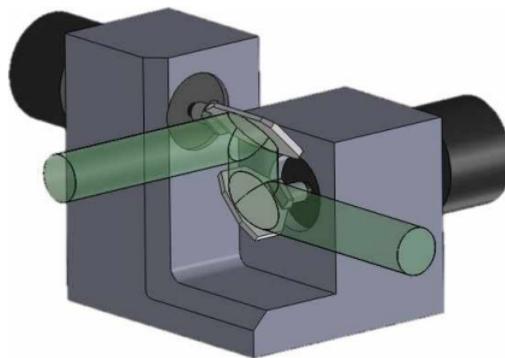




POLYTECH CLERMONT

Amélioration de l'asservissement de la visée laser

RAPPORT DE PROJET 5A DE 2025



Élèves :

Yoan DOUARRE
Marouane HSAINI
Mathis PASCAL

Enseignant Référent :

Jacques Laffont

Client :

JTL Electronique

Résumé

Ce rapport est destiné à présenter le travail que nous avons réalisé au cours de notre 4ème et 5ème année en Génie Électrique sur le projet d'amélioration de l'asservissement de la visée laser, proposé par l'industriel JTL Électronique pour la 6ème année consécutive.

Il présente une nouvelle méthode d'identification, différente de celle utilisée l'année passée, ainsi que la conception d'un correcteur RST par placement de pôles. Ce correcteur est évalué en simulation pour son efficacité face à la saturation de la commande et au bruit de mesure, avant d'être reproduit en VHDL.

Les résultats obtenus montrent que le système corrigé converge systématiquement malgré la saturation, mais atteint une limite de temps de montée, quels que soient les objectifs de poursuite. Le bruit de mesure quant à lui impacte la commande, mais cet effet peut être atténué en intégrant un filtre au correcteur. Enfin, la solution développée en VHDL reflète fidèlement les performances obtenues en simulation.

Mots-clés

Système laser - Identification SBPA - Asservissement de galvanomètres
Correcteur RST - Carte FPGA

Abstract

This report aims to present the work we carried out during our 4th and 5th years in Electrical Engineering on the project to improve laser aiming control, proposed by the industrial partner JTL Électronique for the 6th consecutive year.

It introduces a new identification method, different from the one used last year, as well as the design of an RST controller using pole placement. This controller is evaluated in simulation for its effectiveness against command saturation and measurement noise before being implemented in VHDL.

The obtained results show that the corrected system consistently converges despite saturation but reaches a limit in rise time, regardless of the tracking objectives. Measurement noise impacts the command, but this effect can be mitigated by integrating a filter into the controller. Finally, the VHDL implementation faithfully reproduces the performance obtained in simulation.

Keywords

Laser system - SBPA identification - Galvanometer control
RST controller - FPGA board

Remerciements

Nous tenons d'abord à remercier M. Pierre CHAMBERT, de l'entreprise JTL Électronique, pour sa confiance renouvelée et pour nous avoir permis de continuer de travailler sur ce projet.

Nous tenons ensuite à remercier M. Jacques LAFFONT ainsi que M. Sébastien LENGAGNE pour leur soutien et leurs encouragements tout au long de notre projet à Polytech Clermont. Leur aide nous a été très précieuse.

Nous tenons également à remercier M. François KERSULEC, de l'entreprise Punch Powertrain, pour nous avoir encadré pour les aspects de gestion de projet et qui a accepté de revenir bénévolement faire une séance de tutorat supplémentaire suite à un problème d'organisation.

Nous tenons enfin à remercier Mme. Véronique QUANQUIN et Mme. Myriam DOGHMI pour nous avoir encadré sur la partie communication autour du projet avec la réalisation d'une vidéo de présentation et sur la rédaction du rapport et des livrables.

Table des matières

1	Prise en main du projet	2
1.1	Contexte	2
1.2	Objectif du projet	2
1.3	Compréhension du fonctionnement du système	2
1.4	Définition du cahier des charges avec le client	3
1.5	Planification de l'ensemble des tâches à réaliser	4
2	Simulation d'identifications SBPA selon différentes méthodes	7
2.1	Simulation des identifications	7
2.2	Simulation des identifications pour un signal soumis à un bruit blanc	9
3	Identification réelle du système	12
3.1	Détermination du temps de montée du système	12
3.2	Réalisation d'acquisitions pour plusieurs temps d'échantillonnage .	13
3.3	Application des identifications sur les différentes acquisitions	14
4	Conversion des coefficients discrets en paramètres continus	15
4.1	Test de la conversion des coefficients discrets en paramètres continus	16
5	Calcul des coefficients RST en simulation	18
5.1	Calcul du correcteur RST par placement de pôles sans intégrateur .	18
5.2	Impact de la saturation de la commande sur la réponse du système	22
5.3	Impact du bruit de mesure sur la réponse du système	25
6	Implémentation du correcteur RST et tests	34
6.1	Test RST existant	34
6.2	Conception du RST	34
6.3	Correcteur RST en VHDL	34
6.4	Assemblage du correcteur RST	37
6.5	Validation correcteur RST en VHDL avec la simulation	39

Table des figures

1	Schéma de fonctionnement de la carte	3
2	Cahier des charges	4
3	WBS : Vue globale	5
4	WBS : Partie Automatique	5
5	WBS : Partie Utilisation de la carte	6
6	Simulation de l'identification d'un système d'ordre 3 avec les différentes méthodes sans bruit	8
7	Comparaison des coefficients du système d'ordre 3 avec les différentes méthodes	8
8	Simulation de l'identification d'un système d'ordre 3 soumis à un bruit blanc de 20 % échantillonné à $T_e = t_m/5$	9
9	Comparaison des coefficients du système d'ordre 3 avec les différentes méthodes	10
10	Simulation de l'identification d'un système d'ordre 3 soumis à un bruit blanc de 20 % échantillonné à $T_e = t_m/5$	10
11	Comparaison des coefficients du système d'ordre 3 avec les différentes méthodes	11
12	Réponse en Boucle Ouverte du galvanomètre voie X	12
13	Schéma électrique et mécanique d'un galvanomètre	13
14	Série d'acquisitions pour $T_e = 120\mu s$	14
15	Résultats de l'identification pour $T_e = 120\mu s$	14
16	Évolution de l'optimisation des variables K , τ , ξ et t_m	16
17	Comparaison de la sortie du système identifié discret avec le système optimisé en continu avec l'algorithme d'ADAM	17
18	Comparaison entre la sortie souhaitée et le correcteur RST développé l'année dernière	18
19	Réponse du correcteur RST pour un modèle à asservir représentant fidèlement le système réel	19
20	Réponse du correcteur RST pour un modèle à asservir représentant fidèlement le système réel zoomé sur $800\mu s$	20
21	Réponse du correcteur RST pour un modèle à asservir différent d'environ 20% du système réel pour $T_e = 10\mu s$	21
22	Réponse du correcteur RST pour un modèle à asservir différent d'environ 20% du système réel pour $T_e = 1\mu s$	22
23	Commande du système non saturée	23
24	Commande du système saturée	24
25	Réponse du système asservi avec saturation	24
26	Zoom sur le temps stable de la réponse du système asservi avec saturation	25
27	FFT du bruit de mesure de la réponse du système	26
28	Variance du bruit de mesure des différentes séries d'acquisitions	27
29	Réponse du correcteur RST pour un modèle à asservir différent d'environ 20% du système réel pour $T_e = 1\mu s$	28
30	Commande du système asservi avec un bruit de mesure	29
31	FFT de la réponse du système asservi non bruitée	30
32	Réponse du système asservi avec bruit de mesure filtré	31
33	Zoom sur le temps stable de la réponse du système asservi avec bruit de mesure filtré	32

34	Commande du système asservi avec bruit de mesure filtré	33
35	Comparaison méthode calcul avec théorie	34
36	Simulation Bloc T	35
37	Cycle 1	35
38	Cycle 2	35
39	Cycle 3	36
40	Résultat théorique	37
41	Simulation ModelSim	37
42	SignalTap cycle 1	38
43	SignalTap cycle 2	38
44	SignalTap cycle 3	38
45	SignalTap cycle 4	38
46	Test final	39

Introduction

Le nombre de cas de diabète de type 2 ne cesse d'augmenter en France. Cette maladie endommage les vaisseaux sanguins et peut provoquer une multitude de complications dont des troubles oculaires qui peuvent affecter la vue. Avec les progrès récents de la médecine, des opérations laser ont été mises au point afin de traiter les vaisseaux sanguins oculaires endommagés pour tenter de rétablir la vision du patient.

Le projet d'asservissement de la visée laser est un projet proposé depuis 6 ans à Polytech Clermont par l'entreprise JTL Électronique qui sous-traite pour des entreprises du domaine médical. L'objectif de ce projet était à l'origine d'afficher une matrice de 25 points avec un laser de visée sur l'oeil d'un patient à l'aide d'une carte contrôlant un système de deux galvanomètres et dont les consignes étaient calculées directement par un ordinateur utilisant Scilab et transmises en temps réel à la carte.

À la fin de ce projet il y a 3 ans, le client de JTL-Électronique a fait un retour sur le fait que la matrice affichée scintillait, ce qui était désagréable pour le chirurgien lors des opérations et a donc demandé d'améliorer la rapidité de l'affichage de cette matrice. Une nouvelle structure a donc été imaginée pour optimiser les temps de calcul qui étaient alors longs du fait de la communication systématique avec l'ordinateur et ces derniers ont été déplacés au sein d'une nouvelle carte comprenant un circuit FPGA et 2 mémoires permettant de s'affranchir de ces temps de communication et ainsi gagner en temps de calcul. Une communication avec un ordinateur subsiste mais n'est effective qu'au démarrage du système, pour identifier les transmittances des galvanomètres et calculer les coefficients du correcteur RST à envoyer dans le circuit FPGA.

Nous reprenons donc un projet d'amélioration ayant déjà été développé pendant 2 ans par d'anciens étudiants du département et avons le point de départ suivant :

- La carte électronique est programmée pour communiquer avec le circuit FPGA et l'ordinateur.
- Le circuit FPGA contenant le correcteur RST est programmé mais non testé.
- L'identification SBPA du système réel donne des résultats cohérents.
- Le calcul des coefficients du correcteur RST ne satisfait pas le cahier des charges en simulation.

Dans ce rapport de projet, nous présenterons dans un premier temps notre organisation pour cette reprise de projet, puis dans un second temps nous présenterons le travail réalisé tout au long de cette période en commençant l'utilisation de méthodes différentes de celles proposées pour l'identification du système ainsi que leurs essais en présence de bruit, puis nous aborderons la conception du correcteur RST par placement de pôles et ses essais en simulation de conditions réelles, nous continuerons alors par la présentation de l'élaboration d'une méthode permettant de reproduire les calculs théoriques du RST par une suite d'opérations réalisables en VHDL, de son implémentation dans la carte et de sa validation et nous concluerons par un mot sur notre bilan personnel.

1 Prise en main du projet

1.1 Contexte

Les personnes diabétiques peuvent souffrir de rétinopathie diabétique. Cette pathologie qui est due à un excès de sucre dans le sang va provoquer la détérioration des vaisseaux rétiniens. Cette détérioration peut alors causer la dilatation, la perte d'étanchéité, l'obstruction de ces vaisseaux ou encore d'autres complications comme le décollement de la rétine qui peuvent provoquer une sévère baisse de la vision, voire la cécité.

Pour traiter cette pathologie, les patients peuvent avoir recours à une photocoagulation au laser. Cette méthode consiste à venir viser avec un laser les zones endommagées de la rétine pour venir la blesser et ainsi, provoquer une cicatrisation pour réparer les vaisseaux détériorés ou recoller la rétine.

1.2 Objectif du projet

Actuellement, les appareils utilisés par les chirurgiens possèdent un système de visée qui va venir afficher sur l'œil du patient une matrice de points pour que le chirurgien puisse viser les zones à traiter.

Le problème est que l'affichage de cette matrice de points ne s'opère pas suffisamment rapidement dès lors que l'on donne plus de 20 points et fait alors apparaître un phénomène de scintillement des points qui rend l'étape de visée désagréable pour les chirurgiens qui peuvent aller jusqu'à afficher 25 points.

Notre objectif est donc d'afficher une matrice de 25 points suffisamment rapidement pour que les chirurgiens ne perçoivent plus cet effet de scintillement.

1.3 Compréhension du fonctionnement du système

Pour mener à bien notre objectif, notre client nous a fourni le matériel à utiliser :

- Deux galvanomètres montés sur un support commun
- Une carte électronique disposant d'un circuit FPGA pour commander les galvanomètres.

Le matériel étant imposé, il nous a donc fallu dans un premier temps comprendre comment fonctionne le système à l'aide des documents rédigés par les anciens étudiants travaillant sur le projet [1, 2, 3, 4, 5] afin de pouvoir par la suite développer notre solution pour afficher la matrice.

La carte électronique est composée d'un microcontrôleur NIOS II communiquant avec le circuit FPGA via un bus Avalon et pouvant communiquer avec un ordinateur via un port série. Le FPGA peut quant à lui communiquer avec les galvanomètres via un bus SPI. La carte comprend tous les composants nécessaires pour alimenter les galvanomètres en fonction des signaux d'informations fournis par le FPGA [7] figure 1 :

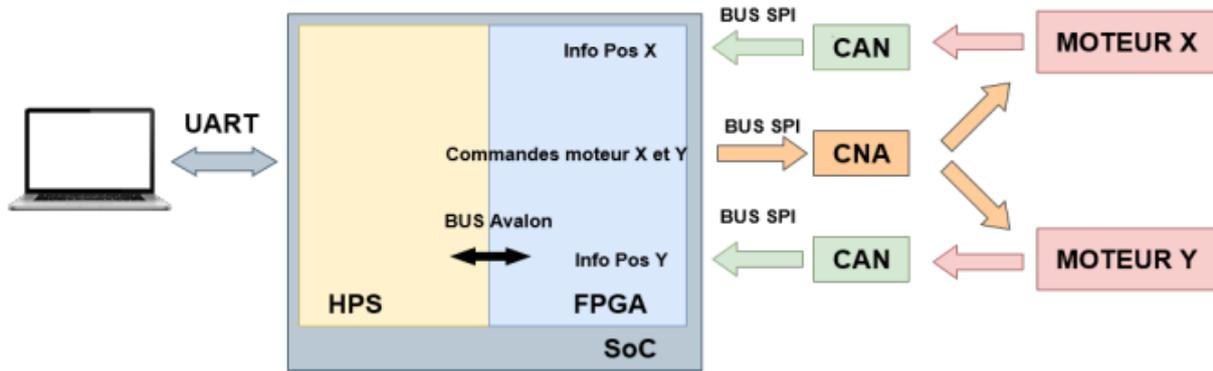


FIGURE 1 – Schéma de fonctionnement de la carte

L'objectif de cette carte est de pouvoir calculer les consignes des galvanomètres directement depuis la carte en programmant le FPGA pour qu'il serve de correcteur RST plutôt que de calculer chaque consigne depuis Scilab pour après leur envoyer. En faisant cela, on diminue considérablement le temps de calcul qui devient de l'ordre de la microseconde, ce qui nous laisse alors plus de temps pour effectuer la régulation.

1.4 Définition du cahier des charges avec le client

Une fois le fonctionnement du système assimilé, nous avons pris contact avec notre client afin de connaître précisément ses attentes ainsi que le cahier des charges à respecter.

Pour ce qui est de ses attentes, il lui faut les programmes Scilab et Quartus permettant de répondre au cahier des charges ainsi que des notes d'application expliquant le fonctionnement de ces derniers ainsi que la théorie utilisée pour l'identification SBPA et pour les calculs du correcteur RST.

Pour le cahier des charges présent figure 2, nous avons repris et paufiné avec lui celui établi l'an dernier :

Type	Nature	Détails
Fonction	Affichage d'une matrice de 25 points	<ul style="list-style-type: none"> Distance entre deux points consécutifs : 11mm pour une distance de 30cm entre les miroirs et la cible. Sans scintillements : affichage de la matrice complète en 20ms.
Contrainte	Temps de déplacement et de stabilisation d'un point	800µs par point : <ul style="list-style-type: none"> Temps de montée : 400µs Temps de stabilisation : 400µs
Contrainte	Contrôle de deux galvanomètres	Alimentation en -5V/+5V
Contrainte	Récupération des données sur la position des galvanomètres	Récupération de la position sur un CAN 14 bits
Contrainte	Calculs des commandes au sein de la carte	<ul style="list-style-type: none"> Utilisation du FPGA pour implémenter le correcteur RST Envoi des données sur un CNA 16 bits
Contrainte	Communication entre Scilab et la carte via le port série	Les caractéristiques du galvanomètre doivent être redéterminés : <ul style="list-style-type: none"> Identification du système calcul des coefficients du RST envoi des coefficients dans le FPGA

FIGURE 2 – Cahier des charges

Les éléments en rouge constituent les conditions à remplir pour afficher la matrice sans scintillements.

1.5 Planification de l'ensemble des tâches à réaliser

Maintenant que nous savons ce que nous devons faire et le cahier des charges à respecter, nous devons dresser une liste de toutes les tâches à réaliser pour concevoir notre solution. Cependant, le projet n'étant pas nouveau, il a d'abord fallu tester l'ensemble des programmes réalisés par les groupes de l'année passée.

Nous avons alors réussi à compiler le projet Quartus permettant de programmer la carte, tester la communication entre l'ordinateur et cette dernière et la commande des galvanomètres à l'aide d'un script Scilab. En revanche, nous n'avons pas validé le bon fonctionnement du programme en VHDL du RST ainsi que les scripts Scilab d'identification et de calcul du correcteur.

Le travail à réaliser est donc de refaire la partie identification de Scilab, en simulation puis sur le système réel, le calcul des coefficients du correcteur RST pour satisfaire les performances imposées par le cahier des charges, tester l'impact de la saturation de la commande et du bruit de mesure sur la réponse du système et enfin, reprogrammer le FPGA pour obtenir un comportement similaire au correcteur simulé.

Pour parvenir à cela, nous avons réalisé un WBS et planifié leur réalisation dans le Gantt disponible dans la section documents sur la forge. Le WBS est séparé en trois parties :

- Une partie sur sa vue globale (figure 3).
- Une sur la partie automatique (figure 4).
- Une sur la partie utilisation de la carte (figure 5).

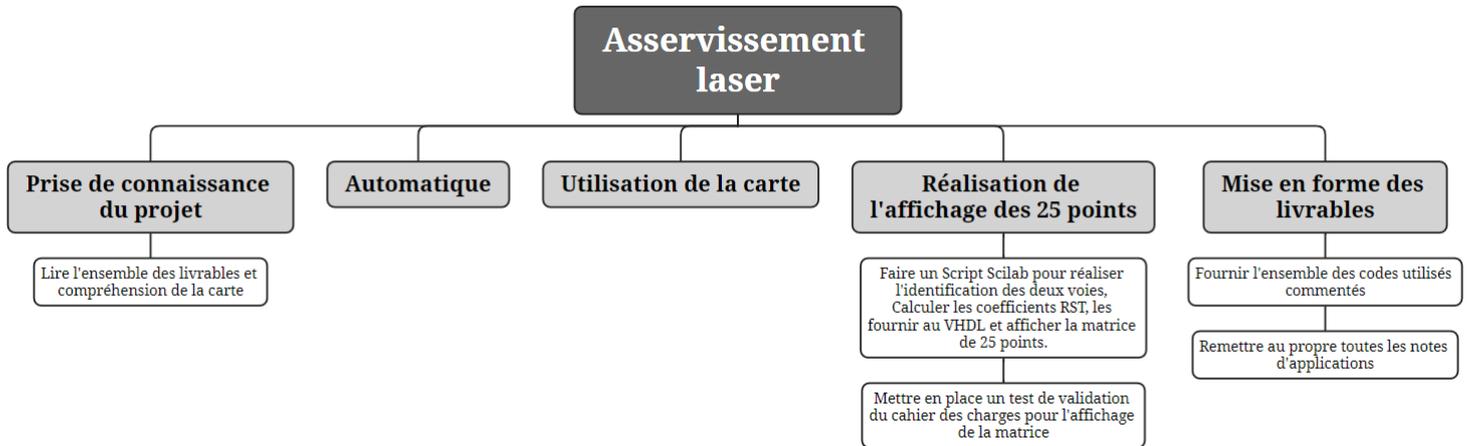


FIGURE 3 – WBS : Vue globale

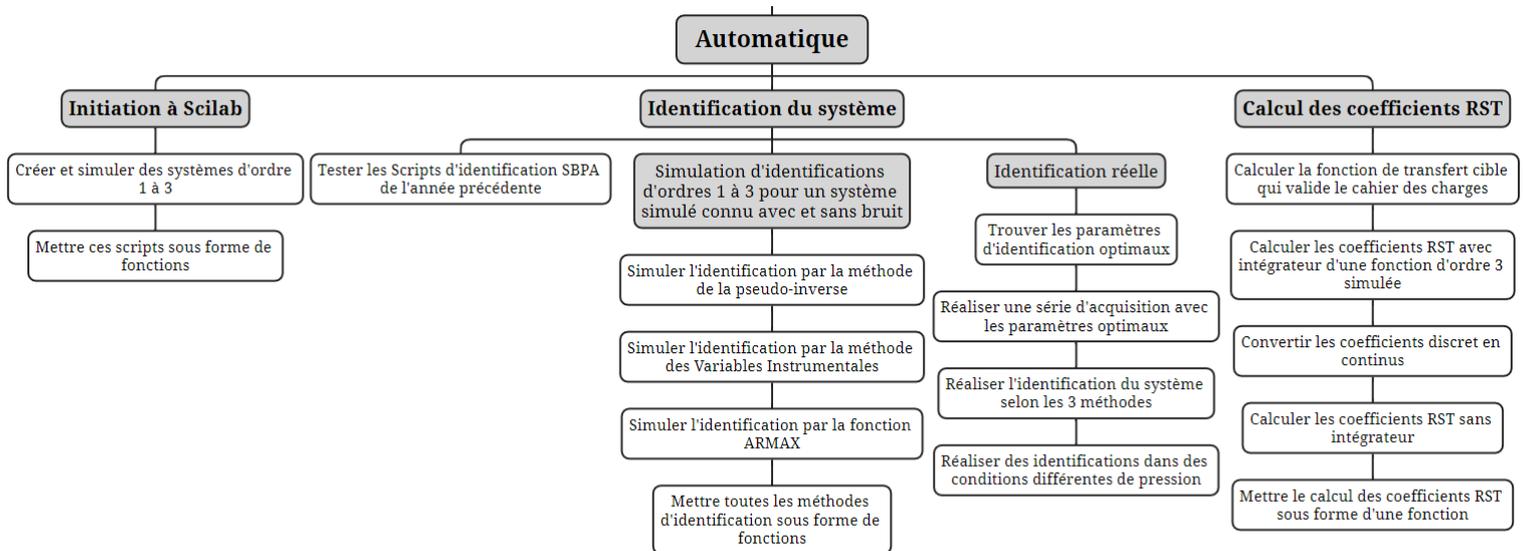


FIGURE 4 – WBS : Partie Automatique

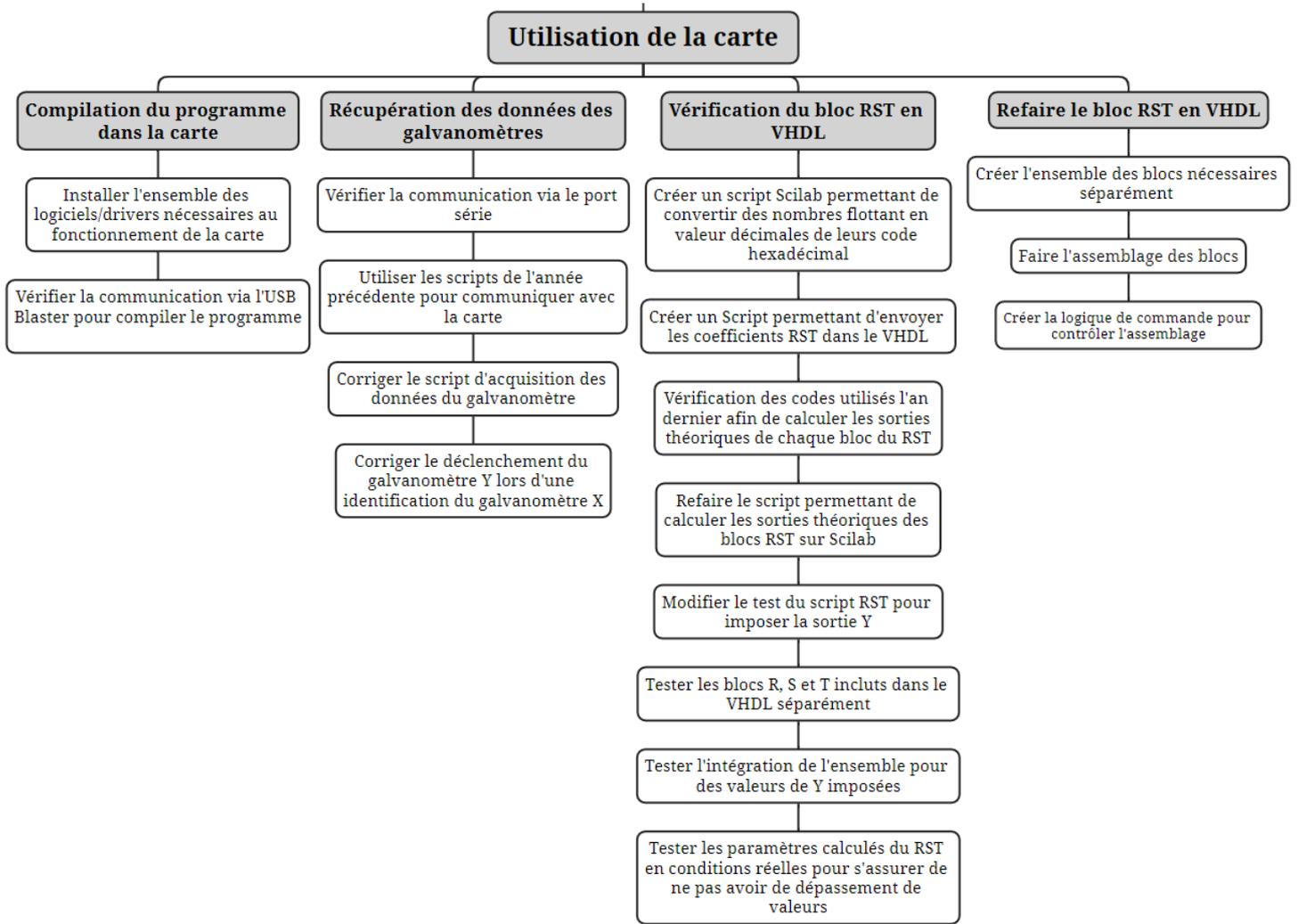


FIGURE 5 – WBS : Partie Utilisation de la carte

2 Simulation d'identifications SBPA selon différentes méthodes

Nous nous concentrons d'abord sur la partie automatique, dont le premier objectif est de déterminer une fonction de transfert qui reflète au mieux le comportement réel des galvanomètres. À partir de cette fonction, nous pourrions alors calculer les coefficients RST nécessaires pour que le système réponde aux exigences spécifiées dans le cahier des charges, puis les transmettre à la carte de contrôle.

Toutes ces étapes s'appuient sur l'utilisation de Scilab. Nous avons donc commencé par une phase d'initiation consistant à simuler des systèmes d'ordre 1 à 3, ce qui nous a permis de bien comprendre les concepts de base et les outils nécessaires. Une fois cette introduction maîtrisée, nous avons pu entamer le développement des méthodes d'identification.

Pour l'identification, nous prévoyons de tester deux méthodes SBPA distinctes : la méthode de la pseudo-inverse et celle des variables instrumentales. L'objectif est de comparer leurs performances afin de déterminer laquelle offre les meilleurs résultats lors de l'identification réelle des galvanomètres. Cette approche nous permettra de garantir une modélisation précise et robuste, essentielle pour la conception de la commande.

Des détails complémentaires sur ces méthodes et leur application sont disponibles dans la note d'application "Partie Contrôle : Identification SBPA" [6].

2.1 Simulation des identifications

Nous appliquons les 2 méthodes d'identification à un système d'ordre 3, en utilisant des scripts pour simuler des systèmes d'ordres 3 avec Scilab. Une fois les vecteurs d'entrées et de sorties générés, nous les utilisons comme arguments pour nos fonctions et constatons leur bon fonctionnement.

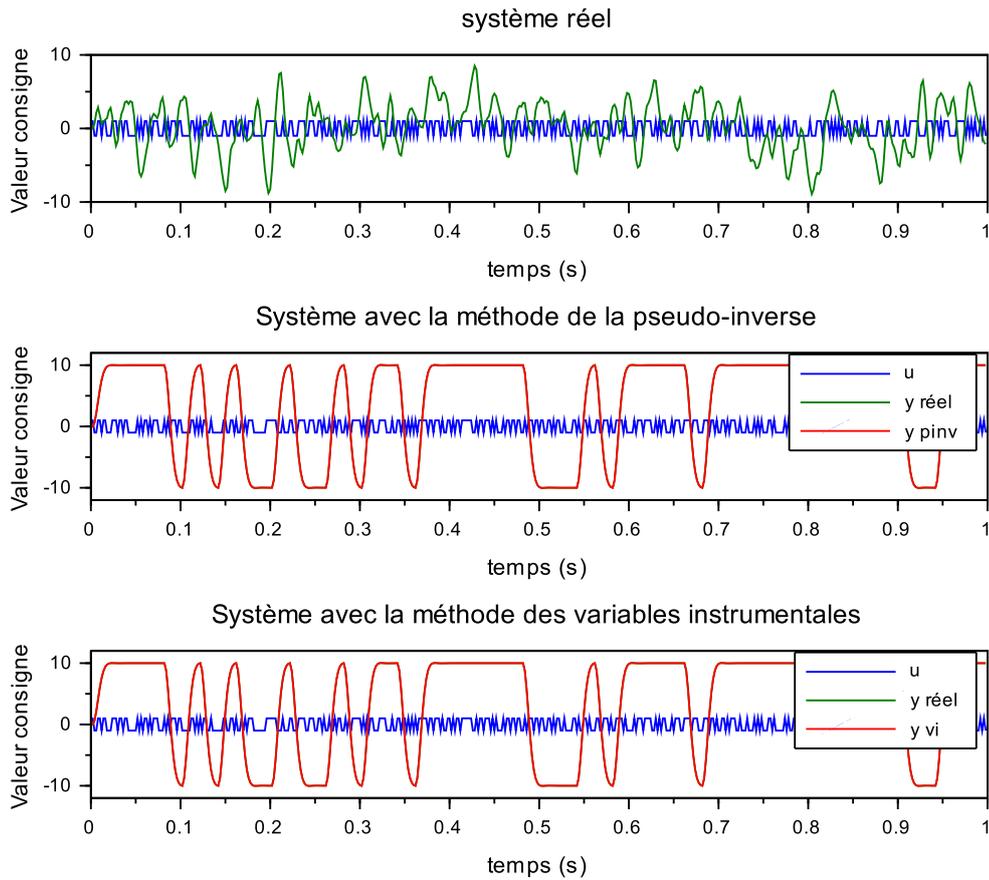


FIGURE 6 – Simulation de l'identification d'un système d'ordre 3 avec les différentes méthodes sans bruit

```

"paramètres discrets réels :"
-1.7326914
 1.0816131
-0.2417401
 0.6180884
 0.4537274
"identification pinv"
-1.7326914
 1.0816131
-0.2417401
 0.6180884
 0.4537274
"identification variables instrumentales"
-1.7326914
 1.0816131
-0.2417401
 0.6180884
 0.4537274
"erreur Pseudo-inverse :"
 1.255D-27
"erreur Variable instrumentale :"
 1.141D-25
  
```

FIGURE 7 – Comparaison des coefficients du système d'ordre 3 avec les différentes méthodes

Les figures 6 et 7 ci-dessus montrent que les deux méthodes retrouvent les paramètres exacts du système simulé.

2.2 Simulation des identifications pour un signal soumis à un bruit blanc

Nous évaluons ensuite l'efficacité des méthodes en présence de bruit. D'après nos cours d'automatique, la fréquence d'échantillonnage Te doit être choisie entre 3 et 11 fois la fréquence de montée pour garantir une identification optimale.

En ajoutant un bruit blanc de 20 % et en fixant $Te = t_m/5$, nous obtenons les résultats suivants :

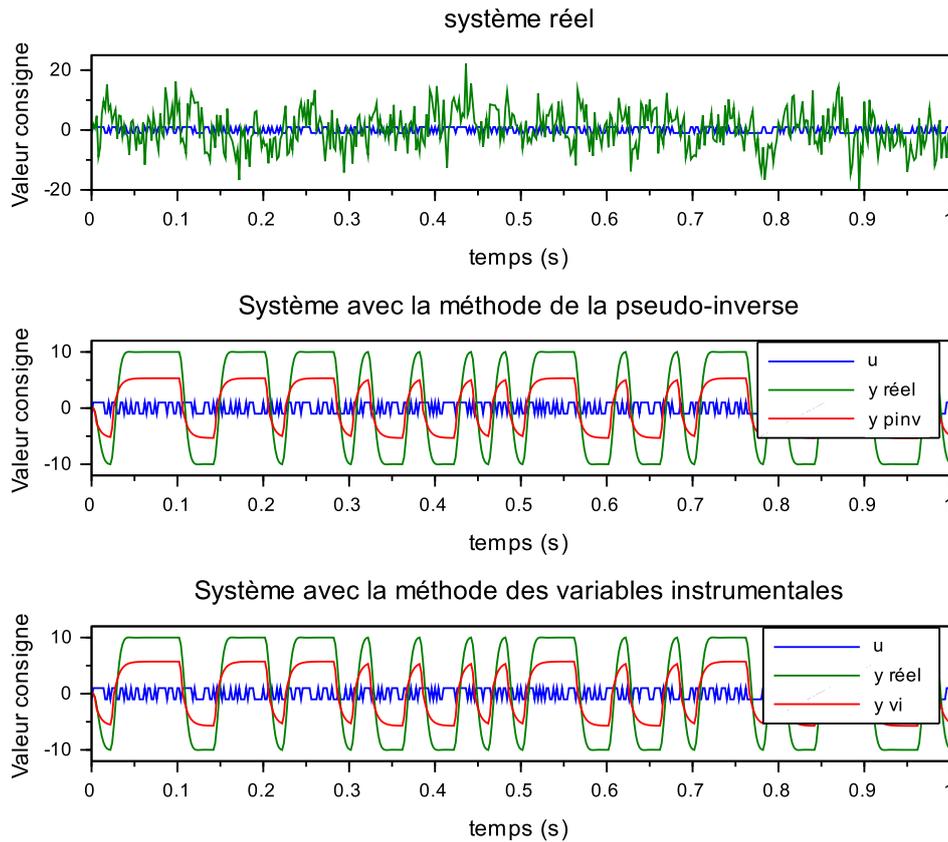


FIGURE 8 – Simulation de l'identification d'un système d'ordre 3 soumis à un bruit blanc de 20 % échantillonné à $Te = t_m/5$

```

"paramètres discrets réels :"
-1.7326914
 1.0816131
-0.2417401
 0.6180884
 0.4537274
"identification pinv"
-0.2399613
-0.1584592
-0.1177041
 0.8317717
 1.7441483
"identification variables instrumentales"
-0.3501898
-0.1243331
-0.0968575
 0.7833037
 1.6686867
"erreur Pseudo-inverse :"
 19.740293
"erreur Variable instrumentale :"
 17.077338
  
```

FIGURE 9 – Comparaison des coefficients du système d'ordre 3 avec les différentes méthodes

Les résultats sur les figures 8 et 9 montrent que, bien que les deux méthodes suivent globalement la dynamique du système simulé et que si on regarde un peu plus en détail la valeur de l'erreur pour chaque méthode sur la figure 9, la méthode des variables instrumentales donne un meilleur résultat que celle de la pseudo-inverse. Nous faisons plusieurs essais afin de confirmer les résultats et dans certains cas nous obtenons les résultats suivant (figure 10 et figure 11) :

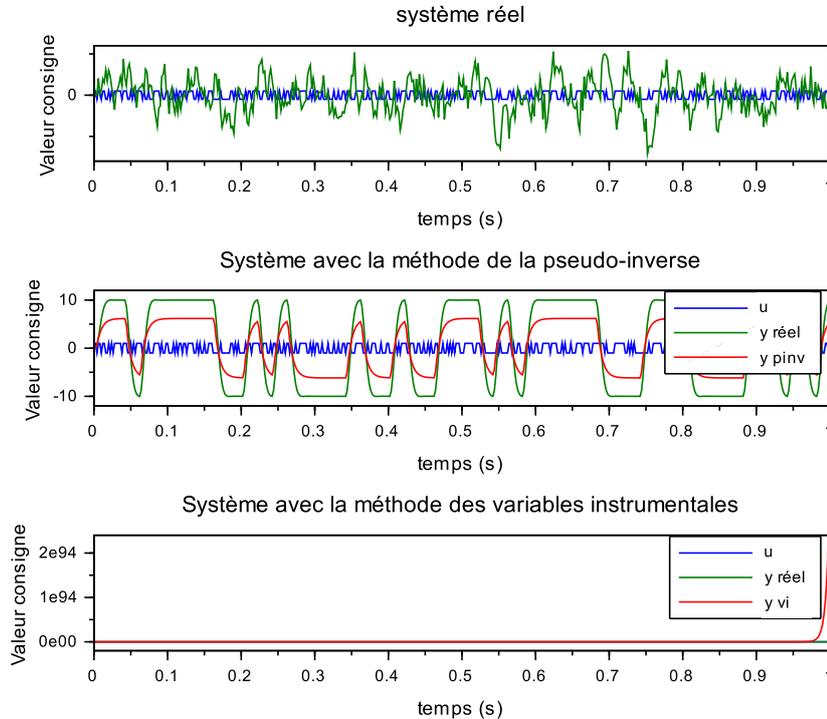


FIGURE 10 – Simulation de l'identification d'un système d'ordre 3 soumis à un bruit blanc de 20 % échantillonné à $T_e = t_m/5$

```
"paramètres discrets réels :"  
-1.7326914  
 1.0816131  
-0.2417401  
 0.6180884  
 0.4537274  
"identification pinv"  
-0.5461464  
-0.2555039  
 0.0891060  
 0.6376409  
 1.1383583  
"identification variables instrumentales"  
-2.1457915  
 0.7297927  
 0.3082402  
 0.7569387  
 0.1146772  
"erreur Pseudo-inverse :"  
 14.222582  
"erreur Variable instrumentale :"  
 1.84D+186
```

FIGURE 11 – Comparaison des coefficients du système d'ordre 3 avec les différentes méthodes

Les résultats sur les figures 10 et 11 montrent que, bien que les deux méthodes suivent globalement la dynamique du système simulé pendant plusieurs essais, la méthode des variables instrumentales peut diverger, tandis que celle de la pseudo-inverse reste stable. Nous choisissons donc cette dernière pour garantir une meilleure fiabilité.

3 Identification réelle du système

Maintenant que les simulations sont concluantes, on peut basculer sur le système réel.

3.1 Détermination du temps de montée du système

Comme observé précédemment, pour pouvoir identifier un système bruité, il faut choisir judicieusement la période d'échantillonnage ce qui impose de connaître le temps de montée de notre système.

Pour déterminer le temps de montée d'un système, la méthode la plus courante est de soumettre ce système à une entrée en échelon pour tracer sa réponse en boucle ouverte.

Si le système converge, on peut alors déterminer son temps de montée, qui correspond au temps qu'il a mis le système pour passer de 10% de son évolution à 90% lors d'une soumission à un échelon.

Si le système ne converge pas, on ne pourra pas déterminer de temps de montée puisqu'il n'y aura pas de valeur finale. On peut alors se demander s'il ne serait pas judicieux de considérer une autre modélisation de ce système qui elle convergerait.

Lors de nos essais avec une entrée en échelon, nous avons obtenus la courbe suivante présentée sur la figure 12 :

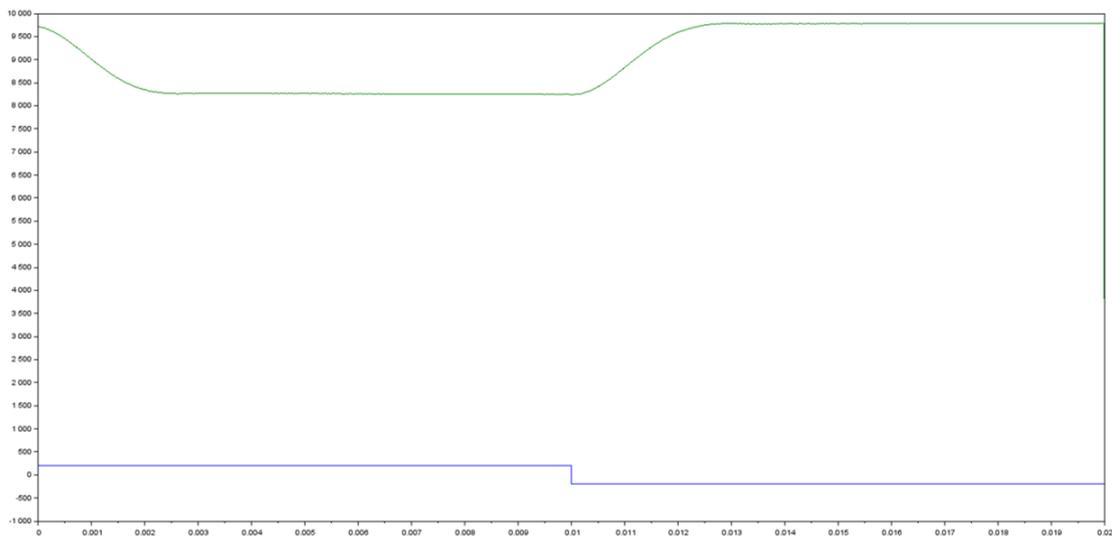


FIGURE 12 – Réponse en Boucle Ouverte du galvanomètre voie X

En observant cette courbe, on remarque que le temps de montée du galvanomètre serait autour de 2ms.

Une autre façon de déterminer le temps de montée serait de se servir des équations électriques et mécaniques pour déterminer sa fonction de transfert [7] :

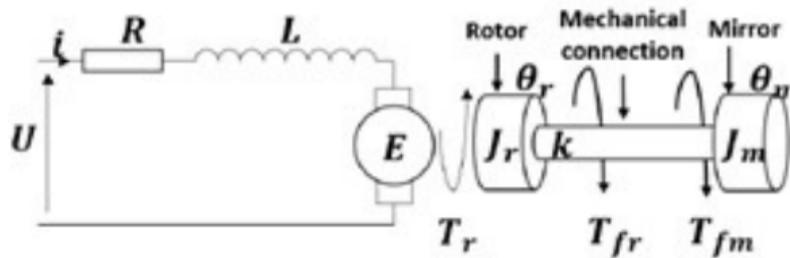


FIGURE 13 – Schéma électrique et mécanique d'un galvanomètre

$$G(p) = \frac{\theta_r(p)}{U(p)} = \frac{K}{(R + Lp)(J_r p^2 + Bp + K_\tau)}$$

A noter que cette fonction confirme que l'ordre du système à identifier est un troisième ordre et qu'il suffirait alors de remplacer les variables par les données fournies par le fabricant pour obtenir une première estimation du temps de montée.

3.2 Réalisation d'acquisitions pour plusieurs temps d'échantillonnage

Notre objectif est maintenant de réaliser plusieurs séries d'acquisitions du système soumis à une SBPA afin de créer une base de données regroupant l'entrée et les différentes sorties obtenues pour plusieurs choix de périodes d'échantillonnage T_e .

Une fois ces acquisitions réalisées, nous avons éliminé pour chaque configuration les acquisitions divergeant trop des autres pour obtenir des courbes représentatives du comportement du galvanomètres, que l'on pourra par la suite moyenner afin de réduire l'impact du bruit sur l'identification. On obtient alors à chaque fois des courbes ressemblant à celles figure 14 :

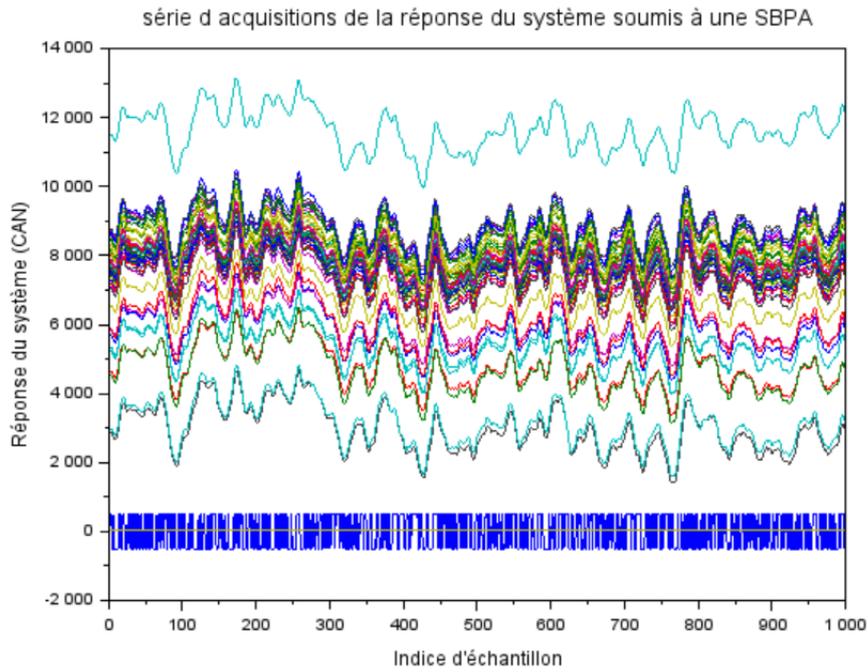


FIGURE 14 – Série d’acquisitions pour $T_e = 120\mu s$

3.3 Application des identifications sur les différentes acquisitions

Pour déterminer la meilleure période d’échantillonnage, nous avons alors comparé la réponse de chaque système identifié avec le système réel pour une commande SBPA différente et avons calculé leur erreur quadratique par rapport au système réel. On trouve alors les meilleurs résultats pour $T_e = 120\mu s$ qui nous donne alors la réponse figure 15 :

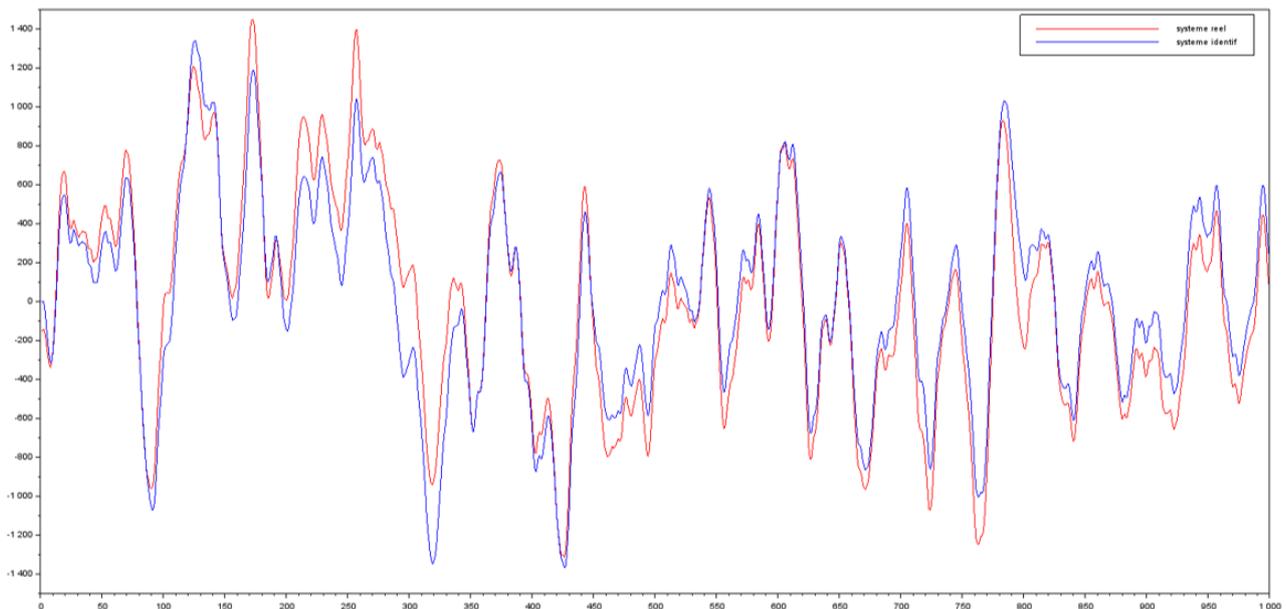


FIGURE 15 – Résultats de l’identification pour $T_e = 120\mu s$

Nous observons alors que le système d'ordre 3 identifié suit l'allure du système réel mais nous ne pouvons pas certifier que cette identification suffise pour que la correction fonctionne. Nous pourrions vérifier cela qu'une fois la correction appliquée directement sur les galvanomètres.

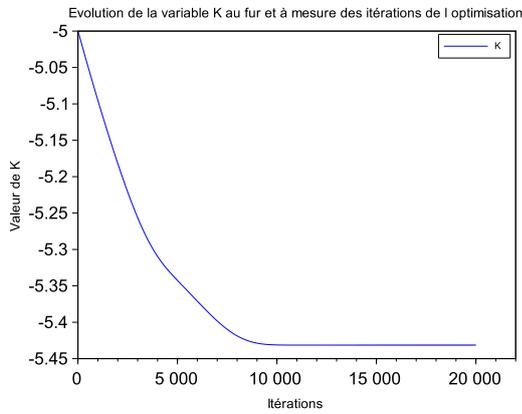
4 Conversion des coefficients discrets en paramètres continus

Les coefficients discrets obtenus précédemment ne peuvent pas être utilisés tels quels pour calculer la correction car ils dépendent de la période d'échantillonnage, hors, les identifications ont été réalisées pour une période d'échantillonnage bien plus élevée que celle qui sera utilisée lors de l'asservissement. Il faut donc au préalable retrouver les coefficients continus correspondants aux coefficients discrets identifiés.

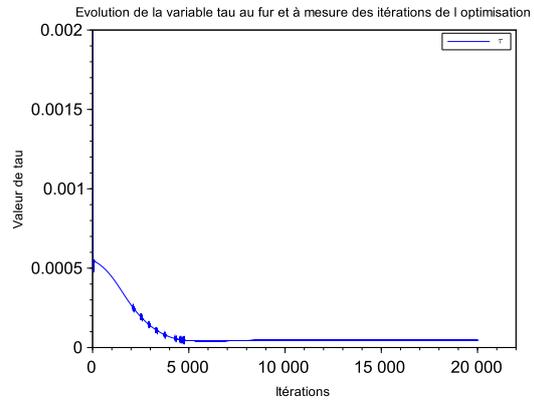
Pour convertir les coefficients discrets en paramètres continus, nous utilisons l'algorithme d'optimisation d'Adam, choisi pour sa capacité à converger rapidement vers un minimum local. Cet algorithme va optimiser les paramètres K , τ , ξ et t_m d'un système continu initial choisi pour minimiser l'erreur quadratique moyenne entre sa réponse et celle du système discret identifié.

Les détails complets sur cette conversion sont également disponibles dans la note d'application "Partie Contrôle : Identification SBPA" [6].

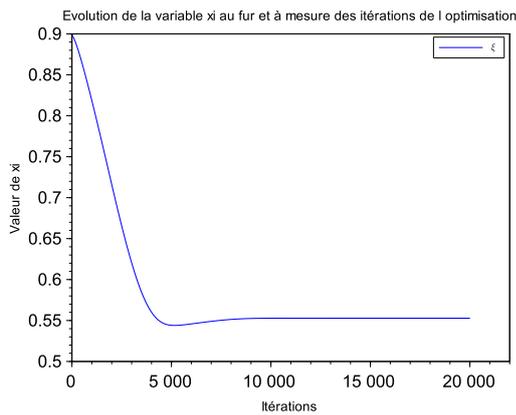
4.1 Test de la conversion des coefficients discrets en paramètres continus



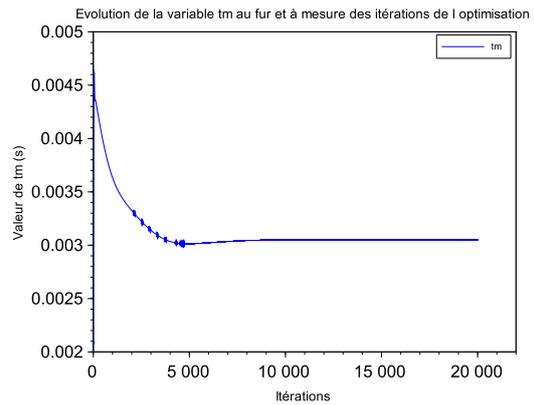
(a) Évolution de la variable K



(b) Évolution de la variable τ



(c) Évolution de la variable ξ



(d) Évolution de la variable t_m

FIGURE 16 – Évolution de l'optimisation des variables K , τ , ξ et t_m .

D'après la figure 16 ci-dessus, on observe que les variables K , τ , ξ et t_m arrivent à converger vers une valeur finie. Si on observe la comparaison de la sortie du système identifié discret avec le système optimisé en continu avec l'algorithme d'ADAM on obtient le résultat suivant (figure 17) :

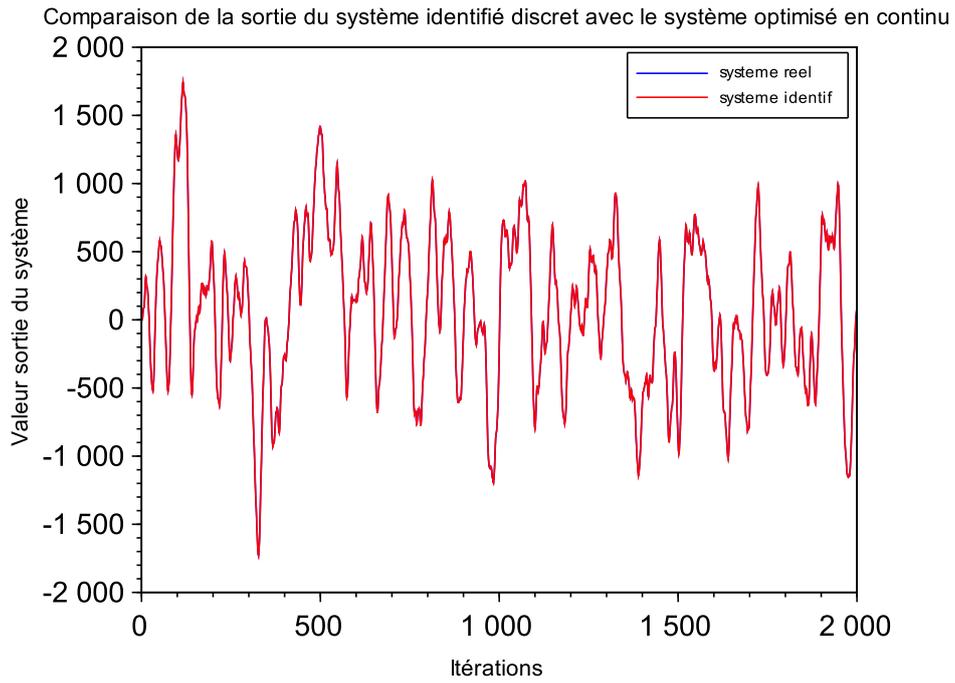


FIGURE 17 – Comparaison de la sortie du système identifié discret avec le système optimisé en continu avec l’algorithme d’ADAM

La figure 17 ci-dessus permet de valider le bon fonctionnement de la conversion des coefficients discrets en paramètres continus car on observe que le système identifié se superpose sur le système réel.

5 Calcul des coefficients RST en simulation

Nous allons maintenant aborder la réalisation du correcteur RST qui est une partie importante de notre projet car cette dernière présentait un problème à la fin de la période de projet de l'an dernier.

Pour rappel, voici les résultats en simulation que donnait le correcteur développé par les étudiants l'année passée [5] (figure 18) :

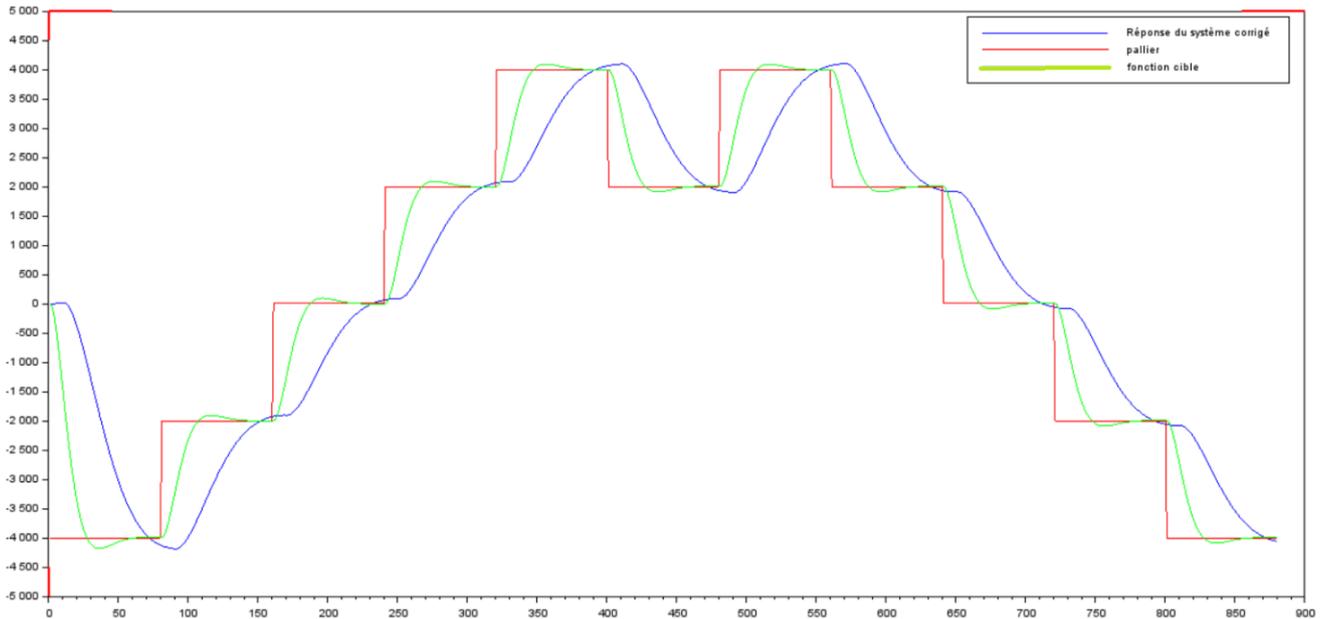


FIGURE 18 – Comparaison entre la sortie souhaitée et le correcteur RST développé l'année dernière

On remarque que les coefficients du correcteur RST calculés ne permettaient pas d'obtenir le comportement souhaité.

5.1 Calcul du correcteur RST par placement de pôles sans intégrateur

Maintenant que le système est identifié, nous pouvons réaliser le calcul du correcteur RST en suivant la méthode développée dans la note d'application "Conception théorique d'un correcteur RST par placement de pôles" [8], issue du cours de M. Lengagne [9].

Il s'agit d'un correcteur RST par placement de pôles qui est une méthode permettant d'asservir des systèmes d'ordres quelconques avec des retards quelconques.

De plus, la particularité des correcteurs RST est qu'ils permettent un asservissement à deux degrés de libertés : un pour la poursuite et l'autre pour la régulation.

Lorsque l'on réalise ces calculs pour le système du troisième ordre identifié précédemment, composé du produit d'un premier ordre et d'un second ordre dont les

paramètres continus sont :

- $\tau = 3.1\mu s$ temps de réponse du premier ordre.
- $\xi = 0.69$ facteur d'amortissement du second ordre.
- $t_m = 3.03ms$ temps de montée du second ordre.
- $K = -5.24$ gain statique du système.

En les convertissant en discret pour une période d'échantillonnage $T_e = 10\mu s$ à l'aide des tables [I,II], on obtient les résultats figure 19 :

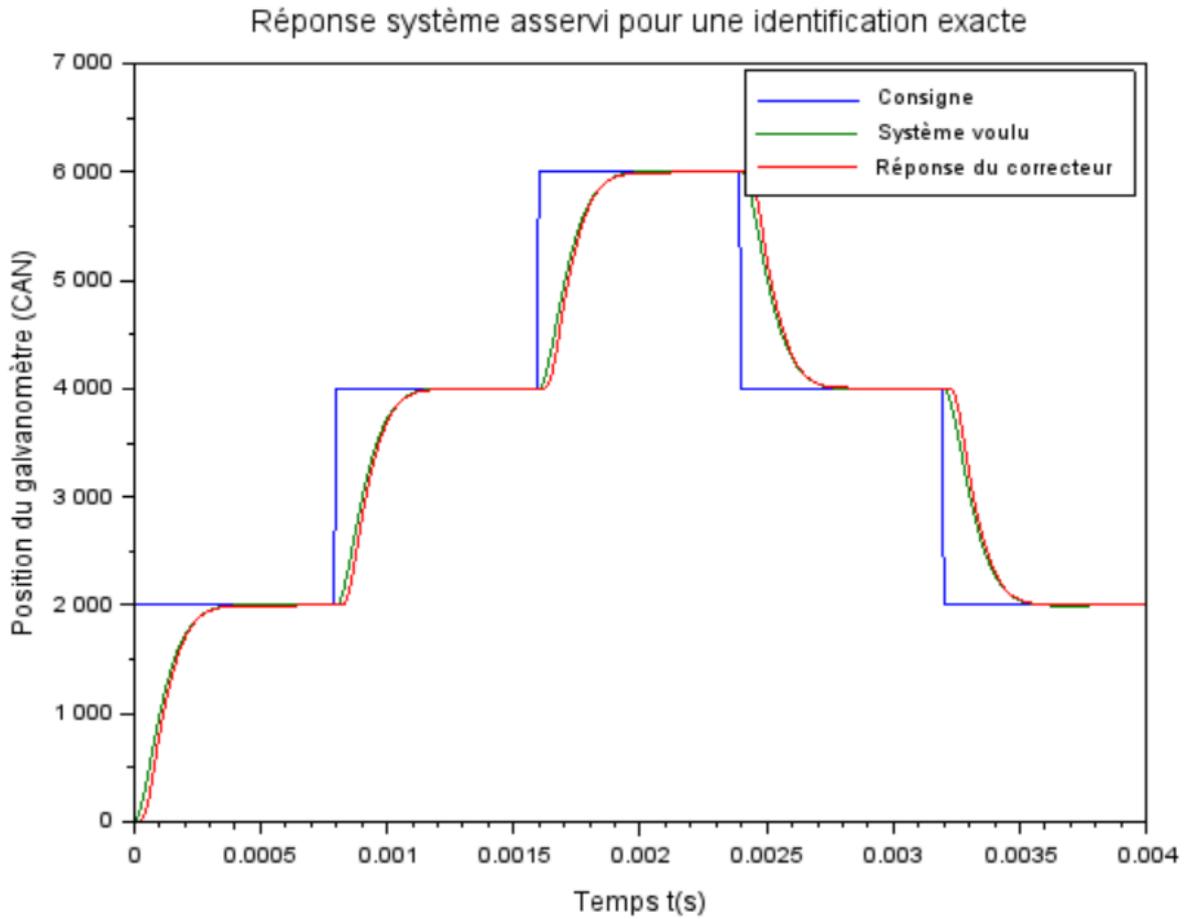


FIGURE 19 – Réponse du correcteur RST pour un modèle à asservir représentant fidèlement le système réel

On remarque alors figure 20 que la réponse du correcteur est très proche de la réponse que nous voulions, l'écart observé durant la montée jusqu'à la consigne étant dû à une approximation lors du calcul de $T(z)$, détaillée dans la note d'application [8], mais qui n'impacte ni le temps de montée, ni le gain statique de l'asservissement.

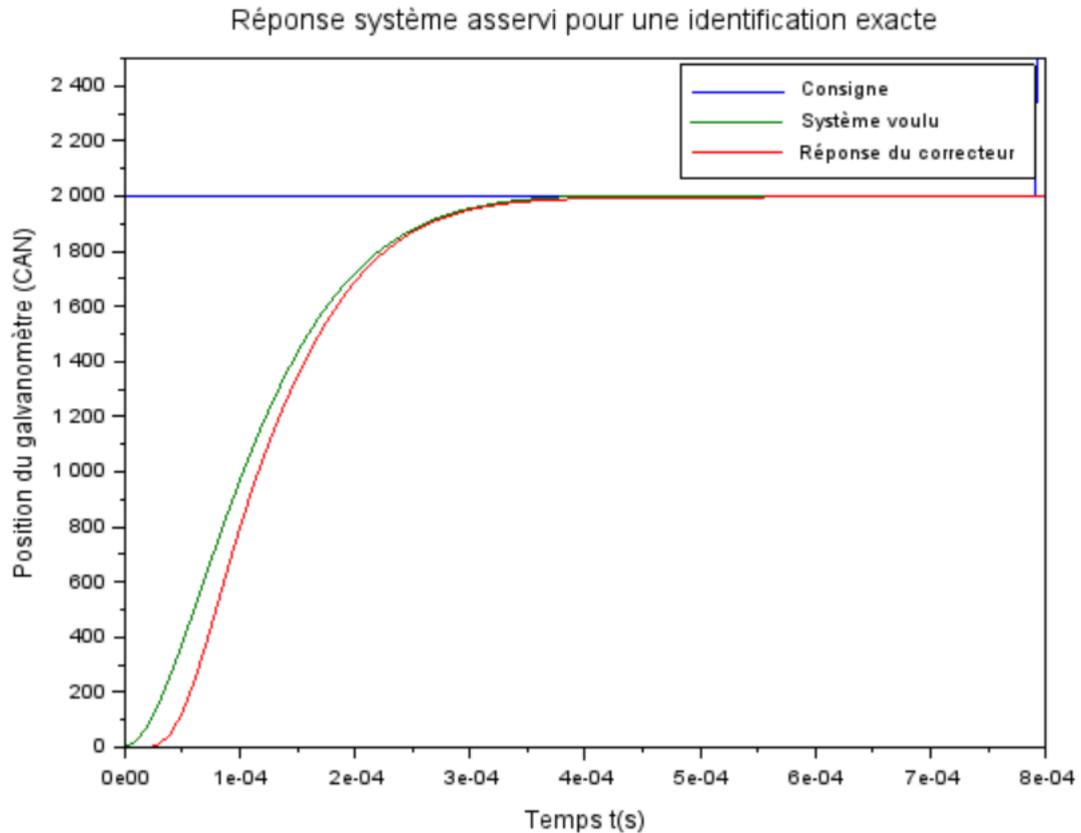


FIGURE 20 – Réponse du correcteur RST pour un modèle à asservir représentant fidèlement le système réel zoomé sur $800\mu s$

La question qui se pose maintenant est celle de l'impact d'une imprécision du modèle à asservir, ce qui risque d'être le cas avec l'identification SBPA de notre système. Nous avons alors repris les mêmes coefficients du correcteur mais modifié le système réel à asservir comme suit :

- $\tau = 37.2\mu s$ temps de réponse du premier ordre.
- $\xi = 0.55$ facteur d'amortissement du second ordre.
- $t_m = 3.6ms$ temps de montée du second ordre.
- $K = -4.19$ gain statique du système.

et nous avons obtenu les résultats figure 21 :

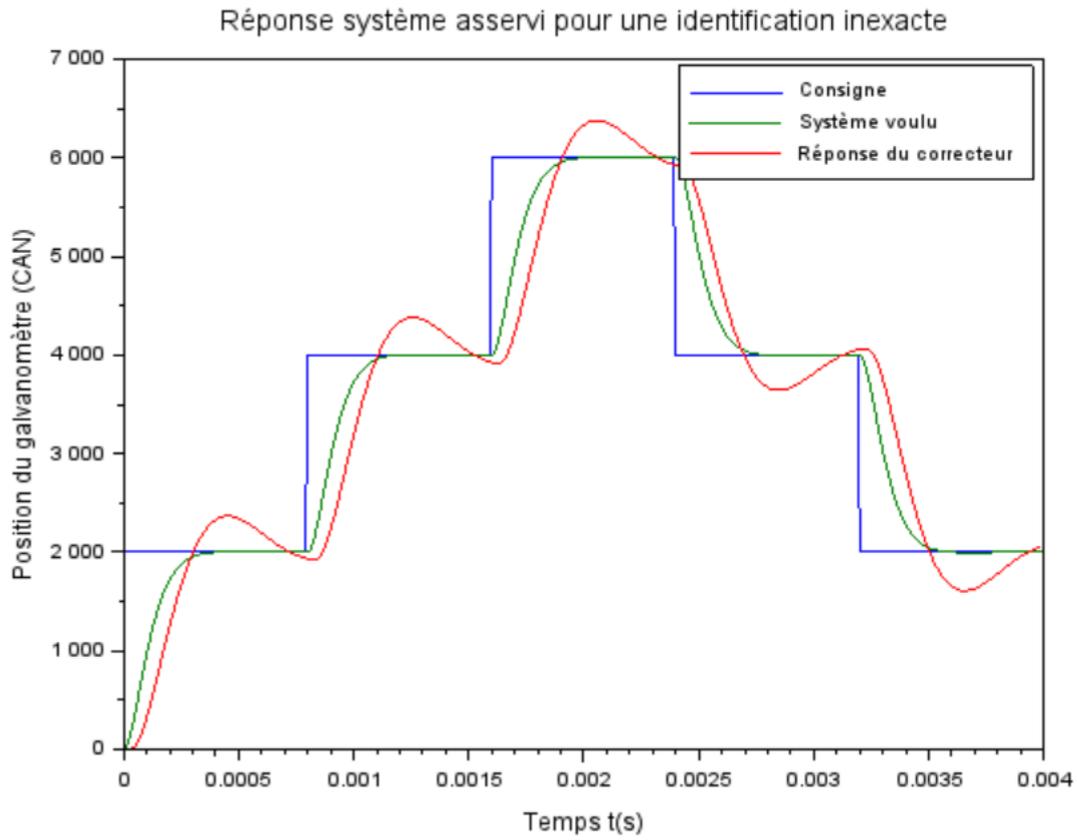


FIGURE 21 – Réponse du correcteur RST pour un modèle à asservir différent d'environ 20% du système réel pour $T_e = 10\mu s$

On observe alors que le système asservi ne satisfait plus le cahier des charges.

Nous avons alors retenté l'expérience en modifiant $T_e = 1\mu s$ et avons obtenu les résultats figure 22 :

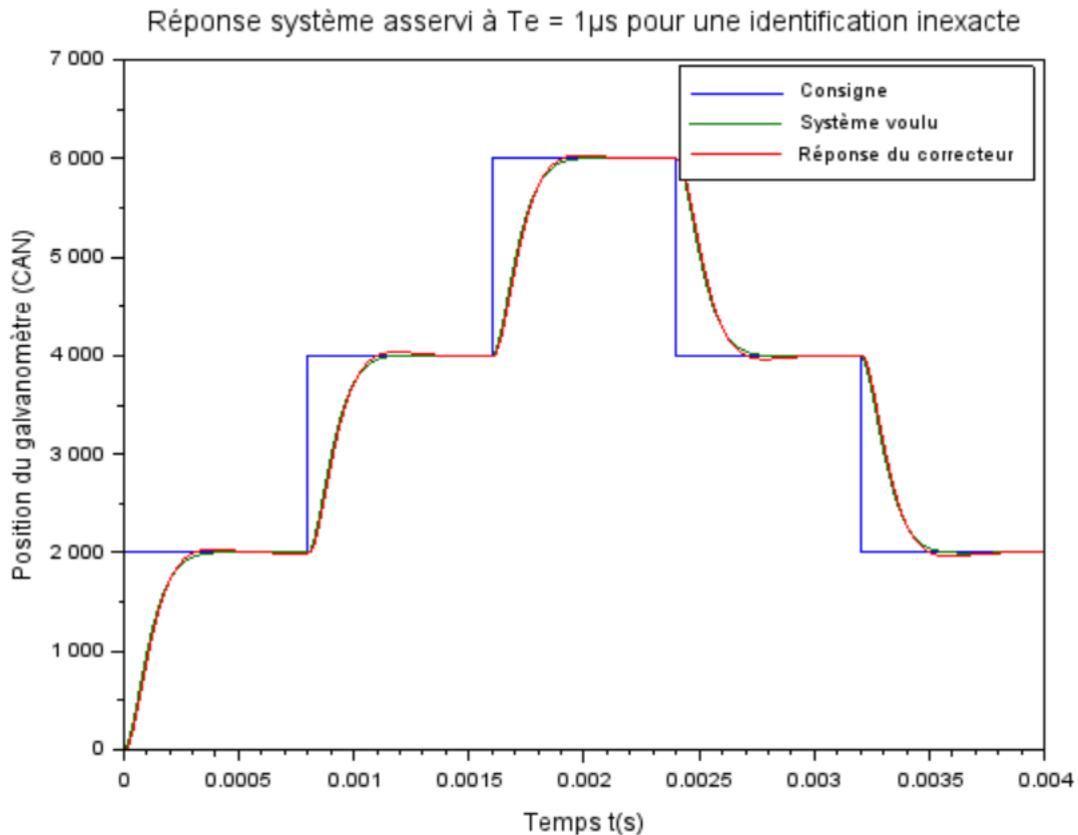


FIGURE 22 – Réponse du correcteur RST pour un modèle à asservir différent d'environ 20% du système réel pour $T_e = 1\mu s$

Nous observons maintenant que le système nous renvoie de nouveau l'allure souhaitée ce qui montre que le choix de la période d'échantillonnage pour l'asservissement peut palier une inexactitude de l'identification.

5.2 Impact de la saturation de la commande sur la réponse du système

Le correcteur simulé jusqu'à présent ne prenait pas en compte la contrainte matérielle sur la commande qui est que la plage de tensions possible à envoyer aux galvanomètres est de $-5V$ à $+5V$ ou $(-32768$ à $+32768$ en valeur du CAN).

Pour pouvoir accéder aux signaux internes du système, et donc à la commande, nous avons dû refaire le script de simulation en remplaçant la fonction `dsimul()`, qui était alors utilisée pour effectuer les calculs, par la fonction `filter()`. Ce changement de fonction nous a permis de réaliser la simulation en pas à pas et donc, de visualiser la commande et de venir ajouter la saturation.

La commande envoyée au système sans saturation suit alors l'allure figure 23 :

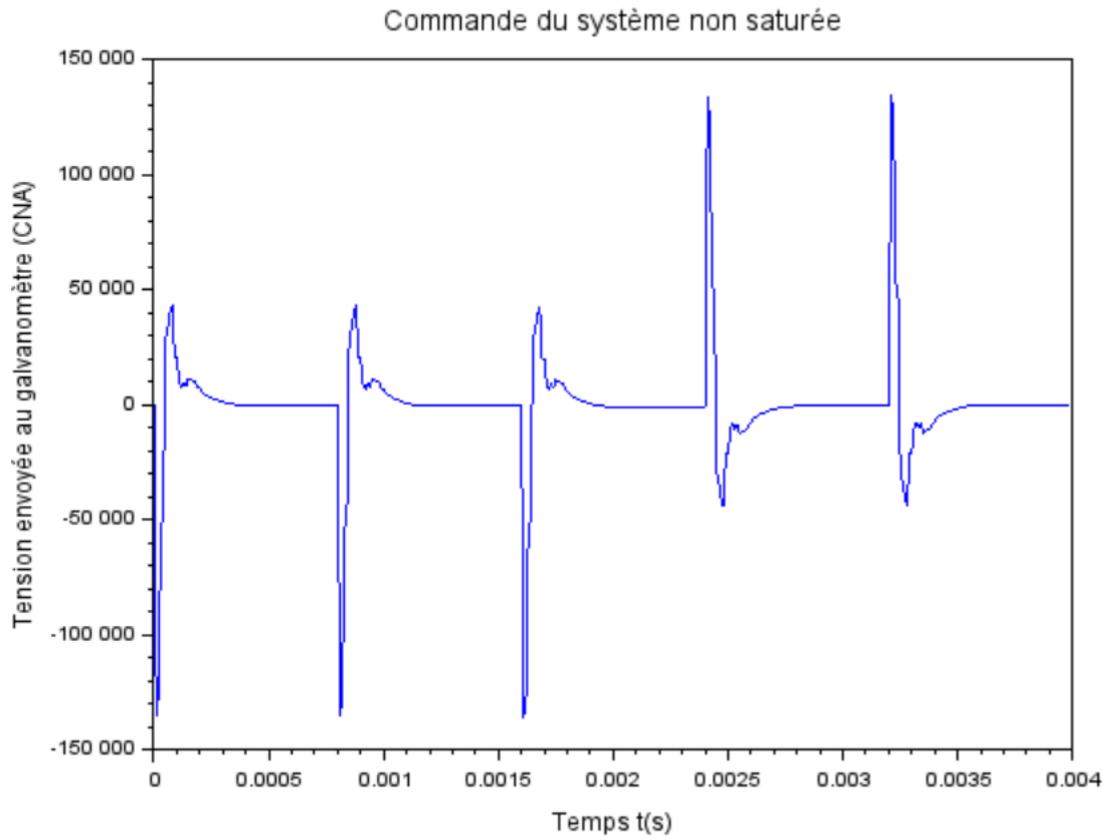


FIGURE 23 – Commande du système non saturée

On constate qu'à chaque nouvelle coordonnée envoyée, l'asservissement demande un pic de tension de commande bien supérieur aux bornes de saturations du système.

Si l'on ajoute la saturation, on obtient alors les résultats figures 24 et 25 :

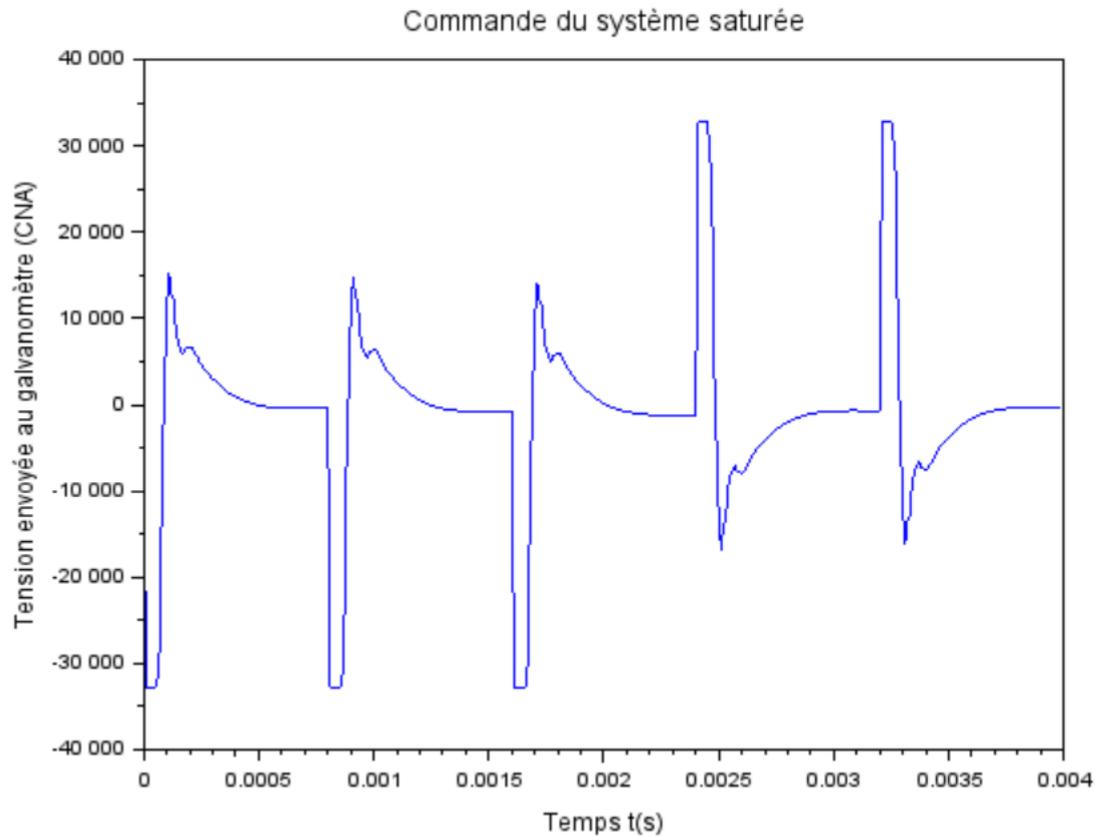


FIGURE 24 – Commande du système saturée

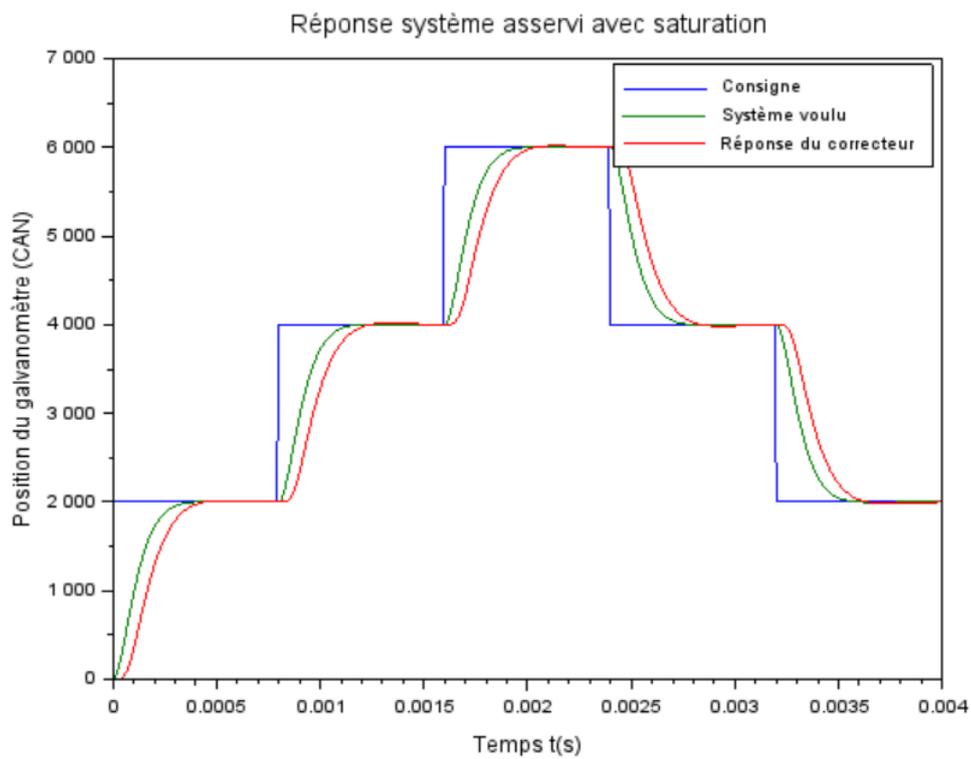


FIGURE 25 – Réponse du système asservi avec saturation

On remarque que le système est plus lent que précédemment car il met plus de temps avant de commencer de croître. Cependant, si l'on s'intéresse à son temps stable à 1% figure 26, on obtient :

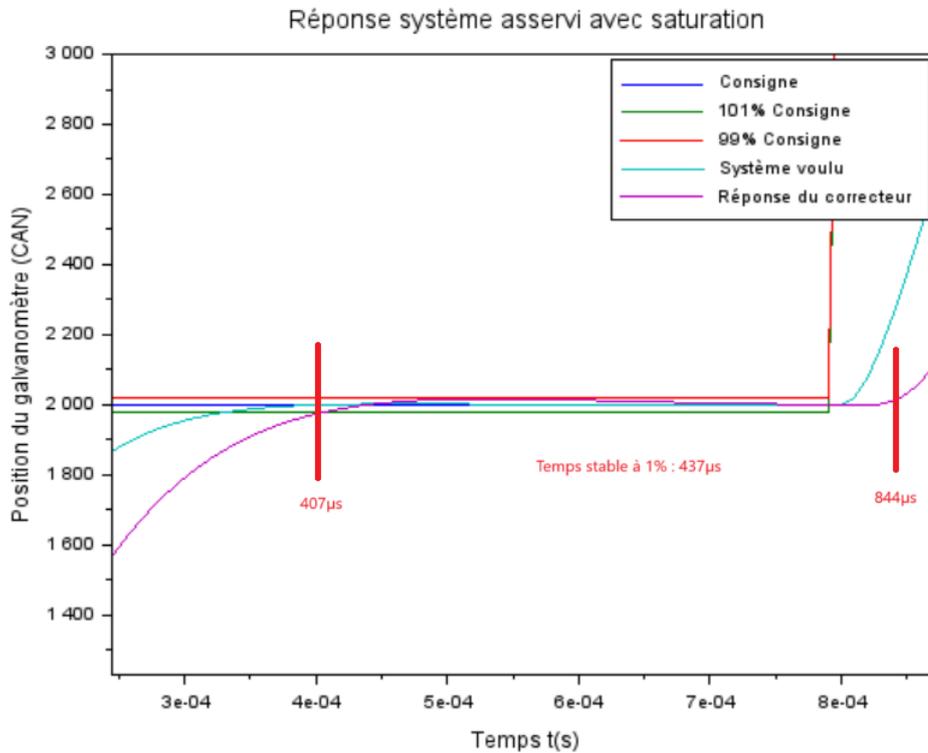


FIGURE 26 – Zoom sur le temps stable de la réponse du système asservi avec saturation

Malgré le ralentissement du système, on observe que ce dernier reste stable pendant $437\mu s$ ce qui signifie que le temps de déplacement entre 2 points est de $363\mu s$, ce qui satisfait encore le cahier des charges.

5.3 Impact du bruit de mesure sur la réponse du système

La dernière simulation à faire est maintenant d'asservir un système dont la mesure de la réponse est bruitée. En effet, lors des différentes acquisitions réalisées sur le système réel, on a constaté que les mesures étaient légèrement bruitées.

Pour pouvoir simuler au mieux l'impact de ce bruit sur notre asservissement, il faut dans un premier temps le caractériser pour pouvoir le remodeliser. Pour cela, nous avons réalisé une série d'acquisitions à divers moments de la journée, à froid comme à chaud, de la mesure de la réponse du galvanomètre lorsque ce dernier était immobile pour calculer leurs FFTs (Fast Fourier Transforms¹).

1. La Transformée de Fourier Rapide (FFT) est un algorithme permettant de calculer la Transformée de Fourier Discrète. Elle permet de convertir un signal du domaine temporel au domaine fréquentiel de manière rapide et efficace.

On obtient alors à chaque fois des courbes semblables à celle figure 27 :

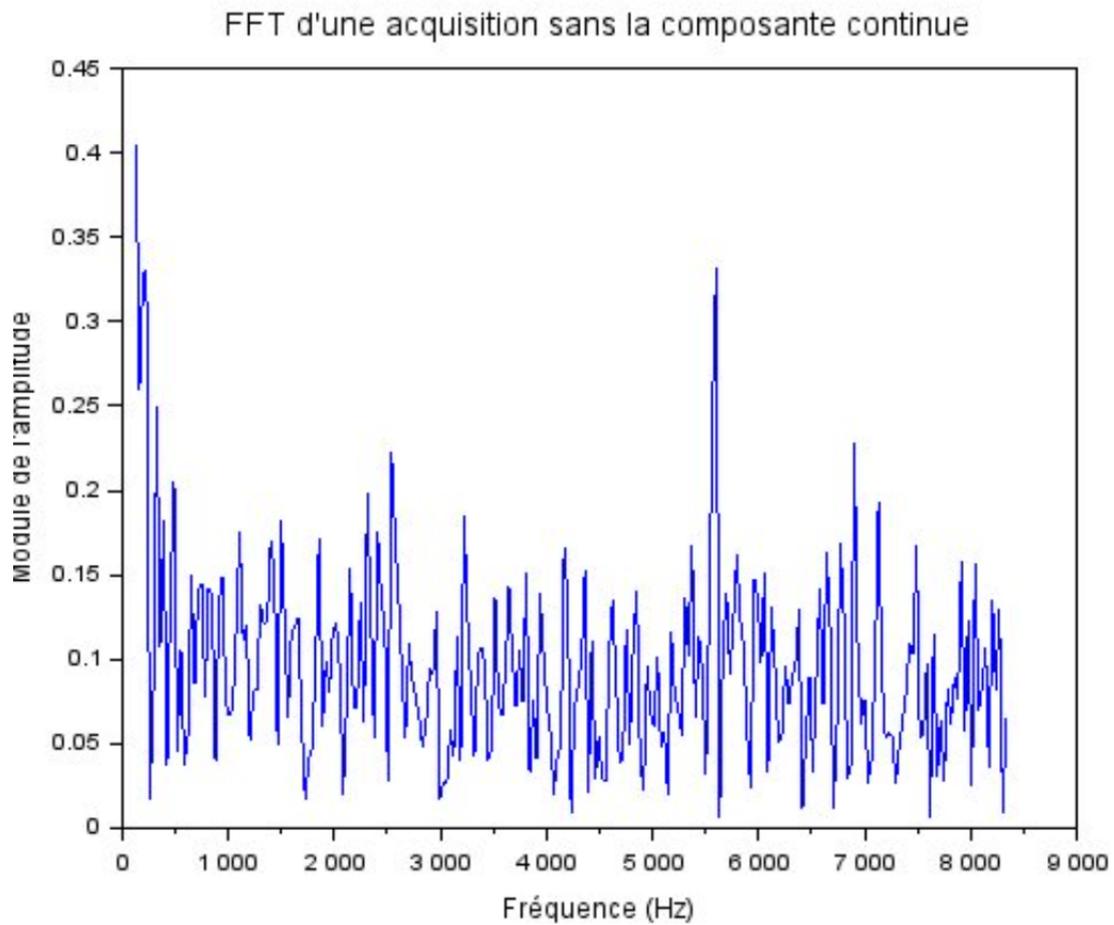


FIGURE 27 – FFT du bruit de mesure de la réponse du système

On remarque alors que ce bruit peut être approximé à un bruit blanc.

Nous avons alors calculé la variance de ce bruit pour chaque série d'acquisitions et avons obtenus les résultats figure 28 :

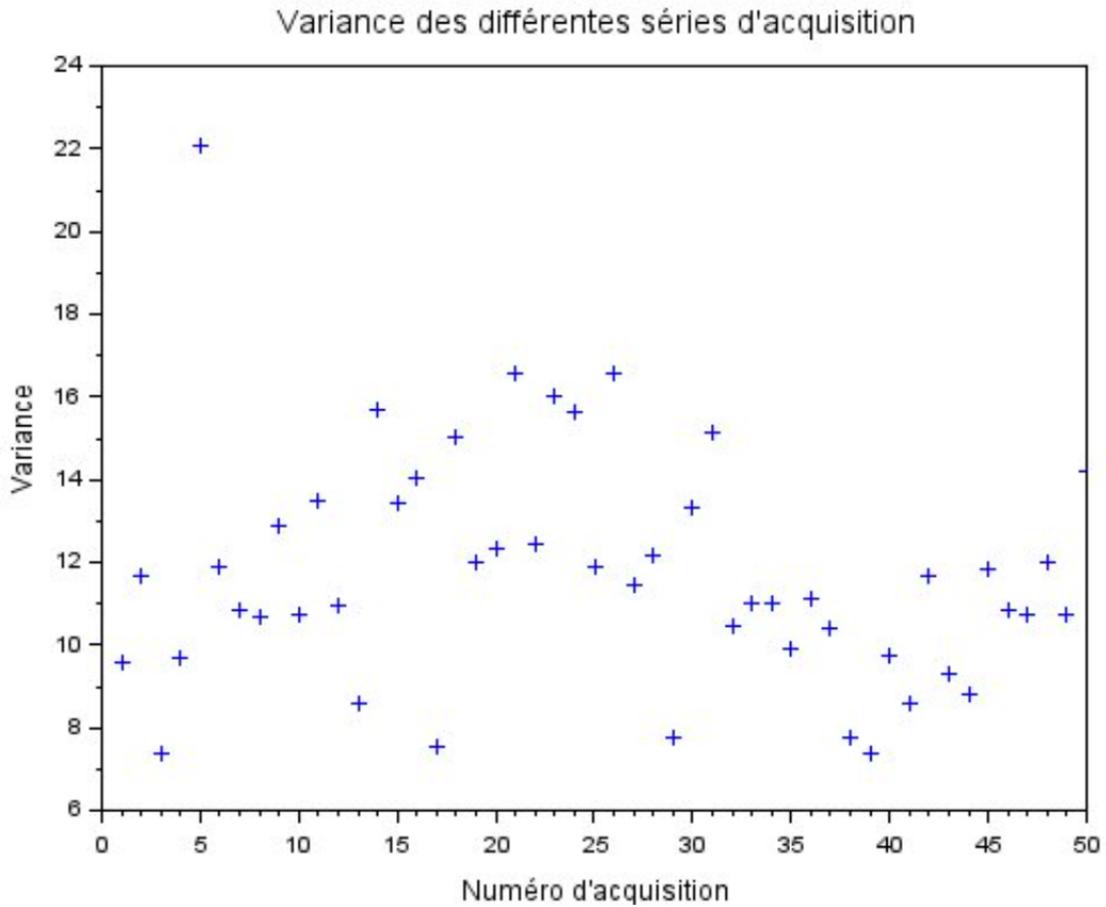


FIGURE 28 – Variance du bruit de mesure des différentes séries d’acquisitions

Nous observons alors une variance moyenne de 11.7 avec une acquisition où la variance a atteint un pic à 2205, ce qui nous fait un rapport signal sur bruit d’environ 500 (car la réponse du galvanomètre est comprise entre 0 et 16384). Nous avons donc choisi de modéliser le bruit par un bruit blanc de variance 25.

Si l’on reprend notre correcteur avec la saturation et que nous lui ajoutons ce bruit, on obtient alors les résultats figure 29 :

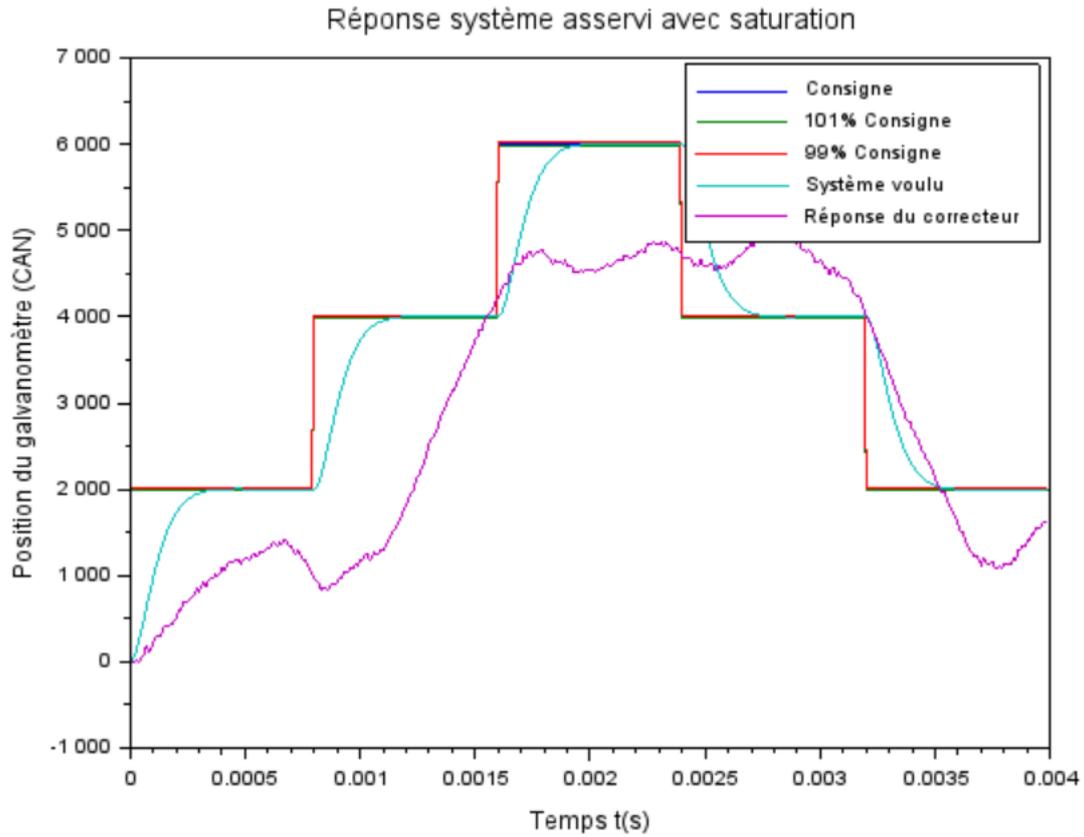


FIGURE 29 – Réponse du correcteur RST pour un modèle à asservir différent d'environ 20% du système réel pour $T_e = 1\mu s$

On constate alors qu'une mesure à peine bruitée fausse complètement l'asservissement.

Pour comprendre d'où vient ce phénomène, il faut regarder la commande figure 30 :

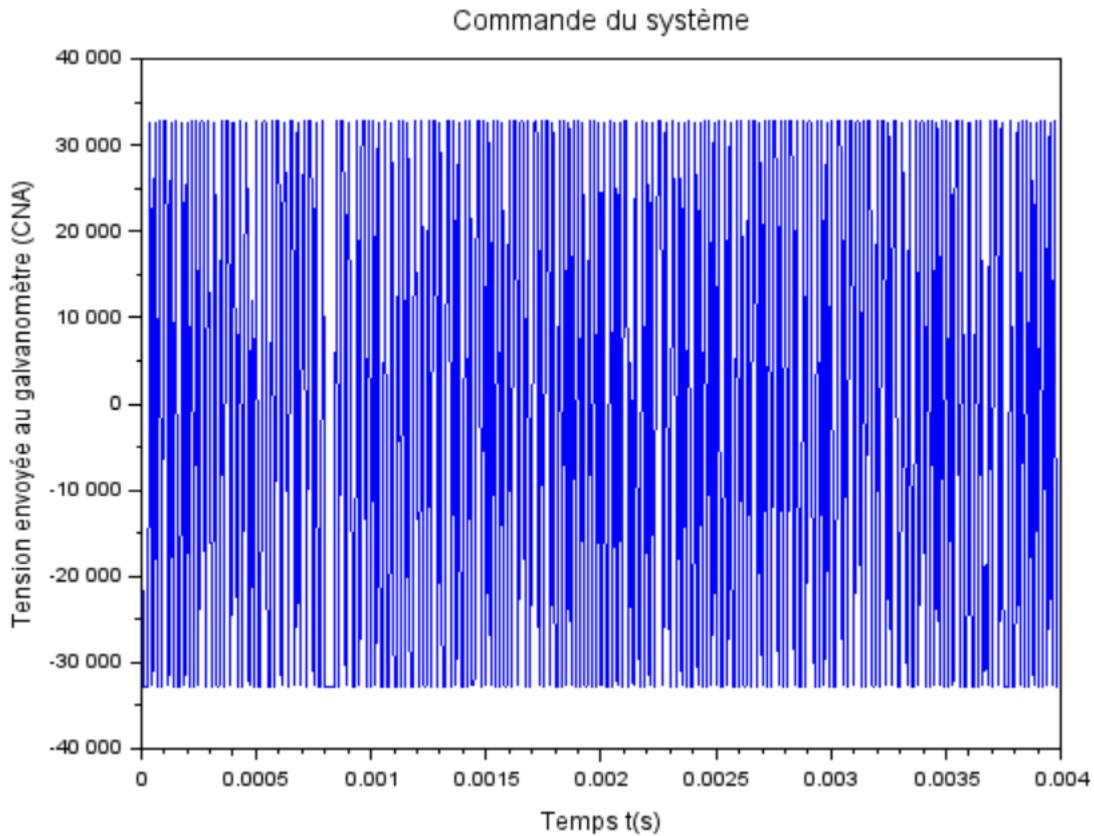


FIGURE 30 – Commande du système asservi avec un bruit de mesure

Le système cherchant à réguler chaque bruit sur la mesure, ce dernier génère une commande chaotique qui va alors saturer en permanence. Cette commande est d'autant plus chaotique que le système est asservi rapidement, car le correcteur s'attend alors à avoir un écart moins important entre chaque mesure.

Pour palier ce problème, nous ajoutons un filtre au correcteur (cf. Note d'application RST Théorique). Pour savoir quel type de filtre ajouter au correcteur, il faut auparavant connaître les fréquences utiles de la réponse du système pour ne pas détériorer le signal. Pour cela, on reprend notre simulation sans bruit et on trace la FFT de sa réponse figure 31 :

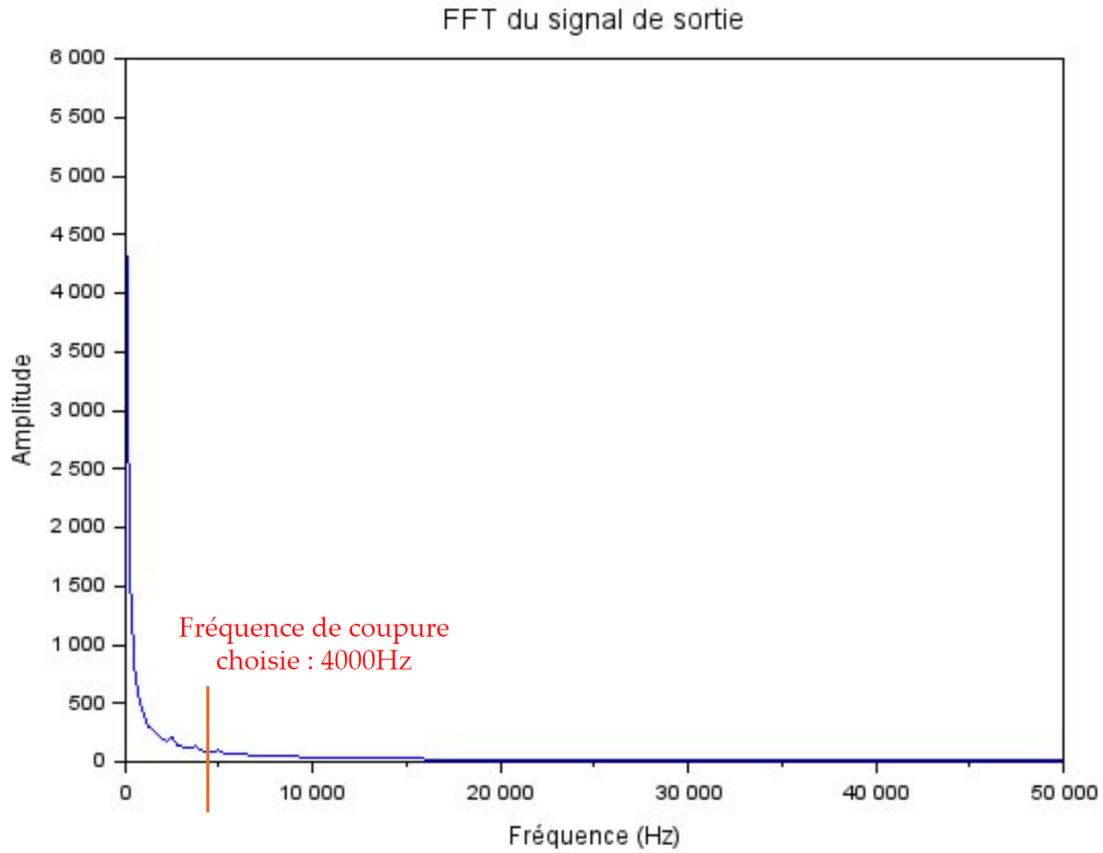


FIGURE 31 – FFT de la réponse du système asservi non bruitée

On observe alors que seules les basses fréquences sont utiles et nous choisissons donc un filtre passe-bas d'ordre 2 dont la fréquence de coupure a été trouvée expérimentalement à 4000Hz .

Lorsque l'on simule l'asservissement avec l'ajout du filtre, on obtient alors les résultats de la figure 32 :

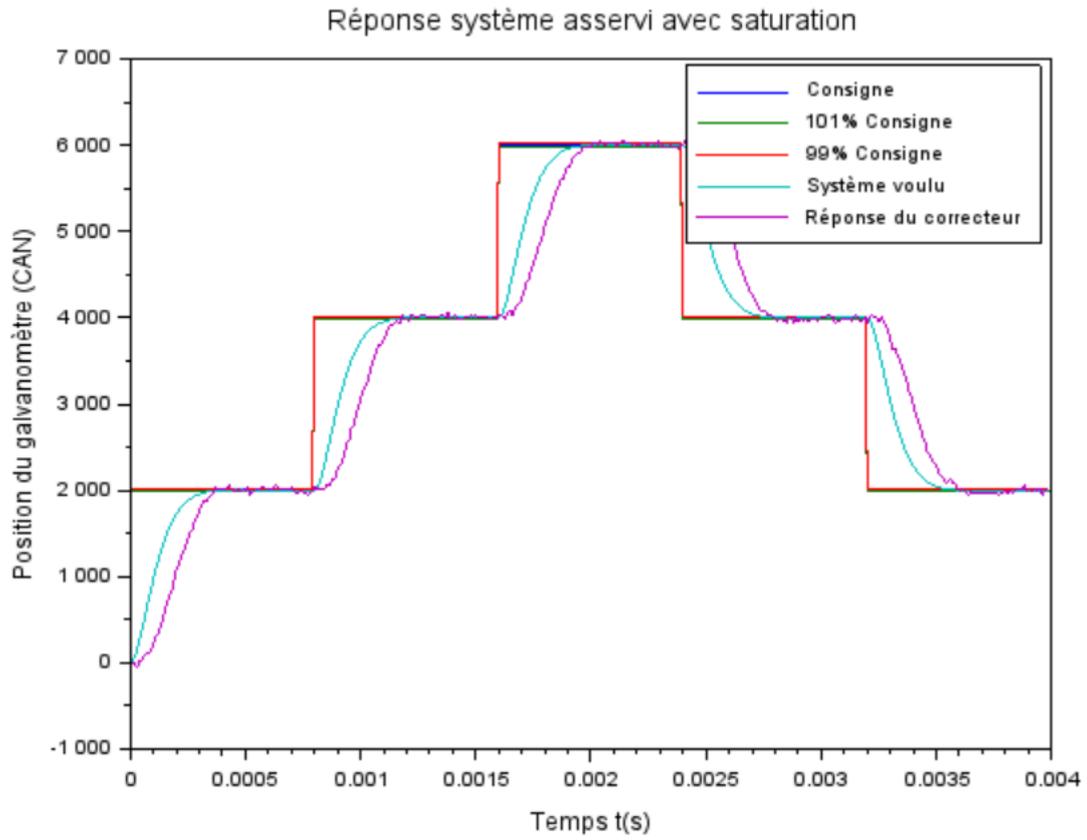


FIGURE 32 – Réponse du système asservi avec bruit de mesure filtré

On constate que l’asservissement suit de nouveau la consigne, avec une légère déformation de la réponse du système car filtrée mais si l’on s’intéresse au temps stable à 1% figure 33, on obtient les valeurs suivantes :

Réponse du système sans bruit, avec bruit moyen et avec bruit max

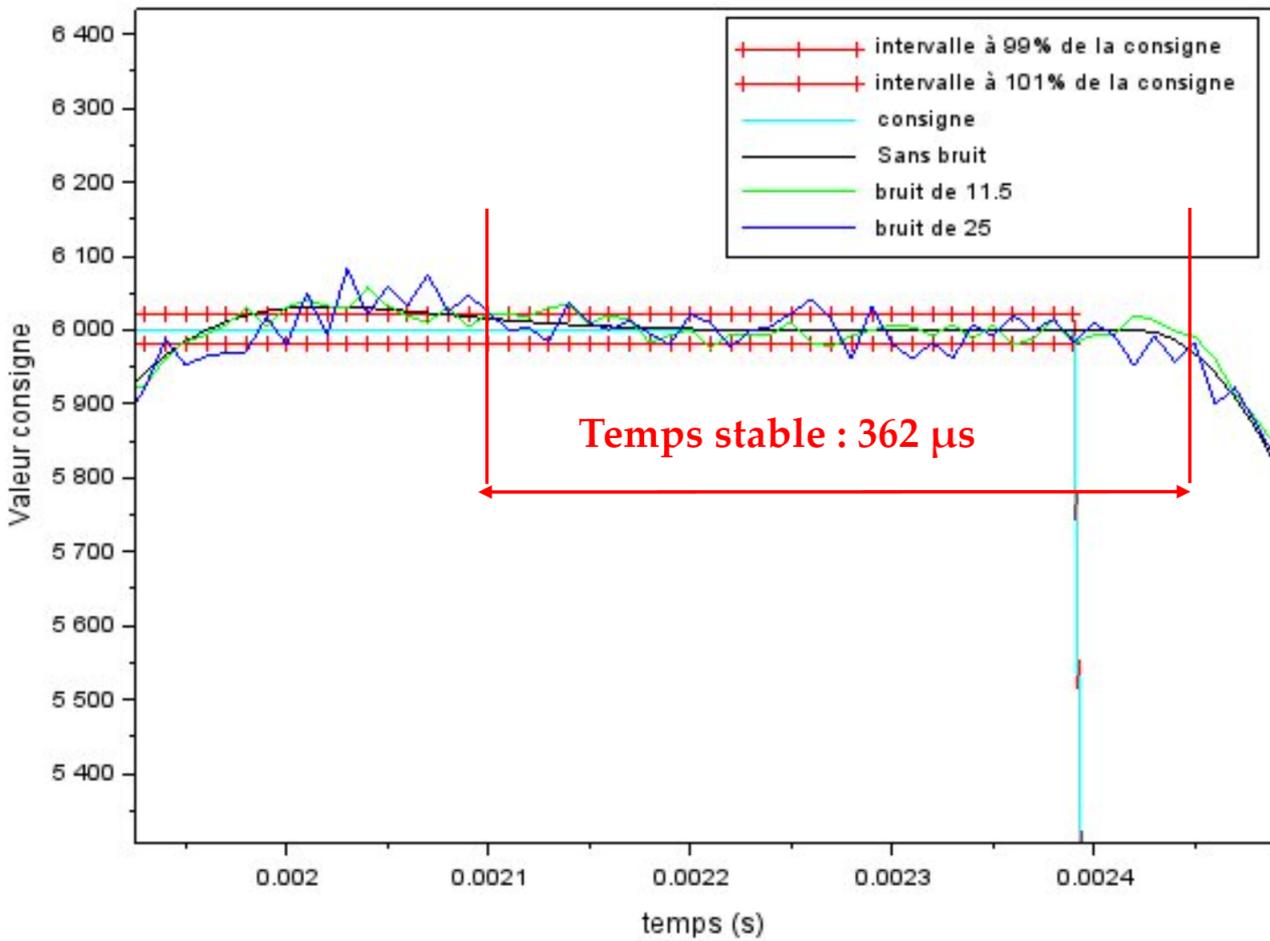


FIGURE 33 – Zoom sur le temps stable de la réponse du système asservi avec bruit de mesure filtré

Le temps stable à 1% est inférieur à $400\mu s$ mais cette valeur avait été choisie en prenant une marge et certains cahiers des charges évoquaient une valeur de $280\mu s$. Il faudrait donc voir directement si la matrice scintille avec cette valeur.

Si l'on regarde la nouvelle commande figure 34, on obtient :

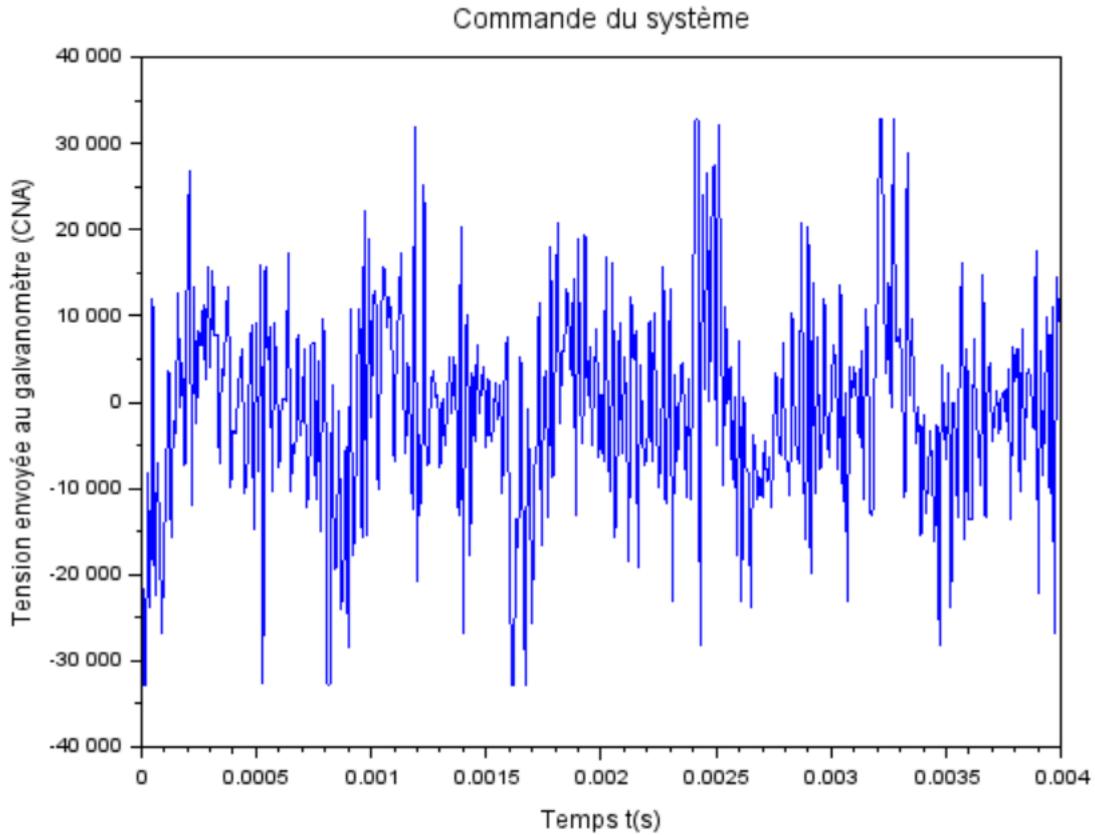


FIGURE 34 – Commande du système asservi avec bruit de mesure filtré

On constate alors que le filtre a bien diminué les problèmes de saturation de la commande.

Pour conclure ce chapitre, nous pouvons valider le fonctionnement de notre correcteur en simulation, avec la prise en compte de la saturation et du bruit de mesure, ce qui est un avancement par rapport à l'année dernière. Nous avons également pu observer l'importance d'avoir un modèle du système à asservir le plus représentatif du système réel possible mais qu'un certain degré d'imprécision pouvait être compensé par le choix de la période d'échantillonnage de l'asservissement.

6 Implémentation du correcteur RST et tests

6.1 Test RST existant

Dans un premier temps, dès lors que l'identification est terminée, le correcteur RST réalisé l'an passé est testé. Ainsi, le correcteur est implémenté sur la carte et les résultats obtenus sont comparés à la théorie et les résultats ne correspondent pas.

Pour ces tests le mode debug est utilisé, ce mode permet de vérifier les résultats cycle par cycle.

Compte tenu de l'échec des tests, le correcteur est recréé depuis zéro. Cela permettra d'avoir une meilleure compréhension du code et donc de pouvoir déboguer plus facilement. L'interface des années précédentes est conservée, ainsi l'implémentation sera plus facile et uniquement la partie calcul du RST est refaite.

6.2 Conception du RST

Pour réaliser le correcteur RST en VHDL, il n'est pas possible de simplement inclure une équation comme en théorie. Ainsi, une méthode comprenant des registres à décalage est envisagée. Cette méthode permet de stocker les résultats intermédiaires et donc de réaliser le calcul sur plusieurs cycles.

Cependant, avant de réaliser directement ce correcteur en VHDL, cette façon de calculer est codée sur scilab puis simulée afin d'être comparé à la théorie.

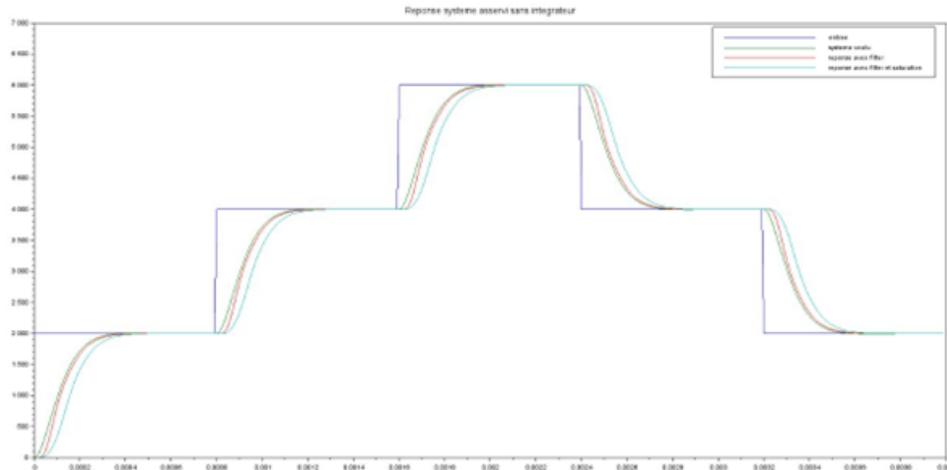


FIGURE 35 – Comparaison méthode calcul avec théorie

Cette simulation nous permet de confirmer que la méthode de calcul envisagée est viable car l'allure des courbes, figure 35, est identique.

6.3 Correcteur RST en VHDL

Il faut donc réaliser ce correcteur en VHDL. Dans un premier temps, les composants primaires sont réalisés, ensuite les blocs du RST sont réalisés séparément. Ces blocs sont réalisés de façon à ce qu'ils puissent fonctionner en autonomie.

Ainsi il a été possible de faire des tests intermédiaires permettant de valider le fonctionnement des blocs séparément avant leur assemblage.

Par exemple voici les tests effectués sur le bloc T sur la figure 36.

Pour ces tests intermédiaires les coefficients sont pris de manière à ce que le calcul puisse se faire de tête afin de pouvoir vérifier instantanément le résultat.

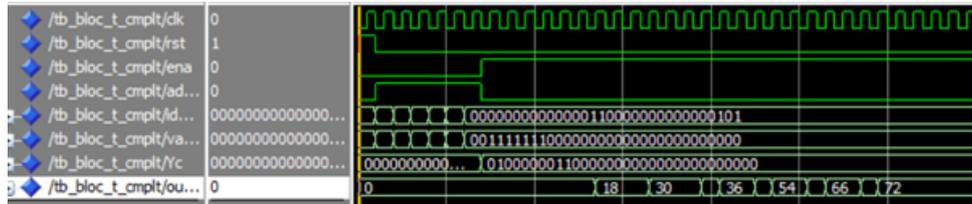


FIGURE 36 – Simulation Bloc T

Ce test correspond à une simulation réalisée sur ModelSim, ainsi ModelSim va simuler le comportement du RST codé.

Pour cette simulation, les coefficients choisis sont 3-2-1-3-2-1 et la consigne appliqué est 6.

Ainsi pour le cycle, il est obtenu 18 ce qui correspond bien à 3×6 , puis 36 pour $3 \times 6 + 2 \times 6$, et ce pour tous les cycles de calculs.

Cependant, actuellement les blocs sont testés en simulation, afin d’élargir les tests, les blocs sont implémentés seuls sur la carte dans des conditions réelles afin de valider définitivement leurs fonctionnements.

Ensuite, le mode debug est utilisé pour pouvoir vérifier le résultat des blocs cycles par cycles. Pour ces tests, SignalTap est utilisé, cet outil permet de visualiser la valeur des variables interne directement sur le FPGA.

Par exemple avec le bloc T, les coefficients envoyés à la carte sont 1-2-3-4-5-6 et la consigne est 3.

Le premier cycle (figure 37) donne le résultat suivant :

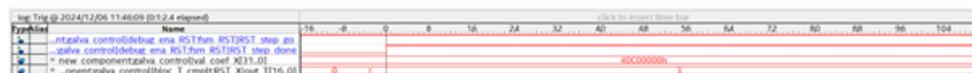


FIGURE 37 – Cycle 1

Le résultat correspond bien à 1×3 .

Pour le deuxième cycle (figure 38), $1 \times 3 + 2 \times 3$ est bien obtenu.

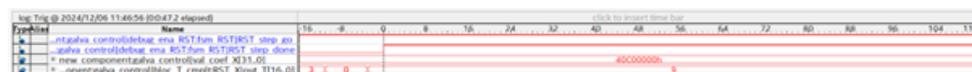


FIGURE 38 – Cycle 2

Et enfin, pour le troisième cycle (figure 39), en théorie il faut ajouter 3×3 au résultat précédent, c'est bien ce qui est obtenu en réel.



FIGURE 39 – Cycle 3

Ces tests permettent de valider le fonctionnement des blocs afin de créer le correcteur complet.

6.4 Assemblage du correcteur RST

Les trois blocs précédemment testés sont assemblés pour former le correcteur RST.

Une fois le correcteur créé puis implémenté sur la carte. Ce correcteur est testé de la même façon que pour les blocs, c'est à dire en simulation et en réel grâce au mode debug.

Cependant plutôt que de prendre des coefficients aléatoire, les coefficients issus de l'identification des galvanomètres sont utilisés. Ainsi, les tests effectués seront plus représentatifs des conditions réelles d'utilisation des galvanomètres.

De fait, les résultats ne sont plus calculables directement de tête, ainsi, ils sont produits sous scilab.

Voici les résultats théoriques obtenus sur scilab :

```

yt : 10.000000      ys : 10.000000      yg : 0.000000
yt : 10.000000      ys : -16.946300     yg : 0.000000
yt : 10.000000      ys : 65.090288     yg : 0.000000
yt : 10.000000      ys : -153.848301   yg : 0.000000
yt : 10.000000      ys : 439.283957    yg : 0.000000
yt : 10.000000      ys : -1139.600972  yg : 0.000000
yt : 10.000000      ys : 3071.493800   yg : 0.000000
yt : 10.000000      ys : -8131.849674  yg : 0.000000
yt : 10.000000      ys : 21681.420272  yg : 0.000000
yt : 10.000000      ys : -57625.963765 yg : 0.000000
yt : 10.000000      ys : 153349.314264 yg : 0.000000
    
```

FIGURE 40 – Résultat théorique

La deuxième colonne de la figure 40, indique la sortie du bloc S et donc du correcteur RST, cette colonne va ensuite être comparée aux simulations.

Dans un premier temps la simulation sur ModelSim est réalisée figure 41 :

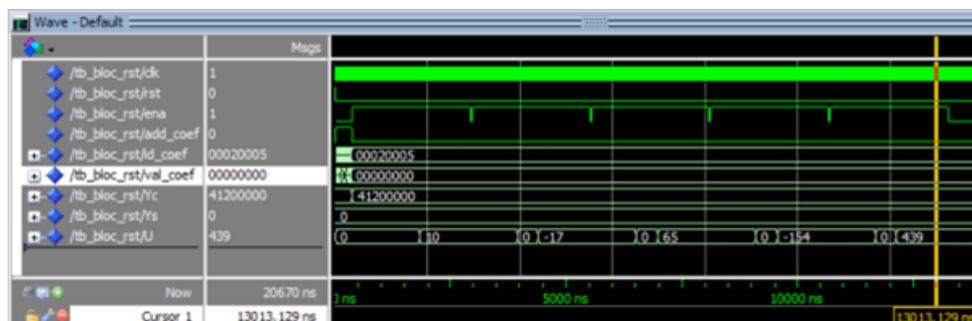


FIGURE 41 – Simulation ModelSim

Sur cette simulation, il est possible de voir que les valeurs de U correspondent aux valeurs de sortie du RST sont identiques aux valeurs théoriques, aux arrondis

près.

Enfin, le dernier test consiste à vérifier si les résultats restent les mêmes en réel.

Ainsi, sur SignalTap les résultats obtenus sont les suivants(figures42, 43, 44 et 45) :



FIGURE 42 – SignalTap cycle 1

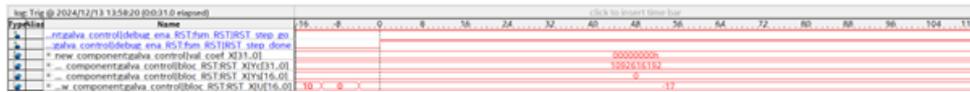


FIGURE 43 – SignalTap cycle 2



FIGURE 44 – SignalTap cycle 3



FIGURE 45 – SignalTap cycle 4

Une nouvelle fois les résultats concordent, ces tests permettent de vérifier le fonctionnement du correcteur sur un exemple avec des coefficients représentant le comportement réel du galvanomètre.

Ces tests montrent la correspondance entre le VHDL et la simulation scilab, cependant ils ne sont pas exhaustifs puisqu'ils ont été réalisés sur un nombre réduit de cycles.

6.5 Validation correcteur RST en VHDL avec la simulation

Ainsi, afin d'accroître la fiabilité de nos tests, un script scilab est écrit, ce script calcule les coefficients et les résultats théoriques. En parallèle, le script envoie les valeurs au correcteur RST en VHDL, enfin le script va lire les résultats émis par le VHDL.

Enfin, le script va tracer les courbes obtenues avec le correcteur en VHDL et avec la simulation théorique.

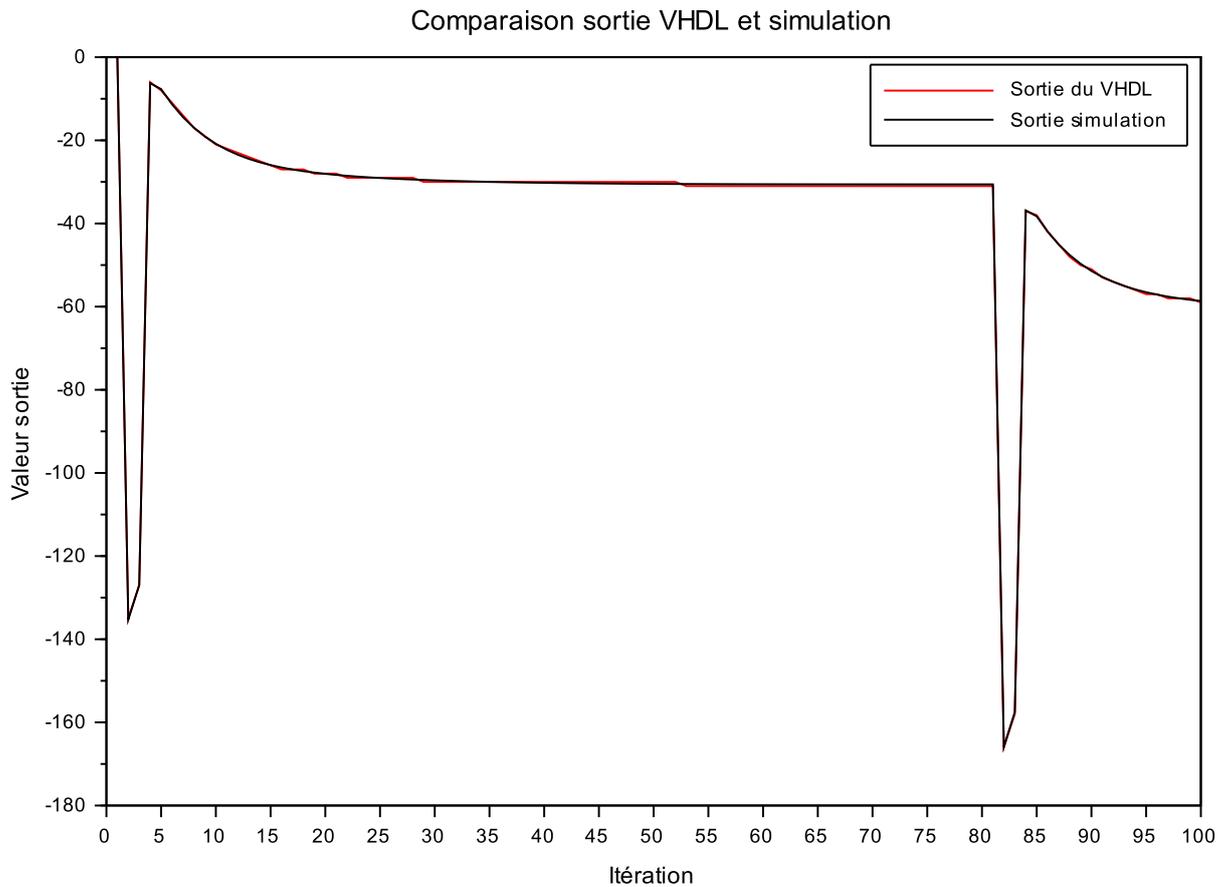


FIGURE 46 – Test final

Le test réalisé figure 46 a été fait sur 100 cycles.

On constate que les courbes coïncident à l'unité près, ce qui permet de valider le correcteur en VHDL sur un test plus complet.

Plus d'informations relatives à la conception du correcteur en RST sont disponibles dans la note d'application "Conception du correcteur RST en VHDL" [10].

Conclusion

Pour conclure ce rapport, nous ne sommes pas en mesure de prédire si l'identification réalisée sera suffisamment fidèle au système réel pour que l'asservissement fonctionne comme souhaité mais pour le cas où ce ne serait pas le cas, nous pouvons dire que le travail fourni offre une base de simulation complète permettant de tester différents types de correcteurs RST à placement de pôles, avec des conditions de simulation diverses permettant de rapidement obtenir de nouveaux coefficients à tester sur la carte en VHDL, qui maintenant calculera des commandes identiques à celles obtenues en simulation.

L'affichage de la matrice quant à lui n'a pas pu être réalisé à cause d'un problème sur l'aiguillage de la commande calculée par le RST en FPGA vers les CANs qui n'a pas pu être résolu par manque de temps.

Bilan personnel

Pour clore ce rapport de projet sur une appréciation personnelle, nous avons pris beaucoup de plaisir à essayer de formuler des hypothèses, les tester, les valider ou non... car l'ensemble des domaines abordés nous passionnent.

Nous avons constaté la difficulté de reprendre un projet avancé par d'autres et l'importance de se réunir pour définir et planifier les tâches à réaliser en début de projet, mais également pour rectifier les prévisions au fur et à mesure de nos résultats. La communication au sein du groupe était excellente ce qui s'est ressenti en termes de productivité.

Concernant les compétences techniques acquises, nous avons pu grandement développer nos compétences sur l'utilisation de Scilab, mettre en pratique sur un cas beaucoup plus complexe les notions d'automatique vues en cours et également découvrir pour certains ou développer pour d'autres la programmation en VHDL d'une carte Nios II. Enfin, nous avons pu mettre à l'épreuve nos capacités d'adaptation et d'autonomie pour réagir face aux imprévus et être forces de propositions.

Références

- [1] Léa Duvivier. *Note d'Application Partie Contrôle : Identification*. Note d'application. Polytech Clermont. 2024. Disponible sur : <https://forge.polytech.uca.fr/documents/1808> (2024).
- [2] Tristan Podgorski. *Note d'application Partie Contrôle : RST*. Note d'application. Polytech Clermont. 2024. Disponible sur : <https://forge.polytech.uca.fr/documents/1817> (2024).
- [3] Damien Régent. *Note d'application sur le fonctionnement du système. les registres et l'identification matérielle*. Note d'application, Polytech Clermont. 2024. Disponible sur : <https://forge.polytech.uca.fr/documents/1871> (2024).
- [4] Clara Soulard. *Note d'application sur le fonctionnement du bloc RST et du mode debug*. Note d'application. Polytech Clermont. 2024. Disponible sur : <https://forge.polytech.uca.fr/documents/1819> (2024).
- [5] L. Duvivier, T. Podgorski, D. Régent et C. Soulard. *Soutenance 23/01/2024*. Support de présentation. Polytech Clermont. 2024. Disponible sur : <https://forge.polytech.uca.fr/documents/1820> (2024).
- [6] Marouane Hsaini. *Partie Contrôle : Identification SBPA*. Note d'application. Polytech Clermont. 2025. Disponible sur : <https://forge.polytech.uca.fr/documents/1869> (2025).
- [7] Sébastien Lengagne. *Control of complex systems : Galvanometer for the laser eye surgery*. Support de présentation. Polytech Clermont. [s.d.].
- [8] Yoan Douarre. *Conception théorique d'un correcteur RST par placement de pôles*. Note d'application. Polytech Clermont. 2025. Disponible sur : <https://forge.polytech.uca.fr/documents/1870> (2025).
- [9] Sébastien Lengagne. *Correction des systèmes linéaires échantillonnés*. Support de cours. Polytech Clermont. [s.d.]. Disponible sur : https://ent.uca.fr/moodle/pluginfile.php/1384211/mod_resource/content/2/Auro3seance.pdf (2024).
- [10] Mathis Pascal. *Conception du correcteur RST en VHDL*. Note d'application. Polytech Clermont. 2025. Disponible sur : <https://forge.polytech.uca.fr/documents/1872> (2025).
- [11] M. Rivoire et J-L. Ferrier. *Cours d'automatique*. Tome 2. [Nouvelle Édition]. Paris : Eyrolles. 1996. 172p.

Annexes

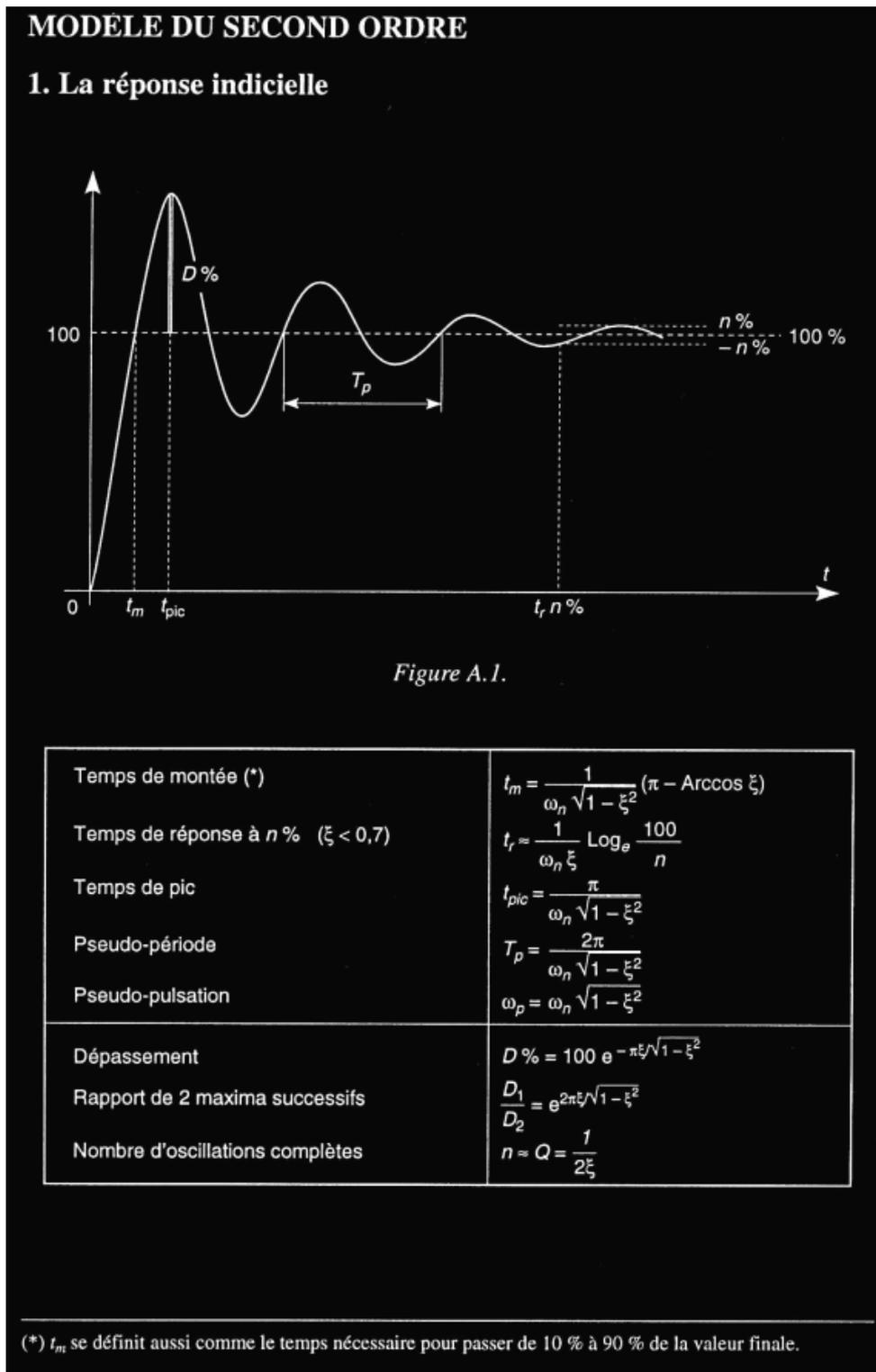
Table des annexes

Annexe 1 : Modèle du second ordre	I
Annexe 2 : Tableau des valeurs numériques des paramètres du second ordre	II

L'ensemble des codes utilisés sont disponibles sur la forge à cette adresse :

<https://forge.polytech.uca.fr/projects/p24ab01-2/repository/3864/show/trunk/Scilab>

Annexe 1 : Modèle du second ordre



Annexe 1 - Modèle du second ordre [11]

Annexe 2 : Tableau des valeurs numériques des paramètres du second ordre

Le tableau des valeurs numériques

Amortissement inférieur à 1										
ξ	Paramètres temporels					Paramètres fréquentiels				ξ
	$t_m \omega_n$	$t_r \omega_n$ (5 %)	$t_{pic} \omega_n$	$T_p \omega_n$	D %	$\frac{\omega_R}{\omega_n}$	$\frac{\omega_c}{\omega_n}$	$\frac{\omega_c}{\omega_R}$	M_{dB}	
0,1	1,68	30	3,16	6,31	73	0,99	1,54	1,56	14	0,1
0,15	1,74	20	3,18	6,36	62	0,98	1,53	1,56	10,5	0,15
0,2	1,81	14	3,21	6,41	53	0,96	1,51	1,57	8,1	0,2
0,25	1,88	11	3,24	6,49	44	0,94	1,48	1,59	6,3	0,25
0,3	1,97	10,1	3,29	6,59	37	0,91	1,45	1,61	4,8	0,3
0,35	2,06	7,9	3,35	6,71	31	0,87	1,42	1,63	3,6	0,35
0,4	2,16	7,7	3,43	6,86	25	0,82	1,37	1,67	2,7	0,4
0,45	2,28	5,4	3,52	7,04	21	0,77	1,33	1,72	1,9	0,45
0,5	2,42	5,3	3,63	7,26	16	0,71	1,27	1,80	1,2	0,5
0,55	2,58	5,3	3,76	7,52	12,6	0,63	1,21	1,93	0,7	0,55
0,6	2,77	5,2	3,93	7,85	9,5	0,53	1,15	2,17	0,3	0,6
0,65	3,00	5,0	4,13	8,27	6,8	0,39	1,08	2,74	0,1	0,65
0,7	3,29	3	4,40	8,80	4,6	0,14	1,01	7,14	0	0,7
0,75	3,66	3,1	4,75	9,50	2,84	–	0,94	–	–	0,75
0,80	4,16	3,4	5,24	10,5	1,52	–	0,87	–	–	0,80
0,85	4,91	3,7	5,96	11,93	0,63	–	0,81	–	–	0,85
0,90	6,17	4	7,21	14,41	0,15	–	0,75	–	–	0,90
0,95	9,09	4,1	10,06	20,12	0,01	–	0,69	–	–	0,95

Annexe 2 - Tableau des valeurs numériques des paramètres du second ordre [11]