MINISTERE DE L'ENSEIGNEMENT SUPERIEUR

ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE FERHAT ABBAS-SETIF1

UFAS1 (ALGERIE)

MEMOIRE

Présenté à la Faculté de Technologie

Département d'Electrotechnique

Pour l'Obtention du diplôme de

MAGISTER

Option : Automatique

Par

Mr : BELHADJ DJALLEL

THEME

Commande des systèmes temps réel, Implémentation sur la carte dSPACE1104

Soutenu le : 23 / 06 / 2013 devant le jury composé de :

RAHMANI LAZHAR	Prof à l'université de Sétif	Président
MOSTEFAI MOHAMMED	Prof à l'université de Sétif	Rapporteur
KHEMLICHE MABROUK	Prof à l'université de Sétif	Examinateur
LAMAMRA ATHMANE	Dr à l'université de Sétif	Examinateur

Remerciements …

Je remercie tout premièrement Dieu pour la volonté, la santé et la patience qu'il m'a donné durant ces années de travail.

Je remercie ensuite les membres du jury pour avoir accepté d'examiner ce travail : M. Rahmani Lazhar, M. Khemlich Mabrouk, M. Lamamra Athmane.

Je tiens aussi à remercier tout particulièrement mon encadreur M. Mostefai Mohammed qui m'a suivi tout au long de ce travail.

Je remercie très sincèrement les ingénieurs de notre laboratoire, M. Babas Badredine et M. Idriss Azizi qui m'ont assisté avec leur compétence remarquable.

Mes remerciements vont aussi aux enseignants des départements d'Electrotechnique et d'Anglais spécialement M^{me} et M. Khemliche, et M. Rafik Mosbah.

Je tiens à remercier vivement M. Babas Walid, M. Zitouni Marwan, et M. Khabbach Hicham qui m'ont aidé à élaborer et réaliser ce mémoire, ainsi que M. Hamdi Mohamed mehdi, Issam Ouzal, Foued Nedjma, Rami Mihoubi, Hichem Sadeddin, Bizak nawi, Jalaliw mazoz, Soraya, Hanane, MahorBacha, Ayoub Bouhezila, Sibyle Novack et Sara Bouharkat, pour leur soutien moral et matériel.

En fin, j'adresse mes chaleureux remerciements à tous mes chers collègues.

Dédicaces

Je dédie ce travail à :

Mes très chers parents à qui je n'arriverai jamais à exprimer ma gratitude et ma reconnaissance pour leurs amours et soutiens durant mes études.

Mes chers frères et sœurs, à toute ma famille.

Tous les amis du D81 et d'Elhidhab

Tous mes amis sans exception.

Tous ceux qu'ils ont aimé me voir ce jour-là ..

Djallel

Résumé : Ce mémoire présente une étude de faisabilité de l'implémentation de plusieurs techniques de commande des systèmes temps réel sur la carte dSPACE 1104. Des algorithmes issus des principales catégories de techniques de commandes ont été développés en utilisant un environnement complet composé de SimulinK, MATLAB et ControlDesk : La commande LQR pour la catégorie des contrôleurs basés sur la linéarisation des systèmes non linéaires autour d'un point de fonctionnement, la commande ISMC pour la catégorie des contrôleurs qui se basent sur les dynamiques non linéaires des systèmes, la commande par réseau de neurones pour la catégorie des contrôleurs basés sur l'intelligence artificielle, et la commande robuste adaptative par backstepping, laquelle est développée afin d'étudier l'aspect adaptatif. Chaque algorithme est appliqué sur le système du pendule inversé dans le cadre d'une simulation, implémenté et testé sur une plateforme réelle. Les résultats obtenus sont discutés et comparés.

Table des matières

Liste des figures et des tableaux

Chapitre 1

Chapitre 2

Chapitre 4

Chapitre 5

Chapitre 6

Chapitre 7

Chapitre 8

Symboles

M	*La masse du chariot.*
Beq	*Coefficient d'atténuation visqueux équivalent du chariot.*
Mp	*La masse du pendule.*
lp	*La demi-longueur du pendule.*
Ip	*Moment d'inertie du pendule.*
Bp	*Coefficient d'atténuation visqueux du pendule.*
g	*Force de gravité.*
Vm	*Tension d'armature du moteur.*
Im	*Courant d'armature du moteur.*
Rm	*Résistance d'armature du moteur.*
Jm	*Moment d'inertie du rotor.*
η_m	*Rendement du moteur.*
K_m	*Constante de la Force Electromotrice (FEM).*
η_g	*Rendement de la boite de vitesse.*
K_g	*Rapport de la boite de vitesse.*
K_t	*Constante du couple du moteur.*
r_{mp}	*Rayon Du Pignon Du Moteur.*
α, θ	*Angle du pendule.*
F_c, f	*Force exercée.*
E_c, E_p	*L'énergie cinétique du système, l'énergie potentielle.*
T_c, T_p	*L'énergie cinétique du chariot, l'énergie cinétique du pendule.*
T_{cc}	*L'énergie cinétique de translation du chariot.*
T_{cp}	*L'énergie cinétique rotationnelle pour la motorisation.*
y_p, x_p	*Les coordonnées du centre de gravité du pendule.*
x_c	*Distance de translation du chariot.*
T_{pt}	*L'énergie cinétique de translation.*
T_{pr}	*L'énergie cinétique rotationnelle.*
L	*Coefficient de Lagrange.*
E_c	*L'énergie cinétique totale.*
X	*vecteur d'état.*
P(t)	*solution de l'équation de Riccati.*
K	*gain du régulateur.*
Q, R	*matrices de pondération du régulateur LQR.*
u	*La commande du système.*

s	*Surface de glissement.*
u_{eq}, u_{gliss}	*Commande équivalente, commande régime glissant.*
$f(x), g(x), b_i(x)$	*Fonctions non linéaires du système.*
λ_i	*Paramètres de la surface de glissement.*
e, e_i	*Les erreurs.*
x_d	*État désiré du système.*
Δ_s	*Le gradient de s.*
$<, >$	*Le produit scalaire.*
Φ	*Paramètre de la fonction Saturation « Sat ».*
y	*La sortie du système.*
y_d, y_r, x_{ref}, x_d	*Signaux de référence.*
v, V_i	*Fonction de Lyapunov.*
u_{sw}	*Commande de commutation.*
C_i	*Constantes.*
X_e	*Point d'équilibre du système.*
l	*Longueur du pendule.*
Ψ_i	*Paramètres inconnus du système.*
z_i, x_i	*Variables d'état.*
α_i	*Fonctions des variables d'erreur du backstepping.*
h	*Fonction normalisée des paramètres inconnus du système.*
$K(z_i), g(z_i)$	*Fonctions des paramètres inconnus du système.*
θ_x	*Coefficient de compensation angulaire.*
θ_{ref}	*Angle de référence avec compensation.*
α_{ref}, θ_d	*Angle de référence.*
z_{ref}	*Loi de commande virtuelle.*
$\hat{g}, \hat{h}$	*Paramètres adaptatifs.*
$\bar{g}, \bar{h}$	*Erreurs des paramètres adaptatifs.*
σ_{sh}, σ_{gs}	*Fonctions continues de commutation pour l'adaptation.*
$\sigma_{g0}, \sigma_{h0}, g_0, h_0$	*Constantes de conception du mécanisme d'adaptation.*
t, t_i	*Indices de temps.*
Sec	*La fonction trigonométrique «la sécante ».*
E_{max}	*Erreur maximale.*
ϱ	*Fonction du pas d'apprentissage.*
ξ	*Coefficient d'amortissement.*
Lcmd	*loi de commande par Backstepping sans le mécanisme d'adaptation.*

Symboles spécifiques aux réseaux de neurones « RNA »

Y^d	*Vecteur de sortie désirée.*
T	*Longueur de l'ensemble d'apprentissage.*
W_{ij}^k	*Poids associé à chacune des connexions neuronales.*
Sn	*Nombre de neurones d'une couche neuronale.*
R	*Dimension du vecteur d'entrées pour une couche de neurones.*
W_{i0}^k	*Seuil adaptable du neurone i.*
m	*Nombre d'entrées du réseau de neurones.*
ni	*Nombre de sorties du réseau de neurones.*
L	*Nombre de couches.*
k	*Indice de couche.*
$f^k(.), \varphi^{(k)}$	*Fonction d'activation de la couche k.*
n_{k-1}	*Nombre de neurones de la couche k.*
$O_i^k(t)$	*Sortie du neurone i de la couche k.*
x , y	*Vecteur d'entrée, vecteur de sortie.*
$X_i(t)$	*$i^{ème}$ composante du vecteur d'entrée.*
$Y_i(t)$	*$i^{ème}$ composante du vecteur de sortie.*
E	*Fonction coût.*
μ	*Pas d'apprentissage.*
n	*indice de l'itération.*
$\delta_i^k(t)$	*l'erreur équivalente à la sortie du neurone i de la couche k.*
$v_i^{(k)}$, S_k^i	*Entrées des fonctions d'activations.*
F_c	*Coefficients de Coulomb.*
F_v	*Coefficients viscosité.*
F_s	*Coefficients de frottement statique.*
η_s	*Taux de décroissance du frottement statique.*
e_x	*Erreur de position linéaire.*
e_θ	*Erreur d'angle.*
ψ_x, ψ_θ	*Gains du modèle de référence.*
S	*Sortie du modèle de référence.*
ncc	*Nombre de couches cachées.*
K_d	*Gain du coefficient de robustesse.*
F_r	*Coefficient de robustesse.*
F_{RNN}	*Commande de force estimée par le réseau de neurones.*

INTRODUCTION

Le système temps réel basé sur la technologie PowerPC et son jeu d'interfaces d'E/S font de la carte contrôleur dSPACE1104 une solution idéale pour le développement des contrôleurs dans divers domaines, tels que, la robotique, l'aérospatial… etc.

Les techniques de simulation et du prototypage rapide des commandes sont de plus en plus utiles dans divers domaines aussi variés que l'automobile, l'aéronautique, les énergies renouvelables…etc., afin d'avoir un développement plus rapide et plus efficace en évitant le processus coûteux du prototypage physique.

Dans ce cadre, de nombreux concepts et outils informatiques permettent aux industriels de spécifier et concevoir des systèmes informatiques temps réel dédiés à la commande de processus. A l'échelle industrielle, ce sont des applications largement utilisées, car elles permettent au système étudié de respecter des règles de sécurité et de fiabilité.

Une application temps réel est indispensable au bon fonctionnement d'un processus de fabrication ou de surveillance, car elle contrôle l'environnement de travail tout en collectant des informations. Ces informations sont ensuite traitées, puis le système temps réel interagit avec l'application pour retourner des résultats. Ces résultats, envoyés le plus rapidement possible, permettent de piloter ou de modifier certains paramètres afin d'optimiser le fonctionnement de l'application.

Le produit dSPACE offre l'intégralité de ces concepts et outils, ce qui facilite toutes les phases de conception de commandes des processus, de la simulation sous Matlab/Simulink à la génération/chargement du code C dans la carte, tout en passant par le Prototypage Rapide de Commande (RCP) et le Hardware In the Loop (HIL).

Le but de ce travail est d'étudier la faisabilité d'implémentation des différentes techniques de commande sur la carte dSPACE.

Organisation du manuscrit :

En commençant par présenter des notions d'initiation sur les systèmes, la commande des systèmes et l'aspect du temps réels dans le chapitre 1.

Ensuite dans le chapitre 2 on effectue la description et la modélisation d'un système de pendule inversé IP02.

Les chapitres 3, 4, 5 et 6 sont consacrés à la synthèse d'une variété de différentes techniques de commande, chaque technique est choisie pour représenter une catégorie parmi les principales catégories des systèmes de commande.

Dans le 7eme chapitre une brève description de l'environnement de développement de commandes Matlab/Simulink/dSPACE est présentée. Puisque les logiciels Matlab et Simulink sont très connus, on va s'intéresser au produit dSPACE. Nous présentons dans un premier temps le matériel (carte) dSPACE 1104. Ensuite, on va présenter deux logiciels associés à la carte dSPACE : le logiciel de commande et de visualisation ControlDesk et l'interface en temps réel (RTI).

Dans le dernier chapitre, on procédera à l'application des techniques de commande sur le pendule inversé et l'implémentation sur plateforme réelle en utilisant la carte dSPACE1104. Les résultats sont présentés, discutés et comparés. Et enfin on termine par une conclusion.

Chapitre 1

Commande des systèmes temps réel

1.1. Introduction [1]

Jusqu'à une date récente, la commande des systèmes temps réel était destinée quasi-exclusivement aux applications de procédés (ou de phénomènes) physiques (tels que les usines de fabrication de voitures) et applications militaires. Grace au développement scientifique, la commande des systèmes temps réel est présente aujourd'hui dans des domaines très variés comme le montre la liste suivante, même si elle n'est pas exhaustive :

- télécommunications (transport de la parole, systèmes de commutation, …),
- domaine médical (assistance et supervision de malades, …),
- contrôle de différents équipements dans les voitures, bus, trains, avions, …,
- contrôle et régulation de trafic en milieu urbain,
- guidage de véhicules en milieu urbain,
- industrie (contrôle/commande de procédés…),
- domaine militaire (suivi de trajectoires de missiles, …)
- aérospatial (suivi de satellites, simulation de vols, pilotage automatique, …),
- multimédia (transport d'images et de voie, téléconférences, …),
- finance (suivi du cours des devises et actions, ...),
- domotique (sécurité d'habitations, …),
- contrôle/commande d'appareils électroménagers.

On veut, dans ce chapitre, clarifier le concept de contraintes de temps qui caractérise la commande des systèmes temps réel. On s'attachera à donner quelques définitions de base sur ces types de systèmes, en suite on présentera la notion de temps réel ainsi que les différentes recherches et solutions pour leurs commandes.

1.2. Définition d'un Système [6]

Un système est une modélisation d'un procédé en fonctionnement. Il possède une ou plusieurs entrées, et une ou plusieurs sorties. Les sorties du système sont appelées variables commandées, mesures ou grandeurs réglées. Il existe une infinité d'exemples de systèmes : des systèmes mécaniques, des systèmes électriques ou des procédés chimiques,…etc. La représentation du système ne pourra alors se faire qu'avec de bonnes connaissances dans le domaine physique correspondant. Un système peut représenter soit : un ensemble d'éléments en interaction organisé pour un but précis, un ensemble de composants qui sont intégrés pour accomplir une tâche (un mécanisme), Un ensemble d'objets en interaction qui peut, en première approximation, être considéré comme avoir peu de relation avec l'extérieur (le système solaire), ou encore le contenu d'une boite noire mal connue par l'observateur. La notion de système est large, pour ne pas dire vague. Le seul point commun entre les systèmes métrique, un système d'équations ou le système solaire est, justement le mot système.

1.3. Classification des systèmes [2] [3] [4]

Les systèmes peuvent être classés en plusieurs catégories suivant différents critères :

1.3.1. Systèmes à temps continus

Ce sont les systèmes qui existent naturellement. Pour ces systèmes, le temps t décrit la droite réelle.

1.3.2. Systèmes à temps discret

Ce sont des systèmes pour lequel le temps est une variable discrète Sauf exception, ces systèmes n'existent pas à l'état naturel (la majorité des systèmes physiques naturels sont à temps continu), mais étant donné que la plupart des contrôleurs utilisés en automatique sont calculés par des processeurs numériques, il est parfois intéressant de modéliser le système commandé comme un système à temps discret.

1.3.3. Systèmes à événements discrets

C'est des systèmes dont le fonctionnement peut être modélisé par des événements discrets. Généralement, ces systèmes sont modélisés par des réseaux de Pétri. Des exemples sont les réseaux ferroviaires, ou le fonctionnement d'une chaîne de montage.

1.3.4. Systèmes hybrides

Ces systèmes dont la modélisation nécessite l'utilisation des techniques liées aux systèmes continus et aux systèmes à évènements discrets, par exemple : une boîte de vitesse de voiture.

1.3.5. Systèmes mono-variables, multi-variables

Quatre possibilités existent:

- le système à une entrée et une sortie, c'est un système monovariable ou SISO (*Single Input Single Output*),
- le système a plusieurs entrées et plusieurs sorties, c'est un système multivariable ou MIMO (*Multiple Input Multiple Output*),
- le système a une entrée et plusieurs sorties, système SIMO,
- le système a plusieurs entrées et une sortie, système MISO.

Néanmoins, ces deux derniers termes sont peu utilisés.

1.3.6. Système invariant (ou stationnaire)

Ce sont des systèmes dont les paramètres du modèle mathématique ne varient pas au cours du temps.

1.3.7. Système linéaire et non linéaire

1.3.7.1. Système linéaire

Un système est dit linéaire s'il est régi par des équations différentielles linéaires à coefficient constant, c'est-à -dire si sa fonction de transfert est une fraction rationnelles en p (ou p est la variable de Laplace). Cela implique ainsi :

- Si la réponse d'un système linéaire à un $x(t)$ est $y(t)$, sa réponse à l'entrée $kx(t)$ est $ky(t)$ (proportionnalité des effets aux causes).
- Si $y_1(t)$ *et* $y_2(t)$ sont les réponses respectives aux entrées $x_1(t)$ *et* $x_2(t)$ la réponse à l'entrée $x_1(t) + x_2(t)$ est $y_1(t) + y_2(t)$ (additivité).

1.3.7.2. Système non linéaire

On appel système non linéaire : un système qui ne peut pas être représenté par une équation différentielle linéaire à coefficients constants, c'est pour lequel le théorème de superposition ne s'applique pas.

On suppose que pour un système linéaire donné on a $y_1(t)$ est la réponse à l'entrée $x_1(t)$ et $y_2(t)$ est la réponse à l'entrée $x_2(t)$, alors la réponse à l'entrée $a_1\ x_1(t) + a_2\ x_2(t)$ où a_1 et a_2 sont des constantes, est: $a_1\ y_1(t) + a_2\ y_2(t)$ .Par exemple si l'équation du système non linéaire est : $\dot{y}(t) + y^2(t) = x(t)$ (1.1)

Alors : $$\dot{y}_1(t) + y_1^2(t) = x_1(t) \quad (1.2)$$

$$\dot{y}_2(t) + y_2^2(t) = x_2(t) \quad (1.3)$$

Donc : $$\begin{cases} \frac{d}{dt}\left[a_1\ y_1(t) + a_2\ y_2(t)\right] + \left[a_1\ y_1(t) + a_2\ y_2(t)\right]^2 \\ = a_1\ [\frac{dy_1(t)}{dt} + a_1\ y_1^2(t)]\ + a_2\ [\frac{dy_2(t)}{dt} + a_2 y_2^2(t))] + 2\ a_1 a_2 y_1(t) y_2(t) \\ \neq a_1\ x_1(t) + a_2\ x_2(t) \end{cases} \quad (I.4)$$

1.3.7.3. Les non linéarités

On peut distinguer deux types de non linéarité, la non linéarité essentielle, voulue ou faisant partie inhérente du système (relais, butées, hystérésis magnétique,…), et la non linéarité accidentelle, causée par l'imperfection des organes du système (frottement sec, jeu mécanique,…).

1.3.7.4. Différents cas de non linéarité

A. Plus ou moins idéal :

$$s = M.sign(e) \quad (1.5)$$

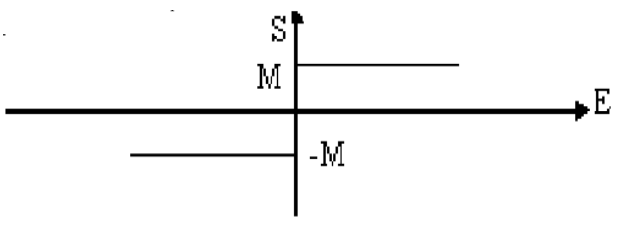


Figure 1.1. *Non linéarité Plus ou moins idéal.*

B. Par saturation :

Compression d'un ressort jusqu'à ce que les spires se touchent

$$s = \begin{cases} \alpha.e & pour\ |e| < e_M\ ;e_M = \dfrac{M}{\alpha} \\ M.sign(e) & sinon \end{cases} \qquad (I.6)$$

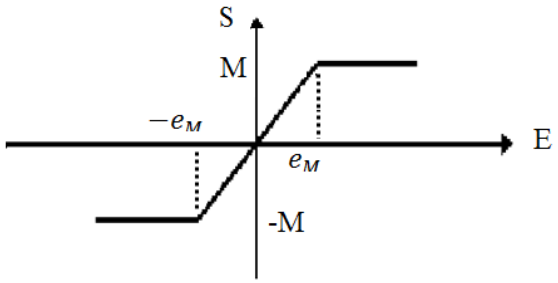


Figure 1.2. *Non linéarité Par saturation.*

C. Par hystérésis :

Magnétisation d'un acier, non élasticité des matériaux plastiques.

$$s = \begin{cases} -M & pour \quad e < \dfrac{h}{2} \\ M & sinon \end{cases} \qquad (I.7)$$

$$s = \begin{cases} M & pour \quad e > -\dfrac{h}{2} \\ -M & sinon \end{cases} \qquad (I.8)$$

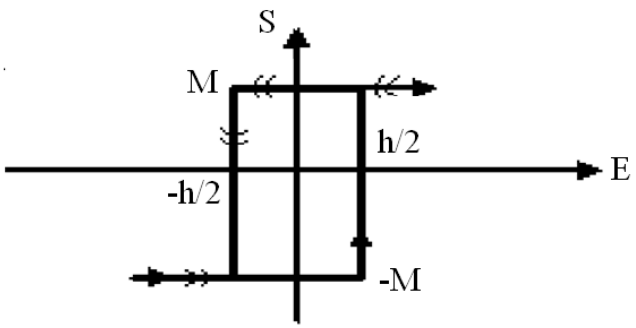


Figure 1.3. *Non linéarité Par hystérésis.*

D. Par adhérence effet de seuil :

Les actions exercées sur le système sont trop faibles pour provoquer une variation de la sortie.

$$s = \begin{cases} 0 & pour \ |e| < \frac{\Delta}{2} \\ \alpha(e - \frac{\Delta}{2}) & pour \ |e| > \frac{\Delta}{2} \\ \alpha(e + \frac{\Delta}{2}) & pour \ |e| < -\frac{\Delta}{2} \end{cases} \quad (I.9)$$

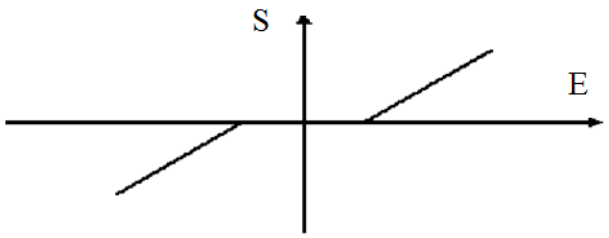


Figure 1.4. *Non linéarité Par adhérence.*

E. Par résonance :

Divergence des oscillations d'un système élastique non amorti.

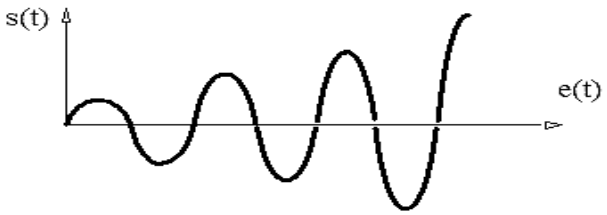


Figure 1.5. *Non linéarité Par résonance.*

1.3.8. Système complétement actionné, sous actionné [5]

On trouve ces termes généralement en mécanique. Un système est dit complètement actionné si le nombre des entrées de commande est égal au nombre de degrés de liberté pour les systèmes mécanique et égal au nombre de variables de sortie dans le cas général.

Les systèmes sous-actionnés sont des systèmes avec un nombre d'actionneurs moins que le nombre de degrés de liberté en mécanique, dans le cas général on a moins de variables de commande. Parmi les avantages de ces systèmes on trouve la réduction du poids et du coût et l'intervention d'un contrôleur sous actionné dans le cas de l'échec de l'un des actionneurs qui travaillent en parallèle.

Les types principaux d'un système sont résumés dans l'organigramme suivant :

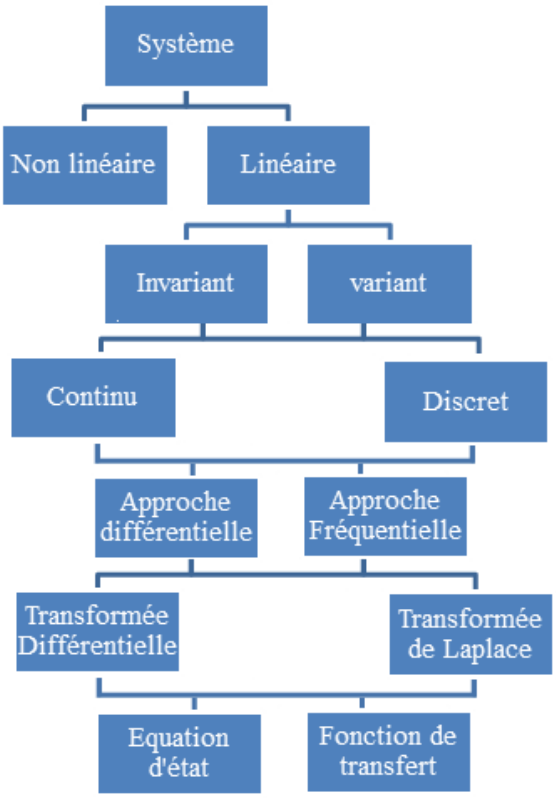


Figure 1.6. *Les types principaux d'un système.*

1.4. Notions du temps réel [7] [8]

On entend souvent qu'une application réalise une tâche donnée en « temps réel ». Cela signifie souvent que l'application est suffisamment performante pour traiter à la volée et sans perte d'information toutes les données provenant d'un ou plusieurs capteurs, sans atteindre la capacité maximale de calcul qu'elle a à sa disposition. Ainsi, toutes les données en entrée sont prises en compte par l'acquisition et participent à la tâche réalisée. Dans la plupart des cas, les données sont traitées avec une rapidité suffisante pour permettre l'obtention des résultats en un temps inférieur à la constante de temps régissant l'évolution du système perçu par les capteurs. Ceci permet à l'application de « réagir » aux évolutions de l'environnement.

1.5. Système temps réel

Un système est dit temps réel lorsque l'information après acquisition et traitement reste encore pertinente". Ce qui signifie que dans le cas d'une information arrivant de façon régulière (sous forme d'une interruption périodique du système), les temps d'acquisition et de traitement doivent rester inférieurs à la période de rafraîchissement de cette information.

Il est évident que la structure de ce système dépendra de ces fameuses contraintes. Un système temps réel n'est pas forcément plus rapide. Il devra par contre satisfaire à des contraintes temporelles strictes, prévues à l'avance et imposées par le processus extérieur à contrôler. Les contraintes temporelles pouvant s'échelonner entre quelques micro-secondes (µs) et quelques secondes.

La figure suivante permet d'illustrer la notion des systèmes temps réel sur le cas particulier de l'exécution d'une tâche périodique. Idéalement, une tâche périodique doit être exécutée toutes les m secondes (son temps d'exécution reste inférieur à m). Dans le cas d'un système non temps réel, un temps de latence apparaît avant l'exécution effective de la tâche périodique. Il varie fortement au cours du temps (charge du système...). Dans le cas d'un système temps réel, ce temps de latence doit être borné et garanti inférieur à une valeur fixe et connue à l'avance.

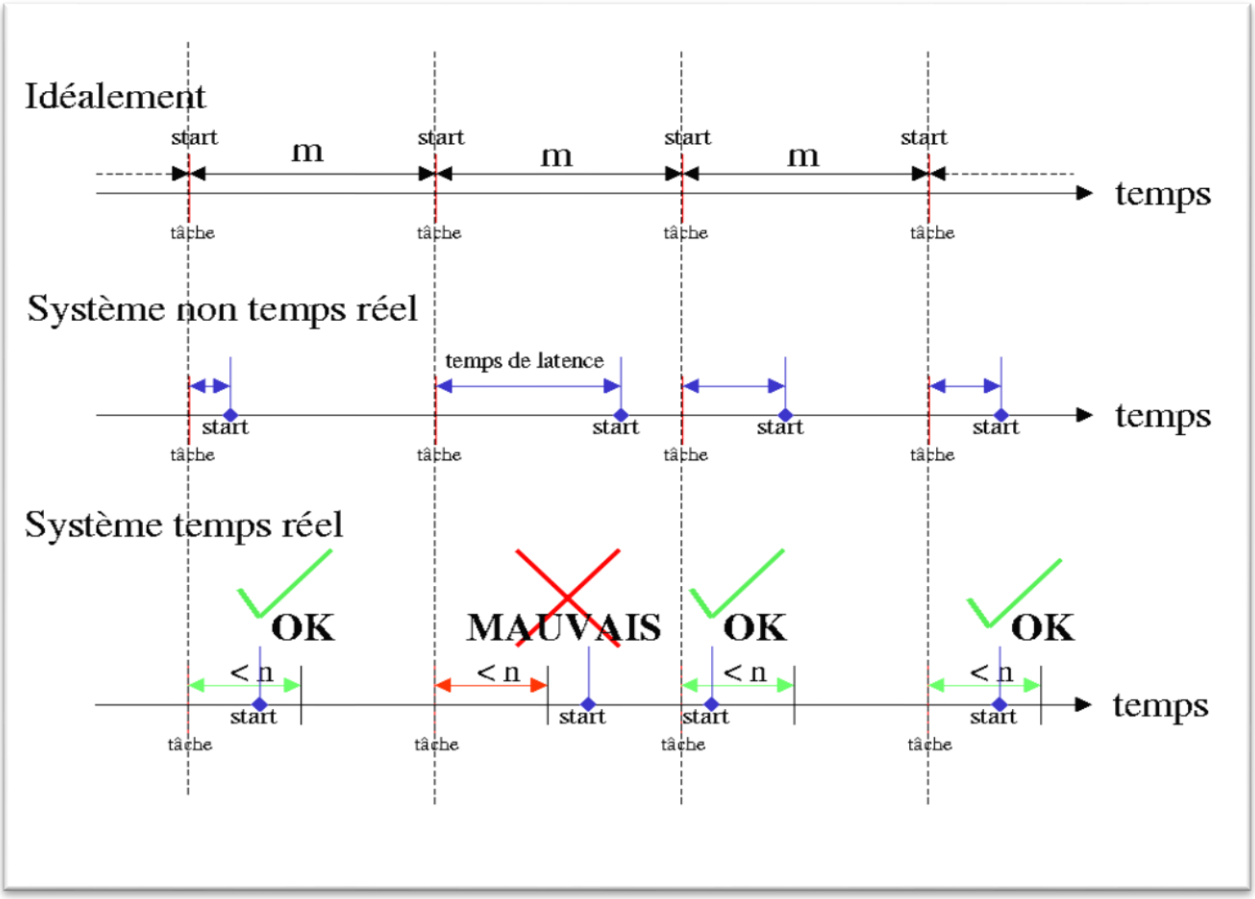


Figure 1.7. *Notions du système temps réel.*

1.6. Types de systèmes temps réels

On pourra diviser les systèmes temps réel en trois catégories :

1.6.1. Les systèmes dits à contraintes souples ou molles (*soft real time constraint*)

La réponse du système après les délais comme ce que montre la figure (1.8) réduit progressivement son intérêt. Les pénalités ne sont pas catastrophiques. On peut citer l'exemple des systèmes multimédia : si quelques images ne sont pas affichées, cela ne met pas en péril le fonctionnement correct de l'ensemble du système. Ces systèmes se rapprochent fortement des systèmes d'exploitation classiques à temps partagé. Ils garantissent un temps moyen d'exécution pour chaque tâche.

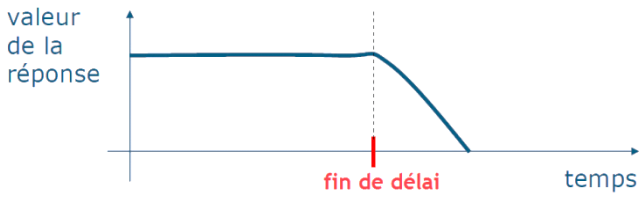


Figure 1.8. *Délai de réponse d'un système a contraintes souples.*

1.6.2. Les systèmes dits à contraintes dures (hard real time constraint)

La réponse du système dans les délais est vitale. L'absence de réponse est catastrophique (plus qu'une réponse incorrecte). Une gestion stricte du temps est nécessaire pour conserver l'intégrité du service rendu. On citera comme exemples les commandes de processus industriels sensibles comme la régulation des centrales nucléaires ou les systèmes embarqués utilisés dans l'aéronautique. Ces systèmes garantissent un temps maximum d'exécution pour chaque tâche.

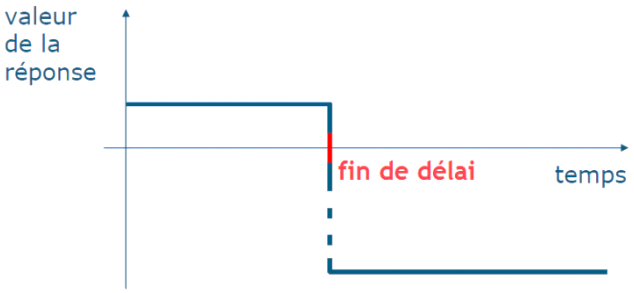


Figure 1.9. *Délai de réponse d'un système a contraintes dure.*

Les systèmes à contraintes dures doivent répondre à trois critères fondamentaux :

- **Le déterminisme logique** : les mêmes entrées appliquées au système doivent produire les mêmes effets.

- **Le déterminisme temporel** : une tâche donnée doit obligatoirement être exécutée dans les délais bornés.

- **La fiabilité** : le système doit être disponible. Cette contrainte est très forte dans le cas d'un système embarqué car les interventions d'un opérateur sont très difficiles voire même impossibles. Cette contrainte est indépendante de la notion de temps réel mais la fiabilité du système sera d'autant plus mise à l'épreuve dans le cas de contraintes dures.

1.6.3. Système a contraintes temps réel ferme

La réponse du système dans les délais est essentielle. Le résultat ne sert plus à rien une fois le délai dépassé. La pénalité de non-réponse est dans le même ordre de magnitude que la valeur de la réponse. Exemples : transactions en bourse…, le temps réel ferme de l'un peut être le temps réel dur de l'autre.

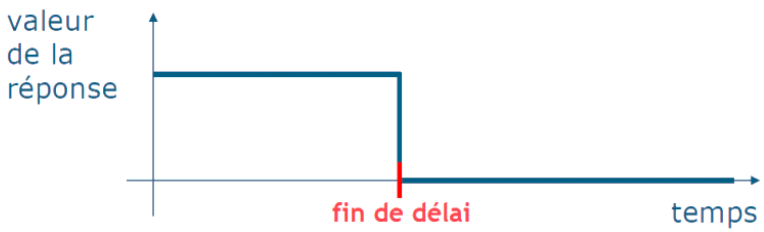


Figure 1.10. *Délai de réponse d'un système a contraintes ferme.*

1.7. Sources de contraintes de temps

Pour comprendre les origines des contraintes temporelles imposées à une application temps réel, il faut remonter à l'environnement dans lequel cette application est destinée à fonctionner.

En général, les sources de contraintes temporel sont liées aux points suivants :

A. Origines externes :

- Caractéristiques physiques (vitesse, durée de réaction chimique,…etc.).
- Caractéristiques liées aux lois de commande du système physique.
- Qualité de service requise en termes de délai de réponse tolérable.
- (qualité audio/vidéo).
- Perception sensorielle et le temps de réaction de l'homme.
- Contraintes à caractère commercial.
- Autres.

B. Origines internes :

- **Choix de conception**
  - architecture centralisée ou répartie (données, traitements, contrôle)
  - actions périodiques (avec les périodes adéquates) ou apériodiques
  - choix d'une structure d'application (interface entre composants, …)
  - autres

- **Choix d'architecture matérielle et logicielle**
  - processeurs avec des vitesses particulières
  - système d'exploitation (intégrant des algorithmes d'ordonnancement)
  - réseau avec des débits et des temps de réponse donnés
  - autres

1.8. Commande des systèmes [6] [1]

Le but de la commande consiste à déterminer les entrées qu'il faut appliquer au système pour obtenir un comportement désiré tout en satisfaisant des contraintes temporelles strictes, cela signifie de déterminer un moyen pour qu'un système physique se comporte d'une manière choisie et ceci en manipulant certaines variables précises.

1.8.1. Commande en boucle ouverte (BO) [9]

On dit que le système est commandé en boucle ouverte si le signal de commande est indépendant du signal de sortie.

Les avantages de cette structure de commande sont la simplicité et le faible coût, mais malheureusement ses applications sont limitées à cause des imprécisions particulièrement là où la grande précision est demandée et où les paramètres du système à commander sont variantes. Cette technique présente un certain nombre d'inconvénients :

- Elle peut aboutir à un correcteur instable inutilisable.
- Elle n'est pas robuste car elle nécessite une connaissance précise du modèle du système.

Ces inconvénients font qu'elle est rarement utilisée seule.

1.8.2. Commande en boucle fermée (BF) [9]

Pour améliorer les performances d'une commande, il est indispensable d'observer les sorties du système pour les comparer à ce que l'on désire obtenir.

Dans ce deuxième type de commande, les sorties du système se sont contrôlées. Contrairement à la commande en boucle ouverte la structure de commande en boucle fermé a les avantages inverses tel que :

- elle peut stabiliser un système instable.
- elle peut s'avérer robuste. Notamment, elle peut garder un certain niveau de performance malgré des incertitudes sur le modèle du système.

1.9. Techniques de commande des systèmes

Les types et les méthodes de commande sont très variés. Il existe différentes stratégies pour la commande, on cite principalement les principales techniques suivantes :

1.9.1. Commande par régulateur PID [10]

C'est la technique industrielle la plus largement utilisée qui calcule une action Proportionnelle, Intégrale et Dérivée en fonction de l'erreur consigne/mesure. Cette technique permet de satisfaire la régulation de plus de 90% des procédés industriels. Une généralisation de ce type de régulateur a permis d'avoir plus de possibilité en utilisant un prédicteur de smith dans les travaux de Manfred Morari et Evanghelos Zafiriou 1989.

1.9.2. Commande par retour d'état

La commande par retour d'état est un moyen de modifier le comportement en boucle fermée d'un système dynamique donné par une représentation d'état. Cette approche suppose l'état connu.

Quand ce n'est pas le cas, on peut utiliser un observateur d'état de manière à reconstruire l'état à partir des mesures disponibles. Ces types de commande peuvent être conçus de différentes manières notamment par minimisation d'un critère quadratique: commande LQ ou LQG.

La notion de commande par retour d'état remonte aux travaux de Pontryagin (en Russie) et de Bellman (aux États-Unis) sur la commande optimale.

1.9.3. Commande prédictive [11]

La commande prédictive (ou compensation ou correction anticipatrice) est une technique de commande avancée de l'automatique. Elle a pour objectif de commander des systèmes industriels complexes. Le principe de cette technique est d'utiliser un modèle dynamique du processus à l'intérieur du contrôleur en temps réel afin d'anticiper le futur comportement du procédé. La commande prédictive fait partie des techniques de contrôle à modèle interne (IMC: Internal Model Controler).

Cette méthode a été inventée par un Français, J. Richalet, en 1978 et généralisée par D.W. Clarke en 1987 en accord avec de grands groupes industriels aux États-Unis et en Europe (Shell et Adersa).

La commande prédictive peut être utilisée pour commander des systèmes complexes comportant plusieurs entrées et sorties où le simple régulateur PID est insuffisant. Cette technique est particulièrement intéressante lorsque les systèmes possèdent des retards importants. La figure suivante montre un exemple qui illustre le principe de prédiction:

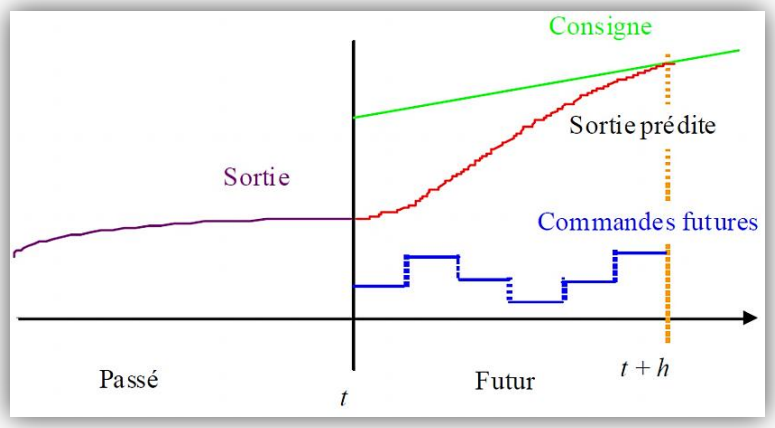


Figure 1.11. *Principe de la commande prédictive.*

1.9.4. Commande robuste [12]

C'est un type de commande qui vise à garantir les performances et la stabilité d'un système face à des perturbations du milieu et les incertitudes du modèle. En effet, le modèle mathématique qui modélise un système réel est une représentation qui vise à approximer au mieux, avec des hypothèses simplificatrices, le système qu'on veut commander. Il existe donc un écart entre le comportement observé du système réel et son modèle interne. La commande robuste vise à déterminer une loi de commande qui soit capable de garantir des critères de performance et stabilité pour un système dont le modèle varie autour du modèle théorique ou nominal.

Par la même approche, on peut rendre le système robuste face aux perturbations extérieures qui en somme peuvent être considérées comme une modification du modèle interne. Une commande robuste peut être conçue par exemple par minimisation d'un critère (par exemple de nature H-infini).

1.9.5. Commande adaptative [13]

La commande adaptative est un ensemble de techniques utilisées pour l'ajustement automatique en ligne et en temps réel des régulateurs des boucles de commande afin de réaliser ou maintenir un certain niveau de performances quand les paramètres du procédé à commander sont soit inconnus soit/et varient dans le temps.

1.9.6. Commande non linéaire utilisant le critère de lyaponov [14] [15]

Ce type de commande est largement étudié et développée depuis leurs introduction, on cite notamment la commande par mode glissant introduit par les chercheurs soviétiques [Slotin-91 ; Utkin-92], Sa caractéristique principale est sa structure variable avec commutation d'une part et d'autre d'une surface appelée surface de glissement, choisie à priori de telle sorte que le comportement résultant correspond aux dynamiques souhaitées. On cite Aussi la commande par Backstepping développée en 1991.

1.9.7. La commande par platitude différentielle [17]

Cette technique qui permet l'inversion de modèle sans passer par l'intégration des équations différentielles, et ainsi de calculer les signaux nécessaires sur les entrées pour garantir les trajectoires voulues en sortie, on cite notamment les travaux de Hebertt J. Sira Ramírez et Sunil Kumar Agrawal, *en* 2004.

1.9.8. Commande par logique floue

Elle a été formalisée par Lotfi Zadeh en 1965, Elle s'appuie sur la théorie mathématique des ensembles flous. La figure suivante illustre les principaux modules composant la commande floue :

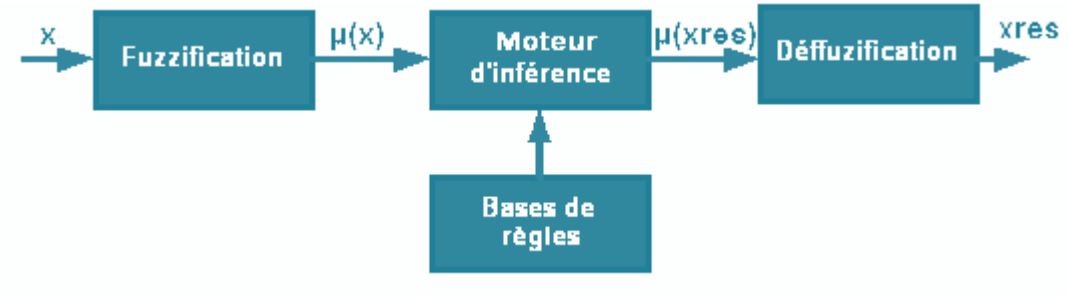


Figure 1.12. *Schéma de base d'un contrôleur floue*

Ou x représente le vecteur des entrées, x_{RES} celui des commandes, μ(x) et μ(x_{RES}) Les fonctions d'appartenances correspondantes. Le premier module, appelée Fuzzification, consiste à attribuer à la valeur réelle de chaque entrée, au temps t, sa fonction d'appartenance à chacune des classes préalablement définies, donc à transformer l'entrée réelle en un sous ensemble floue. Le deuxième module consiste en l'application de règles. Il se constitue d'une base de règles et d'un moteur d'inférence qui permet le calcul. Le dernier module décrit l'étape de Défuzzification qui est la transformation inverse de la première.

1.9.9. Commande par réseaux de neurones [16]

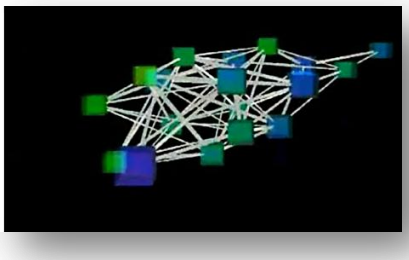


Figure 1.13. *Représentation graphique 3D d'un exemple de réseau de neurones artificielle.*

Par leur capacité d'approximation universelle, les réseaux de neurones inspirés des modèles biologiques lors des travaux de Warren McCulloch et Walter Pitts en 1950 sont bien adaptés pour la commande des systèmes non linéaires. Caractérisés par leurs natures parallèles et non linéaires, les réseaux de neurones sont souvent utilisés en classification.

1.10. Regroupement des techniques de commande en catégorie [18]

Dans la littérature, les techniques de commande peuvent être divisées en 3 catégories principales :

- La première catégorie consiste en la linéarisation des systèmes non linéaires autour d'un point de fonctionnement d'état. Dans ce cas, les lois de commande linéaires classiques sont appliquées pour le système approximé. En dépit de la simplicité des lois de commande, la performance du système de commande et la stabilité ne sont pas garantis pour l'ensemble du système.

- La deuxième catégorie traite de la conception des contrôleurs non linéaires basés sur les dynamiques non linéaires des systèmes. Dans cette catégorie, les caractéristiques des systèmes non-linéaires sont préservées. En outre, ces approches supposent la précision du modèle mathématique du système et ont tendance à bien fonctionner en théorie, Mais, leur performance se dégrade en présence de diverses conditions de fonctionnement, tel que les incertitudes et les perturbations externes.

- La troisième catégorie met en œuvre des contrôleurs non linéaires basés sur les outils d'intelligence artificielle, tels que les réseaux de neurones (RNA) et les systèmes de logique floue. Ces techniques ont été créditées dans diverses applications comme des outils puissants.

1.11. Conclusion

Dans ce chapitre, on a donné un panorama des notions et des définitions essentielles pour la compréhension des systèmes en général et des systèmes temps réel en particulier. Puis, on a présenté les différentes recherches et solutions concernant les différentes techniques de commande adaptées à ce type de systèmes. Dans le chapitre suivant, on va présenter la description et la modélisation du système utilisé dans ce mémoire. Il s'agit du pendule inversé. Cette modélisation servira pour la mise en œuvre des différentes techniques de commande.

Chapitre 2

Modélisation du pendule inversé

2.1. Introduction [19] [20]

Le pendule inversé est un outil expérimental parmi les meilleurs outils de tests pratiques des différentes méthodologies de commande. C'est une plateforme idéale pour la recherche caractérisée par son aspect de prototypage universel et d'une signification profonde dans la théorie de commande des systèmes temps réel. Pour cela, et afin de profiter de ces points avantageux, le pendule inversé est choisi comme notre environnement d'application.

Deux grandes familles de modèles sont présentées dans la littérature :

- Les modèles de connaissance utilisant les lois de la physique, des relations mathématiques,….etc. Ce sont des modèles paramétriques (par exemple: fonction de transfert, équations différentielles,...).

- Les modèles de représentation: lorsque l'analyse interne du système n'est pas possible, il est alors considéré comme une boîte noire (identification).

Notre choix s'est porté sur le modèle de connaissance puisque tous les composants du système peuvent être représentés par des équations mathématiques.

L'objectif de ce chapitre est la modélisation de l'ensemble : moteur et chariot-pendule, de la plateforme **IP02** disponible au sein du laboratoire d'Automatique.

2.2. Description du pendule inversé IP02 [21]

Le pendule inversé, utilisé dans le banc d'essai, se compose d'un chariot en aluminium plein conduit par un moteur à CC de haute qualité équipé d'un système d'entrainement comme illustré dans les figures (2.1), (2.2).

Le chariot glisse le long d'un axe d'acier inoxydable suivant des mouvements linéaires. Il se déplace par l'intermédiaire d'un mécanisme d'engrenages crémaillère/pignon, par opposition aux ceintures ou aux roues, afin d'éliminer le patinage, le glissement en cas de frein et d'autres effets indésirables. Ce mécanisme assure, donc, la traction conformée et continue.

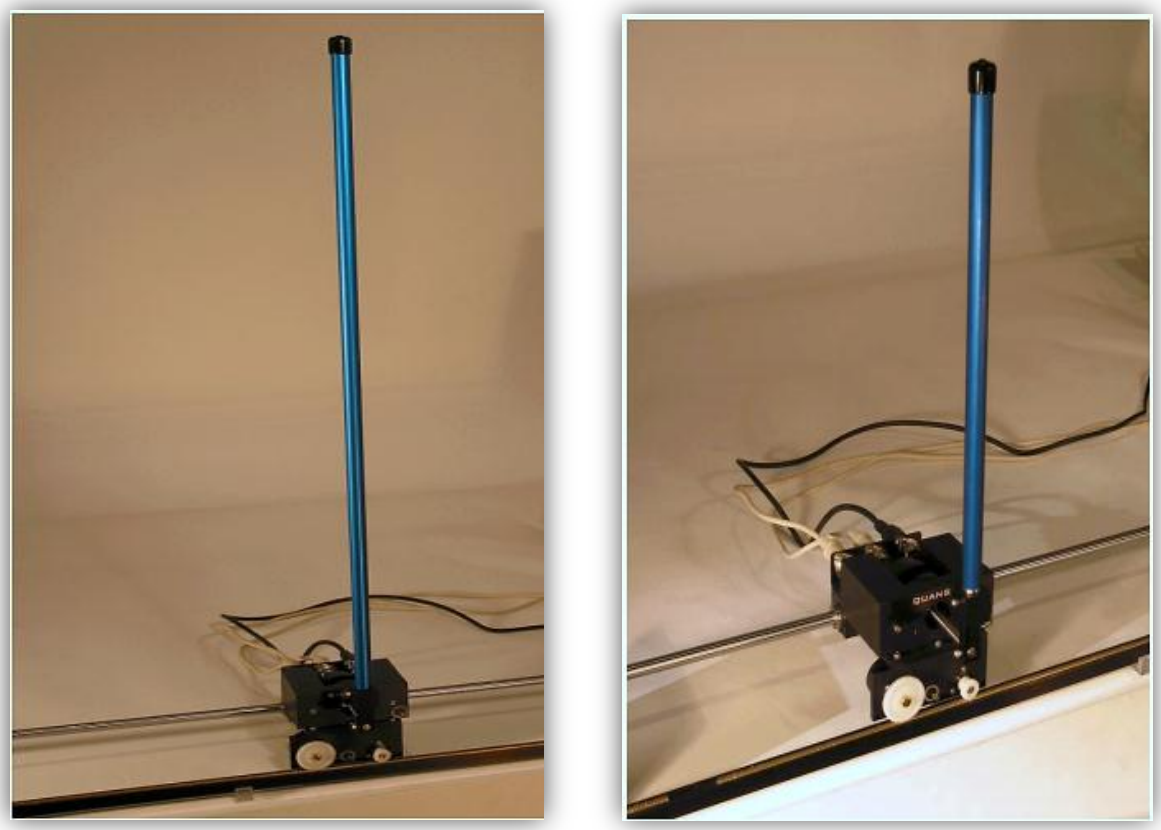


Figure 2.1. *Le pendule inversé IP02 à tige longue et moyenne.*

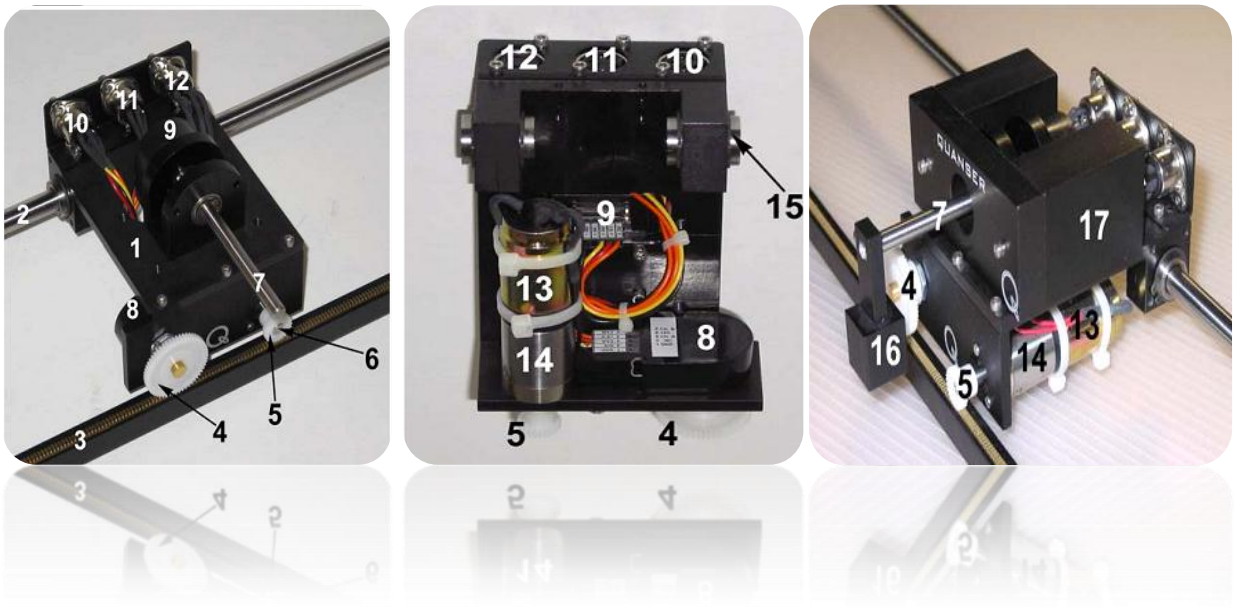


Figure 2.2.a. *Différentes parties constituant le chariot.*

Le tableau ci-dessous regroupe les différents composants du système cités (figure 2.2) :

Composants (1 à 9) :	*Composants (10 à 17) :*
1 : Chariot IP02.	*10 :* Connecteur d'encodeur du chariot
2 : Axe en acier inoxydable.	*11 :* Connecteur du moteur.
3 : Support.	*12 :* Connecteur d'encodeur du pendule.
4 : Pignon de position du chariot.	*13 :* Moteur à CC.
5 : Pignon du moteur du chariot.	*14 :* Boîte réducteur.
6 : Axe du pignon du moteur du chariot.	*15 :* Roulement linéaire.
7 : Axe du pendule.	*16 :* Douille du pendule (tige).
8 : Encodeur pour la position du chariot.	*17 :* Poids supplémentaire IP02.
9 : Encodeur pour l'angle du Pendule.	

Tableau 2.2.b. *Noms des éléments constituant le chariot.*

2.3. Les entrées et les sorties du système

L'IP02 a une seule entrée et deux sorties.

2.3.1. L'entrée (actionneur)

L'IP02 incorpore, comme actionneur, un moteur à CC de type Faulhaber Coreless (2338S006), comme le montre la figure (2.2) (composant # 13).Ce modèle est un moteur de basse inductance et de rendement très élevé ayant pour résultat une réponse beaucoup plus rapide qu'un moteur à CC conventionnel.

Le moteur est couplé à un système d'entrainement (réducteur) 23/1 de Faulhaber Gearhead, représenté sur la figure (2.2) (composant # 14), Son rapport de réduction est de 3,71:1.

2.3.2. Les sorties (capteurs)

Dans l'IP02, les deux sorties sont la position du chariot et l'angle du pendule, elles sont mesurées à l'aide de deux encodeurs incrémentaux optiques très sensibles, représentés dans la figure (2.2) (composants # 8 et 9, respectivement).
Ces encodeurs sont des encodeurs optiques asymétriques d'axe d'USA Digital S1. Il offre une haute résolution de 4096 comptes par révolution (i.e. 1024 lignes par révolution avec deux canaux dans la quadrature).

Disposer d'un encodeur sensible aux petites variations de la position et de l'angle est nécessaire surtout dans les mouvements du pendule inversé qui demandent de la précision.
Par conséquent, l'expérience d'auto-érection du pendule devient possible. Le principe de mesure de l'encodeur de position linéaire du chariot est identique à celui du moteur (crémaillère-pignon). Notre système a deux degrés de liberté (DDL), donc on aura besoin de deux coordonnées généralisées.

Les deux coordonnées choisies lors de la modélisation sont : $x_c(t)$ et $\alpha(t)$. En outre, l'entrée du système est notée F_c , la force linéaire appliquée sur le chariot motorisé.
Le tableau suivant regroupe la liste complète de symboles et notations utilisées dans la modélisation du système.

notation	*description*	*valeur*
M	*La masse du chariot.*	$0.94\ \ kg$
Beq	*Coefficient d'atténuation visqueux équivalent du chariot.*	$5.4\ \ N.s/m$
Mp	*La masse du pendule.*	$0.127\ \ kg$
lp	*La demi-longueur du pendule.*	$0.1778\ \ m$
Ip	*Moment d'inertie du pendule.*	$1.2 \times 10^{-3}\ \ kg.m^2$
Bp	*Coefficient d'atténuation visqueux du pendule.*	$0.0024\ \ N.m.s/rad$
g	*Force de gravité.*	$9.81\ \ m/s^2$
Vm	*Tension d'armature du moteur.*	
Im	*Courant d'armature du moteur.*	
Rm	*Résistance d'armature du moteur.*	$2.6\ \Omega$
Jm	*Moment d'inertie du rotor.*	$3.9 \times 10^{-7}\ \ kg.m^2$
η_m	*Rendement du moteur.*	1
K_m	*Constante de la Force Electromotrice (FEM).*	$0.00767\ \ V.s/rad$
η_g	*Rendement de la boite de vitesse.*	1
K_g	*Rapport de la boite de vitesse.*	3.71
K_t	*Constante du couple du moteur.*	$0.00767\ N.m/A$
r_{mp}	*Rayon Du Pignon Du Moteur*	$6.35 \times 10^{-3} m$

***Tableau 2.2**. Liste des notations utilisées dans la modélisation du pendule.*

Le pendule inversé étudié (IP02) est constitué d'un chariot mobile en translation sur un axe horizontal. Le pendule, tout en étant fixé (suspendu) sur le chariot, est libre en rotation. Figure (2.3).

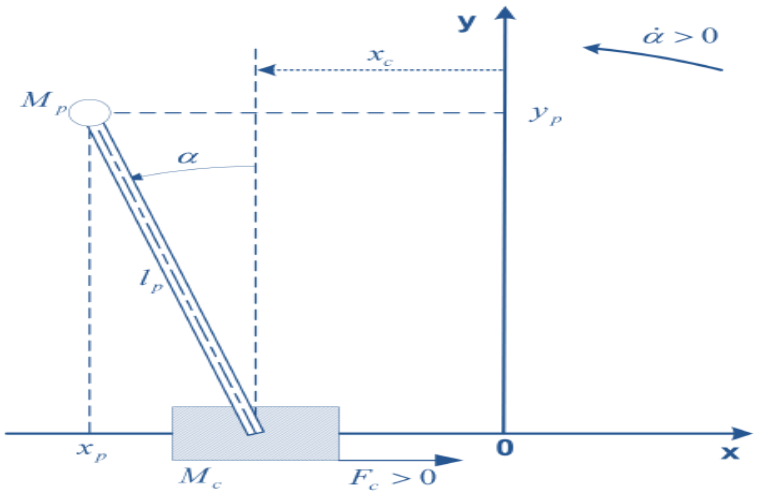


***Figure 2.3.** Le pendule inversé.*

En exerçant une force horizontale $F_c(t)$ sur le chariot, une translation de X mètres de celui-ci est obtenue ainsi qu'une rotation de α radians du pendule.

2.4. Modélisation du système

On va, dans la suite de cette partie, extraire les équations non linéaires du mouvement du système (le modèle dynamique). Pour se faire, deux approches sont possibles Newtonienne et Lagrangienne. Cette dernière est la méthode utilisée pour faciliter les calculs.

A partir de la figure (2.3), Considérons comme entrée du système une force horizontale F_c appliquée sur le chariot.

Les coordonnées généralisées du centre de gravité du pendule (partie suspendue du système) sont données par projection sur x et y :

$$\begin{cases} x_p(t) = x_c(t) - l_p sin(\alpha(t)) \\ y_p(t) = l_p cos(\alpha(t)) \end{cases} \quad (2\text{-}1)$$

L'application de la méthode de Lagrange nécessite le calcul de l'énergie potentielle et cinétique.

$$L = E_c - E_p \quad (2\text{-}2)$$

2.4.1. L'énergie cinétique du système (E_c)

Calculons, en premier lieu, l'énergie cinétique du système. Par définition, l'énergie cinétique d'un système est la somme des énergies responsables du mouvement de ce système.

Ici, l'énergie cinétique totale (E_c) est la somme des deux énergies de translation et rotationnelle produites respectivement par les deux composantes du système : le chariot et le pendule :

$$E_c = T_c + T_p \quad (2\text{-}3)$$

2.4.2. L'énergie cinétique du chariot (T_c)

L'énergie cinétique du chariot motorisé est-elle même divisée en deux parties: de translation pour le chariot et rotationnelle pour la motorisation :

$$T_c = T_{cc} + T_{cp} \quad (2\text{-}4)$$

Premièrement, l'énergie cinétique de translation est définie par :

$$T_{cc} = \frac{1}{2} M\left(\dot{x}_c(t)\right)^2 \quad (2\text{-}5)$$

Deuxièmement, l'énergie cinétique rotationnelle due à la rotation du moteur alimentant le chariot est :

$$T_{cp} = \frac{1}{2}\frac{J_m {K_g}^2}{{r_{mp}}^2}\left(\dot{x_c}(t)\right)^2 \qquad (2\text{-}6)$$

Donc, et à partir de ces deux dernières équations on obtient l'expression de l'énergie cinétique totale du chariot ainsi :

$$T_c = \frac{1}{2}M_c\left(\dot{x_c}(t)\right)^2 \qquad (2\text{-}7)$$

Avec: $$M_c = M + \frac{J_m {K_g}^2}{{r_{mp}}^2} \qquad (2\text{-}8)$$

2.4.3. L'énergie cinétique du pendule (T_p)

Elle est donnée par :

$$T_p = T_{pt} + T_{pr} \qquad (2\text{-}9)$$

Avec: T_{pt} l'énergie cinétique de translation.

T_{pr} L'énergie cinétique rotationnelle.

2.4.4. L'énergie cinétique de translation T_{pt}

Supposons que la masse du pendule est assumée concentrée dans son centre de gravité (CDG). Dans ce cas, l'énergie cinétique de translation du pendule T_{pt} peut être exprimée en fonction de son CDG comme suit :

$$T_{pt} = \frac{1}{2}M_p\left[\sqrt{\left(\dot{x_p}(t)\right)^2 + \left(\dot{y_p}(t)\right)^2}\right]^2 \qquad (2\text{-}10)$$

Où, les coordonnées du vecteur de vitesse du CDG du pendule sont données par :

$$\begin{cases} \dot{x_p}(t) = \left(\dot{x_c}(t)\right) - l_p cos\left(\alpha(t)\right)\left(\dot{\alpha}(t)\right) \\ \dot{y_p}(t) = -l_p sin\left(\alpha(t)\right)\left(\dot{\alpha}(t)\right) \end{cases} \qquad (2\text{-}11)$$

2.4.5. L'énergie cinétique rotationnelle T_{pr}

Elle est donnée par :

$$T_{pr} = \frac{1}{2}I_p\left(\dot{\alpha}(t)\right)^2 \qquad (2.12)$$

Finalement, l'expression finale de l'énergie cinétique de l'ensemble pendule + chariot motorisé est donnée par la somme des quatre énergies partielles comme suit :

$$E_c = \frac{1}{2}\left(M_c + M_p\right)\left(\dot{x_c}(t)\right)^2 - M_p l_p cos\left(\alpha(t)\right)\left(\dot{\alpha}(t)\right)\left(\dot{x_c}(t)\right) + \frac{1}{2}\left(I_p + M_p {l_p}^2\right)\left(\dot{\alpha}(t)\right)^2 \qquad (2\text{-}13)$$

Il est vu de cette dernière équation que toute l'énergie cinétique du système peut être exprimée en termes de coordonnées généralisées et de leurs dérivés d'ordre un.

2.4.6. L'énergie potentielle (E_p)

L'énergie potentielle dans un système quelconque est la quantité d'énergie que ce système, ou un élément de ce système, a en raison d'un certain genre de travail étant, ou après avoir été, fait à lui. Elle est habituellement provoquée par son déplacement vertical de la normalité (énergie potentielle de la gravité) ou par une sorte ressort-connexe de déplacement (énergie potentielle élastique).

Ici, il n'y a aucune énergie potentielle élastique dans le système. L'énergie potentielle du système est seulement due à la gravité. Le mouvement linéaire de chariot est horizontal, et dans un tel système, il n'y aura jamais le déplacement vertical. Par conséquent, toute l'énergie potentielle est entièrement exprimée par l'énergie potentielle de la gravité du pendule, comme indiqué ci-dessous :

$$E_p = M_p g l_p cos\big(\alpha(t)\big) \qquad (2\text{-}14)$$

On a : $$L = E_c - E_p \qquad (2\text{-}15)$$

Donc :

$$L = \frac{1}{2}\Big[\big(M_c + M_p\big)\big(\dot{x}_c(t)\big)^2 + \big(I_p + M_p {l_p}^2\big)\big(\dot{\alpha}(t)\big)^2\Big] - M_p l_p cos\big(\alpha(t)\big)\big(\dot{\alpha}(t)\big)\big(\dot{x}_c(t)\big) - M_p g l_p cos\big(\alpha(t)\big) \qquad (2\text{-}16)$$

Considérons, maintenant les équations de Lagrange pour notre système. Par définition, les deux équations de Lagrange, résultant des deux coordonnées généralisées, $x_c(t)$ et $\alpha(t)$, précédemment définies, ont les formes suivantes :

$$\begin{cases} \frac{d}{dt}\left(\frac{d}{d\dot{x}_c(t)}L\right) - \frac{d}{dx_c(t)}L = F_{x_c} \\ \frac{d}{dt}\left(\frac{d}{d\dot{\alpha}(t)}L\right) - \frac{d}{d\alpha(t)}L = F_\alpha \end{cases} \qquad (2\text{-}17)$$

Avec: F_{x_c} et F_α sont les forces appliquées sur les coordonnées généralisées $x_c(t)$ et $\alpha(t)$ respectivement. Pour notre système, c'est deux forces sont :

$$F_{x_c} = F_c(t) - B_{eq}\big(\dot{x}_c(t)\big) \qquad (2\text{-}18)$$

$$F_\alpha = -B_p\big(\dot{\alpha}(t)\big) \qquad (2\text{-}19)$$

En fin, les équations de Lagrange décrivant la dynamique du système sont :

$$(M_c + M_p)(\ddot{x_c}(t)) - M_p l_p cos(\alpha(t))(\ddot{\alpha}(t)) + M_p l_p sin(\alpha(t))(\dot{\alpha}(t))^2 + B_{eq}(\dot{x_c}(t)) = F_c(t) \quad (2\text{-}20)$$

$$-M_p l_p cos(\alpha(t))(\ddot{x_c}(t)) + (I_p + M_p {l_p}^2)(\ddot{\alpha}(t)) - M_p g l_p sin(\alpha(t)) + B_p(\dot{\alpha}(t)) = 0 \quad (2\text{-}21)$$

Ces deux équations peuvent être réécrites sous la forme suivante :

- $\ddot{x_c}(t) =$
  $\Big(-(I_p + M_p {l_p}^2)B_{eq}(\dot{x_c}(t)) - (M_p{}^2 {l_p}^3 + I_p M_p l_p)sin(\alpha(t))(\dot{\alpha}(t))^2 - M_p l_p B_p cos(\alpha(t))(\dot{\alpha}(t)) + (I_p + M_p {l_p}^2)F_c(t) + M_p{}^2 {l_p}^2 g cos(\alpha(t))sin(\alpha(t))\Big) / \Big((M_c + M_p)I_p + M_c M_p {l^2}_p + {M^2}_p {l^2}_p sin(\alpha(t))^2\Big)$ (2-22)
- $\ddot{\alpha}(t) =$
  $\Big((M_c + M_p)M_p l_p g sin(\alpha(t)) - (M_c + M_p)B_p\dot{\alpha}(t) - M_p{}^2 {l_p}^2 sin(\alpha(t))cos(\alpha(t))(\dot{\alpha}(t))^2 - M_p l_p cos(\alpha(t))B_{eq}(\dot{x_c}(t)) + M_p l_p cos(\alpha(t))F_c(t)\Big) / \Big(\big((M_c + M_p)I_p + M_c M_p {l^2}_p\big) + {M^2}_p {l^2}_p sin(\alpha(t))^2\Big)$ (2-23)

Dont l'entrée du système est la force d'entraînement linéaire du chariot F_c. En vérité, l'entrée du système réel est la tension d'alimentation du moteur V.

On prend l'ensemble moteur/système d'engrenages comme un convertisseur tension/force dont la relation entre V et F est donnée dans [7] par :

$$F_c = -\frac{\eta_g K_g{}^2 \eta_m K_t K_m(\dot{x_c}(t))}{R_m {r_{mp}}^2} + \frac{\eta_g K_g \eta_m K_t V_m}{R_m r_{mp}} \quad (2\text{-}24)$$

2.5. Linéarisation du modèle

Afin de pouvoir mettre les équations précédentes sous la forme :

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Du \end{cases}$$

Avec A, B, C etD sont des matrices réelles constantes, on est besoin de linéariser le système.

Le pendule IP02 a deux points d'équilibre :

- Point d'équilibre stable pour $\alpha = \pi$ (le pendule est en bas).
- Point d'équilibre instable pour $\alpha = 0$ (le pendule est en haut).

On prend le deuxième point de fonctionnement (cas du pendule inversé), et en notant comme vecteur d'état $X = [x_1\ x_2\ x_3\ x_4\ ]^T = [x\ \alpha\ \dot{x}\ \dot{\alpha}\ ]^T$.

Pour des petites variations de l'angle autour de ce point on aura :

$\cos(\alpha) = 1,\ \sin(\alpha) = \alpha$ et $\dot{\alpha} = 0$.

En prenant en compte ces hypothèses simplificatrices, on obtient le modèle linéarisé du pendule inversé ainsi :

$$\dot{X} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & \frac{gM^2{}_p l^2{}_p}{(M_c+M_p)I_p + M_c M_p l_p{}^2} & \frac{-B_{eq}(I_p + M_p l_p{}^2)}{(M_c+M_p)I_p + M_c M_p l_p{}^2} & \frac{-M_p l_p B_p}{(M_c+M_p)I_p + M_c M_p l_p{}^2} \\ 0 & \frac{gM_p l_p (M_c+M_p)}{(M_c+M_p)I_p + M_c M_p l_p{}^2} & \frac{-M_p l_p B_{eq}}{(M_c+M_p)I_p + M_c M_p l_p{}^2} & \frac{-(M_c+M_p)B_p}{(M_c+M_p)I_p + M_c M_p l_p{}^2} \end{pmatrix} X + \begin{pmatrix} 0 \\ 0 \\ \frac{I_p + M_p l_p{}^2}{(M_c+M_p)I_p + M_c M_p l_p{}^2} \\ \frac{M_p l_p}{(M_c+M_p)I_p + M_c M_p l_p{}^2} \end{pmatrix} U \qquad (2\text{-}25)$$

Et ainsi on obtient :

$$\dot{X} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1.5216 & -11.6513 & -0.0049 \\ 0 & 26.1093 & -26.8458 & -0.00841 \end{bmatrix} X + \begin{bmatrix} 0 \\ 0 \\ 1.5304 \\ 3.5261 \end{bmatrix} U$$

$$y = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix} X$$

2.6. Conclusion :

Dans ce chapitre, nous avons effectué une modélisation du pendule inversé IP02 en exploitant les lois de la mécanique et la méthode de Lagrange. Ce modèle sera la plateforme sur laquelle on va tester et valider les algorithmes de commande qui seront développés durant notre étude dans les chapitres suivants.

Chapitre 3

Commande linéaire quadratique « L.Q.R »

3.1. Introduction [20] [22]

Les problèmes de commande optimale se rencontrent dans la vie de tous les jours : comment arriver à destination le plus rapidement possible, comment minimiser sa consommation... Pour un système dynamique donné et dont les équations sont connues, le problème de commande optimale consiste alors à trouver la commande minimisant un critère donné.

C'est sous cette forme que la commande optimale a été étudiée dès le XIXème siècle avec le calcul des variations. Une des grandes applications de la commande optimale a été l'application au lanceur Apollo dans les années 1960.

Notons néanmoins que les difficultés soulevées par ce genre de problèmes sont loin d'être complètement résolues comme en témoignent les sessions dédiées à la commande optimale dans les conférences d'automatique. La commande optimale reste donc un sujet de recherche d'actualité.

La commande linéaire quadratique, connue sous le nom LQR « Linear Quadratic Regulator », représente l'une des techniques de commande optimale les plus utilisées.

3.2. Théorie de la commande linéaire quadratique (LQR) [23] [24]

La théorie de la commande LQR, est un outil important dans la théorie de commande moderne. Elle fournit une méthode efficace d'analyse multi-variable pour la conception de système de rétroaction, et manipule des problèmes de bruit de signal et de mesure de perturbation.

La commande linéaire quadratique (LQR), s'applique aux systèmes (linéaires) ayant pour formes :

$$\begin{cases}\dot{x}(t) = Ax(t) + Bu(t)\\ y(t) = Cx(t) + Du(t)\end{cases} \quad (3\text{-}1)$$

Introduisant la fonction de coût J :

$$J(Q,R) = \frac{1}{2}\int_0^{\infty}\left(X^T(t)Q(t)X(t) + u^T(t)R(t)u(t)\right)d(t) \quad (3\text{-}2)$$

Avec Q et R dénotent les matrices de pondération.

Le but est de concevoir une commande u(t) dont la fonction J dans l'équation précédente soit minimale.

Si ce système est perturbé et excentré de l'état zéro, la commande u(t) peut faire revenir le système à l'état zéro, tout en conservant J minimal. Ici, la valeur u(t) de commande s'appelle la commande optimale.

Le signal de commande devrait être :

$$u(t) = -R^{-1}B^TP(t)X(t). \quad (3\text{-}3)$$

$$= -K\,X(t) \quad (3\text{-}4)$$

Où :

P(t) : Est la solution de l'équation de Riccati.

K : Est la matrice optimale linéaire du gain de retour.

Maintenant nous devons seulement résoudre l'équation de Riccati, pour donner une valeur à P(t):

$$PA + A^TP - PBR^{-1}B^TP + Q = 0 \quad (3\text{-}5)$$

Alors on peut obtenir la valeur de K par substitution de P dans l'équation :

$$K = R^{-1}B^TP = [k_1, k_2, k_3, k_4]^T \quad (3\text{-}6)$$

Selon la loi de commande (LQR), son optimalité dépend totalement du choix de Q et de R.

Cependant, il n'y a aucune méthode de résolution pour choisir ces deux matrices. La méthode répandue employée pour choisir Q et R est la simulation et l'épreuve. Généralement, Q et R devraient être deux matrices diagonales. Pour avoir une plus petite entrée, un plus grand R est nécessaire ; si l'entrée d'un état plus petit est voulue, l'élément dans la colonne correspondante de Q doit être plus grand …etc.

3.3. Gains du régulateur

L'utilisation de l'algorithme LQR (Linear-Quadratic-Regulator) pour la recherche des gains optimums est d'une grande efficacité. Une fois les coefficients de pondération Q et R sont fixés, on obtient les gains du régulateur d'état K en utilisant l'instruction Matlab suivante:

$$\mathbf{K} = \mathbf{lqr}\,(\mathbf{A}, \mathbf{B}, \mathbf{Q}, \mathbf{R})$$

3.4. Conclusion

La commande LQR fait partie de la 1^{ere} catégorie des techniques de commandes qui se base sur la linéarisation d'un système non linéaire et qui opèrent autour d'un point de fonctionnement dans la zone linéaire.

Cette commande est la plus importante et la plus complète d'une classe de synthèse du problème basée sur l'optimisation [20], puisque elle se charge à la fois de la stabilisation du système, du rejet de perturbation et de la minimisation de l'énergie de commande.

Chapitre 4

Commande par mode de glissement

4.1.Introduction [25]

La commande par mode de glissement est une stratégie très populaire pour la commande des systèmes non linéaires incertains, avec un large domaine d'applications. Due à l'utilisation de fonctions discontinues et de commande à gain élevé, son aspect principal est la robustesse pour les systèmes en boucle fermée et le temps de convergence finit. Cependant sa conception nécessite la connaissance des incertitudes. Qui peut être, du point de vue pratique, une tâche difficile.

Dans ce chapitre, on présente la commande par mode de glissement. Ensuite, on effectue une application sur un système linéaire et sur le pendule inversé modélisé, ainsi une nouvelle approche incrémentale est proposée. En fin, on cite les avantages et les inconvénients du mode de glissement.

4.2. Aspects théoriques de la commande à structure variable

La théorie des systèmes à structure variable (SSV) et les modes de glissements associés a fait l'objet d'études détaillées au cours des trente dernières années [26] [27], aussi bien par les chercheurs soviétiques, que par les chercheurs des autres pays. La discussion qui a eu lieu en 1960 entre Neimark et Philipov a été concluante dans la mesure où un nouveau problème sur la théorie des modes de glissements a été posé.

La technique de commande par mode de glissement consiste à amener la trajectoire d'état d'un système vers la surface de glissement et de la faire commuter à l'aide d'une logique de commutation appropriée autour de celle-ci jusqu'au point d'équilibre, d'où le phénomène de glissement [28].

Les systèmes de commande à structure variable, en anglais Variable Structure control (VSC), (ou par mode de glissement, Sliding Mode Control (SMC)) sont des systèmes de commande non linéaires discontinus ou la structure du contrôleur varie entre deux structures, la commutation d'une valeur à l'autre se fait suivant le signe d'un hyperplan de l'espace d'état du système de façon à forcer le point représentatif de son mouvement à rester sur cet hyperplan d'où le nom de régime glissant idéal, ceci dans le but d'obtenir une meilleure stabilité et précision que celles obtenues généralement avec les régulateurs classiques.

La commande à structure variable n'affecte plus ce nouveau régime qui est commandé uniquement par une commande appelée commande équivalente. Cette commande est indépendante des variations de paramètres et des perturbations d'où l'invariance de la dynamique du système dans le mode de glissement (qui est gouverné uniquement par le choix des coefficients de la surface de glissement), le système commandé est alors complètement insensible aux paramètres incertains et aux perturbations.

4.2.1. Principe de la commande par mode de glissement

L'idée de base de la commande par régime glissant est premièrement d'attirer les états du système dans une région convenablement sélectionnée, puis de concevoir une loi de commande qui maintiendra toujours le système dans cette région. En résumé, une commande par régime glissant est divisée en deux parties [29] :

$$u = u_{gliss} + u_{eq} \tag{4-1}$$

u_{gliss} : Le glissement est utile pour compenser les incertitudes du modèle. Il est constitué de la fonction signe « Sign » de la surface de glissement s, multipliée par une constante K_{gliss}.

La surface de glissement est définie dans l'espace d'état des erreurs afin de garantir la convergence des états.

u_{eq} : La commande équivalente ou nominale est déterminée par le modèle du système, dans ce cas il s'agit d'un modèle en immersion linéaire ou non linéaire. Cette partie est conçue avec la méthode de la commande équivalente, dont le principe est basé sur la détermination du comportement du système lorsqu'il est sur la surface de glissement s.

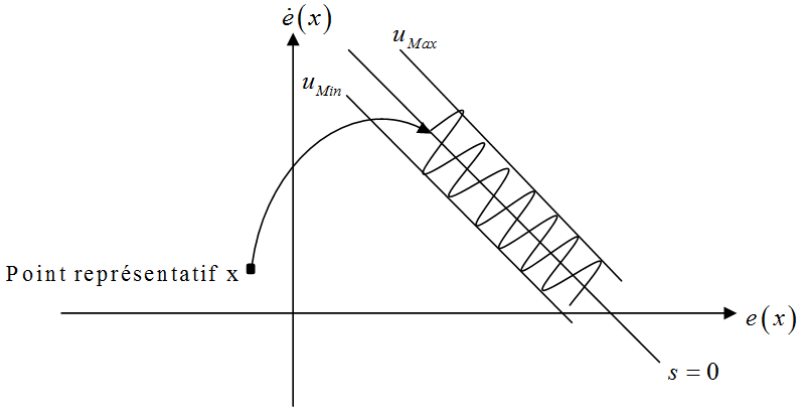


Figure 4.1. *Mode de glissement.*

4.2.2. Notions de base de la commande SMC

Soit le système donné par :

$$\dot{x} = f(x) + g(x)u \tag{4-2}$$

Où $x = (x_1, ..., x_n)^T \in X$ une variété différentiable, u représente la commande du système, f et g sont des champs de vecteur, définis sur X.

A. Surface de glissement

Pour des raisons de stabilisation et de définition d'une dynamique désirée du système dans le mode de glissement, la surface de glissement $s(x)$ peut être choisie en générale comme étant un hyperplan passant par l'origine de l'espace.

La surface s est donnée par :

$$s_i = \dot{e}_i + \lambda_i e_i \tag{4-3}$$

avec :

$$e_i = x_i - x_{d_i} \tag{4-4}$$

λ_i : Paramètre de la surface de glissement.
x_i : État du système.
x_{d_i} : État désiré.

B. Régime glissant idéal

Ce régime correspond à une oscillation de fréquence infinie et d'amplitude nulle, d'où le point représentatif du mouvement du système glisse parfaitement sur l'hyperplan de commutation ($s(x)=0$).

C. Régime glissant réel

La trajectoire d'état dans ce régime reste autour de l'hyperplan de glissement ($s(x)=0$) jusqu'à au point d'équilibre.

D. Condition d'existence et d'unicité du régime glissant

Un régime glissant existe sur une surface de glissement si et seulement si, dans un voisinage de la surface de glissement toutes les trajectoires du système sont dirigées vers elle. En d'autres termes :

$$\lim_{s\to 0^-} \dot{s}(x) > 0 \quad et \quad \lim_{s\to 0^+} \dot{s}(x) < 0 \tag{4-5}$$

4.2.3. Méthodes de synthèse classiques de la commande SMC

Dans ce qui suit on va présenter deux expressions équivalentes pour la détermination de l'état $x(t)$, qui est la solution du système en mode de glissement.

La commande u est construite de façon à ce que les trajectoires du système soient amenées vers la surface de glissement et soient ensuite maintenues dans un voisinage de celle-ci.
u est une loi de commande à structure variable définie comme suit :

$$\begin{cases} u^+(x) & si \quad s(t,x) > 0 \\ u^-(x) & si \quad s(t,x) < 0 \end{cases}, \; u^+ \neq u^- \tag{4-6}$$

u^+ et u^- étant des fonctions continues. Il est à noter que le caractère discontinu de la loi de commande qui permet d'obtenir une convergence en temps fini sur la surface ainsi que des propriétés de robustesse vis-à-vis de certaines perturbations.

A. Méthode de Philipov

L'approche de Philipov est l'une des premières approches. Cette méthode est basée sur les résultats des travaux effectués par ce chercheur sur les équations différentielles à second membre discontinu.

Considérons le système non linéaire suivant :

$$\dot{x}(t) = f(x) + g(x)u \tag{4-7}$$

Avec la stratégie de commande suivante :

$$u = \begin{cases} u^{+}(x) & si \quad s(t,x) > 0 \\ u^{-}(x) & si \quad s(t,x) < 0 \end{cases} \tag{4-8}$$

On peut montrer à partir des travaux de Philipov que les trajectoires d'états (4-7) et (4-8) à $s(x) = 0$ sont les solutions de l'équation :

$$\dot{x} = \alpha f^{+} + (1-\alpha) f^{-} = f^{0}, avec \quad 0 \le \alpha \le 1 \tag{4-9}$$

Où, $f^{+} = f(x) + g(x)u^{+}$, $f^{-} = f(x) + g(x)u^{-}$ et f^{0} est le vecteur de vitesse résultant de la trajectoire d'état et qui est tangentiel à la surface de glissement, il est obtenu par la moyenne géométrique comme le montre la figure suivante :

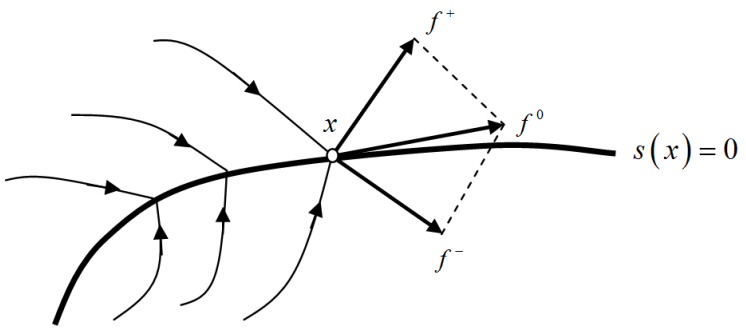


Figure 4.2. *Illustration de la résolution de Philipov.*

en mode de glissement : $s(x) = 0$,
en dérivant $s(x) = 0$ par rapport au temps, on aura :

$$\frac{ds(x)}{dt} = \left(\frac{\partial s}{\partial x}\right)^{T} f^{0} + \frac{\partial s}{\partial t} = <\nabla s, f^{0}> + \frac{\partial s}{\partial t} = 0 \tag{4-10}$$

d'où :

$$\alpha = \frac{<\nabla s, f^{-}> + \dfrac{\partial s}{\partial t}}{<\nabla s, f^{-} - f^{+}>} \tag{4-11}$$

avec :

- ∇s : Est le gradient de *s*.
- <, > : Est le produit scalaire.

En utilisant l'expression de α dans (4-9), l'équation caractérisant la trajectoire d'état du système en régime glissant est comme suit :

$$\frac{dx}{dt}=\left[\frac{<\nabla s, f^{-}>+\frac{\partial s}{\partial t}}{<\nabla s, f^{-}-f^{+}>}\right]f^{+}-\left[\frac{<\nabla s, f^{+}>+\frac{\partial s}{\partial t}}{<\nabla s, f^{-}-f^{+}>}\right]f^{-}=f^{0} \tag{4-12}$$

La méthode de Philipov est l'une des possibilités permettant la détermination de la trajectoire d'état en mode de glissement, une autre méthode, appelée méthode de la commande équivalente, a été proposée par Utkin.

B. Méthode d'Utkin (méthode de la commande équivalente) [27]

La méthode de la commande équivalente est un moyen pour déterminer le mouvement sur la surface de glissement.

Soit le système donné par :

$$\dot{x}=f(x)+g(x)u \tag{4-13}$$

en régime de glissement

$$\begin{cases} s(x)=0 \\ \dot{s}=\dfrac{ds}{dt}=\left(\dfrac{\partial s}{\partial x}\right)^{T}\dfrac{dx}{dt}+\dfrac{\partial s}{\partial t}=\left(\dfrac{\partial s}{\partial x}\right)^{T}\left(f(x)+g(x)u_{eq}\right)+\dfrac{\partial s}{\partial t}=0 \end{cases} \tag{4-14}$$

Où u_{eq} est la commande équivalente. Elle est donc définie par l'équation suivante :

$$u_{eq}=-\left(\left(\frac{\partial s}{\partial x}\right)^{T}g(x)\right)^{-1}\left(\left(\frac{\partial s}{\partial x}\right)^{T}f(x)+\frac{\partial s}{\partial t}\right), \text{ telque : } \left(\left(\frac{\partial s}{\partial x}\right)^{T}g(x)\right)\neq 0 \tag{4-15}$$

en remplaçant l'expression de u_{eq} dans (4-13), on obtient la trajectoire d'état en mode de glissement.

$$\frac{dx}{dt}=f(x)-g(x)\left(\left(\frac{\partial s}{\partial x}\right)^{T}g(x)\right)^{-1}\left(\left(\frac{\partial s}{\partial x}\right)^{T}f(x)+\frac{\partial s}{\partial t}\right) \tag{4-16}$$

La commande équivalente peut être interprétée comme la valeur moyenne que prend la grandeur de commande discontinue lors de la commutation rapide entre $u_{\max}$ et $u_{\min}$ et comme il est représenté sur la figure (4-3).

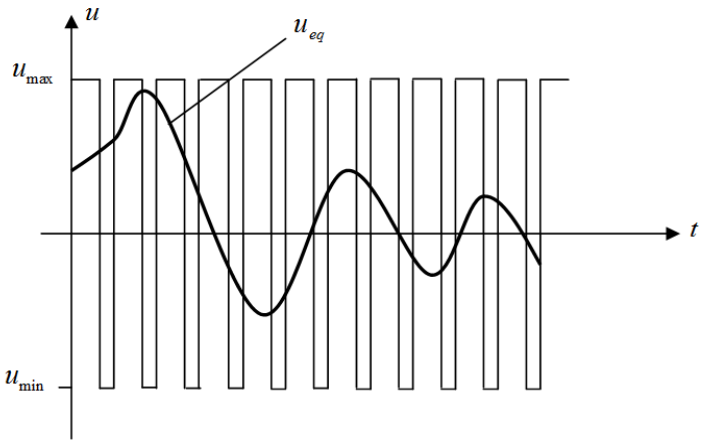


Figure 4.3. *Commande équivalente.*

4.2.4. Le broutement dans le mode de glissement (chattering)

La commande des systèmes à structure variable peut être commutée d'une valeur à une autre suivant le signe d'une fonction de commutation, avec une fréquence infinie. Cependant, dans les systèmes réels, il est impossible de réaliser une telle commutation de la commande pour des raisons technologique telles que : l'hystérésis, la présence de retard, limitation des actionneurs…etc.

La commande discontinue engendre des oscillations du vecteur d'état et de la commande à une fréquence finie. Ce phénomène est appelé « *le phénomène de broutement* » (chattering en anglais).

Ce phénomène a plusieurs effets indésirables sur le comportement du système. Il peut :

- Exciter des dynamiques non modélisées.
- Diminuer la précision.
- Produire une grande perte d'énergie.
- Créer une fatigue des parties mécanique mobile.

Plusieurs techniques ont alors été proposées pour réduire ou éliminer ce phénomène. Parmi ces techniques on présente deux solutions :

A. La proposition de Slotine [30] [31]

Elle consiste à approximer la commande discontinue par une loi continue au voisinage de la surface, cette fonction a le nom « ***sat*** ». Dans ce cas la commande devient :

$$u = -k\, sat(s) \tag{4-17}$$

la fonction « *sat* » est donnée par :

$$sat(s)=\begin{cases}\dfrac{s}{\Phi} & si\ |s|\le\Phi\\ sign(s) & si\ |s|>\Phi\end{cases} \tag{4-18}$$

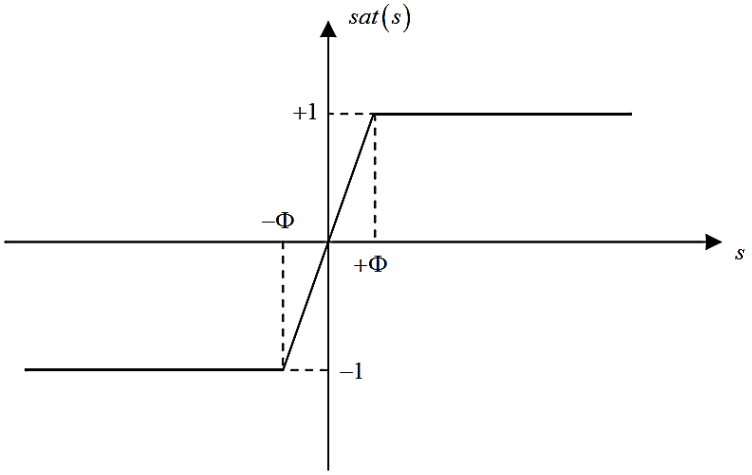


Figure 4.4. *Fonction de saturation « sat ».*

B. La proposition de Harshima

Dans celle-ci, la fonction signe est remplacée par une fonction de lissage appelée « ***cont*** ». Dans ce cas la commande devient :

$$u=-k\,cont(s) \tag{4-19}$$

avec :

$$cont(s)=\begin{cases}\dfrac{s}{|s|+\delta} & si\ |s|\le\Phi\\ sign(s) & |s|>\Phi\end{cases}\qquad avec\ \delta>0 \tag{4-20}$$

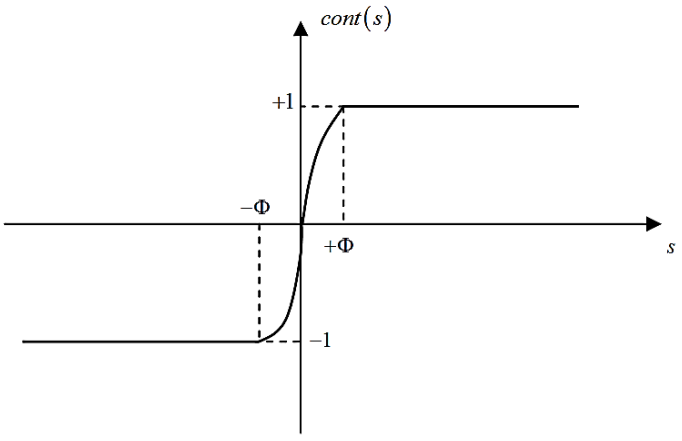


Figure 4.5. *Fonction de saturation « cont ».*

4.2.5. Structures de contrôle par mode de glissement

Dans les systèmes à structure variable utilisant la commande par mode de glissement, on peut trouver trois configurations de base pour la synthèse des différentes commandes. La première correspond à la structure la plus simple où la commutation a lieu au niveau de l'organe de commande lui-même.

On l'appellera, structure par commutation au niveau de l'organe de commande. La deuxième structure fait intervenir la commutation au niveau d'une contre-réaction d'état. Enfin, la dernière structure est la structure par commutation au niveau de l'organe de commande avec ajout de la « *commande équivalente* ». On s'intéresse dans cette étude à cette dernière structure.

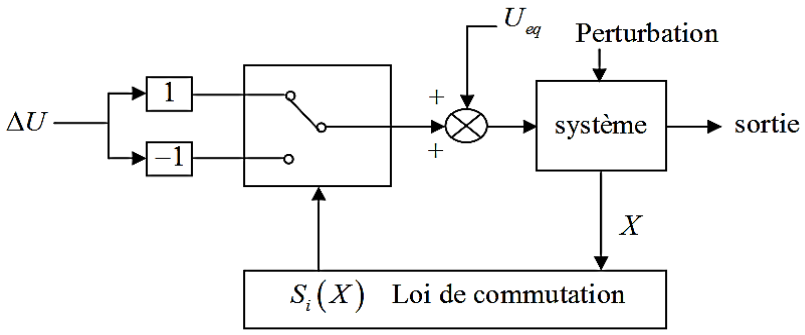


Figure 4.6. *Structure de régulation par ajout de la commande équivalente.*

4.3. Application à la commande d'un système linéaire [32]

4.3.1. Commande discontinue

Considérons le système représenté par l'équation différentielle suivante :

$$\ddot{y} + \dot{y} = u \tag{4-21}$$

Où

u : La commande du système.

y : La sortie du système.

L'objectif est de commander ce système dans le but de suivre un signal de référence y_d

4.3.2. Synthèse de l'algorithme de commande

A. le modèle d'état du système

La représentation d'état du système et donnée par :

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -x_2 + u \end{cases} \quad , \quad y = x_1 \tag{4-22}$$

Où :

$x = [x_1, x_2]^T$ est le vecteur d'état.

Le passage à l'espace des phases de l'erreur, donne :

$$\begin{cases} \dot{e}_1 = e_2 \\ \dot{e}_2 = -e_2 - \dot{y}_d - \ddot{y}_d + u \\ e_1 = y - y_d \end{cases} \tag{4-23}$$

Où : $e = [e_1, e_2]^T$ est le vecteur d'erreur.

$y_d(t)$: est le signal de référence que doit suivre la sortie.

B. La surface de glissement :

La commande à structure variable permet d'amener l'état du système dans l'espace de l'erreur sur une surface de glissement d'équation :

$$s(t) = e_2 + \lambda e_1 \qquad (\lambda > 0) \tag{4-24}$$

Où : λ est une constante de la surface de glissement

C. La dérivée de la surface :

Elle est donnée par :

$$\begin{cases} \dot{s}(t) = \lambda e_2 + \dot{e}_2 \\ \quad = (\lambda - 1)e_2 - \dot{y}_d - \ddot{y}_d + u \end{cases} \tag{4-25}$$

D. Condition de glissement par la méthode de Lyapunov :

La commande u doit être conçue pour rendre la surface $s=0$ attractive en un temps fini, pour cela elle doit donc satisfaire la condition de glissement.

On considère alors l'expression de la fonction de Lyapunov :

$$v = \frac{1}{2}s^2 > 0 \tag{4-26}$$

Sa dérivée par rapport au temps est donnée par :

$$\dot{v} = s.\dot{s} \qquad \forall s \neq 0 \tag{4-27}$$

Comme le système est de degré relative 1 ($\frac{\partial \dot{s}}{\partial u} \neq 0$) avec s comme sortie**,** on peut résoudre par rapport à u l'équation :

$$\dot{s} = -k.sign(s) \tag{4-28}$$

k : constante positive.

Cette équation garantie la convergence vers la surface de glissement en un temps fini, ceci résulte de fait que $\dot{v}$ est définie négative.

En développant les deux membres de l'équation (4-25)

On aura :

$$-k.sign(s) = (\lambda - 1)e_2 + \dot{y}_d - \ddot{y}_d + u$$

$$u = -k.sign(s) + (\lambda - 1)e_2 + \dot{y}_d + \ddot{y}_d \tag{4-29}$$

4.3.3. Résultat de simulation

A. Cas de stabilisation.

Les résultats de simulation pour ($k=5$ et $\lambda=3.5$) sont illustrés (environnement MATLAB 2011) pour les conditions initiales $x = (-0.5,0)^T$ $y_d(t) = 0$ par les figures suivantes :

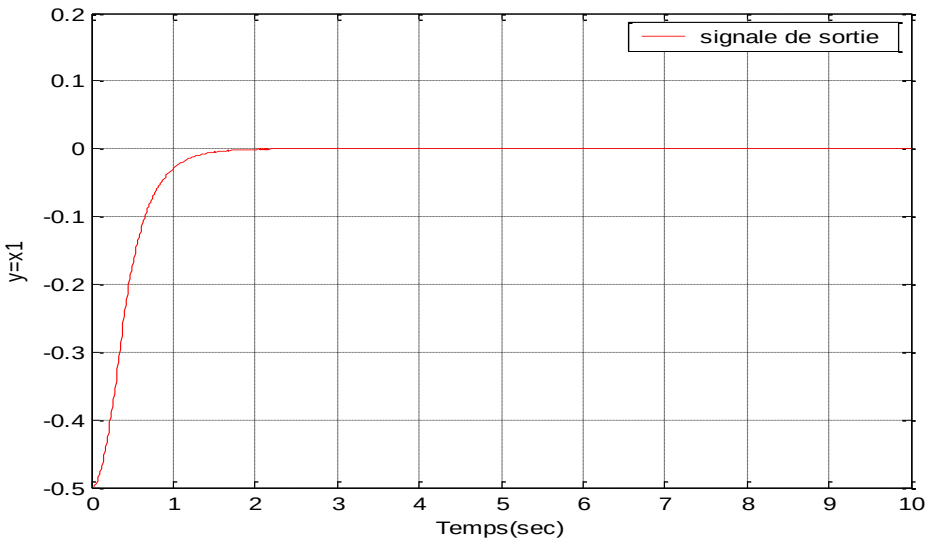


Figure 4.7. *La sortie de système.*

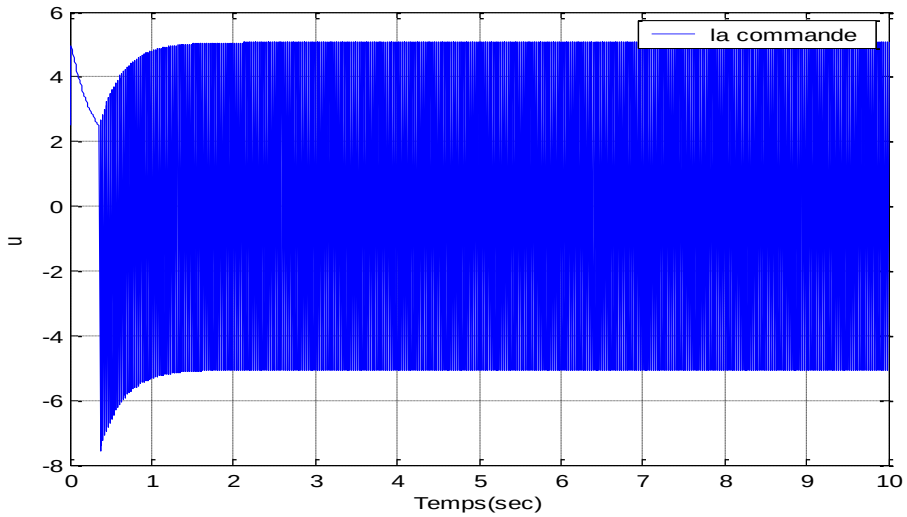


Figure 4.8. *La commande u.*

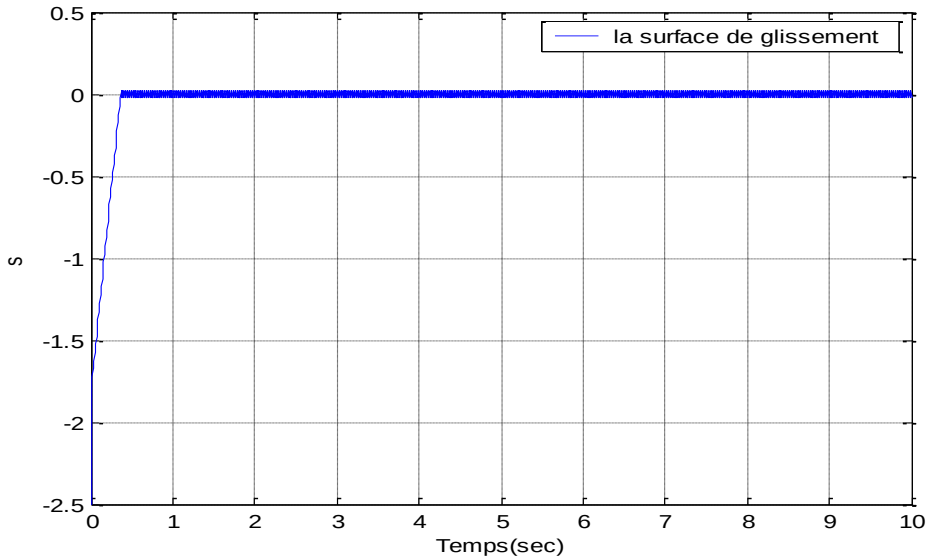


Figure 4.9. *La surface de glissement.*

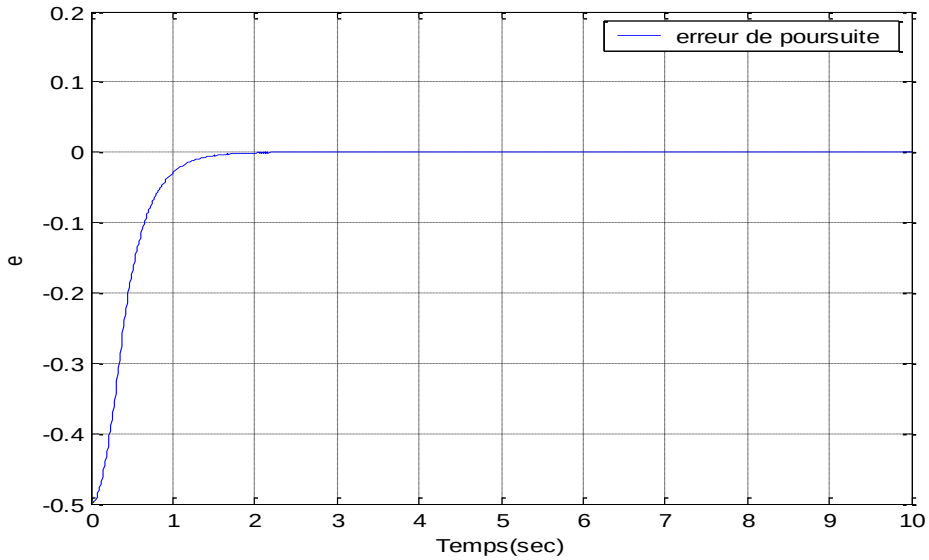


Figure 4.10. *L'erreur de poursuite.*

B. Cas de poursuite

Les résultats de simulation pour *(k=7* et $\lambda=3.5$) sont illustrés (environnement MATLAB 6.5) pour les conditions initiales $x = (0,0)^T$ $y_d(t) = cos(t)$ par les figures suivantes :

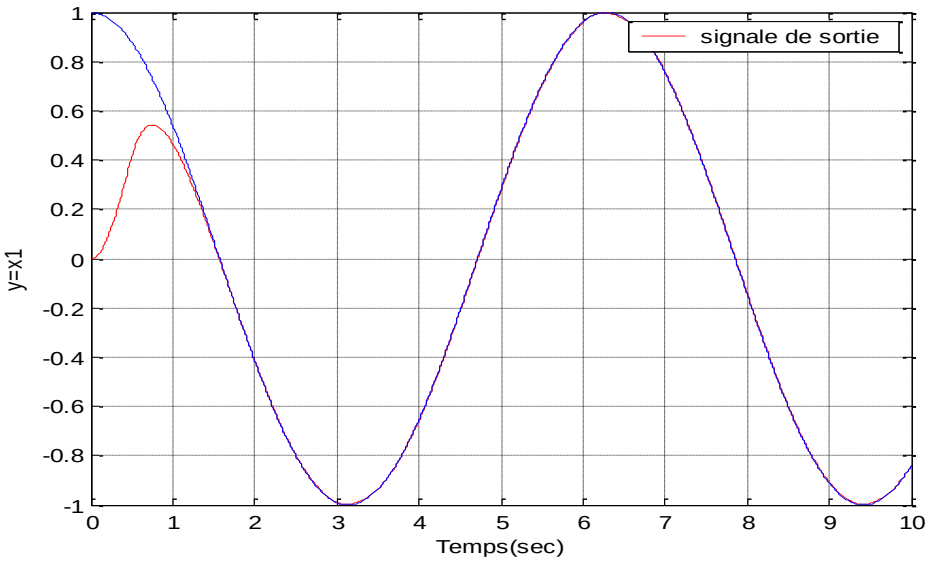


Figure 4.11. *La sortie du système et du signal de référence.*

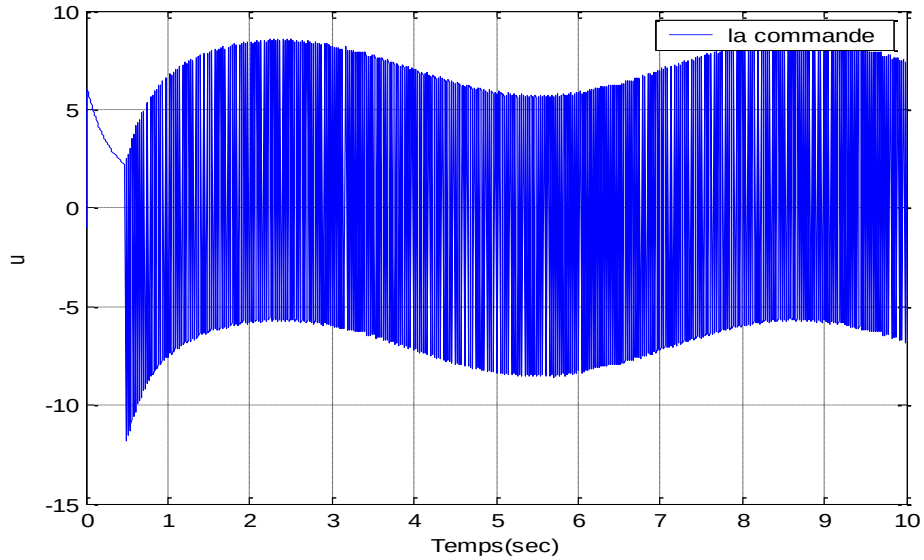


Figure 4.12. *La commande u.*

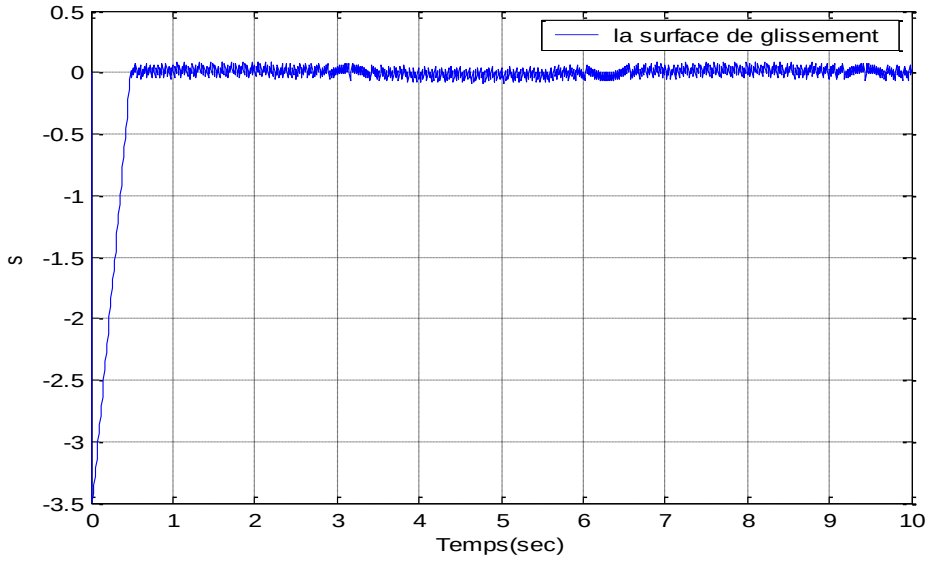


Figure 4.13. *La surface de glissement.*

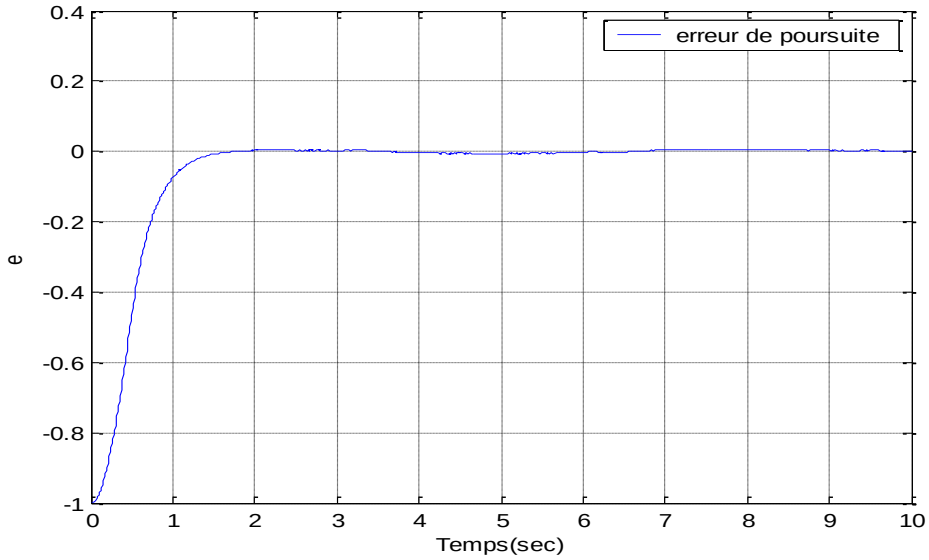


Figure 4.14. *L'erreur de poursuite.*

C. la commande continue (proposition de Slotine) :

Reprenons l'exemple précédent dans le cas de poursuite. On remplace la fonction *sign(s)* par la fonction *sat(s/φ)*, avec φ=0.2. Les résultats obtenus avec cette commande continue sont Illustrés par les figures suivantes :

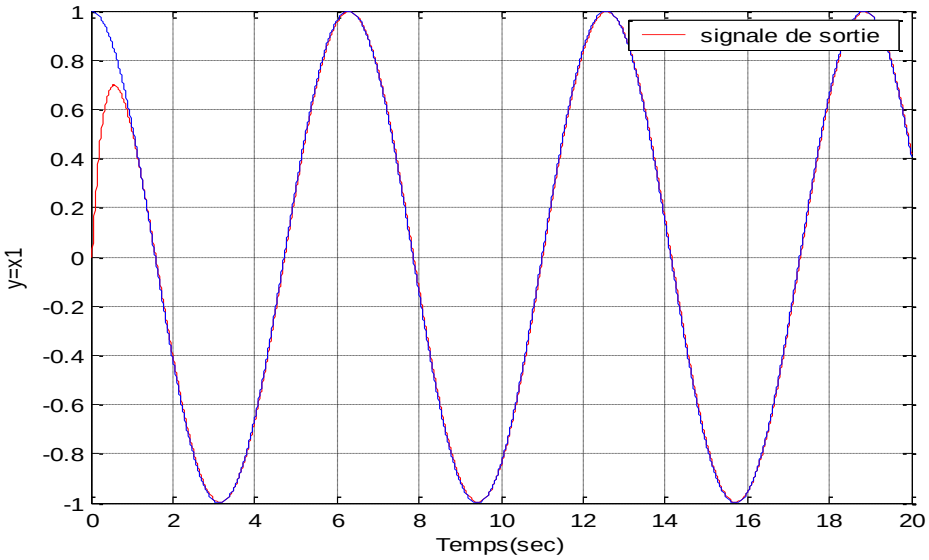


Figure 4.15. *La sortie de système et du signal de référence.*

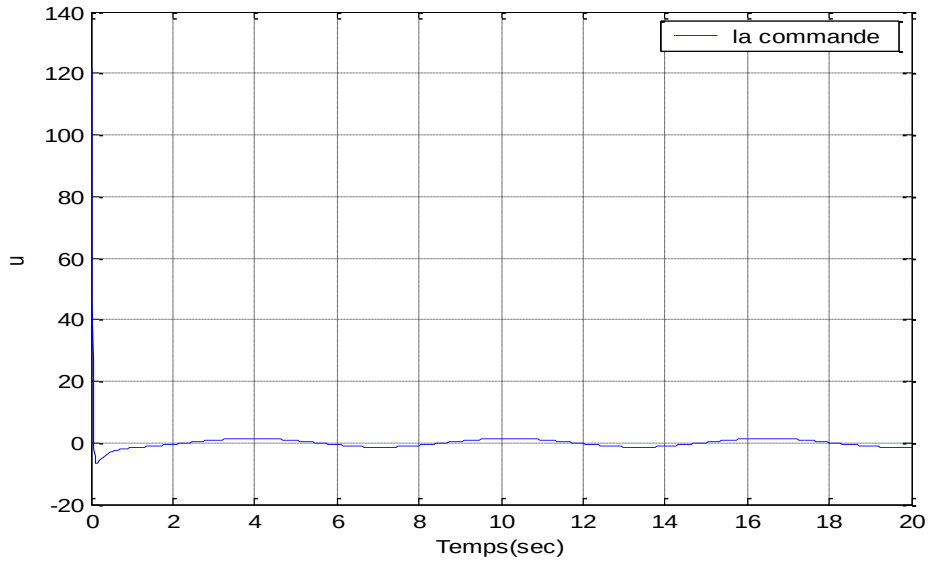


Figure 4.16. *La commande u.*

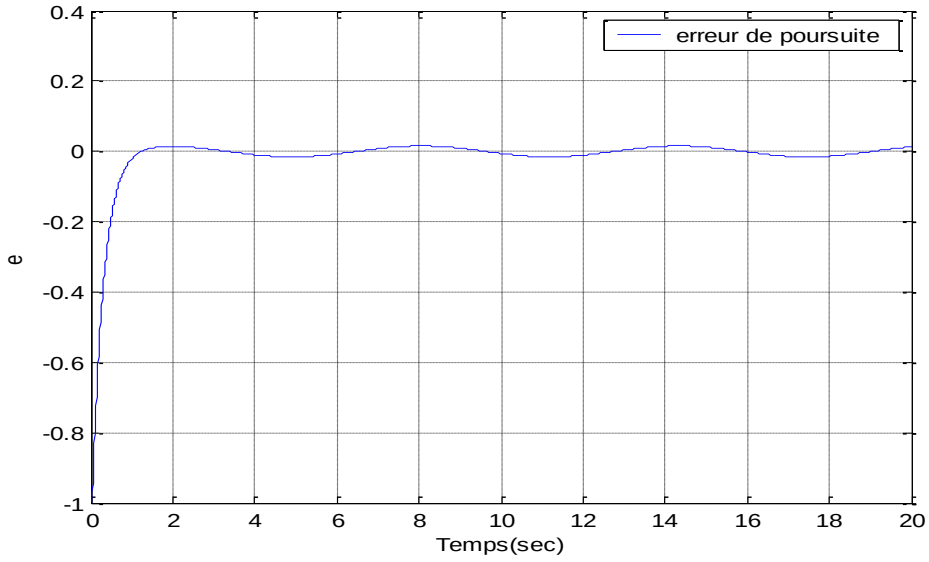


Figure 4.17. *L'erreur de poursuite.*

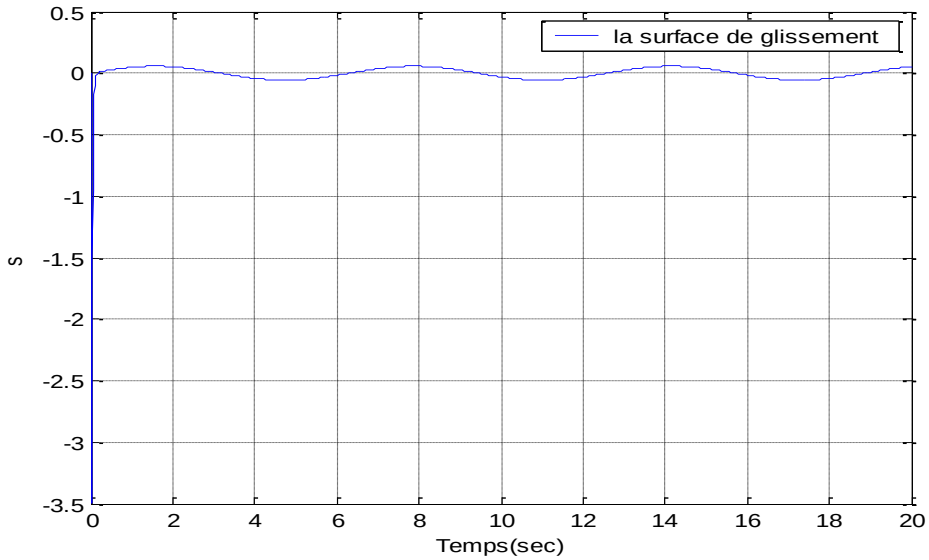


Figure 4.18. *La surface de glissement.*

Il est intéressant de noter que la commande par mode de glissement classique induit d'importantes oscillations indésirables tandis que l'amplitude du broutement après application de la proposition de Slotine est négligeable.

En plus, la convergence de la trajectoire réelle du système vers celle désirée est plus rapide avec la commande continue.

Remarque :

Les valeurs des coefficients λ et k, ont été déterminées par simulation numérique et on tenant compte des critères suivants :

- Rapidité de la réponse sans dépassement important.
- Réduction de l'amplitude des oscillations du régime glissant.
- Optimisation de l'erreur statique.

4.4. Commande par mode de glissement incrémental du pendule inversé (ISMC) [5] [33] [34] [35] [36] [37] [38]

4.4.1. Introduction

La commande des systèmes sous actionnées a été largement étudiée par les chercheurs dans divers domaines tels que la robotique, l'aérospatial, l'ingénierie de la marine…etc. Contrairement aux systèmes qui sont complètement actionnés, ces systèmes sont difficiles à contrôler à cause de la propriété de la sous action.

Plusieurs classifications dans des travaux de recherche qui traitent le problème de stabilisation et de commande ont été présentés pour ces systèmes. Parmi ces travaux on trouve celui de Spong (1996) et de Bloch (1999) où ils ont utilisé la technique de la linéarisation partielle et celui de Fantoni (2000) dont l'approche est basée sur l'énergie du système, mais ces contrôleurs élaborés commutent à un contrôleur linéaire une fois le point d'équilibre est atteint. L'inconvénient de ces approches c'est que le domaine d'attraction est restreint. On trouve aussi les travaux de Reza Olfati-Saber (2001) et Sun et Lee (2001) qui ont introduit l'approche du backstepping aux systèmes sous actionnés mais cette dernière exige au système qu'il soit de la forme triangulaire.

Les travaux de Yi-Jen Mon (2002) et Yinxing Hao (2006) montre l'efficacité et l'applicabilité de la commande par mode de glissement aux systèmes sous actionnés. Dans ce travail on propose un algorithme de commande en temps réel par mode de glissement de type incrémental (en anglais ISMC) qui stabilise d'une manière robuste un système sous-actionné.

4.4.2. Principe de la commande ISMC

La forme normale d'un système sous actionnés classe deux, est donnée comme suit:

$$\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= f_1(x) + b_1(x)u \\ \dot{x}_3 &= x_4 \\ \dot{x}_4 &= f_2(x) + b_2(x)u \\ y(t) &= [x_1, x_2, x_3, x_4]^T \end{aligned} \tag{4-30}$$

Où $x = (x_1, x_2, x_3, x_4)^T$ est le vecteur d'état, $f_1(x)$, $f_2(x)$ $b_1(x)$ et $b_2(x)$ sont les fonctions non linéaires nominales bornées. (x_1, x_2) et (x_3, x_4) sont les états de deux sous-systèmes.

L'objective est de construire l'entrée de commande *u* qui mènent simultanément les erreurs (e_1,e_2,e_3,e_4) vers zéro tel que :

$$\begin{aligned} e_1 &= x_1 - x_{1d} \\ e_2 &= x_2 - x_{2d} \\ e_3 &= x_3 - x_{3d} \\ e_4 &= x_4 - x_{4d} \end{aligned} \tag{4-31}$$

et x_{1d} , x_{2d} , x_{3d} , x_{4d} sont les signaux de référence.

4.4.3. Choix des surfaces de glissement

D'abord, les variables (e_1 , e_2) sont choisis pour construire la première surface, comme suit :

$$s_1 = \lambda_1 e_1 + \lambda_2 e_2 \tag{4-32}$$

Avec λ_1 et λ_2 sont des constantes ayant le même signe.

Puis, la première surface de la couche s_1 peut être considérée comme une des variables d'état général, nous pouvons l'utiliser avec une des variables d'état pour construire la seconde surface de la couche s_2 :

$$s_2 = \lambda_3 e_3 + s_1 \tag{4-33}$$

De même, la troisième surface de la couche s_3 peut être écrite comme:

$$s_3 = \lambda_4 e_4 + s_2 \tag{4-34}$$

λ_3 et λ_4 sont des constantes.

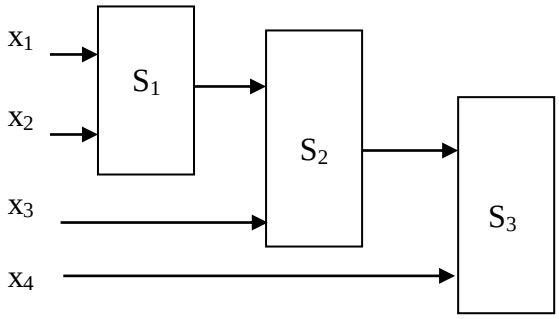


Figure 4.19. *Structures des surfaces de glissement.*

4.4.4. Condition d'existence et élaboration de la loi de commande

La loi totale de la commande du système est donnée par :

$$u = u_{eq} + u_{sw} \tag{4-35}$$

Tel que : u_{eq} est la commande équivalente.

u_{sw} est la commande de commutation.

La loi de la commande équivalente du système complet est construite à base des lois de commande équivalente des sous-systèmes, de la manière suivante :

$$\dot{s}_1 = 0 \Rightarrow \lambda_1 \dot{e}_1 + \lambda_2 \dot{e}_2 = 0 \Rightarrow \lambda_2 \dot{e}_2 = -\lambda_1 \dot{e}_1 = \lambda_2 (f_1 + b_1 u_{eq1} - \dot{x}_{2d})$$

On a (4-36)

$$\Rightarrow u_{eq1} = \frac{-\lambda_1 \dot{e}_1 - \lambda_2 f_1 + \lambda_2 \dot{x}_{2d}}{\lambda_2 b_1} = \frac{-(\lambda_1 x_2 + \lambda_2 f_1 - \lambda_2 x_{2d} - \lambda_1 \dot{x}_{1d})}{\lambda_2 b_1}$$

On a :

$$\dot{s}_2 = 0 \Rightarrow \lambda_3 \dot{e}_3 + \dot{s}_1 = 0 \Rightarrow \lambda_3 \dot{e}_3 + \lambda_1 \dot{e}_1 + \lambda_2 \dot{e}_2 = 0 \Rightarrow \lambda_2 \dot{e}_2 = -(\lambda_3 \dot{e}_3 + \lambda_1 \dot{e}_1)$$
$$-\lambda_3 (x_4 - \dot{x}_{3d}) - \lambda_1 (x_2 - \dot{x}_{1d}) = \lambda_2 (f_1 + b_1 u_{eq2} - \dot{x}_{2d})$$

$$\Rightarrow u_{eq2} = \frac{-(\lambda_1 x_2 + \lambda_3 x_4 + \lambda_2 f_1 + \lambda_1 \dot{x}_{1d} + \lambda_2 \dot{x}_{2d} + \lambda_3 \dot{x}_{3d})}{\lambda_2 b_1} \tag{4-37}$$

On a :

$$\dot{s}_3 = 0 \Rightarrow \lambda_4 \dot{e}_4 + \dot{s}_2 = 0 \Rightarrow \lambda_4 \dot{e}_4 + \lambda_3 \dot{e}_3 + \lambda_1 \dot{e}_1 + \lambda_2 \dot{e}_2 = 0$$
$$\Rightarrow \lambda_4 (f_2 + b_2 u_{eq3} - \dot{x}_{4d}) + \lambda_3 (x_4 - \dot{x}_{3d}) + \lambda_1 (x_2 - \dot{x}_{1d}) + \lambda_2 (f_1 + b_1 u_{eq3} - \dot{x}_{2d}) = 0$$

$$\Rightarrow u_{eq3} = \frac{-(\lambda_1 x_2 + \lambda_3 x_4 + \lambda_4 f_2 + \lambda_2 f_1 - \lambda_1 \dot{x}_{1d} - \lambda_2 \dot{x}_{2d} - \lambda_3 \dot{x}_{3d} - \lambda_4 \dot{x}_{4d})}{\lambda_4 b_2 + \lambda_2 b_1} \tag{4-38}$$

Remarque

Quand s_3 converge vers zéro, la commande u_{eq3} garantie que le système reste sur la surface de glissement jusqu'à ce que tous les erreurs de s_3 (e_4 et s_2) convergent vers zéro. La commande u_{eq3} se dégrade à la commande u_{eq2} qui garantit que le système reste sur la surface de glissement s_2 jusqu'à ce que tous les erreurs de s_2 (e_3 *et* s_1) convergent vers zéro. La commande u_{eq2} se dégrade à la commande u_{eq1} qui mène les erreurs e_1 *et* e_2 vers zéro.

A partir de la fonction de Lyapunov v_3 on peut faire sortir l'expression de la commande de commutation :

On a :
$$v_3 = \frac{1}{2}s_3^2 \tag{4-39}$$

et :
$$\begin{cases} \dot{v}_3 = s\dot{s}_3 = s_3(\lambda_4\dot{e}_4 + \dot{s}_2) = s_3(\lambda_4 f_2 + \lambda_4 b_2 u + \lambda_3 x_4 + \lambda_1 x_2 + \\ \quad \lambda_2 f_1 + \lambda_2 b_1 u - \lambda_1 \dot{x}_{1d} - \lambda_2 \dot{x}_{2d} - \lambda_3 \dot{x}_{3d} - \lambda_4 \dot{x}_{4d}) \\ \quad = s_3(\lambda_4 f_2 + \lambda_3 x_4 + \lambda_1 x_2 + \lambda_2 f_1 + (\lambda_4 b_2 + \lambda_2 b_1)u_{eq} + \\ \quad (\lambda_4 b_2 + \lambda_2 b_1)u_{sw} - \lambda_1 \dot{x}_{1d} - \lambda_2 \dot{x}_{2d} - \lambda_3 \dot{x}_{3d} - \lambda_4 \dot{x}_{4d}) \end{cases} \tag{4-40}$$

Choisissons :

$$u_{sw} = \frac{-1}{(\lambda_4 b_2 + \lambda_2 b_1)}(k\,sign(s_3) + Cs_3) \tag{4-41}$$

$$u_{eq} = \frac{-1}{(\lambda_4 b_2 + \lambda_2 b_1)}(\lambda_4 f_2 + \lambda_3 x_4 + \lambda_1 x_2 + \lambda_2 f_1 - \lambda_1 \dot{x}_{1d} - \lambda_2 \dot{x}_{2d} - \lambda_3 \dot{x}_{3d} - \lambda_4 \dot{x}_{4d}) \tag{4-42}$$

Pour que $\dot{v}_3 \le 0$

$$\dot{v}_3 = s_3(-k\,sign(s_3) - Cs_3) = -ks_3 sign(s_3) - Cs_3^2$$
$$\dot{v}_3 = -k|s_3| - Cs_3^2 \le 0 \tag{4-43}$$

k *et* C : sont des constantes positives.

4.4.5. Théorème [2]

On considère le système sous actionné d'ordre deux (4-30) avec les lois de commande définies par (4-35), (4-41) et (4-42) dont, les surfaces de glissement sont donnés par (4-32) (4-33) et (4-34), ce système est globalement stable quand tous les signaux sont bornés, les erreurs convergent asymptotiquement vers zéro et les surfaces de glissement sont asymptotiquement stables.

4.4.6. Simulation numérique

Une simulation numérique est effectuée, la commande ISMC est appliquée sur le système du pendule inversé modélisé dans le chapitre 2. Les constantes sont déterminées par simulation numérique et données par: $\lambda_1 = 100$, $\lambda_2 = 28.489$, $\lambda_3 = -50$,$\lambda_4 = -41$, $C = 0.4$, $k = 50$.

Les signaux de référence de la position et de la vitesse désirées sont pris comme un signal sinusoïdal. Les résultats obtenus sont représentés dans les figures suivantes :

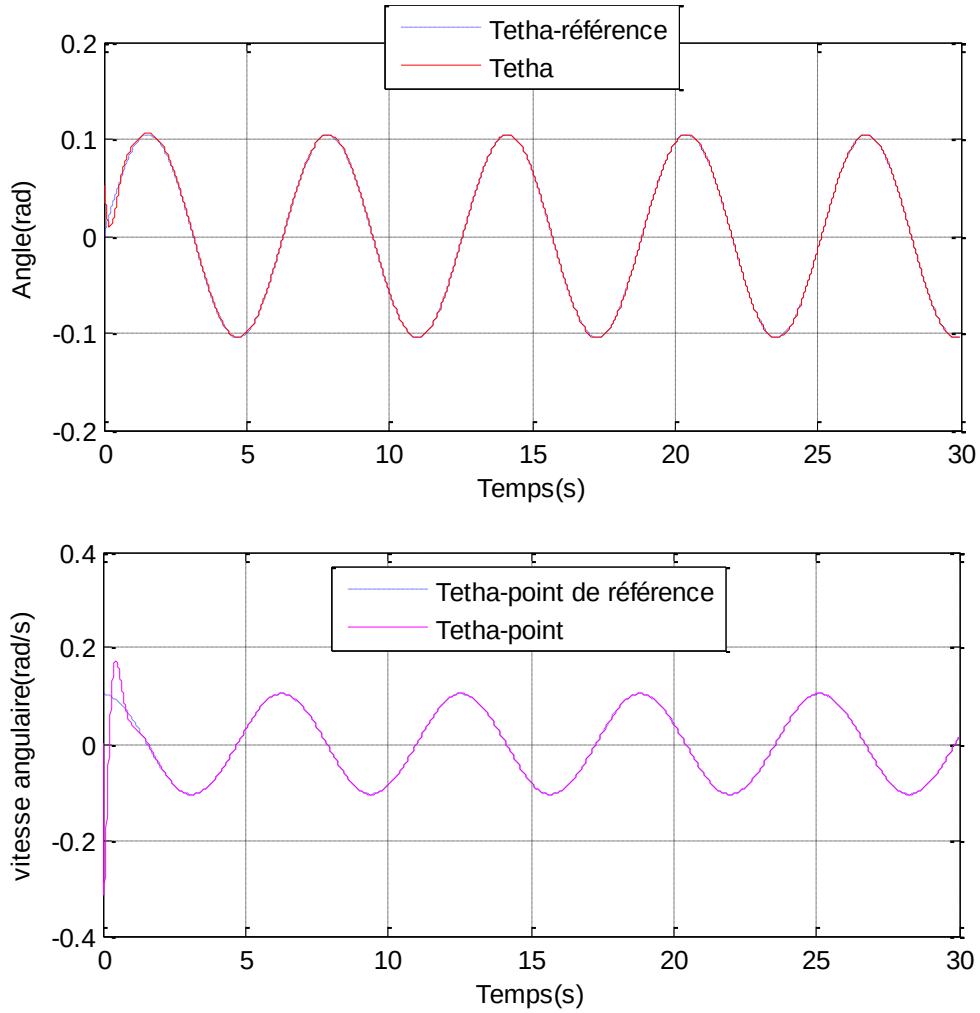


Figure 4.20. *Courbes de l'angle et de la vitesse angulaire.*

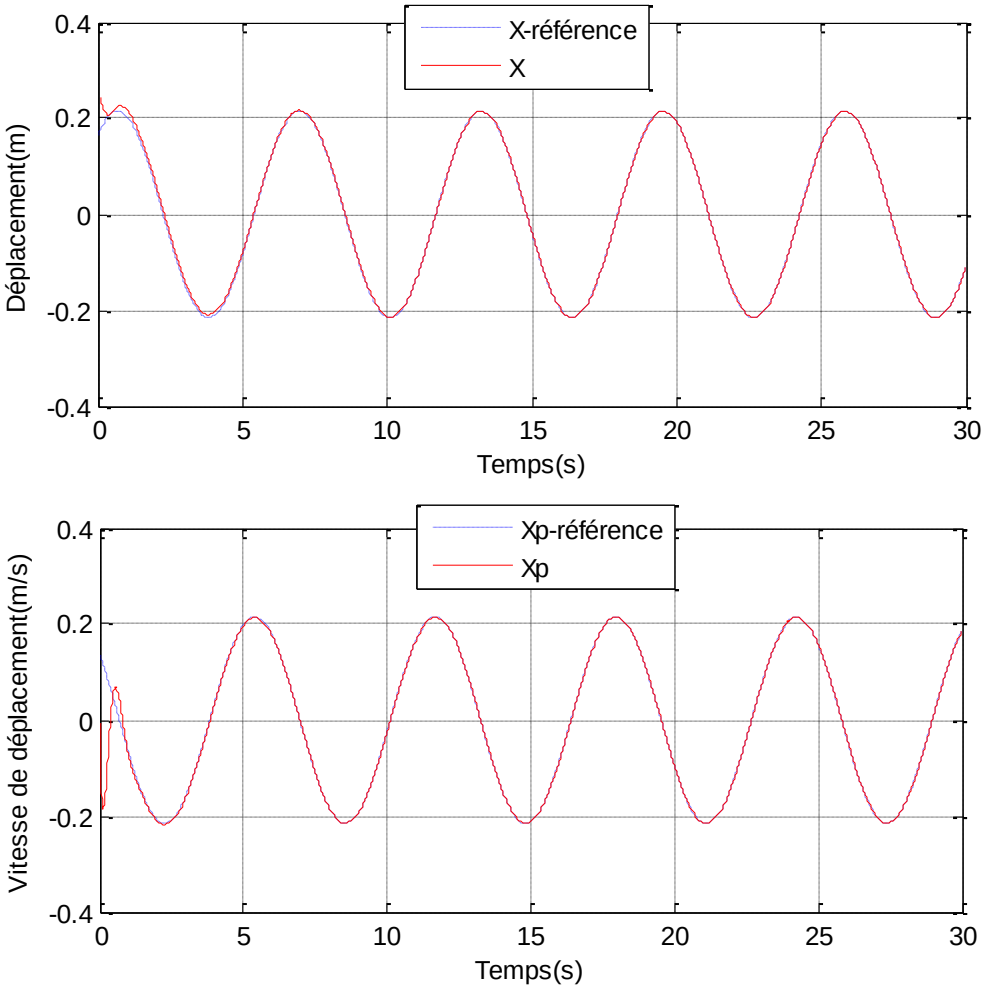


Figure 4.21. *Courbes de position et de vitesse linéaire du chariot.*

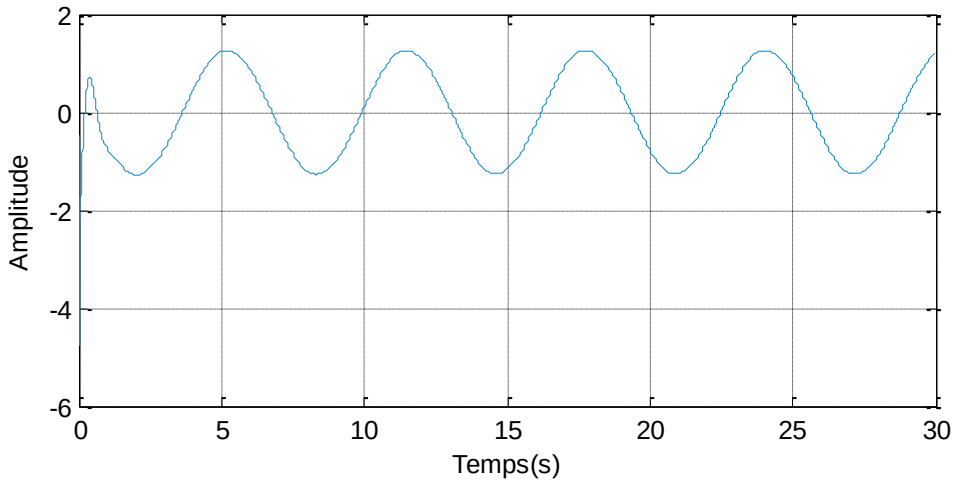


Figure 4.22. *Courbe de commande.*

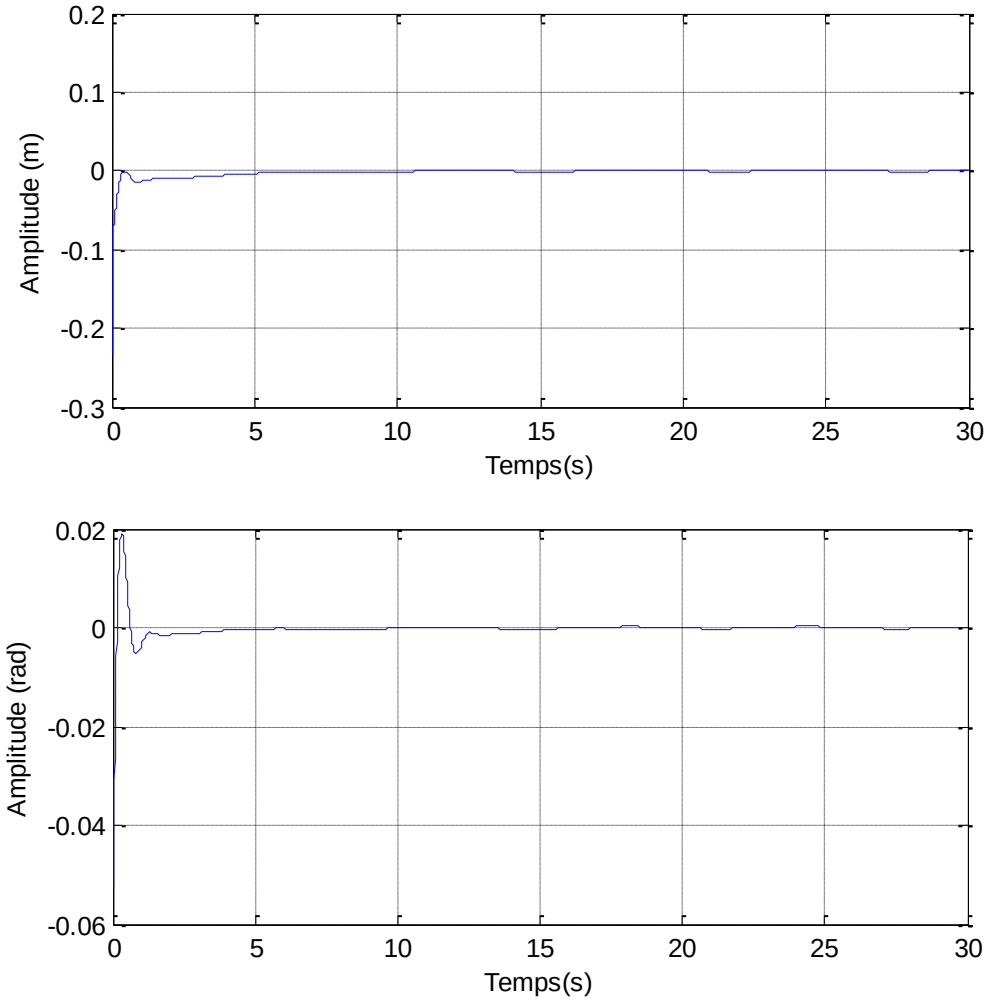


Figure 4.23. *Courbes de l'erreur d'angle et de déplacement (position).*

Les résultats obtenus montrent que toutes les variables d'états du système convergent vers l'état désirée quel que soit les conditions initiales, ce qui prouve la robustesse de la commande.

4.5. Avantages et inconvénient de la commande par mode de glissement [4]

- Un des principaux avantages résidant dans l'utilisation de la commande par mode de glissement est que tout système commandé par cette technique aura des propriétés implicites de robustesse.
- En revanche, théoriquement l'utilisation directe d'une telle commande, impose la nécessité d'appliquer au système une commutation logique à vitesses élevées, bien que dans la pratique, les actionneurs ne puissent pas produire la commutation à ces vitesses, celle-ci doivent être suffisamment rapide.

- Quand l'état du système se trouve à la limite de changement de structure, des commutations successives à fréquences élevées peuvent se produire, ce phénomène de broutement pouvant produire une sollicitation excessive des actionneurs. Pour certaine applications, le broutement n'est pas acceptable à cause de deux raisons principales :
  - Pour des fréquences de broutement élevé, dans certains cas, le système n'est plus modélisé correctement et peut devenir instable en boucle fermée.
  - Le broutement produit une dynamique importante sur les actionneurs (moteurs), ce qui peut réduire leurs (durée de vie) et d'être la cause de vibrations dangereuses.

4.6. Conclusion

La commande par mode de glissement est une commande qui fait partie de la 2^{eme} catégorie des approches de commande des systèmes non-linéaires et qui se basent sur les dynamiques non linéaires des systèmes.

Dans ce chapitre on a proposé un algorithme de commande en temps réel par mode de glissement incrémental, où on a appliqué cet algorithme sur le pendule inversé qui est un système sous actionnées.

Les résultats obtenus montrent l'efficacité et la robustesse de la commande par mode de glissement. En effet, Concernant l'application du pendule inversé, les variables d'états convergent vers l'état désirée quel que soit les conditions initiales.

Chapitre 5

Commande par réseaux de neurones artificiels

5.1. Introduction [18]

Les réseaux de neurones « RNA » sont issus des techniques de l'intelligence artificielle. Ils sont caractérisés par leurs comportements non-linéaires, leur parallélisme, et leurs capacités d'optimisation et d'apprentissage.

Ces caractéristiques très avantageuses sont derrière l'augmentation de la popularité des réseaux de neurones pour la modélisation numérique, l'estimation et la commande, spécialement pour les systèmes peu connus sur leur dynamique et leur environnement d'opération. Le théorème d'approximation universelle des réseaux de neurones, montre que les réseaux de neurones sont capable d'approximer n'importe quel fonction non-linéaire arbitraire à n'importe quel niveau de performance.

Aussi c'est un outil très puissant pour les systèmes de dynamiques inconnus ou incertains. Dans ce chapitre, on va présenter des notions de base sur les réseaux de neurones, ainsi que leur utilisation pour résoudre des problèmes de commande en temps réel.

5.2. Notions sur les réseaux de neurones

5.2.1. Réseaux de neurones biologiques [49]

Les cellules nerveuses, appelées neurones, sont les éléments de base du système nerveux central. Celui-ci en posséderait environ cent milliards. Les neurones possèdent de nombreux points communs dans leur organisation générale et leur système biochimique avec les autres cellules (figure 5.1).

Ils présentent cependant des caractéristiques qui leur sont propres et se retrouvent au niveau des cinq fonctions spécialisées qu'ils assurent et qui sont les suivantes :

- Recevoir des signaux en provenance de neurones voisins.
- Intégrer ces signaux.
- Engendrer un influx nerveux.
- Le conduire.
- Le transmettre à un autre neurone capable de le recevoir.

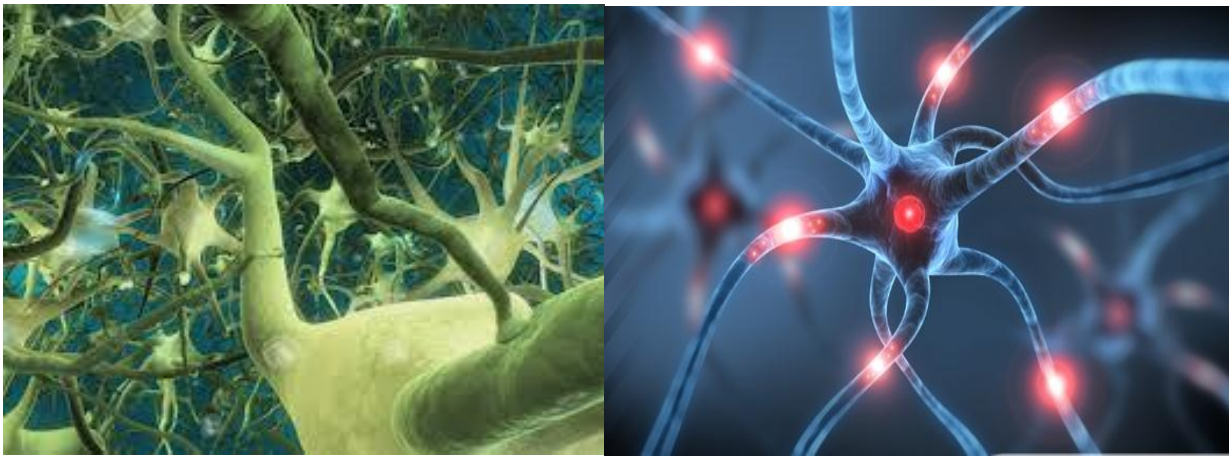


Figure 5.1. *Représentation d'un réseau de neurone biologique.*

Le neurone biologique est composé de quatre parties distinctes **[39]** :

- ***Le corps cellulaire*** **:** qui contient le noyau de la cellule nerveuse, est l'endroit où prend naissance l'influx, qui représente l'état d'activité du neurone.
- ***Les dendrites*** **:** sont des ramifications tubulaires courtes formant une espèce d'arborescence autour du corps cellulaire. Elles sont les entrées principales du neurone, qui captent l'information venant d'autres neurones.
- ***L'axone*** **:** est une longue fibre nerveuse qui se ramifie à son extrémité. C'est la sortie du neurone et le support de l'information vers les autres neurones.
- ***La synapse*** **:** communique l'information, en la pondérant par un poids synaptique, à un autre neurone. Elle est essentielle dans le fonctionnement du système nerveux.

5.2.2. Réseaux de neurones artificiels

5.2.2.1 Neurone formel

La première étude systématique du neurone artificiel est due au neuropsychiatre McCulloch et au logiciel Pitts qui, s'inspirant de leurs travaux sur les neurones biologiques, proposèrent en 1943 le modèle illustré à la figure (5.2).

Ce neurone formel est un processeur élémentaire qui réalise une somme pondérée des signaux qui lui parviennent. La valeur de cette sommation est ensuite comparée à un seuil, et la sortie du neurone est une fonction non linéaire du résultat [39].

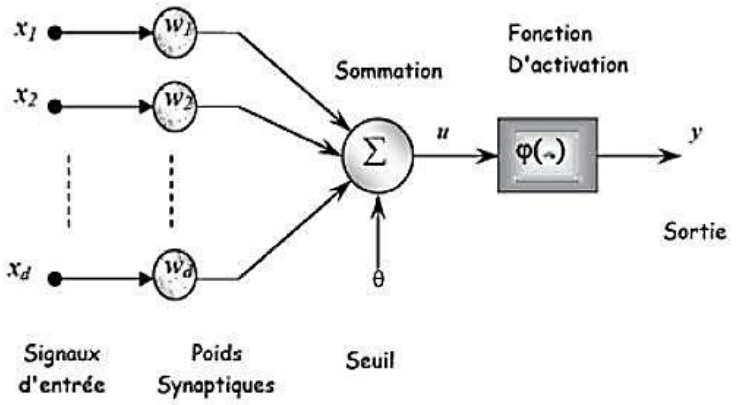


Figure 5.2. *Le modèle de neurone formel [39].*

Les équations qui représentent le modèle :

$$u = \sum_{j=1}^{d} w_j x_j - \theta \qquad (5\text{-}1)$$

$$y = f(u) \qquad (5\text{-}2)$$

- En notant par x le vecteur d'entrée : $x = [\, x_1, x_2, \ldots, x_d]^T$
- Et par w le vecteur des poids: $w = [\, w_1, w_2, \ldots, w_d\,]^T$

On obtient : $y = f\,(w^T, x)$ Avec :

- θ: Un terme additionnel représentant le seuil interne du neurone, ce terme est considéré comme un poids w_0 associé à une entrée constante.
- $\varphi(.)$: Fonction d'activation.
- $\sum$: Fonction de sommation.

D'une façon plus générale, on peut définir un neurone formel par les cinq éléments suivants [40]:

- La nature de ses entrées.
- La fonction d'entrée totale qui définit le prétraitement effectué sur les entrées.
- La fonction d'activation (ou d'état) du neurone qui définit son état interne en fonction de son entrée totale.
- La fonction de sortie qui calcule la sortie du neurone en fonction de son état d'activation.
- La nature de la sortie du neurone.

5.2.2.2 . Fonction d'activation

C'est une fonction appelée aussi fonction de seuil. Elle permet de définir l'état interne du neurone en fonction de son entrée totale [41].

Différentes fonctions pouvant être utilisées comme fonction d'activation du neurone. Elles sont énumérées au tableau donné par la figure (5.3). Les trois les plus utilisées sont les fonctions 'seuil', 'linéaire' et 'sigmoïde'.

Nom de la fonction	Relation d'entrée/sortie	Icône	Nom Matlab
seuil	$a = 0 \quad si\ n < 0$ $a = 1 \quad si\ n \geq 0$		hardlim
seuil symétrique	$a = -1 \quad si\ n < 0$ $a = 1 \quad si\ n \geq 0$		hardlims
linéaire	$a = n$		purelin
linéaire saturée	$a = 0 \quad si\ n < 0$ $a = n \quad si\ 0 \leq n \leq 1$ $a = 1 \quad si\ n > 1$		satlin
linéaire saturée symétrique	$a = -1 \quad si\ n < -1$ $a = n \quad si\ -1 \leq n \leq 1$ $a = 1 \quad si\ n > 1$		satlins
linéaire positive	$a = 0 \quad si\ n < 0$ $a = n \quad si\ n \geq 0$		poslin
sigmoïde	$a = \frac{1}{1+\exp^{-n}}$		logsig
tangente hyperbolique	$a = \frac{e^n - e^{-n}}{e^n + e^{-n}}$		tansig
compétitive	$a = 1 \quad si\ n\ maximum$ $a = 0 \quad autrement$		compet

Figure 5.3. *Fonctions de transfert a = f(n) [42].*

5.2.2.3 . Perceptron simple [42]

Un réseau de neurones est un maillage de plusieurs neurones, généralement organisé en couches. Pour construire une couche de S neurones, il s'agit simplement de les assembler comme à la figure (5.4).

Les S neurones d'une même couche sont tous branchés aux R entrées. On dit alors que la couche est totalement connectée.

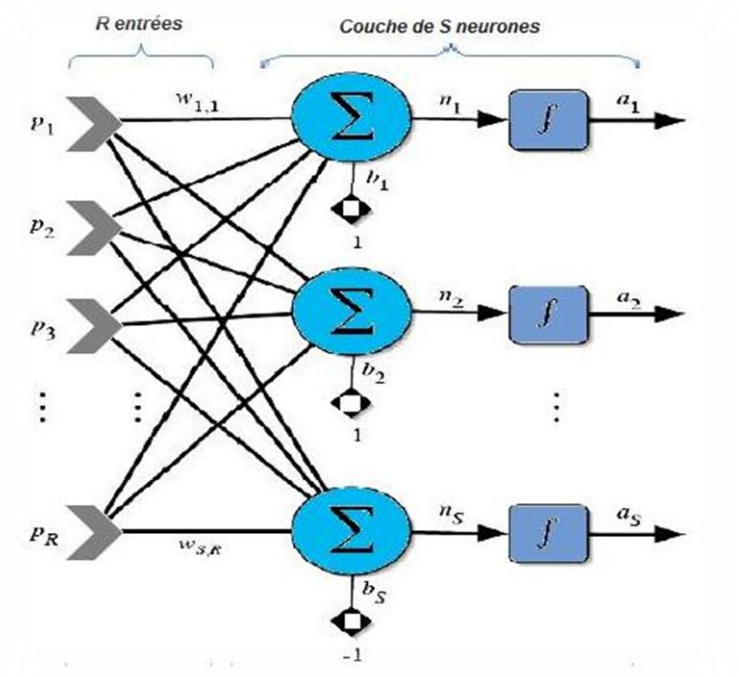


Figure 5.4. *Couche de Sn neurones [42].*

Un poids $W_{i,j}$ est associé à chacune des connexions. Le premier indice est noté par i et le deuxième par j (jamais l'inverse). Le premier indice (rangée) désigne toujours le numéro de neurone sur la couche, alors que le deuxième indice (colonne) spécifie le numéro de l'entrée. Ainsi, $W_{i,j}$ désigne le poids de la connexion qui relie le neurone i à son entrée j. L'ensemble des poids d'une couche forme donc une matrice W de dimension Sn × R :

$$W = \begin{pmatrix} w_{1,1} & w_{1,2} \dots & w_{1,R} \\ w_{2,1} & w_{2,2} \dots & w_{2,R} \\ \vdots & \vdots & \vdots \\ w_{S,1} & w_{1,2} \dots & w_{S,R} \end{pmatrix}$$

On note que :

Sn : est le nombre de neurones, R : est le nombre d'entrées.

5.2.2.4 . Perceptron multicouche (MLP)

Le perceptron multicouche (Multi Layer Perceptron (MLP)) est une amélioration du perceptron simple comprenant une ou plusieurs couches intermédiaires dites cachées. Il utilise pour modifier ses poids, un algorithme de rétro-propagation du gradient [39].

Les réseaux multicouches sont beaucoup plus puissants que les réseaux simples à une seule couche. La mise en cascade du perceptron conduit à ce qu'on appelle le perceptron multicouche comme indique la figure (5.5).

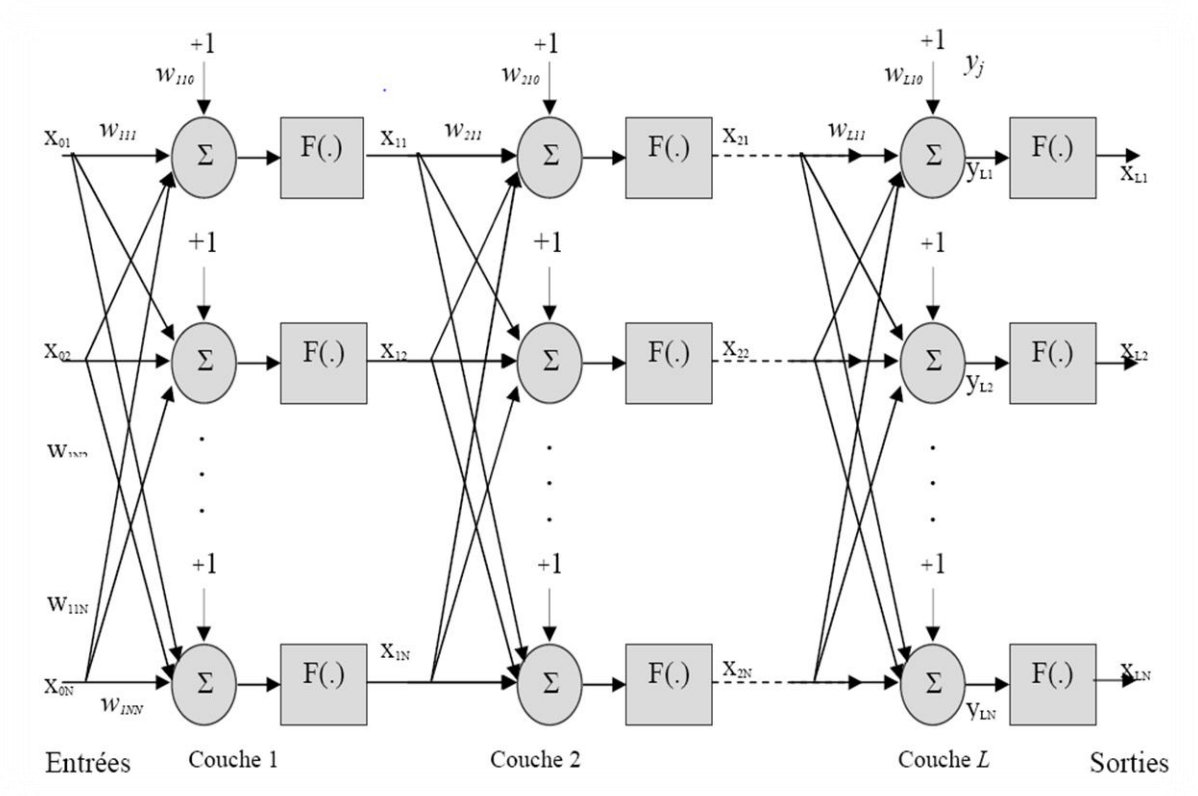


Figure 5.5. *Architecture du perceptron multicouche [41].*

Lorsque le vecteur de caractéristiques d'un objet est présenté à l'entrée du réseau, il est communiqué à tous les neurones de la première couche. Les sorties des neurones de cette couche sont alors communiquées aux neurones de la couche suivante et ainsi de suite.

La dernière couche du réseau est appelée couche de sortie, les autres étant désignées sous le terme de couches cachées car les valeurs de sortie de leurs neurones ne sont pas accessibles de l'extérieur [39].

5.2.3 . Architecture du réseau de neurones artificiel [43]

L'architecture définit le fonctionnement du réseau, il existe deux types de réseaux à architecture différentes.

5.2.3.1 .Réseaux statiques (non bouclés)

Ce type de réseaux est organisé généralement en couches de neurones. Chaque neurone d'une couche reçoit ses entrées à partir de la couche précédente ou de l'entrée du réseau. Dans de tels réseaux, il n'existe pas de 'feedback', c'est à dire qu'il n'y a pas de retour d'information.

5.2.3.2 .Réseaux dynamiques (bouclés)

L'introduction du retour 'feedback' entre les neurones, rend le réseau dynamique, cette structure dynamique est modélisée dans la figure (5.6).

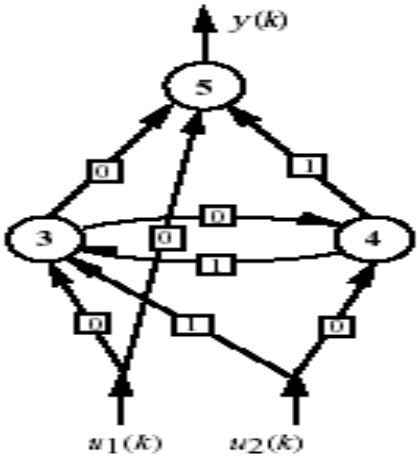


Figure 5.6. *Architecture générale d'un réseau dynamique [44].*

Avec : $u_1(k)$ et $u_2(k)$ sont les entrées du réseau. $y(k)$: est la sortie du réseau.

5.2.4 . L'apprentissage

L'objectif de l'apprentissage est de fournir une méthode au réseau afin qu'il puisse ajuster ces paramètres lorsqu'on lui présente des exemplaires à traiter **[**40**]**. On distingue habituellement trois types d'apprentissage, l'apprentissage supervisé, l'apprentissage semi-supervisé et l'apprentissage non supervisé.

5.3 . Rétro-propagation (Back propagation)

5.3.2 . Introduction

Le problème de calcul des erreurs de sortie pour les neurones cachés, limite l'efficacité des algorithmes d'apprentissage supervisé pour les réseaux multicouches.

Donc à cause de l'absence d'une règle d'apprentissage convenable, les perceptrons étaient limités à une seule couche pendant plusieurs années. Le problème du calcul des erreurs associées aux neurones cachés a été levé par la découverte de l'algorithme de rétro propagation qui peut être appliqué à n'importe quel système composé de plusieurs sous-systèmes élémentaires qui peuvent être représentés par des fonctions connues, continues et dérivables [40].

5.3.3 . Définition [43]

C'est une technique de calcul des dérivées qui peut être appliquée à n'importe qu'elle structure. Dans cette méthode, de même que l'on est capable de propager un signal provenant des neurones d'entrée vers la couche de sortie, on peut, en suivant le chemin inverse, rétro-propager l'erreur commise en sortie vers les couches cachées, d'où le nom rétro-propagation. La figure (5.7) représente le principe de l'entrainement du réseau par rétro-propagation de l'erreur.

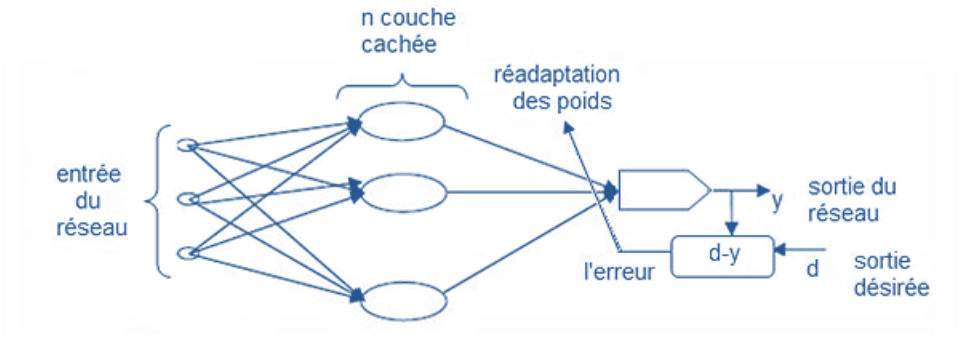


Figure 5.7. *Principe de l'entrainement du réseau par rétro-propagation de l'erreur [40].*

5.3.4 . Equations du réseau [41]

Avant de définir la règle d'apprentissage, on doit définir la relation entre les sorties du réseau, d'une part, et les entrées et les poids d'autre part. On considère dans ce qui suit les réseaux multicouches.

Pour un réseau multicouches à m entrées et ni sorties, composé de L couches (couches cachées et couche de sortie), les états des neurones sont donnés par les équations suivantes :

$$S_k^i = \sum_{j=0}^{n_{k-1}} W_{ij}^k \, O_j^{k-1}(t) \quad i = 1,2,\dots n_k \quad ; \quad k = 1,2,\dots L \qquad (5\text{-}3)$$

Avec :

- $O_0^k(t) = 1 \qquad k = 0,1,2,\dots L$ (5-4)
- $O_i^0(t) = X_i(t) \qquad i = 1,2,\dots m$ (5-5)
- $Y_i(t) = O_i^L(t) \qquad i = 1,2,\dots ni$ (5-6)
- $O_i^k(t) = f^k\left[S_i^k(t)\right] = f^k\left(\sum_{j=0}^{n_{k-1}} W_{ij}^k \, O_j^{k-1}(t)\right)$ (5-7)

Avec pour la couche k :

- $f^k(.)$: est la fonction d'activation.
- n_{k-1} : est le nombre de neurones.
- $O_i^k(t)$: est la sortie du neurone i.
- W_{ij}^k : est le coefficient synaptique de la connexion entre le neurone *i* de la couche *k* et le neurone *j* de la couche précédente (*k*-1).
- W_{i0}^k : est le seuil adaptable du neurone *i*.
- $X_i(t)$ Et $Y_i(t)$ sont les $i^{ème}$ composantes du vecteur d'entrée *x* (t) et du vecteur de sortie *y*(t) respectivement**.**
- La fonction d'activation généralement choisie est la fonction sigmoïde :

$$F(x) = \frac{1-e^{-x}}{1+e^{-x}} \qquad (5\text{-}8)$$

5.3.5 . Principe [41]

L'objectif de la méthode de rétro-propagation est d'adapter les Paramètres W_{ij}^k de façon à minimiser la valeur moyenne de l'erreur sur l'ensemble d'entraînement. La fonction coût la plus utilisée est donnée par :

$$E = \frac{1}{2}\sum_{t=1}^{T} E(t) = \frac{1}{2}\sum_{t=1}^{T}\left[Y^d(t) - Y(t)\right]^2 \qquad (5\text{-}9)$$

Où :

- Y^d est le vecteur de sortie désirée.
- Y est le vecteur de sortie du réseau.
- T est la longueur de l'ensemble d'apprentissage.

Cependant, dans quelques situations d'autres critères d'erreur peuvent être plus appropriés. L'approche la plus utilisée pour la minimisation de la fonction E est basée sur les méthodes de gradient. On commence l'entraînement par un choix aléatoire des valeurs initiales des poids. On présente le premier vecteur d'entrée.

Une fois la sortie du réseau obtenue, l'erreur correspondante et le gradient de l'erreur par rapport à tous les poids est calculée, les paramètres sont ajustés dans la direction opposée à celle du gradient de l'erreur. On refait la même procédure pour tous les exemples d'apprentissage. Ce processus est répété jusqu'à ce que les sorties du réseau soient suffisamment proches des sorties désirées.

5.3.6 . Adaptation des poids [41]

Pour un ensemble de poids donné, il est facile de calculer la sortie $Y(t)$ et l'erreur $E(t)$ correspondant à une entrée X (t), en utilisant les équations (5-3) et (5-9). Les paramètres du réseau sont alors ajustés par la méthode de gradient en utilisant la formule itérative :

- $W_{ij}^{k}(n) = W_{ij}^{k}(n-1) + \Delta W_{ij}^{k}(n)$ (5-10)
- $\Delta W_{ij}^{k}(n) = -\mu \frac{\delta E}{\delta W_{ij}^{k}(n)}$ (5-11)

- μ est le facteur ou pas d'apprentissage.
- n est le numéro de l'itération.

La vitesse de convergence dépend de la constante μ. Sa valeur est généralement choisie expérimentalement. Si μ est trop petit la convergence est lente mais la direction de descente est optimale. Si μ est trop grand la convergence est rapide mais la précision est médiocre, un phénomène d'oscillation intervient dès qu'on approche du minimum. La dérivée de l'erreur E par rapport au poids $W_{ij}^{k}(n)$ est donnée par :

$$\frac{\delta E}{\delta W_{ij}^{k}(n)} = \sum_{t=1}^{T} \frac{\delta E(t)}{\delta W_{ij}^{k}(n)} \quad (5\text{-}12)$$

Avec :

$$\frac{\delta E(t)}{\delta W_{ij}^{k}(n)} = -\delta_{i}^{k}(t) O_{i}^{k-1}(t) \quad (5\text{-}13)$$

Où $\delta_{i}^{k}(t)$ est l'erreur équivalente à la sortie du neurone i de la couche k.

Pour les neurones des couches de sortie :

$$\delta_{i}^{1}(t) = f^{1\prime}[S_{i}^{1}(t)]\left[Y_{i}^{d}(t) - Y_{i}(t)\right]^{2} \quad (5\text{-}14)$$

Pour les neurones des couches cachées :

$$\delta_{i}^{k}(t) = f^{k\prime}\left[S_{i}^{k}(t)\right] \sum_{j=1}^{n_{k+1}} \delta_{j}^{k+1}(t)\, W_{ij}^{k+1}(n) \quad (5\text{-}15)$$

Pour minimiser l'erreur totale E sur l'ensemble d'entraînement, les poids du réseau doivent être ajustés après la représentation de tous les exemples. Cependant on peut ajuster les poids après la représentation de chaque exemple, les corrections sont assez faibles et la minimisation de $E(t)$ est une bonne approximation de la minimisation de E, l'équation (5-11) est remplacée par :

$$\Delta W_{ij}^{k}(n) = -\mu \frac{\delta E(t)}{\delta W_{ij}^{k}(n)} \qquad (5\text{-}16)$$

Alors :

$$W_{ij}^{k}(n+1) = W_{ij}^{k}(n) + \mu\, \delta_{i}^{k} O_{i}^{k-1} \qquad (5\text{-}17)$$

5.3.7 . Algorithme [41]

Les différentes étapes de l'algorithme de rétro-propagation sont les suivantes :

- Etape 1**:** Initialiser les poids W_{ij} et les biais des neurones à des petites valeurs aléatoires.
- Etape 2**:** Présenter le vecteur d'entrée et de sortie désirés correspondants.
- Etape 3**:** Calculer :
  - La somme des entrées des neurones d'une couche cachée (équation (5.3)).
  - Les sorties des neurones de la couche cachée (équation (5.7)).
  - La somme des entrées de la couche de sortie (équation (5.3)).
  - Les sorties du réseau (équation (5.7)).
- Etape 4**:** calculer
  - Les termes de l'erreur pour les neurones de la couche de sortie (équation (5.14))
  - Les termes de l'erreur pour les neurones de la couche cachée (équation (5.15))
- Etape 5 **:** Ajuster les poids de la couche de sortie et la couche cachée (équation (5.17))
- Etape 6 **:** Si la condition sur l'erreur ou sur le nombre d'itération est atteinte, aller à l'étape 7, sinon revenir à l'étape 3 jusqu'à la stabilisation du système.
- Etape 7 **:** Fin.

5.3.8 . Le minimum local

Si le modèle est linéaire par rapport à ses paramètres, la fonction de coût des moindres carrés est quadratique par rapport à ceux-ci : elle ne présente qu'un minimum. Si le modèle n'est pas linéaire par rapport aux paramètres, la fonction de coût présente plusieurs minimums, parmi lesquels il convient de faire un choix [44]. La figure (5.8) donne une explication du phénomène des minimums locaux.

On cite les méthodes à suivre afin d'éviter ces derniers:

- Modifier le pas d'apprentissage du réseau pour pousser le réseau hors des minimums locaux.
- Réduire les poids du réseau par une petite quantité à chaque pas d'apprentissage.

L'activation d'un neurone sature quand ses liens possèdent de trop grands poids synaptiques. Ce problème est difficile à résoudre lors de l'apprentissage, car les valeurs d'activation extrêmes se traduisent souvent au niveau de la rétro-propagation par de petites corrections. Le réseau se trouve alors dans un minimum local. Réduire systématiquement tous les poids par une petite valeur à chaque correction contribue à garder les petits poids synaptiques, pour résoudre ce problème on peut:

- Relancer l'apprentissage plusieurs fois en utilisant des poids initiaux différents, ce qui entraîne un temps de calcul plus élevé.
- Ajouter une dose de bruit aléatoire aux poids du réseau, puis relancer l'apprentissage, dans le but de " déloger " le réseau hors de bassins peu profonds [41].

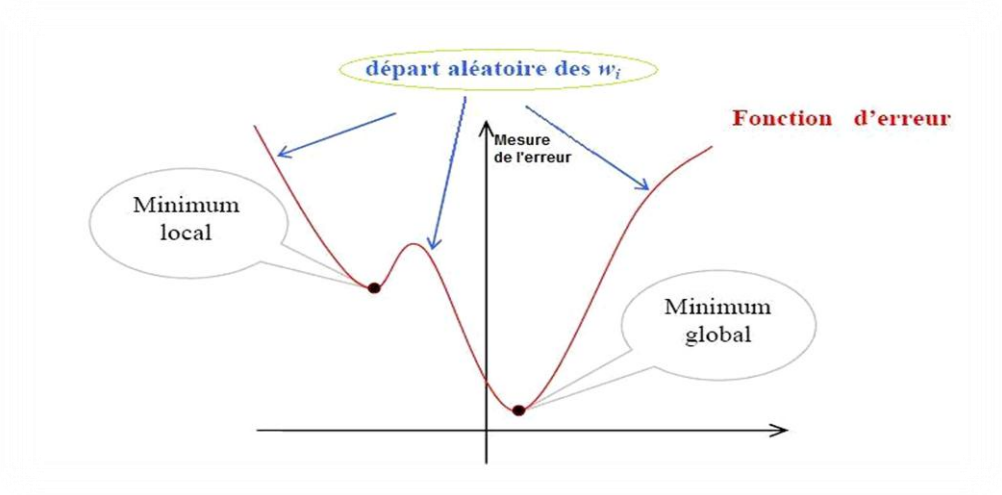


Figure 5.8. *Courbe explicative du phénomène du minimum local [41].*

5.4 . Commande par réseaux de neurones [45] [46] [47] [48]

Par leur capacité d'approximation universelle, les réseaux de neurones sont bien adaptés pour la commande des systèmes. En effet dans ce cas la fonction commande est une fonction non linéaire, l'objectif est alors d'approximer cette fonction par les réseaux de neurones artificiels.

Cette approximation est réalisée par apprentissage des poids du réseau, l'apprentissage peut s'effectuer selon deux types principaux :

- Apprentissage indirecte hors ligne : Dans ce cas, l'apprentissage est basé sur un ensemble de données définissant la fonction commande.

- Apprentissage en temps réel en ligne : Dans ce cas, la mise à jour des poids est essentiellement adaptative.

Il existe alors plusieurs algorithmes et stratégie de commande basés sur ces deux structures.

5.4.1 . Commande par Apprentissage supervisé

A. Commande basé sur un contrôleur conventionnel

Il est possible d'apprendre au réseau de neurones les actions correctes en utilisant un système de commande qui existe déjà, mais pourquoi faire copier un contrôleur qui existe déjà ? La plupart des contrôleurs traditionnels sont basés autour d'un point d'opération. Ceci signifie que le système de commande peut opérer correctement autour d'un point défini. Ces systèmes de commande sont défaillants en cas de présence d'incertitude ou un changement dans le plan inconnu. L'avantage de la commande par réseau de neurones c'est qu'en cas de présence de perturbation dans le plan inconnu, le réseau de neurones est capable d'adapter les paramètres de ses poids et maintenir la commande.

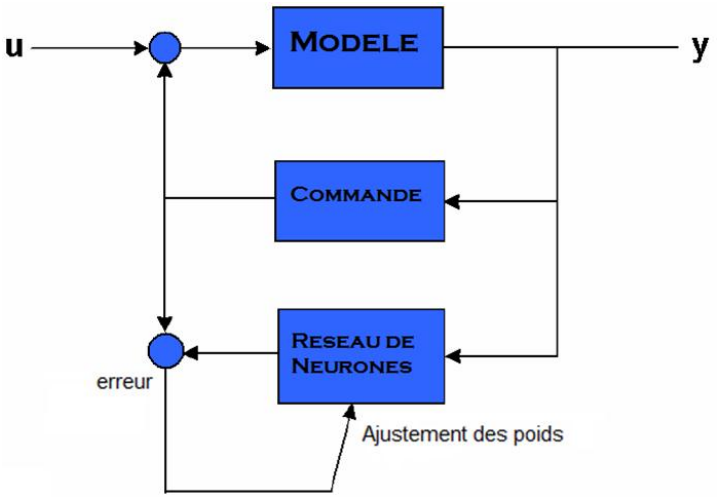


Figure 5.9. *Identification d'un contrôleur conventionnel.*

Un réseau de neurones peut par exemple reproduire le comportement d'un contrôleur conventionnel déjà existant (PI, PID, RST, ...) grâce à ses facultés d'apprentissage et d'approximation. Il suffit de le soumettre à un apprentissage hors ligne pendant une phase d'identification directe en considérant que le contrôleur est lui-même un processus. La figure (5.9) montre le principe de l'identification directe d'un contrôleur conventionnel.

Le but de cette architecture n'est pas seulement de perfectionner les performances du contrôleur conventionnel déjà existant, mais de s'affranchir des contraintes d'implémentations matérielles que peuvent nécessiter certains régulateurs.

La méthode de régulation de type RST par exemple est reconnue pour ses bonnes performances en commande mais elle pose de sérieux problèmes en intégration numérique.

B. Commande inverse

L'avantage de l'utilisation de la commande inverse c'est qu'elle ne nécessite pas la présence d'un système de commande déjà existant lors de l'apprentissage. Le réseau de Neurones est entrainé à modéliser le processus inverse, lorsque le système de commande est mis en cascade avec le système réel, les non-linéarités du système de commande annulent les non-linéarités du système réel.

La modélisation inverse est utilisée pour générer l'inverse du système réel. La sortie est utilisée comme entrée pour le réseau de neurones, cette méthode d'entrainement va forcer le Réseau à représenter l'inverse du système.

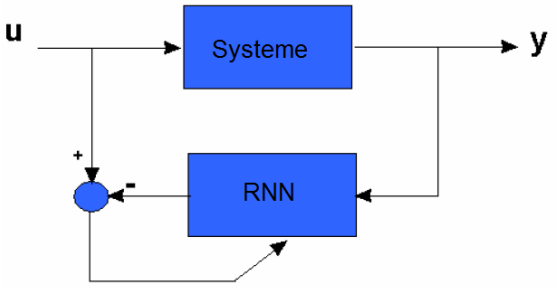


Figure 5.10. *Modélisation inverse du Système.*

C. Commande basée sur l'erreur de sortie

Dans la commande par modèle de référence, la sortie désirée en boucle fermée est spécifiée à travers un modèle de référence stable. Le système de commande tend à rendre la sortie du système similaire à la sortie du modèle de référence.

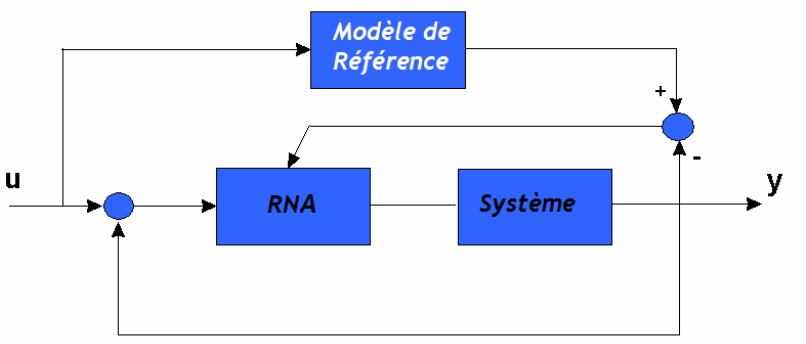


Figure 5.11. *Commande à modèle de référence.*

D. Commande hybride adaptative

Dans cette stratégie de contrôle, les réseaux de neurones sont introduits pour construire des systèmes de commande adaptative stable basés sur la théorie de la stabilité de Lyapunov. L'idée est que les réseaux de neurones sont utilisés pour approximer les non linéarités du système non linéaire en utilisant une loi de commande non linéaire.

Une des méthodes qui a connu le plus de développement est basée sur la méthode de linéarisation par retour (feedback linéarisation). Lorsque les fonctions non linéaires f(x) et g(x) sont inconnues, deux réseaux de neurones sont utilisés pour obtenir leurs approximations. Ce type de commande récemment utilise beaucoup plus des réseaux de neurones de type RBF avec différentes lois de commande inspirées des méthodes qui se basent sur les non linéarités des systèmes tell que le Sliding mode dans les travaux effectué en 2010 et 2011 par WangWu Xuchang et xing [49]

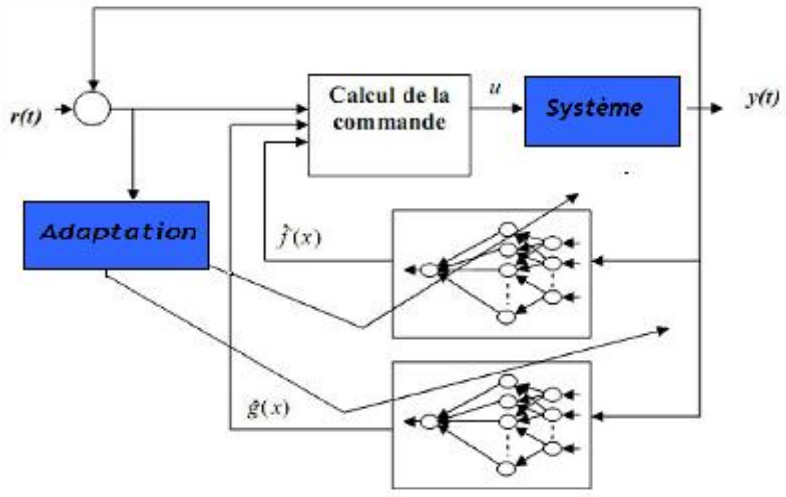


Figure 5.12. *Principe d'identification des fonctions non linéaire.*

5.4.2 Commande par Apprentissage non supervisé [46]

Les types de systèmes de commande précédents sont tous entrainé en utilisant une pré-connaissance telle qu'un contrôleur déjà existant qui fournit les actions correctes. Dans ce type d'apprentissage la connaissance de la sortie désirée n'est pas nécessaire, et la procédure d'apprentissage est basée uniquement sur les valeurs d'entrées. Le réseau s'auto organise de façon à optimiser une certaine fonction de coût, sans qu'on lui fournisse la réponse désirée.

Cette propriété est appelée auto-organisation, ce type de commande représente le futur de la commande des systèmes par les réseaux de neurones.

5.5 Application à la commande du pendule inversé par apprentissage hors ligne (RNA-hors ligne) [50]

5.5.1 Structure générale du réseau

Le choix de la structure adéquate d'un tel système d'identification à base de réseaux de neurones qu'on envisage de concevoir, est déterminé par la nature des phénomènes à identifier. Beaucoup d'approches et types de réseaux de neurones ont été étudiés, mais il reste le schéma du perceptron multi-couche (MLP) à apprentissage supervisé le schéma le plus simple et le plus populaire.

5.5.2 Apprentissage du réseau

L'entrainement de notre réseau nécessite une préparation de l'espace des exemples, ces exemples représentent les actions correctes que doit le réseau apprendre. Chaque couple d'exemple est composé d'un ensemble d'entré sortie enregistré lors de la stabilisation du pendule par un autre contrôleur déjà existant.

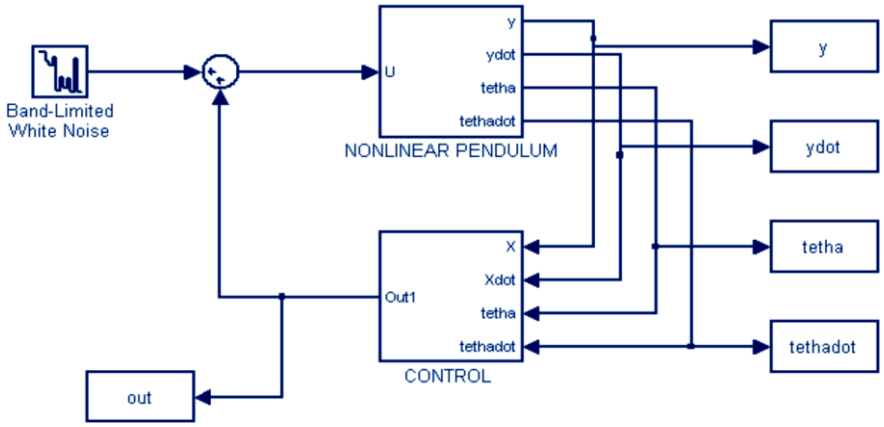


Figure 5.13. *Exemple de préparation sur Simulink des données d'apprentissage* [45].

5.5.3 Configuration optimale

La performance du système à concevoir dépend d'une façon directe de sa configuration lors de l'apprentissage. Un réseau de neurones optimum est celui qui est composé d'un nombre minimum de neurones et qui nécessite moins d'information à l'entrée tout en gardant un certain seuil de l'erreur quadratique moyenne. Il est possible de rechercher une bonne configuration en utilisant le principe de discriminance.

5.5.4 Principe de discriminance

Avec ce principe les essais sont effectués en réduisant le nombre de neurones à chaque fois toute en modifiant sa configuration.

5.6 Commande adaptative à apprentissage en temps réel du Mouvement et de la balance du pendule inversé avec frottement non linéaire et perturbations (RNA-en ligne) [18]

Dans cette partie une méthode de commande du mouvement et d'équilibre du pendule inversé est introduite en utilisant un réseau de neurones artificiel. La stratégie de contrôle utilise une stratégie de compromis pour réaliser le suivi de mouvement et le contrôle de l'équilibre simultanément avec un seul contrôleur. Contrairement à d'autres stratégies de commandes, pas besoin d'apprentissage hors ligne et la connaissance priori des dynamiques du système n'est pas nécessaire.

la commande se base sur la capacité d'apprentissage et de généralisation des réseaux de neurones pour apprendre en ligne les dynamiques du pendule inversé et atteindre la robustesse aux incertitudes non structurées sous la forme de frottement non linéaire et la perturbation externe. Par ailleurs, le parallélisme inhérent des réseaux de neurones en fait un bon candidat pour la mise en œuvre et l'implémentation en temps réel.

5.6.1 Modélisation du frottement non linéaire

La complexité du Modèle du pendule inversé étudié dans le chapitre 2 est encore augmentée par l'adoption d'une modélisation d'un frottement fortement non-linéaire. Ce modèle comporte le frottement Coulombien, visqueux, et des termes de frottement statique. Le modèle d'une telle exploitation du frottement le long d'un déplacement est décrit par :

$$\tau_F = F_c\, sign\,(\dot{\sigma}) + F_v\dot{\sigma} + F_s\, sign\,(\dot{\sigma})e^{-(\frac{\dot{\sigma}}{\eta_s})^2} \qquad (5\text{-}16)$$

D'où F_c, F_v et F_s sont les coefficients de coulomb, de viscosité et de frottement statique respectivement, η_s est le taux de décroissance du frottement statique.

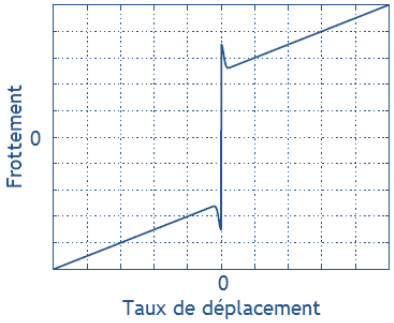


Figure 5.14. *Modèle de frottement de Stribeck.*

Remarque : Il est important de souligner qu'un tel modèle de frottement provoque une augmentation des complexités non linéaires du système, il permet généralement une représentation plus précise de la dynamique du système.

5.6.2 Techniques du Soft Computing

Dans la vraie vie, les non-linéarités élevées dans un système rend la dérivation d'un modèle mathématique précis une tâche difficile. Puisque les techniques de commande conventionnelles dépendent fortement des modèles précis, ils ne parviennent pas à réaliser une performance satisfaisante dans la présence d'incertitudes de modélisation.

Cependant, les techniques de commande basées sur le soft computing n'ont pas de telles limitations et, par conséquent, leur performance dépasse celle des approches classiques en raison de leur indépendance des modélisations mathématiques.

En fait, les outils d'intelligence artificielle ont été utilisés dans un grand nombre de domaines là où on n'a pas un modèle mathématique précis et ont fourni des résultats satisfaisants en présence de signaux de fortes magnitudes de bruit, et changeant dynamiquement les paramètres.

Puisque l'outil qu'on va utiliser est un réseau de neurones artificiel de type MLP qui est la structure la plus utilisée, on peut décrire la relation entre les entrées et les sorties du réseau de neurones par :

$$v_j^{(k)}(t) = \sum_{i=1}^{m} w_{ji}^{(k)}(t) x_i^{(k)}(t) \qquad (5\text{-}17)$$

$$O_j^{(k)}(t) = \varphi_j^{(k)}\left(v_j^k(t)\right) \qquad (5\text{-}18)$$

D'où $\varphi_j^{(k)}$ est la fonction d'activation du j^{eme} nœud de la couche k, m est le nombre de nœud d'entrée, $x_i^{(k)}(t)$ et $O_j^{(k)}(t)$ sont respectivement, l'entrée du nœud i et la sortie du nœud j de la couche k à l'instant t. v_j est l'entrée du réseau du j^{eme} nœud. $w_{ji}^{(k)}$: est le poids qui lie le nœud i de la couche $(k-1)$ au j^{eme} nœud de la couche k. Alors la sortie du réseau est décrit par :

$$y(t) = \sum_{j=1}^{ncc} w_{1j}^{(2)}(t)\, O_j^{(1)}(t) \qquad (5\text{-}19)$$

$$y(t) = \sum_{j=1}^{ncc} w_{1j}^{(2)}(t)\ \varphi\left(\sum_{i=1}^{m} w_{ji}^{(1)}(t)\ x_i(t)\right) \qquad (5\text{-}20)$$

D'où ncc est le nombre de couches cachées. Un algorithme de rétro propagation intégré permet la correction des poids en temps réel (en ligne).

5.6.3 . Stratégie de commande

Dans cette section, un réseau de neurones est conçu pour commander le mouvement le long de l'axe x et l'équilibre du pendule avec dynamique inconnue (Figure 5.15) :

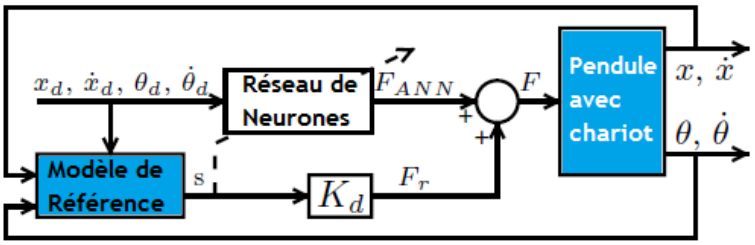


Figure 5.15. *Diagramme bloc de la stratégie de commande.*

Etant donné :

- x: La position du chariot.
- x_d: La vitesse linéaire désirée du chariot.
- θ: L'angle du pendule.
- θ_d: L'angle désiré du pendule.
- $e_x = x - x_d$ et $\dot{e}_x = \dot{x} - \dot{x}_d$ les erreurs de la position linéaire et la vitesse linéaire du chariot, respectivement.
- $e_\theta = \theta - \theta_d$ et $\dot{e}_\theta = \dot{\theta} - \dot{\theta}_d$ représente les erreurs de l'angle de balancement et la vitesse angulaire du pendule, respectivement.

L'objectif de la commande c'est de ramener les erreurs $e_x, \dot{e}_x$, e_θ et $\dot{e}_\theta$ vers zéro. Pour cela on définit le modèle de référence de l'erreur comme suit :

- $S_x = \dot{e}_x + \psi_x e_x$ (5-21)
- $S_\theta = \dot{e}_\theta + \psi_\theta e_\theta$ (5-22)

D'où ψ_x et ψ_θ sont des gains constants et positifs.

Puisque le système en question est un système sous actionné, le contrôleur a besoin de ramener les deux signaux d'erreurs vers zéro en utilisant un seul signal d'entrée.

Il n'est pas considérable si la commande de la position et de l'angle du pendule n'est pas atteinte en même temps, sauf qu'une stratégie d'équilibre entre les erreurs S_x et S_θ est indispensable. Pour cela une erreur composée est définie par :

$$S = (1 - \lambda)S_x + \lambda S_\theta \qquad (5\text{-}23)$$

D'où λ est le coefficient d'équilibre entre la commande de position et de l'angle.

Etant donné Les signaux désirés x_d, $\dot{x}_d$, θ_d *et* $\dot{\theta}_d$, le Réseau de Neurones minimise l'erreur S pour atteindre la commande du chariot et du pendule simultanément.

À cause de la nature itérative du mécanisme d'apprentissage des réseaux de neurones et à cause du grand ordre de complexité du modèle dynamique du système, le réseau de neurones peut prendre relativement une grande période de temps pour converger, ce qui mène à une dynamique instable ou à une performance insatisfaisante. Pour cela le coefficient de robustesse F_r correspondant à un régulateur PD est introduit :

$$F_r = K_d s \qquad (5\text{-}24)$$

Avec K_d est un gain constant, la loi de commande devient :

$$F = F_{RNN} + F_r \qquad (5\text{-}25)$$

Le réseau de neurone se compose de 3 couches, une couche d'entrée avec 2 neurones, une couche cachée avec 6 neurones et un neurone pour la couche de sortie. La fonction sigmoïde est utilisée comme fonction d'activation de tous les neurones sauf le neurone de sortie ou on utilise la fonction purement linéaire.

5.6.4 . Simulation numérique

Pour démontrer les performances du système de commande proposé, un ensemble de simulations numériques sont exécutées et effectuées sur le pendule inversé modélisé au chapitre2.

Les gains des différents coefficients sont définis comme suit :

ψ_x=-1751, ψ_θ=144, K_d =1, λ=0.9,F_{cx} = 5.10e^{-3}, $F_{c\theta}$ = 3.10e^{-3},F_{vx}= 2.10e^{-3}, $F_{v\theta}$= 1.10e^{-3}
F_{sx}= 3.10e^{-3}, $F_{s\theta}$= 2.10e^{-3}, η_{sx}= 5.10e^{-3}, $\eta_{s\theta}$=4.10e^{-3}.

Les signaux de référence de la position et la vitesse désirées sont pris comme la réponse indicielle d'une rampe critique du 2eme ordre avec une fréquence naturelle de 1 rad/s comme le montre les figures suivantes :

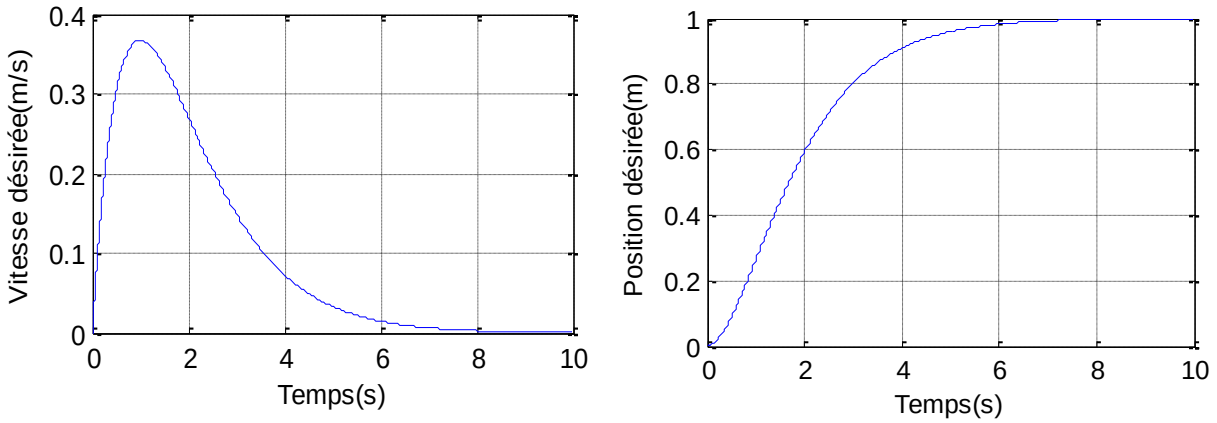


Figure 5.16. *Signaux de références de la position et l'angle désirés.*

D'un autre coté les signaux de référence de l'angle et la vitesse angulaire désirée sont pris à zéro (θ_d = $\dot{\theta}_d$=*0).*

5.6.5 . Résultats

La simulation est divisée en trois étapes, la première étape comporte la simulation sans l'introduction du contrôleur neuronal, dans ce cas-là la stabilisation et la commande de la position et du balancement sont dues au coefficient d'amortissement F_r .

Dans la 2eme étape le contrôleur neuronal est introduit pour évaluer sa capacité de mieux compenser les non linéarités du système, dans la 3eme étape un échelon de force externe est introduit à t = 5s afin de montrer la capacité du système de commande a rejeter et compenser les perturbations extérieures.

Les figures suivantes regroupent les résultats obtenus dans les 3 cas :

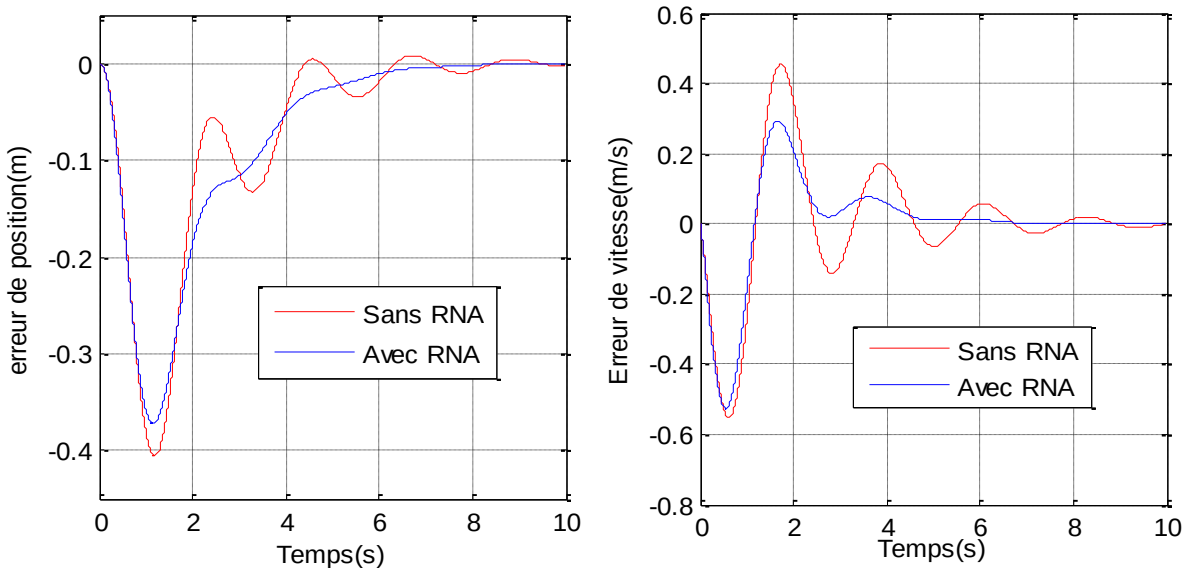


Figure 5.17. *L'erreur de position et de vitesse linéaire simulées sans et avec la commande neuronale (RNA).*

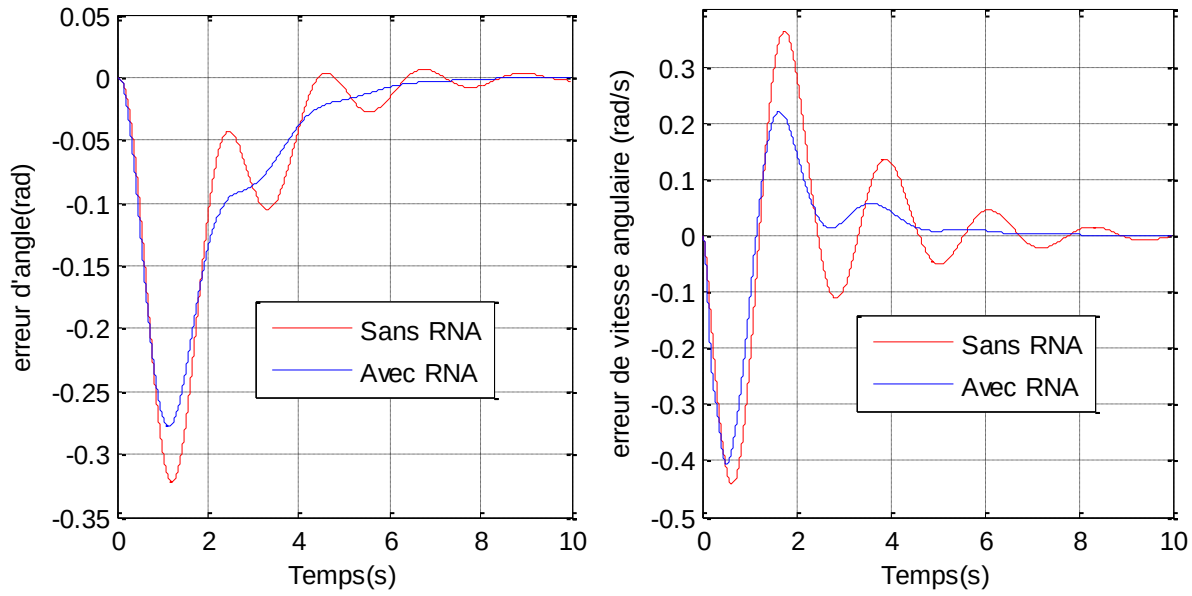


Figure 5.18. *Erreur d'angle et de vitesse angulaire simulées sans et avec la commande neuronale(RNA).*

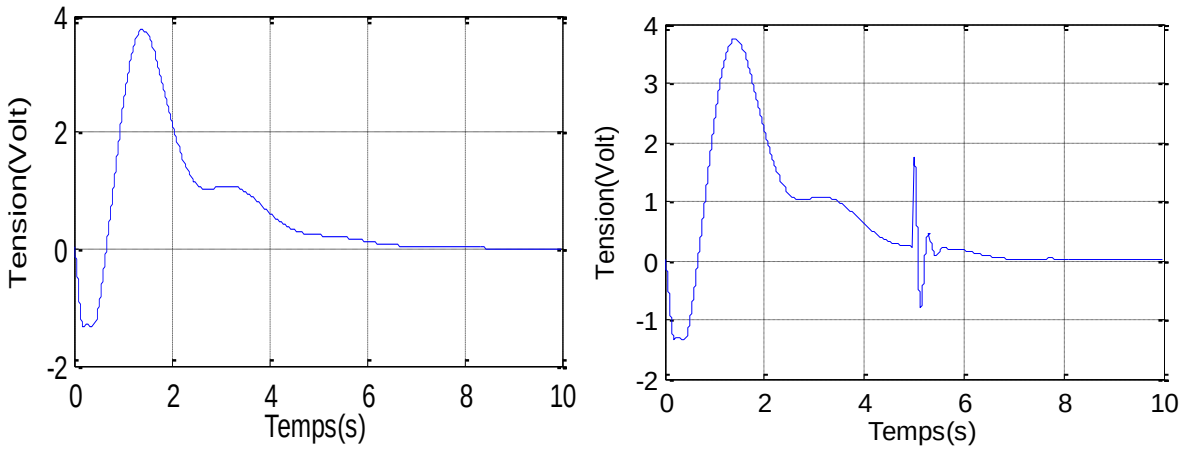


Figure 5.18. *Commande globale sans et avec perturbation.*

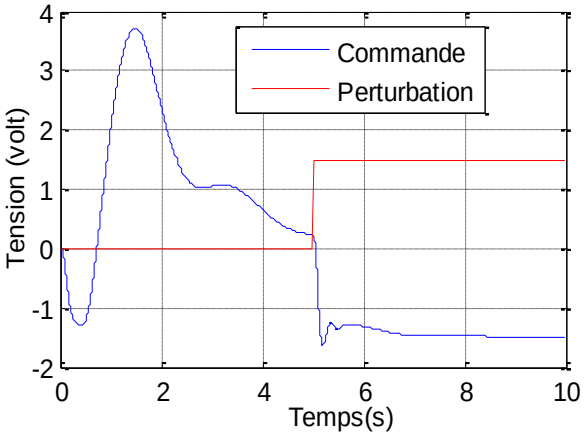


Figure 5.19. *Commande du contrôleur neuronal avec perturbation extérieure.*

La simulation sans le contrôleur neuronal montre la stabilisation du pendule par le coefficient d'amortissement ainsi que l'atténuation des erreurs de position et d'angle. Cependant des oscillations souples tout au long des signaux d'erreur sont présentes. D'un autre côté lorsqu'on introduit la commande neuronale, on constate clairement sur les figures (5.18) et (5.19) une convergence plus douce d'une manière plus régulière vers zéro avec moins d'effort de commande.

Bien qu'on note que la perturbation externe introduite n'est pas modélisée dans la conception du dispositif de commande. En général, la perturbation externe peut significativement affecter la précision du système de positionnement et provoque d'inacceptables oscillations à haute fréquence.

La figure (5.19) montre clairement la compensation de la perturbation par la commande neuronale et le système de commande réussit à faire face aux changements de force inattendue, la figure (5.18) montre que la commande globale reste inchangée malgré la présence permanente de la perturbation. Par conséquent, les erreurs de mouvement et d'équilibre restent au même niveau.

En conclusion, Les résultats de simulation pour les différentes situations soulignent la performance du système de commande.

5.7 . Avantages et inconvénient [50]

- Les réseaux de neurones sont composés d'éléments fonctionnant en parallèle. Le traitement parallèle permet une vitesse accrue de calcul par rapport à un traitement séquentiel plus lent ainsi qu'ils se caractérisent par leurs comportement non-linéaire et leurs capacité d'optimisation et d'apprentissage.

- Les réseaux de neurones artificiels (RNA) ont une mémoire. La mémoire dans les réseaux de neurones correspond à des coefficients de pondération dans les neurones.

- L'inconvénient réside par le fait du besoin d'un contrôleur déjà existant ou d'un maitre qui fournit les actions correctes, et la nécessité d'exemple dans le cas d'apprentissage hors ligne.

- Le principal inconvénient est qu'ils fonctionnent comme des boîtes noires. Les règles de fonctionnement des réseaux de neurones sont complètement inconnues. Il n'est pas possible de convertir la structure neuronale dans les structures des modèles connus.

- Un autre inconvénient est la quantité de temps prises pour former des réseaux. Cela peut prendre beaucoup de temps pour former un réseau de neurones pour certaines fonctions.

5.8 . Conclusion.

La commande par réseaux de neurones est un outil très puissant pour la commande en temps réel vu ses caractéristiques. On a présenté dans ce chapitre des notions de bases ainsi que les différentes techniques de commande neuronale utilisées dans les systèmes temps réel.

Puis on a étudié l'application de ces techniques sur le pendule inversé. Pour cela, on s'est intéressés au deux principales méthodes suivantes :

- La commande neuronale hors ligne à base d'un contrôleur conventionnel.
- La commande neuronale adaptative à apprentissage en temps réel « en ligne ».

Ces deux techniques font partie de la 3eme catégorie des systèmes de commande qui se basent sur l'intelligence artificielle.

Chapitre 6

Commande par Backstepping

6.1. Introduction

La technique de commande par Backstepping a été développée par X.F.Liu [51] en 1991, et inspirée des travaux de Z. Gao, and P.J. Antsaklis [52] d'une part, et [53] et [54] d'autre part. Cette technique offre une méthode systématique pour effectuer la conception d'un contrôleur pour les systèmes non linéaires, l'idée consiste à calculer une loi de commande afin de garantir pour une certaine fonction (*Lyapunov*) définie positive que sa dérivée soit toujours négative.

L'objectif de cette technique est de calculer, en plusieurs étapes, une commande qui garantit la stabilité globale du système [55]. Contrairement à la plupart des autres méthodes, le backstepping n'a aucune contrainte de non linéarité.

Dans ce chapitre, on présente l'aspect théorique de la commande par backstepping, ensuite un algorithme de commande par backstepping robuste adaptative est proposé et appliqué sur le pendule inversé modélisé dans le chapitre 2.

6.2. Définitions

- **Point d'équilibre :** physiquement, un système est en équilibre quand il conserve son état en absence de forces externes. Mathématiquement, cela équivaut à dire que la dérivée $\dot{x}$ de son état est nulle.

$$\dot{x} = f(x) = 0 \tag{6-1}$$

- **Stabilité :** on dit qu'un système est stable lorsque il revient à son état d'équilibre si l'on excite par une impulsion, il est instable lorsque il s'éloigne.
- **Stabilité de Lyapunov [**55] **:** Considérons le système invariable suivant :

$$\dot{x} = f(x) \tag{6-2}$$

Commençons à l'état initial $x(0)$. Supposons que x_e est un point d'équilibre du système, $f(x_e)=0$. On dit que le point d'équilibre est :

- Stable, si pour chaque $\varepsilon>0$ existe $\delta(\varepsilon)>0$, tel que :

$$\|x(0)-x_e\|<\delta \Rightarrow \|x(t)-x_e\|<\varepsilon, \text{ pour tout } t\geq 0$$

- Asymptotiquement stable, s'il est stable et en plus il existe $r>0$ tels que

$$\|x(0)-x_e\|<r \Rightarrow x(t)\to x_e, \text{ lorsque } t\to\infty$$

- Globalement asymptotiquement stable, s'il est asymptotiquement stable pour tous les états initiaux.

6.3. Méthodes d'analyse de la stabilité des systèmes

L'étude de la stabilité des systèmes non linéaires est très complexe. L'approche de Lyapunov est l'approche la plus utilisée pour étudier ce problème. Cette approche a été introduite au 19^{iéme} siècle par le mathématicien russe ***Alexandre Mikhailovich Lyapunov*** dans son travail intitulé « *The general problem of the motion stability* ».

On distingue deux méthodes de Lyapunov pour l'analyse de la stabilité :

- La méthode de linéarisation,
- La méthode directe.

La première concerne la stabilité locale autour d'un point d'équilibre. Par contre, la deuxième détermine la stabilité des systèmes en construisant une fonction scalaire des états du système dont on examinera la variation temporelle.

On s'intéresse par la méthode directe de Lyapunov car la commande par backstepping est basée sur cette méthode.

A. Méthode direct de Lyapunov

Son objectif, est de définir une méthode permettant d'analyser la stabilité d'un système linéaire sans connaître explicitement les solutions des équations différentielles qui le décrivent.

La philosophie de cette méthode n'est que l'extension mathématique d'un phénomène physique observé, car les systèmes mécaniques et électriques perdent de l'énergie pour se stabiliser au point d'équilibre.

A.1. Théorème de Lyapunov

La méthode directe de Lyapunov (ou la méthode des fonctions de Lyapunov) est dérivée du critère énergétique de stabilité en appliquant ce critère indépendamment du concept d'énergie, on remplace alors l'énergie du système par une « fonction de Lyapunov » qui est définie positive (comme l'énergie). Soit le système autonome :

$$\dot{x} = f(x) \quad , x_e = 0 \tag{6-3}$$

Ce système aura un point d'équilibre $x_e = 0$, globalement asymptotiquement stable, s'il existe une fonction scalaire $V(x)$ continue avec une dérivée partielle par rapport au temps $\dot{V}(x)$ continue ayant les propriétés suivantes :

1. $V(0) = 0$,
2. $V(x) > 0$, $\quad \forall\, x \neq 0$,
3. $\lim_{\|x\| \to \infty} V(x) = \infty$ (radialement non bornée),
4. $\dot{V} < 0 \quad \forall x \neq 0$.

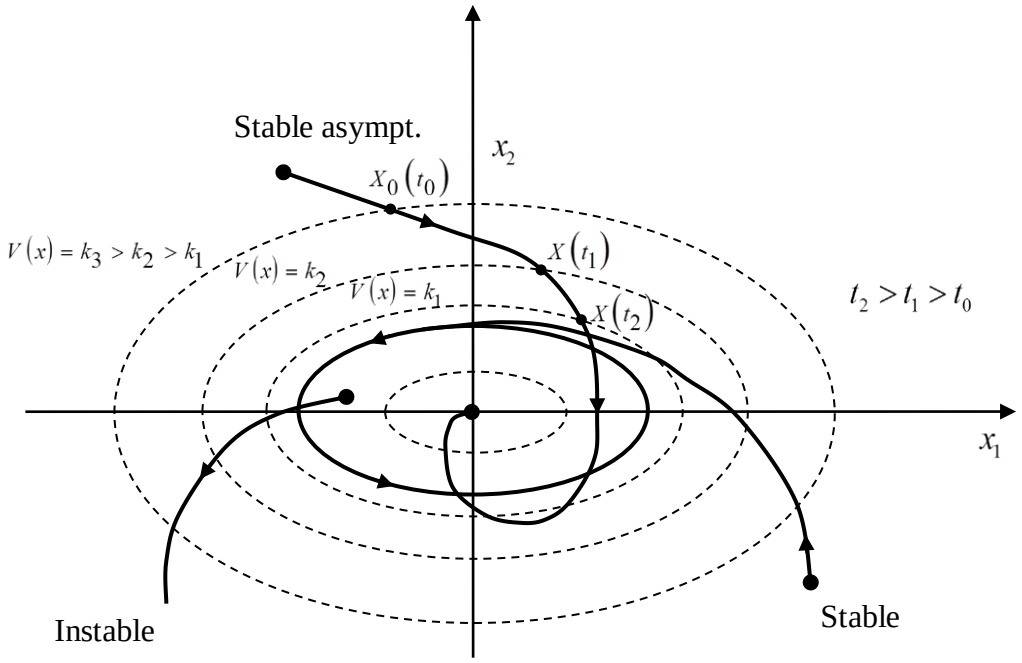


Figure 6.1. *Interprétation géométrique du théorème de Lyapunov.*

B. La synthèse de la commande par la méthode directe de Lyapunov

Dans les paragraphes précédents, on a étudié la stabilité des systèmes où on a supposé implicitement que la loi de commande a été choisie et notre but était de vérifier la stabilité du système avec cette loi de commande, mais le problème dans cette synthèse est comment trouver cette commande qui stabilisera le système.

On va présenter une méthodologie qui combine entre la recherche de la fonction de Lyapunov et la loi stabilisante. En général, il existe deux concepts pour l'application de la méthode directe de Lyapunov pour la synthèse d'une commande stable :

1èr concept : On suppose que la loi de commande existe et on cherche la fonction de Lyapunov.
2éme concept : Cette fois si, on fait un choix sur $V(x)$, la fonction de Lyapunov candidate, et on cherche la loi de commande qui rend cette fonction candidate la fonction de Lyapunov réelle.

Dans la plupart des théorèmes et lemmes de la stabilité au sens de Lyapunov, l'existence de la fonction de Lyapunov était assumée et l'objectif était de déterminer la stabilité de ces systèmes. Mais dans la plupart de cas, la recherche de cette fonction est très difficile. Dans le paragraphe suivant, on présentera la notion du « ***Backstepping*** » qui offre une solution à ce problème.

6.4. Technique de la commande par Backstepping

A. Algorithme de base

Afin d'illustrer le principe de la méthode par Backstepping, on considère le cas des systèmes non linéaires de la forme [55] :

$$\begin{cases} \dot{x}_1 = f_1(x_1) + g_1(x_1)x_2 \\ \dot{x}_2 = f_2(x_1, x_2) + g_2(x_1, x_2)x_3 \\ \dot{x}_3 = f_3(x_1, x_2, x_3) + g_3(x_1, x_2, x_3)\cdot u \end{cases} \tag{6-4}$$

où, g_i et f_i $(i = 1,2,3)$ sont des fonctions non linéaires connues tel que $f_i(0) = 0$ et $g_i(x) \neq 0, \forall (x_1, x_2, x_3) \in \Re^3$. On désire faire suivre à la sortie $y = x_1$ le signal de référence y_r, où y_r, $\dot{y}_r$, $\ddot{y}_r$ et $y_r^{(3)}$ sont supposées connues et uniformément bornées. Le système étant du troisième ordre, le design s'effectue en trois étapes.

Etape 1 :
On considère d'abord le premier sous-système :

$$\dot{x}_1 = f_1(x_1) + g_1(x_1)x_2 \tag{6-5}$$

La variable d'état x_2 est traitée comme une commande et l'on définit la première valeur désirée :

$$\alpha_0 = y_r \tag{6-6}$$

La première variable d'erreur se définit par :

$$e_1 = x_1 - \alpha_0 \tag{6-7}$$

avec ces variables, le système (6-5) s'écrit :

$$\begin{aligned} \dot{e}_1 &= \dot{x}_1 - \dot{\alpha}_0 \\ &= f_1 + g_1 x_2 - \dot{\alpha}_0 \end{aligned} \tag{6-8}$$

pour un tel système, la fonction quadratique,

$$V_1(e_1) = \frac{1}{2} e_1^2 \tag{6-9}$$

Constitue un bon choix. Sa dérivée le long de la solution de (6-8), est donnée par :

$$\dot{V}_1 = e_1 \dot{e}_1 = e_1 \left[f_1 + g_1 x_2 - \dot{\alpha}_0 \right] \tag{6-10}$$

Un choix judicieux de x_2 rendrait $\dot{V}_1$ négative et assurerait la stabilité de l'origine du sous-système décrit par (6-8). Prenons comme valeur de x_2, la fonction α_1, telle que :

$$f_1 + g_1 \alpha_1 - \dot{\alpha}_0 = -k_1 e_1 \tag{6-11}$$

Où $k_1 > 0$ est un paramètre de design. Cela donne :

$$\alpha_1 = \frac{1}{g_1} [-k_1 e_1 - f_1 + \dot{\alpha}_0] \tag{6-12}$$

et la dérivée s'écrit

$$\dot{V}_1 = -k_1 e_1^2 \leq 0 \tag{6-13}$$

D'où la stabilité asymptotique de l'origine de (6-8).

Etape 2 :

On considère, dans ce cas, les deux premiers sous-systèmes :

$$\begin{cases} \dot{x}_1 = f_1(x_1) + g_1(x_1) x_2 \\ \dot{x}_2 = f_2(x_1, x_2) + g_2(x_1, x_2) x_3 \end{cases} \tag{6-14}$$

On définit la nouvelle variable d'erreur :

$$e_2 = x_2 - \alpha_1 \tag{6-15}$$

Les équations du système à commander, dans l'espace (e_1, e_2), s'écrivent :

$$\begin{cases} \dot{e}_1 = f_1 + g_1 (e_2 + \alpha_1) - \dot{\alpha}_0 \\ \dot{e}_2 = f_2 + g_2 x_3 - \dot{\alpha}_1 \end{cases} \tag{6-16}$$

Pour lequel on choisit comme fonction de Lyapunov :

$$V_2(e_1,e_2)=V_1+\frac{1}{2}e_2^2 \tag{6-17}$$

Cette dernière a pour dérivée, le long de la solution de (6-16)

$$\begin{aligned}\dot{V}_2(e_1,e_2)&=\dot{V}_1+e_2\dot{e}_2\\&=e_1\left[f_1+g_1(\alpha_1+e_2)-\dot{\alpha}_0\right]+e_2\left[f_2+g_2x_3-\dot{\alpha}_1\right]\\&=-k_1e_1^2+e_2\left[f_2+g_1e_1+g_2x_3-\dot{\alpha}_1\right]\end{aligned} \tag{6-18}$$

le choix de la valeur désirée (la fonction stabilisante) de x_3 devient évident. Ce dernier est donné par :

$$\alpha_2=\frac{1}{g_2}[\dot{\alpha}_1-g_1e_1-f_2-k_2e_2] \tag{6-19}$$

où $k_2>0$, avec $\dot{\alpha}_1$ calculée analytiquement

$$\dot{\alpha}_1=\frac{\partial\alpha_1}{\partial x_1}\dot{x}_1+\frac{\partial\alpha_1}{\partial y_r}\dot{y}_r+\frac{\partial\alpha_1}{\partial\dot{y}_r}\ddot{y}_r \tag{6-20}$$

un tel choix permet de réduire la dérivée à :

$$\dot{V}_2\le-k_1e_1^2-k_2e_2^2\le 0 \tag{6-21}$$

ce qui assure la stabilité asymptotique de l'origine de (6-16).

Etape 3 :

Le système (6-4) est maintenant considéré dans sa globalité. La variable d'erreur :

$$e_3=x_3-\alpha_2 \tag{6-22}$$

est définie, ce qui permet d'écrire les équations du système, dans l'espace des erreurs (e_1,e_2,e_3):

$$\begin{cases}\dot{e}_1=f_1+g_1(e_2+\alpha_1)-\dot{\alpha}_0\\\dot{e}_2=f_2+g_2(e_3+\alpha_2)-\dot{\alpha}_1\\\dot{e}_3=f_3+g_3u-\dot{\alpha}_2\end{cases} \tag{6-23}$$

On prend comme fonction de Lyapunov :

$$V_3(e_1,e_2,e_3)=V_2+\frac{1}{2}e_3^2 \tag{6-24}$$

La dérivée, le long de la solution de (6-23), devient :

$$\dot{V}_3(e_1,e_2,e_3)=\dot{V}_2+e_3\dot{e}_3=-k_1e_1^2-k_2e_2^2+e_3\left[g_3u+g_2e_2+f_3-\dot{\alpha}_2\right] \tag{6-25}$$

À présent, on est en présence de la vraie commande u. Un bon choix de celle-ci est donné par :

$$u = \frac{1}{g_3}\left[\dot{\alpha}_2 - g_2 e_2 - f_3 - k_3 e_3\right] \tag{6-26}$$

où $k_3 > 0$ et $\dot{\alpha}_2$ est également calculée analytiquement,

$$\dot{\alpha}_2 = \frac{\partial \alpha_2}{\partial x_1}\dot{x}_1 + \frac{\partial \alpha_2}{\partial x_2}\dot{x}_2 + \frac{\partial \alpha_2}{\partial y_r}\dot{y}_r + \frac{\partial \alpha_2}{\partial \dot{y}_r}\ddot{y}_r + \frac{\partial \alpha_2}{\partial \ddot{y}_r}\dddot{y}_r \tag{6-27}$$

avec ce choix, on a :

$$\dot{V}_3(e_1, e_2, e_3) \le -k_1 e_1^2 - k_2 e_2^2 - k_3 e_3^2 \le 0 \tag{6-28}$$

D'où la stabilité asymptotique de l'origine de (6-23). Ceci se traduit par la stabilité, en boucle fermée, du système original (6-4) et la régulation à zéro de l'erreur de poursuite $(y - y_r)$. Les deux principaux objectifs du design sont alors atteints.

6.5. Commande robuste adaptative par backstepping du pendule inversé [56]

L'application de la commande adaptative pour la stabilisation d'un pendule inversé est une proposition tout à fait intéressante par le fait que le pendule inversé sur un chariot peut être classé comme parmi les systèmes dynamiques non linéaires incertains. Toutefois, lorsque la connaissance des paramètres du système est incomplète ou approximative, les erreurs d'approximation s'introduisent dans la boucle de rétroaction qui rend le système difficile à stabiliser ou à maintenir dans l'équilibre stable pour une longue période de temps.

Dans cette partie on va aborder un problème très difficile de concevoir un contrôleur adaptatif pour la commande du pendule inversé sur chariot sans aucune connaissance préalable des paramètres du modèle du pendule inversé. Ce type de problèmes peut être résolu avec l'aide du contrôleur adaptatif robuste. La loi de commande adaptative robuste a été dérivée en deux étapes. Tout d'abord, la loi d'adaptation a été obtenue en utilisant la technique de backstepping pour garder le pendule inversé dans sa position verticale.

L'idée principale de cette technique de backstepping est que l'ensemble du système dynamique est divisée en deux sous-systèmes de la série en cascade, dans lequel les états du premier sous-système sont les variables de commande pour le second.

Dans cette approche, la méthode calcule l'entrée de commande souhaitée pour le deuxième sous-système, dont l'entrée de commande du premier sous-système est calculée de manière à réaliser l'état désiré, qui est l'entrée de commande souhaitée pour le deuxième sous-système.

Peu de méthodes d'adaptation des contrôleurs backstepping pour pendules inversés ont été rapportées dans la littérature. Cependant, elles utilisent des techniques simples de backstepping adaptative pour l'adaptation des paramètres. La principale limitation de ces méthodes de conception, c'est qu'il y a un grand nombre incertain de paramètres de régulation qui doivent être estimés au cours de l'exécution de l'algorithme de commande.

En outre, lorsque ces lois de commande agissent sur le système, ils souffrent d'un très grand taux d'adaptation qui n'est pas du tout souhaitable du point de vue de la bonne performance du système. Cela implique que certaines sortes de mesures de robustesse dans les lois d'adaptation (Qui ont été conçues par la méthode de backstepping) peuvent s'avérer utiles pour atténuer les problèmes qui peuvent survenir en raison du taux élevé d'adaptation. Par conséquent, un régime pour robustesse est ensuite appliqué sur les lois d'adaptation dans la seconde étape, d'où la conception de la loi de commande à l'aide de la fonction de commutation continue.

L'algorithme de commande proposé est conçu à l'aide d'une normalisation en utilisant la technique de commande adaptative par backstepping. Cela permet la réduction du nombre de paramètres de commande incertains, ce qui à son tour offre un algorithme de commande à un coût de calcul inférieur. L'algorithme de commande utilise également une fonction de commutation continue dans l'adaptation pour empêcher un taux de variation brutal de l'adaptation des paramètres. La conception de la loi de commande adaptative robuste pour stabiliser un pendule inversé sur un chariot en utilisant la technique de backstepping n'a jamais été abordée dans la littérature.

On commence par la synthèse de la loi de commande en utilisant la technique de commande adaptative par backstepping pour le maintien du pendule dans sa position verticale incluant la commande simultanée de l'angle et de la position du chariot, et par la suite on présente les résultats de simulation. Différents types de signaux parasites sont également appliquées sur la l'entrée et sur la position angulaire du pendule.

6.5.1. Synthèse de la loi de commande robuste adaptative par backstepping

Dans cette section, on va calculer la loi de commande adaptative robuste par backstepping en deux étapes :

- Nous allons d'abord calculer la loi de commande adaptative par backstepping en tenant compte du critère de stabilité au sens de Lyapunov.
- Ensuite, on va formuler la loi de commande en intégrant des mesures nécessaires pour la robustesse.

6.5.2. Loi de commande adaptative par backstepping :

L'objectif principal de commande du pendule inversé sur chariot est de ramener l'angle θ du pendule vers zéro degrés par rapport à la position verticale. Ici, on va en tirer une loi de commande adaptative par backstepping qui va générer une entrée de commande pour le système de façon à rendre l'angle d'oscillation du pendule très proche à zéro.

Etant donnée les équations généralisées des dynamiques du pendule inversé sur chariot obtenu par la méthode de Lagrange [57] :

$$\begin{cases} (M+m)\ddot{x} + ml\ddot{\theta}cos\theta - ml\dot{\theta}^2 = f \\ (-I+ml^2)\ddot{\theta} + mgl\,sin\theta = ml\dot{\theta}^2\ddot{x} \end{cases} \quad (6\text{-}29)$$

D'où :

- f est l'entré de commande appliquée sur le système.
- m et M sont la masse du pendule et du chariot respectivement.
- l est la longueur de la tige du pendule et g est la constante de gravitation.
- θ et x sont l'angle du pendule et la position du chariot respectivement.
- I est le moment d'inertie.

On peut trouver une relation entre la position angulaire et la commande appliquée sur le chariot du système de pendule inversé en tant que:

$$\Psi_1\, sec\theta\ddot{\theta} + \Psi_2\, tan\theta + \Psi_3\left(\ddot{\theta}\cos\theta - \dot{\theta}^2\sin\theta\right) = f \quad (6\text{-}30)$$

$d'ou$

$$\Psi_1 = (M+m)\left[\frac{(I+ml^2)}{ml}\right] \quad (6\text{-}31)$$

$$\Psi_2 = (M+m)g \quad (6\text{-}32)$$

$$\Psi_{3=ml} \quad (6\text{-}33)$$

Sont les paramètres inconnus du système.

Maintenant, on va introduire les deux variables d'état $z_1 = \theta \ \ et \ \ z_2 = \dot{\theta}$. Ensuite, l'équation (6-30) peut être écrite sous forme d'espace d'état comme indiqué ci-dessous:

$$\dot{z}_{1=} z_2 \qquad (6\text{-}34)$$

$$g(z_1)\dot{z}_2 = f - \Psi_2 tanz_1 + \Psi_3 z_2^2 sin\, z_1 \qquad (6\text{-}35)$$

d′ou $$g(z_1) = -\Psi_1 \sec z_1 + \Psi_3 \cos z_1 \qquad (6\text{-}36)$$

On peut également réécrire l'équation (6-35) sous la forme suivante:

$$g(z_1)\dot{z}_2 = f - K(z_1) \qquad (6\text{-}37)$$

d'où $$K(z_1) = \Psi_2 \tan z_1 - \Psi_3 z_2^2 \sin z_1 \qquad (6\text{-}38)$$

Pour la simplicité et la facilité de calcul, on va introduire une fonction normalisée inconnu h, qui est défini comme $h = K(z_1)/g(z_1)$. Avec l'introduction de cette fonction normalisée h, l'équation (6-37) devient alors:

$$f = g(Z_1)\left(\dot{Z}_2 + h\right) \qquad (6\text{-}39)$$

6.5.3. Commande de la position linéaire du chariot

Pour commander la position linéaire de notre pendule inversé modélisé dans le chapitre 2, nous étions obligés d'utiliser une stratégie d'équilibre pour ramener les erreurs de position et d'angle vers zéro simultanément. Cette stratégie est basée sur une loi d'équilibre linéaire qui force la commande à compenser l'erreur de déplacement du chariot, cela est effectué grâce à un coefficient de compensation, ce coefficient représente l'erreur angulaire nécessaire pour compenser l'erreur de déplacement linéaire du chariot et, ainsi, il est définit comme suit:

$$\theta_x = \lambda(x_{ref} - x) \qquad (6\text{-}40)$$

Où :

- θ_x est le coefficient de compensation angulaire.
- x_{ref} est le signal de référence du déplacement linéaire du chariot.
- λ est une constante positive non nulle.

Maintenant on définit la variable d'erreur e_i comme:

$$e_1 = \theta_{ref} - \theta \qquad (6\text{-}41)$$

D'où $$\theta_{ref} = \alpha_{ref} - \theta_x$$

- α_{ref} est l'angle de référence.
- e_1 représente la première variable d'erreur avec coefficient de compensation angulaire.

Maintenant, notre objectif est de concevoir une loi de commande virtuelle qui ramène e_1 vers 0. Prenons la fonction de Lyapunov suivante :

$$V_1 = \frac{1}{2} e_1^2 \qquad (6\text{-}42)$$

En calculant la dérivée du premier ordre par rapport au temps on obtient :

$$\dot{V}_1 = e_1 \dot{e}_1 = \dot{\theta}_{ref} - \dot{\theta} = \dot{\theta}_{ref} - Z_2 \qquad (6\text{-}43)$$

Maintenant on définit la 2eme variable d'erreur

$$e_2 = z_{ref} - z_2 \qquad (6\text{-}44)$$

Où z_{ref} est une loi de commande virtuelle pour l'équation (6-43). Dans cette étape, l'objectif est de trouver une loi de commande virtuelle appropriée pour l'équation (6-43) qui permet de rendre le système de commande du premier ordre mentionné ci-dessus stabilisable.

Un choix approprié de la loi de commande virtuelle est donnée ci-dessous dans l'équation suivante :

$$Z_{ref} = c_1 e_1 + \dot{\theta}_{ref} \qquad (6\text{-}45)$$

Où c_1 est une constante positive, ce qui garantit la stabilité asymptotique du système. Puis la dérivée par rapport au temps de e_2 devient :

$$\frac{de_2}{dt} = \dot{z}_{ref} - \dot{z}_2 = \left(c_1 \dot{e}_1 + \ddot{\theta}_{ref}\right) - \frac{f}{g} + h \qquad (6\text{-}46)$$

$$= \left(c_1(-c_1 e_1 + e_2) + \ddot{\theta}_{ref}\right) - \frac{f}{g} + h$$

$$= \left(-c_1 e_1^2 + c_1 e_2 + \ddot{\theta}_{ref}\right) - \frac{f}{g} + h$$

Dans l'équation ci-dessus c_1 est une constante positive. Maintenant, d'après l'équation (6-45) nous pouvons trouver l'expression de la force *f*, qui est en fait l'entrée de commande dans notre cas, ce permet d'avoir la dynamique voulue du système exprimée par les équations (6-34) et (6-35). Ainsi,

$$f = \hat{g}(Z_1)\left((1 - c_1^2)e_1 + (c_1 + c_2)e_2 + \ddot{\theta}_{ref} + \hat{h}\right) \qquad (6\text{-}47)$$

Dans l'équation (6-46) les fonctions estimées $\hat{g}$ et $\hat{h}$ représentent les fonctions g et h de l'équation (6-45).

Bien que l'expression de la force soit déterminée par l'équation (6-46), la conception du système de commande est encore incomplète, il reste à déterminer la loi d'adaptation des paramètres. Maintenant, on définit les erreurs de paramètres suivantes qui représentent l'équation (6-48):

$$\bar{g} = g - \hat{g}, \bar{h} = h - \hat{h} \qquad (6\text{-}48)$$

En substituant l'expression de la force f de l'équation (6-46) dans l'expression (6-45), nous obtenons les dynamiques d'erreur suivantes:

$$\frac{de_2}{dt} = -c_2 e_2 - e_1 + \frac{\bar{g}}{g}\left\{(1 - c_1^2)e_{1+} (c_1 + c_2)e_2 + \ddot{\theta}_{ref} + \hat{h}\right\} + \bar{h} \qquad (6\text{-}49)$$

Afin de connaître les lois de mise à jour des paramètres $\hat{g}$ et $\hat{h}$ la fonction de Lyapunov élargie a été formulé pour le système en boucle fermée:

$$V_2 = \frac{1}{2}e_1^2 + \frac{1}{2}e_2^2 + \frac{1}{2\gamma_1 g}\bar{g}^2 + \frac{1}{2\gamma_2}\bar{h}^2 \qquad (6\text{-}50)$$

Ici, la fonction de Lyapunov élargie comprend toutes les variables d'erreur ainsi que les erreurs des paramètres. La dérivée par rapport au temps de V_2 est donnée par:

$$\dot{V}_2 = e_1\dot{e}_1 + e_2\dot{e}_2 + \frac{\bar{g}}{g\gamma_1}\left(-\frac{d\hat{g}}{dt}\right) + \frac{\bar{h}}{h\gamma_2}\left(-\frac{d\hat{h}}{dt}\right) \qquad (6\text{-}51)$$

$$= -c_1 e_1^2 - c_2 e_2^2 + \frac{\bar{g}}{g}\left\{e_2\left((1 - c_1^2)e_1 + (c_1 + c_2)e_2 + \ddot{\theta}_{ref} + \hat{h}\right) - \frac{1}{\gamma_1}\frac{d\hat{g}}{dt}\right\} + \bar{h}\left(e_2 - \frac{1}{\gamma_2}\frac{dh}{dt}\right)$$

De l'équation ci-dessus on peut déterminer la loi des mises à jours des paramètres $\hat{g}$ et $\hat{h}$:

$$\frac{d\hat{g}}{dt} = \gamma_1 e_2\left((1 - c_1^2)e_1 + (c_1 + c_2)e_2 + \ddot{\theta}_{ref} + \hat{h}\right) \qquad (6\text{-}52)$$

$$\frac{d\hat{h}}{dt} = \gamma_2 e_2 \qquad (6\text{-}53)$$

Les lois ci-dessus de mise à jour de paramètre rendent la dérivée de la fonction de Lyapunov élargie. V_2 est définie négative comme ce que montre l'équation ci-dessous :

$$\dot{V}_2 = -c_1\dot{e}_1{}^2 - c_2\dot{e}_2{}^2 \qquad (6\text{-}54)$$

Maintenant, il est évident d'après l'équation (6-54) que l'erreur des dynamiques du système est asymptotiquement stable. Par conséquent, la loi de commande adaptative par backstepping obtenu stabilise le système.

6.5.4. Loi de la commande robuste adaptative par backstepping

Lors de la conception de la loi de commande adaptative par backstepping, tout d'abord, on a supposé que le modèle est exempt de la dynamique non modélisées, la dérive des paramètres, et le bruit. Mais dans les systèmes réels ces hypothèses pourraient ne pas être valides. Lorsque la connaissance des paramètres du pendule inversé sur chariot est incomplète ou approximative, les erreurs d'approximation se glissent dans la boucle de retour qui rend le système difficile à se stabiliser dans l'équilibre stable pendant une longue période [2,9]

En outre, la stabilité des systèmes réels peuvent être gravement affectée par certaines perturbations bornées et le taux élevé d'adaptation. Par conséquent, lors de l'utilisant d'un taux élevé d'adaptation de paramètres, une modification de la loi de commande ci-dessus est nécessaire pour que la stabilité du système global ne s'affecte pas. Ici, on a proposé l'introduction d'une fonction de commutation continue [9] dans la loi d'adaptation des paramètres qui vise à atténuer le problème ci-dessus.

Maintenant, la loi modifiée d'adaptation peut être écrite comme:

$$\dot{\hat{g}} = \gamma_1 e_2 \left((1 - c_1^2)e_1 + (c_1 + c_2)e_2 + \ddot{\theta}_{ref} + \hat{h}\right) - \gamma_1 \sigma_{gs} \hat{g} \qquad (6\text{-}55)$$

$$\dot{\hat{h}} = \gamma_2 e_2 - \gamma_2 \sigma_{hs} \hat{h} \qquad (6\text{-}56)$$

D'où $\sigma_{gs}\, et\, \sigma_{hs}$ sont appelés fonctions continues de commutations et sont représentées par :

$$\sigma_{gs} = \begin{cases} 0 & si\ |\hat{g}| < g_0 \\ \sigma_{g0}\left(\frac{|\hat{g}| - g_0}{|\hat{g}|}\right) & si\ g_0 \leq |\hat{g}| \leq 2g_0 \\ \sigma_{g0} & \end{cases} \qquad (6\text{-}57)$$

$$\sigma_{hs} = \begin{cases} 0 & si\ |\hat{h}| < h_0 \\ \sigma_{h0}\left(\frac{|\hat{h}| - h_0}{|\hat{h}|}\right) & si\ h_0 \leq |\hat{h}| \leq 2h_0 \\ \sigma_{h0} & \end{cases} \qquad (6\text{-}58)$$

Dans les ensembles d'équations (6-56) et (6-57) $g_0, h_0, \gamma_1, \gamma_2, \sigma_{g0} et\, \sigma_{h0}$ sont les constantes de conception. Maintenant, les valeurs des constantes de l'équation (6-56) et (6-57) peuvent être choisies dans des systèmes réels par essais et erreurs. La structure du système de commande a été représentée sur la figure (6.2).

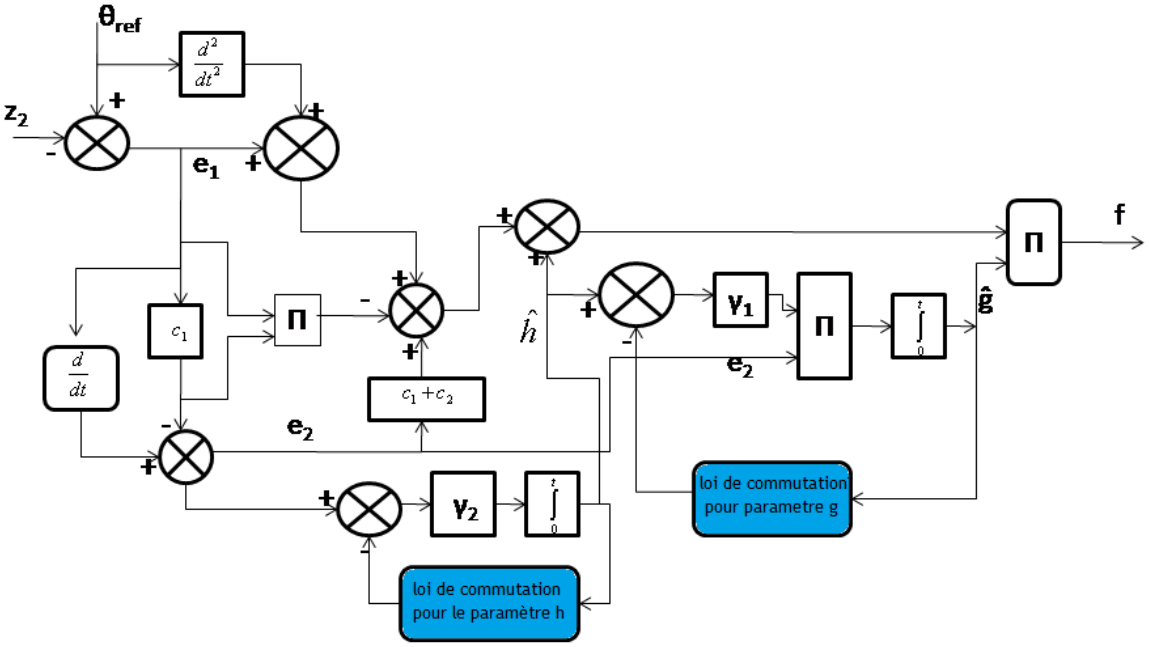


Figure 6.2. *Structure de la commande.*

6.5.5. Résultats de Simulation

Dans cette section, les performances de la loi de commande robuste adaptative par Backstepping ont été vérifiées dans des études de simulation. Pour effectuer cette simulation on a utilisé le modèle mathématique du pendule inversé dérivé dans le chapitre2.

Le modèle du pendule inversé et le modèle du contrôleur sont simulés dans MATLAB ® 2011 et Simulink.

Pendant la simulation, on a choisi $c_1 = c_2 = 6$, $\gamma_1 = 9$, $\gamma_2 = 9$, $g_0 = 300$, $h_0 = 10$ et $\sigma_{g0} = \sigma_{h0} = 1$. Conditions initiales : $\theta = 0,\ x = 0$.

On a choisi un signal carrée de fréquence égal à 0.1 Hz, d'amplitude de 10 cm comme signal de référence pour la commande de position et on a choisi $\alpha_{ref} = 0$ comme angle de référence pour le balancement.

Les résultats obtenus sont illustrés sur les figures suivantes :

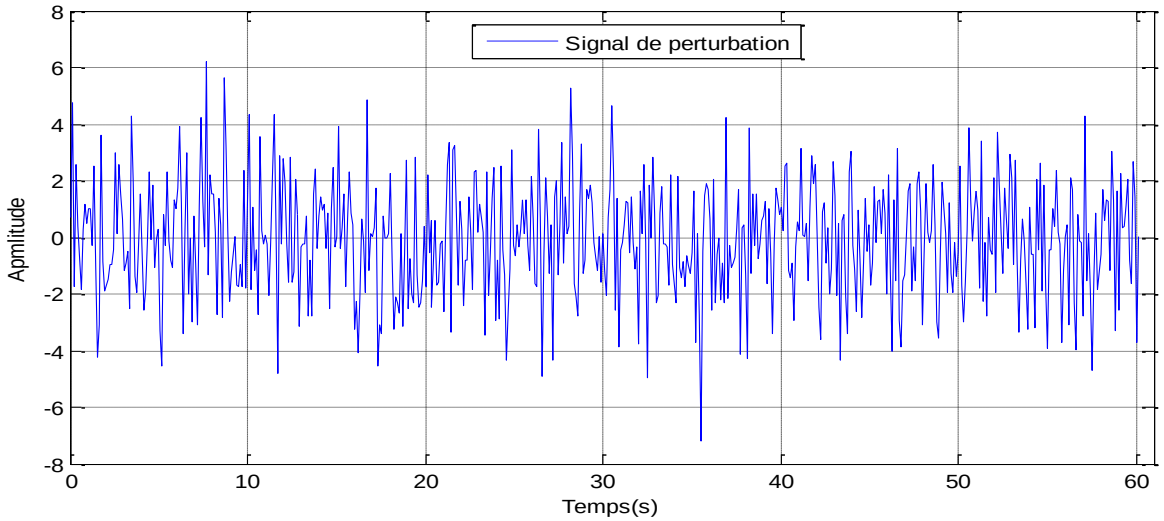


Figure 6.3. *Signal de perturbation.*

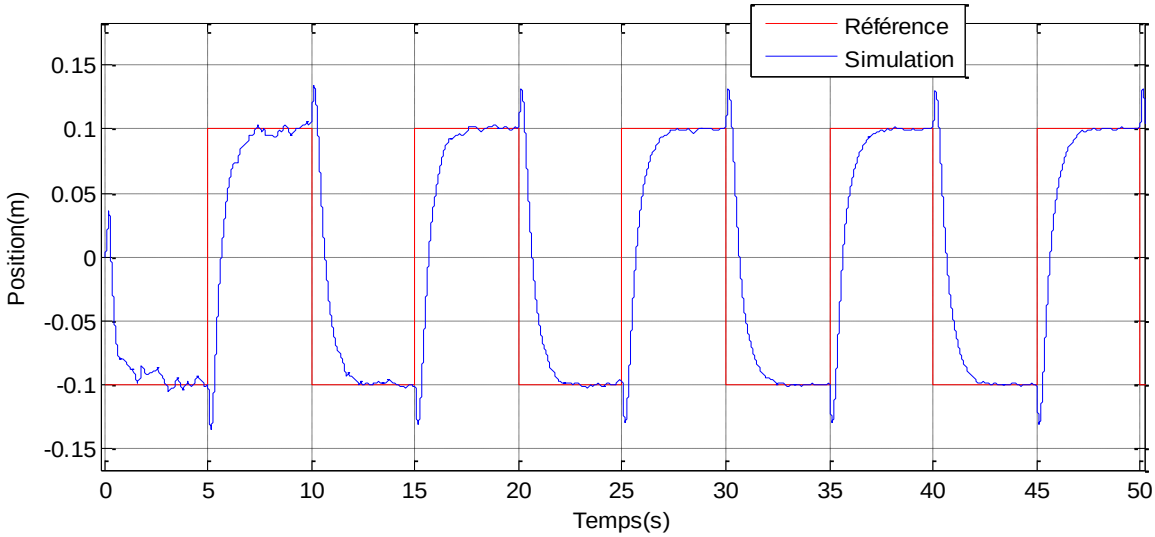


Figure 6.4. *Position linéaire du chariot.*

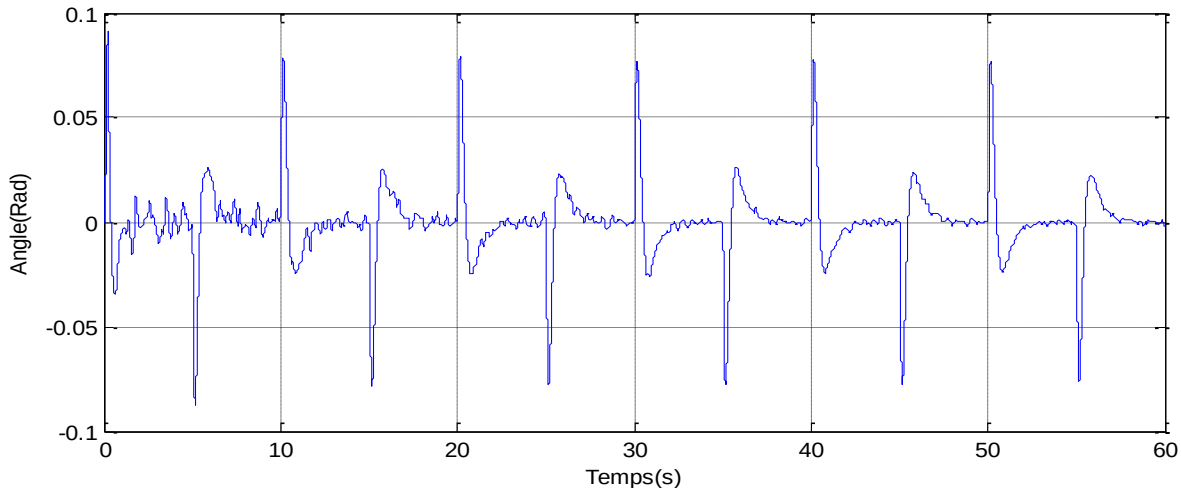


Figure 6.5. *Position angulaire du pendule.*

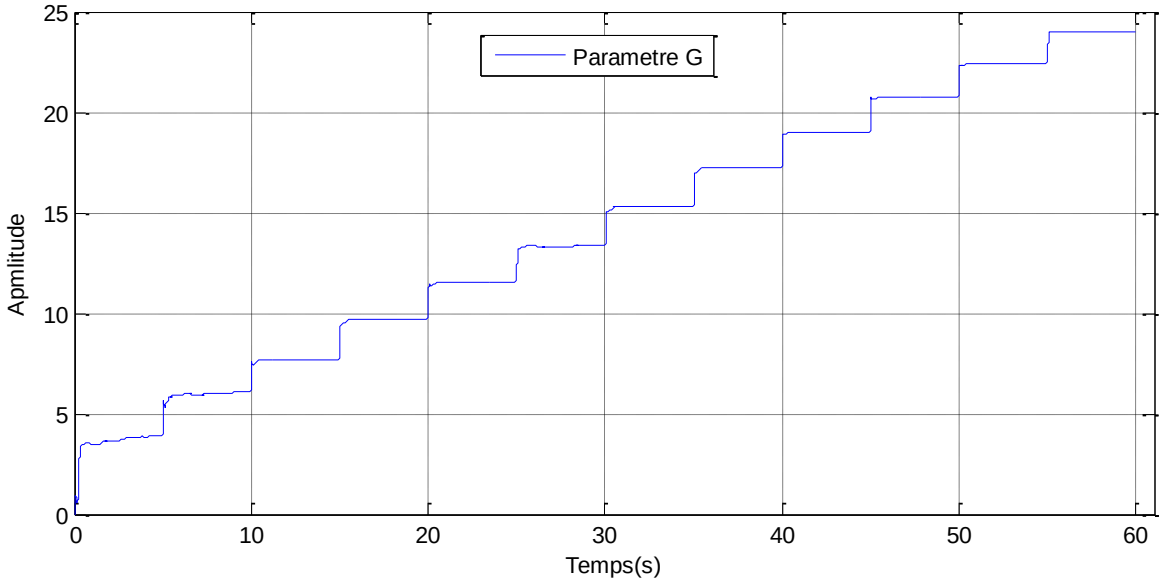


Figure 6.6. *Evolution du paramètre g avec le temps.*

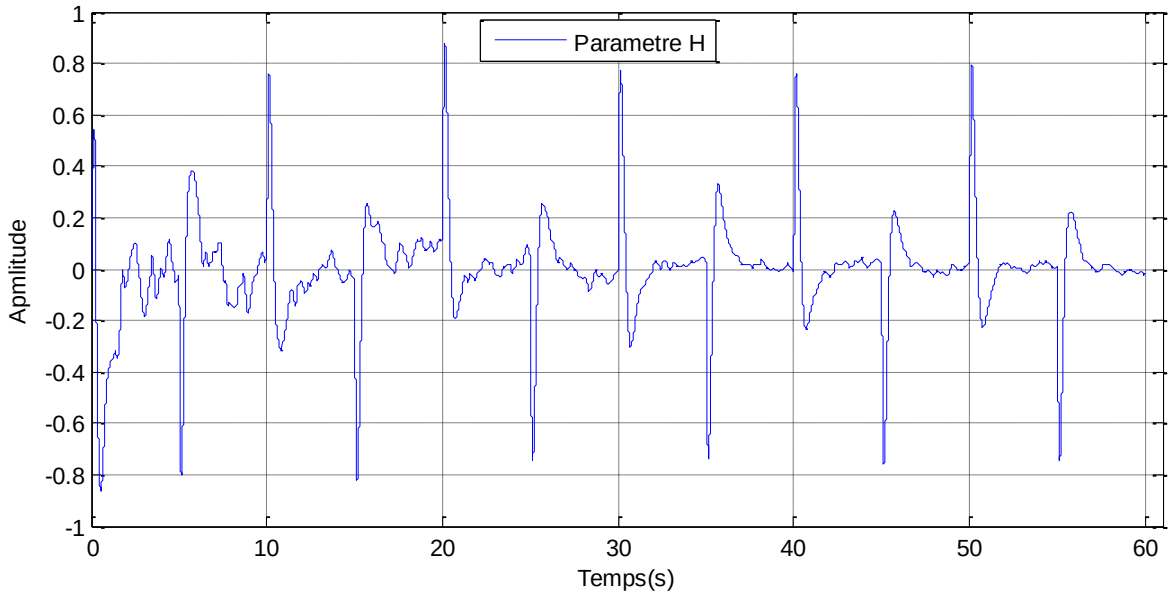


Figure 6.7. *Evolution du paramètre h avec le temps.*

Un signal de bruit blanc a bande limitée et un signal de bruit gaussien aussi sont directement ajouté à l'angle du pendule comme une perturbation au signal d'entrée afin d'examiner la stabilité du dispositif de commande contre les perturbations. La position angulaire contrôlée du pendule en fonction du temps est représentée sur la figure (6.5).

Le signal de perturbation utilisé est représenté sur la figure (6.3). La variation des paramètres g et h est représentée sur les figures (6.6) et (6.7) respectivement.

Il est clair à partir des résultats obtenus que le système de commande est capable de maintenir l'angle du pendule dans sa position d'équilibre instable (verticale) toute en déplaçant le chariot selon le signal de référence de la position linéaire, ceci est effectué en présence de perturbation, de bruit, ou la dérive des paramètres. Les résultats de simulation montrent également que le mécanisme d'adaptation des paramètres est robuste et la commande a réussi d'atténuer l'influence du bruit sur le résultat finale.

Il ressort clairement de la figure (6.6) et figure (6.7) que le régulateur adaptatif est capable d'exécuter un rythme régulier de l'adaptation des paramètres en dépit de la présence de signal de perturbation.

6.6. Avantages et inconvénients de la commande par backstepping [56] [3] [58]

- La commande est effectuée sans aucune connaissance préalable des paramètres du modèle mathématique du système dans le cas de la commande robuste adaptative par backstepping proposée.

- Contrairement à la plupart des autres méthodes, le backstepping n'a aucune contrainte de non linéarité.

- Dans la pratique, le backstepping pose certains problèmes qui limitent son applicabilité.

- En présence de bruit de mesure, le contrôleur par retour d'état résultant du backstepping, provoque un comportement très peu désirable de la variable manipulée.

- La nature avance de phase du contrôleur obtenu par le backstepping ne lui permet pas, en présence de perturbations à moyenne non nulle à l'entrée du procédé, d'éliminer les erreurs résiduelles.

- En présence de bruit de mesure, le contrôleur par retour d'état, généré par le backstepping provoque un comportement peu désirable [58].

6.7. Conclusion

Dans ce chapitre, on a proposé une technique de commande adaptative robuste par backstepping qu'on a appliqué dans le cadre d'une simulation numérique sur le système du pendule inversé. L'algorithme de commande utilisé présente un comportement de régulation stable en présence de paramètres inconnus. Aussi, la simplicité de l'algorithme permet de réduire considérablement la complexité de calcul.

Chapitre 7

Description de l'environnement Matlab, Simulink et dSPACE

7.1. Introduction [59]

L'objectif de ce chapitre est la présentation de la carte de commande dSPACE 1104 et son utilisation en complément avec l'environnement de développement MATLAB et son outil Simulink.

dSPACE est une société allemande créée en 1988 à Paderborn. Elle produit et commercialise des solutions de développement et de tests pour les systèmes de commandes assistés par ordinateurs et les systèmes embarqués.

La société dSPACE a produit plusieurs plateformes matérielles et logicielles très répondues (DS1102, DS1103, MicroAutoBox…). L'intérêt majeur de ces systèmes est qu'ils permettent de développer un schéma de simulation dans l'environnement Simulink et par compilation en langage C on l'exécute en temps réel sur des cartes contenant des DSP (Digital Signal Processor) tout en interagissant avec l'environnement extérieur (système physique). On va ci-dessous désigner quelques systèmes offerts par la société dSPACE :

- **Cartes (matériel) :**

Cette société développe des systèmes mono cartes, modulaires ou bien embarqués sur véhicule :

- Les systèmes mono cartes possèdent des processeurs rapides et s'intègrent dans le PC via des ports PCI ou ISA, exemple le DS1104.
- Les systèmes modulaires offrent de très hautes performances et répondent à toutes les exigences. Ces systèmes peuvent s'intégrer dans les PC ou dans des boitiers externes spéciaux, (DS1105).
- Le MicroAutoBox représente le type des systèmes embarqués sur véhicule, il est à double titre, un calculateur automobile et un système de prototypage.

- **Logiciels :**

dSPACE propose six types de logiciels :

- ControlDesk : permet la visualisation et la gestion de l'expérience en temps réel.
- MotionDesk : permet l'animation du système réel en prenant en compte et en temps réel les contraintes de simulation nécessaires.
- AutomationDesk : permet de faciliter les tests automatisés.
- CalDesk : ce logiciel permet de réaliser l'étape finale du développement du calculateur, il permet de modifier les paramètres du calculateur pendant les périodes de mise au point (calibration).

En addition à ces quatre logiciels, dSPACE offre deux autres applications qui s'associent avec Matlab/Simulink :

- RTI (Real Time Interface) : il se charge de l'implémentation, c'est une interface entre le système développé sous Simulink et les E/S de la carte dSPACE (il contient des blocs Simulink représentant les E/S de la carte dSPACE).
- TargetLink : sert à la compilation du schéma Simulink développé en un code C.

Dans ce projet, on va travailler dans l'environnement Matlab/Simulink/dSPACE DS1104, et on va utiliser pour l'IHM (Interface Homme Machine) le logiciel ControlDesk tout en passant par RTI et TargetLink.

7.2. La carte dSPACE 1104 [62][60][61].

7.2.1. Description introductive

La carte utilisée dans ce travail est " DS1104 R&D Controller Board" développée par la société dSPACE. (R&D désigne recherche et développement).

La carte contrôleur DS1104 est une carte standard. Elle est spécialement conçue pour le développement des contrôleurs numériques multi-variables temps réel, ainsi que pour la simulation temps réel dans différents domaines. C'est un système de commande temps réel complet, intégré dans un PC, basé sur un processeur maître MPC8240.

Pour les manipulations avancées des entrées/sorties, le DS1104 contient un DSP (Digital Signal Processor) esclave basé sur le microcontrôleur DSP TMS320F240.

Donc, le DS1104 est un système mono carte à double processeurs (maître et esclave), avec des contrôleurs d'interruptions, des mémoires, des temporisateurs (timers) et des interfaces d'E/S (figure 7.1).

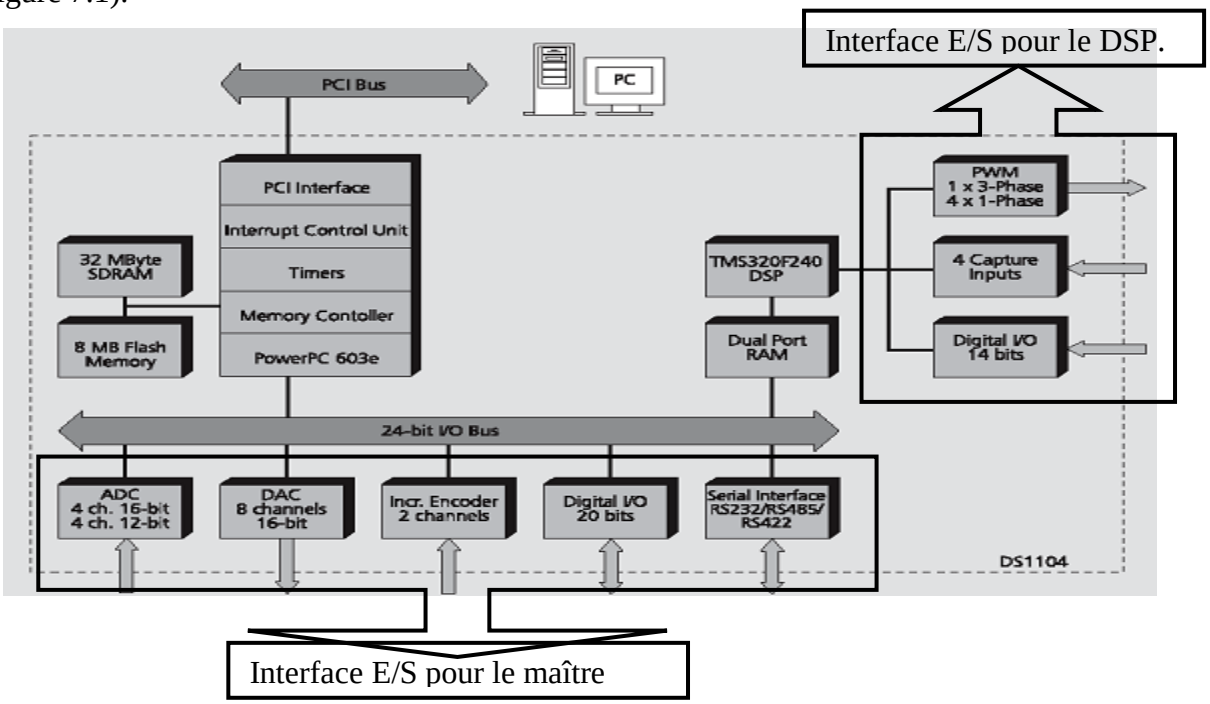


Figure 7.1. *Schéma synoptique de l'architecture du DS1104.*

7.2.2. Composition et caractéristiques

7.2.2.1. Le processeur principal (maître)

C'est l'unité centrale de calcul et de traitement, Motorola MPC8240, il se compose de :

- **Un noyau PowerPC 603^e core (maître PPC)**
  - Vitesse de 250 MHz.
  - Mémoire cache 32 Kbyte.
- **Entrées analogiques ADC (Analog to Digital Converter)** on distingue deux types de convertisseurs ADC :
  - Un CAN (convertisseur analogique/numérique), nommé ADC1 multiplexé sur 4 canaux (ADCH1…ADCH4) sur 16 bits.
  - Intervalle de tension d'entrée ±10v.
  - Erreur ±5mv.
  - Temps d'échantillonnage 2µs.
  - Quatre CAN parallèles (ADC2, … ADC5) avec un canal chacun (ADCH5…ADCH8) sur 12 bits. Mêmes caractéristiques que le premier.

- **Sorties analogiques DAC (Digital to Analog Converter)**

  8 (CNA) Convertisseurs Numérique/Analogique (DACH1…DACH8) sur 16 bits délivrant une tension de ± 10 V avec une erreur de ± 1 mV.

- **Encodeurs incrémentaux :**
  - 2 entrées encodeurs supportant les deux technologies TTL et RS422.
  - Sur 24 bits.
- **Entrées/Sorties numériques :**
  - 20 bits d'E/S (les bits sont sélectionnables comme des entrées ou bien des sorties).
  - Courant maximal de sortie : ± 5 mA.
- **L'interface série :**

  Le DS1104 dispose UART (Universal Asynchronous Receiver and Transmitter), c'est un récepteur/émetteur asynchrone universel, son rôle est l'accomplissement de la communication série asynchrone avec les dispositifs externes. L'UART peut être configuré comme émetteur/récepteur RS232, RS422 ou bien RS485.

7.2.2.2. Le DSP esclave

Le sous ensemble du DSP contient :

- **Un DSP (Digital Signal Processor) :** c'est le processeur de traitement de signal TMS320F240 développé par Texas Instruments, il se caractérise par :
  - Calculant à 25 MHz.
  - Une mémoire de 4K × 16 réservée pour la communication avec le maître.

- **Interface d'E/S pour le DSP :** l'interface du DSP avec son environnement extérieur se compose de :
  - Unité d'E/S numérique sur 14 bits, direction sélectionnable, courant maximal de ± 13 mA et supportant la technologie TTL.
  - Unité d'E/S numérique de synchronisation qui permet de générer et mesurer des signaux PWM (Pulse Width Modulated) et des signaux carrés.

7.2.2.3. Les mémoires

- 32 Mo en SDRAM.
- 8 Mo en Flash pour les applications.

7.2.2.4. Temporisateurs(Timers)

- 6 Temporisateurs sont disponibles dans le DS1104.

7.2.2.5. Unité de contrôle d'interruptions

Elle se charge de la gestion des interruptions issues des timers, des interfaces séries, du DSP esclave, encodeurs incrémentaux, ADC, etc. Il y a trois différentes méthodes pour connecter le DS1104 avec l'environnement extérieur :

- A travers le P1 du DS1104, c'est un connecteur d'E/S à 100 épingles (pins).
- A travers un câble adaptateur avec deux connecteurs d'E/S subdivisés (50 pins chacun).
- Via le panneau de connexion appelé CP1104 ou également CPL1104.

Cette dernière est la méthode choisie pour notre manipulation.

En effet, Le module dSPACE DS1104 utilisé se compose de deux parties :

- Une carte d'interface, équipée d'un maître PPC et d'un DSP esclave, permettant l'acquisition et le traitement de données sur ordinateur ainsi que l'envoi de signaux vers les autres éléments du banc (figure 7.2.A).
- Un panneau de connexion composé de 8 prises BNC permettant la conversion de signaux analogiques en numériques (ADCH) avant de les transmettre à la carte et 8 sorties (DACH) délivrant en analogique les signaux numériques envoyés par la carte. Le panneau comporte aussi une connexion d'Entrée/Sortie de l'esclave, deux connexions série, deux connexions pour encodeurs et une connexion Entrée/Sortie numérique (figure 7.2.B).

Figure 7.2.A. *La carte dSPACE DS1104.*

Figure 7.2.B. *Le panneau de connexion.*

Ces deux parties constituant le DS1104 sont interconnectées par l'intermédiaire d'un câble adaptateur dit : ADP_CAB1104.

7.2.3. La programmation de la carte DS1104

Pour programmer le DS1104, il faut réaliser tout d'abord un schéma de commande dans l'environnement Matlab/Simulink. Il est préférable de mettre Matlab et Simulink dans le même répertoire de travail, car, la compilation génère de nombreux fichiers ayant différentes extensions.

Dans "Simulink Library Browser", et après l'installation de la carte dSPACE, on trouve une librairie nommée "dSPACE RTI1104"dans laquelle on peut choisir les composants utilisés pour la réalisation du schéma Simulink (Master Bit in/out, set encoder, ADC_C1…).

La figure (7.3) illustre des exemples de blocs trouvés dans la librairie "dSPACE RTI1104". En faisant un double clic sur chaque bloc, on peut spécifier l'Entrée/Sortie dans laquelle on souhaite acquérir ou affecter des données.

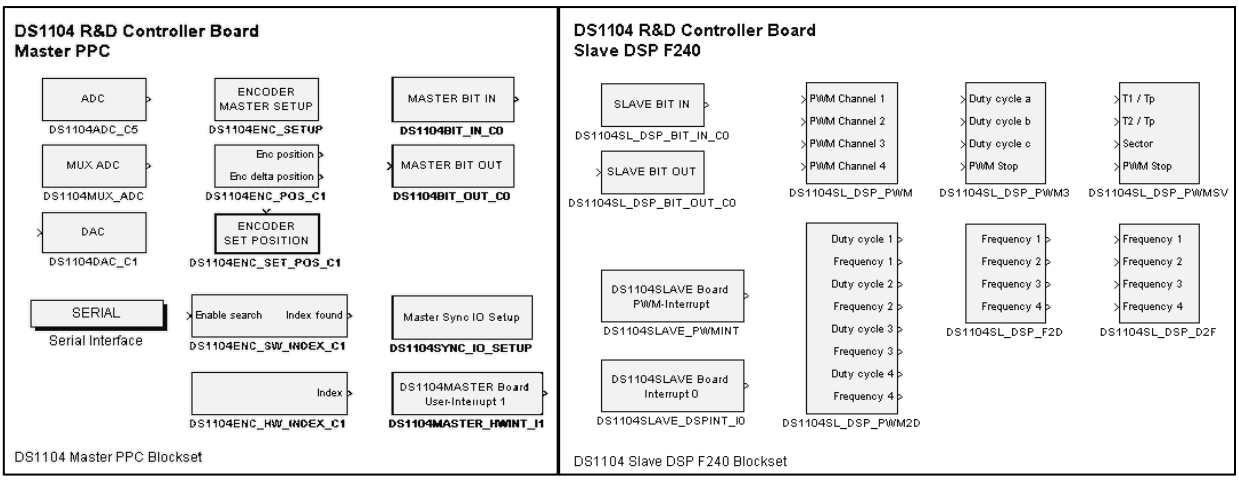


Figure 7.3. *Exemple de blocs du DS1104.*

Une fois le schéma Simulink est terminé, testé en simulation et sauvegardé, l'étape suivante consiste à générer le code C associé au schéma Simulink et le charger dans le DS1104. La compilation du schéma Simulink s'effectue en faisant suivre les étapes suivantes :

- Aller dans le menu ***simulation***, puis ***configuration parameters*** (ou également aller dans le menu ***Tools***, puis ***Real-Time Workshop)***.
- Dans la fenêtre qui va apparaître, dans l'onglet ***Solver***, fixer Type du pas d'échantillonnage à *fixed-step* et sa valeur à Ts (Ts $= 10^{-4}$ pour le DS1104).
- Dans l'onglet ***real-time workshop***, choisir la cible puis ***Build***.

Si la compilation a réussi, le chargement du programme et son exécution dans le DSP se font automatiquement. Après la compilation, la suite de l'expérience se fait au niveau d'un autre logiciel appelé ControlDesk.

7.3. Le logiciel ControlDesk [62][63][64][20]

Control Desk est un logiciel de développement des Interfaces Homme/Machine (IHM) qui permettent à l'utilisateur la visualisation et la gestion, en temps réel, des différentes variables du fichier développé sous MATLAB/Simulink ou également sous C. La visualisation de variables ou de signaux et la modification de paramètres sont possibles par l'intermédiaire d'instruments graphiques fournis par ControlDesk. La figure (7.4) représente la fenêtre principale du ControlDesk.

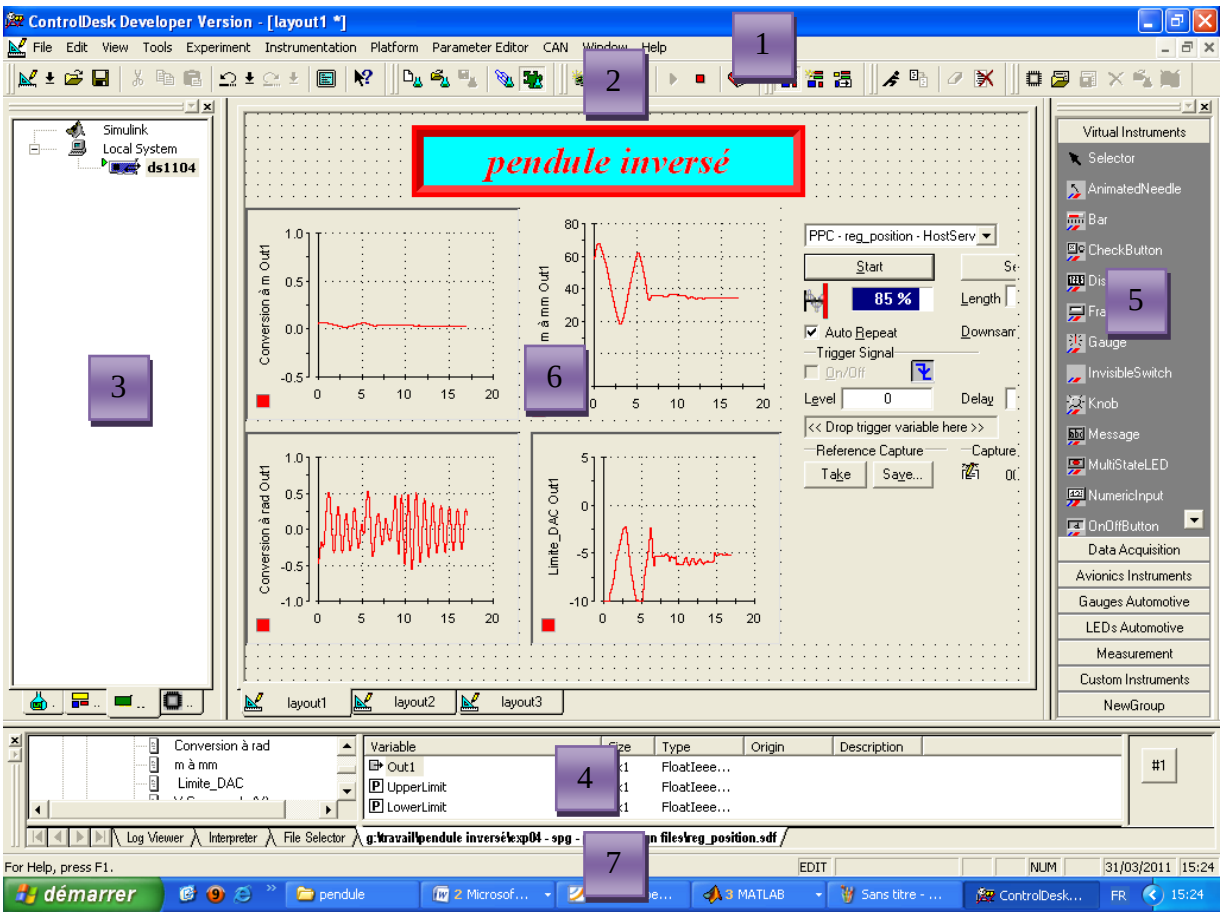


Figure 7.4. *Environnement du logiciel ControlDesk.*

L'installation de la carte dSPACE sur le PC ajoute automatiquement le logiciel ControlDesk dans le bureau.

7.3.1. Eléments de base de ControlDesk

Après le lancement du ControlDesk, la fenêtre de la figure (7.4) s'ouvre, elle est constituée de :

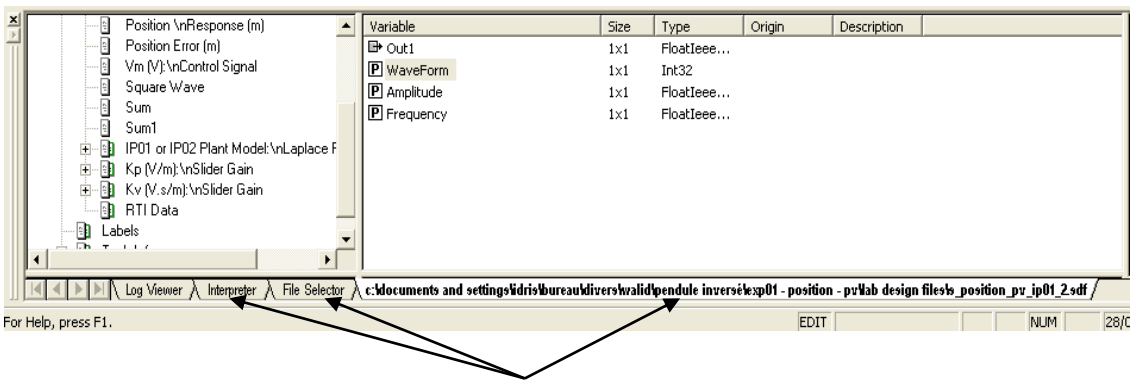
1. **La barre de menus :** Comme tout logiciel, la barre de menu du ControlDesk contient ses différentes fonctions (File, Edit…), Et chacune de ces fonctions contient des commandes auxiliaires.
2. **La barre d'outils :** Elle contient les commandes fréquemment utilisées (open, Save …).

3. **Le navigateur :** Il contient quatre fenêtres de navigation (Experiment, Instrumentation, Platform et CAN).

   - ***Le navigateur Experiment* :** il affiche tous les dossiers (des panneaux d'instruments, des ensembles de paramètre, des données de référence…) appartenant à l'expérience ouverte et fournit des fonctions pour manipuler l'expérience et ses composants.
   - ***Le navigateur Instrumentation :*** nous permet de construire des panneaux d'instruments pour commander et surveiller les variables d'un modèle.
   - ***Le navigateur Platform :*** montre toutes les plateformes enregistrées dans votre système et fournit des fonctions pour manipuler des applications telles que la charge, le début et l'arrêt.
   - ***Le navigateur CAN :*** montre la configuration du bus CAN qui contient tous les contrôleurs, bases de données, et messages d'une application en temps réel établie avec RTI et RTICANMM.

4. **La fenêtre d'outils**

   Cette fenêtre fournit certains outils selon le navigateur activé.



Les navigateurs de la fenêtre d'outils

Figure 7.5. *La fenêtre d'outils.*

- **Le navigateur Log Viewer:** montre les messages générés par la carte ou par ControlDesk.
- **Le navigateur Interpreter:** pour la manipulation des commandes et scripts écrits en Python.
- **Le navigateur File Selector:** pour la charge de l'application par outil ''glisser et déposer''.
- **Le navigateur Reference Data Manager:** la manipulation des données capturées et données de référence.

5. **La fenêtre d'instruments (instrument selector) :**

   Elle contient tous les instruments pour le développement de l'interface. Pour plus de détail sur les instruments virtuels utilisés sous ControlDesk se référer au manuel des instruments [20].

6. **L'espace de travail (working area):**

   Cette zone sert à la création des layouts et à la connexion des données, elle permet de visualiser et d'éditer les expérimentations.

7. **La barre d'état (status bar) :**

   Indique l'état actuel du ControlDesk.

7.3.2. Manipulation des fichiers

ControlDesk utilise quelques techniques standards et spéciales pour manipuler des dossiers. Il combine entre plusieurs types de fichiers afin de pouvoir gérer les données et les applications. Ces fichiers sont enregistrés sous différentes extensions (.CDX, .CON, .SDF…), cette extension est un moyen de reconnaitre le type de programme avec lequel ce fichier peut être ouvert.

7.3.3. Chargement d'une application sous ControlDesk

Il y a plusieurs méthodes pour charger une application, déjà développée sous Simulink et builder en C, une de ces méthodes consiste à :

- Aller dans le navigateur ***File Selector***.
- Récupérer le fichier ayant pour extension '' ****.sdf*** '' qui est créé par Simulink au moment du ***'build***' en glissant et déposant ce fichier dans la plate-forme (Le navigateur ***Platform*** => DS1104).

Après le chargement du projet dans ControlDesk, on insérera ensuite à l'espace de travail (layout) de notre projet (fichier => new => layout). Puis et en fin, on ajoutera tous les instruments de visualisation et de modification des variables. Pour se faire, on clique sur ‘’model root ‘’ et on fait par un click gauche sur la variable, glisser le « out » (lecture de la variable) ou le «value» (modification de la variable) vers l'instrument visuel qui doit y être relié. Pour visualiser, modifier et tester l'interface de commande développée, ControlDesk dispose de trois modes de fonctionnement :

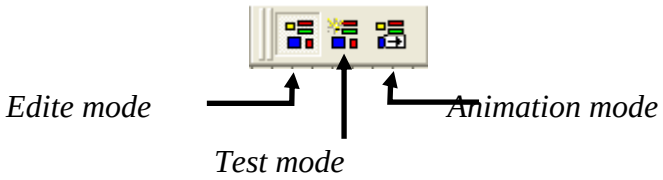


Figure 7.6. *Barre d'outils d'instrumentations et de layout.*

- ***Mode Edition (Edite mode) :*** permet de créer et de modifier les instruments.
- ***Mode Test (Test mode):*** permet de tester l'interface.
- ***Mode Animation (Animation mode) :*** rend les panneaux opérationnels.

Dans les deux parties précédentes, On a fait une brève description de la carte DS1104 et du logiciel ControlDesk. Et on a vu que sa programmation se fait dans l'environnement Simulink. Cependant, il est à noter que ces deux logiciels ne suffisent pas pour accomplir la programmation de la carte. En effet, on aura besoin d'un autre logiciel : c'est l'interface en temps réel (RTI).

7.4. L'interface en temps réel RTI [61]

L'interface en temps réel (RTI) dans les systèmes dSPACE se charge de lier Simulink avec le matériel de dSPACE. Elle permet d'ajouter au modèle Simulink des interfaces d'Entrées/Sorties, des timers, des interruptions…etc., propres à une carte spécifique.

7.4.1. L'accès aux blocs de l'RTI

L'installation de la carte dSPACE sur le PC ajoute automatiquement l'RTI au Matlab/Simulink.

L'RTI contient une variété de blocs permettant l'accès au matériel de dSPACE. En effet, Il y a deux méthodes distinctes pour obtenir un bloc de RTI : à partir de la bibliothèque de blocs de l'RTI, ou bien en utilisant le browser de la bibliothèque de Simulink.

- **A partir de la bibliothèque de l'RTI :**

  Pour accéder à un bloc de RTI à partir de sa bibliothèque on doit procéder comme suit :

  - On ouvre la bibliothèque de RTI en tapant ''rti '' dans la fenêtre de Matlab,
  - On ouvre la sous bibliothèque voulue, on choisit le bloc, puis on l'ajoute au modèle de Simulink.

Figure 7.7. *Bibliothèque de l'RTI 1104.*

- **A partir du Simulink :**

  La bibliothèque complète de l'RTI s'ajoute au browser de Simulink. Donc pour accéder aux blocs de l'RTI il suffit d'ouvrir le browser de Simulink puis la bibliothèque correspondante et choisir le bloc voulu, puis on l'ajoute au modèle Simulink tel que tous les autres blocs de Simulink. L'interface en temps réel RTI est donc essentielle pour accomplir la programmation de la carte dSPACE. Donc pour aboutir à une commande convenable il faut retenir la configuration des paramètres de simulation appropriés, la méthode de construction et le chargement du modèle sur le matériel du dSPACE.

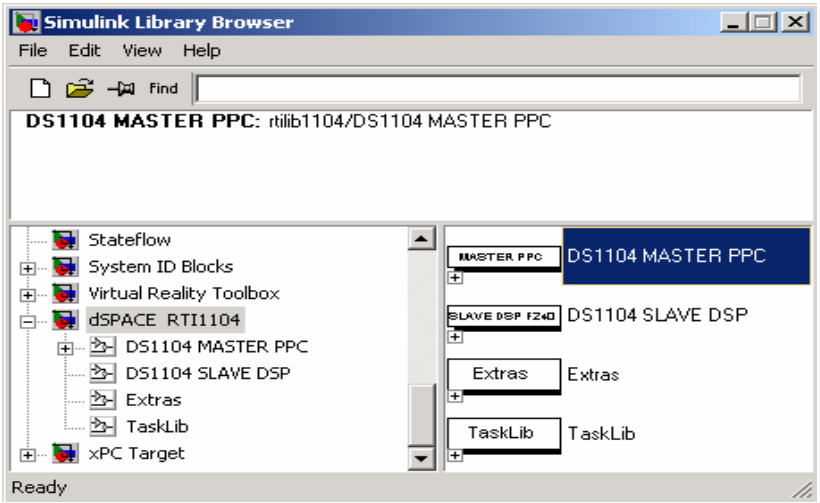


Figure 7.8. *Bibliothèque de RTI 1104 dans le browser du Simulink.*

7.5. Conclusion

L'association du produit dSPACE avec Matlab/Simulink résulte un environnement de travail très souple et très répandu. Car cet environnement offre aux développeurs de commandes par calculateurs et systèmes de commande embarqués la possibilité de réduire leurs temps de développement et d'augmenter la qualité de leurs produits, grâce à un ensemble de matériels et logiciels rendant aisées toutes les phases de conception de commandes, de la simulation sous Matlab/Simulink à la génération/chargement du code C dans la carte, tout en passant par le Prototypage Rapide de Commande (RCP) et le Hardware In the Loop (HIL). Donc c'est un environnement de développement de commandes complet.

Chapitre 8

Application et implémentation

8.1. Introduction

Dans ce chapitre, nous avons effectué des commandes en temps réel, d'abord en simulation et ensuite sur le système réel du pendule inversé en utilisant l'environnement de commande présenté dans le chapitre 7.

Les algorithmes associés aux différentes techniques de commande étudiées dans les chapitres précédents seront donc mises en œuvre dans cette partie. La configuration des paramètres de chaque technique est effectuée par programmation sur MATLAB. Cependant, la simulation numérique a été facilitée par l'utilisation de l'environnement Simulink. Ceci est nécessaire pour pouvoir faire l'implémentation sur la carte dSPACE 1104 et valider les résultats de simulation sur la plate-forme réelle. Les résultats seront présentés, discutés et analysés.

8.2. Conditions d'application

Afin de comparer les résultats des différentes méthodes, on a essayé de respecter les mêmes conditions d'expérimentation conformément aux critères suivants:

- Conditions initiales: toutes les vitesses et les positions sont égales à Zéro.
- Pas d'échantillonnage égal à 10^{-3} sec.
- Angle de référence $\theta = 0$.
- La position de référence est un signal carré de fréquence 0.1 Hz et d'amplitude 10 cm.
- Temps total égal à 60 secondes.

8.3. Conditions de validation

1. Satisfaction des contraintes de temps imposées par le système temps réel du pendule inversé.
2. L'amplitude de débattement angulaire n'est pas très importante.
3. Le déplacement du chariot sur le rail doit être limité.
4. On ne se préoccupe pas de limiter les vitesses atteintes.
5. Le signal de commande $u(t)$ doit varier dans le régime nominal dans la plupart du temps à $\pm$ 6 Volt et ne pas dépasser la tension maximale de $\pm$ 13 Volt.

8.4. Critères d'évaluation

- **Stabilité et suivi du signal de référence**:

  On impose à la position du chariot une trajectoire de référence carrée de valeur $x_{ref} = \pm$ 10 cm. La commande doit stabiliser le pendule au point critique instable de la position verticale tout en déplaçant le chariot.

- **Robustesse :** la capacité de résister aux perturbations extérieures.

8.5. Simulation numérique sous Simulink

Dans cette partie on va développer le modèle Simulink de chaque technique de commande. Après simulation, les différentes courbes de position (en mètre), de vitesse (en mètre/s), d'angle (en degré), de vitesse angulaire (en radian/s), de commande (en volt) et des paramètres spécifiques pour chaque cas sont présentées.

Afin d'évaluer la robustesse, une perturbation est imprimée à l'angle, à l'instant t=14s.
Les figures ci-dessous regroupent pour chaque technique de commande les résultats obtenus et les modèles Simulink développés.

8.5.1. Commande par le régulateur LQR

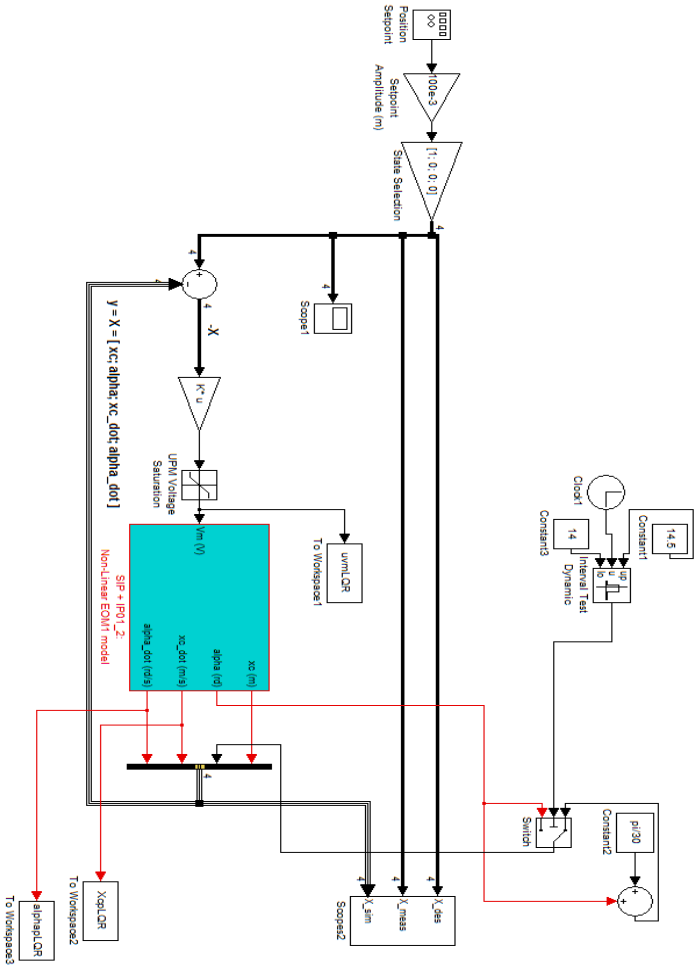


Figure8.1. *Schéma Simulink global pour la simulation de la commande.*

En utilisant l'algorithme LQR (Linear-Quadratic-Regulator), on obtient les gains du régulateur d'état K, l'instruction Matlab utilisée est définit comme suit :

$$\mathbf{K} = \mathbf{lqr}\,(\mathbf{A}, \mathbf{B}, \mathbf{Q}, \mathbf{R}).$$

On a donc opté pour la configuration suivante:

- $K = [-50.0000 \;\; 180.1629 \;\; -50.263 \;\; 28.4893]$.
- On a choisi les pondérations suivantes :

$$Q = \begin{pmatrix} 0.7 & 0 & 0 & 0 \\ 0 & 4 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

Et $R = 3.10^{-4}$

8.5.1.1. Résultats de simulation

Les résultats obtenus sont représentés dans les figures suivantes :

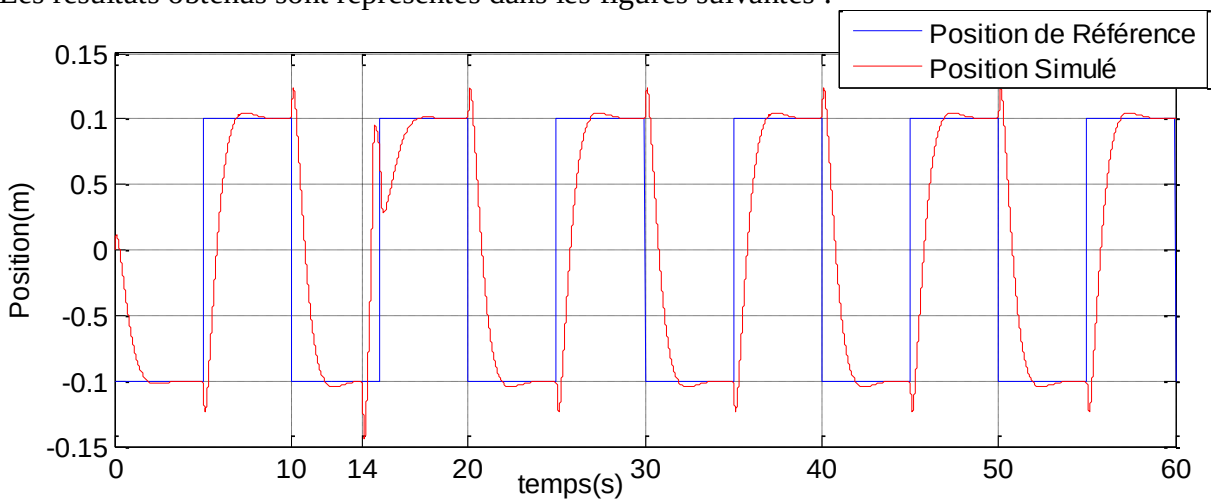


Figure 8.2. *Courbe de position de simulation en mètre.*

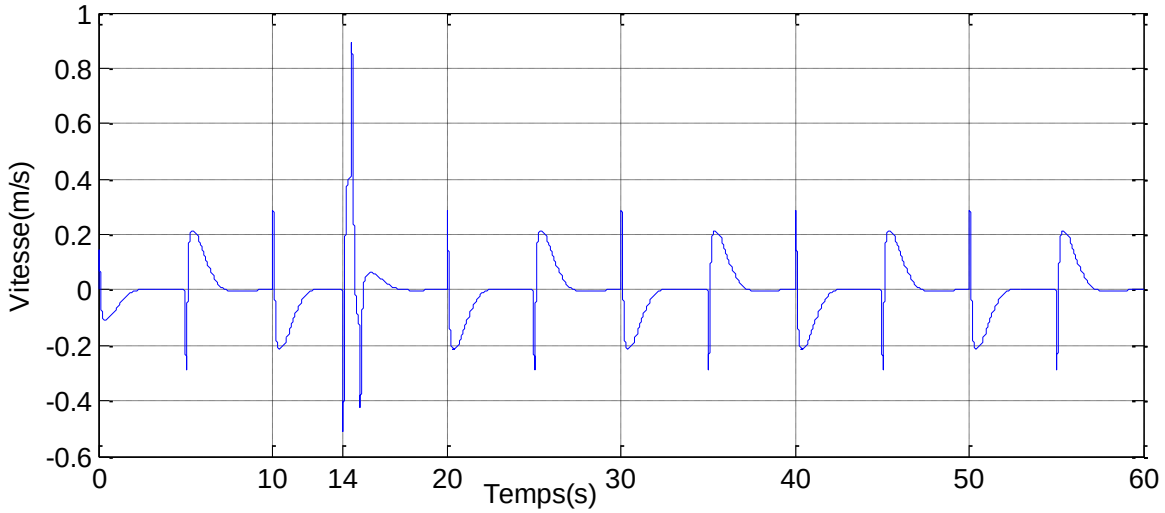


Figure 8.3. *Courbe de la vitesse linéaire en m/s.*

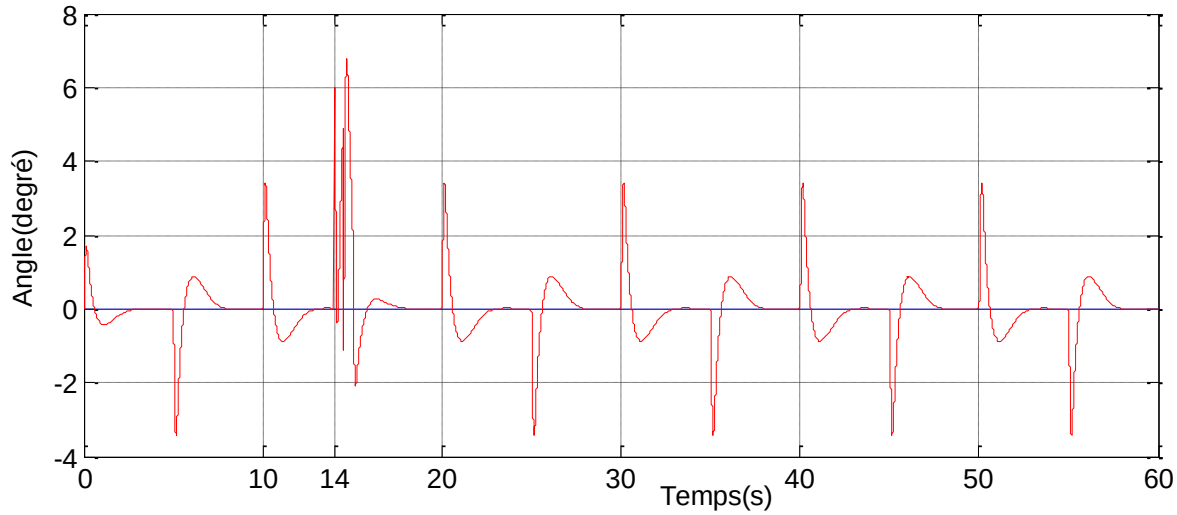


Figure 8.4. *Courbe d'angle de simulation en degré.*

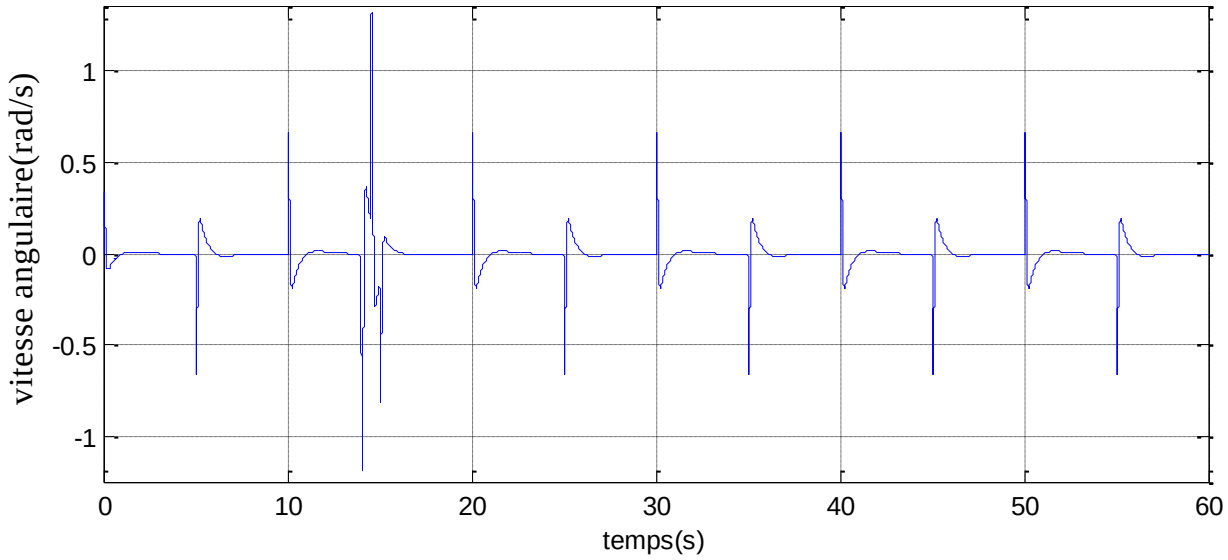


Figure 8.5. *Courbe de vitesse angulaire en radian/s.*

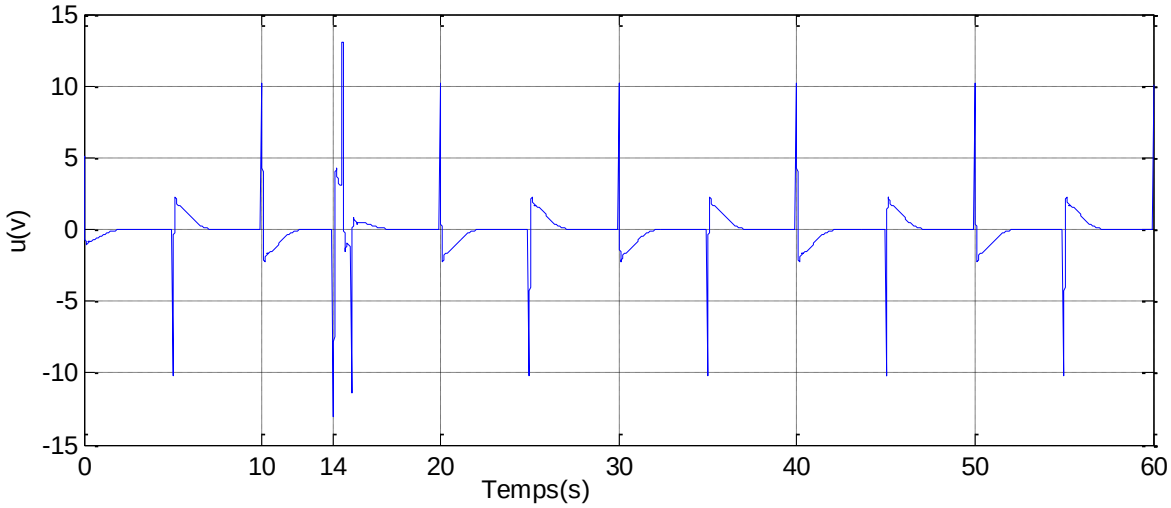


Figure 8.6. *Courbe de commande appliquée sur le moteur.*

- **Interprétation des résultats :**

- La commande de la tension d'alimentation du moteur est acceptable et varie dans le régime nominal inférieur à 6v dans la plupart du temps.
- La position du chariot suit le signal de référence tout en maintenant le pendule stable dans la position verticale.
- Le temps de réponse est acceptable et la commande satisfait les contraintes de temps réel.
- La commande LQR est robuste envers les perturbations extérieures.

8.5.2. Commande par réseau de neurones à apprentissage hors ligne (RNA hors ligne)

L'apprentissage du réseau est effectué en utilisant le régulateur LQR, les exemples pour l'entrainement sont générés à partir des signaux réels issus de l'expérimentation sur plateforme réelle. Le vecteur d'entrée est composé des variables d'état du pendule inversé ainsi que les variables représentant les signaux désirés.

8.5.2.1. Structure générale du réseau

Le choix de la structure adéquate du réseau qu'on envisage de concevoir est déterminé par la nature des phénomènes à identifier. Beaucoup d'approches et de types de réseaux de neurones ont été étudiés mais le schéma du Perceptron Multi Couche (MLP) à apprentissage supervisé demeure le schéma le plus simple et le plus populaire [18], ce qui justifie son choix pour cette étude.

8.5.2.2. Apprentissage du réseau

L'entrainement du réseau nécessite une préparation de l'espace des exemples. Ces exemples représentent les actions correctes que le réseau doit apprendre. Chaque couple d'exemple est composé d'un ensemble d'entrées-sorties enregistrées lors de la stabilisation du pendule par la technique de commande LQR sur le système réel.

8.5.2.3. Configuration optimale

La performance du système à concevoir dépend directement de sa configuration lors de l'apprentissage. Un réseau de neurones optimum est celui qui est composé d'un nombre minimum de neurones et qui nécessite moins d'informations à l'entrée tout en gardant un certain seuil de l'erreur quadratique moyenne.

Il est possible de rechercher une bonne configuration en utilisant le principe de discriminance. Avec ce principe les essais sont effectués en réduisant le nombre de neurones à chaque fois tout en modifiant sa configuration.

L'organigramme suivant résume la méthodologie de conception utilisée :

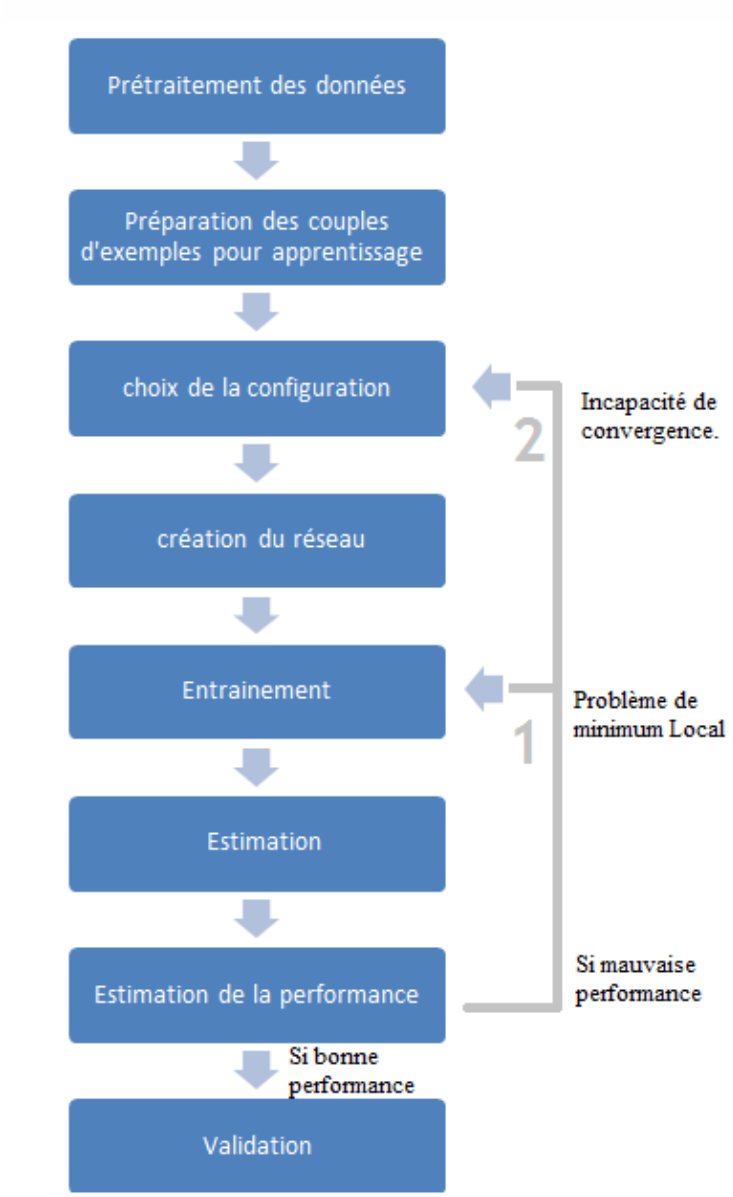


Figure 8.7. *Organigramme de méthodologie pour la conception de la commande neuronale.*

Le choix de la configuration est fixé après plusieurs essais en utilisant l'environnement de programmation Matlab, le réseau final est configuré comme suit:

- 5 neurones composant la couche d'entrée.
- 4 et 3 neurones composant les deux couches cachées successivement.
- 1 neurone pour la couche de sortie.
- La fonction d'activation est la tangente sigmoïde pour toutes les couches sauf pour la couche de sortie où on a utilisé la fonction linéaire.

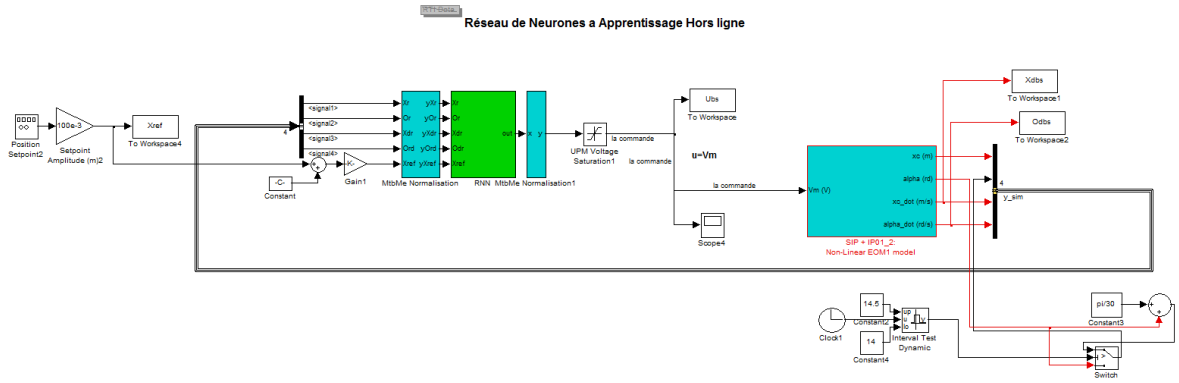


Figure 8.8. *Schéma Simulink global pour la simulation de la commande.*

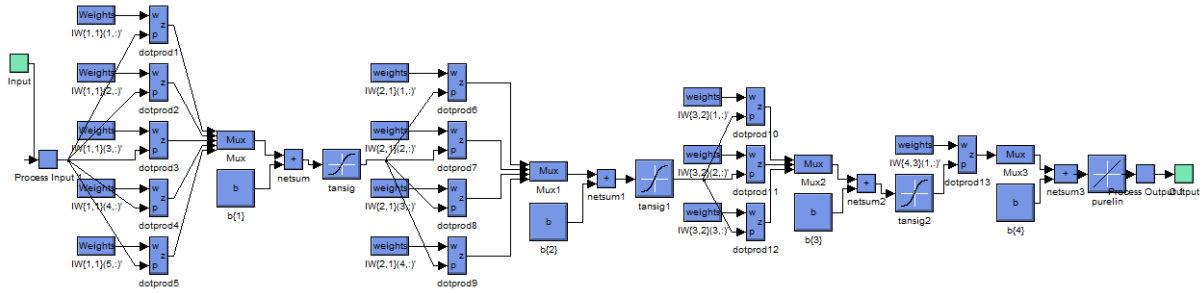


Figure 8.9. *Architecture interne du réseau de neurones utilisé.*

Les figures (8.8) et (8.9) représentent les schémas Simulink réalisés, la performance au sens des moindres carrés obtenu après apprentissage sur les signaux réel est assez satisfaisante et elle est de l'ordre de $10e^{-11}$, après une durée correspondant à environ 3000 époques.

8.5.2.4. Résultats de simulation

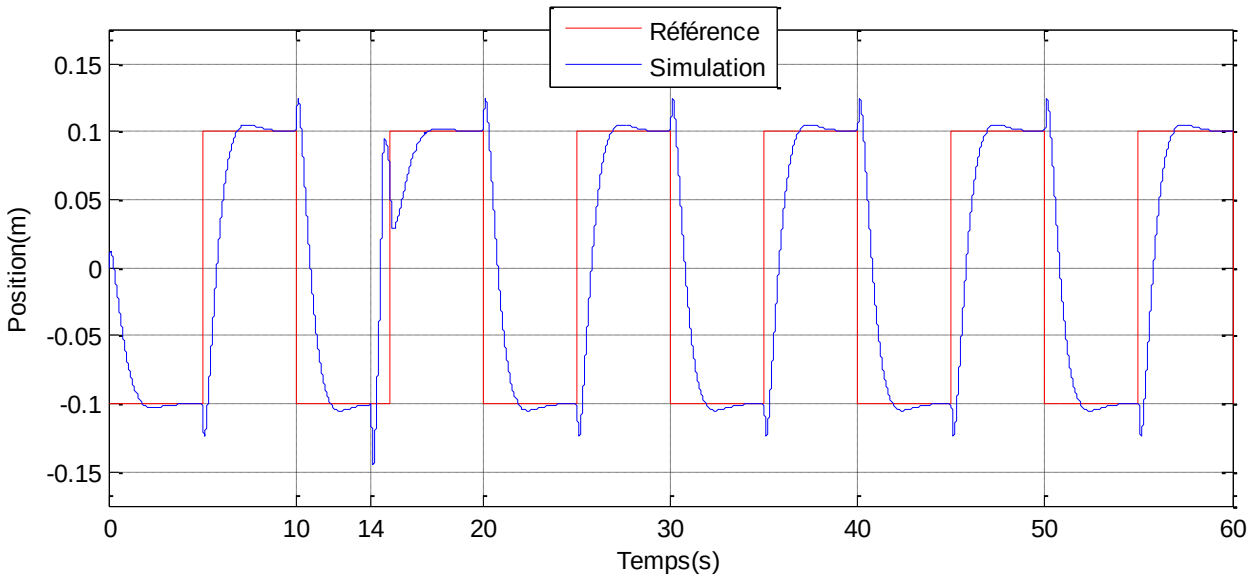


Figure 8.10. *Courbe de position de simulation en m.*

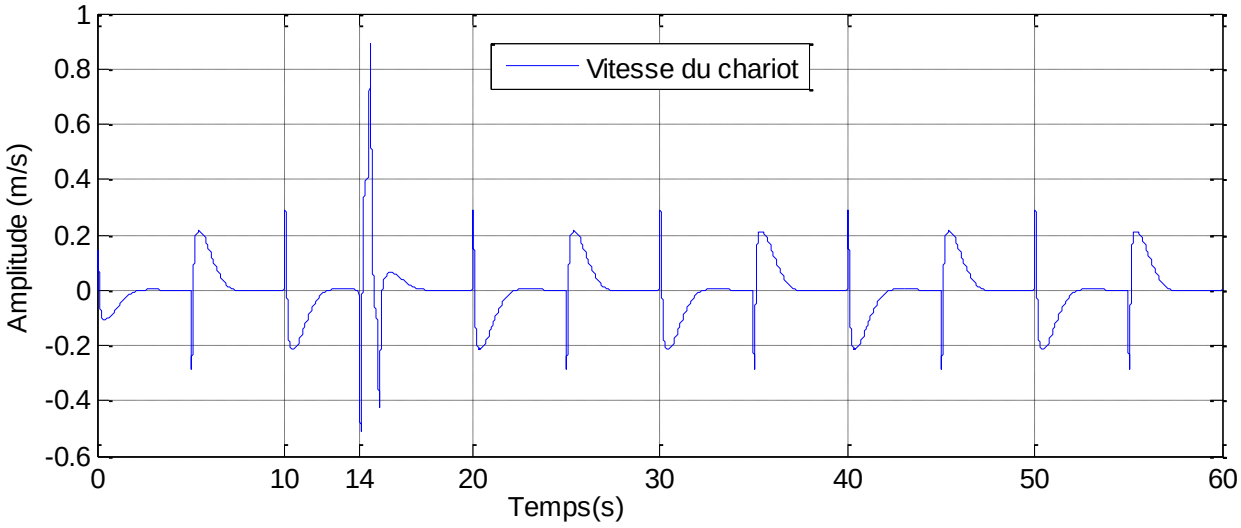


Figure 8.11. *Courbe de vitesse linéaire en m/s.*

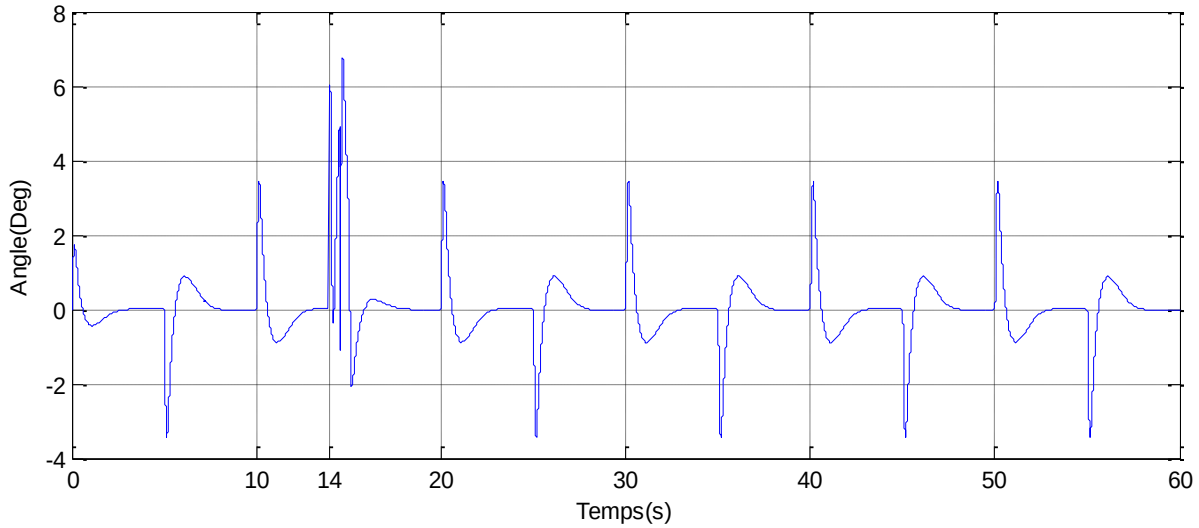


Figure 8.12. *Courbe d'angle de simulation en degré.*

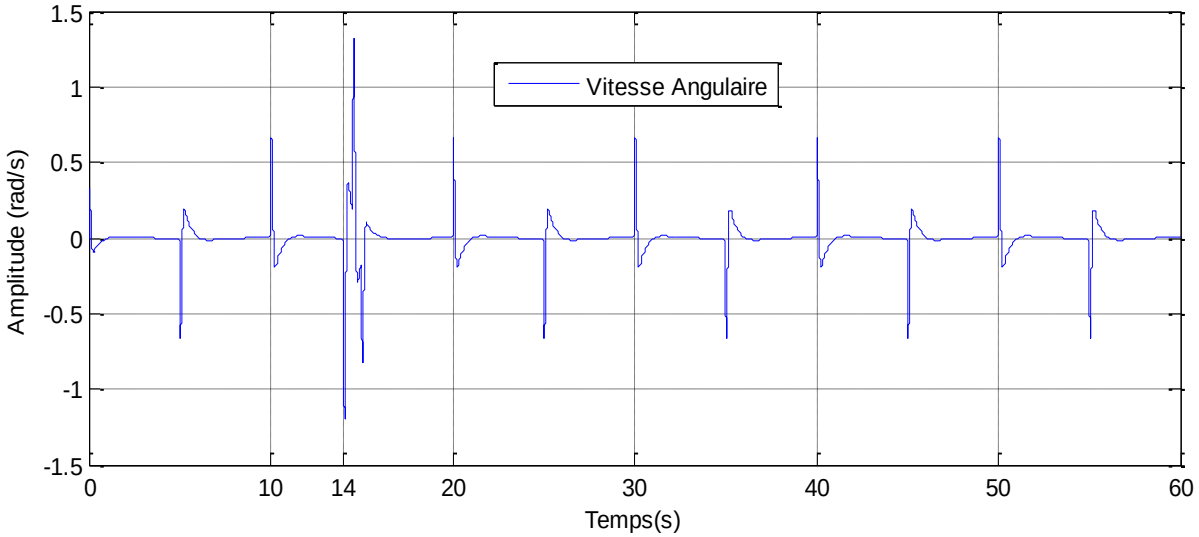


Figure 8.13. *Courbe de vitesse angulaire en radian/s.*

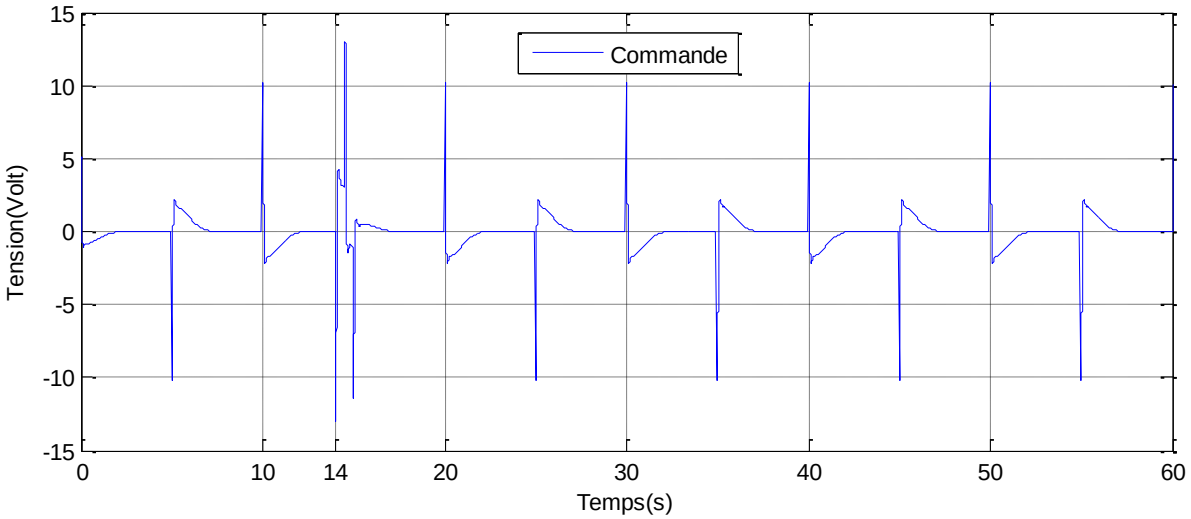


Figure 8.14. *Courbe de commande appliquée sur le moteur.*

- **Interprétation des résultats:**
- La commande de la tension d'alimentation du moteur est acceptable et varie dans le régime nominal inférieur à 6v dans la plupart du temps.
- La position du chariot suit le signal de référence tout en maintenant le pendule stable dans la position verticale.
- Le temps de réponse est acceptable et la commande satisfait les contraintes de temps réel.
- La commande est robuste envers les perturbations extérieures.

8.5.3. Commande par mode de glissement incrémental (ISMC)

8.5.3.1. Estimation des constantes de conception

La loi de commande comporte 5 constantes de conception qui sont arbitraires. L'estimation de ses constantes se fait souvent par le principe de l'essai et de l'erreur.

Après plusieurs essais, on a constaté que l'estimation des 5 constantes est très difficile et semble impossible à cause de la sensibilité de certaines d'entre elles. Afin de régler ce problème, on a opté pour des méthodes numériques en programmation sur Matlab. Ces méthodes permettent d'évaluer l'ensemble de ces constantes. Les valeurs obtenues sont définies comme suit :

$$\lambda_1 = 100, \lambda_2 = 28.489, \lambda_3 = -50\ , \lambda_4 = -41, C = 0.4, k = 50.$$

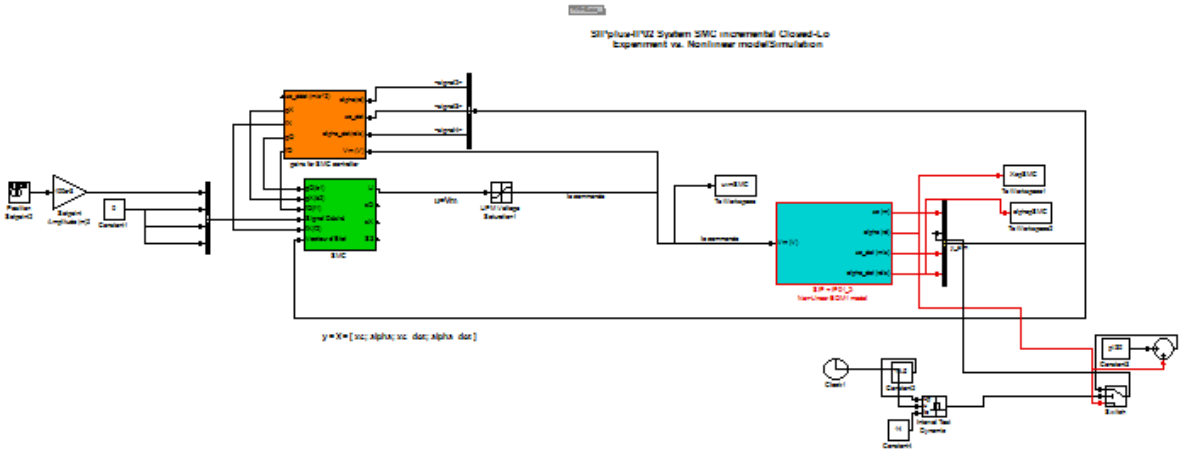


Figure 8.15. *Schéma Simulink global pour la simulation de la commande.*

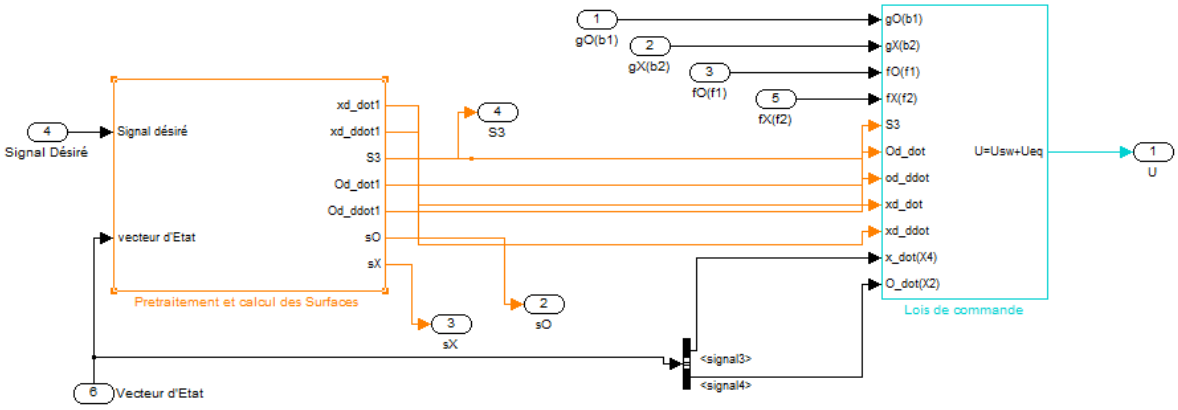


Figure 8.16. *Schéma Simulink de la loi de commande.*

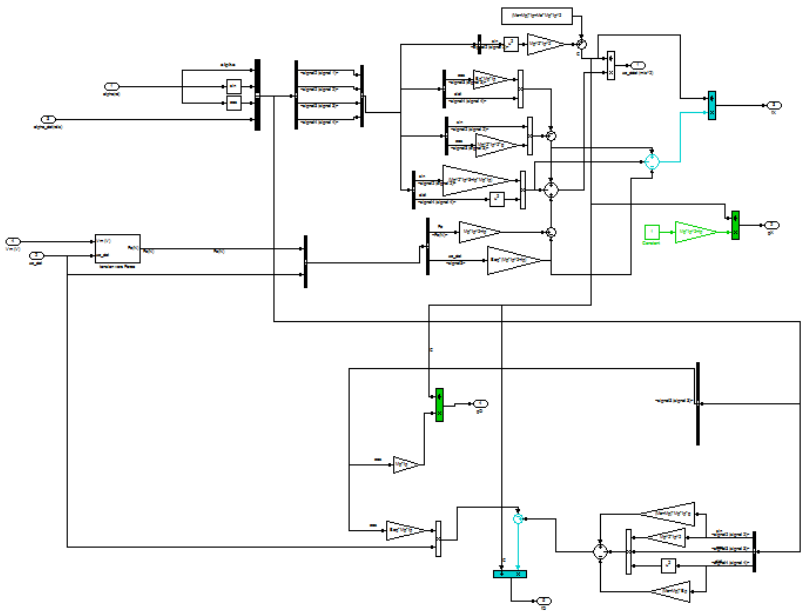


Figure 8.17. *Schéma Simulink des paramètres non linéaires du pendule inversé.*

Les figures (8.15), (8.16) et (8.17) représentent les schémas Simulink réalisés.

8.5.3.2. Résultats de Simulation

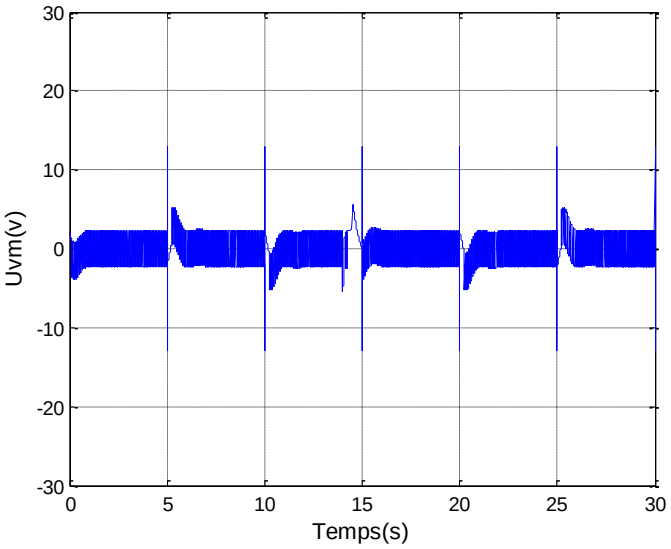


Figure 8.18. *Courbe de commande appliquée sur le moteur.*

Il est intéressant de noter que la commande obtenue induit d'importantes oscillations indésirables qu'on appelle souvent broutement ou Chattering. L'effet du broutement comme le montre le résultat obtenu sur la figure (8.18) est le plus grand inconvénient de la commande par mode de glissement. Afin de diminuer son effet, on a remplacé la fonction de commutation « Sign» par la fonction continue « Sat ». La nouvelle courbe de commande est illustrée sur la figure suivante:

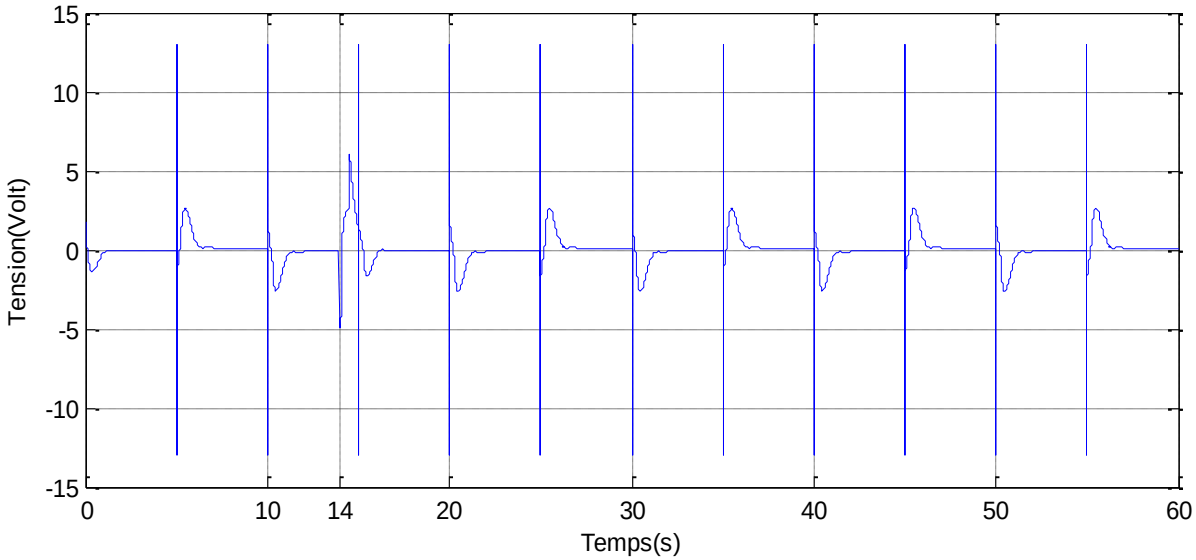


Figure 8.19. *Courbe de commande appliquée sur le moteur sans broutement.*

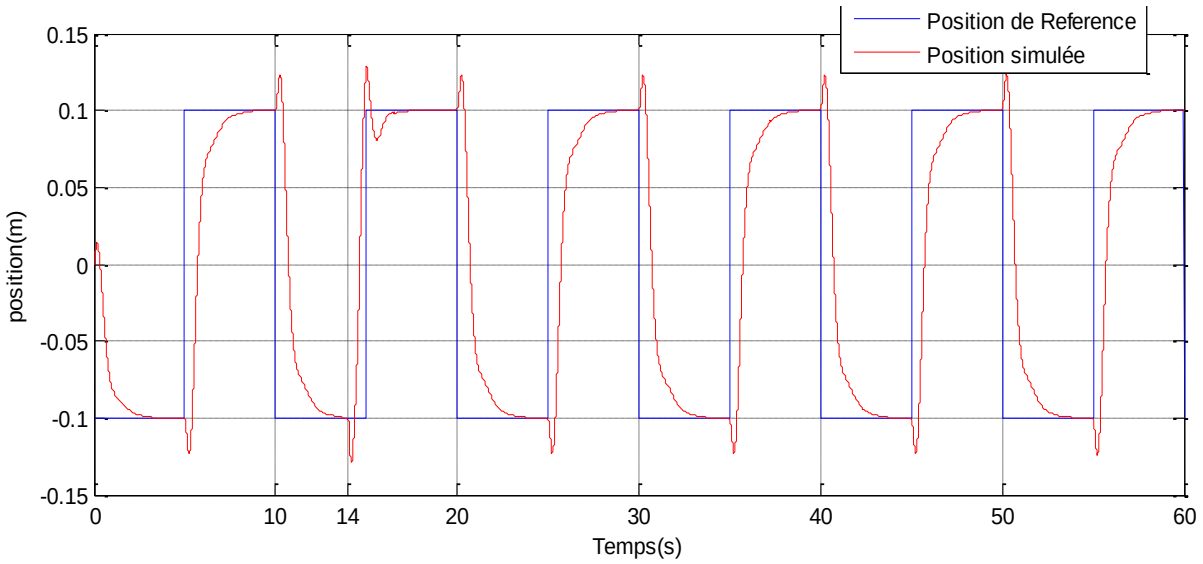


Figure 8.20. *Courbe de position de simulation en m.*

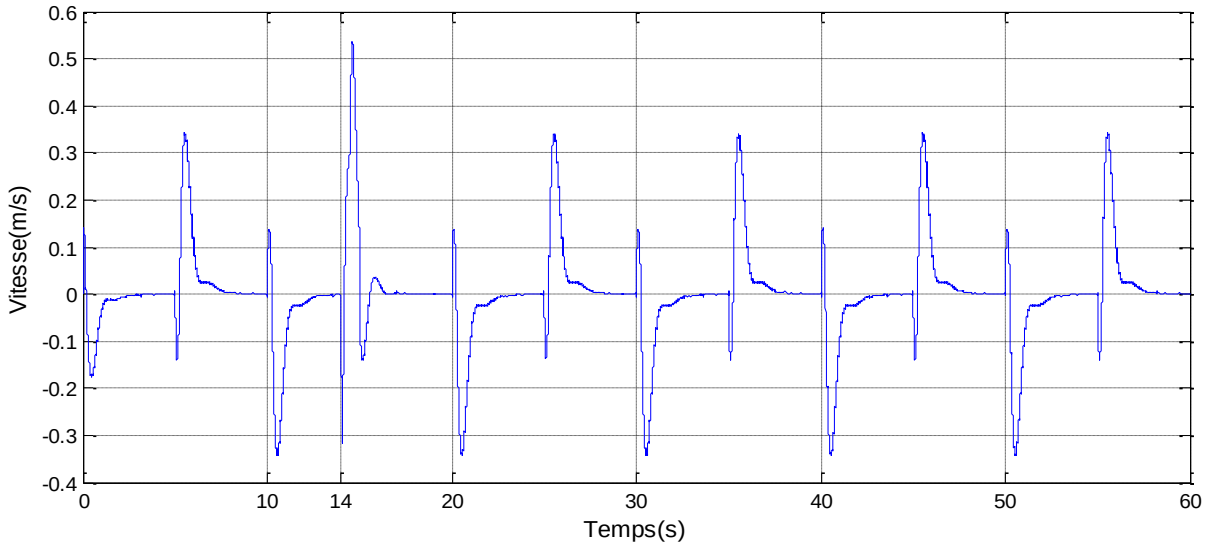


Figure 8.21. *Courbe de la vitesse linéaire en m/s.*

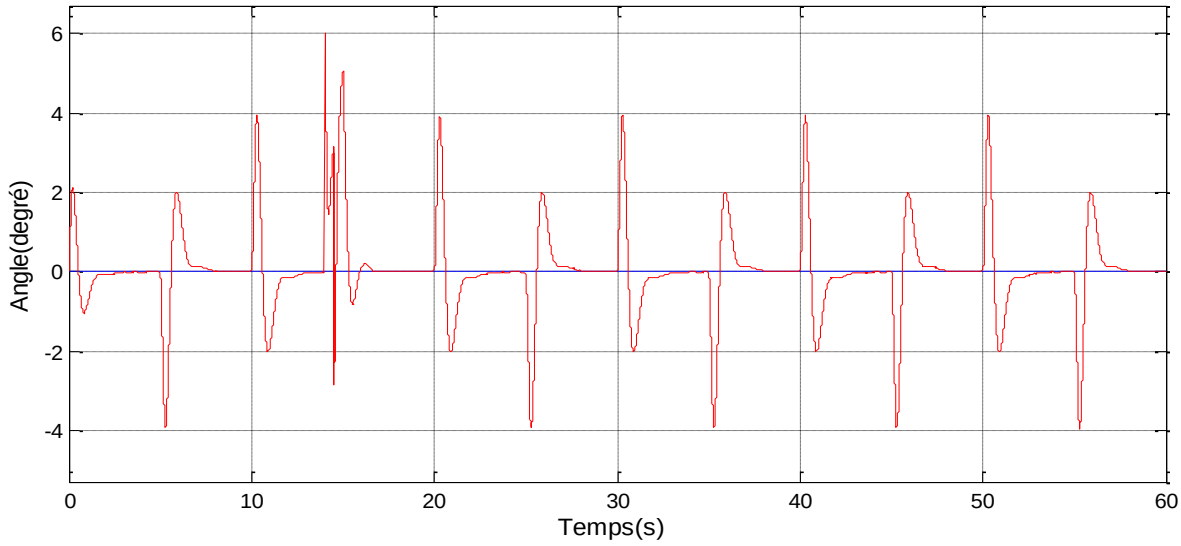


Figure 8.22. *Courbe d'angle de simulation en degré.*

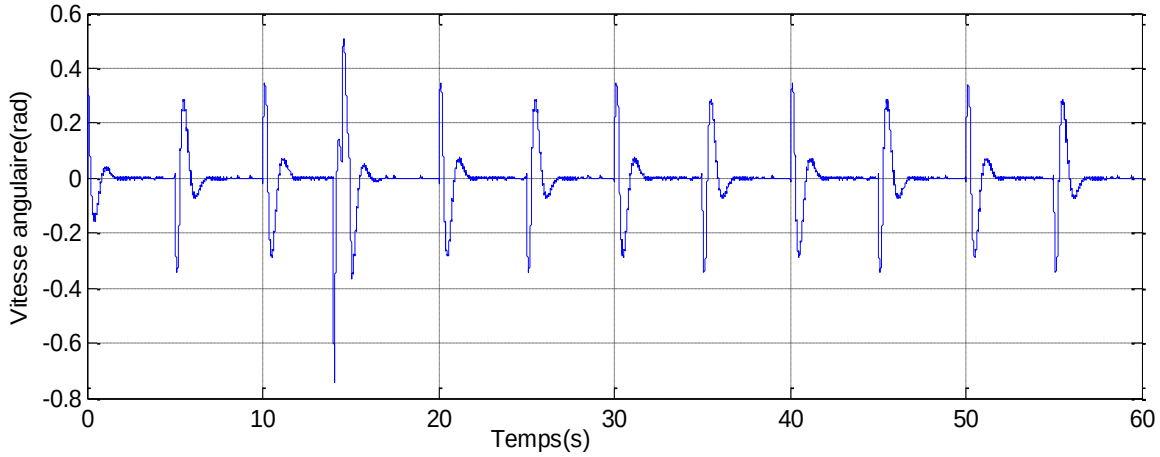


Figure 8.23. *Courbe de vitesse angulaire en radian/s.*

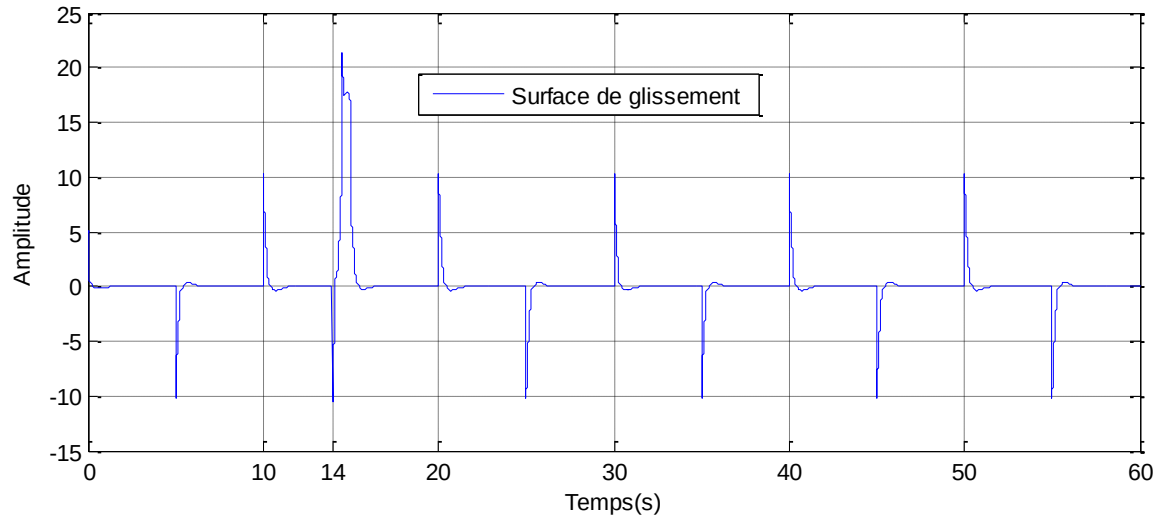


Figure 8.24. *Courbe de la 3eme surface de glissement.*

- **Interprétation des résultats:**
- La commande de la tension d'alimentation du moteur est acceptable et varie dans le régime nominal inferieur à 6v dans la plupart du temps.
- La position du chariot suit le signal de référence tout en maintenant le pendule stable dans la position verticale.
- Le temps de réponse est acceptable et la commande satisfait les contraintes de temps réel.
- La commande par mode de glissement incrémental est robuste envers les perturbations extérieures.

8.5.4. Commande robuste adaptative par Backstepping

8.5.4.1. Estimation des constantes de conception

Cette technique par backstepping comporte 8 constantes de conception arbitraires. Une partie des constantes concerne l'aspect d'adaptation tel que la vitesse et l'ampleur. L'autre partie concerne l'équilibre entre la régulation angulaire et la régulation longitudinale liées au backstepping. L'obtention des valeurs de chaque constante a été possible après peu d'essais en simulation numérique en utilisant le principe de l'essai et de l'erreur. Les valeurs obtenues sont définies comme suit :

$h_0 = 10, g_0 = 300, \gamma_1 = 9\ , \gamma_2 = 9, C_1 = 10, C_2 = 10\ , \lambda_1 = -0.4, \lambda_2 = -0.328.$

Les figures suivantes représentent les schémas Simulink réalisés :

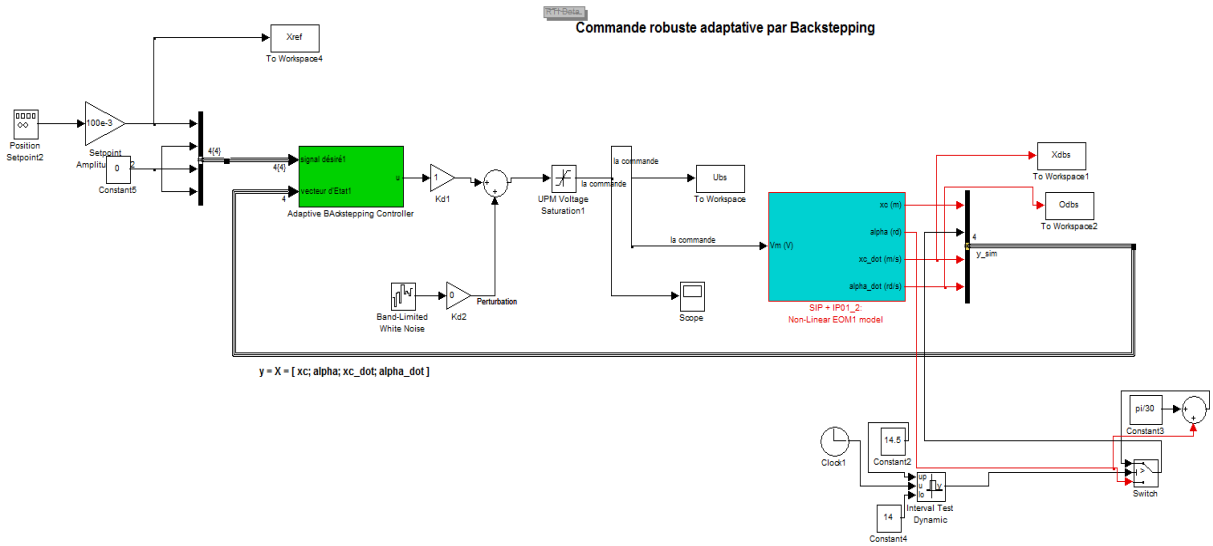


Figure 8.25. *Schéma Simulink global pour la simulation de la commande.*

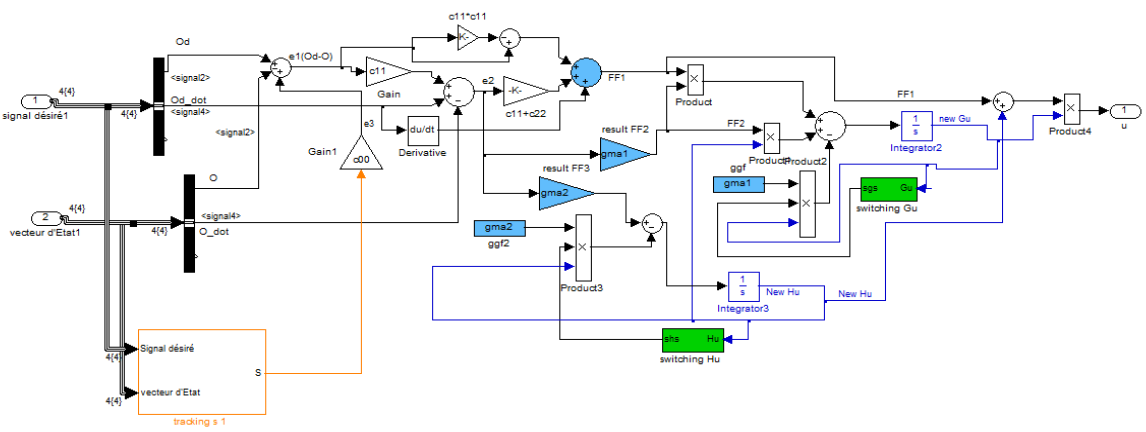


Figure 8.26. *Schéma Simulink de la loi de commande.*

8.5.4.2. Résultats de Simulation

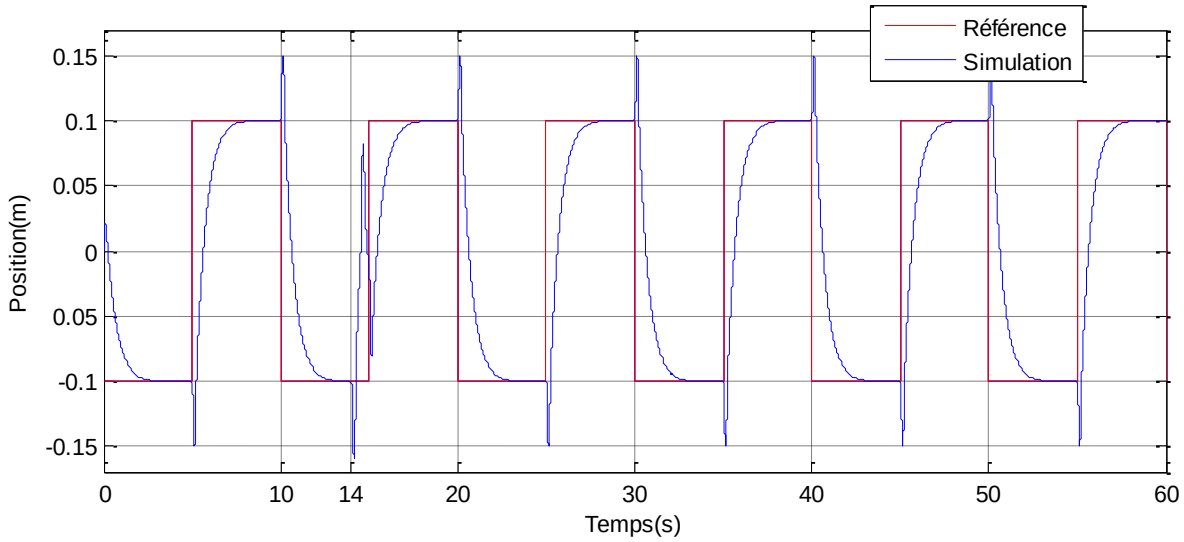


Figure 8.27. *Courbe de position de simulation en m.*

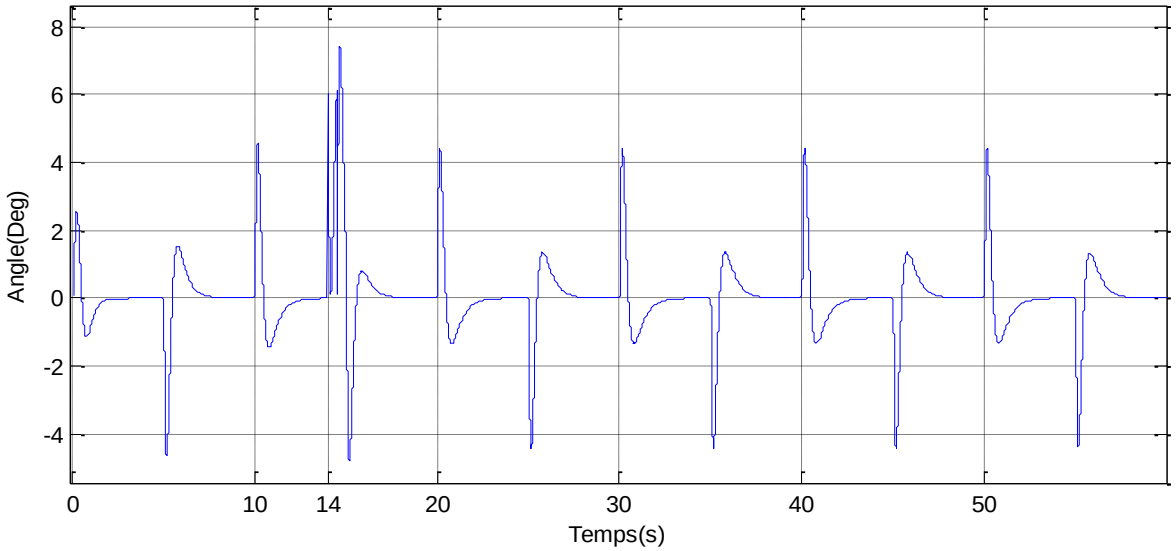


Figure 8.28. *Courbe d'angle de simulation en degré.*

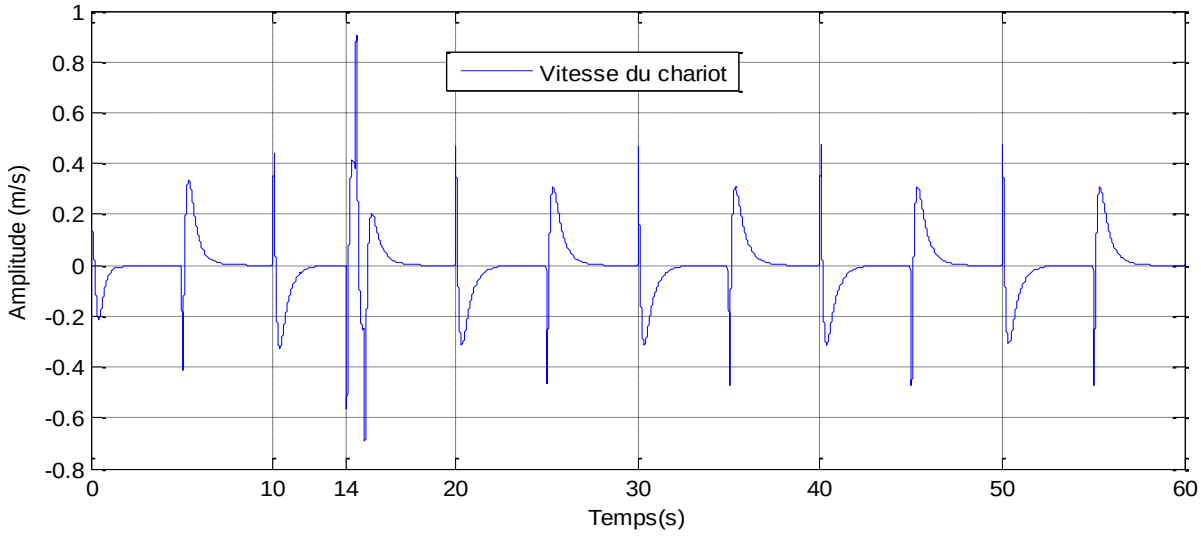


Figure 8.29. *Courbe de vitesse linéaire en m/s.*

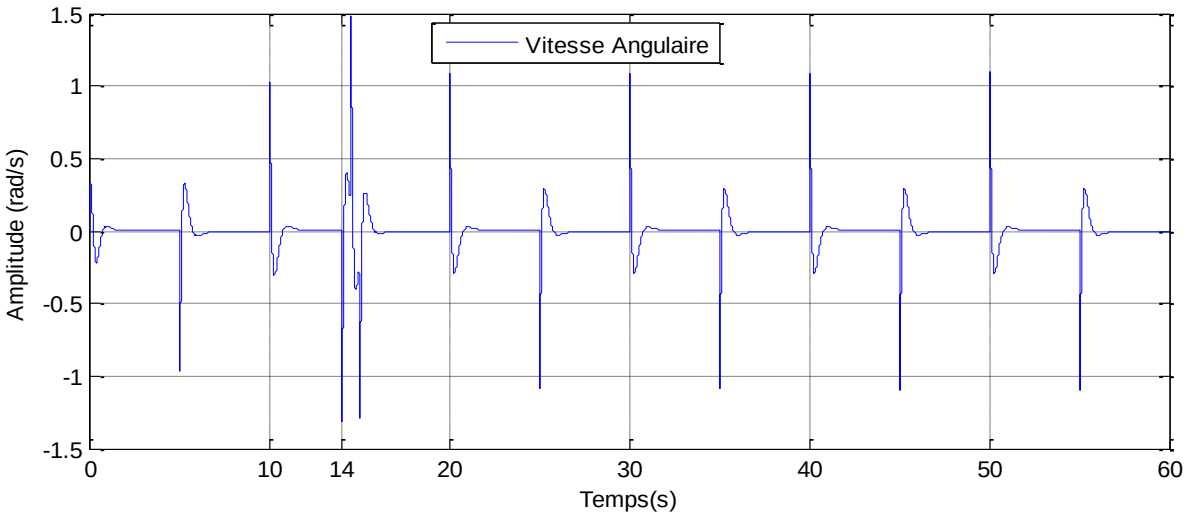


Figure 8.30. *Courbe de vitesse angulaire en radian/s.*

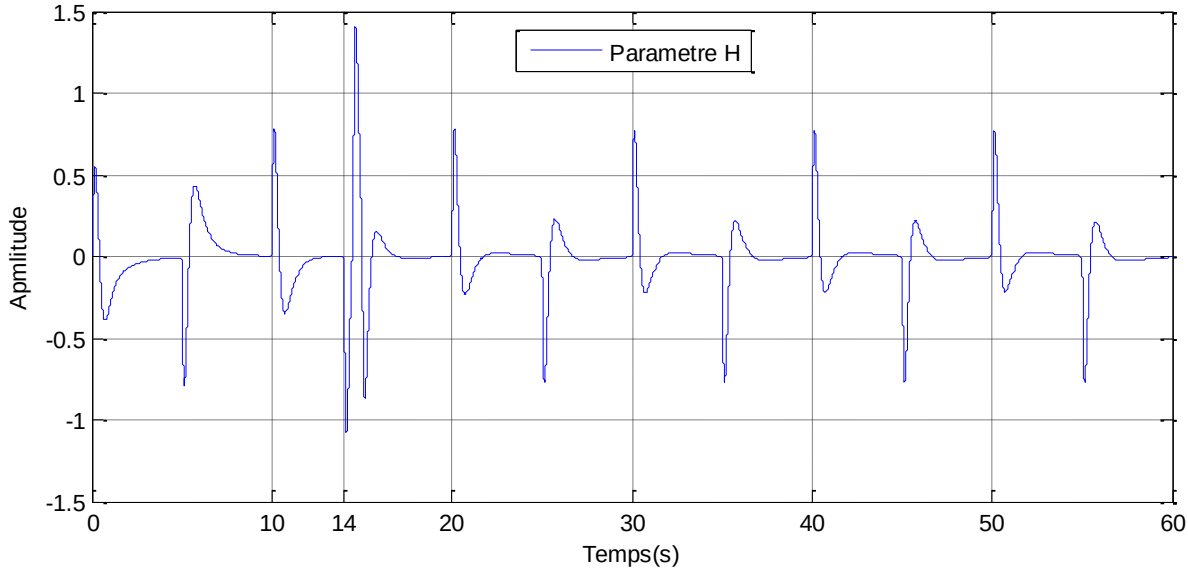


Figure 8.31. *Courbe d'évolution du paramètre d'adaptation h.*

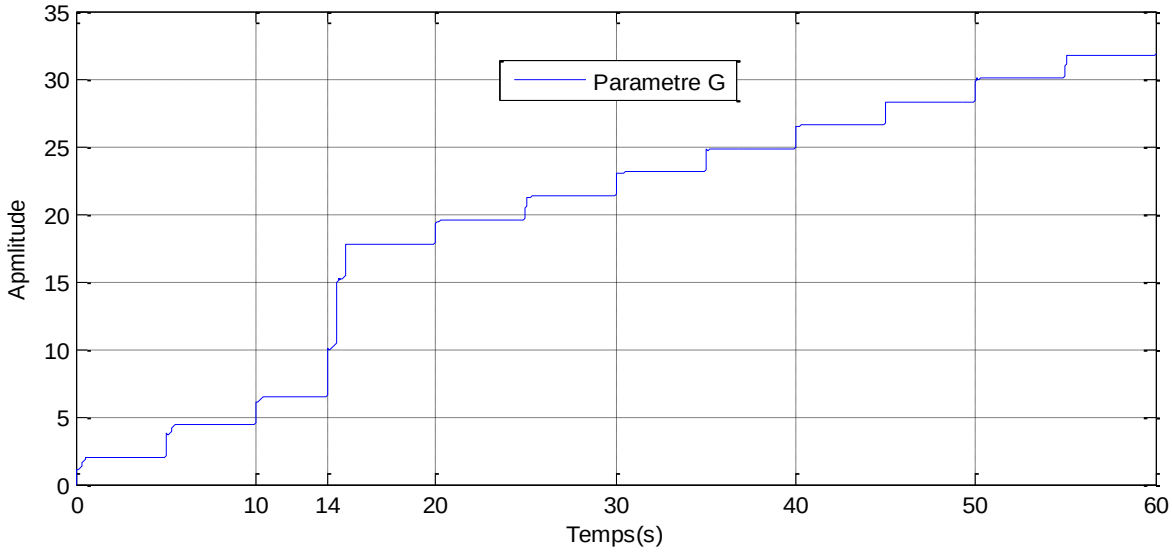


Figure 8.32. *Courbe d'évolution du paramètre d'adaptation g.*

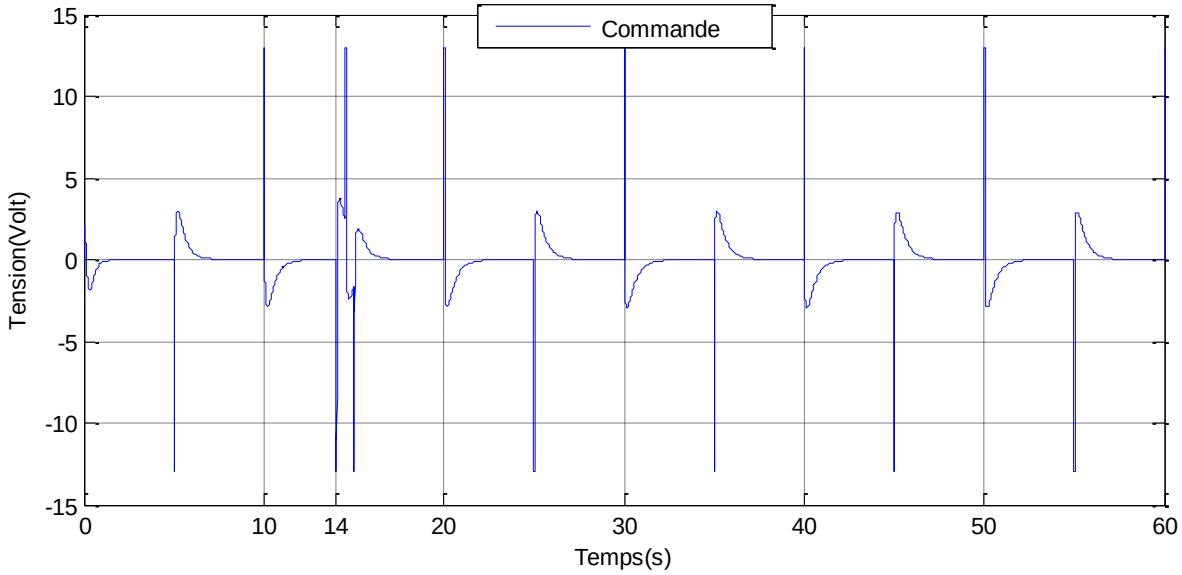


Figure 8.33. *Courbe de commande appliquée sur le moteur.*

- **Interprétation des résultats :**
- La commande de la tension d'alimentation du moteur est acceptable et varie dans le régime nominal inférieur à 6v dans la plupart du temps.
- La position du chariot suit le signal de référence tout en maintenant le pendule stable dans la position verticale.
- Le temps de réponse est acceptable et la commande satisfait les contraintes de temps réel.
- La commande est robuste envers les perturbations extérieures.

8.5.5. Commande adaptative par réseau de neurones à apprentissage en temps réel

8.5.5.1. Structure générale du réseau

Le choix de la structure adéquate du réseau de neurones qu'on envisage de concevoir, est déterminé par la capacité du réseau à apprendre les dynamiques du système de pendule inversé. Le théorème d'approximation universelle des réseaux de neurones [18] montre que ces derniers sont capables d'approximer n'importe quelle fonction non linéaire arbitraire en utilisant seulement une seule couche cachée. Sur cette base, on a opté pour la structure d'un perceptron multicouche avec une seule couche cachée. La configuration globale est choisie comme suit:

- Une couche d'entrée avec 2 neurones.
- Une couche cachée avec 6 neurones.
- Une couche de sortie avec 1 neurone.
- La fonction d'activation utilisée est la fonction sigmoïde sauf pour la couche de sortie ou on a utilisé la fonction linéaire.

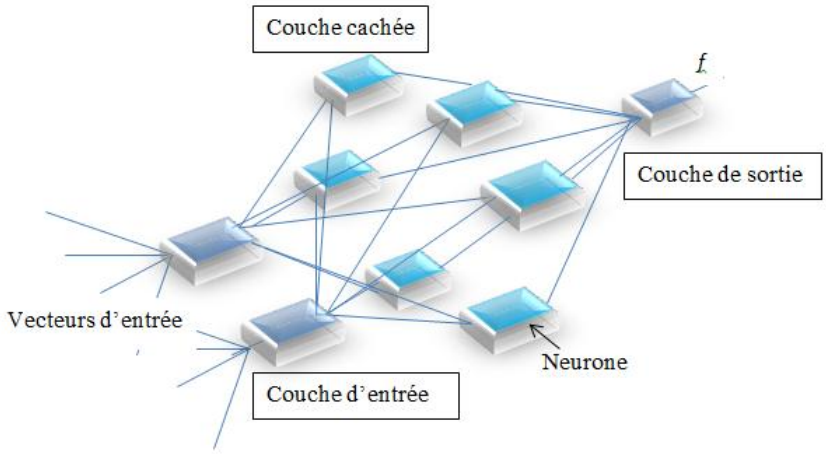


Figure 8.34. *Représentation graphique du réseau utilisé.*

8.5.5.2. Estimation des constantes de conception

La méthode de commande neuronale adaptative à apprentissage en temps réel (en ligne) comporte 3 constantes de conception arbitraires, la détermination de ses constantes se fait avant l'introduction de l'algorithme de rétro-propagation. Après plusieurs essais en simulation, on a obtenu les valeurs suivantes :

$\psi_x = 1.05, \psi_\theta = 3.5101, \lambda = 28.489$

8.5.5.3. Algorithme de rétro-propagation

L'algorithme de rétro-propagation est conçu et configuré par simulation numérique sur l'environnement de programmation Matlab. La conception se base sur la structure du réseau de neurones qu'on a choisi. L'organigramme suivant représente les différentes étapes de conception de l'algorithme de rétro-propagation.

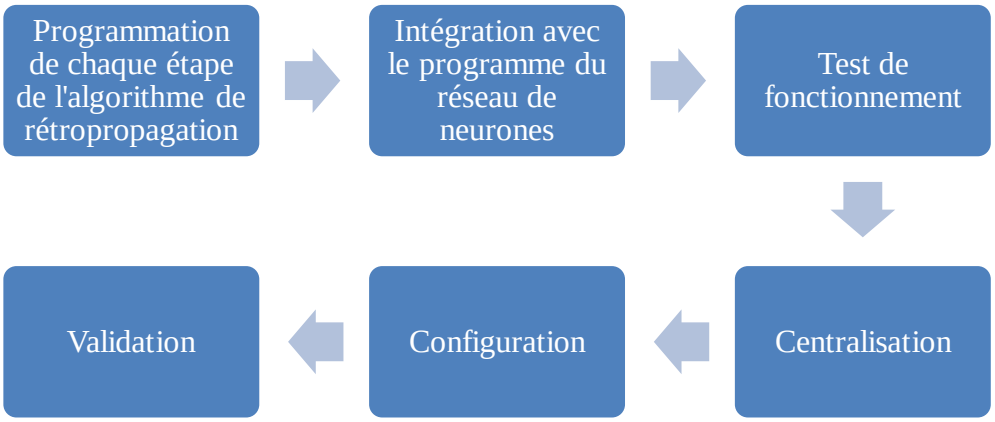


Figure 8.35. *Etapes de conception de l'algorithme de rétro-propagation.*

Afin de tester le fonctionnement et l'efficacité de l'algorithme, on a effectué un certain nombre de tests. Les figures suivantes montrent la capacité de l'algorithme à approximer n'importe quelle valeur dans l'intervalle étudié. Ainsi, la capacité d'apprentissage est prouvée par l'amélioration de la performance au fil du temps.

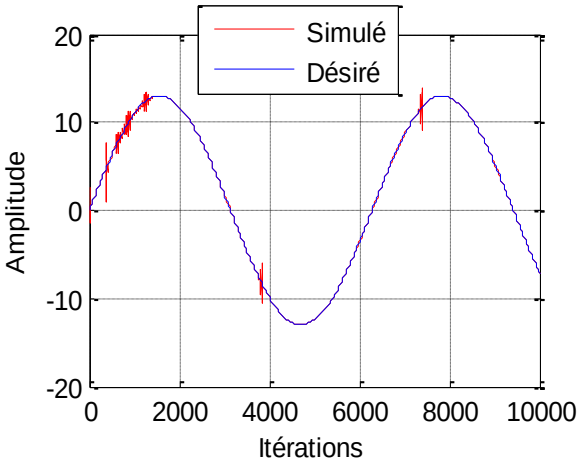


Figure 8.36. *Apprentissage après une époque (10000 itérations).*

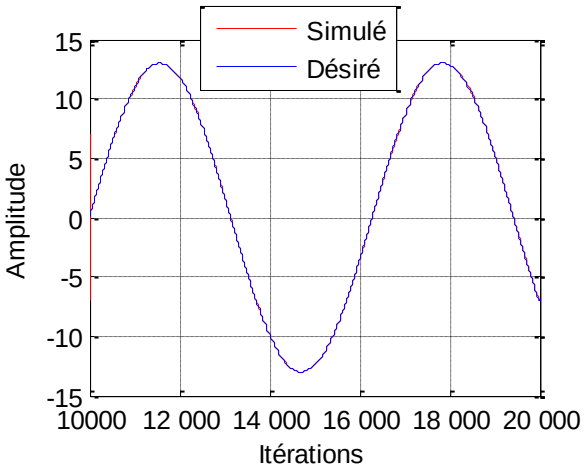


Figure 8.37. *Apprentissage après deux époques (20 000 itérations).*

Le signal d'erreur sinusoïdale est utilisé pour balayer l'intervalle ± 13 qui représente l'intervalle de variation de la tension de commande à appliquer sur le moteur.

8.5.5.4. Centralisation

L'algorithme de rétro-propagation doit satisfaire aux contraintes de temps réel. Le choix des poids du réseau de neurones est effectué arbitrairement. Cela implique qu'à l'état initial la sortie du réseau obtenue sera aussi arbitraire et peut atteindre des valeurs trop grandes ou trop petites. Une telle situation provoque un retard de convergence et par conséquent la contrainte de temps n'est pas satisfaite. Une centralisation est donc nécessaire pour résoudre le problème. Les poids à l'état initial sont ajustés de façon à placer la sortie du réseau de neurone au centre de l'intervalle de commande.

La figure suivante montre la convergence de la sortie vers zéro.

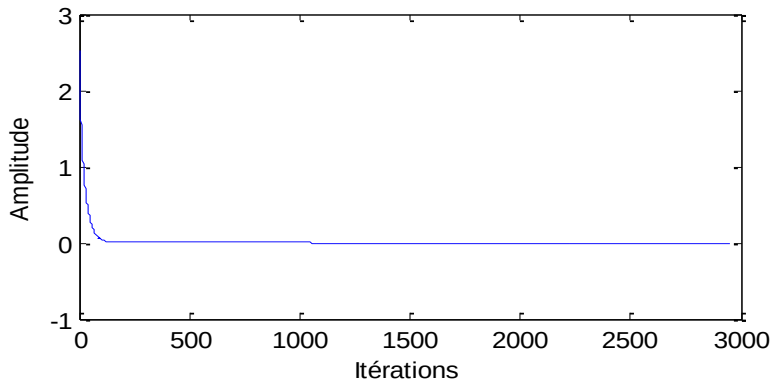


Figure 8.38. *Convergence de la sortie du réseau de neurones vers zéro.*

Les figures suivantes représentent les schémas Simulink réalisés, le pas d'apprentissage est fixé comme suit : $\mu = 0.7$.

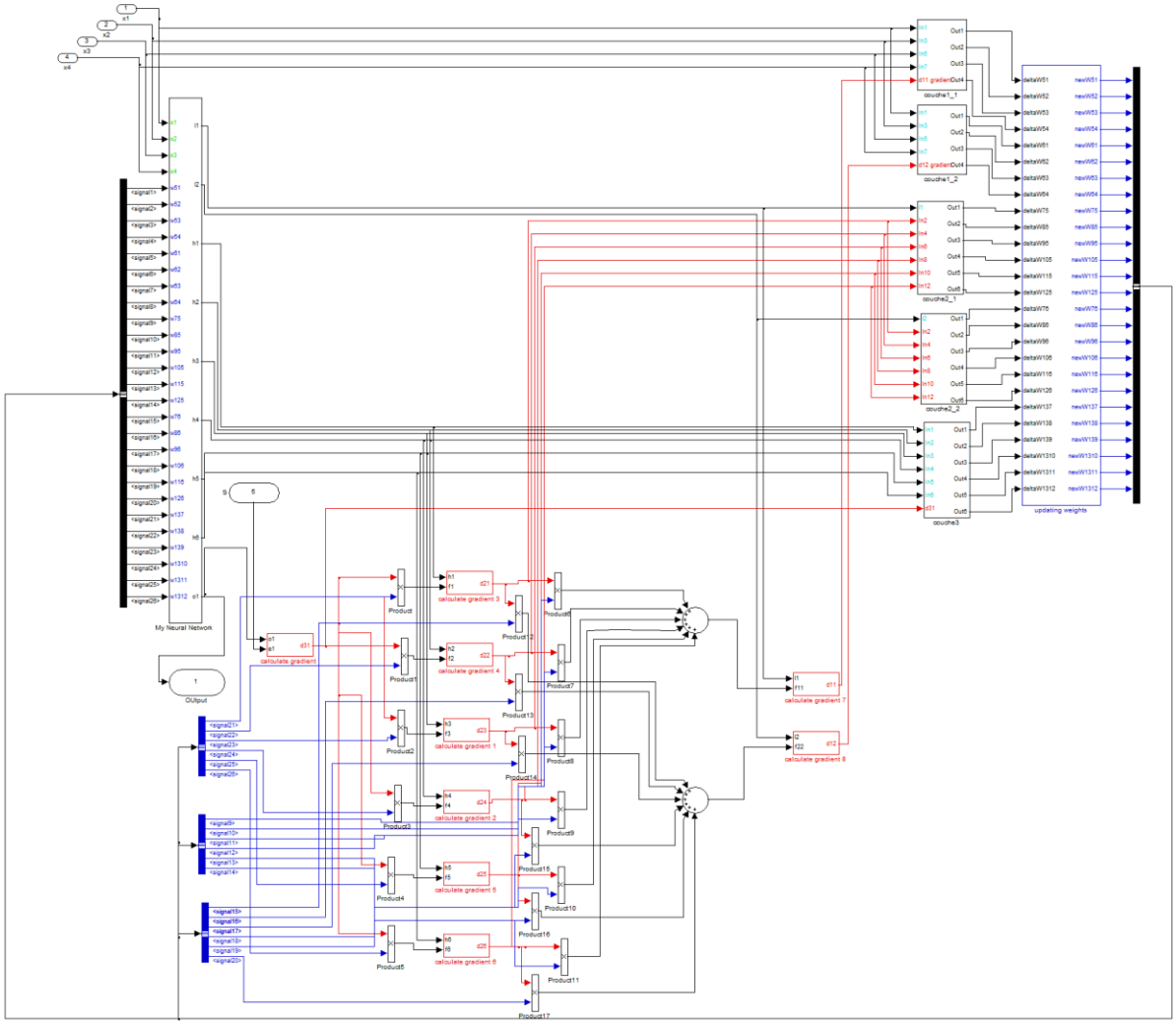


Figure 8.39. *Architecture interne du système de commande développé.*

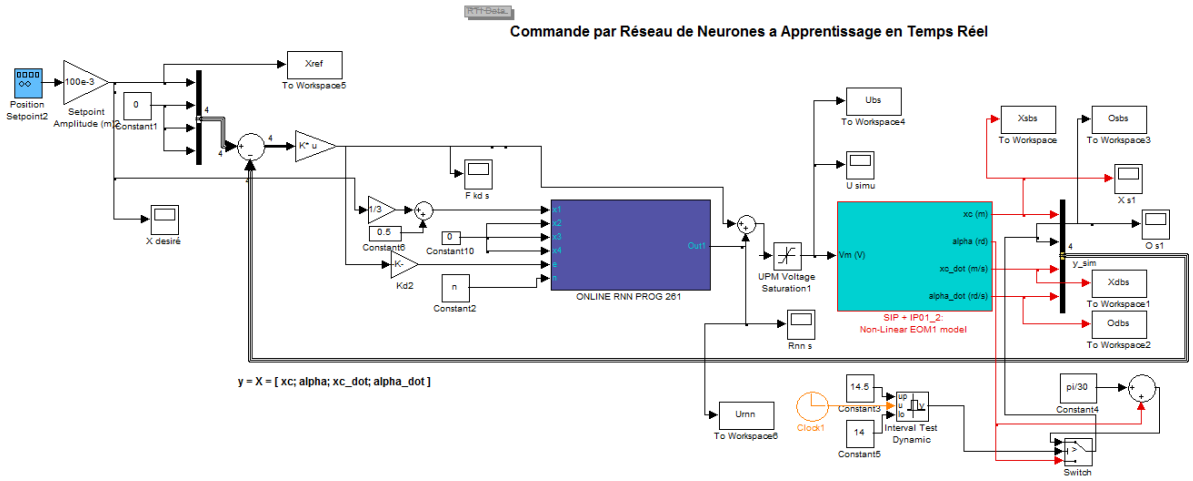


Figure 8.40. *Exemple de schéma Simulink global pour la simulation de la commande.*

8.5.5.5. Résultats de Simulation

Après des essais de simulation la configuration choisie est:ψ_x =1.05, ψ_θ =3.5101, λ =28.489.

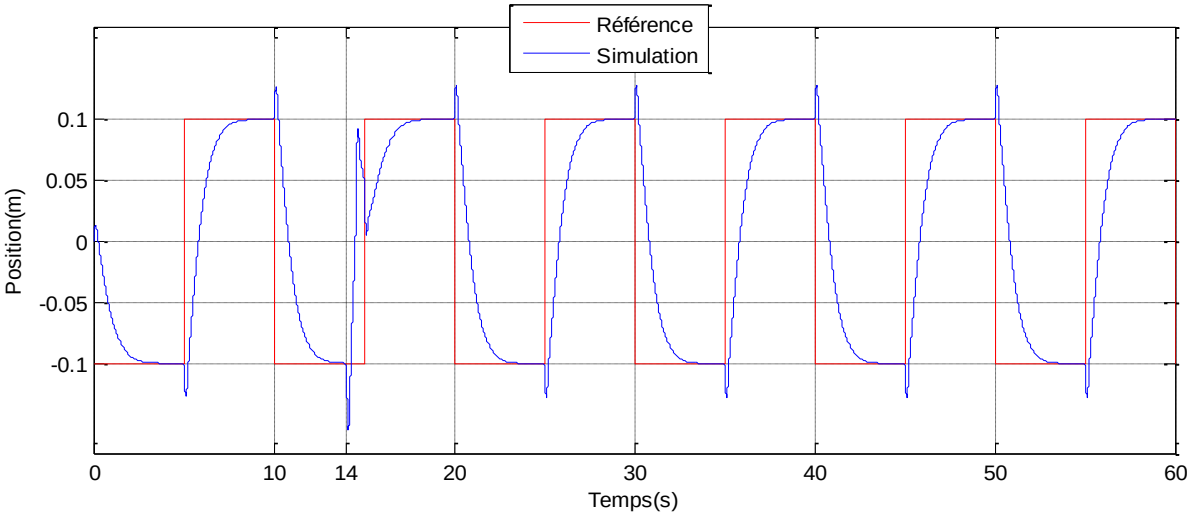


Figure 8.41. *Courbe de position de simulation en m.*

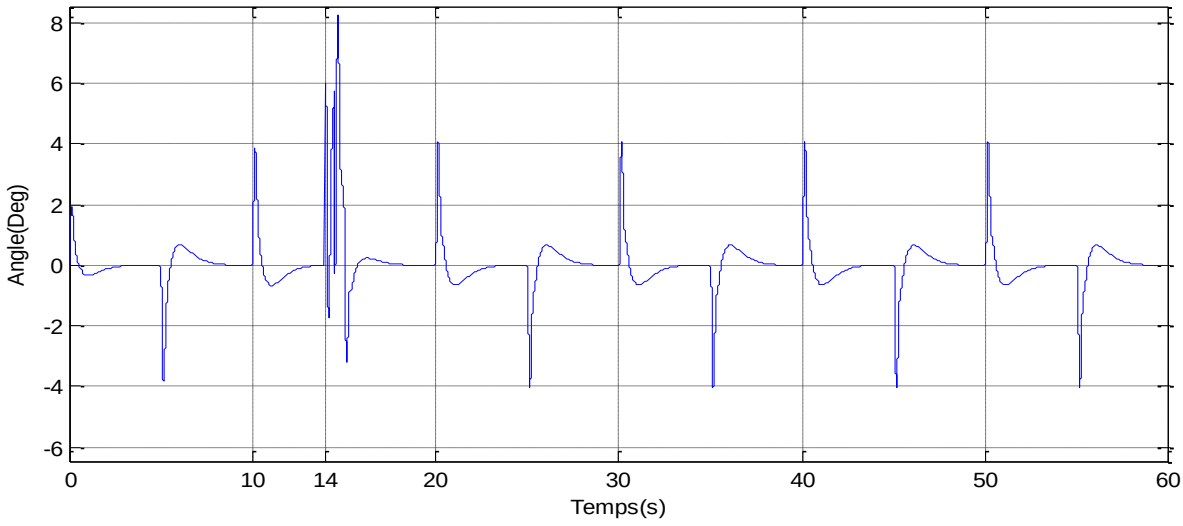


Figure 8.42. *Courbe d'angle de simulation en degré.*

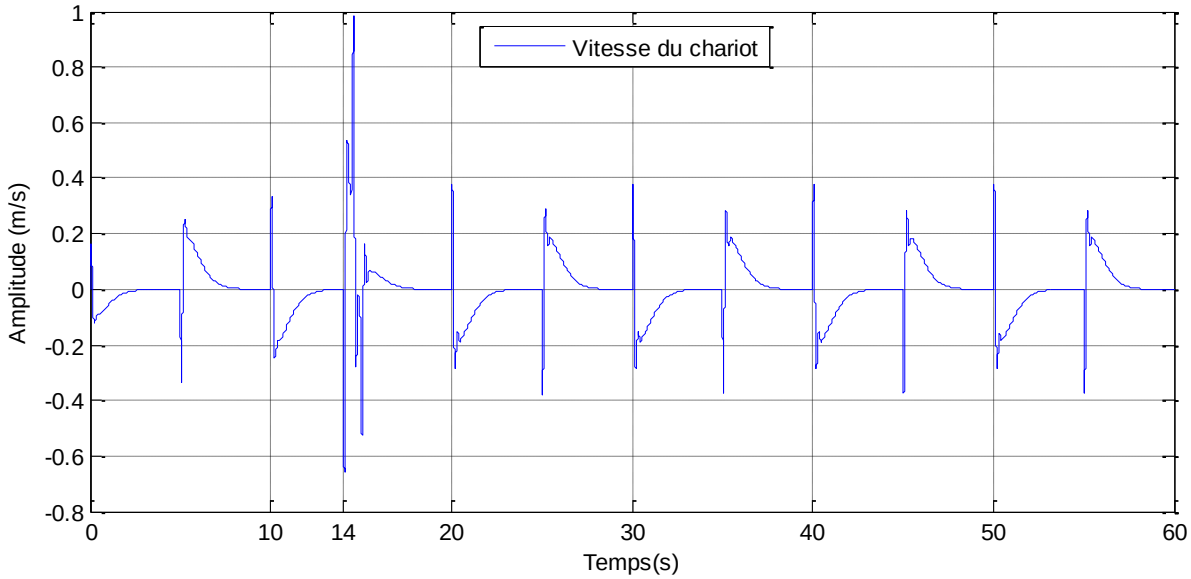


Figure 8.43. *Courbe de vitesse linéaire en m/s.*

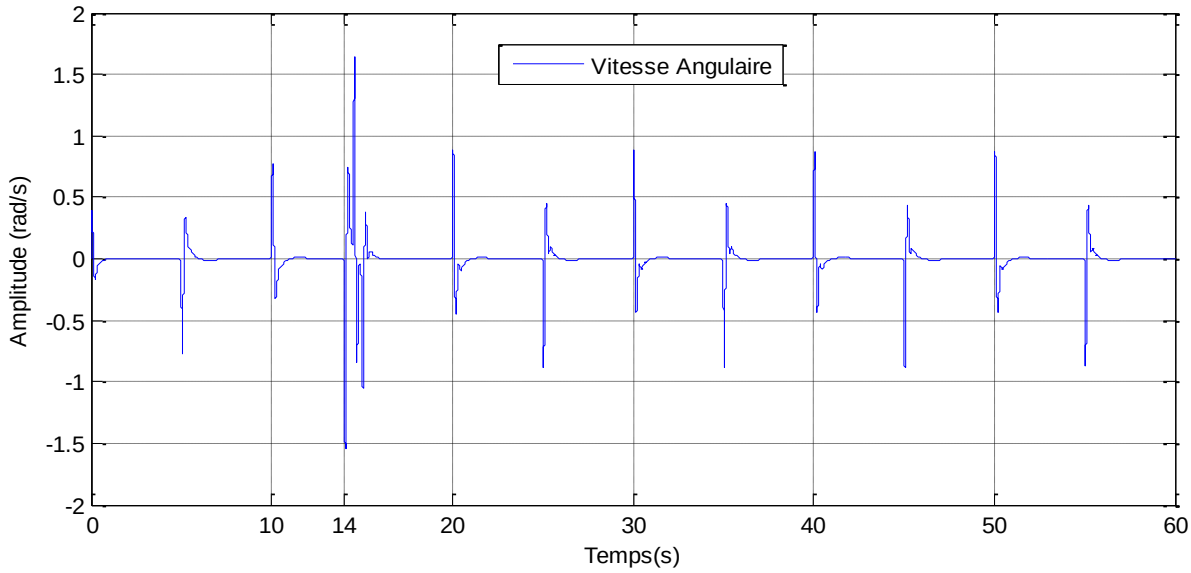


Figure 8.44. *Courbe de vitesse angulaire en radian/s.*

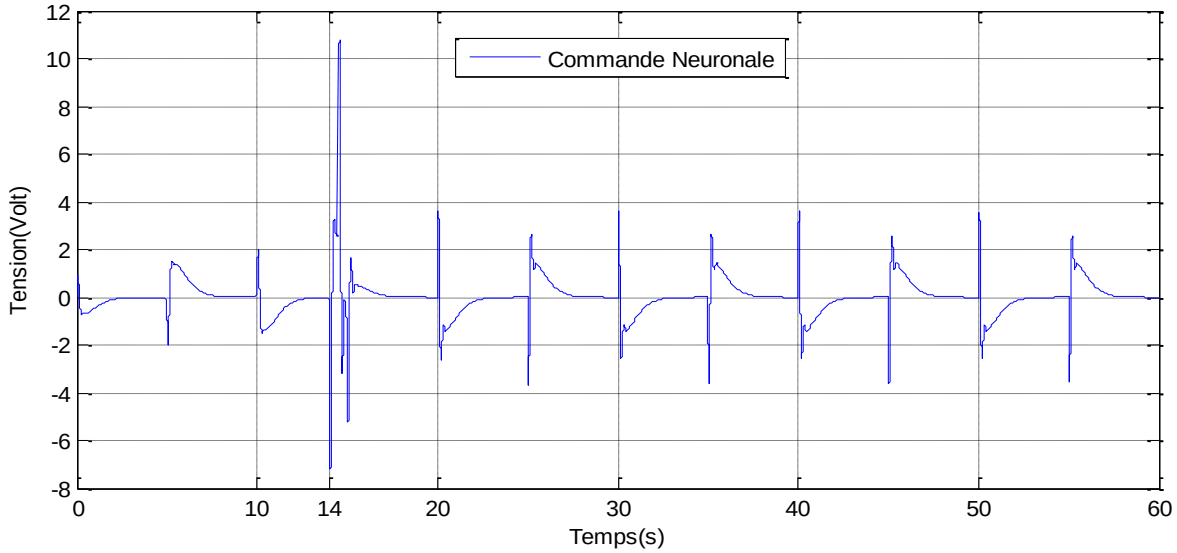


Figure 8.45. *Courbe de commande neuronale.*

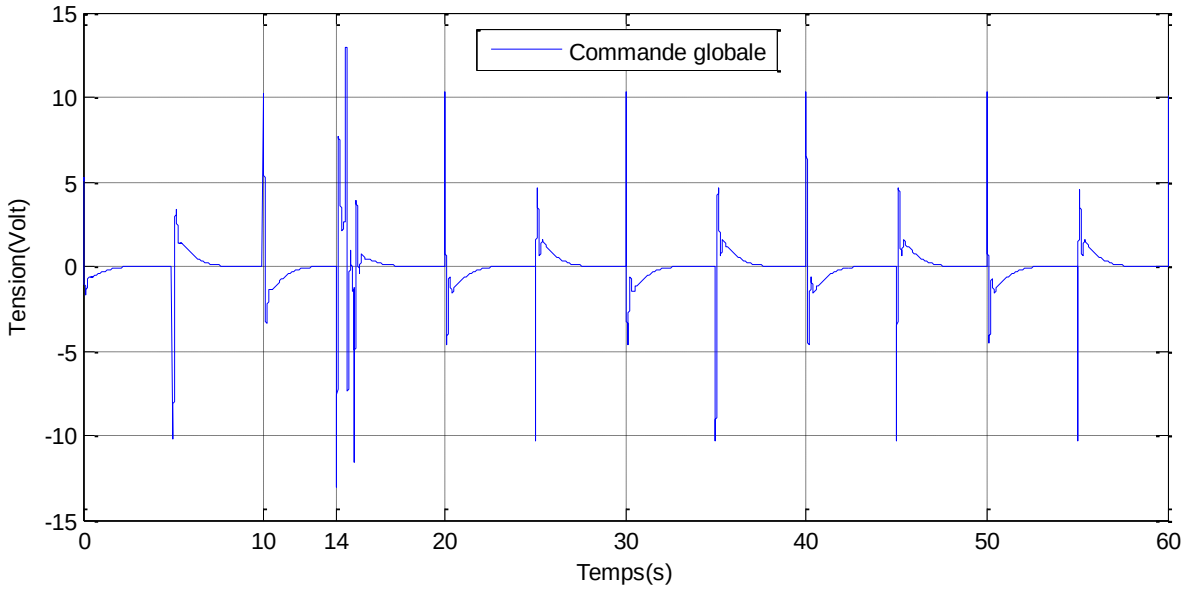


Figure 8.46. *Courbe de commande globale appliquée sur le moteur.*

- **Interprétation des résultats:**

- La commande de la tension d'alimentation du moteur est acceptable et varie dans le régime nominal dans la plupart du temps.
- La position du chariot suit très bien la position de référence tout en maintenant l'angle du pendule très proche de l'angle initial.
- Le temps de réponse est acceptable et satisfait aux contraintes de temps réel.
- La commande adaptative par réseau de neurones à apprentissage en temps réel est robuste envers les perturbations extérieures.

8.6. Validation expérimentale des résultats de simulation

On va, dans cette partie, valider les résultats obtenus en simulation sur un banc d'essai réalisé. Pour ce faire, on va appliquer les commandes élaborées précédemment sur le système réel en utilisant la carte de commande dSPACE1104 (Figure 8.47).

Figure 8.47. *Réalisation du banc d'essai.*

Le banc d'essai est composé de :

- Un PC contenant la carte dSPACE 1104 avec les logiciels Matlab & Simulink, ControlDesk et RTI.
- Un écran pour l'interface Homme/Machine développé sous ControlDesk.
- Le panneau de connexion CP1104.
- Un amplificateur de puissance (UPM).
- Le pendule inversé IP02.

Les modèles Simulink développés pour la partie expérimentale ont été conçus en utilisant les blocs élémentaires après plusieurs échecs d'implémentation en raison de la non reconnaissance des blocs complexes de la bibliothèque Simulink par la carte dSPACE1104.

8.6.1. Résultats pratiques

On place le chariot au centre de la crémaillère qui sera notre position 0 et le pendule vers le bas, puis, et avant de lancer la commande, on affecte une érection au pendule vers sa position verticale, et on lance la commande à travers l'IHM faite sous control Desk.

On a récupéré les résultats réels qui correspondent à la position du chariot et à l'angle du pendule, ainsi que les paramètres propres à chaque technique de commande au moyen de l'outil ControlDesk. Ces résultats sont transférés à Matlab à l'aide de la capture fournie par le logiciel et superposés aux résultats de simulation (figures suivantes).

8.6.1.1. Commande par le régulateur LQR

On a effectué des perturbations aux instant t=15.7s et t=30.6s et on a obtenu les résultats suivants :

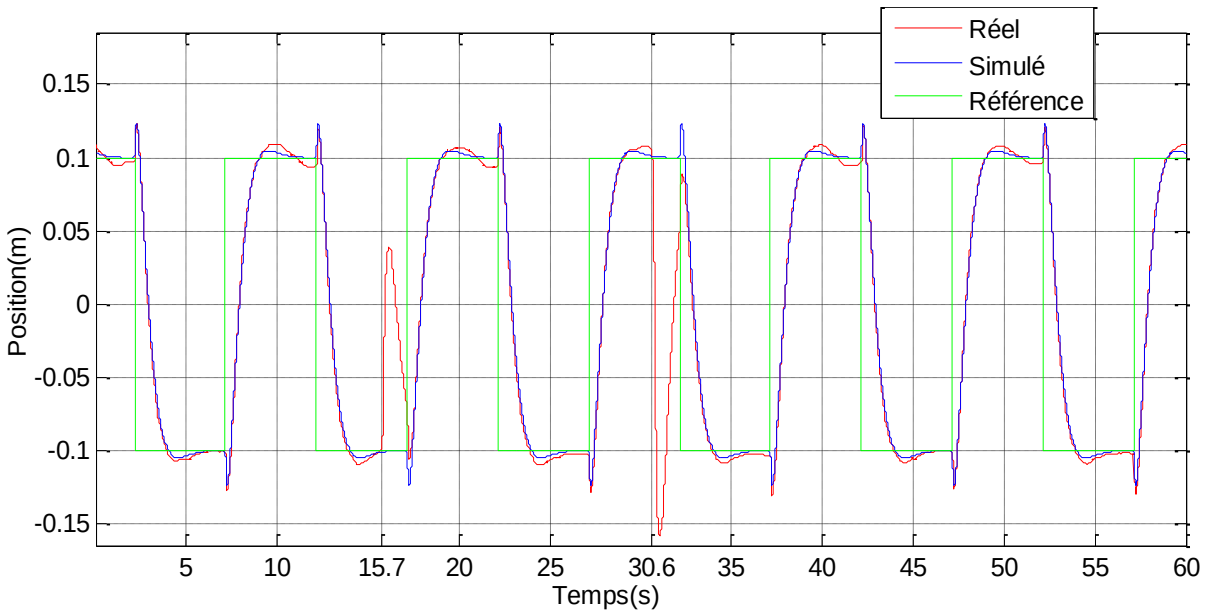


Figure 8.48. *Courbe de position en m.*

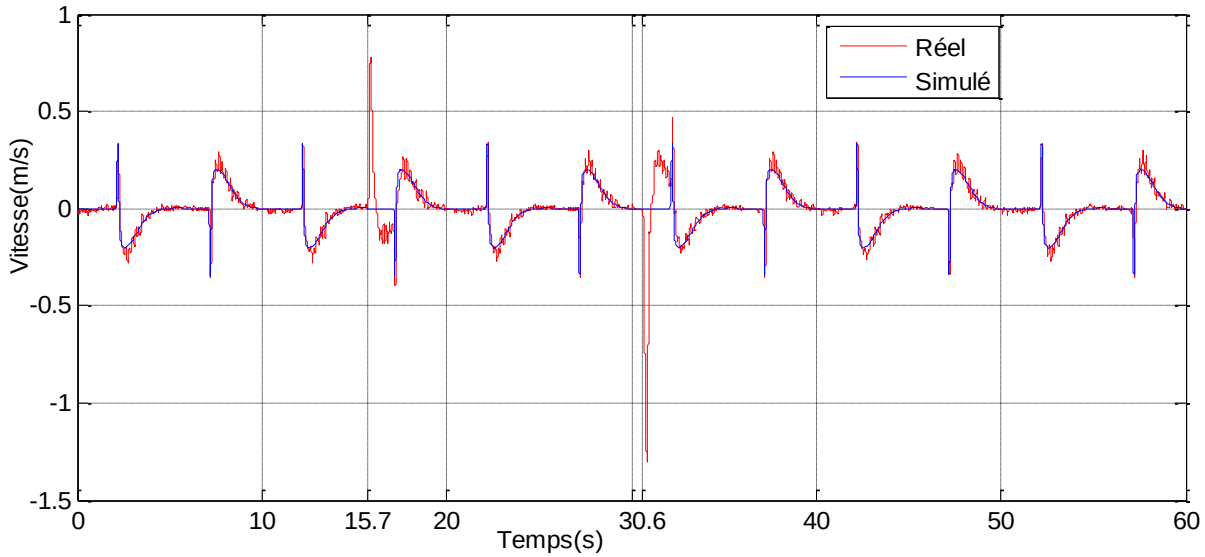


Figure 8.49. *Courbe de vitesse linéaire en m/s.*

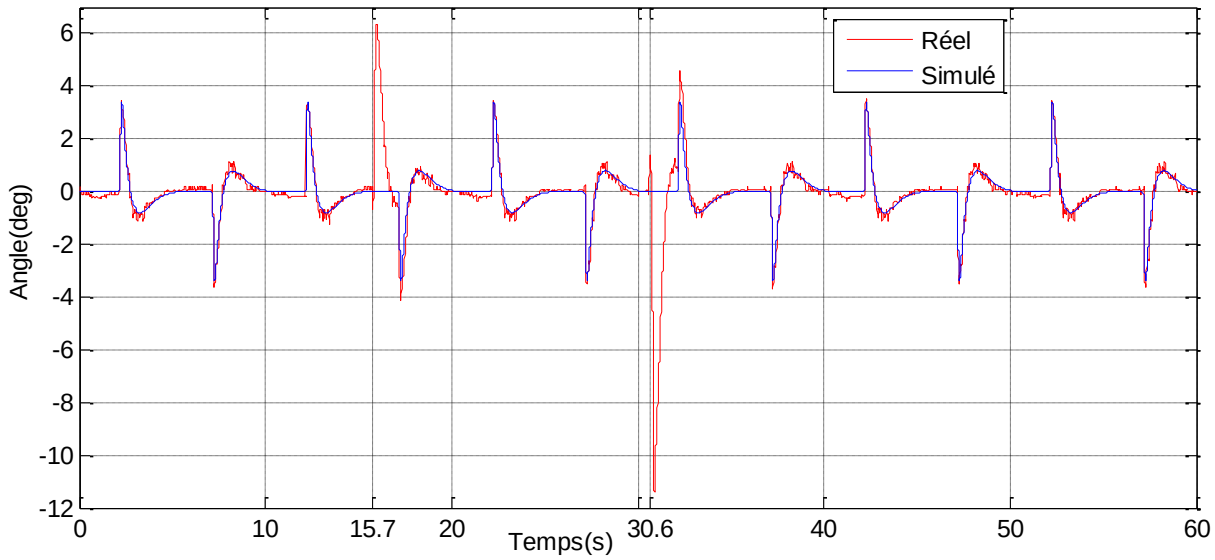


Figure 8.50. *Courbe d'angle en degré.*

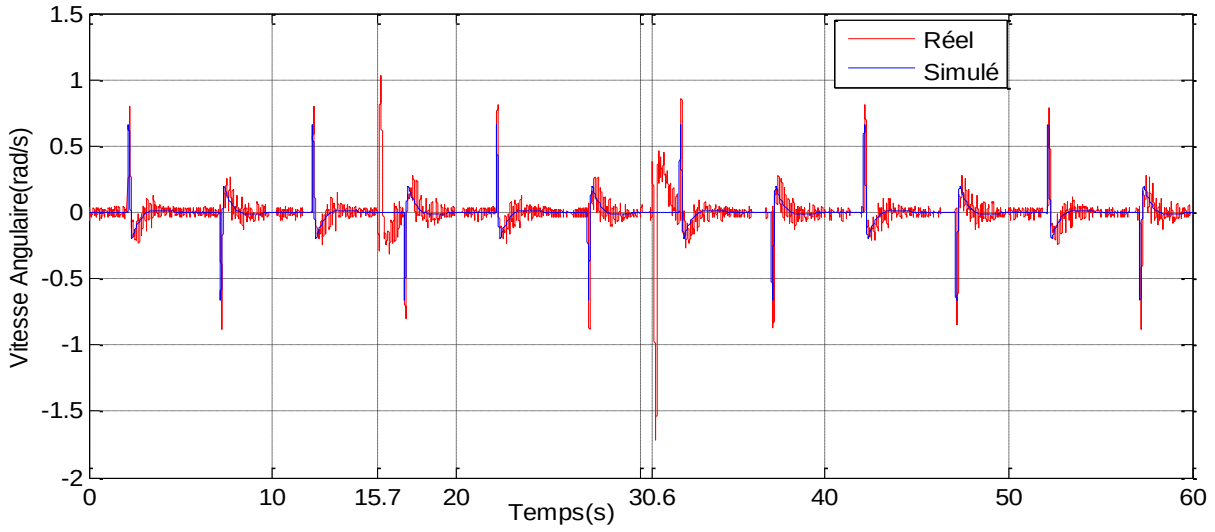


Figure 8.51. *Courbe de vitesse angulaire en radian/s.*

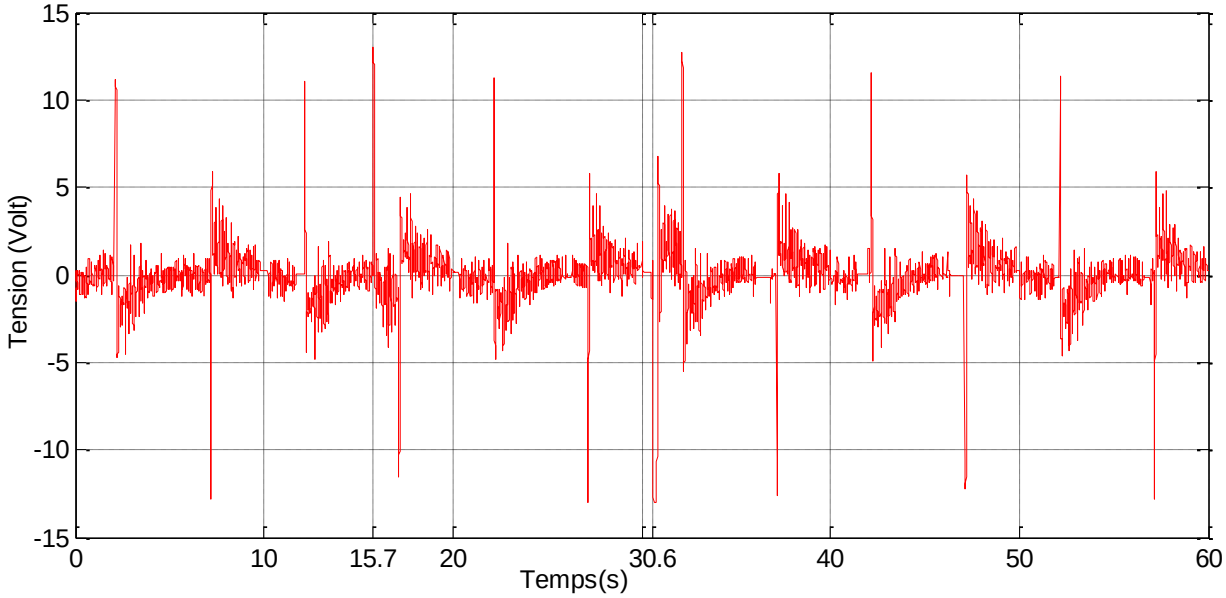


Figure 8.52. *Courbe de commande appliquée sur le moteur.*

- **Interprétation :**

- La commande de la tension d'alimentation du moteur est acceptable et varie dans le régime nominal dans la plupart du temps.
- La position du chariot est très proche de la position simulée avec quelques dépassements.
- Le pendule est stabilisé dans la position verticale.
- Le temps de réponse est acceptable et satisfait la contrainte du temps réel.
- La commande par régulateur LQR est robuste par rapport aux perturbations extérieures et réussit à réhabiliter « rétablir » le système dans des délais acceptables.
- La commande a réussi d'accomplir les tâches principales de commande.

8.6.1.2. Commande par réseaux de neurones à apprentissage hors ligne (RNA-Hors ligne)

L'implémentation des modèles Simulink contenant des blocs Neuronaux générés par la fonction « gensim » de Matlab n'est pas possible sur la carte dSPACE1104. Ceci est dû à la présence des blocs fermés qui ne sont pas reconnus par la carte. Pour cela le modèle Simulink utilisé dans la partie simulation (Figure 8.9) ne peut pas être ni implémenté sur la carte dSAPCE ni testé sur le banc d'essais.

Afin de résoudre le problème, on a envisagé la solution suivante :

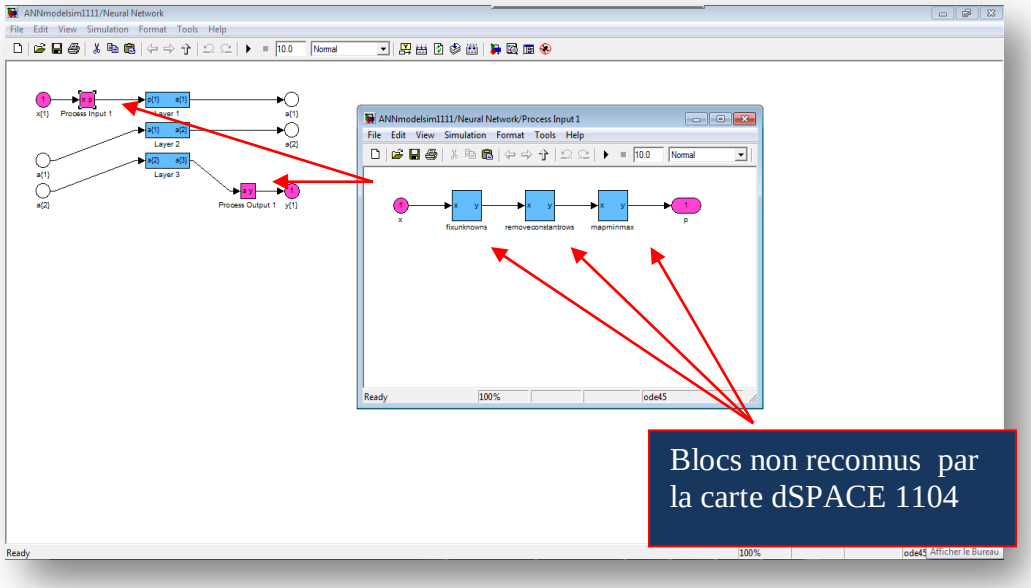


Figure 8.53. *Fenêtres du schéma interne du réseau de neurones créé par « gensim ».*

Les blocs non reconnus sont des blocs fermés qui ne sont pas accessibles par Simulink, le rôle de ces blocs d'optimisation est le blocage des vecteurs d'entrée constants et invariants par rapport au temps, ainsi que la normalisation.

Pour résoudre le problème on remplace les 3 blocs non reconnus par une fonction équivalente (figure 8.54) pour chaque entrée constituant le vecteur d'entrée et pour chaque sortie du RNA.

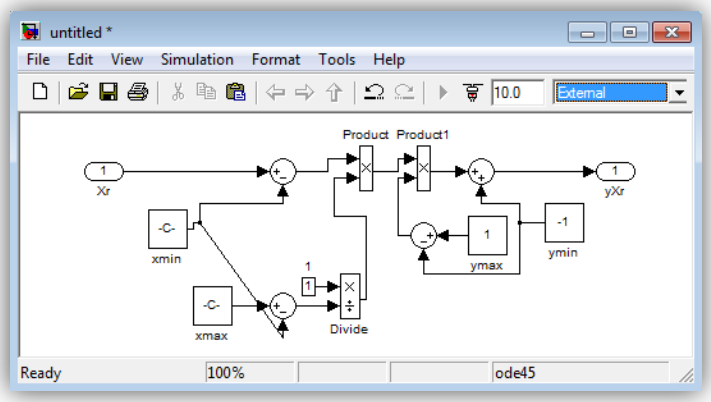


Figure 8.54. *Exemple de fonctions équivalentes aux blocs de « gensim».*

Les figures suivantes regroupent les résultats obtenus, des perturbations sont effectuées aux instant t=37s et t=41s.

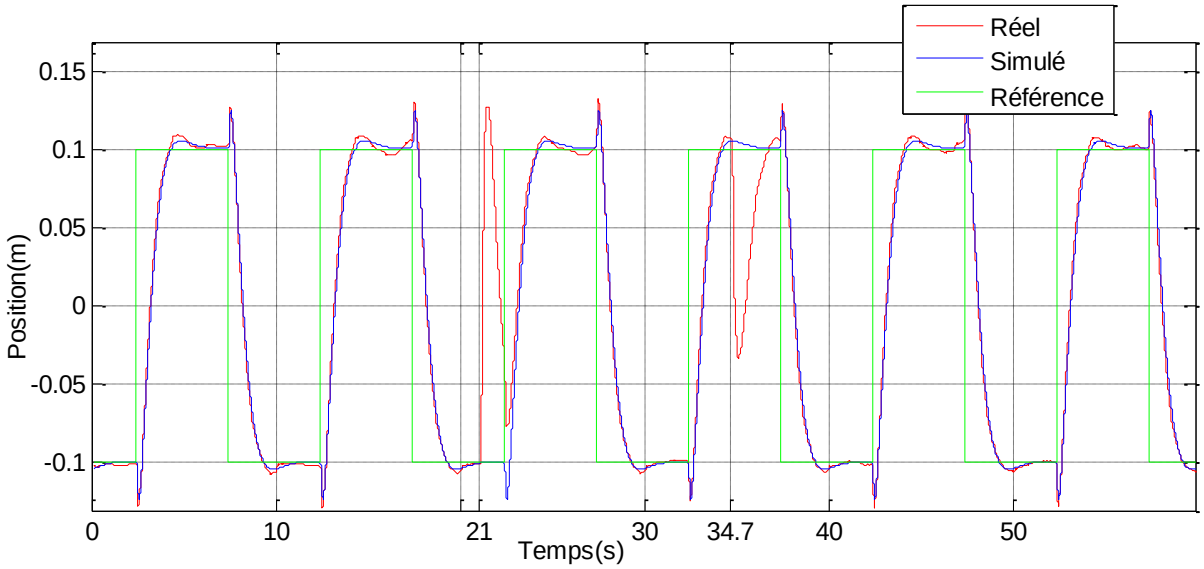


Figure 8.55. *Courbe de position linéaire en m.*

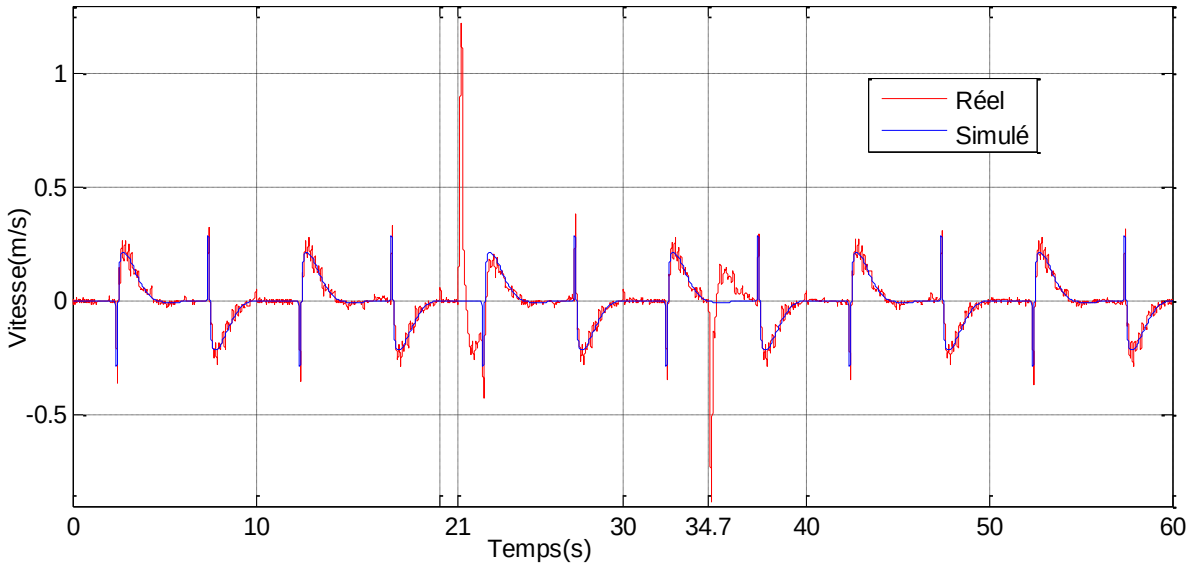


Figure 8.56. *Courbe de vitesse linéaire en m/s.*

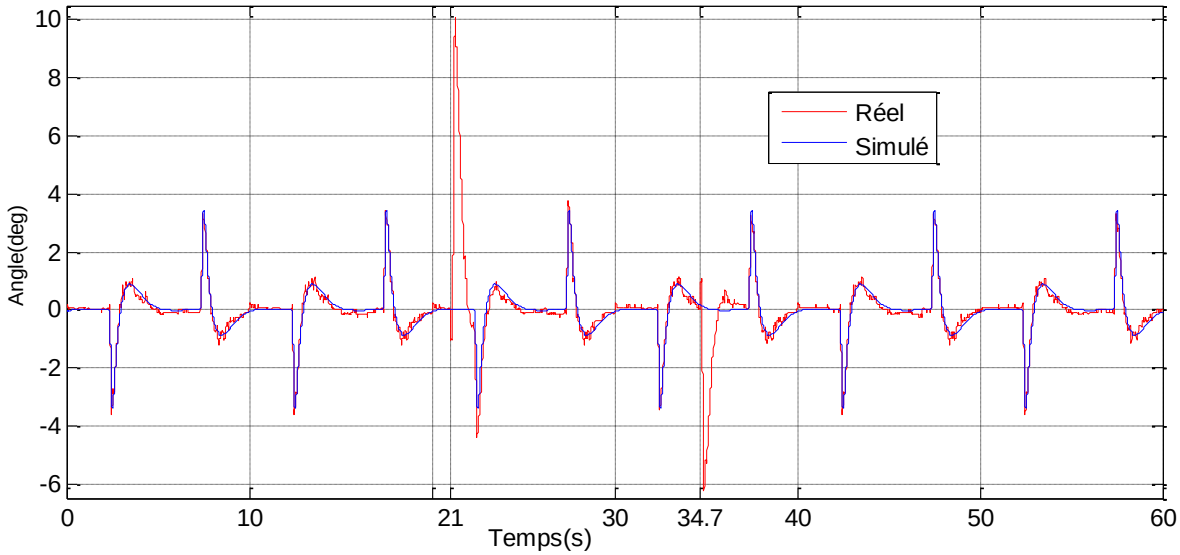


Figure 8.57. *Courbe de l'angle en degré.*

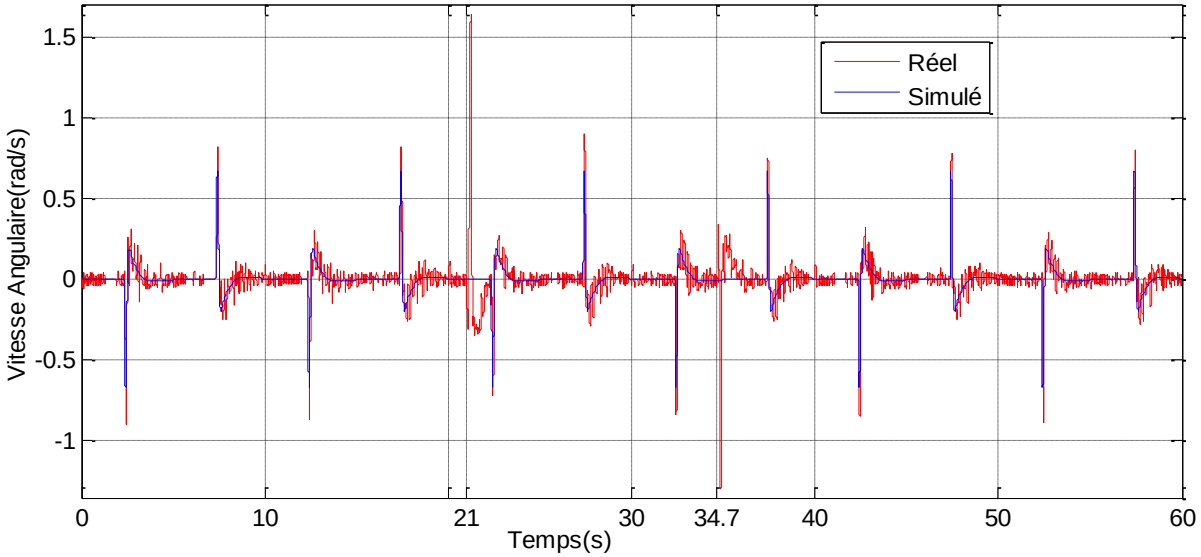


Figure 8.58. *Courbe de vitesse angulaire en radian/s.*

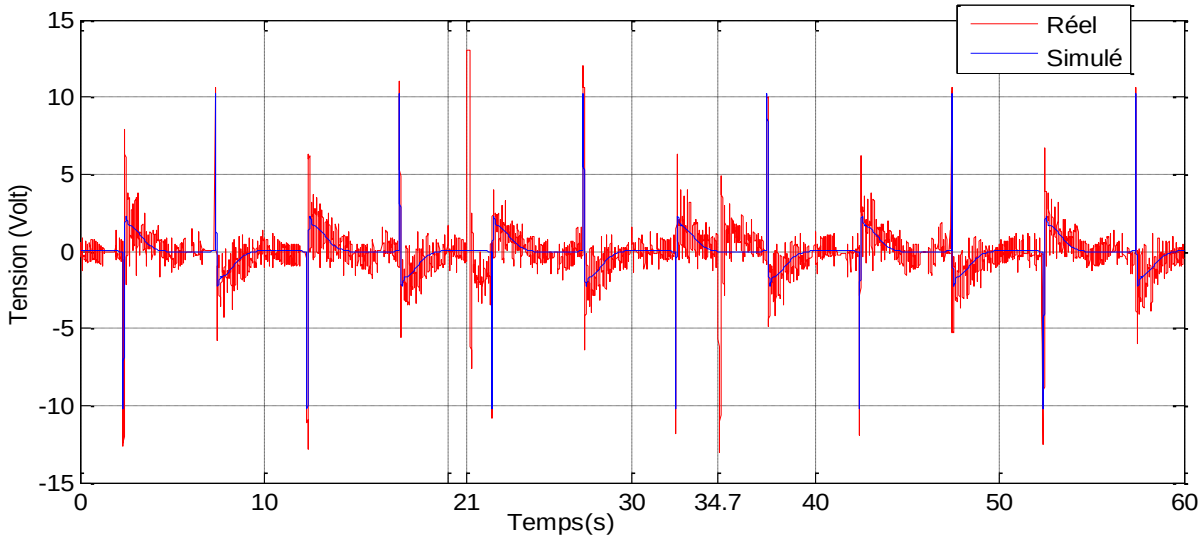


Figure 8.59. *Courbe de commande appliquée sur le moteur.*

- **Interprétation :**

- La commande de la tension d'alimentation du moteur est acceptable et varie dans le régime nominal dans la plupart du temps.
- La position du chariot est très proche de la position simulée avec peu de dépassement.
- Le temps de réponse est acceptable et satisfait les contraintes du temps réel.
- La commande par réseau de neurones à apprentissage hors ligne a réussi à rejeter les perturbations extérieures et à réhabiliter le système dans des délais acceptables.
- La commande a réussi à accomplir les tâches principales désirées.

8.6.1.3. Commande par mode de glissement incrémental (ISMC)

Résultats : On a effectué les Perturbations aux instants t=19.2s, t=26.6s et t=47.7

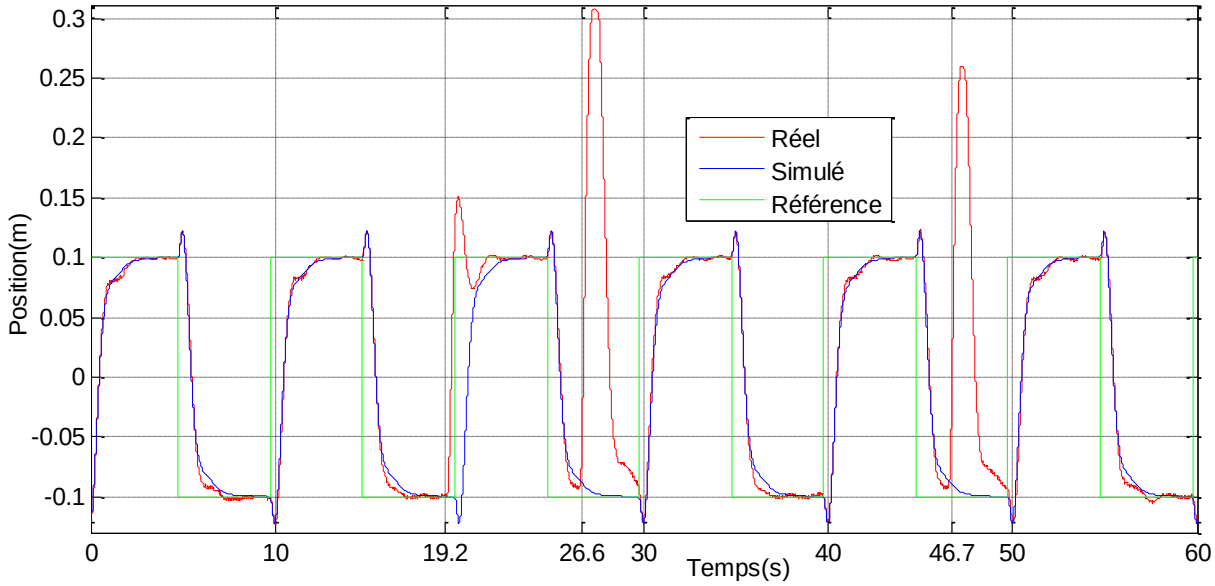


Figure 8.60. *Courbe de position en m.*

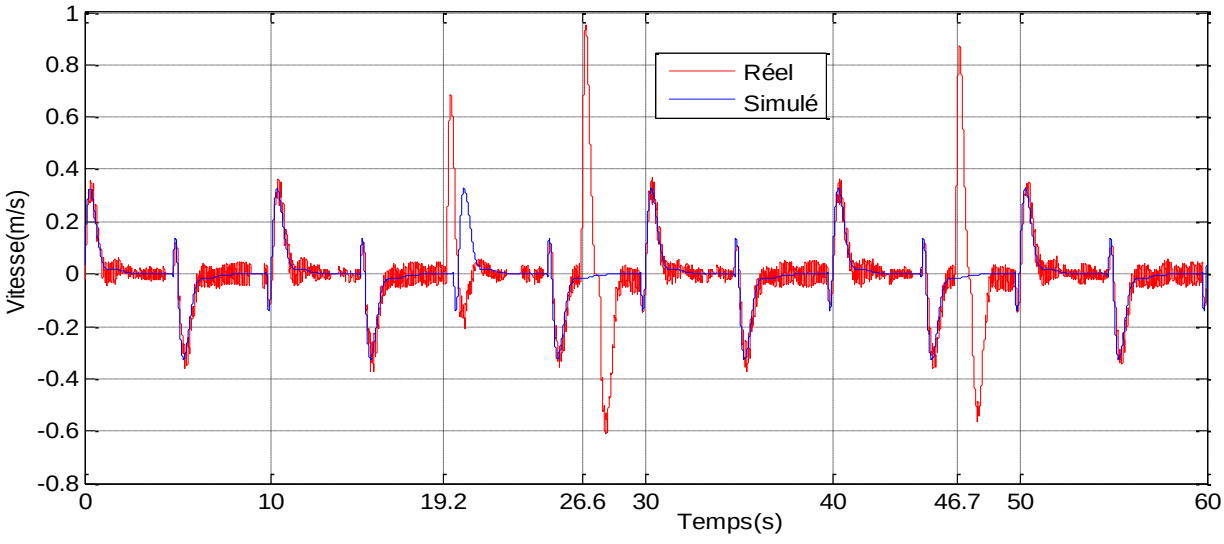


Figure 8.61. *Courbe de vitesse linéaire en m/s.*

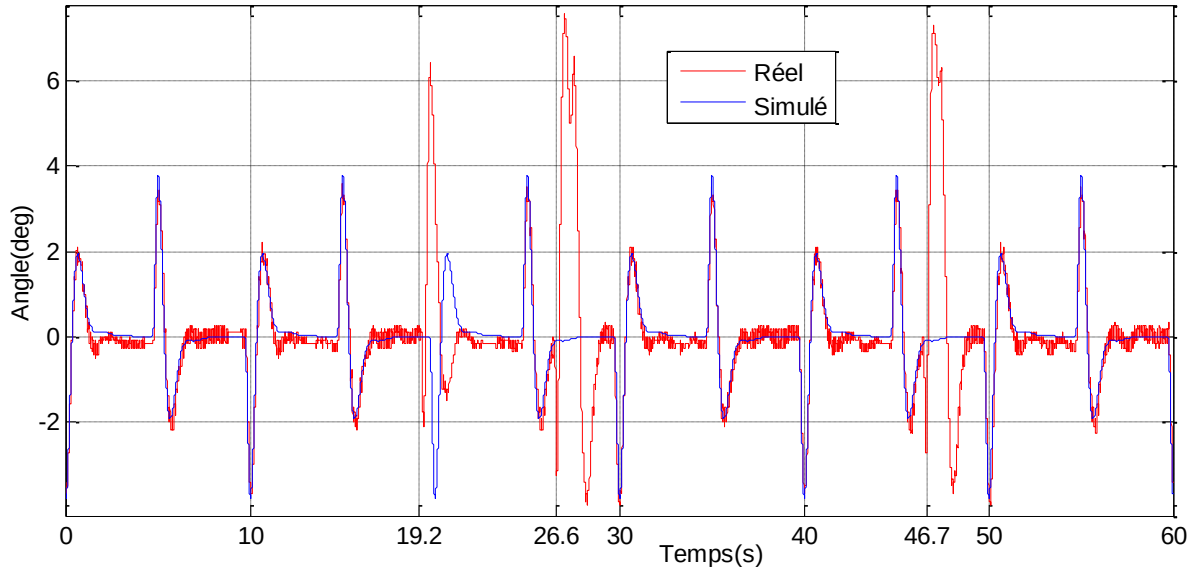


Figure 8.62. *Courbe d'angle en degré.*

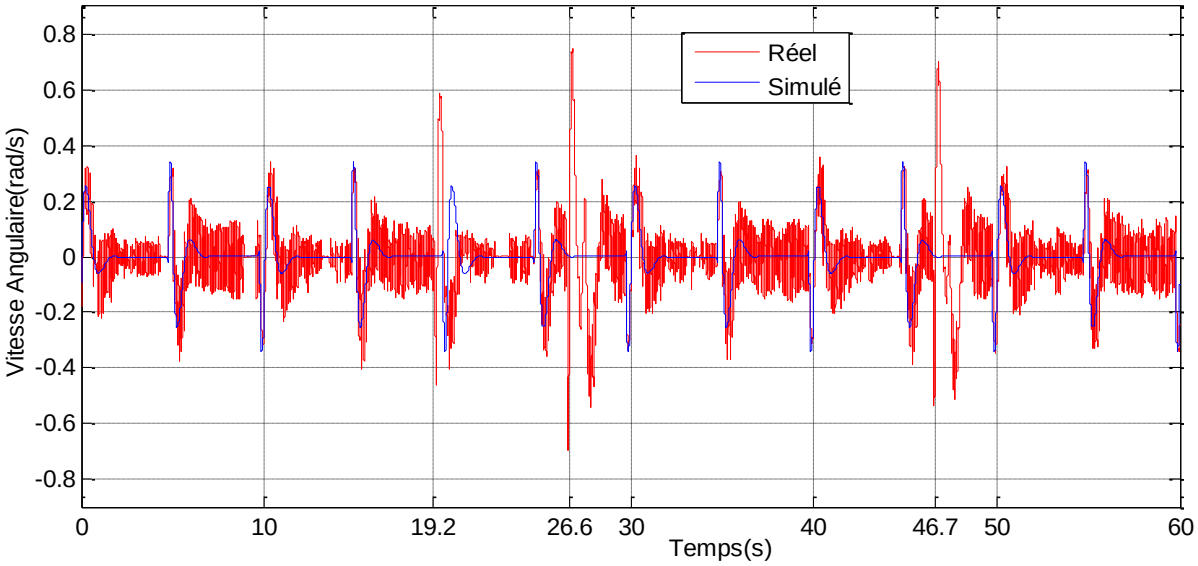


Figure 8.63. *Courbe de vitesse angulaire en radian/s.*

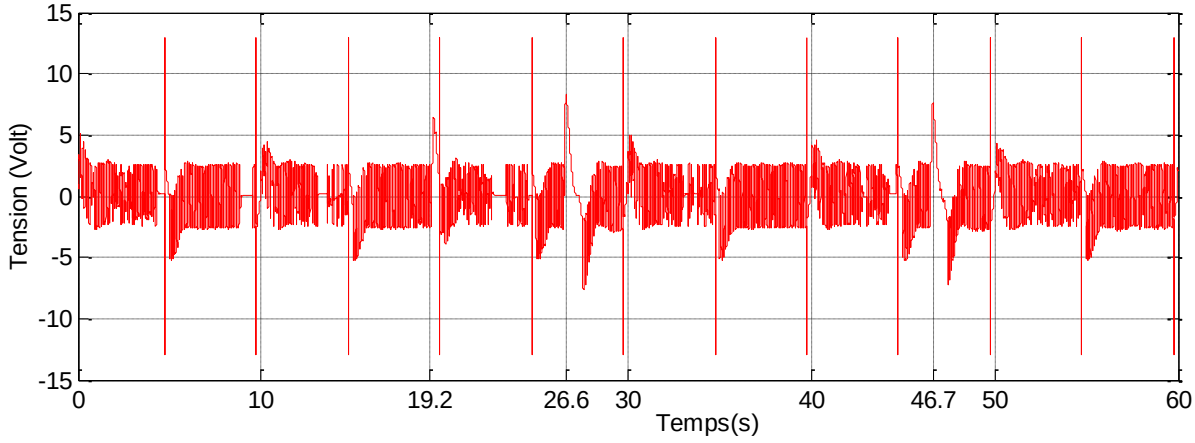


Figure 8.64. *Courbe de commande appliquée sur le moteur.*

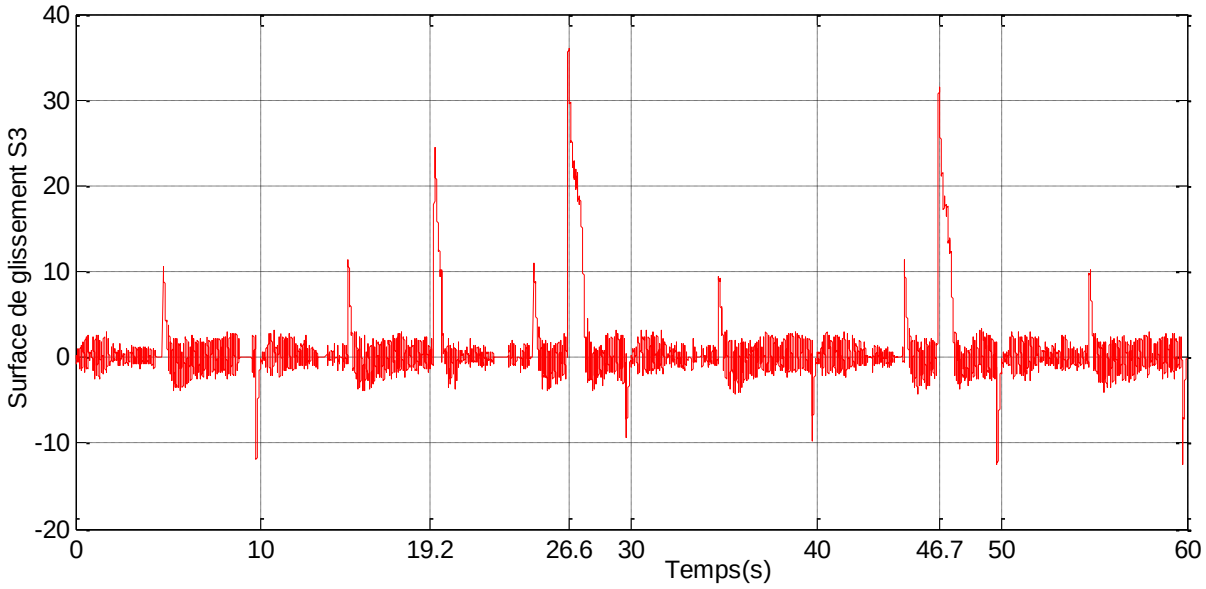


Figure 8.65. *Surface de glissement S3.*

- **Interprétation des résultats:**
- La commande de la tension d'alimentation est acceptable et varie à l'intérieur de l'intervalle du régime nominal.
- La position du chariot suit la position simulée.
- Le pendule est maintenu dans la position verticale très proche de l'angle initial.
- Le temps de réponse est acceptable et satisfait les contraintes du temps réels.
- La commande par mode de glissement incrémental a réussi à rejeter les perturbations extérieures.
- Cette commande a réussi à accomplir les tâches principales désirées.

8.6.1.4. Commande adaptative par réseau de neurones à apprentissage en temps réel

Les résultats obtenus lors de l'expérimentation sur plateforme réelle montrent la capacité de la commande à achever les principales tâches de commande. Cependant la tension appliquée au moteur varie pendant un temps trop long en dehors de l'intervalle du régime nominal ce qui n'est pas acceptable. D'après les résultats obtenus on constate clairement les deux points suivant :

- La commande neuronale induit de fortes amplitudes de tension.
- Aucun signe d'adaptation, le comportement de la commande reste inchangé au fil du temps.

La figure suivante montre la courbe de la commande neuronale enregistrée:

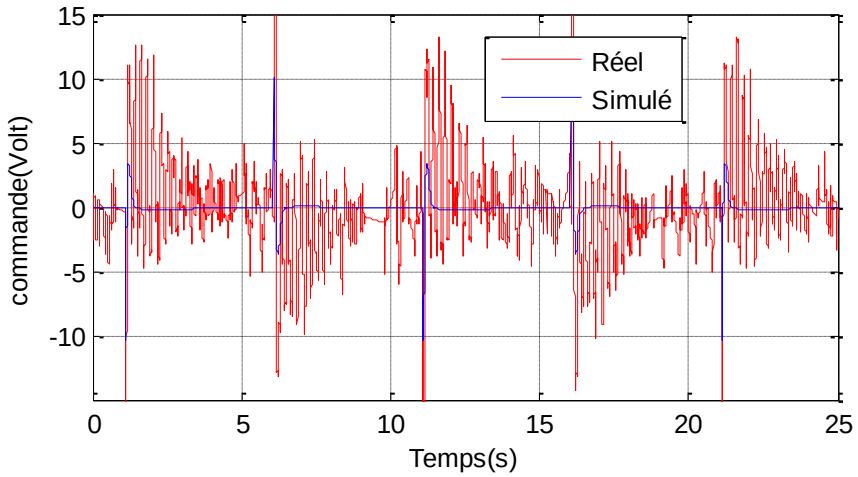


Figure 8.66. *Commande neuronale de la tension appliquée sur le moteur.*

Les fortes amplitudes, peuvent être expliquées par une vitesse de convergence non convenable de la sortie du réseau vers la sortie désirée. Cela implique que le pas d'apprentissage fixé en simulation est inadéquat. On a donc effectué des essais pratiques pour adapter le pas d'apprentissage en temps réel. Le résultat obtenu est illustré sur les deux figures suivantes :

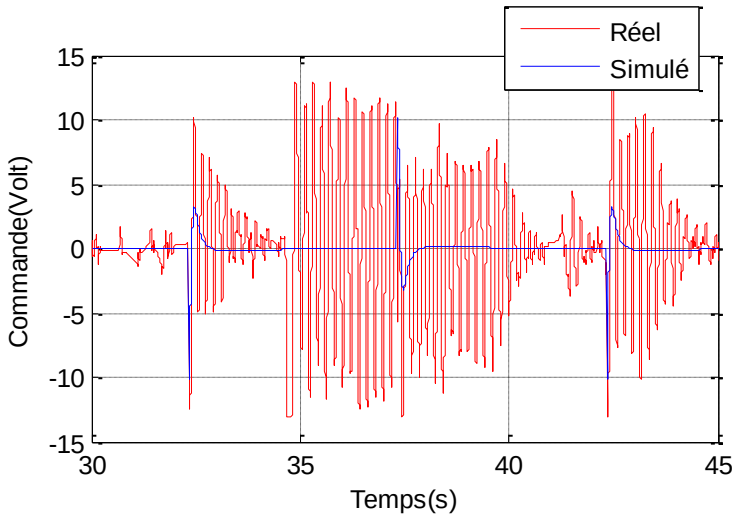


Figure 8.67. *Commande de la tension appliquée sur le moteur.*

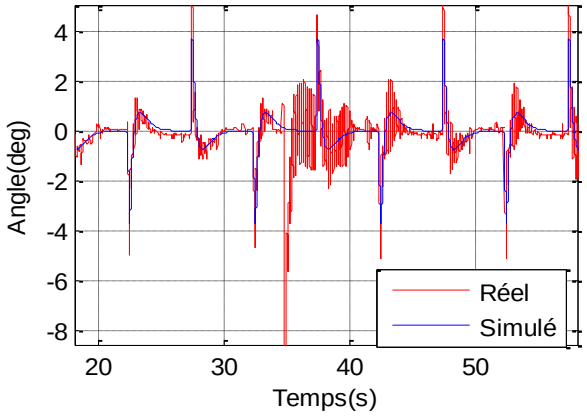


Figure 8.68. *Position angulaire.*

La courbe de commande obtenue montre une petite amélioration, la tension appliquée sur le moteur varie moins en dehors de l'intervalle du régime nominal. Cependant, on remarque encore une fois l'apparition de fortes amplitudes dans le signal de commande notamment en présence de perturbations extérieures. La figure (8.68) montre clairement les vibrations causées par ces fortes amplitudes. En agissant sur le pas d'apprentissage, la commande reste incapable de fournir le résultat souhaité et la variation du pas est limitée par le fait que:

- L'augmentation du pas d'apprentissage génère un bruit de commande.
- La diminution du pas d'apprentissage cause l'augmentation des erreurs de poursuites des signaux de références.

L'absence de l'aspect adaptatif peut être expliquée par un problème de dimensionnement, la configuration du réseau de neurones est choisie en simulation en absence du bruit lié à la pratique. On pense que les dimensions et la structure actuelle rend la commande incapable d'apprendre les dynamiques réelles du système en présence de perturbations.

La limitation de la variation du pas d'apprentissage est aussi liée au problème de dimensionnement, ainsi qu'à une insuffisance de conception au niveau de l'algorithme de rétro-propagation classique. Cette insuffisance provient du fait que le pas reste statique face aux changements de rythme des variations d'amplitude des erreurs de poursuites.

Pour cela on a opté pour les solutions suivantes :

- Choisir une nouvelle structure et une configuration adaptée à la pratique.
- L'introduction d'une fonction dynamique qui permet la variation du pas d'apprentissage.
- L'introduction d'un coefficient d'amortissement pour ajouter de l'inertie face aux modifications brutales des poids et aussi pour éliminer le phénomène d'oscillations qui apparait au point critique d'apprentissage.

8.6.1.4.1. Correction de la structure

En utilisant des signaux réels, la simulation numérique montre que malgré l'augmentation du nombre de neurones en chaque couche, le réseau reste incapable d'apprendre les dynamiques en présence de bruit, par contre, en ajoutant une 2^{eme} couche cachée, les résultats se sont remarquablement améliorés. La structure finale choisie est similaire à celle utilisée dans la commande à apprentissage hors ligne.

8.6.1.4.2. Correction de l'algorithme de rétro-propagation

En introduisant une fonction dynamique pour adapter la vitesse de convergence, le nouveau pas d'apprentissage sera donc définit comme suit :

$$\mu = \varrho(e, E_{max}) \qquad (8\text{-}1)$$

D'où :

- e : est l'erreur d'entrée.
- E_{max} : est l'erreur d'entrée maximale qui est dans notre cas égale à 26.
- ϱ : est une fonction positive.

Maintenant, il reste à ajouter de l'inertie à l'adaptation des poids. Etant donnée la définition de la loi d'adaptation des poids décrit par l'équation (5-17), on a:

$$W_{ij}^{k}(n+2) = W_{ij}^{k}(n+1) + \mu\, \delta_i^k O_i^{k-1} \qquad (8\text{-}2)$$

A partir des deux équations (3-17) et (8-2), la loi finale d'adaptation des poids tout en introduisant le coefficient d'amortissement est donnée par:

$$W_{ij}^{k}(n+2) = \xi W_{ij}^{k}(n) + W_{ij}^{k}(n+1) + \mu\, \delta_i^k O_i^{k-1} \quad (8\text{-}3)$$

- Où ξ est le coefficient d'amortissement.

8.6.1.4.3. Test et Configuration:

Des tests ont été effectués pour étudier la performance des nouvelles corrections. Les figures suivantes montrent les résultats avant et après l'introduction du coefficient d'amortissement lors d'un pas d'apprentissage élevé:

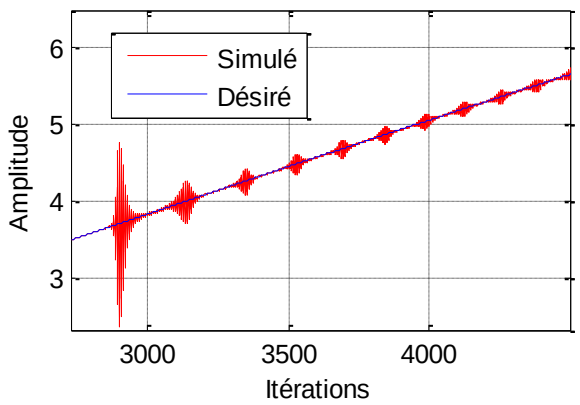


Figure 8.69. *Apprentissage avant l'introduction du coefficient d'amortissement.*

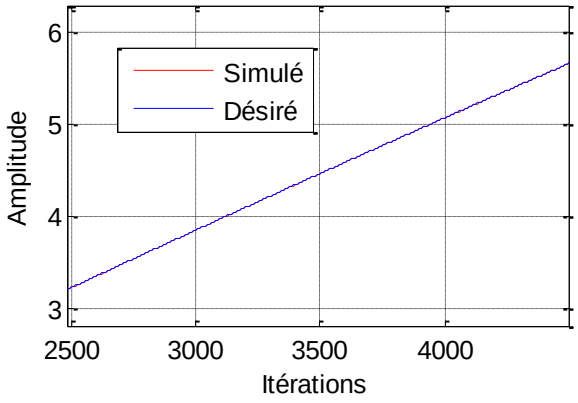


Figure 8.70. *Apprentissage après introduction du coefficient d'amortissement.*

La sensibilité envers les oscillations est contrôlée en agissant sur les paramètres d'apprentissage, la figure (8.71) montre la capacité d'approximation en présence d'oscillations.

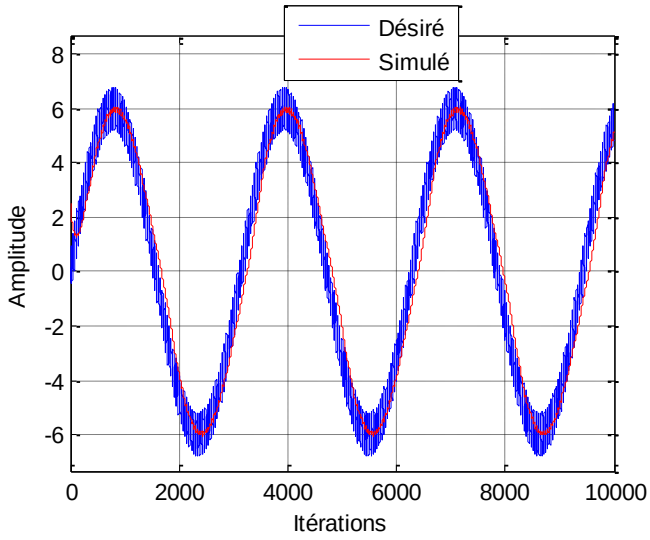


Figure 8.71. *Approximation d'un signal sinusoïdal avec de fortes oscillations.*

L'apprentissage des dynamiques réelles du système par la nouvelle structure neuronale est vérifié en effectuant un test d'approximation d'un signal réel d'erreur. Ce signal est enregistré lors des expériences précédentes. Les figures suivantes montrent les résultats obtenus en comparaison avec l'ancienne structure neuronale :

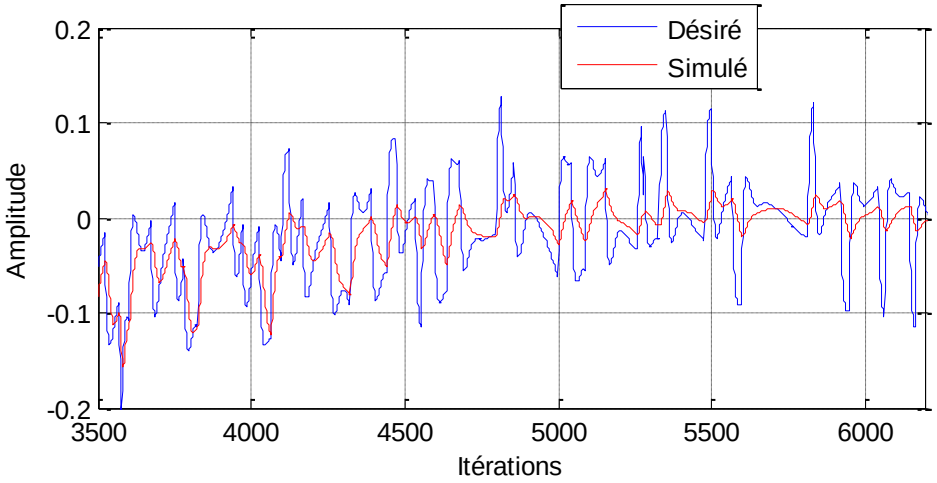


Figure 8.72. *Performance d'approximation par l'ancienne structure.*

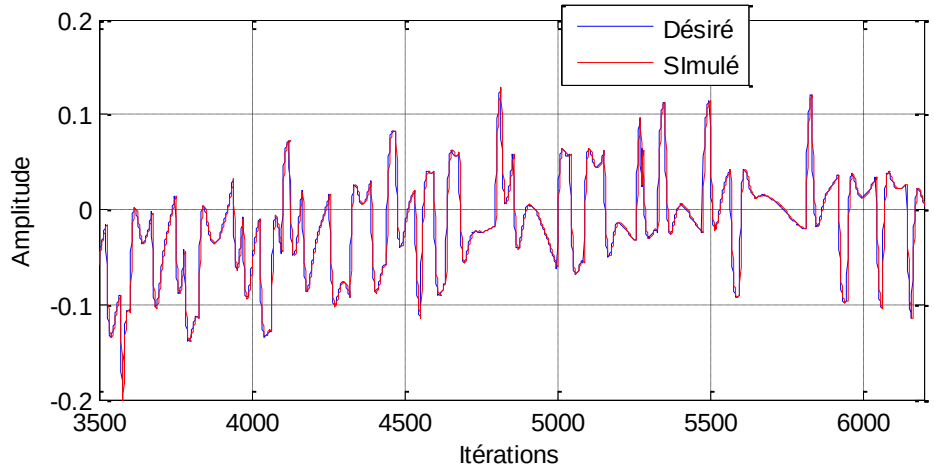


Figure 8.73. *Performance d'approximation par la nouvelle structure.*

Malgré plusieurs époques d'apprentissage en agissant sur le pas d'apprentissage, l'ancienne structure neuronale n'a pas réussi à approximer les variations dynamiques réelles, contrairement à la nouvelle structure, qui elle a réussi dès la première époque tout en gardant un pas d'apprentissage moins élevé.

8.6.1.4.4. Résultats

L'essai pratique de la commande corrigée sur plateforme réelle est effectué, les résultats obtenus montrent très bien l'efficacité des modifications apportées. La courbe de la commande neuronale obtenue est représentée sur la figure ci-après :

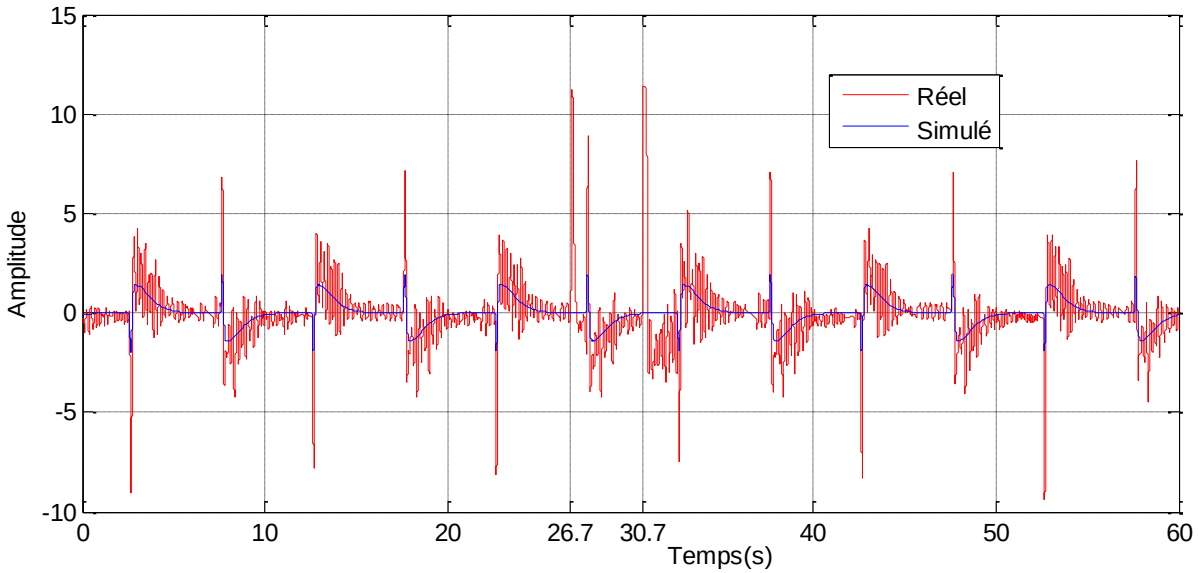


Figure 8.74. *Commande neuronale.*

On remarque clairement l'absence d'oscillations indésirables de fortes amplitudes même en présence de perturbations extérieures. L'apprentissage des dynamiques du système réel est réussi, cela est traduit par une amélioration progressive au fil du temps. Cette amélioration est illustrée sur la figure suivante :

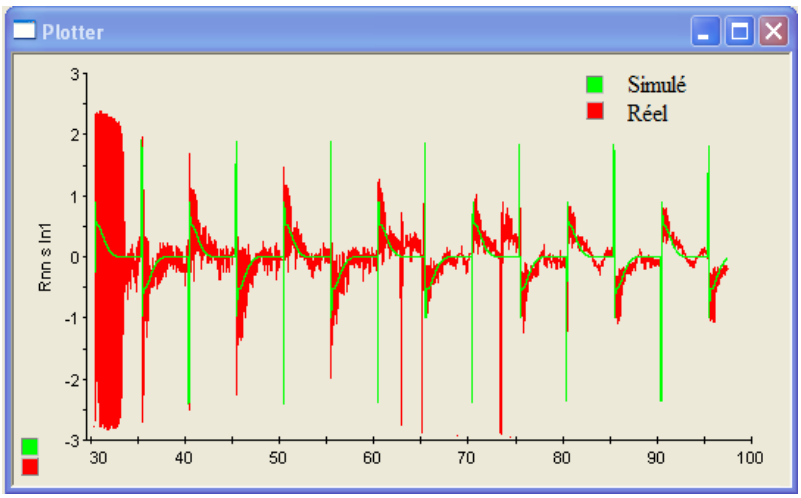


Figure 8.75. *Fenêtre de l'application ControlDesk représentant l'affichage en temps réel de l'évolution de la commande neuronale.*

En effectuant des Perturbations aux instants t=30.7s et t=26.6s, voici les résultats obtenus :

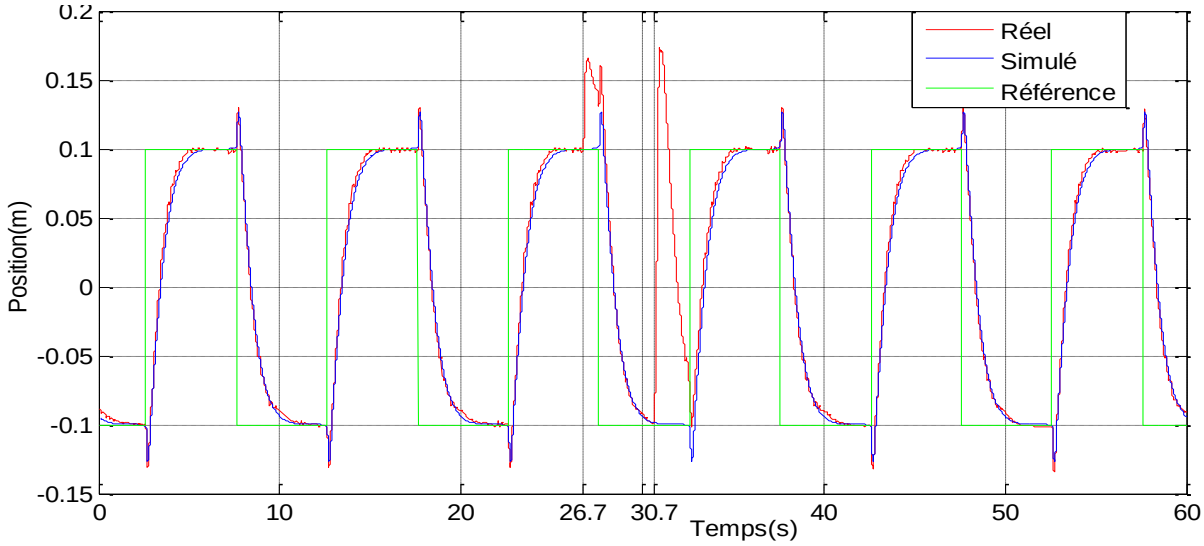


Figure 8.76. *Courbe de position linéaire en m.*

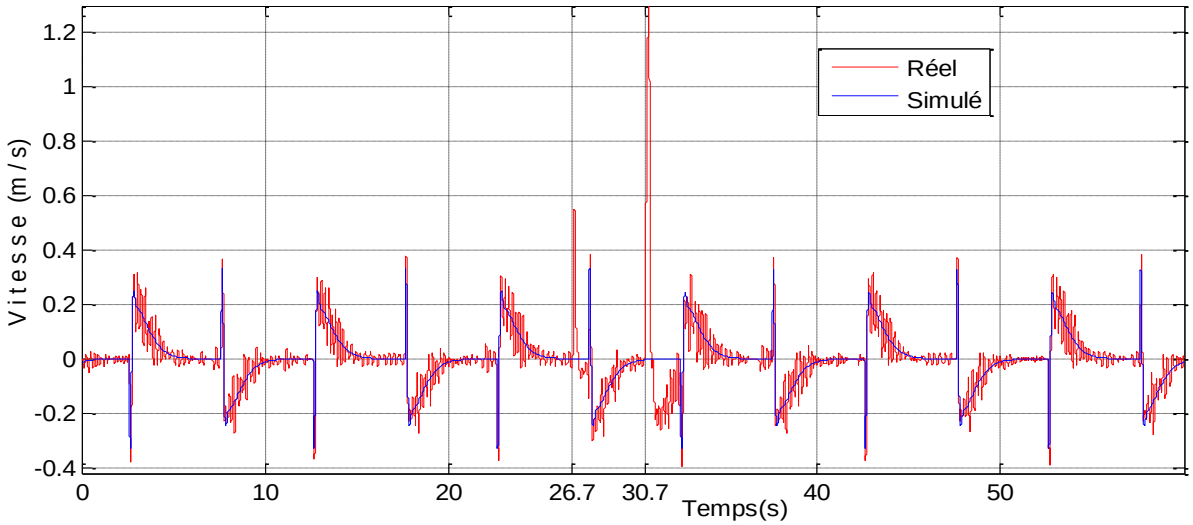


Figure 8.77. *Courbe de vitesse linéaire en m/s.*

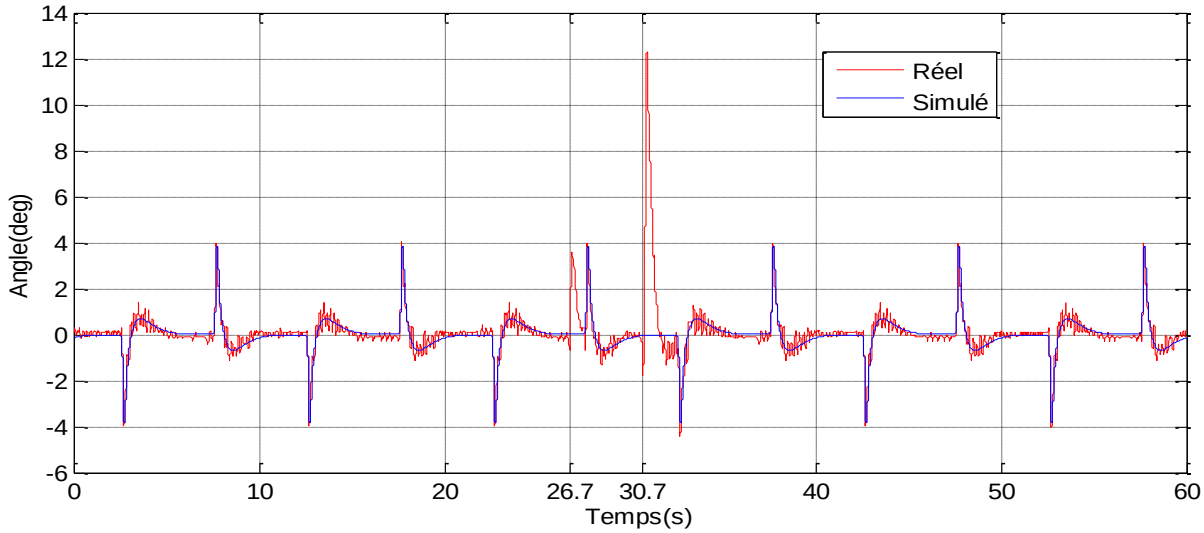


Figure 8.78. *Courbe de l'angle en degré.*

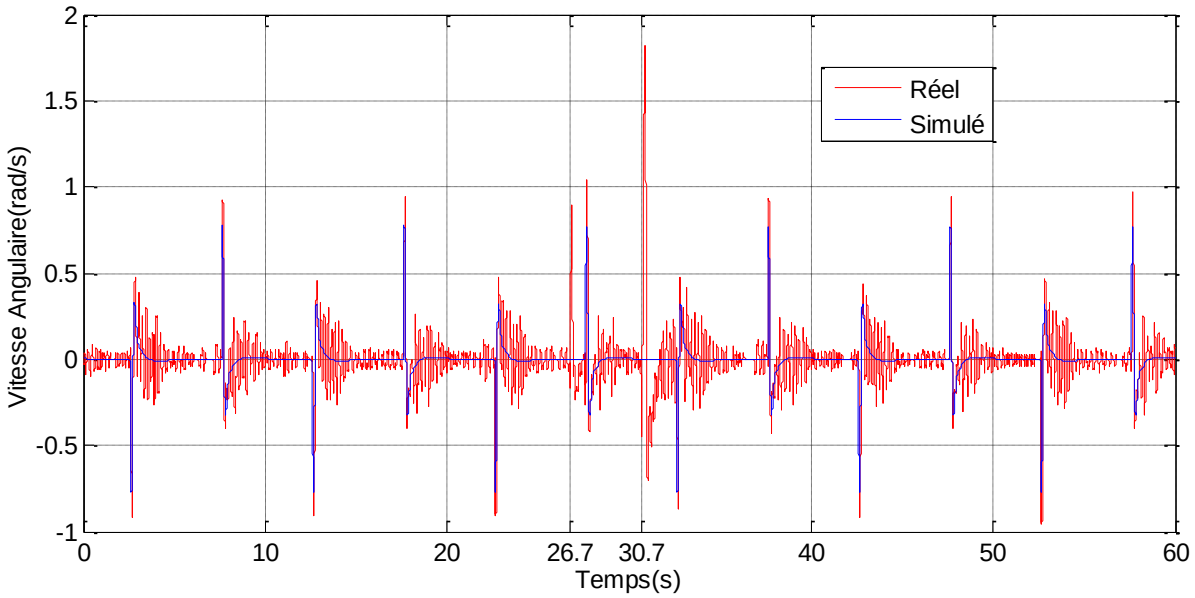


Figure 8.79. *Courbe de vitesse angulaire en radian/s.*

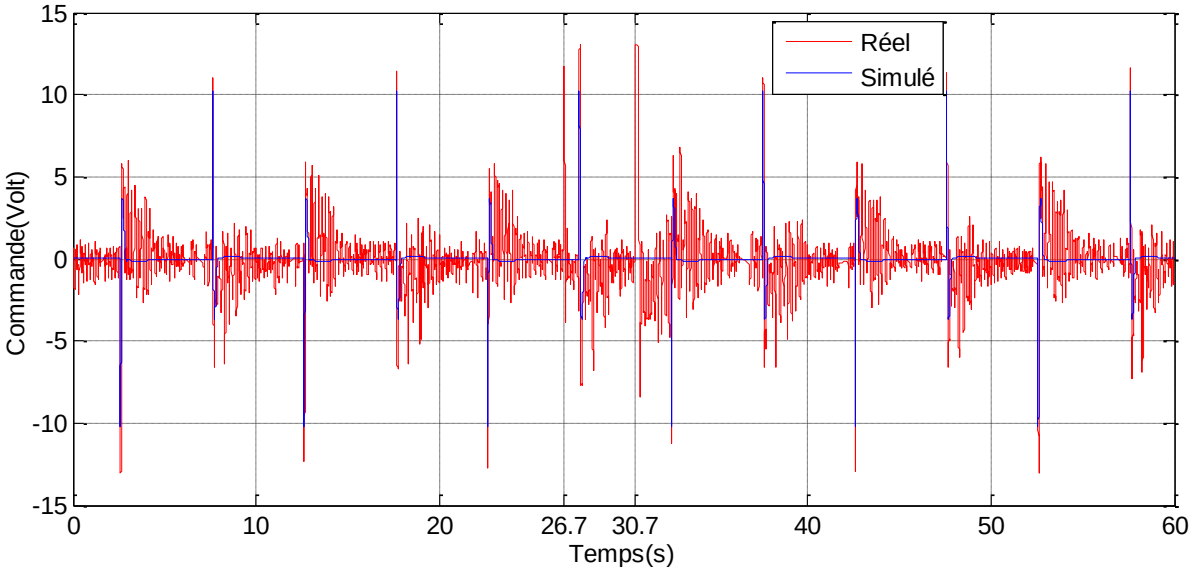


Figure 8.80. *Courbe de commande appliquée sur le moteur.*

- **Interprétation :**
  - La commande de la tension d'alimentation est acceptable et varie à l'intérieur de l'intervalle du régime nominal.
  - La position du chariot suit la position simulée avec une très grande précision.
  - Le pendule est maintenu en une position verticale acceptable proche de l'angle zéro.
  - Le temps de réponse est acceptable et satisfait les contraintes du temps réel.
  - La commande par réseaux de neurones à apprentissage en temps réel a réussi à rejeter les perturbations et rétablir l'équilibre du système en un délai acceptable.
  - Cette commande a réussi à accomplir les tâches principales de commande.

8.6.1.5. Commande robuste adaptative par Backstepping

Après plusieurs essais effectués pour le réglage des gains adaptatifs de la commande, les résultats obtenus lors de l'expérimentation sur plateforme réelle ne sont pas acceptables et ne satisfont pas les conditions de validations par rapport aux points suivants:

- La commande de la tension d'alimentation du moteur varie à l'extérieur du régime nominal, et par conséquent il a été nécessaire de limiter la variation afin de protéger le système. Un exemple de la courbe de la commande enregistré lors des essais est illustré sur la figure (8.81).

- La stabilité n'est pas assurée tout au long de l'axe du temps, et la poursuite de la trajectoire de référence qu'on juge mauvaise n'est pas assurée tel que montré sur l'exemple de la courbe de position enregistrée sur la figure (8.82).

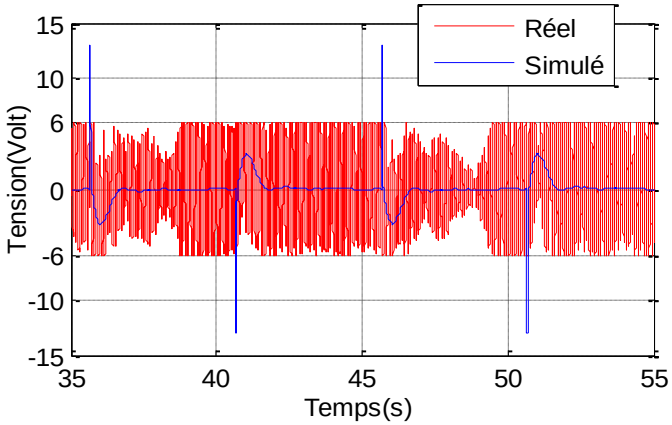


Figure 8.81. *Commande de la tension appliquée sur le moteur après limitation.*

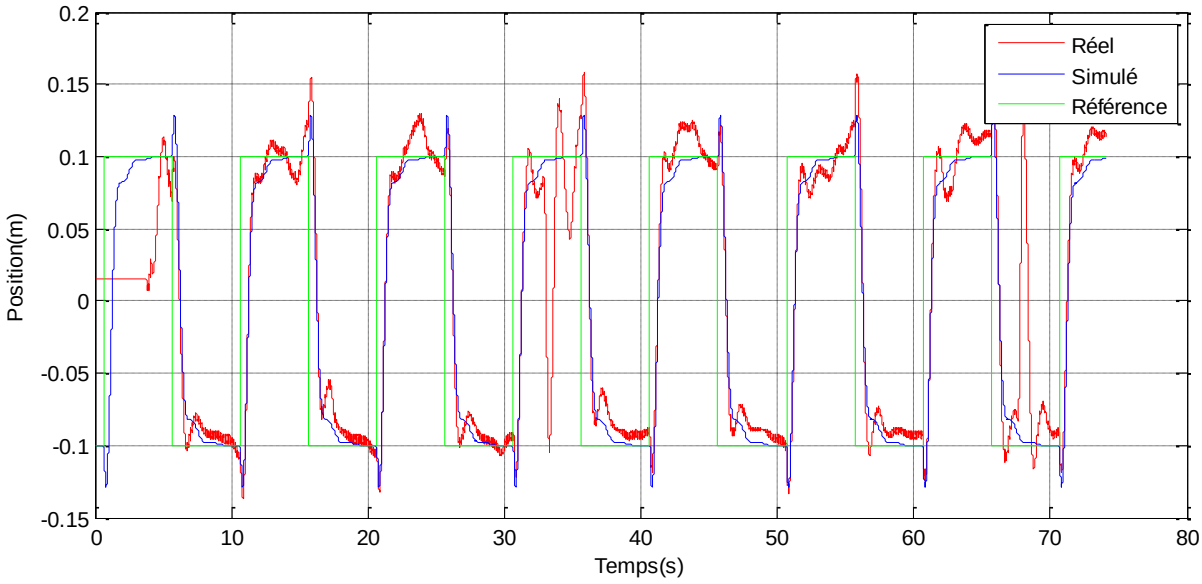


Figure 8.82.*Courbe de position linéaire en m.*

D'après les résultats obtenus, on a essayé de déterminer la cause à l'origine de l'échec de cette technique. On a pensé à la nécessité d'une estimation plus exacte des constantes de conception liées à la commande. Pour cela et afin de localiser le problème on a opté pour la stratégie résumée dans le cheminement suivant :

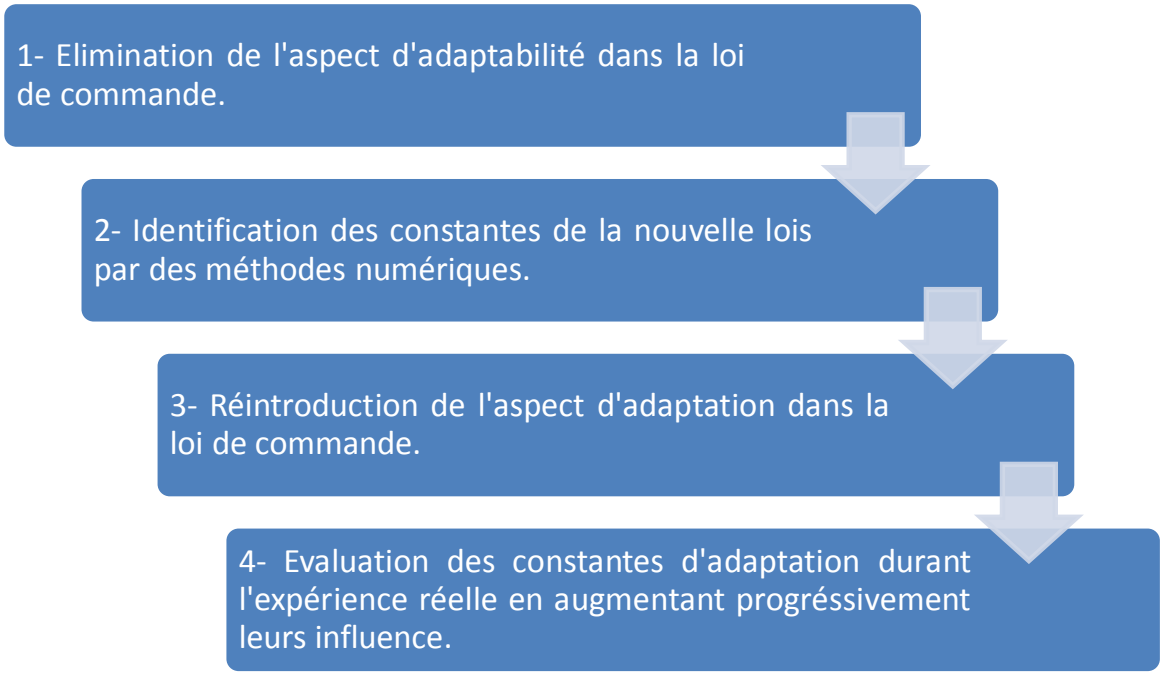


Figure 8.83. *Stratégie utilisée pour corriger la configuration de la loi de commande.*

L'élimination de l'aspect d'adaptation dans la loi de commande est effectuée par la fixation des valeurs des paramètres adaptatifs, Etant donné l'équation (29) chapitre 6 :

On a donc $$f = \hat{g}\left(Lcmd + \hat{h}\right) \quad (8\text{-}4)$$

D'où

- $Lcmd$: est définit comme la loi de commande par Backstepping sans le mécanisme d'adaptation.
- $\hat{g}\ et\ \hat{h}$: sont les paramètres adaptatifs.

Donc si on impose $\hat{g}$=1, et $\hat{h} = 0$, la loi de commande non adaptative devient donc:

$$f = Lcmd \quad (8\text{-}5)$$

L'identification des constantes de la nouvelle loi de commande sans mécanisme d'adaptation est effectuée par des algorithmes de méthodes de calcul numérique.

La nouvelle configuration obtenue est :

$h_0 = 10, g_0 = 10, \gamma_1 = 2\ , \gamma_2 = 2, C_1 = 9.3715, C_2 = 19.1178\ , \lambda_1 = 0.2775\ , \lambda_2 = 1.7643.$

Après essai on a obtenu les résultats sur les figures suivantes :

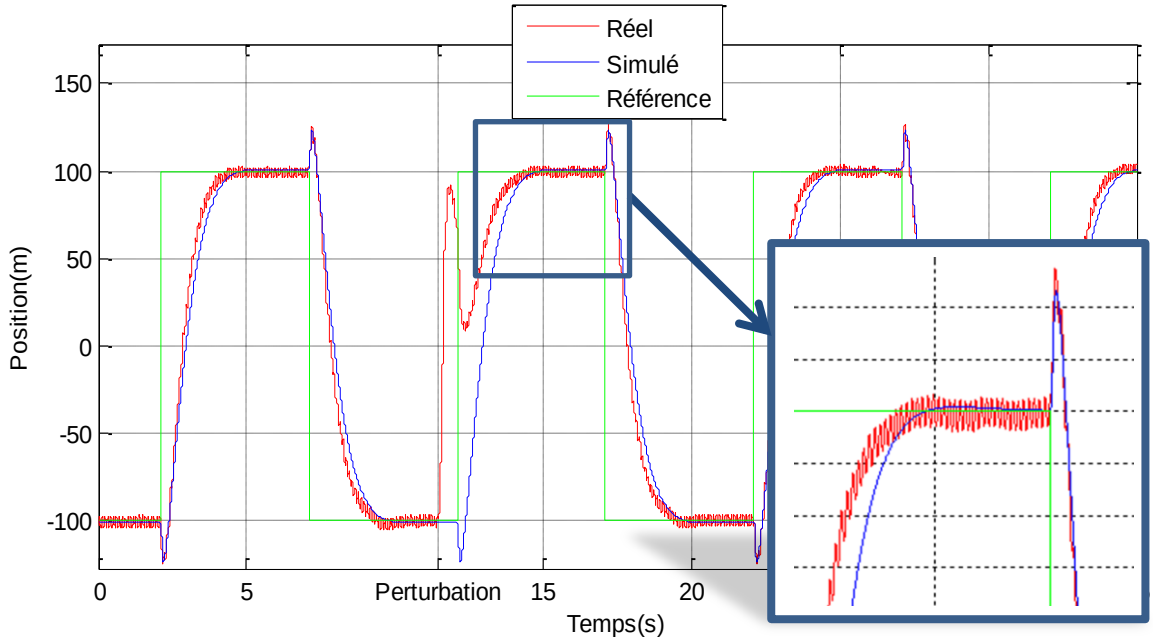


Figure 8.84. *Bruit indésirable dans la courbe de la position linéaire.*

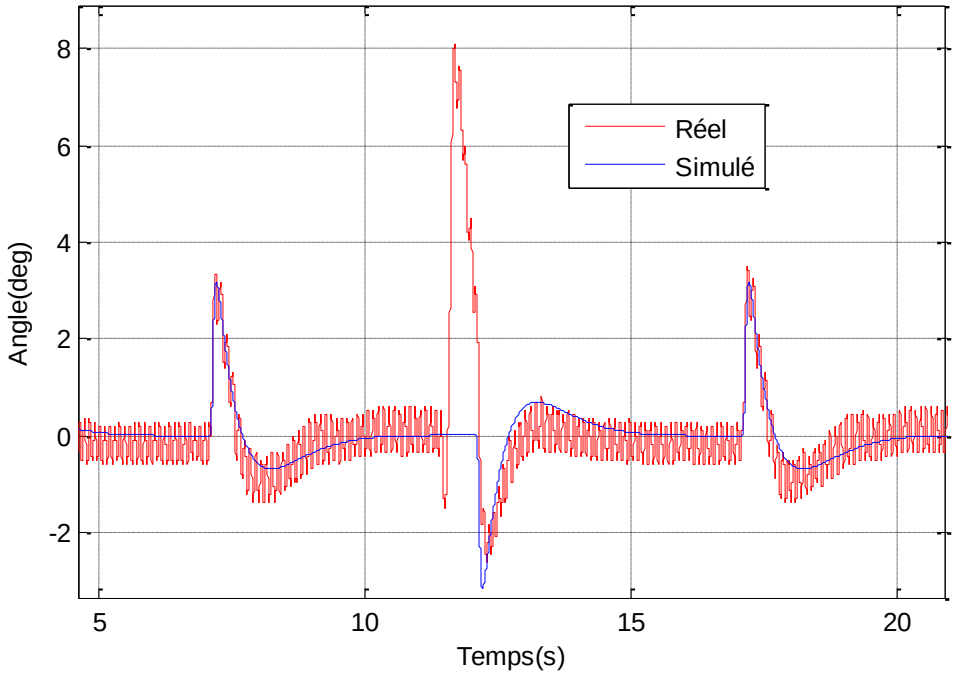


Figure 8.85. *Bruit indésirable dans la courbe de la position angulaire.*

Les résultats obtenus montrent clairement la capacité de la commande à effectuer les tâches principales. Cependant, ces résultats induisent de fortes oscillations de bruit indésirables et inacceptables comme le montrent les figures (8.84) et (8.85). De telles oscillations sont dues à la forte saturation qui apparait au fil du temps dans la sortie du système de commande. Cette saturation apparait sur la figure ci-dessous :

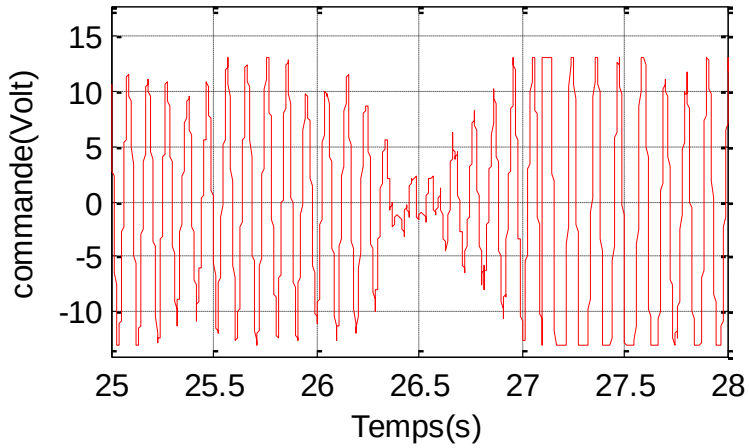


Figure 8.86. *Courbe de commande appliquée sur le moteur*

Il est clair qu'une telle dégradation au fil du temps est causée par le mécanisme d'adaptation, bien que les courbes enregistrées après plusieurs essais et plusieurs modifications montrent la divergence des paramètres adaptatifs. Ceci s'explique par l'incapacité d'adaptation du mécanisme utilisé en présence de bruit réel.

On a effectué énormément de modifications des valeurs des constantes d'adaptation durant les essais, mais le résultat reste inchangé. On a essayé alors de diminuer l'influence du bruit sur le mécanisme d'adaptation par l'insertion d'un filtre, mais on a constaté qu'un filtrage efficace rend la commande incapable de stabiliser le système.

On a donc décidé de fixer la variation des paramètres adaptatifs autour des valeurs liées à la meilleure performance enregistrée.

8.6.1.5.1. Résultats

En effectuant la Perturbation à l' instant t=33.24, voici les résultats obtenus :

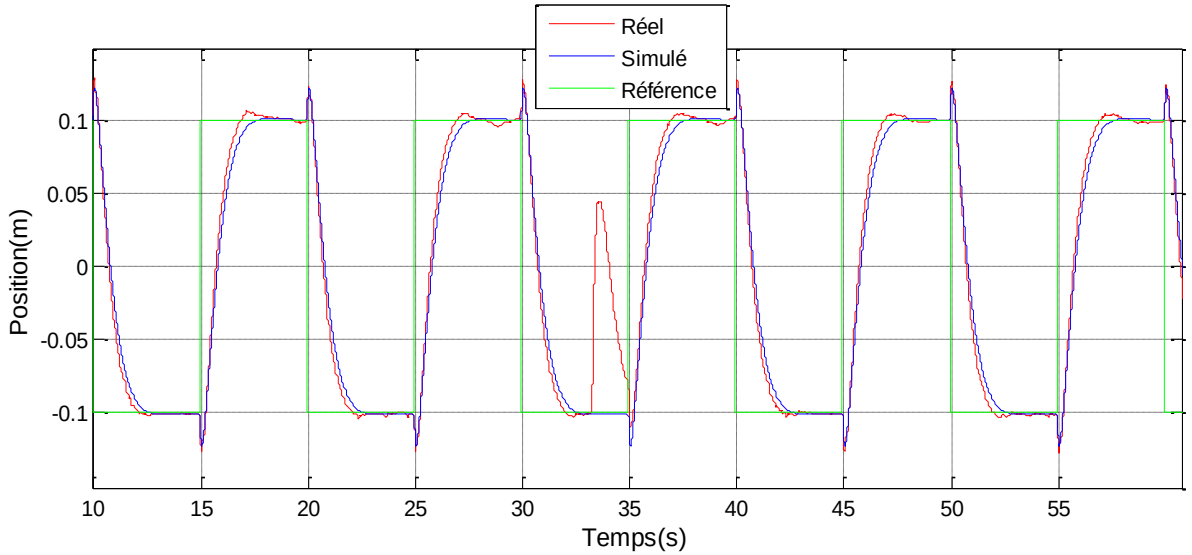


Figure 8.87. *Courbe de position linéaire en m.*

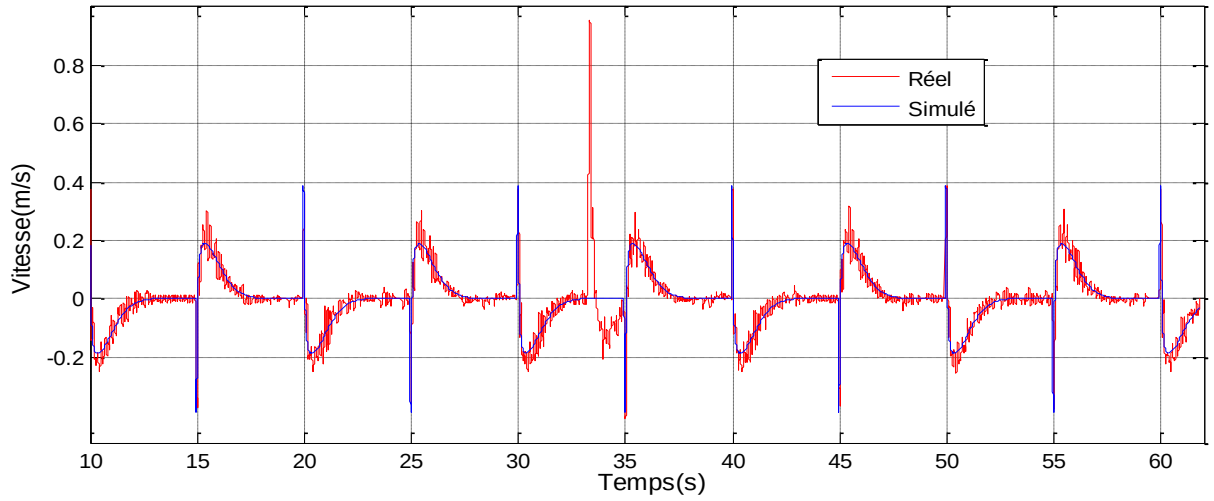


Figure 8.88. *Courbe de vitesse linéaire en m/s.*

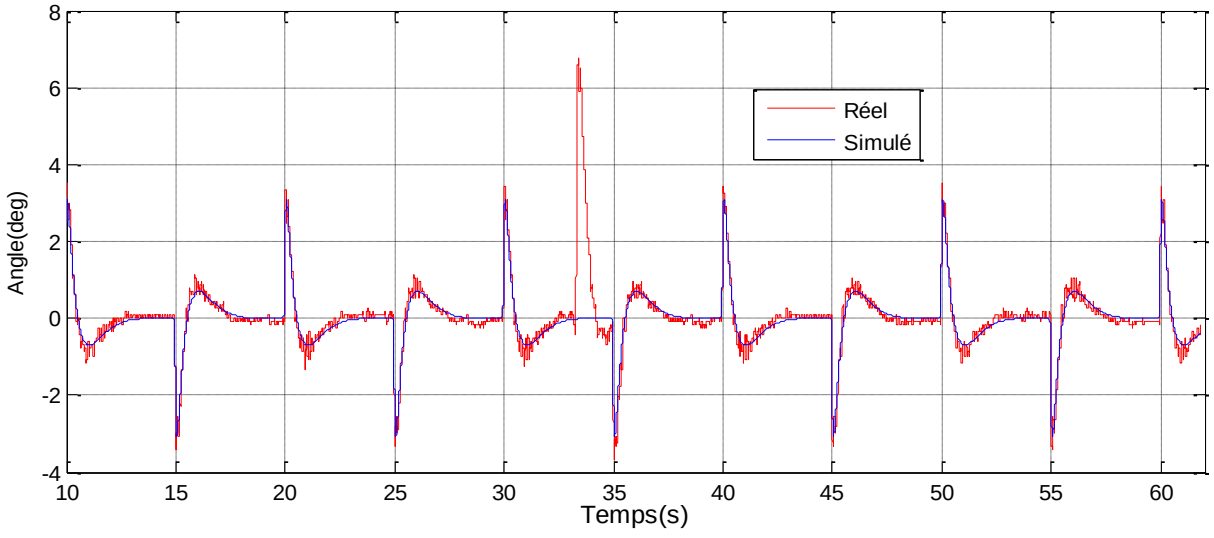


Figure 8.89. *Courbe d'angle en degré.*

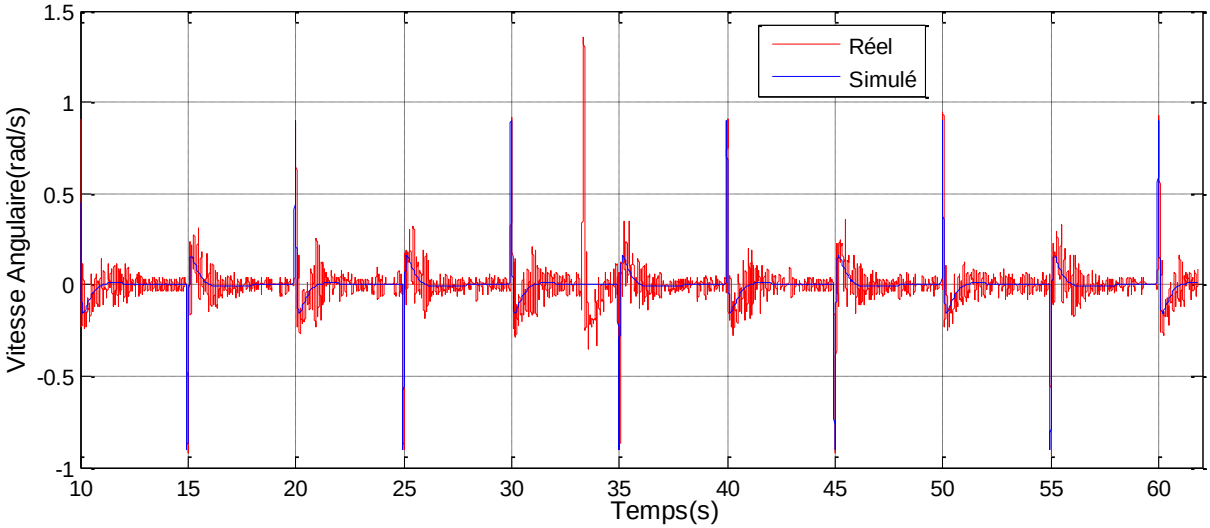


Figure 8.90. *Courbe de vitesse angulaire en radian/s.*

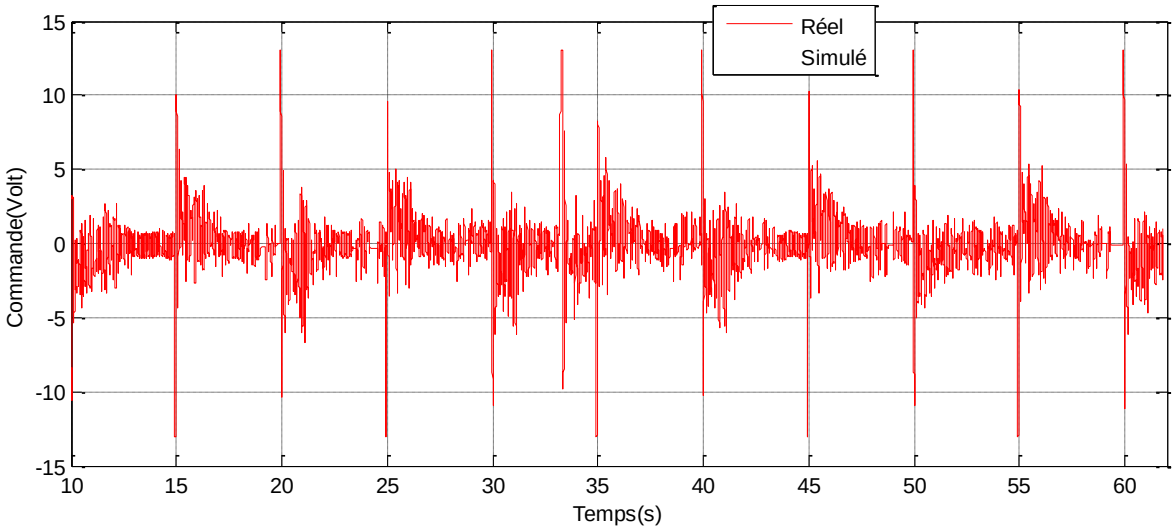


Figure 8.91. *Courbe de commande appliquée sur le moteur.*

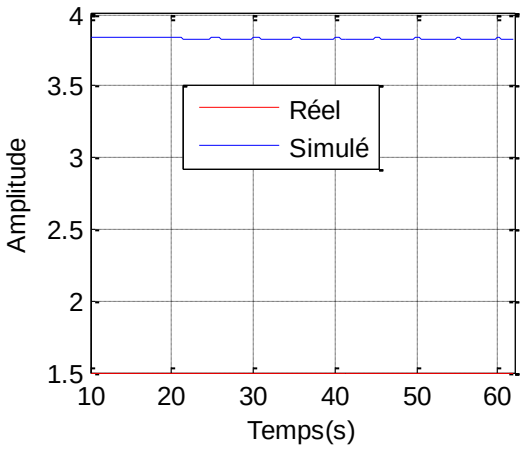


Figure 8.93.*Paramètre adaptatif g.*

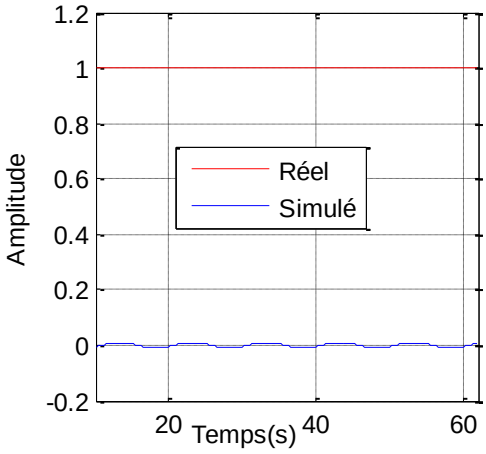


Figure 8.92.*Paramètre adaptatif h.*

- **Interprétation des résultats:**

- La commande de la tension d'alimentation est acceptable et varie dans l'intervalle du régime nominal.
- La position du chariot suit très bien la position simulée.
- Le pendule est maintenu dans la position verticale acceptable très proche de l'angle initial.
- Le temps de réponse est acceptable et satisfait les contraintes de temps réels.
- La commande a réussi à rejeter les perturbations.
- La commande a réussi à accomplir les principales tâches désirées.

8.7. Conclusion

Dans ce chapitre, on est arrivé à effectuer des commandes en temps réel sur le système du pendule inversé en utilisant la carte de commande dSPACE1104. Ce système fait partie de la catégorie des systèmes temps réel à contrainte dure.

On constate clairement la différence importante entre la conception d'un système de commande performant dans le cas réel et dans le cas de simulation. Les résultats obtenus montrent qu'une technique de commande peut donner de bonnes performances en simulation, et de très mauvais résultats sur le système réel.

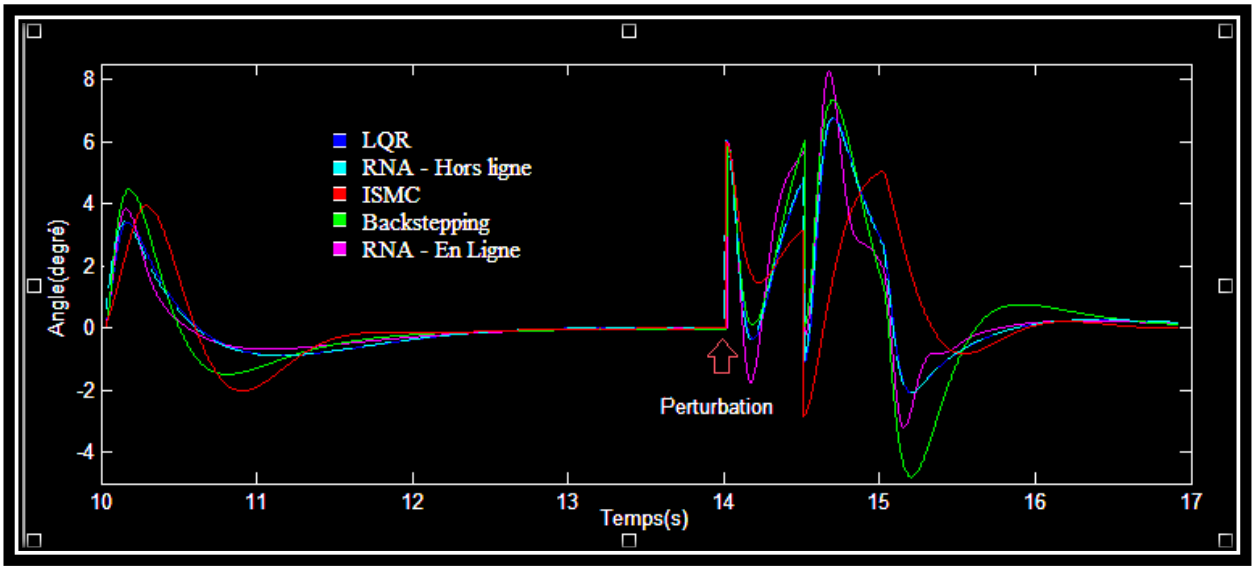


Figure 8.94.*Positions angulaires en présence de perturbation lors de la simulation.*

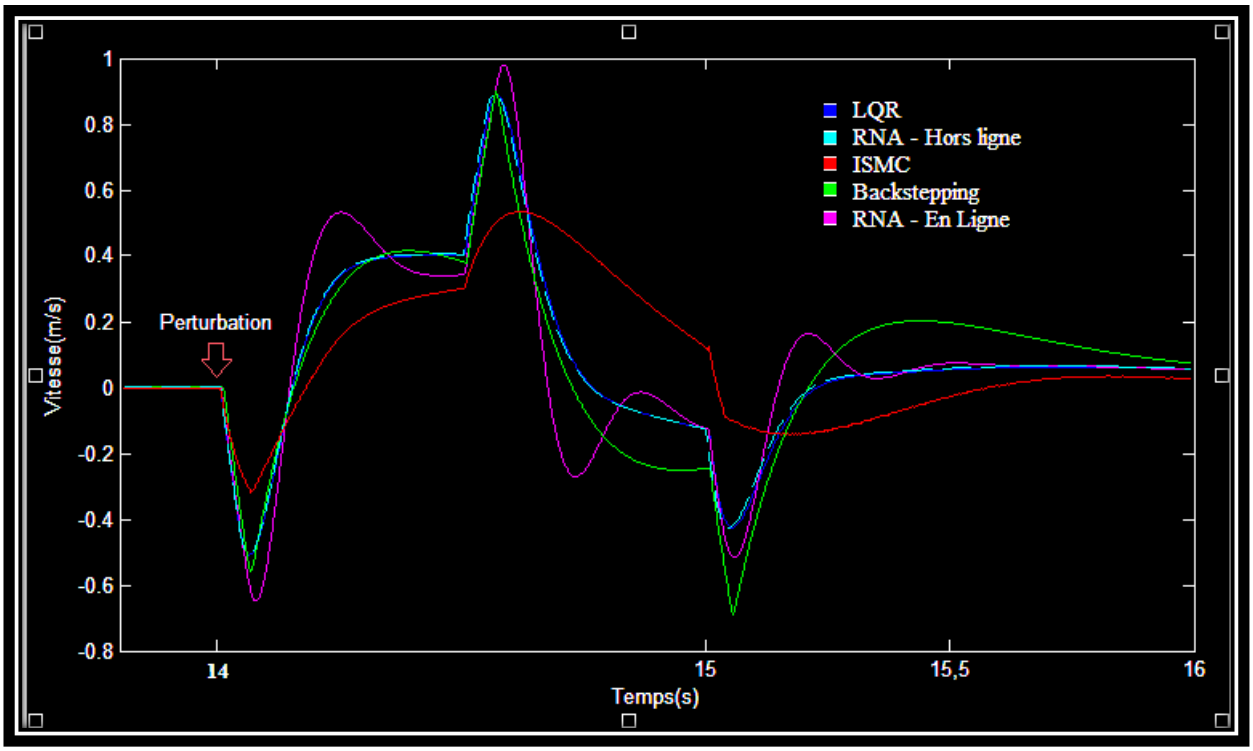


Figure 8.95.*Vitesses linéaires en présence de perturbation lors de la simulation.*

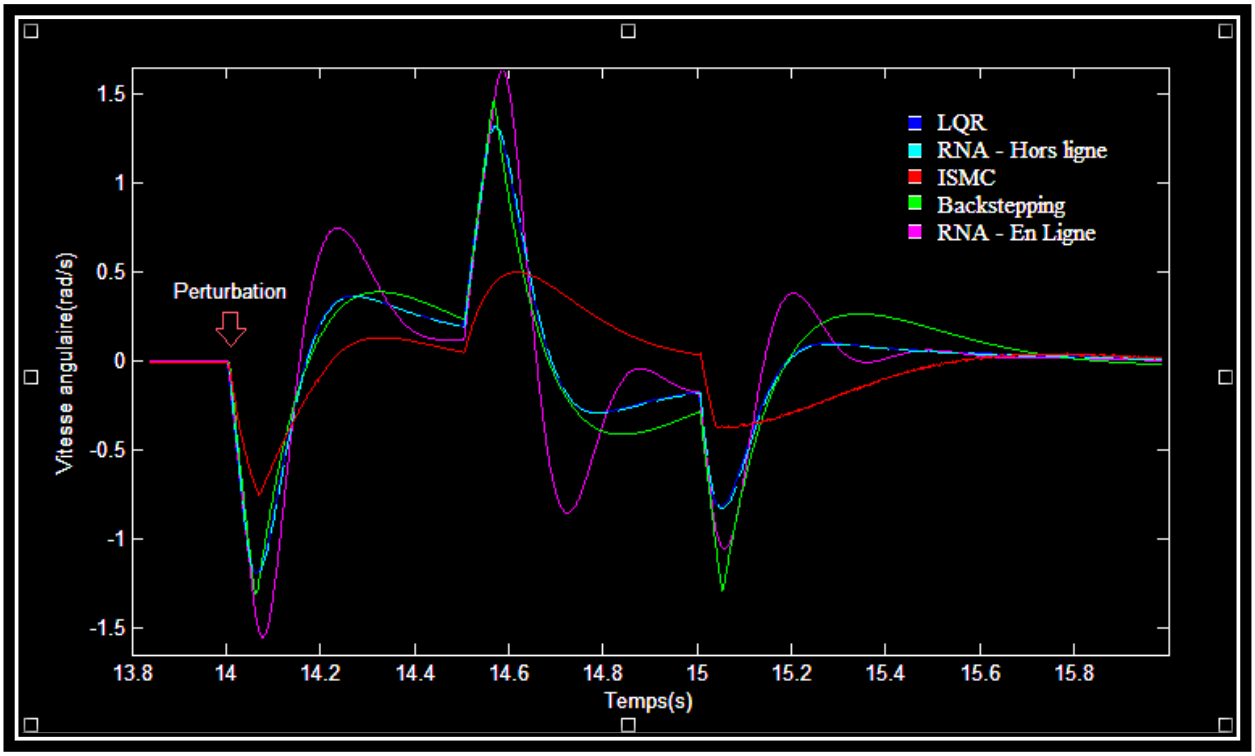


Figure 8.96.*Vitesses angulaires en présence de perturbation lors de la simulation.*

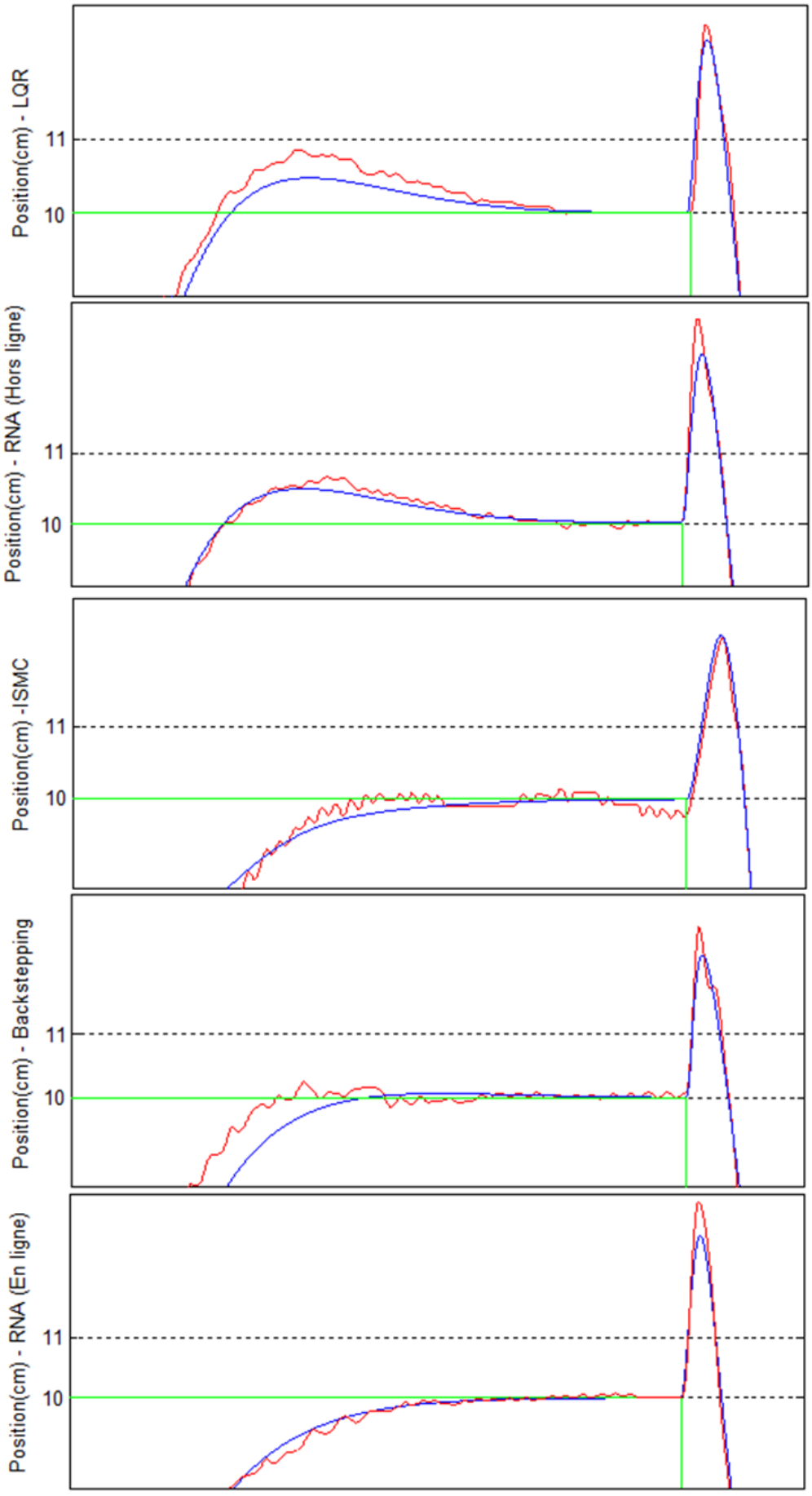


Figure 8.97.*Positions linéaires réels du chariot.*

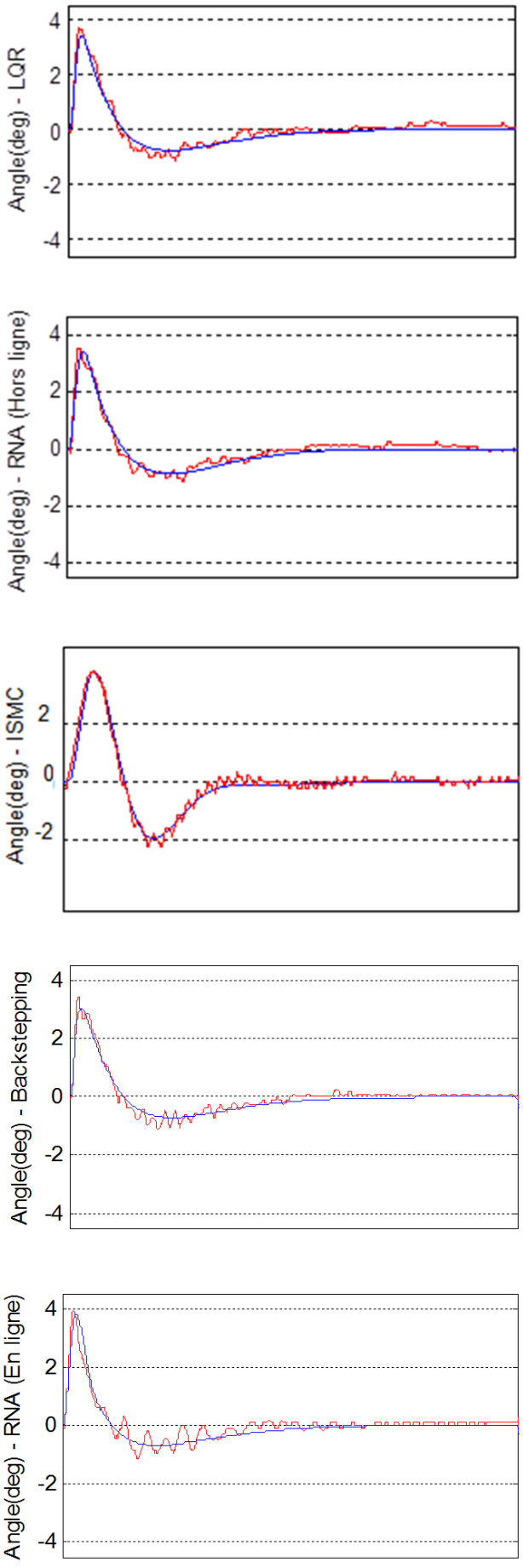


Figure 8.98.*Positions angulaires réelles.*

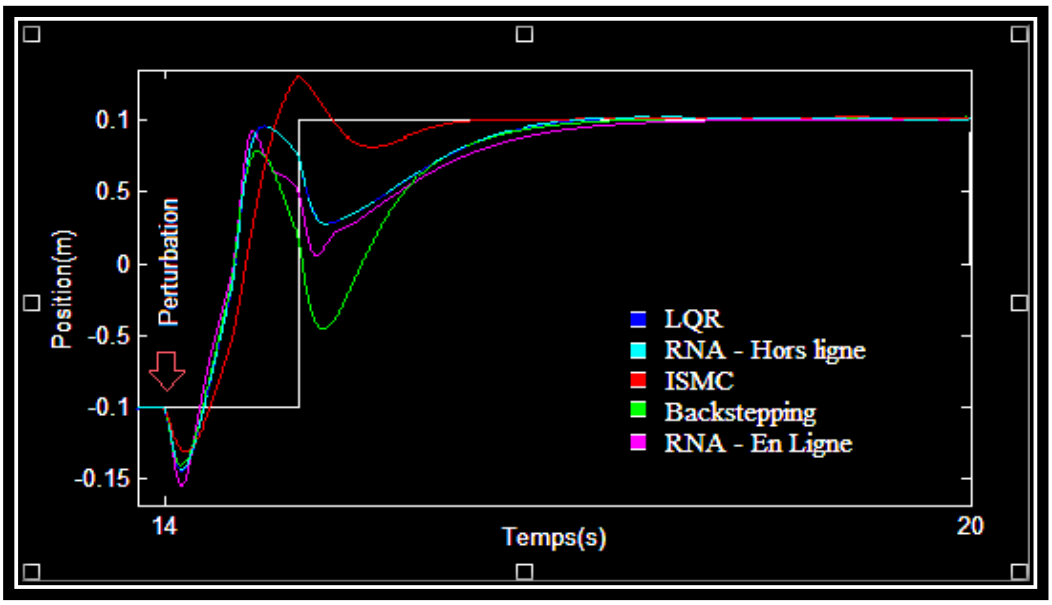


Figure 8.99.*Positions linéaires en simulation avec perturbation.*

Les résultats enregistrés par cette commande sont une preuve de la qualité de la configuration, du bon choix de la structure et de la stratégie adéquate de prétraitement de données utilisés. Ces éléments constituent, en fait et en général des difficultés majeures pour la conception de la commande par réseau de neurones à apprentissage hors ligne.

Cette technique comme le montre les figures (8.94) et (8.99) a réussi à reproduire le comportement de la commande LQR laquelle a servi comme maître fournissant les exemples d'apprentissage. Malgré le fait que le réseau n'a pas été entrainé sur le rejet des perturbations, la commande a réussi à le faire et à estimer la sortie nécessaire grâce aux capacités du réseau de neurones. En simulation les figures (8.95) (8.96) montrent bien la qualité de l'estimation durant la perturbation et la superposition est quasiment absolue avec les courbes de la commande LQR que ce soit pour les positions ou les vitesses enregistrées.

Tout comme la commande LQR, on remarque un dépassement lors du changement de position du chariot. Cependant ce qui justifie la copie du comportement d'un autre contrôleur déjà existant c'est la capacité des réseaux de neurones à estimer la commande au-delà de la zone de fonctionnement de ces contrôleurs. Les courbes réelles enregistrées lors de l'expérimentation montrent une petite amélioration par rapport à la commande LQR comme le montre l'exemple sur la figure (8.97).

La conception de la commande ISMC est difficile notamment pour ce qui est du choix des valeurs des constantes arbitraires. La loi de commande inclut les fonctions non linéaires du modèle mathématique du système ce qui complique de plus en plus la conception, surtout lorsqu'il s'agit d'un modèle très complexe. Une fois les constantes de conception fixées sur MATLAB, la création du bloc Simulink et l'implémentation ne pose pas de problème.

Les résultats obtenus de la commande ISMC sont remarquablement différents par rapport aux résultats de toutes les autres techniques étudiées. On peut facilement remarquer sur la figure-(8.98) que l'angle de réaction est le moins aigüe par rapport à la verticale au moment du changement de position linéaire du signal de référence. Aussi, on remarque que les sommets des courbes de la commande ISMC sur les figures (8.99) et (8.94) ont les amplitudes les moins fortes, cela signifie que cette commande réagit moins brutalement faces aux variations brutales des signaux de référence et aussi lors des perturbations. Cette caractéristique est confirmée comme le montre l'énorme différence sur les figures (8.95) et (8.96) ou la commande ISMC enregistre les vitesses linéaires et angulaires les moins élevées et donc les réactions les moins brutales. Cet avantage est souhaitable pour de nombreux systèmes telle que les systèmes mécaniques, car il permet d'éviter le vieillissement des composants mécaniques.

On note aussi que la commande ISMC est spécialement conçue pour les systèmes mécaniques sous-actionnés [2] tel que le pendule inversé utilisé en cette étude. L'aspect incrémental et la structure de plusieurs surfaces de glissement donnent plus de douceurs à la réaction de la commande ISMC. D'un autre côté, sur la figure (8.99) on remarque que le temps de réaction après la perturbation est le plus long par rapport aux autres techniques, le chariot a dû prendre une longue distance lors des perturbations avant que la commande puisse stabiliser le système. Du côté du balancement, on remarque en simulation comme en réel sur les figures-(8.94) et (8.98) que la commande enregistre le temps remarquablement le plus court, le pendule atteint la proximité de 0 degré en un temps record par rapport aux autres techniques. Cependant la figure (8.97) montre une variation qui présente trop de sommets, cela se traduit par un phénomène vibratoire dû à l'influence du caractère commutatif qui caractérise les techniques basées sur le mode de glissement.

En simulation la conception de la commande robuste adaptative par Backstepping est moins difficile que la commande ISMC, son aspect d'adaptation rend les constantes arbitraires moins sensibles et faciles à configurer, la création des blocks Simulink ainsi que l'implémentation sont faciles à effectuer.

Malgré que la commande a donné de très bons résultats en simulation avec sa capacité d'adaptation, la réalisation pratique a été très différente et difficile, la commande n'a pas donné de bons résultats sur le système réel, le mécanisme d'adaptation a été complétement divergeant dans la pratique. Cela montre qu'une technique de commande peut donner de bonnes performances en simulation, et de très mauvais résultats sur le système réel. Les résultats réels sont obtenus en limitant la variation des paramètres adaptatifs pour éviter la divergence du mécanisme d'adaptation. En simulation, on remarque qu'après la perturbation et avant que la commande puisse réagir pour stabiliser le système, le chariot a pris la plus courte distance en comparaison aux autres techniques comme montré sur la figure (8.99). Aussi sur la figure (8.97) on voit clairement que la commande est la plus rapide à stabiliser la position linéaire du chariot lors du changement de position.

La conception de la commande par réseau de neurones à apprentissage en ligne est très difficile, notamment le choix des valeurs des constantes arbitraires, choix de la structure, choix de la configuration adéquate du réseau et configuration de l'algorithme de rétro- propagation. La réalisation sur Simulink est très difficile notamment de par la complexité de réalisation des réseaux de neurones, la limitation d'utilisation des blocs de bases de Simulink, et la complexité d'intégration de l'algorithme de rétro-propagation.

L'implémentation est difficile également, de nombreux problèmes sont dûs à la complexité qui augmente en fonction de la taille du réseau utilisé. L'on cite, par exemple, le problème de boucles algébriques qui caractérise la simulation sous Simulink, ou le solveur est souvent incapable d'effectuer un calcul sur ces boucles à cause de leurs complexités.

Les résultats obtenus montrent une très grande performance, la commande est complètement indépendante du modèle mathématique sans aucune connaissance à priori des dynamiques du système. Après les corrections effectuées durant la partie expérimentale, la commande a prouvé une forte tolérance envers les incertitudes. Du coté balancement, les résultats obtenus montrent une meilleure performance par rapport aux autres techniques notamment en pratique (figures 8.97 et 8.98). On peut constater sur les figures que les variations de la position angulaire et de la position linéaire induisent moins de sommets, et que la convergence est plus régulière. Par contre durant la commande de poursuite du chariot, la performance est faible par rapport aux autres techniques mais a tendance à s'améliorer avec le temps. Sur les figures (8.95) et (8.96), on constate que la commande enregistre les vitesses linéaire et angulaire les plus forte, cela se traduit par une réaction moins douce, rapide et plus brutale.

Contrairement aux autres techniques étudiées celle-ci est la plus avantageuse, avec une commande de position sans dépassement, sans phénomène de broutement, et avec une meilleure stabilisation dans la partie balancement plus une capacité d'adaptation.

L'inconvénient remarquable durant les essais pratiques, c'est que les performances se dégradent après qu'on effectue des perturbations et le niveau de performance n'est rétabli qu'après une période relativement longue par rapport aux autres techniques étudiées.

La figure (8.100) montre une comparaison du temps d'exécution des différentes techniques étudiées après une simulation de 2 minutes sur Simulink. La simulation est effectuée sur un ordinateur Intel core i3 Cpu2.53 GHz. Il est clair que les réseaux de neurones marquent les plus grandes durées. Ceci traduit la complexité des algorithmes basés sur les réseaux de neurones.

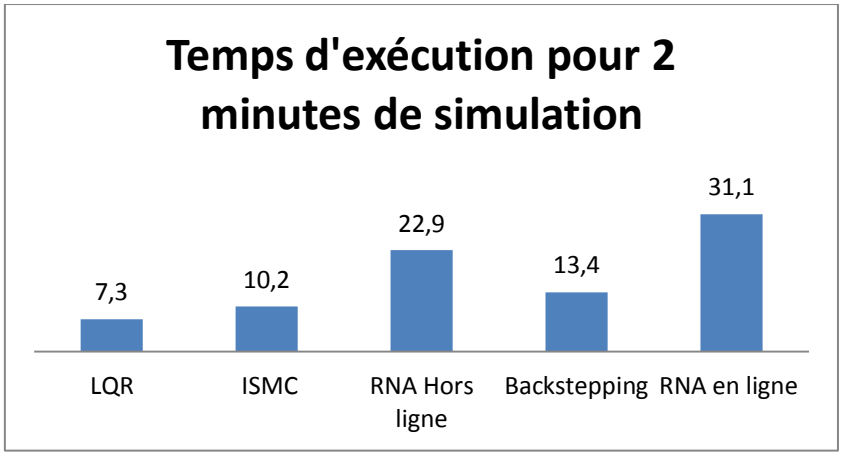


Figure 8.100. *Temps d'exécutions sur Simulink exprimé en seconde.*

Conclusion

Ce travail a permis la capitalisation de connaissances théoriques et techniques importantes et nécessaires pour la simulation et la mise en œuvre de processus dynamiques. Un système de commande basé sur une carte dSPACE a été exploité pour implémenter un certain nombre de techniques issues de l'état de l'art en matière de commande de systèmes dynamiques temps réel. Ainsi, des méthodes de commande basées sur des modèles d'apprentissage complètement autonomes, tels que les réseaux de neurones à apprentissage en temps réel, ont pu être expérimentées avec succès. Même si le pendule inversé peut paraître un cas de figure simpliste des environnements dynamiques réels, il a permis néanmoins de démontrer la faisabilité de l'implémentation des techniques de commandes utilisées avec une carte dSPACE. Ceci étant le but principal de ce travail.

Les aspects théoriques des systèmes de commande et des processus temps réel ont été abordés dans le Chapitre 1.

Le chapitre 2 porte sur le modèle mathématique du pendule inversé. Il a permis de cerner sa complexité théorique. En effet, c'est un système non linéaire, instable en boucle ouverte et sous-actionné. Ces caractéristiques sont généralement représentatifs des problèmes que poseraient des systèmes dynamiques réels. Ceci le qualifie pour les tests unitaires. Par ailleurs il s'agit d'un système qu'on rencontre souvent dans la littérature du domaine de l'automatique.

Les chapitres 3, 4, 5 et 6 abordent les principales catégories d'algorithmes de commande utilisées. Il y est présenté une méthode particulière pour chaque catégorie : (1) La commande LQR pour la catégorie des contrôleurs basés sur la linéarisation des systèmes non linéaires autour d'un point de fonctionnement, (2) La commande ISMC pour la catégorie des contrôleurs qui se basent sur la dynamique non linéaire des systèmes, (3) la commande par réseau de neurones pour la catégorie des contrôleurs basés sur l'intelligence artificielle. Un aspect supplémentaire d'adaptabilité a été abordé dans le chapitre 6 portant sur la commande robuste adaptative par backstepping.

Dans le chapitre 7 une description matérielle et fonctionnelle du produit technologique dSPACE est développée ainsi que son interface graphique de visualisation ControlDesk.

Cet environnement s'est avéré d'une grande efficacité grâce à la facilité d'utilisation et la rapidité de mise en œuvre.

Les cinq techniques de commande en question ont été simulées et implémentées avec succès sur la carte dSPACE 1104. Leur faisabilité est ainsi validée. Le chapitre 8 présente les résultats obtenus. Nous relevons en outre qu'il existe une divergence en terme de performance entre les résultats simulés et les résultats réels. Ceci reflète certainement des insuffisances dans les modèles de l'environnement de fonctionnement. L'utilisation des méthodes simulées pour évaluer des systèmes de commandes notamment dans des conditions plus strictes en termes de robustesse, de fiabilité, de performance dynamique et de sécurité, doit être appuyée par des méthodes appliquées.

Les méthodes implémentées varient en termes de complexité allant du LQR (simple) aux réseaux de neurones réputés intractables dans certains cas. En dépit de la difficulté liée à la conception, toutes les techniques ont été implémentées avec succès. Ceci donne une idée assez précise sur leur tractabilité par une carte de type dSPACE. La difficulté d'implémentation varie selon la complexité de l'algorithme et subit les limitations et inconvénients de Simulink, ce qui rend parfois la tâche difficile. On peut citer par exemple :

- Le problème des contraintes des boucles algébriques qui caractérise l'environnement Simulink.

- L'utilisation de la bibliothèque Simulink est limitée par les blocs de bases, la plupart des outils ne sont pas reconnaissable par la carte dSPACE.

- L'intégration de programmation Matlab dans le modèle Simulink est très limitée par un langage basique et très simplifié du code Matlab.

- La carte dSPACE exige un pas d'échantillonnage fixe, ce qui ne permet pas l'implémentation de certains blocs Simulink qui nécessitent un pas d'échantillonnage variable.

En revanche en utilisant l'environnement global de commande composé de la carte dSPACE, SimulinK, MATLAB et ControlDesk, les algorithmes proposés ont été efficace pour commander en temps réel le pendule inversé, et à accomplir toutes les tâches de commande

exigées. En conséquence, nous pouvons dire que la technologie dSPACE constitue une solution idéale pour la commande en temps réel et que l'objectif de ce travail est ainsi atteint.

Etant donné que l'environnement technique basé sur dSPACE est apte à supporter des processus dynamiques temps réels, de futurs développements peuvent être orientés vers l'utilisation de techniques de l'intelligence artificielle plus poussées. On pourrait citer, entre autres, l'apprentissage non supervisé ou automatique. En effet, cette technique ou d'autre encore plus complexes peuvent être testées afin de supporter des processus dynamiques plus complexes. Nous pensons particulièrement aux systèmes où il est nécessaire de prendre en compte des informations de natures différentes : images, connaissances de l'expert… Sur le plan technique, le système dynamique global basé sur dSPACE, peut être amélioré en ménageant la possibilité d'utiliser des langages plus évolué qui facilitent l'implémentation.

BIBLIOGRAPHIE

[1] Z. Mammeri, « Systèmes temps réel et embarqués, Concepts de base, expression des contraintes temporelles », cours M2P GLRE, université de Toulouse mise-ajour 04-2011.

[2] Mr. Z.Merouane, « Commande des systèmes sous actionnés classe 2 », mémoire de fin d'étude, département d'électromécanique, université de Bordj Bou Arreridj 2011.

[3] Mr. K.Hicham, « Tolérance aux défauts via la méthode backstepping des systèmes non linéaires », mémoire de magistère, département d'automatique, université de Sétif 2012.

[4] I.RABA ARBI & M.S.Bougurlid, « commande par mode glissant d'ordre deux », mémoire d'ingéniorat automatique, université de Sétif 2002.

[5] Reza Olfati-Saber, « Control of underactuated mechanical systems with two degrees of freedom and symmetry », American Control Conference, Massachusetts Institute of Technology (MIT), IEEE Cat. No.00CH36334, page 4092–6, USA 2000.

[6] MULLHAUPT P, « Introduction à la Commande des Systèmes Dynamiques », Cours ME-273 de l'EPFL, 21 décembre 2012.

[7] Nicolas DU LAC, « Prototypage des applications temps réel embarquées », journal ed. Techniques de l'ingénieur ISSN 1632-3831 Référence S8192, INIST CNRS, vol. S3, noS8192, [S8192.1-S8192.11] (10 ref.), Paris France, 2005.

[8] Pierre Ficheux , « Temps réel embarqué & Linux », Livre, 3eme édition, 378 pages, Editions Eyrolles, déc 2010.

[9] Mr. Bakhouch Lemnouer, « Commande par linéarisation entrée sortie du couple du flux de la machine asynchrones », mémoire de magistère, département d'électrotechnique, université Sétif 2009.

[10] Manfred Morari et Evanghelos Zafiriou, « *Robust Process Control* », Livre, Prentice-Hall, USA 1989.

[11] Sorin Olaru, « commande prédictive : (interactions, optimisation, commande) », Cours de JD-JN-MACS, Ecole des JDMACS, Angers, 19-21 mars 2009.

[12] Kemin Zhou, John Comstock Doyle et Keith Glover, « *Robust and optimal control* », Livre, Prentice Hall, Englewood cliffs new jersey 07632, 1995.

[13] Karl Johan Åström et Björn Wittenmark, « *Adaptive control* », Livre, Dover Publications, Addison-Wesley Longman Publishing Co, Inc. Boston, MA, 2e éd.573 p, ISBN:0201558661, USA, 2008.

[14] Alberto Isidori, « *Nonlinear control systems 2* », Livre, Birkhäuser, 3e éd (978-3-540-19916-8, 293 p, Springer edition, 1999.

[15] Jean-Jacques E. Slotine et Weiping Li, « *Applied nonlinear control* », Livre, Prentice Hall, , 459 p, Englewood cliffs, new jersey, 1991.

[16] W. Thomas Miller, Paul John Werbos, Richard S. Sutton Werbos, « Neural networks for control », Livre, MIT Press Massachusetts Institute of Technology, 1995.

[17] Hebertt Sira-Ramirez and Sunil K. Agrawal Marcel Dekker, « *Differentially flat systems* », Livre, 270 Madison Ave New York, 10016-0602. 467pages, NY 2004.

[18] Hicham Chaoui and Pierre Sicard, « Motion and Balance Neural Control of Inverted Pendulums with Nonlinear Friction and Disturbance », article IEEE, Industrial Electronics Research Group CCECE 2011 978-1-14244-9789-8/11/$26, CANADA 2011.

[19] Lopez-Toribio C, Patton R, Daley S , « Takagi–Sugeno fuzzy fault-tolerant control of an induction motor », Journal of Intelligent & Fuzzy Systems 121 ISSN: 0941-0643 1433-3058 21 2000.

[20] Mr. Babbas .walid, « Implémentation de la commande des systèmes non-linéaires sur une carte dSPACE », Mémoire de Master 2, département d'électrotechnique université de Sétif, 2011.

[21] Michael Armata, « linear motion servo plants: IP02 & IP01 User Manual », catalogue, Quanser document 503 119 Spy Crt MARKHAM, Ontario L3R 5H6, Canada, 2005.

[22] Edward .L, « Commande Optimale », Thèse IVRI, Ecole Nationale Supérieure de Physique de Strasbourg, Automatique et Vision, Images, Robotique et Ingénierie pour le Vivant, Fr 2010.

[23] Hongliang Wang Haobin Dong, Lianghua He, Yongle Shi, YuanZhang, « Design and Simulation of LQR Controller with the Linear Inverted Pendulum ». International Conference on Electrical and Control Engineering, 25-27 June 699 - 702 IEEE, China 2010,.

[24] Xianmin Chen, Huixing Zhou, Ronghua Ma, Fuchang Zuo, Guofang Zhai, Minli Gong « Linear Motor Driven Inverted Pendulum and LQR Controller Design », IEEE International Conference on Automation and Logistics 18-21 Aug. 1750 – 1754 , Univ Beijing China 2007.

[25] Plestan F, Shtessel Y, Bregeault V, Poznyak A, «New methodologies for adaptive sliding mode control », International Journal of control Hal-000626498, version 1- 26, Sep 2011.

[26] E. Eryurek, and B.R. Upadhyaya , « An integrated fault tolerant control and diagnostics system for nuclear power plants », Proceedings of the Topical Meeting on Computer Based Human Support Systems : Technology, Methods and Future, IEEE Control Systems Magazine, 15(5):34–42 pp. 267-274, 1995.

[27] K. Zhang, S. Hu, and B. Jiang, « Sliding Mode Integral Observers for Sensor Faults Detection and Isolation in Nonlinear Systems », IEEE International Conference on Control and Automation Guangzhou, CHINA, pp. 147-151, 10.1109 ICCA, 30 Mai - 1 Juin 2007.

[28] Mouloud Bouchoucha, « Conception d'un contrôleur à logique floue basée sur la théorie des modes glissantes », Mémoire de Magister, Ecole Militaire Polytechnique, septembre 1999.

[29] Ahcéne Boubakir, « Méthodes de commande par mode de glissement appliquées au bras manipulateur et au système hydrolique CE105 à réservoirs couplés », mémoire de magister, Ecole militaire polytechnique, février 2007.

[30] D. Theilliol, D. Sauter, and J.C. Ponsart, « A multiple model based approach for fault tolerant control in nonlinear systems », Proceedings of the 5th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes SAFEPROCESS'03, Washington, D.C, pp. 151-156, USA 2003.

[31] J. Tsinias, « Sufficient lyapunov-like conditions for stabilization », Journal of Mathematics of Control, Signals, and Systems, ISSN: 0932-4194 1435-568X Syst. Vol. 2, pp. 343-357, 1989.

[32] V.MINZU & B.LENG, « commande automatique des systèmes linéaires continue », livre, édition ellipses, 2001.

[33] YIN XING HAO, JIAN QIANG YI, DON GBIN ZHAO, DIANWEI QIAN, « Robust control using incremental sliding mode for underactuated systems with mismatched uncertainties », American Control Conference IEEE , Lab. of Complex Syst ISSN : 0743 1619 E-ISBN : 978-1-4244-2079-7, page 532–7, Seattle, WA, 11-13 June 2008.

[34] R. LOZANO, I. FANTONI, D. J. BLOCK . « Stabilization of the inverted pendulum around its homoclinic orbit », article ScienceDirect systems and Control Letters, S0167-6911(00)00025-6, Pages 197–204, 5 July, 2000.

[35] A. M .BLOCH , N. E. LEONARD , AND J. E. MARSDEN, « Controlled Lagrangians and the Stabilization of Mechanical Systems I. The first matching theorem », IEEE Journal of automatic control transactions on, ISSN **:** 0018-9286 INSPEC **:** 6828975, p2253 – 2270, Dec 2000.

[36] Z .SUN,S. Ge, and T. H .Lee, « Stabilization of Underactuated Mechanical Systems: A Nonregular Backstepping Approach », International Journal of Control, DOI**:** 10.1080/002071 70110052248, pages 1045-1051, nov 2001.

[37] MARK W. SPONG, « Partial Feedback Linearization of Underactuated Mechanical Systems » IEEE/RSJ International Conference on Intelligent Robots and Systems, 314 - 321 vol.1 ISBN**:** 0-7803-1933-8, Munich, 12-16 Sep 1996.

[38] Khaled, N. Ofoli, A.R. , « An incremental sliding mode controller (ISMC) for chattering reduction ». Industry Applications Society annual meeting conference IEEE (IAS) ISSN: 0197-2618, 978-1-4244-9498-9, Orlando, FL, 9-13 Oct. 2011.

[39] F. Zebiri Ouad, H. Mehamelle, « Détection et localisation des défauts de la Machine Asynchrone à Double Stator, application de la technique d'intelligence artificielle », Mémoire d'ingéniorat de l'Université Mohamed Boudiaf de M'sila, Algérie, Juin 2006.

[40] Saliou Diof , « Contribution au diagnostic industriel de défauts de roulements et de balourd par techniques neuronales », Thèse Doctorat, génie informatique, automatique et traitement de signal, Université paris XII val de marne-cretail, 2007.

[41] Yann Boniface, Nicolas P. Rougier « Apprentissage et mémoires : Introduction aux réseaux de neurones artificiels », Cours sciences cognitives, Université de Lorraine, Janvier 2012.

[42] Souhil Kouda, « Conception d'un capteur d'humidité intelligent », mémoire de magister, université de Batna, département d'électronique, 2005.

[43] Saliha Boudraa, « Analyse des gaz dissous dans les huiles des transformateurs en utilisant les techniques de l'intelligence artificielle », mémoire de magister, université de Laghouat, département d'électrotechnique, 2005.

[44] Meskin .K, « Identification du potentiel d'action d'une unité motrice : Approche réseaux de neurones », mémoire d'ingéniorat, automatique, université de Ferhat Abbas Sétif, 2006.

[45] LEMMOU Amira- BELLAKHDAR Khaoukha- LEDJEDEL Adila, « Identification et commande des systèmes non linéaires », mémoire d'ingéniorat en électronique, université de M'Sila, 2011.

[46] Tim Callinan, « Artificial neural network identification and control of the inverted pendulum », these doctoral, School of electronic engineering, Dublin 2003.

[47] HICHAM CHAOUI « Implémentation sur FPGA d'une loi de commande adaptative neuronale supervisée pour une articulation », thèse doctorale, Université du Québec en Outaouais, 2004.

[48] Banu RW, ShakilaBanu A, Manoj D « Identification and Control of Nonlinear Systems using Soft Computing Techniques », International Journal of Modeling and Optimization, Vol. 1, No. 1, India, April 2011.

[49] WangWu, Xuchang, « Adaptive Neuro Sliding Mode Control for Inverted Pendulum », 2nd International Conference on Computer Engineering and Technology IEEE, University, School of Electrical and information Engineering Xuchang, supported by natural science research, HeNan Province A510014 V1–135–V1–138, China 2011.

[50] Belhadj Djallel, « Estimation de la force musculaire à partir d'un signal SEMG », Mémoire d'ingéniorat, Automatique, Université Ferhat Abbas Sétif 2009.

[51] X.F. Liu, and A. Dexter, « Fault tolerant supervisory control of a vav air conditioning systems », article ScienceDirect, Energy Buildings, Department of Engineering Science, University of Oxford, OX1 3PJ, Vol. 33, pp. 379-389, Oxford UK, april 2001.

[52] Z. Gao, and P.J. Antsaklis, « reconfigurable control system design via perfecy model-following », International Journal of Control, Taylor & Francis, DOI:10.1080/002071791 08953643 53(3):717–29 Vol. 56, No. 4, pp. 783-798, University of Notre Dame, U.S.A 1992.

[53] Mohamed Riyad Zemouri, « Contribution à la surveillance des systèmes de production : Application à la e-maintenance », thèse doctorat, Université de Franche-Comté, 2003.

[54] C.J. Lopez-Toribio, R.J. Patton, and S. Daley, « A multiple model approach to fault-tolerant control using takagi-sugeno fuzzy modeling: real application to an induction motor drive system », Proceedings of the 3rd European Control Conference ECC, The University of Hull, Cottingham Road, Hull HU6 7RX, UK , 5-9 July 1999.

[55] Abder Rezak Benaskeur, « Aspects de l'application du Backstepping adaptatif à la commande décentralisée des systèmes nonliéaires », thèse PHD, Université du Laval, février 2000.

[56] Shubhobrata Rudra and Ranjit Kumar Barai, March, « Robust Adaptive Backstepping Control of Inverted Pendulum on Cart System » , International Journal of Control and Automation Vol. 5, No. 1, , Department of Electrical Engineering , Jadavpur University, Kolkata-700032, India 2012.

[57] Ö. Tolga Altýnöz, « Adaptive Integral Backstepping Motion Control for Inverted Pendulum », journal waste 5-51 World Academy of Science, Engineering and Technology 29 2007.

[58] Abder Rezak Benaskeur, Louis–Nicolas Paquin and André Desbiens, « Toward Industrial Process Control Applications of the Backstepping », Article du Groupe de Recherche sur les Applications de l'Informatique à l'Industrie mineral, Department of Electrical and Computer Engineering, Université Laval, Sainte-Foy (Québec), G1K 7P4, Canada 2000.

[59] Equipe dSPACE GmbH, « Profil société dSPACE », dSPACE GmbH Rathenaustraße 26 33102 Paderborn, Allemagne, 2013.

[60] Equipe dSPACE GmbH, « DS1104 R&D Controller Board Features », Catalogue, Technologiepark 25 33100, Paderborn Germany 2004.

[61] Equipe dSPACE GmbH , « DS1104 R&D Controller Board RTI Reference », Catalogue, Technologiepark 25 33100, Paderborn Germany 2004.

[62] Equipe dSPACE , « Help Desk »,manuel d'aide électronique, Paderborn, Germany 2004.

[63] Equipe dSPACE, « ControlDesk Reference », dSPACE GmbH Rathenaustraße 26 33102 Paderborn, Allemagne, 2004.

[64] Equipe dSPACE , « ControlDesk Experiment Guide », Test and Experiment Software ControlDesk, Paderborn, Germany 2004.

Résumé : Ce mémoire présente une étude de faisabilité de l'implémentation de plusieurs techniques de commande des systèmes temps réel sur la carte dSPACE 1104. Des algorithmes issus des principales catégories des techniques de commandes ont été développés en utilisant un environnement complet composé de Simulink, MATLAB et ControlDesk : La commande LQR pour la catégorie des contrôleurs basés sur la linéarisation des systèmes non linéaires autour d'un point de fonctionnement, la commande ISMC pour la catégorie des contrôleurs qui se basent sur les dynamiques non linéaires des systèmes, la commande par réseau de neurones pour la catégorie des contrôleurs basés sur l'intelligence artificielle, et la commande robuste adaptative par backstepping, laquelle est développée afin d'étudier l'aspect adaptatif. Chaque algorithme est appliqué sur le système du pendule inversé dans le cadre d'une simulation, implémenté et testé sur une plateforme réelle. Les résultats obtenus sont discutés et comparés.

Mots clés:
Systèmes de commande, Implémentation, dSPACE, pendule inversé, systèmes temps réel, LQR, mode de glissement, réseaux de neurones, Backstepping, ControlDesk, MATLAB, SIMULINK, intelligence artificielle, non linéaire, Simulation.

ملخص: تعرض هذه الوثيقة دراسة امكانية تنفيذ مختلف تقنيات التحكم الآلي للأنظمة الآنية باستعمال تكنولوجيا dSPACE 1104. العديد من الخوارزميات التي تمثل الفئات الرئيسية لمختلف تقنيات التحكم الآلي تم اعدادها باستعمال بيئة التطوير الكاملة والمتكونة من ControlDesk, MATLAB, Simulink.تمثل تقنية LQR فئة أنظمة التحكم التي ترتكز على تحويل الأنظمة الى أنظمة خطية وذلك حول نقطة عمل وتشغيل معينة , تم اعتماد تقنية ISMC من فئة أنظمة التحكم التي تعتمد على الديناميكيات الغير خطية, تمثل تقنية الشبكات العصبية المصطنعة فئة أنظمة التحكم التي تعتمد على الذكاء الاصطناعي و تم إضافة تقنية Backstepping الذاتية التأقلم لأجل دراسة جانب التحكم الذي يتكيف مع بيئة التشغيل باستمرار. كل خوارزمية تم تطبيقها على نظام النواس المقلوب في اطار محاكات وكذلك في اطار تنفيذ و اختبار حقيقي على الواقع. النتائج المحصل عليها تم مناقشتها ومقارنتها.

Abstract: The aim of this work is to study the feasibility of the implementation of different control technics for real time systems using a control board dSPACE 1104. A set of state-of-the-art control algorithms were developed using an environment integrating Simulink, MATLAB and ControlDesk. Each algorithm corresponds to a particular type of control methods: LQR control for the class of controllers based on the linearization of nonlinear systems around an operating point, the ISMC control for the class of controllers that are based on the nonlinear dynamics of the systems, neural network control for the class of controllers based on artificial intelligence, and adaptive backstepping robust control, which was developed precisely to study the adaptive aspect. Each algorithm is applied to the inverted pendulum system in the framework of a simulation, then, implemented and tested on real platform. The results are discussed and compared.

Keywords:
Control system, implementation, dSPACE, inverted pendulum, real time systems, LQR, sliding mode, neural network, Backstepping, ControlDesk, MATLAB, Simulink, artificial intelligence, nonlinear, Simulation.