE LA OU DES RÉPONSES
%=================================================================
        if F==1,
                subplot(224),plot(pv1); grid
                subplot(211),plot(pv11); grid
        end;

        if F==2,
                subplot(224),plot(pv2); grid
                subplot(211),plot(pv21); grid
        end;

        if F==3,
                subplot(211),plot(pv11); grid
                subplot(212),plot(pv21); grid
        end;

        pause(5);

        raz;    % Remise à zéro

        if F==1,
                subplot(211),plot(pv11); grid
        end;

        if F==2,
                subplot(211),plot(pv21); grid
        end;

        if F==3,
                subplot(211),plot(pv11); grid
                subplot(212),plot(pv21); grid
        end;

        pause(5);

%=================================================================
%                    PASSAGE AU PAS SUIVANT
%=================================================================
        if ((F==1)|(F==3)),
                K11=k1n;
                K12=k2n;
        end;

        if ((F==2)|(F==3)),
                K21=k1t;
                K22=k2t;
        end;

        z=z+1;
```

```
end;

%============================================================
%    Stockage des données et nettoyage de la mémoire
%============================================================

eval(['save ' nom0 ' pv11 pv21 comm1 comm2']);

clear pv11;
clear pv21;
clear comm1;
clear comm2;
pack;
pause(5);

z0=z-1;
z=1;

%============================================================
%                        réglage de k3
%============================================================
if arret==0
        fprintf('\n=================================================\
n');
        fprintf('RÉGLAGE DE k3');

        if ((F==1)|(F==3))
                K11=k10n;
                K12=k20n;
                A11=alfa_10n;
                A12=alfa_20n;
        end

        if ((F==2)|(F==3)),
                K21=k10t;
                K22=k20t;
                A21=alfa_10t;
                A22=alfa_20t;
        end

        Fr=F;

        reglak3;

        if ((F==1)|(F==3)),
                K13=K13-incK13;
        end;

        if ((F==2)|(F==3)),
                K23=K23-incK23;
        end;
```

```
        if (F==3),
                Fr=1;
                reglak3;
                K13=K13-incK13;

                Fr=2;
                reglak3;
                K23=K23-incK23;
        end;

%================================================================
%                  AFFICHAGE DES RÉSULTATS
%================================================================

        pv111=pv11;
        pv121=pv21;
        comm11=comm1;
        comm21=comm2;

        eval(['load ' nom0]);

        pv11=[pv11 pv111];      % Concaténation des réponses obtenues
        pv21=[pv21 pv121];      % lors des 2 phases de réglage
        comm1=[comm1 comm11];
        comm2=[comm2 comm21];

        temps1=0.5*[1:1:max(size(pv11))];
        subplot(411),plot(temps1,pv11);
        axis([1,0.5*max(size(pv11)),floor(min(pv11)),floor(max(pv11))+1
]);
        ylabel('niveau (V)');
        grid;

        temps2=0.5*[1:1:max(size(pv21))];
        subplot(412),plot(temps2,pv21);
        axis([1,0.5*max(size(pv21)),floor(min(pv21)),floor(max(pv21))+1
]);
        ylabel('température (V)');
        grid;

        temps3=0.5*[1:1:max(size(comm1))];
        subplot(413),plot(temps3,comm1);
        axis([1,0.5*max(size(comm1)),floor(min(comm1)),floor(max(comm1)
)+1]);
        ylabel('u1 (V)');
        grid;

        temps4=0.5*[1:1:max(size(comm2))];
        subplot(414),plot(temps4,comm2);
        axis([1,0.5*max(size(comm2)),floor(min(comm2)),floor(max(comm2)
)+1]);
        xlabel('temps (s)');
```

```
        ylabel('u2 (V)');
        grid;

        fprintf('\n Les gains trouvés sont:\n');
        fprintf('K11= %g.\n',K11);
        fprintf('K12= %g.\n',K12);
        fprintf('K13= %g.\n',K13);
        fprintf('A11= %g.\n',A11);
        fprintf('A12= %g.\n',A12);
        fprintf('K21= %g.\n',K21);
        fprintf('K22= %g.\n',K22);
        fprintf('K23= %g.\n',K23);
        fprintf('A21= %g.\n',A21);
        fprintf('A22= %g.\n',A22);

%================================================================
%                  SAUVEGARDE DES RÉSULTATS
%================================================================
        if arret==1
                eval(['save ' nom ' F pv11 pv21 comm1 comm2 K11 K12 K13
 A11 A12 K21 K22 K23 A21 A22 gain1n gain2n']);
        else
                if z==1
                        depassement3n=0;
                        depassement3t=0;
                end
                if F==1
                        eval(['save ' nom ' F pv11 pv21 comm1 comm2 K11
 K12 K13 A11 A12 K21 K22 K23 A21 A22 gain1n gain2n a1n a2n tpdn depasse
mentn depassement3n']);
                end
                if F==2
                        eval(['save ' nom ' F pv11 pv21 comm1 comm2 K11
 K12 K13 A11 A12 K21 K22 K23 A21 A22 gain1t gain2t a1t a2t tpdt depasse
mentt depassement3t']);
                end
                if F==3
                        eval(['save ' nom ' F pv11 pv21 comm1 comm2 K11
 K12 K13 A11 A12 K21 K22 K23 A21 A22 alfa_10 alfa_20 gain1n gain2n a1n
a2n gain1t gain2t a1t a2t tpdn tpdt depassementn depassementt depasseme
nt3n depassement3t']);
                end
        end
end

ecr_na (0,0);   % remise à zéro des commandes
ecr_na (1,0);
```

```
%=================================================================
% SOUS-PROGRAMME DU PROGRAMME PINTPRI2.m CONSTITUANT LE RÉGULATEUR
%                    PID DUAL LOOP NUMÉRIQUE
% Ce programme permet l'acquisition et l'écriture de données avec
% une carte Labmaster
% Version : 27 mai 1997 par CB/EC/RMDS
%=================================================================

ret1=ret;
if (retour==1),
        ret=0;
        fprintf('\n REMISE À ZERO \n');
end;

T = 0.5;

alpha1 = 1.723;
alpha11=1/(1+alpha1);
alpha12=alpha1*alpha11;

alpha21=0.36;
alpha22=-0.64;

pv1 = zeros(1,nb_pts);
pv2 = zeros(1,nb_pts);
u1 = zeros(1,nb_pts);
u2 = zeros(1,nb_pts);

pvv1 =zeros(1,nb_pts);
pvv2 =zeros(1,nb_pts);

err1 = zeros(1,nb_pts);
err2 = zeros(1,nb_pts);

int_err1= zeros(1,nb_pts);
int_err2= zeros(1,nb_pts);

der1=zeros(1,nb_pts);
der2=zeros(1,nb_pts);

U1 = zeros(1,nb_pts);
U2 = zeros(1,nb_pts);

fprintf('\n SOYEZ PATIENT!\n');
est=0;
fprintf('0---1---2---3---4---5---6---7---8---9---10\n');

init_lab;       % INITIALISATION CONVERTISSEUR


dep_horl;       % DÉPART HORLOGE
```

```
pv1(1)=lire_an(7);
ref1=ref1-pv1(1);
consigne1 = ref1*ones(1,nb_pts);

pv3(1)=lire_an(6);
pv2(1)=pv3(1);
ref2=ref2-pv2(1);
consigne2 = ref2*ones(1,nb_pts);

att_horl (1000*T);      % temporisation de T seconde

%=====================================================================
%                        BOUCLE DE RÉGULATION
%=====================================================================

for i=2:nb_pts,

        dep_horl;

        if (100*i/nb_pts)>(est+2.45),    % Indication graphique de
                est=est+2.5;             % l'avancement de l'expérience
                fprintf('*');
        end;

        pv1(i) = lire_an(7);     % ACQUISITION DES DONNÉES DE LA CARTE
        pv3(i) = lire_an(6);

%niveau
        pvv1(i)=pv1(i)-pv1(1);
        err1(i)= consigne1(1,i)-pvv1(i);
        int_err1(i)=int_err1(i-1)+err1(i)*T;
        der1(i)=(pvv1(i)-pvv1(i-1))/T;
        U1(i)=K11*err1(i)+K13*(int_err1(i)-A11*pvv1(i)-A12*der1(i))+K14
*ref1-K12*der1(i);

%temperature
        pv2(i)=(T*pv3(i)+20*pv2(i-1))/(T+20);

        pvv2(i)=pv2(i)-pv2(1);
        err2(i)=consigne2(1,i)-pvv2(i);
        der2(i)=(pvv2(i)-pvv2(i-1))/T;
        int_err2(i)=int_err2(i-1)+err2(i)*T;
        U2(i)=K21*err2(i)+K23*(int_err2(i)-A21*pvv2(i)-A22*der2(i))+K24
*ref2-K22*der2(i);

%=====================================================================
        if F==1,
                if ret==1,
                        if i<=retard,
                                UU1(i)=0;
                        elseif i>retard,
                                UU1(i)=U1(i-retard);
```

```
                end
                u11(i)=alpha11*UU1(i);
                u12(i)=alpha12*UU1(i);
                u21(i)=0;
                u22(i)=0;
        end

        if ret==0,
                u11(i)=alpha11*U1(i);
                u12(i)=alpha12*U1(i);
                u21(i)=0;
                u22(i)=0;
        end
end

%=================================================================

if F==2
        if ret==1,
                if i<=retard,
                        UU2(i)=0;
                elseif i>retard,
                        UU2(i)=U2(i-retard);
                end
                u21(i)=alpha21*UU2(i);
                u22(i)=alpha22*UU2(i);
                u12(i)=0;
                u11(i)=0;
        end


        if ret==0,
                u21(i)=alpha21*U2(i);
                u22(i)=alpha22*U2(i);
                u12(i)=0;
                u11(i)=0;
        end
end

%=================================================================

if F==3,
        if ret==1,
                if i<=retard,
                        UU1(i)=0;
                        UU2(i)=0;
                elseif i>retard,
                        UU1(i)=U1(i-retard);
                        UU2(i)=U2(i-retard);
                end
                u11(i)=alpha11*UU1(i);
                u12(i)=alpha12*UU1(i);
```

```
                        u21(i)=alpha21*UU2(i);
                        u22(i)=alpha22*UU2(i);
                end

                if ret==0,
                        u11(i)=alpha11*U1(i);
                        u12(i)=alpha12*U1(i);
                        u21(i)=alpha21*U2(i);
                        u22(i)=alpha22*U2(i);
                end
        end

%=====================================================================
% commandes avec prédécouplage

        u1(i) = u11(i) + u21(i);
        u2(i) = u12(i) + u22(i);

        ecr_na (0, u1(i)+w1);    % ENVOI DES COMMANDES A LA CARTE
        ecr_na (1, u2(i)+w2);

        att_horl (1000 * T);

end;

%=====================================================================
%                     SAUVEGARDE DES DONNÉES
%=====================================================================

if ((z==1)&(retour~=1))
        pv11(1:max(size(pv1)))=pv1;
        pv21(1:max(size(pv2)))=pv2;
        comm1(1:max(size(u1)))=u1;
        comm2(1:max(size(u2)))=u2;
else
        pv11(max(size(pv11)+1):(max(size(pv11))+max(size(pv1))))=pv1;
        pv21(max(size(pv21)+1):(max(size(pv21))+max(size(pv2))))=pv2;
        comm1(max(size(comm1)+1):(max(size(comm1))+max(size(u1))))=u1;
        comm2(max(size(comm2)+1):(max(size(comm2))+max(size(u2))))=u2;
end

fprintf(' \n');

ret=ret1;
```

```
% ======================================================
% SUB PROGRAM OF PINTPRI2.m
% which computes from current PD gains
% and associated results, a new set of
% alfa_1, alfa_2 and  k1, k2 gains that are susceptible
% to improve system response.
% Version: 27 may 1997 by RMDS/CB/EC
%======================================================


N=max(size(pos)); % (N=number of response samples)
tt=1:N;           % (tt=time scale)
T=tech;           % (T= sampling period)
N1=.7*N;
N2=.9*N;

mx_1=max(pos(N1:N2));
mn=min(pos(N1:N2));
srnawrt=0;
if (mx_1-mn)/(mx_1+mn)>.08,
        input('stationary regime not achieved within required time');
        srnawrt=1;
end;
                  % verification that stationary regime has indeed
                  % be attained N1*T,N2*T interval

if srnawrt==0
        stat_v=mean(pos(N1:N2));
        mx_2=max(pos(3:N1));    % (peak value of response)
        overshoot=(mx_2-stat_v)/stat_v;

% ========================================================
% = We compute values of alfa_1, alfa_2, and wn,
% = associated with current test results. We do this by
% = evaluating the best second order approximant
% ========================================================
%  (uses ordre_2.m)
% ====================================
% procedure: 1: rough evaluation of alfa_1, alfa_2, ksi, wn
% =========     from collected data
%            2: use of rough values of alfa_1, alfa_2
%               to determine alfa_1, alfa_2 of a 2nd order system
%               with same overshoot and same time_of_first_peak
%               as the test_bench system


        if overshoot<=.05
                ksi=1;
                i=1;

                while pos(i)<=.9*stat_v,
                        i=i+1;
```

```
                end;

                trn=i*T;
                wn=4/trn;
                alfa_1=2*ksi/wn;
                alfa_2=1/wn^2;
                tfpa=trn;
                ordre_2;

        else
                temp=-log(overshoot)/pi;
                ksi=sqrt(temp^2/(1+temp^2));
                i=1;

                while pos(i)~=mx_2,
                        i=i+1;
                end;

                tfp=i*T;                % (time first peak)
                wn=pi/(tfp*sqrt(1-ksi^2));
                alfa_1=2*ksi/wn;
                alfa_2=1/wn^2;

                ordre_2;

                i=1;
                mx_3=max(y_alfa(1:N));

                while (mx_3-y_alfa(i))>0.000001,
                        i=i+1;
                end;

                tfpa=i*T;               % (time first peak)

                wn=(tfpa/tfp)*wn;
                alfa_1=2*ksi/wn;
                alfa_2=1/wn^2;
        end;

% ======================================================
% = From current values of k1, k2 and alfa_1, alfa_2,
% = and wn, we compute new values of k1, k2. These values
% = are computed so as to have ksi=ksid and wn = mu*wn0.
% ======================================================

% Old values

        k10=k1;
        k20=k2;
        alfa_10=alfa_1;
        alfa_20=alfa_2;
        wn0=wn;
```

```
        ksi0=ksi;

% New values

        if F==1,
                inst=instn;
                ampl=ampln12;
        end;
        if F==2,
                inst=instt;
                ampl=amplt12;
        end;

        mu=1.1;
        if(tfpa<inst),
                mu=1;
        end;

        muk=1.2;
        if(overshoot<ampl),
                muk=1;
        end;

        wn=mu*wn0;
        ksid=muk*ksi0;

        alfa_1=2*ksid/wn;
        alfa_2=1/wn^2;

        k1 = k10*alfa_20/alfa_2;
        k2 = k20 + k1*alfa_1-k10*alfa_10;
end
```

```
%=====================================================================
%  PROGRAMME POUR DÉTECTER LA PRESENCE ÉVENTUELLE D'OSCILLATIONS
%                     DANS LE VECTEUR pos (variable du procédé)
% Version 12 juin 1997 par CB/EC/RMDS
%=====================================================================


%=====================================================================
%                            INITIALISATION
%=====================================================================

N=max(size(pos));
i=30;
maxloc=0;
oscil=0;

%=====================================================================
%           RECHERCHE DU PREMIER MAXIMUM LOCAL SIGNIFICATIF
%=====================================================================

while ((maxloc==0)&(i<N-30)),
        i=i+1;
        if ((pos(i+1)<=pos(i))&(pos(i-1)<=pos(i))),
                maxloc=1;
                minimum=min(pos(i-30:i+30));
                maximum=max(pos(i-30:i+30));
                dif=maximum-minimum;

                if dif<0.1,
                        maxloc=0;
                end;

                if maxloc==1,
                        for m=-30:30,
                                if pos(i+m)>pos(i),
                                        maxloc=0;
                                end
                        end
                end;
        %keyboard
        end;
end;
%=====================================================================
%  RECHERCHE DES OSCILLATIONS ÉVENTUELLES APRÈS CE MAXIMUM LOCAL
%=====================================================================

if maxloc==1,                   % recherche du minimum global
        mini=min(pos(i:N));     % entre l'abscisse du 1er maximum
        k=i;                    % local retenu et la fin de l'expérienc
e
        while (pos(k)-mini)>0.000001,
                k=k+1;
```

```
        end;

        moy=mean(pos(k:N));      % recherche du maximum global entre

        maxi=max(pos(k:N));      % l'abscisse du minimum global retenu e
t la fin de l'expérience
        l=k;
        while (maxi-pos(l))>0.000001,
                l=l+1;
        end;

        ondul=(maxi-mini)/moy;
        oscil=0;
        if ((ondul>0.05)&(l-k)<100),
                oscil=1;
                fprintf('\nPrésence d''oscillations\t');
        end;
end;
```

```
%=====================================================================
%              SOUS-PROGRAMME DU PROGRAMME PINTPRI2.m
%       PERMETTANT LA REMISE A ZERO AUTOMATIQUE DU RÉSERVOIR SELON LA
%       BOUCLE DE REGULATION.
% Version: 15 juin 1997 par CB/EC/RMDS
%=====================================================================

%=====================================================================
%RAZ du niveau seul si F=1
%RAZ du niveau ET de la temperature si F=2 ou F=3
%=====================================================================

clear pv1;
clear pv2;
clear u1;
clear u2;

retour=1;
Fb=0;
nb_pts=2000;
ref1=0;
ref2=2;

sauv1n=K11;
K11=1;
sauv2n=K12;
K12=0;
sauv3n=K13;
K13=0;

if F==2,
        Fb=F;
        F=3;
end;
if F==3,
        sauv1t=K21;
        K21=1;
        sauv2t=K22;
        K22=0;
        sauv3t=K23;
        K23=0;

        fprintf('1ère partie de RAZ, soyez patient\n');
        init_lab;                       %fermeture totale des vannes
        i=1;                            %pour d'abord faire descendre l
e niveau
        dep_horl;
        pv1(1)=lire_an(7);
        pv3(1)=lire_an(6);
        pv2(1)=pv3(1);
        while pv1(i)>-9
                i=i+1;
```

```
                dep_horl;
                pv1(i)=lire_an(7);
                pv3(i)=lire_an(6);
                pv2(i)=(T*pv3(i)+20*pv2(i-1))/(T+20);
                ecr_na(0,-10);
                ecr_na(1,-10);
                u1(i)=-10;
                u2(i)=-10;
                att_horl (1000*T);
        end
        i0=i;
        pv11(max(size(pv11)+1):(max(size(pv11))+max(size(pv1))))=pv1;
        pv21(max(size(pv21)+1):(max(size(pv21))+max(size(pv2))))=pv2;
        comm1(max(size(comm1)+1):(max(size(comm1))+max(size(u1))))=u1;
        comm2(max(size(comm2)+1):(max(size(comm2))+max(size(u2))))=u2;
        clear pv1;
        clear pv2;
        clear u1;
        clear u2;
        init_lab;                       %ouverture de la vanne d'eau fr
oide
        i=1;                            %pour ramener la température à
2
        dep_horl;
        pv1(1)=lire_an(7);
        pv3(1)=lire_an(6);
        pv2(1)=pv3(1);
        while pv2(i)>2.3
                i=i+1;
                dep_horl;
                pv1(i)=lire_an(7);
                pv3(i)=lire_an(6);
                pv2(i)=(T*pv3(i)+20*pv2(i-1))/(T+20);
                ecr_na(0,-10);
                ecr_na(1,5);
                u1(i)=-10;
                u2(i)=5;
                att_horl (1000*T);
        end
        i1=i;
        pv11(max(size(pv11)+1):(max(size(pv11))+max(size(pv1))))=pv1;
        pv21(max(size(pv21)+1):(max(size(pv21))+max(size(pv2))))=pv2;
        comm1(max(size(comm1)+1):(max(size(comm1))+max(size(u1))))=u1;
        comm2(max(size(comm2)+1):(max(size(comm2))+max(size(u2))))=u2;
        fprintf('2ème partie de RAZ\n');
        nb_pts=2500-(i0+i1);
end


%*****************************************************************
```

```
pintauto;

%*********************************************************************

K11=sauv1n;
K12=sauv2n;
K13=sauv3n;
if (F==3),
        K21=sauv1t;
        K22=sauv2t;
        K23=sauv3t;
end;

if Fb==2,
        F=Fb;
end;

w1=u1(nb_pts)+w1;        % Détermination des nouveaux offsets
w2=u2(nb_pts)+w2;        % sur les commandes des vannes

retour=0;

clear pv1;
clear pv2;
clear u1;
clear u2;
```

```
%================================================================
%               SOUS-PROGRAMME DU PROGRAMME PINPRI2.m
%            INTERVENANT SANS LA PHASE DE RÉGLAGE DE k3
% VERSION:1 JUIN 1997 PAR CB/EC/RMDS
%================================================================
overshootn=0;
overshoott=0;

while ((overshootn<ampln3)&(overshoott<amplt3)),

        if ((Fr==1)|(Fr==3))
                K13=K13+incK13;
        end

        if ((Fr==2)|(Fr==3))
                K23=K23+incK23;
        end

        ref1=2;
        ref2=5;
        nb_pts=1500;
        if F==3
                nb_pts=2000;
        end

        fprintf('\nEXPÉRIENCE %g.\n',z+z0)
        fprintf ('K13= %g.\t', K13);
        fprintf ('K23= %g.\n', K23);

        pintauto

        if ((F==1)|(F==3)),

                pos=pv1;
                rech_osc;
                osciln=oscil;

                overshootn=ampln3+1;
                if osciln==1,
                        fprintf('sur le niveau\n');
                else
                        maxin=max(pv1);
                        statn=mean(pv1(0.7*max(size(pv1)):0.9*max(size(
pv1))));
                        overshootn=(maxin-statn)/statn;
                        depassement3n(z)=overshootn;
                        fprintf('\ndépassement niveau= %g.\n',overshoot
n);
                end;
        end;

        if ((F==2)|(F==3)),
```

```
                pos=pv2-pv2(1);
                rech_osc;
                oscilt=oscil;

                overshoott=amplt3+1;
                if oscilt==1,
                        fprintf('sur la température\n');
                else
                        maxit=max(pv2);
                        statt=mean(pv2(0.7*max(size(pv2)):0.9*max(size(
pv2))));
                        overshoott=(maxit-statt)/(statt-pv2(1));
                        depassement3t(z)=overshoott;
                        fprintf('\ndépassement température= %g.\n',over
shoott);
                end;
        end;

        if F==1,
                subplot(224),plot(pv1);grid;
                subplot(211),plot(pv11); grid;
        end;

        if F==2,
                subplot(224),plot(pv2);grid;
                subplot(211),plot(pv21); grid;
        end;

        if F==3,
                subplot(211),plot(pv11); grid;
                subplot(212),plot(pv21); grid;
        end;

        pause(5);

        raz;

        if F==1,
                subplot(211),plot(pv11); grid;
        end;

        if F==2,
                subplot(211),plot(pv21); grid;
        end;

        if F==3,
                subplot(211),plot(pv11); grid;
                subplot(212),plot(pv21); grid;
        end;

        pause(5);
```

```
        z=z+1;
end;
```

Annexe 3 :

Listing du programme utilisé pour tester la robustesse de l'asservissement

```
% ********************************************************************
% Ce programme a été écrit avec MATLAB 4.2c. Il permet d'implanter un
% ou deux régulateurs PID Dual Loop pour l'asservissement en niveau
% et/ou en température de l'eau dans le réservoir du banc d'essai de
% la section automatique de l'École Polytechnique. Les données en
% entrée sont essentiellement les mesures du niveau et de la
% température fournies par une carte d'entrées/sorties Labmaster
% (canaux 7 et 6). Les sorties sont les commandes envoyées sur la vanne
% d'eau froide et la vanne d'eau chaude par l'intermédiaire de la même
% carte Labmaster (canaux 0 et 1). Les variables que l'utilisateur peut
% modifier sont :
% - F (indicateur de la ou des boucles à asservir)
% - les gains des régulateurs
% - la durée de l'expérience
% - la durée du retard introduit sur les commandes
% - la valeur de la perturbation ajoutée au milieu de l'expérience
%       - sur la vanne d'eau froide  pour tester la robustesse de
%       l'asservissement de niveau seul ou de l'ensemble asservissement
%       de niveau/asservissement de température
%       - sur la vamnne d'eau froide et sur la vanne d'eau chaude pour
%       tester la robustesse de l'asservissement de température seule
%*********************************************************************


fprintf('*********************************************************\n');

fprintf('                Compensateur PID Dual-Loop               \n');

fprintf('         Asservissement de niveau et température.        \n');

fprintf('Version: 10/06/97. par: EC/RMDS/CB                       \n');

fprintf('*********************************************************\n');


fprintf('*********************************************************\n')
fprintf('       Données par défaut:   \n ');
fprintf('*********************************************************\n')


% INITIALISATION

clear

    echo on

    nb_pts=2000;          % nombre de points de l'expérience
    T=0.5;                % temps d'échantillonnage

% Paramètres du régulateur PID Dual Loop
```

```
    K11 = 0.847;
    K12 = 12.1644;
    K13 = 0.006;
    K14 = 0;
    A11 = 37.25;
    A12 = 346.891;

    ref1 =2;
    K21 = 2.42;
    K22 = 34.905;
    K23 = 0.05;
    K24 = 0;
    A21 = 75.25;
    A22 = 1415.64;

    ref2=5;         % consigne desirée
    alpha1=1.723;   % rapport d'eau froide/eau chaude

    echo off

F=input('Asservissement de: 1.-Niveau 2.-Température 3.-Niveau et Tempé
rature=');
P=input(' Échelon de perturbation sur le niveau/température: 1.-oui 0.-
non=');
ret=input(' Retard sur le niveau/température, 1.-oui, 0.-nom=');

if ret==1,
   retard=input('valeur de retard*0.5 en secondes=');
end

% OPPORTUNITÉ DE CHANGEMENT

fprintf('************************************************************\n')
;
fprintf('  Si vous voulez changer les données, tapez maintenant \n ')
fprintf('  les variables desirées et tapez break, enter.\n')
fprintf('  Sinon pour lancer l"expérience, tapez break, enter.\n  ')

fprintf('************************************************************\n')

keyboard


%  Conditions initiales d'opération

alpha11=1/(1+alpha1);
alpha12=alpha1*alpha11;

alpha21=0.36;
alpha22=-0.64;

u11 = zeros(1,nb_pts);
```

```
u12 = zeros(1,nb_pts);
u21 =zeros(1,nb_pts);
u22 = zeros(1,nb_pts);
pv1 = zeros(1,nb_pts);
pv2 = zeros(1,nb_pts);
pvv2 =zeros(1,nb_pts);
err1 = zeros(1,nb_pts);
err2 = zeros(1,nb_pts);
int_err1= zeros(1,nb_pts);
der1=zeros(1,nb_pts);
der2=zeros(1,nb_pts);
int_err2= zeros(1,nb_pts);
U1 = zeros(1,nb_pts);
U2 = zeros(1,nb_pts);

if F==1,
   alpha21=0;
   alpha22=0;
end

if F==2,
   alpha11=0;
   alpha12=0;
end

fprintf('SOYEZ PATIENT!!!!\n');
est=0;
fprintf('---1---2---3---4---5---6---7---8---9---0\n');

        init_lab;% INITIALISATION CONVERTISSEUR

% BOUCLE DE REGULATION

pert=0;

dep_horl;
pv1(1)=lire_an(7);
ref1=ref1-pv1(1);
consigne1=ref1*ones(1,nb_pts);

pv3(1)=lire_an(6);
pv2(1)=pv3(1);
ref2=ref2-pv2(1);
consigne2=ref2*ones(1,nb_pts);

for i=2:nb_pts,

        dep_horl;

    if (100*i/nb_pts)>(est+2.45) est=est+2.5; fprintf('*');
    end
```

```
        pv1(i) = lire_an(7);
        pv3(i) = lire_an(6);

% PERTURBATION pert

        if P==1,
           if i>=nb_pts/2;
              pert=1;
           end
        end

% Commande U1, niveau

        pvv1(i)=pv1(i)-pv1(1);
        err1(i) =  consigne1(1,i)-pvv1(i);
        int_err1(i)=int_err1(i-1)+err1(i)*T;
        der1(i)=(pvv1(i)-pvv1(i-1))/T;
        U1(i)=K11*err1(i) + K13*(int_err1(i)-A11*pvv1(i)-A12*der1(i))+K
14*ref1-K12*der1(i);


% Filtre numérique pour la température:

        pv2(i)=(T*pv3(i)+20*pv2(i-1))/(T+20);

% commande U2, température:

        pvv2(i)=pv2(i)-pv2(1);
        err2(i) = consigne2(1,i)-pvv2(i);
        der2(i) = (pvv2(i)-pvv2(i-1))/T;
        int_err2(i) = int_err2(i-1)+err2(i)*T;
        U2(i)=K21*err2(i) + K23*(int_err2(i) - A21*pvv2(i)-A22*der2(i))
+K24*ref2-K22*der2(i);


% Conditions de retard, pour toutes les boucles:


        if F==1,
                if ret==1,
                        if i<=retard,
                                UU1(i)=0;
                        elseif i>retard,
                                UU1(i)=U1(i-retard);
                        end
                        u11(i)=alpha11*UU1(i);
                        u12(i)=alpha12*UU1(i)+pert;
                        u21(i)=0;
                        u22(i)=0;
                end

                if ret==0,
```

```
                u11(i)=alpha11*U1(i);
                u12(i)=alpha12*U1(i)+pert;
                u21(i)=0;
                u22(i)=0;
        end
end

if F==2
        if ret==1,
                if i<=retard,
                        UU2(i)=0;
                elseif i>retard,
                        UU2(i)=U2(i-retard);
                end
                u21(i)=alpha21*(UU2(i)+pert);
                u22(i)=alpha22*(UU2(i)+pert);
        end

        if ret==0,
                u21(i)=alpha21*(U2(i)+pert);
                u22(i)=alpha22*(U2(i)+pert);
                u12(i)=0;
                u11(i)=0;
        end
end

if F==3,
        if ret==1,
                if i<=retard,
                        UU1(i)=0;
                        UU2(i)=0;
                elseif i>retard,
                        UU1(i)=U1(i-retard);
                        UU2(i)=U2(i-retard);
                end
                u11(i)=alpha11*UU1(i);
                u12(i)=alpha12*UU1(i)+pert;
                u21(i)=alpha21*UU2(i);
                u22(i)=alpha22*UU2(i);
        end

        if ret==0,
                u11(i)=alpha11*U1(i);
                u12(i)=alpha12*U1(i)+pert;
                u21(i)=alpha21*U2(i);
                u22(i)=alpha22*U2(i);
        end
end

% commandes avec prédécouplage
```

```
        u1(i) = u11(i) + u21(i);
        u2(i) = u12(i) + u22(i);

% signaux de commandes envoyés aux interfaces pour chaque vanne

        ecr_na (0, u1(i));
        ecr_na (1, u2(i));

        att_horl (1000 * T);

end;

        ecr_na (0, 0);
        ecr_na (1, 0);

%  AFFICHAGE DE LA REPONSE SELON LE CHOIX DE L'ASSERVISSEMENT :

temps = [1:1:nb_pts] * T;

if F==1,
        subplot(211)
        plot(temps,pv1,'r')
        title('Comportement du niveau')
        xlabel('temps en secondes')
        ylabel('valeurs en volts')
        grid

        subplot(212)
        plot(temps, u1,'g', temps,u2,'y')
        title('Comportement de la commande u1 et u2')
        xlabel('temps en secondes')
        ylabel('valeurs en volts')
        grid
end

if F==2,
        subplot(211)
        plot(temps,pv2,'r')
        title('Comportement de la temperature')
        xlabel('temps en secondes')
        ylabel('valeurs en volts')
        grid

        subplot(212)
        plot(temps,u1,'g',temps,u2,'y')
        title('Comportement de la commande u1 et u2')
        xlabel('temps en secondes');
        ylabel('valeurs en volts')
        grid
end

if F==3,
```

```
        subplot(221)
        plot(temps,pv1,'r')
        title('Comportement du niveau')
        xlabel('temps en secondes')
        ylabel('valeurs en volts')
        grid
                subplot(222)
        plot(temps,u1,'y')
        title('commande vanne d"eau chaude')
        xlabel('temps en secondes')
        ylabel('valeurs en volts')
        grid
                subplot(223)
        plot(temps,pv2,'g')
        title('Comportement de la temperature')
        xlabel('Temps en secondes')
        ylabel('valeurs en volts')
        grid

        subplot(224)
        plot(temps,u2,'w')
        title('commande vanne d"eau froide')
        xlabel('Temps en secondes')
        ylabel('valeurs en volts')
        grid
end

% CHOIX DE SAUVERGADER LES DONNÉES DE L'EXPERIENCE

E=input('Vous voulez enregistrer les données(o/n):','s');
if E=='o',
        nom='a:\test1'
        keyboard
        eval(['save ' nom ' pv1 pv2 u11 u12 u21 u22 U1 U2 temps  K11 K12
 K13 K14 A11 A12 K21 K22 K23 K24 A21 A22 T ref1 ref2']);
end
```

```
        subplot(221)
        plot(temps,pv1,'r')
        title('Comportement du niveau')
        xlabel('temps en secondes')
        ylabel('valeurs en volts')
        grid
                subplot(222)
        plot(temps,u1,'y')
        title('commande vanne d"eau chaude')
        xlabel('temps en secondes')
        ylabel('valeurs en volts')
        grid
                subplot(223)
        plot(temps,pv2,'g')
        title('Comportement de la temperature')
        xlabel('Temps en secondes')
        ylabel('valeurs en volts')
        grid

        subplot(224)
        plot(temps,u2,'w')
        title('commande vanne d"eau froide')
        xlabel('Temps en secondes')
        ylabel('valeurs en volts')
        grid
end

% CHOIX DE SAUVERGADER LES DONNÉES DE L'EXPERIENCE

E=input('Vous voulez enregistrer les données(o/n):','s');
if E=='o',
        nom='a:\test1'
        keyboard
        eval(['save ' nom ' pv1 pv2 u11 u12 u21 u22 U1 U2 temps K11 K12
 K13 K14 A11 A12 K21 K22 K23 K24 A21 A22 T ref1 ref2']);
end
```

École Polytechnique de Montréal
C.P. 6079, Succ. Centre-ville
Montréal (Québec)
H3C 3A7