REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
ECOLE NATIONALE POLYTECHNIQUE
DEPARTEMENT DE GENIE ELECTRIQUE

Mémoire de fin d'études

En vue d'obtention du diplôme d'ingénieur d'état en Automatique

Thème

Commande neuro-floue optimisée par un algorithme génétique, d'un bras manipulateur de robot flexible

Proposé par
M M.LOUDINI

Etudié par
M^elle Nabila BOUNCEUR

Dirigé par
M M.LOUDINI
M M.BOUKHETALA

promotion 2005

ENP 10, Avenue Hassen Badi- El-Harrach - Alger

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
ECOLE NATIONALE POLYTECHNIQUE
DEPARTEMENT DE GENIE ELECTRIQUE

Mémoire de fin d'études

En vue d'obtention du diplôme d'ingénieur d'état en Automatique

Thème

Commande neuro-floue optimisée par un algorithme génétique, d'un bras manipulateur de robot flexible

Proposé par
M M.LOUDINI

Etudié par
Melle Nabila BOUNCEUR

Dirigé par
M M.LOUDINI
M M.BOUKHETALA

promotion 2005

ENP 10, Avenue Hassen Badi- El-Harrach - Alger

Toute ma reconnaissance et remerciements,
A Monsieur Malik LOUDINI, mon promoteur pour la confiance qu'il m'a témoignée et de l'autonomie qu'il m'a donnée.
Pour son suivi de près et de loin, qui m'a permise grâce à sa rigueur scientifique, ses idées et conseils, de bien mener le travail.
A Monsieur Djamel BOUKHETALA, mon co-promoteur de m'avoir inscrite pour travailler sur ce sujet qui m'a tant passionné, et de m'avoir donné une autonomie et témoigné de beaucoup de confiance.
Merci pour vos encouragements et toute l'attention que vous avez eu pour moi.

Aux deux êtres chers à mon cœur, mes parents à qui je dois ma réussite,
A ma très chère sœur, pour sa grande aide et sa présence continue,
A mes très chers frères, pour leur soutien moral,
A tous les membres de ma grande famille maternelle et paternelle,
A tous mes amis, présents de loin ou de prés,
Respectueusement, à tous ceux qui me connaissent.

Glossaire

L : Longueur de la lame droite
E : Module de Yang
I : Inertie de la section
ρ : Masse par unité de longueur
m_c : Masse de la charge terminale considérée comme ponctuelle
J_c : Moment d'inertie de la charge
J_a : Inertie du rotor (articulation – moteur)
C_m : Couple moteur
x : Distance le long de la longueur du bras
θ : Position angulaire rigide (angle de rotation de l'articulation ou du moteur)
α : Position angulaire flexible (angle de rotation d'un point du bras)
J_b : Inertie du bras par rapport à l'axe de l'articulation
$\Phi(x)$: Les déformées modales
w_i : Moment de déformation d'ordre un de la liaison pour le mode i
σ_i : Moment transversal pour le mode i
k_i : Coefficient d'élasticité transversale du mode i
V : Vecteur des coordonnées généralisées
L : Lagrangien donnée par $L = T_T - U$
T_T : Energie cinétique totale du système
T_A : Energie cinétique de l'articulation
T_L : Energie cinétique de la liaison
T_c : Energie cinétique de la charge
U : Energie potentielle
D : Energie dissipative totale du système
Q : Forces généralisées
$w(x,t)$: Déflexion (écartement dû à la flexion) par rapport à la position d'équilibre final.
(X_0, Y_0) : représente le repère absolu
(X_1, Y_1) : est un repère associé au mouvement rigide de la liaison 1
(X_2, Y_2) : est un repère associé au mouvement flexible de la liaison 2
$w_1(x_1)$: est la déflexion d'un point de la liaison exprimée dans le repère (X_1, Y_1)
${}^{i}P_1(x_1)$: Composantes d'un point P de la liaison dans le repère i
${}^{0}P_1(x_1)$: Composantes d'un point P de la liaison dans le repère absolu
$F(x,t)$: Force concentrée au point $P(x)$
$M(x,t)$: Moment concentré au point $P(x)$
${}^{1}r_2$: Position de l'origine du repère 2 par rapport au repère 1
${}^{0}r_2$: Position de l'origine du repère 2 par rapport au repère absolu

sommaire

Introduction générale

« Pour savoir où tu vas, il faut savoir d'où tu viens ! »

Le degré de complexité atteint par les nouvelles techniques productique a entraîné, dès la fin des années 70, un profond changement dans les démarches d'automatisation. L'introduction de robots est devenue indispensable à l'industrie, notamment pour les opérations de soudage ou de peinture. Ceci est dû à leurs performances, à citer l'infatigabilité, l'insensibilité aux environnements hostiles, la répétitivité des tâches, la précision et la rapidité d'action...

Les grandes structures mécaniques, la télé robotique spatiale et la robotique sont des domaines particulièrement intéressés par des problèmes de commande de structures flexibles. Ces effets de flexibilité dans la synthèse de lois de commande performantes sont amenés par l'allégement des structures mécaniques et les performances demandées.

Les avantages d'utilisation de telles structures par rapport à celle rigide, se manifestent par des caractéristiques intéressantes : rapport élevé entre charge et masse propre de la structure, baisse du coût de réalisation, augmentation de la vitesse, amélioration de la manoeuvrabilité, meilleure efficacité d'énergie, sûreté de fonctionnement...

En revanche, on est confronté aux difficultés causées par les bras flexibles de robots manipulateurs, dit aussi souples ou élastiques : systèmes non linéaires à paramètres distribués ayant une dynamique à variation importante en fonction de la charge transportée et de sa configuration, plusieurs modes de vibration faiblement amortis, interaction entre dynamiques rigides et flexible et enfin introduction de couplage supplémentaires entre les mouvements dans le cas de plusieurs degrés de liberté.

La prise en compte des modes souples naturels négligés dans l'étude du robot rigide, dans la commande, introduit des problèmes nouveaux. La non co-localisation de l'actionneur et du capteur est un des problèmes clefs.

Entre l'actionneur et le capteur, on se trouve en présence d'un système dynamique régi par des équations mathématiques, aux dérivées partielles d'ordre infini, ce qui rend la modélisation et l'identification complexe. En effet, les modes propres de vibration sont très sensibles aux variations de charge dans la mesure où les ordres de grandeur des inerties dues au bras et à la charge sont proches. Comme il est difficile de caractériser les amortissements relatifs à chaque mode de vibration naturel.

Afin d'obtenir des performances acceptables, un bon compromis dans le choix du modèle et de la loi de commande doit être réalisé. Les modèles doivent être suffisamment simples, utilisables pour la commande en temps réel. Une bonne commande sous-entend une bonne connaissance des modes naturels et de leur amortissement.

Plusieurs sont les techniques de commande appliquées aux bras flexibles, et toutes nécessitent l'adoption d'un modèle, en général complexe pour le rapprocher du comportement du système réel. Utilisé pour l'ajustement automatique en ligne et en temps réel des régulateurs des boucles de commande, l'ensemble des techniques de la commande adaptative la plus en vogue ces dernières années, permet de réaliser ou de maintenir un certain niveau de performances quand les paramètres du procédé à commander sont inconnus ou varient dans le temps. Répondue en raison de sa flexibilité concernant d'une part le choix de la méthodologie de conception du régulateur (critère quadratique linéaire, variance minimale...) et d'autre part le choix de l'algorithme d'identification (moindre carrée récursif...), cette stratégie dite « conventionnelle » s'appuie crucialement sur une modélisation du processus à commander, ce qui n'est pas toujours évident, surtout dans le cas de système complexe, tel que le bras de robots souples. La quantité d'informations acquises et la complexité numérique des algorithmes d'identification, limitent les performances du système commandé.

La mise en ouvre de lois de commande pose le problème de la robustesse avec des modèles d'ordre réduit. Résoudre un problème de commande de système d'ordre infini, mal connu, variable dans le temps et difficilement modélisable exige l'élaboration de nouvelles stratégies de commande dites « non conventionnelles ». En effet, de nouvelles approches tenant compte des non linéarités et des incertitudes immanentes aux systèmes réels ont été développées.

L'intelligence artificielle (IA) apparaît en automatique pour la commande, lorsque la modélisation classique apparaît trop coûteuse, voir impossible.

On entend par intelligence artificielle la simulation du raisonnement humain pour faire résoudre des problèmes complexes par un calculateur. Concrètement, elle vise à construire un « système formel », composé de symboles, d'expressions construites à partir de ces symboles, et de processus capables de détruire, modifier, créer de nouvelles expressions. Elle doit être vue comme un ensemble de modèles mathématiques et des méthodes informatique pour représenter et utiliser des connaissances humaines.

Une des approches les plus récentes de l'intelligence artificielle vise à simuler non plus les capacités humaines de raisonnement, mais le traitement de l'information dans le cerveau, par un circuit électronique élémentaire, modèle du neurone humain. L'intelligence est reproduite par l'interconnexion de ces circuits élémentaires neuronaux simulés sous forme d'algorithmes, d'où le nom « d'approche connexionniste »

Afin de manipuler des connaissances imparfaites, la logique floue intervient comme un outil performant. Des citations conditionnelles linguistiques du types « Si – Alors » sont utilisées pour résoudre des problèmes de décision (contrôle), ou pour décrire le comportement dynamique d'un système inconnu ou mal défini (identification).

Plusieurs méthodes de commande « expertes » inspirées de l'intelligence artificielle, fondées principalement sur la logique floue sont appliquées aux systèmes complexes. La neuro-floue, la plus attrayante, est naît de l'association de la commande par logique floue aux réseaux de neurones artificiels. L'inconvénient majeur de la commande neuronale étant le temps d'apprentissage très élevé lors de l'initialisation du système sans connaissance, la logique floue permet, à partir d'une connaissance experte linguistique, d'initialiser une commande neuronale de façon beaucoup plus efficace, en proposant un point de départ pas trop éloigné de la commande finale.

Afin d'optimiser l'algorithme de la commande neuro-floue, les algorithmes génétiques basés sur le mécanisme de sélection naturelle et génétique, sont utilisés. Ainsi, la partie de deffuzzification du contrôleur neuro-floue est reconstruite à base de ces algorithmes.

L'objectif du présent travail est l'adoption de cette nouvelle génération de contrôleurs pour contribuer au mouvement de recherche de stratégie de commande adéquate et appropriée au bras manipulateur de robots flexibles.

Des généralités sur les bras manipulateurs de robots flexibles, précisant les différentes méthodes existantes pour leur modélisation et les stratégies de commandes appliquées à ce jour à de tels bras, sont présentées dans le premier chapitre.

Le second chapitre est consacré à la modélisation de notre bras à une liaison flexible en utilisant l'approche modale.

La Méthodologie du contrôleur flou a été montré dans le troisième chapitre en introduisant le principe de la logique floue.

Le quatrième chapitre a été consacré à la commande neuro-floue du bras flexible, tout en présentant la méthodologie de modélisation neuronale comme nouvelle méthode de présentation de fonctions complexes. L'algorithme utilisé étant celui de Back- propagation.

La commande neuro-floue à base d'algorithmes génétiques appliquée au robot flexible a fait l'objet du dernier chapitre. Les algorithmes génétiques ont été introduits ainsi que leur aspect de programmation et leur rôle principal pour l'optimisation du modèle flou.

Et enfin, une conclusion générale, dans laquelle est présentée l'ensemble des perspectives et les difficultés rencontrées lors du travail.

1

Généralités sur La modélisation et commande des bras manipulateurs de robots flexibles

Introduction

Le problème de commande exige la recherche d'un modèle exploitable. Il est illusoire de rechercher un modèle parfait, qui, s'il existait nécessiterait l'utilisation d'un nombre trop grand de paramètres. L'utilisation de la commande en boucle ouverte dans le cas des structures flexibles est rendue impossible, selon un modèle affecté par de multiples incertitudes qui excitent une ou plusieurs résonances mécaniques, ce qui nous mène à nous placer dans le cadre de la commande en boucle fermée.

En plus de perturbations qui se manifestent dans le cadre linéaire, il existe par ailleurs de nombreuses sources de non linéarités, dont certaines sont difficilement modélisables. Elles peuvent être dues au matériau lui-même, ou à des phénomènes mécaniques complexes qui existent y compris à l'intérieur du domaine de déformation élastique. Elles peuvent aussi être liées directement au dispositif flexible, dont les changements de configuration peuvent se traduire par un couplage non-linéaire des variables introduites pour décrire son comportement. Dans tout les cas on aboutit à un modèle de connaissance non-linéaire pour lequel, il est difficile, voire impossible, d'utiliser les méthodes de synthèse linéaires qui constituent à ce jour l'essentiel du savoir faire en automatique.

Deux approches sont alors à considérer. Assez différentes l'une de l'autre, elles ne se traitent pas de la même manière en commande. Dans l'approche multimodèles, plusieurs modèles du système flexible dont chacun restitue une ou plusieurs caractéristiques importantes du systèmes, sont possibles par linéarisation autour d'un point de fonctionnement donné. La difficulté étant de limiter leur nombre pour ne retenir que les plus représentatifs.

L'autre approche, plus souvent utilisée, consiste à associer à un modèle élémentaire un ensemble d'incertitudes, celles-ci étant autant dues à l'impossibilité de décrire complètement le système qu'à la volonté de ne pas surcharger la représentation.

C'est à une représentation ensembliste que l'on aboutit, on peut toujours, en partant d'un modèle nominal et des incertitudes qui l'affectent, exprimer plusieurs modèles du même système. Inversement, il est possible d'extrapoler une structure d'incertitude à partir de différents modèles. Un compromis entre précision et simplicité est à trouvé, afin de pouvoir calculer une loi de commande performante et réalisable [VIN96].

Incertitude non structurée et incertitude structurée

Principalement les performances sont limitées par la qualité du modèle ou la capacité de la commande à prendre en compte convenablement les incertitudes de modélisation ou par la complexité de la loi synthétisée et la puissance de calcul des processeurs permettant sa mise en œuvre sur le système physique.

Deux types d'incertitudes sont à considérer :

- Incertitude non structurée, liée à l'existence d'une dynamique non représentée dans le modèle, qui conduit au risque de « Spillover ». Passer d'un ordre infini à un ordre réduit mène à l'excitation des modes résiduels (ceux qui ne figurent pas dans le modèle réduit), excitation par l'action de la commande (contrôle Spillover) et à une contamination des signaux des capteur (observateur Spillover)
- Incertitude structurée aussi appelée paramétrique, liée aux inexactitudes paramétriques du modèle.

Représentation des incertitudes [VIN96, RYU00]

Le premier type d'incertitude, généralement haute fréquence, s'exprime bien dans le cadre fréquentiel lorsque le modèle est décrit par une matrice de transfert $G(s)$.

Selon les cas, on peut l'écrire sous forme

Additive $G(s) = G_0(s) + \Delta_{add}(s)$ (I.14)

Ou multiplicative $G(s) = G_0(s)[I + \Delta_{mul}(s)]$ (I.15)

Où $G_0(s)$ est le modèle nominal du système/.

Cette incertitude est dite non structurée au sens où l'on connaît rien de sa phase et que la seule information dont on dispose est une borne de sa norme.

Le second type d'incertitude concerne directement la partie modélisée de la dynamique du système et apparaît dans la bande passante que l'on désire commander. Ces perturbations dites paramétriques ou structurées doivent apparaître dans le modèle pour les prendre en compte dans la synthèse d'une li. Elle s'expriment plus facilement lorsque le système est sous forme d'état

$$\begin{cases} \dot{x} = Ax + bu \\ y = Cx + Du \end{cases} \quad (1.16)$$

avec $$\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} A_0 & B_0 \\ C_0 & D_0 \end{bmatrix} + \begin{bmatrix} \Delta A & \Delta B \\ \Delta C & \Delta D \end{bmatrix} \quad (1.17)$$

où (A_0, B_0, C_0, D_0) est la représentation nominale du système.

La synthèse robuste vis-à-vis le premier type d'incertitude, conduit à une commande en gain du système, tandis qu'elle conduit à un contrôle de la phase pour le second type. Ces deux types de perturbation sont complémentaires et coexistent, et devrait être traité simultanément.

Nécessité de la robustesse

Un écart plus au moins important entre le système physique et le modèle est subsisté soit parce que:

- il existe diverses non-linéarités qui n'ont pu être prises en compte dans la modélisation
- la dynamique des capteurs et actionneurs, ou celle du système mécanique lui-même, n'a pu être complètement mesurée ou exprimée, notamment à très basses ou à haute fréquence
- même sur la plage de fréquence où les modes sont convenablement modélisés, certaines singularités telles que les zéros de transmission ne sont pas correctement restituées.

Le niveau de performance atteint en boucle fermée dépend de la capacité à décrire les incertitudes, dues aux imperfections des techniques de modélisation et à l'impossibilité de conserver des modèles d'ordre élevé [BOI88, VIN96], et à les prendre en compte au mieux dans la synthèse. La commande doit donc nécessairement être robuste vis-à-vis des erreurs de modélisation.

Robustesse en stabilité et robustesse en performance

Le but d'une synthèse robuste est le calcul d'un compensateur qui assure la stabilité du système en boucle fermée, et garantie un niveau de performance donnée pour le système nominal seul, ou pour l'ensemble des systèmes perturbés. Cette distinction sur l'objectif de la compensation conduit à définir deux types de robustesse :

- On parle de système robuste en stabilité si ce système demeure stable en présence d'incertitude telles que les erreurs de modélisation, les bruits de mesures ou les perturbations externes ;
- Un système est robuste en performance si les performances sont conservées en dépit des perturbations (les performances pouvant se mesurer en terme de temps de réponse du système, d'amortissement des modes flexibles, de découplage perturbation/sorties...etc)

Ce dernier type de robustesse contient bien entendu le précédent, et l'on conçoit aisément que l'objectif de la synthèse robuste devra être d'assurer autant que faire se peut la robustesse en performance plutôt que la simple stabilité [BOI88, VIN96].

I.1. Modélisation des robots flexibles :

L'objectif du travail n'étant pas de traiter la représentation des systèmes flexibles, nous nous contenterons de donner les étapes qui conduisent à une représentation générale, et nous ne passerons en revues que les plus utilisées. Toutefois, nous exposerons ces méthodes existantes dans la littérature [LOU 97] :

- Méthodes de construction des graphes adjoints
- Méthodes des masses concentrées
- Méthode de la perturbation singulière
- Méthode par programmation symbolique
- Méthode des bonds-graphs
- Méthode des éléments finis
- Approche utilisant l'algèbre de Lie
- Approche utilisant le vecteur de rotation fini et les quaternions
- Approche utilisant le principe de moindre courbure

- Approche fréquentielle
- Approche Modale
- Approche Acoustique

De manière générale les méthodes de modélisation des robots flexibles sont une combinaison des méthodes de modélisation des robots rigides pour décrire les mouvements rigides de la structure qui sont supposés connus et en général de grande amplitude, et des méthodes d'analyse des structures flexibles, méthode des éléments finis et analyse modale, pour la description des déplacements élastiques en général de faibles amplitudes [VIN96, LOU97]

Une des méthodes récentes appliquée à un manipulateur flexible est celle dite « descriptive » (decriptor form representation) [RYU00]. Elle est la mieux adaptée pour la représentation des équations différentielles de la dynamique du système. Obtenue par l'association de l'approche multimodèle à une technique de synthèse conventionnelle, cette méthode nous évite l'inversion de la matrice d'inertie du fait que les incertitudes dans la matrice inertielle sont traitées par l'approche multimodèle séparément des paramètres incertains dans les autres matrices, traités par une technique différente la H_∞ par exemple.

Dans ce qui suit, nous présentons les fondamentaux de la modélisation, suivi par l'étude détaillée de la méthode la plus en vogue : l'approche modale. En dernier nous donnerons un aperçu sur l'approche acoustique et les méthodes discrètes dont la méthode des éléments finis.

I.1.1. Fondamentaux de la modélisation [GOM92, VIN96]

Nous présenterons dans cette partie les étapes fondamentales qui mènent à une représentation générale.

Le point de départ de l'ensemble des approches de modélisation est l'application du principe d'Hamilton, qui lie l'énergie cinétique T au travail virtuel des forces internes et externes W :

$$\delta\left(\int_{t_1}^{t_2}(T+W)dt\right)=0 \qquad (I.1)$$

W s'exprime comme la somme de l'énergie potentielle de déformation élastique V et du travail des forces extérieures. En explicitant chacun des termes de (1.1) ; on obtient directement l'expression générale liant à tout instant le déplacement w en un point P quelconque de la structure aux forces extérieures. Dans le cas d'un système sans dissipation, w vérifie :

$$M(P)\frac{\partial^2 w}{\partial t^2}+L(w)=f(P,t)+\sum_j F_j(t)\delta(P-P_j) \qquad (I.2)$$

Où M est la masse en P, f représente les efforts répartis, F_j les forces ponctuelles aux points P_j, et où L est un opérateur différentiel homogène de la forme :

$$L(w)=A_0 w+A_1\frac{\partial w}{\partial x}+A_2\frac{\partial w}{\partial y}+A_3\frac{\partial w}{\partial z}+A_{11}\frac{\partial^2 w}{\partial x^2}+\cdots \qquad (I.3)$$

En plus du chargement extérieur, le système est monté sur un support ou lié à un autre système flexible par l'intermédiaire de liaisons mécanique qui définissent un ensemble de conditions aux limites. Celles-ci s'expriment à l'aide de l équations qui lient les déplacements de certains points du système, et s'écrivent :

$$B_i(w) = 0 \text{ pour } i = 1, \cdots l \tag{I.4}$$

Les équations (I.2) à (I.4) décrivent un problème aux limites, et suffisent en fait à décrire entièrement le système flexible dans son domaine linéaire. Elles constituent donc le modèle dynamique le plus général d'une structure mécanique. Cette représentation est toutefois trop complexe pour être exploitable, et il convient donc de la transformer pour la simplifier.

I.2. Approches principales [VIN96, GOM92]

I.2.1. Approche modale :

L'approche la plus privilégiée est sans doute l'approche modale, qui conduit à une représentation d'état du système et permet ainsi l'utilisation directe des outils classiques de l'automatique. Elle repose sur l'hypothèse que le système est peu dissipatif, ce qui est généralement vrai pour la plupart des matériaux utilisés en construction mécanique (métaux divers, composites carbones etc..).Le problème aux limites décrit par (I.2) et (I.4), où aucun terme d'amortissement n'apparaît, peut être considéré comme une représentation convenable du système.

Dans le cas des structures, on peut montrer que l'opérateur homogène $L(w)$ est auto-adjoint ; on peut alors appliquer le principe de séparation et écrire le déplacement w au point P sous la forme :

$$w(P,t) = \Phi(w)q(t) \tag{I.5}$$

Pour aboutir à un problème de valeurs propres généralisé qui s'écrit :

$$L(w) = \lambda M w = \omega^2 M w \tag{I.6}$$

Si l'on note Φ_r, appelée déformée modale, la solution de (I.6) associée à la valeur propre ω_r^2, le déplacement w peut s'écrire sous la forme d'une somme infinie :

$$w(P,t) = \sum_{r=1}^{\infty} \Phi_r(P) q_r(t) \tag{I.7}$$

Le déplacement w est donc la somme de fonctions spatiale (les déformées) modulées par des fonctions temporelles périodiques. Cette expression décrit donc bien le comportement vibratoire des structures mécaniques, dont le mouvement en l'absence d'amortissement est la superposition d'une infinité d'oscillations à la pulsation ω_r autour de la déformée Φ_r.

Une propriété importante des déformées Φ_r est qu'elles sont deux à deux orthogonales. q_r, dite coordonnée modale généralisée, est solution de l'équation :

$$\ddot{q}_r + \omega_r^2 q_r = N_r(t) \text{ avec } N_r(t) = \int_D \Phi_r(t) f(P,t) dD + \sum_j \Phi_r(t) F_j(t) \tag{I.8}$$

Pour aboutir à une représentation d'état, le modèle donné en (I.7) d'ordre infini doit être tronqué. En ne retenant que n modes significatifs, l'expression de w devient :

$$w(P,t) = \sum_{r=1}^{n} \Phi_r(P) q_r(t) \tag{I.9}$$

Dans le cas général, il n'existe pas d'écriture analytique des déformées modales, et donc des pulsations, car les expressions (I.6) à (I.8) conduisent à des équations transcendantes (d'ordre supérieur). On peut alors utiliser des fonctions modales approchées, qui ne sont pas solutions

du problème aux limites, mais qui vérifient (I.2) et satisfont les conditions aux limites géométriques. Cette approche, appelée « méthode des modes supposés », suffit dans bon nombre de cas à donner un modèle acceptable de la structure, et convient surtout pour la modélisation de structures simples et continues.

Que l'on ait recours à une méthode exacte ou approchée, l'expression tronquée de w reste globalement la même, et permet de passer d'un modèle continu infini à un modèle discret fini du « second ordre », c'est-à-dire une représentation où apparaissent explicitement les dérivées première et seconde du paramétrage. En effet, en remplaçant le déplacement w donné par (I.9) dans l'expression des énergies cinétique et potentielle, on fait apparaître les matrices de masse M et de raideur K de la structure qui vérifient :

$$T(t) = \dot{q}(t)^T M\dot{q}(t) \tag{I.10}$$

$$V(t) = q(t)^T Kq(t) \tag{I.11}$$

avec :

$q^T = [q_1, q_2, \cdots q_n], M > 0, K \geq 0$

Par application des équations de Lagrange, on obtient alors directement une représentation du second ordre.

Cette représentation est particulièrement intéressante puisqu'elle permet de faire apparaître explicitement les modes de vibrations.

Mais elle présente toutefois un risque, puisque les modes éliminés ne peuvent être pris en compte par la suite lors de la synthèse d'une loi de commande. Si c'est modes ne sont pas suffisamment amortis, ils peuvent être excités par une perturbation extérieure ou par la commande elle même, d'où une dégradation des performances en boucle fermée, soit même être déstabilisés. Ce phénomène de contamination, ou spillover, constitue l'une des faiblesses majeures de la modélisation par approche modale, et devra bien entendu être traité lors de la phase commande.

I.2.2. Approche acoustique :

La méthode acoustique permet de résoudre le problème de contamination en exploitant l'équation (I.2) directement qui décrit la propagation d'une onde acoustique au sein de la structure, et évite donc le problème posé par la troncature. Elle paraît aussi plus rigoureuse que l'approche modale, puisqu'elle n'utilise pas le principe de séparation qui n'est valable que si la structure considérée est faiblement amortie. Or, la réalisation par la commande des objectifs de performance passe généralement par une augmentation sensible de l'amortissement des modes de vibration. Il n'est donc pas évident que le modèle obtenu par application du principe de séparation reste valide lorsque le système est rebouclé. L'approche acoustique est en revanche toujours valable, y compris pour les matériaux viscoélastiques à fort coefficient de dissipation.

Cette méthode permet de simplifier al modélisation, mai elle se heurte toutefois à des obstacles majeurs :

- Il est difficile, voire impossible de décrire un système complexe sous la forme de l'équation de propagation (I.2)
- Il existe peu d'outils de synthèse dans le cadre acoustique
- Les correcteurs calculés sont souvent d'ordre fractionnels, et sont difficilement réalisables.

Les correcteurs synthétisés dans ce cas qui visent à maximiser l'atténuation de l'onde de vibration qui se propage dans la structure, sont fondamentalement non causaux, ce qui rend leur réalisation pas possible dans le cadre modal, mais qui le devient en théorie lorsqu'on considère le problème sous l'angle acoustique, puisque la mesure de l'onde en un endroit du système et à un instant donné permet de prédire le comportement en tout autre point et tout autre instant.

I.2.3. Approches discrètes :

En décomposant la structure en sous-systèmes élémentaires pour lesquels il existe une représentation très simple et de degré faible obtenue de façon exacte ou approchée par approche analytique, un modèle d'ordre fini est directement obtenu.

L'intérêt principal de ces méthodes est bien entendu qu'elles permettent de décrire des structures complexes.

La méthode des éléments finis, la plus connue, utilise une approximation polynomiale de degré minimal des déformées de systèmes élémentaires tels que les poutres, les éléments de plaque plane, de coque, ou de volume. Elle satisfait les conditions aux limites géométriques mais ne respecte pas la continuité des conditions du second ordre (moments de flexion) en raison du faible degré des éléments. Réduire l'erreur qui en résulte rend encor plus complexe un modèle d'ordre déjà élevé, en augmentant sensiblement le nombre d'éléments utilisés.

I.3. Modélisation et Identification :

Notre travail se base sur une étude de modélisation, toutefois il faut noter que sans une étape complémentaire d'identification, la représentation complète du système considéré ne sera atteinte.

Une des approches à utiliser consiste, en partant d'un modèle approché, à essayer de le recaler, même partiellement, à partir des données d'identification. Dans un second temps, on utilise ce modèle pour estimer les variables non mesurables et prédire le comportement du système dans d'autres conditions d'essais différentes de celles du travail, quand l'identification complète du système est impossible par ces raisons.

Identification et Modélisation se complètent, mais la modélisation reste indispensable pour dimensionner ou optimiser une structure, en permettant par exemple de mieux gérer la répartition de matière dans la structure. Elle peut servir particulièrement à détecter la présence des modes locaux qui peuvent interagir avec le spectre des perturbations, et qui, étant faiblement commandables, risquent de dégrader les performances.

I.4. Commande des robots flexibles :

Il est difficile de donner une vision complète de l'ensemble des techniques développées dans le domaine de la commande des structures flexibles voyant sa richesse. Nous ne nous intéressons qu'aux méthodes floues qui feront l'objet d'étude du présent travail, nous passerons en revu d'abord les méthodes robustes qui désensibilisent la loi aux erreurs de modélisation. Nous signalons tout de même l'existence des approches élémentaires telles que celle fondées sur la commande par retour proportionnel - intégral - dérivé (PID), et des techniques fondées sur la commande adaptative qui est une solution intéressante au problème de vibration mais qui n'illustre pas de façon fondamentale les problèmes propres à la commande de flexibilités [VIN96].

I.4.1. Méthodes optimales ou robustes

Les différentes méthodes de commande robuste des structures flexibles considèrent la présence des modes flexibles comme un phénomène perturbateur et non comme une dynamique voulue. Le problème de performance dans ce cas est relatif à la commande des modes rigides et à la rejection de perturbations.
Contrôler les vibrations ou même contrôler les déformées des systèmes flexibles n'est pas l'objectif des commandes robustes [ALA02]. Si la loi de commande permet de plus d'amortir les modes flexibles, alors elle sera d'autant plus satisfaisante mais, ceci reste un objectif secondaire.
La synthèse quadratique robuste est l'une des méthodes utilisées pour la synthèse des flexibilités, en rendant robuste aux variations paramétriques du modèle, des lois obtenues pas synthèse LQG .[GOM92, VIN96, ALA99]

Dans [JEA99] une technique adaptative a été appliquée. Elle permet de rafraîchir le correcteur selon le résultat d'une identification en ligne des paramètres soumis à des variations. Un contrôleur classique assure les performances sur le modèle réduit aux modes rigides, tandis qu'un contrôleur adaptatif restreint au modèle souple assure robustesse et suppression des oscillations internes.

La synthèse LQG est utilisée pour simplifier l'adaptation et l'optimisation du choix du signal de détection dans cette commande.

Une commande basée sur la forme LQ a été appliquée au modèle linéaire d'un bras manipulateur à une liaison flexible dans [RYU00]. Le modèle étant représenté par la méthode « descriptive » (voire partie modélisation)

Le cadre le plus favorable au traitement des incertitudes non structurées est celui des méthodologies de synthèse H_∞ et H_2 qui consistent, sous leur forme la plus simple, à éjecter les perturbations en minimisant leur effet [VIN96, MAH04, ALA99, ALA01]

Dans les structures flexibles, les incertitudes paramétriques et non structurées coexistent. La « mesure structurée » est l'un des outils d'analyse et de synthèse qui permet de les traiter simultanément tout en minimisant le conservatisme introduit par la présence de plusieurs blocs d'incertitudes [VIN96, ALA99, ALA02].

I.4.2. Méthodes floues

De nouvelles stratégies de régulation dans les techniques d'automatisation et du contrôle sont introduites pour l'amélioration des performances des processus industriels et la réduction des coûts. Elles ont pour objectif l'accroissement de la robustesse, le contrôle non linéaire et le contrôle adaptatif.

La logique floue (LF) a été introduite en 1965 par **L.A. Zadeh**. Ses principes ont été appliqués en 1974 par E.H. Mamdani à la construction d'un premier contrôleur flou. [GEN97, BOU02]

La première application industrielle à bas de régulation floue a été mise en œuvre par M. Sugeno en 1985. La logique floue a été élargie aux systèmes à réseaux de neurones et à l'intelligence artificielle en 1995 par J. S. R. Jang.

La commande floue constitue un complément précieux dans le cas de systèmes difficilement identifiables ou dont les paramètres subissent des variations brutales. Les résultats obtenus avec un contrôleur flou sont meilleurs que ceux obtenus avec des algorithmes de contrôle conventionnel dans le cas des structures flexibles, complexes à analyser par des techniques quantitatives ou quand les sources d'information sont jugées sans précision ou incertitudes [BAR96, LOU97, MAM93].

Beaucoup d'applications sont nées au Japon sous l'impulsion du *Laboratory for International Research on Fuzzy Engineering* (LIFE) à Yokohama et aussi du *Fuzzy Logic Institute.* Les premières applications industrielles ont également été développées au Japon. Actuellement, cette technique s'est élargie dans le monde.

Des exemples de contrôle flou sont illustrés dans. On y trouve par exemple le contrôle d'une rame de métro et la purification de l'eau. Dans [GUI99] une illustration détaillée d'un contrôleur de stores a été présentée.

L'application des méthodes floues, multi-échele et hybride (PD+PI), sur un bras de robot à une liaison flexible a fait l'objet d'étude de [LOU97]. Il a été montré que la méthode hybride donne de meilleurs résultats.

On trouve dans [VAN00], une méthode floue applicable aux systèmes non-linéaires et à minimum de phase: la régulation directe adaptative. Minimiser une fonction de coût avec un algorithme d'apprentissage de descente du gradient est son principe.

Un des avantages des algorithmes adaptatifs flous est leur initialisation aléatoire, et leur capacité à résorber des perturbations accidentelles grâce à leur faculté d'apprentissage constant.

Dans [KHA 01], la commande adaptative floue a été appliquée à des systèmes multivariables incertains et non linéaires. Dans cette étude, le principe est l'obtention d'un vecteur de fonction floue, pour approximer les incertitudes du système linéarisé, et d'utilisation de sa dynamique connu pour concevoir un contrôleur par retour d'état pour le stabiliser.

La facilité d'initialisation d'algorithme dans la commande floue ne se trouve pas dans les réseaux artificiels neuronaux (RNA). En revanche ces derniers permettent, pour des raisons de modélisation et d'apprentissages en ligne de processus, de présenter des relations fonctionnelles complexes difficiles à décrire sous une forme analytique de systèmes variables en fonction du temps et donc appréhender trois sources principales de difficultés en contrôle de procédés : complexité, non-linéarité et incertitude [SOR99]. Ils sont les mieux adaptés à la modélisation des structures flexibles.

La commande neuronale est utilisée pour la régulation industrielle [SOR99, RIV95] et bien d'autres domaines. Dans le domaine du contrôle on trouve des exemples en robotique dans [KRO 96] et en contrôle moteur dans [COU02]

Il résulte de l'association de la commande par logique floue aux réseaux de neurones artificiels une commande neuro-floue qui tire profit des capacités d'apprentissage des RNA et de celles du raisonnement approché de la LF [GEN97, FOS93]. Plusieurs sont les modèles flous utilisant les réseaux neuronaux. Hayachi [HAY98] et son équipe de chercheurs japonais, les ont classé en onze catégories, selon le degré de fusion de la LF par rapport aux RNA.

L'algorithme des commandes mentionnées jusqu'à lors est basé sur l'optimisation d'un critère donné en général d'erreur floue. L'algorithme de descente du gradient de 'Back- propagation' utilisé peut tomber sur un optimum local. Pour cela, la recherche de la meilleure technique d'optimisation a fait appel à l'utilisation de nouvelles méthodes heuristiques du domaine de l'Intelligence artificielle.

Les Algorithmes génétiques (AG) sont des techniques robustes d'optimisation de recherche d'un optimum global, qui imitent le mécanisme naturel de l'évolution des êtres vivants. C'est une des méthodes de du calcul évolutionnaire, utilisés pour résoudre des problèmes complexes [DAB05].

Beaucoup de travaux sont nés de l'association des AG à la logique floue d'une part [MAR00, KIM94, POP96] et aux réseaux de neurones d'autre part [HEI92], et ont été appliqués aux structures flexibles [AKB98, LAM01]. Six catégories naissent de ces combinaisons [HAY98].

La fusion des trois techniques de l'intelligence artificielle, la théorie de la logique floue, les réseaux de neurones artificiels et les algorithmes génétiques nous mène à considérer de nouvelles approches, utilisées pour la résolution des problèmes associés aux structures flexibles, et qui ont donné de bons résultats en simulation [JIK97, ZHO00, FAR98, TUN95]

I.4.3. Etude en simulation

Les problèmes les plus importants du contrôle sont relatifs :

- à un positionnement rapide et précis de l'extrémité de préhension du bras en éliminant ou limitant les oscillations résiduelles (commande point à point)
- aux effets de variation des charges et de la configuration du manipulateur.

De ce fait, l'algorithme de contrôle devrait répondre aux exigences suivantes :

- Commande stable
- Rapidité de l'exécution pour contrôler, le mouvement rapide du bras (commande en temps réel)
- Robustesse aux changements du comportement du manipulateur dus aux variations de configurations ou chargement
- aptitude de rejet de perturbations.

Ainsi, un ensemble de test sera effectuer sur la commande neuro-floue, appliquée au bras du robot à une liaison flexible en se basant sur des tests de charge. Le logiciel de simulation utilisé est MATLAB6p1.

Conclusion

Dans ce premier chapitre, nous avons présenté en général les méthodes de modélisation et de commande des structures flexibles, en se basant principalement sur la bibliographie réunie, et avons donné le détail de la théorie de la méthode modale pour la modélisation du robot flexible suivi dans la thèse.

Nous avons rappelé les problèmes essentiels posés par la commande des vibrations, qui sont liés en grande partie aux faiblesses des techniques de modélisation. Celles-ci conduisent à des incertitudes importantes tant sur la fréquence et l'amortissement des modes que sur la position des zéros de transmission, et imposent le recours à de méthodologies de synthèse robuste.

Le rôle des commandes robustes n'étant pas principalement le contrôle des vibrations, de nouvelles stratégies plus appropriées de l'intelligence artificielle ont été introduites.

Ces dernières basées sur des règles floue, plusieurs commandes appliquées aux flexibilités sont apparues ces dernières années en se basant sur la modélisation neuronale et optimisées avec les algorithmes génétiques. De bons résultats en simulation ont été obtenus.

2

Modélisation du bras de robot manipulateur à une liaison flexible

Introduction

Dans ce chapitre nous traitons la modélisation des bras flexibles. Pour cela, il nous faut combiner les méthodes d'analyse modale qui décrivent les flexibilités à celles habituellement utilisées aux robots rigides, les liaisons étant modélisées par des poutres bidimensionnelles d'Euler Bernoulli ou tridimensionnelle de Timoshenko.

Nous nous intéressons à la modélisation d'un bras à une liaison flexible, ou plus exactement d'une charge flexible : une lame souple à l'extrémité de laquelle se trouve une masse rigide.

Nous développons en premier le modèle dynamique du robot, en nous appuyant sur le formalisme Lagrangien associé à la méthode des modes supposés. La liaison étant considérée comme une poutre bidimensionnelle d'Euler Bernoulli, le mouvement étant planaire.

En second, une simulation est effectuée en boucle ouverte sur le modèle non linéaire du bras mettant en considération le comportement vibratoire de son extrémité.

Cette étude s'est effectuée en cinq alinéas. Les caractéristiques du bras en tant que structure physique ainsi que sa présentation sont données dans le premier paragraphe. Le second paragraphe traite l'étude mécanique du bras et qui consiste la première étape vers la modélisation dynamique. L'étude énergétique fait l'objet du troisième paragraphe. Le quatrième paragraphe, comporte une étude par simulation du modèle dynamique non linéaire du bras, en boucle ouverte afin de mettre en évidence son comportement dynamique

II.1. Description du bras

Le bras du robot considéré est à une articulation flexible illustré par la figure (II.1)
Son déplacement est défini par la somme de deux angles l'un dû au couple appliqué et le second est dû aux flexibilités. Comme s'était montré dans le chapitre précédent le déplacement est la somme de fonctions spatiales (les déformées) modulées par des fonctions temporelles périodiques, ce qui décrit le comportement vibratoire de la structure mécanique.

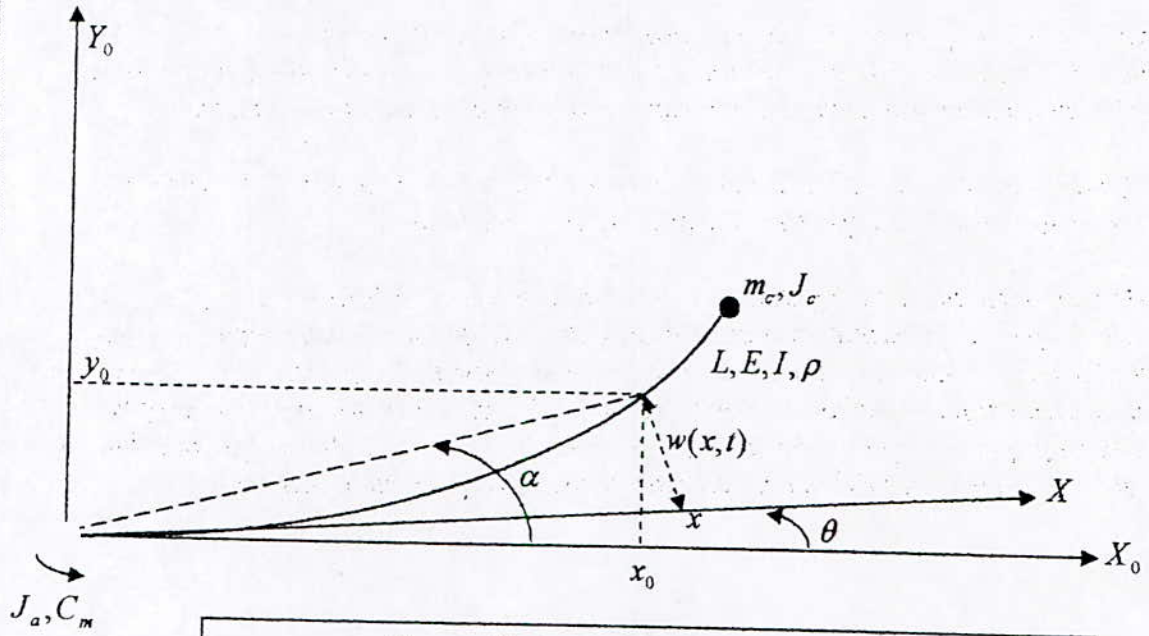


***Figure (II.1)* : bras flexible à une liaison**

Cette figure montre un repère inertiel plan (X, Y) et un axe X tangent à l'origine de la fibre moyenne, où se trouve le rotor. Cet axe tourne avec la vitesse $\dot{\vartheta}(t)$ (mode rigide)

Paramètres caractéristiques

L : Longueur de la lame droite
E : Module de Yang
I : Inertie de la section
ρ : Masse par unité de longueur
m_c : Masse de la charge terminale considérée comme ponctuelle
J_c : Moment d'inertie de la charge
J_a : Inertie du rotor (articulation – moteur)
C_m : Couple moteur
x : Distance le long de la longueur du bras
θ : Position angulaire rigide (angle de rotation de l'articulation ou du moteur)
α : Position angulaire flexible (angle de rotation d'un point du bras)
$w(x,t)$: Déflexion (écartement dû à la flexion) par rapport à la position d'équilibre final. C'est la position de la fibre moyenne sur un point P de l'axe.

Hypothèses

Un ensemble d'hypothèses caractérisant la théorie des poutres d'Euler Bernoulli ont été adoptées :

1. les paramètres caractéristiques sont constants et donc la lame est considérée homogène
2. le bras est rigide par rapport aux forces axiales, à la torsion, aux forces de flexion verticales dues à la gravité
3. seules les déformations élastiques sont présentés
4. le mouvements est transmis à la lame directement (il n'y a pas de réducteur de vitesse)
5. la flexion est plane : la déflexion d'une section le long du bras est due seulement à la flexion et non au cisaillement
6. la dimension de la section droite du bras est négligeable devant sa longueur

II.2. Etude mécanique

L'application des lois de la dynamique des systèmes mécaniques et de la résistance des matériaux sur une portion infinitésimale le long du bras flexible, en négligeant les efforts de l'inertie de rotation, des forces axiales et du cisaillement, mène à la célèbre équation de la poutre prismatique d'Euler Bernoulli

$$EI\frac{\partial^4 w(x,t)}{\partial x^4}+\rho\cdot L^2\cdot\frac{\partial^2 w(x,t)}{\partial t^2}=0 \qquad \text{(II.1)}$$

L'approche modale nous permet de résoudre une telle équation par séparation des variables espace-temps

$$w(x,t)=\Phi(x)\cdot q(t) \qquad \text{(II.2)}$$

Cette approche se base principalement sur la modélisation mécanique du solide donnée en (II.1) nécessaire à la modélisation des flexibilités.
Le résultat final étant l'obtention de coordonnées généralisées qui seront incluses dans le calcul énergétique. Ces coordonnées peuvent décrire des variables physiques ou être fictives.

Calcul des différents modes de vibration

La séparation souhaitée est obtenue en remplaçant (II.2) dans (II.1). Il vient que

$$\left[\frac{\partial^4\Phi(x)}{\partial x^4}\right]\Big/\Phi(x)+\frac{\rho\cdot L^2}{EI}\left[\frac{\partial^2 q(t)}{\partial t^2}\right]\Big/q(t)=0 \qquad \text{(II.3)}$$

Du fait que les deux fonctions $\Phi(x)$ et $q(t)$ sont indépendantes l'une de l'autre, l'écriture suivante découle

$$\left[\frac{\partial^4\Phi(x)}{\partial x^4}\right]\Big/\Phi(x)=-\frac{\rho\cdot L^2}{EI}\left[\frac{\partial^2 q(t)}{\partial t^2}\right]\Big/q(t) \qquad \text{(II.4)}$$

D'où l'obtention des deux équations suivantes en posant $\omega^2 = \beta^4 \frac{EI}{\rho L}$ (II.7)

$$\frac{\partial^4 \Phi(x)}{\partial x} - \beta^4 \Phi(x) = 0 \quad \text{(II.5)}$$

$$\ddot{q}(t) + \omega^2 q(t) = 0 \quad \text{(II.6)}$$

L'équation (II.6) est la représentation d'une équation des vibrations libres d'un système à un degré de liberté sans amortissement. Ainsi, la coordonnée modale généralisée $q(t)$ est sous la forme :

$$q(t) = A\sin(\omega t) + B\cos(\omega t) \quad \text{(II.8)}$$

A et B sont des constantes dépendantes des conditions initiales.

Les déformées modales $\Phi(x)$ sont des fonctions trigonométriques et trigonométriques hyperboliques. [GOM92, VIN96] (*Figure II.13*)

La solution de (II.5) est de la forme $\Phi(x) = C \cdot e^{sx}$

En la substituant dans (II.5) nous obtenons l'équation suivante $(s^2 - \beta^4) \cdot C \cdot e^{sx} = 0$

Dont les solution sont $s = \pm\beta, \pm i\beta$

D'où $$\Phi(x) = C_1 e^{i\beta x} + C_2 e^{-i\beta x} + C_3 e^{\beta x} + C_4 e^{-\beta x}$$

Enfin, $$\Phi(x) = A_1 \sin \beta x + A_2 \cos \beta x + A_3 \sinh \beta x + A_4 \cosh \beta x \quad \text{(II.9)}$$

Les constantes $A_i, i = \overline{1,4}$ définissent la forme de l'amplitude des vibrations.

La considération des conditions aux limites au niveau des deux extrémités du bras aide à les déterminer.

A l'extrémité encastrée, nous avons des conditions relatives à l'appuie $(x = 0)$:

$$w(x,t)\big|_{x=0} = 0 \quad \text{(II.10)}$$

$$\left.\frac{\partial w(x,t)}{\partial x}\right|_{x=0} = 0 \quad \text{(II.11)}$$

En tenant compte de (II.2) et (II.5), il vient que

$$\Phi(x)\big|_{x=0} = 0 \quad \text{(II.12)}$$

$$\Phi'(x)\big|_{x=0} = 0 \quad \text{(II.13)}$$

D'autres conditions relatives à la charge m_c représentant le moment fléchissant (flexion) et l'effort tranchant (cisaillement) sont prises en compte :

$$EI \left.\frac{\partial^2 w(x,t)}{\partial x^2}\right|_{x=L} = -J_c \frac{d^2}{dt^2}\left[\left.\frac{\partial w(x,t)}{\partial x}\right|_{x=L}\right] \quad \text{(II.14)}$$

$$EI\frac{\partial^3 w(x,t)}{\partial x^3}\bigg|_{x=L} = M_c \frac{d^2}{dt^2}\left[w(x,t)\big|_{x=L}\right] \tag{II.15}$$

En tenant compte de (II.2) et (II.6), il vient que

$$EI\Phi''(x)\big|_{x=L} - \omega^2 J_c \Phi'(x)\big|_{x=L} = 0 \tag{II.16}$$

$$EI\Phi'''(x)\big|_{x=L} + \omega^2 M_c \Phi(x)\big|_{x=L} = 0 \tag{II.17}$$

En reportant (II.9) dans (II.12), (II.13)

$$\begin{cases} \Phi(x)\big|_{x=0} = 0 \Rightarrow A_2 = -A_4 \\ \Phi'(x)\big|_{x=0} = 0 \Rightarrow A_1 = -A_3 \end{cases} \tag{II.18}$$

Et de (II.7), (II.16) et (II.17)

$$\begin{cases} \Phi''(x)\big|_{x=L} - \dfrac{\beta^4 J_c}{\rho L^2}\Phi'(x)\big|_{x=L} = 0 \\ \Phi'''(x)\big|_{x=L} + \dfrac{\beta^4 m_c}{\rho L^2}\Phi(x)\big|_{x=L} = 0 \end{cases} \tag{II.19}$$

En substituant (II.9) dans (II.19) et en prenant compte de (II.18), une matrice 2*2 découle telle que

$$\begin{bmatrix} Q_{11}(\beta) & Q_{12}(\beta) \\ Q_{21}(\beta) & Q_{22}(\beta) \end{bmatrix}\begin{bmatrix} A_1 \\ A_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{II.20}$$

$Q(\beta)$ doit être singulière afin d'exclure les solutions nulles. De ce fait, son déterminant devrait âtre nul. On a alors

$$[1+\cos\beta L\cdot\cosh\beta L] - \frac{\beta M_c}{\rho L^2}[\sin\beta L\cosh\beta L - \cos\beta L\sinh\beta L]$$

$$-\frac{\beta^3 J_c}{\rho L^3}[\cos\beta L\sinh\beta L + \sin\beta L\cosh\beta L] + \frac{\beta^4 J_c m_c}{\rho^2 L^4}[1-\cos\beta L\cosh\beta L] = 0 \tag{II.21}$$

La résolution de cette équation transcendante donne une infinité de solution de β qui représente les fréquences de vibration du bras flexible.
Une des valeur de β est déterminée par la condition aux limite suivante $\Phi(x)\big|_{x=L} = 1$

De (II.20), $A_2 = -\dfrac{Q_{11}(\beta)}{Q_{12}(\beta)}A_1$

D'où

$$A_2 = \frac{\beta^3 J_c(\cos\beta L - \cosh\beta L) - \rho L^2(\sin\beta L - \sinh\beta L)}{\beta^3 J_c(\sin\beta L - \sinh\beta L) + \rho L^2(\cos\beta L - \cosh\beta L)}A_1 \tag{II.22}$$

Enfin,(II.23)

$$\Phi(x) = A_1\left[\sin\beta x - \sinh\beta x + \frac{\beta^3 J_c(\cos\beta L - \cosh\beta L) - \rho L^2(\sin\beta L - \sinh\beta L)}{\beta^3 J_c(\sin\beta L - \sinh\beta L) + \rho L^2(\cos\beta L - \cosh\beta L)}(\cos\beta x - \cosh\beta x)\right]$$

II.3. Etude énergétique

Le formalisme Lagrangien nous permet, à partir du calcul des énergies cinétique et potentielle, de modéliser la dynamique de notre système. Il repose sur les équations d'Euler – Lagrange. Dans le cas de notre système complexe, cette méthode est la plus appropriée, pour la simplicité des calculs, par rapport à d'autres telles que la méthode d'Hamilton étendue et la méthode de Newton Euler

II.3.1. Choix du référentiel

Simplifier les calculs revient à choisir un bon référentiel en ayant les bonnes coordonnées.

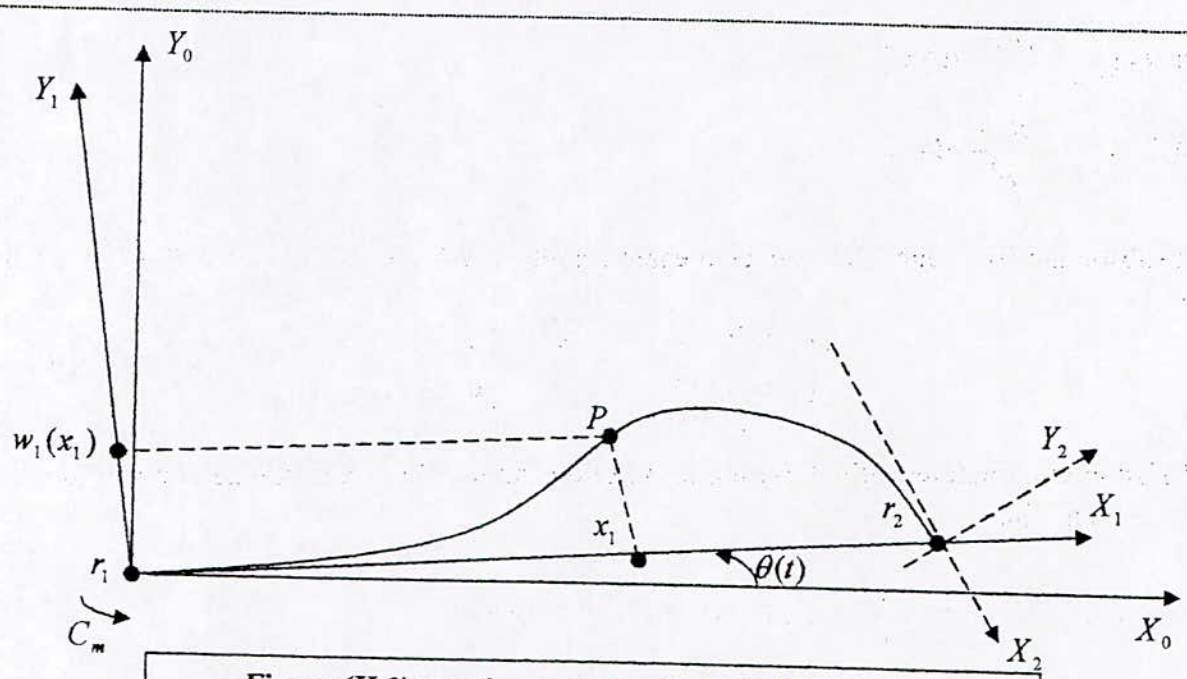


***Figure (II.2)* : repérage cinématique du bras flexible**

Le repérage cinématique est illustré par la figure (II.2) où :

(X_0, Y_0) : représente le repère absolu
(X_1, Y_1) : est un repère associé au mouvement rigide de la liaison
(X_2, Y_2) : est un repère associé au mouvement flexible de la liaison
$w_1(x_1)$: est la déflexion d'un point de la liaison exprimée dans le repère (X_1, Y_1)
$\theta(t)$: est la rotation rigide

Notation

Notation

${}^{i}P_1(x_1)=[x_1, w(x_1)]^T$	: Composantes d'un point P de la liaison dans le repère i
${}^{0}P_1(x_1)$	: Composantes d'un point P de la liaison dans le repère absolu
${}^{1}r_2={}^{1}P_1(L)$	: Position de l'origine du repère 2 par rapport au repère 1
${}^{0}r_2$	: Position de l'origine du repère 2 par rapport au repère absolu

Matrices de transformation

La position absolue d'un point P peut s'exprimée le long du bras par

$$ {}^{0}P_1 = A\cdot{}^{1}P_1 $$

et sa vitesse par

$$ {}^{0}\dot{P}_1 = \dot{A}\,{}^{1}P_1 + A\,{}^{1}\dot{P}_1 $$

où A est la matrice de passage exprimant la rotation rigide de (X_1, Y_1) par rapport au repère absolue.

A est donnée par $\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$

et $\dot{A} = SA\dot{\theta}$ avec $S = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$

II.3.2. Formalisme d'Euler – Lagrange

L'équation le Lagrange est définie par

$$ \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{V}}\right) - \frac{\partial L}{\partial V} + \frac{\partial D}{\partial \dot{V}} = Q \qquad \text{(II.24)} $$

V : Vecteur des coordonnées généralisées
L : Lagrangien donnée par $L = T_T - U$
T_T : Energie cinétique totale du système
U : Energie potentielle
D : Energie dissipative totale du système
Q : Forces généralisées

A partir de cette équation, nous aboutissons au modèle dynamique non linéaire du système.

II.3.2.1. Calcul du Lagrangien

II.3.2.1.1. Calcul énergétique

A. Energie cinétique

Elle est calculée par la somme suivante

$$T = T_A + T_L + T_C \tag{II.25}$$

A.a. Energie cinétique de l'articulation T_A

Elle est donnée par

$$T_A = \tfrac{1}{2} m_a {}^0\dot{r}_1^T {}^0\dot{r}_1 + \tfrac{1}{2} J_a \dot{\theta}^2 \tag{II.26}$$

Le produit ${}^0\dot{r}_1^T {}^0\dot{r}_1$ est nul du fait qu'il s'agit d'un encastrement, ce qui réduit cette énergie à $\tfrac{1}{2} J_a \dot{\theta}^2$

A.b. Energie cinétique de la liaison T_L

Elle est donnée par

$$T_L = \tfrac{1}{2} \int_0^L \rho \, {}^0\dot{P}_1^T {}^0P_1 dx \tag{II.27}$$

$$\begin{aligned} {}^0\dot{P}_1^T {}^0P_1 &= ({}^1P_1^T \dot{A}^T + {}^1\dot{P}_1^T A^T)(\dot{A}\,{}^1P_1 + A\,{}^1\dot{P}_1) \\ &= (x_1^2 + w_1^2)\dot{\theta}^2 + \dot{w}_1^2 + 2x_1 \dot{w}_1 \dot{\theta} \end{aligned}$$

Notons que

$$\begin{aligned} A^T A &= I \\ \dot{A}^T \dot{A} &= \dot{\theta}^2 \\ A^T \dot{A} &= S\dot{\theta} \\ \dot{A}^T A &= S^T \dot{\theta} \end{aligned}$$

d'où le résultat $$T_L = \tfrac{1}{2} \int_0^L \rho (x_1^2 + w_1^2)\dot{\theta}^2 dx + \tfrac{1}{2} \int_0^L \rho \cdot \dot{w}_1^2 dx + \int_0^L \rho \cdot x_1 \dot{w}_1 \dot{\theta} \cdot dx \tag{II.28}$$

A.c. Energie cinétique de la charge T_c

Elle est donnée par

$$T_C = \tfrac{1}{2} m_c {}^0\dot{r}_2^T {}^0\dot{r}_2 + \tfrac{1}{2} J_c (\dot{\theta} + \dot{w}_e')^2 \tag{II.29}$$

où

$w_e = w(x,t)\big|_{x=L}$ (Déflexion de l'extrémité du bras)

$w_e' = \left.\dfrac{\partial w(x,t)}{\partial x}\right|_{x=L}$ (Angle introduit par la charge à l'extrémité du bras)

$$\dot{w}'_e = \frac{d}{dt}\left(\left. \frac{\partial w(x,t)}{\partial x} \right|_{x=L} \right)$$

$$\begin{aligned} {}^0\dot{r}_2^T\,{}^0\dot{r}_2 &= ({}^1r_2^T \dot{A}^T + {}^1\dot{r}_2^T A^T)(\dot{A}\,{}^1r_2 + A\,{}^1\dot{r}_2) \\ &= \dot{\theta}^2(L^2 + w_e^2) + 2\dot{\theta}\cdot L\dot{w}_e + \dot{w}_e^2 \end{aligned}$$

D'où $$T_C = \tfrac{1}{2}m_c\dot{\theta}^2(L^2 + w_e^2) + m_c\dot{\theta}\cdot L\dot{w}_e + \tfrac{1}{2}m_c\dot{w}_e^2 + \tfrac{1}{2}J_c(\dot{\theta} + \dot{w}'_e)^2 \qquad \text{(II.30)}$$

B. Energie potentielle

Le mouvement du bras étant planaire, seule l'énergie élastique de flexion horizontale emmagasinée dans la barre est à considérer. De ce fait l'énergie potentielle se traduit par l'expression suivante :

$$U = \tfrac{1}{2}\int_0^L EI\left(\frac{\partial^2 w(x,t)}{\partial x^2}\right)^2 dx \qquad \text{(II.31)}$$

II.3.2.1.2. Introduction de la fonction modale

Comme c'était précisé dans le chapitre précédent, le déplacement w s'écrit sous la forme d'une somme finie telle que

$$w(x,t) = \sum_{i=1}^{n} \Phi_i(x)\cdot q_i(t) \qquad \text{(II.32)}$$

Où n est le nombre des modes vibratoires considérés.
En substituant cette expression dans celles obtenues pour les différentes énergies, nous obtenons les écritures suivantes :

$$\begin{aligned} T_C = \tfrac{1}{2}\Bigg[m_c L^2 + J_c + + m_c\left(\sum_i \Phi_{ie} q_i\right)\left(\sum_j \Phi_{je} q_j\right)\Bigg]\dot{\theta}^2 + \Bigg[m_c L\sum_i \Phi_{ie}\dot{q}_i + J_c\sum_i \Phi'_{ie}\dot{q}_i \Bigg]\dot{\theta} + \\ \tfrac{1}{2}\left[m_c\left(\sum_i \Phi_{ie}\dot{q}_i\right)\left(\sum_j \Phi_{je}\dot{q}_j\right) + J_c\left(\sum_i \Phi'_{ie}\dot{q}_i\right)\left(\sum_j \Phi'_{je}\dot{q}_j\right)\right] \end{aligned} \qquad \text{(II.33)}$$

$$\begin{aligned} T_L = \tfrac{1}{2}\int_0^L \rho x_1^2\dot{\theta}^2 dx + \tfrac{1}{2}\int_0^L \rho\left(\sum_i \Phi_i q_i\right)\left(\sum_j \Phi_j q_j\right)\dot{\theta}^2 dx + \tfrac{1}{2}\int_0^L \rho x_1\dot{\theta}\left(\sum_i \Phi_i\dot{q}_i\right)dx + \\ \tfrac{1}{2}\int_0^L \rho\left(\sum_i \Phi_i\dot{q}_i\right)\left(\sum_j \Phi_j\dot{q}_j\right)dx \end{aligned} \qquad \text{(II.34)}$$

$$U = \tfrac{1}{2}\int_0^L EI\left(\sum_i \Phi''_i q_i\right)\left(\sum_j \Phi''_j q_j\right)dx \qquad \text{(II.35)}$$

Les fonctions modales sont deux à deux orthogonales. De ce fait, un ensemble de simplification s'introduit dans l'expression de l'énergie cinétique de la liaison. Il vient que

$$T_L = \frac{1}{2}\left[\int_0^L \rho \cdot x_1^2 dx + \sum_i q_i^2 \int_O^L \rho \cdot \Phi_i^2 dx\right]\dot{\theta}^2 + \frac{1}{2}\left[\sum_i \dot{q}_i \int_0^L \rho \cdot x_1 \Phi_i dx\right]\dot{\theta} + \frac{1}{2}\sum_i \dot{q}_i^2 \int_0^L \rho \cdot \Phi_i^2 dx \qquad \text{(II.36)}$$

et l'énergie potentielle s'écrit

$$U = \frac{1}{2}\sum_i q_i^2 \int_0^L EI\Phi_i''^2 dx \qquad \text{(II.37)}$$

Notons que :

$$\Phi_{ie} = \Phi_i(x)\big|_{x=L}$$

$$\Phi'_{ie} = \frac{\partial \Phi_i(x)}{\partial x}\bigg|_{x=L}$$

II.3.2.1.3. Introduction des modes supposés

Les études ont montré que 2 à 3 modes flexibles sont suffisants pour une bonne représentation dynamique de la poutre [GOM92] et que l'hypothèse de non cisaillement est en défaut si le nombre de mode augmente.

Nous adoptons les notations suivantes :

$J_b = \rho \int_0^L x^2 dx$: Inertie du bras par rapport à l'axe de l'articulation

$w_i = \rho \int_0^L x\Phi_i(x) dx$: Moment de déformation d'ordre un de la liaison pour le mode i

$\sigma_i = \rho \int_0^L \Phi_i^2(x) dx$: Moment transversal pour le mode i

$k_i = EI \int_0^L \Phi_i''^2(x) dx$: Coefficient d'élasticité transversale du mode i

Pour les différentes énergies, nous obtenons les écritures suivantes :

$$T_A = \frac{1}{2} J_a \dot{\theta}^2 \qquad \text{(II.38)}$$

$$T_L = \frac{1}{2}\left[J_b + q_1^2\sigma_1 + q_2^2\sigma_2\right]\dot{\theta}^2 + \left[\dot{q}_1 w_1 + \dot{q}_2 w_2\right]\dot{\theta} + \frac{1}{2}\left[\dot{q}_1^2\sigma_1 + \dot{q}_2^2\sigma_2\right] \qquad \text{(II.39)}$$

$$T_C = \frac{1}{2}\left[m_c L^2 + J_c + m_c(\Phi_{1e}q_1 + \Phi_{2e}q_2)^2\right]\dot{\theta}^2 + \left[m_c L(\Phi_{ie}\dot{q}_1 + \Phi_{2e}\dot{q}_2) + J_c(\Phi'_{1e}\dot{q}_1 + \Phi'_{2e}\dot{q}_2)\right]\dot{\theta} + \frac{1}{2}\left[m_c(\Phi_{1e}\dot{q}_1 + \Phi_{2e}\dot{q}_2)^2 + J_c(\Phi'_{1e}\dot{q}_1 + \Phi'_{2e}\dot{q}_2)^2\right] \qquad \text{(II.40)}$$

$$U = \frac{1}{2}\left[q_1^2 k_1 + q_2^2 k_2\right] \qquad \text{(II.41)}$$

II.3.2.1.4. Calcul du Lagrangien

Soit le vecteur V des coordonnées généralisées suivant

$$V = \begin{bmatrix} \theta \\ q \end{bmatrix} = \begin{bmatrix} \theta \\ q_1 \\ q_2 \end{bmatrix} \tag{II.42}$$

où $\theta = \theta(t)$ est la variable articulaire

et $q = q(t)$ est le vecteur des variables temporelles modales

le premier terme de l'équation est donné par :

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{V}}\right) = \left[\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) \quad \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_1}\right) \quad \frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_2}\right)\right]^T \tag{II.43}$$

Les différents éléments du vecteur sont donnés par :

$$\begin{aligned}\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\theta}}\right) &= \left[J_a + J_b + J_c + m_c L^2 + \sigma_1 q_1^2 + \sigma_2 q_2^2 + m_c(\Phi_{1e} q_1 + \Phi_{2e} q_2)^2\right]\ddot{\theta} + \\ &\quad 2\left[\sigma_1 q_1 \dot{q}_1 + \sigma_2 q_2 \dot{q}_2 + m_c(\Phi_{1e} q_1 + \Phi_{2e} q_2)(\Phi_{1e}\dot{q}_1 + \Phi_{2e}\dot{q}_2)\right]\dot{\theta} + \\ &\quad \left[m_c L \Phi_{1e} + J_c \Phi'_{1e} + \tfrac{1}{2} w_1\right]\ddot{q}_1 + \left[m_c L \Phi_{2e} + J_c \Phi'_{2e} + \tfrac{1}{2} w_2\right]\ddot{q}_2\end{aligned} \tag{II.44}$$

$$\begin{aligned}\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_1}\right) &= \left[w_1 + m_c L \Phi_{1e} + J_c \Phi'_{1e}\right]\ddot{\theta} + \left[\sigma_1 + m_c \Phi_{1e}^2 + J_c \Phi_{1e}'^2\right]\ddot{q}_1 + \\ &\quad \left[m_c \Phi_{1e} \Phi_{2e} + J_c \Phi'_{1e} \Phi'_{2e}\right]\ddot{q}_2\end{aligned} \tag{II.45}$$

$$\begin{aligned}\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_2}\right) &= \left[w_2 + m_c L \Phi_{2e} + J_c \Phi'_{2e}\right]\ddot{\theta} + \left[m_c \Phi_{1e} \Phi_{2e} + J_c \Phi'_{1e} \Phi'_{2e}\right] + \\ &\quad \left[\sigma_2 + m_c \Phi_{2e}^2 + J_c \Phi_{2e}'^2\right]\ddot{q}_2\end{aligned} \tag{II.46}$$

Le second terme de l'équation est donnée par :

$$\frac{\partial L}{\partial V} = \left[\frac{\partial L}{\partial \theta} \quad \frac{\partial L}{\partial q_1} \quad \frac{\partial L}{\partial q_2}\right]^T \tag{II.47}$$

Et ses différents termes par :

$$\frac{\partial L}{\partial \theta} = 0 \tag{II.48}$$

$$\frac{\partial L}{\partial q_1} = \left[\left(m_c \Phi_{1e}^2 + \sigma_1\right) q_1 + m_c \Phi_{1e} \Phi_{2e} q_2\right]\dot{\theta}^2 - k_1 q_1 \tag{II.49}$$

$$\frac{\partial L}{\partial q_2} = \left[\left(m_c \Phi_{2e}^2 + \sigma_2\right) q_2 + m_c \Phi_{1e} \Phi_{2e} q_1\right] \dot{\theta}^2 - k_2 q_2 \qquad \text{(II.50)}$$

La fonction de dissipation D est donnée par :

$$D = \tfrac{1}{2} \sum_i \sum_j d_{ij} \dot{V}_i^T(t) \dot{V}_j(t) \qquad \text{(II.51)}$$

En considérant la propriété d'orthogonalité des fonctions modales, une simplification s'apporte. D'où

$$D = \tfrac{1}{2} \sum_i d_i \dot{V}_i^2(t) \qquad \text{(II.52)}$$

D est alors donné par

$$D = \tfrac{1}{2} \left[0 \cdot \dot{\theta}^2 + d_1 \dot{q}_1^2 + d_2 \dot{q}_2^2\right] \qquad \text{(II.53)}$$

Enfin, le troisième terme de l'équation du lagrangien est donné par

$$\frac{\partial D}{\partial \dot{V}} = \begin{bmatrix} 0 & d_1 \dot{q}_1 & d_2 \dot{q}_2 \end{bmatrix}^T \qquad \text{(II.54)}$$

Le rôle de la matrice D est néanmoins important, car elle permet de restituer une stabilité absolue (partie réelle des pôles) de plus en plus importante des modes lorsqu'on monte en fréquence en raison de la viscosité du matériau. C'est pour cette raison qu'elle est généralement choisie proportionnelle à la matrice de raideur.

Dans plusieurs ouvrages, les d_i sont approximés par $0.1\sqrt{k_i}$; $i = \overline{1,2}$

L'expression de la force généralisée Q est déterminée à partir du travail δW dû aux forces appliquées, pour un déplacement δV. Nous avons alors :

$$Q = \frac{\delta W}{\delta V} \qquad \text{(II.55)}$$

δW est donné par

$$\delta W = F(x,t)\delta w(x,t) + M(x,t)\delta\left[\frac{\partial w(x,t)}{\partial x}\right] \qquad \text{(II.56)}$$

où $F(x,t), M(x,t)$ sont respectivement la force et le moment concentré au point $P(x)$

Le bras n'étant soumis qu'au couple moteur au niveau encastré, il vient que

$F(x,t) = 0$ pour $0 \le x \le L$

$M(x,t) = 0$ pour $0 < x \le L$

$M(0,t) = C_m$

L'écriture de δW devient

$$\delta W = C_m \sum_i \left.\frac{\partial \Phi_i(x)}{\partial x}\right|_{x=0} \delta q_i \qquad \text{(II.57)}$$

Les forces généralisées par rapport aux variables temporelles modales sont alors données par

$$Q_i = \frac{\delta W}{\delta q_i} = C_m \sum_i \left. \frac{\partial \Phi_i(x)}{\partial x} \right|_{x=0} \qquad \text{(II.58)}$$

Enfin, la force généralisée Q est donnée par

$$Q = \begin{bmatrix} \frac{\delta W}{\delta \theta} & \frac{\delta W}{\delta q_1} & \frac{\delta W}{\delta q_2} \end{bmatrix}^T = \begin{bmatrix} C_m & 0 & 0 \end{bmatrix}^T, \qquad \text{(II.59)}$$

II.3.2.2. Mise sous forme matricielle

Les différents calculs nous mènent aux équations dynamiques du modèle non linéaire donné sous la forme compact suivante :

$$A(q)\begin{bmatrix} \ddot{\theta} \\ \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} N_1(\dot{\theta}, q, \dot{q}) \\ N_2(\dot{\theta}, q) \\ N_3(\dot{\theta}, q) \end{bmatrix} + \begin{bmatrix} 0 \\ Kq \end{bmatrix} + \begin{bmatrix} 0 \\ D\dot{q} \end{bmatrix} = \begin{bmatrix} C_m \\ 0 \\ 0 \end{bmatrix} \qquad \text{(II.60)}$$

ou

$$A(q)\begin{bmatrix} \ddot{\theta} \\ \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} N_1(\dot{\theta}, q, \dot{q}) \\ N_2(\dot{\theta}, q) \\ N_3(\dot{\theta}, q) \end{bmatrix} + \begin{bmatrix} 0 \\ k_1 q_1 + d_1 \dot{q}_1 \\ k_2 q_2 + d_2 \dot{q}_2 \end{bmatrix} = \begin{bmatrix} C_m \\ 0 \\ 0 \end{bmatrix} \qquad \text{(II.61)}$$

Où $A(q)$ est la matrice d'inertie symétrique définie positive. Ses éléments sont donnés par :

$$A_{11}(q) = J_A + J_b + J_c + m_c L^2 + \sigma_1 q_1^2 + \sigma_2 q_2^2 + m_c (\Phi_{1e} q_1 + \Phi_{2e} q_2)^2$$
$$A_{12}(q) = m_c L \Phi_{1e} + J_c \Phi'_{1e} + w_1$$
$$A_{13}(q) = m_c L \Phi_{2e} + J_c \Phi'_{2e} + w_2$$
$$A_{ii}(q) = \sigma_i + m_c \Phi^2_{(i-1)e} + J_c \Phi'^2_{(i-1)e} \quad , i = \overline{2,3}$$
$$A_{23}(q) = m_c \Phi_{1e} \Phi_{2e} + J_c \Phi'_{1e} \Phi'_{2e}$$

Les termes non linéaires $N_i, i = \overline{1,3}$, représentent les forces de Coriolis et centrifuge. Ils sont comme suit :

$$N_1(\dot{\theta}, q, \dot{q}) = 2\left[\sigma_1 q_1 \dot{q}_1 + \sigma_2 q_2 \dot{q}_2 + m_c (\Phi_{1e} q_1 + \Phi_{2e} q_2)(\Phi_{1e} \dot{q}_1 + \Phi_{2e} \dot{q}_2)\right]\dot{\theta}$$
$$N_2(\dot{\theta}, q) = -\left[(m_c \Phi^2_{1e} + \sigma_1) q_1 + m_c \Phi_{1e} \Phi_{2e} q_2\right]\dot{\theta}^2$$
$$N_3(\dot{\theta}, q) = -\left[(m_c \Phi^2_{2e} + \sigma_2) q_2 + m_c \Phi_{1e} \Phi_{2e} q_1\right]\dot{\theta}^2$$

K est la matrice de raideur constante telle que $K = diag\{k_1, k_2\}$
D est la matrice d'amortissement telle que $D = diag\{d_1, d_2\}$

II.4. Etude par simulation du comportement du bras en boucle ouverte

Afin de mettre en évidence l'état vibratoire du bras, une simulation en boucle ouverte a été effectuée en position et en vitesse, de l'extrémité du bras et des deux premiers modes flexibles, pour un état libre (charge nulle) et chargé (charge de 1kg)

II.4.1. Paramètres physiques :

Les paramètres physiques du bras sont présentés en **Annexe A**.

II.4.2. Réponses à une impulsion

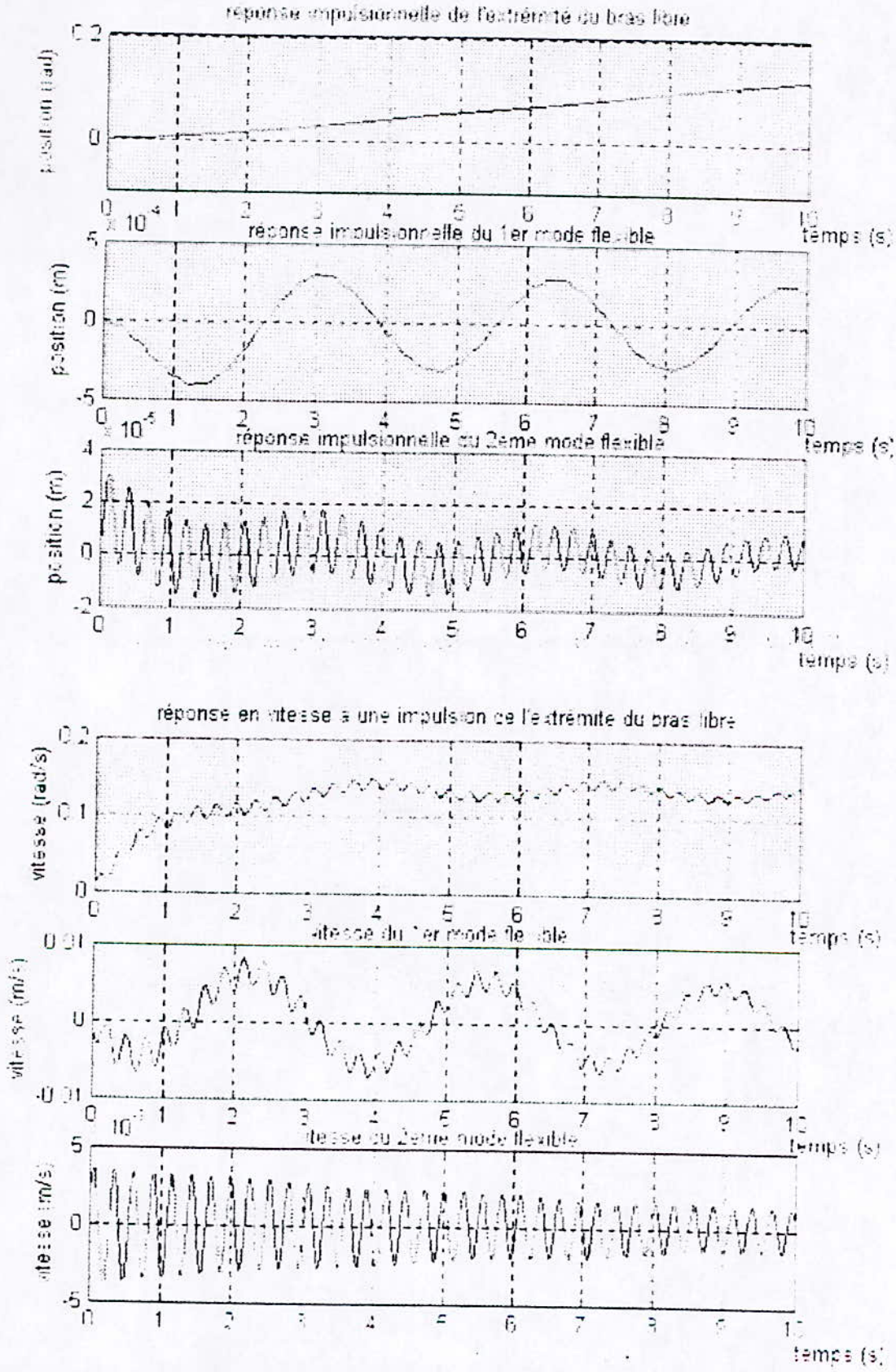


Fig (II.5) : Réponses impulsionnelles en position et en vitesse du bras libre

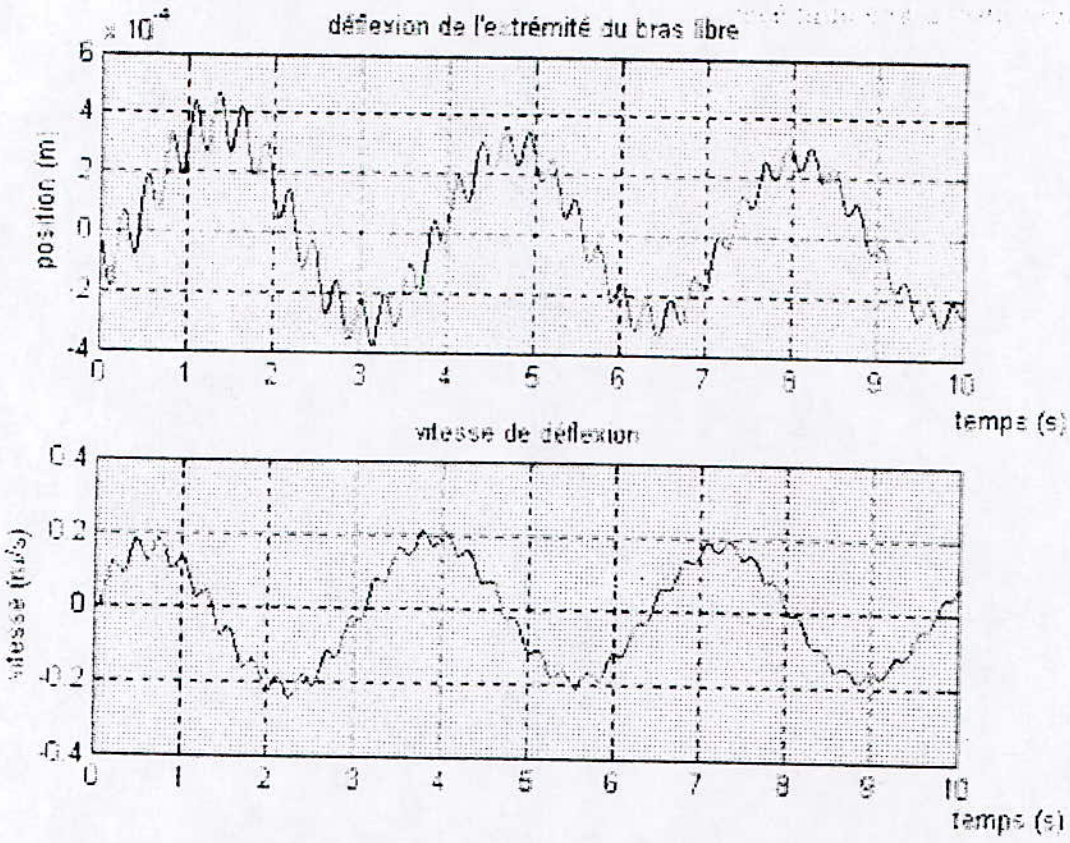


Fig (II,6) : Déflexion et vitesse de déflexion de l'extrémité du bras libre

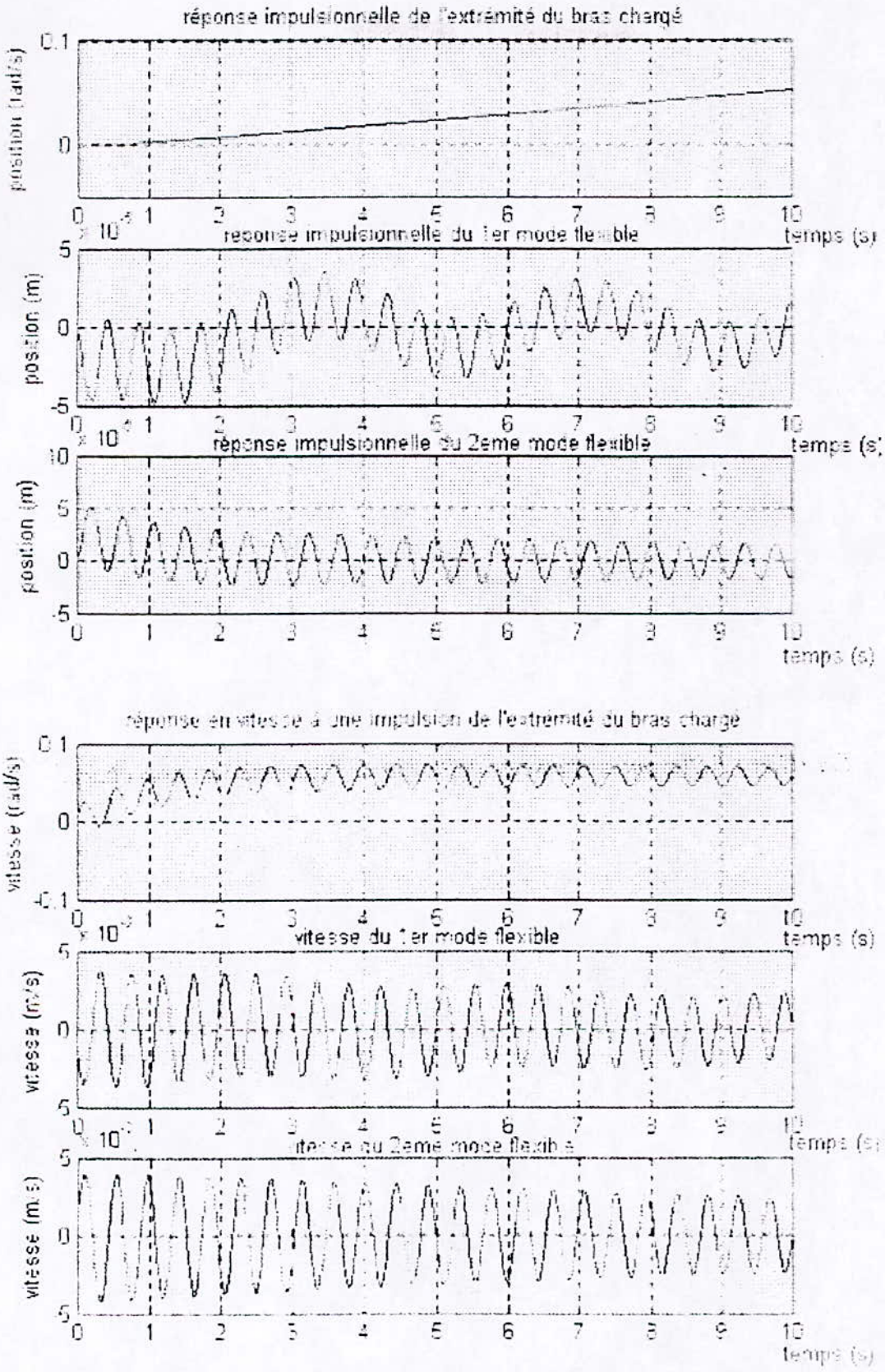


Fig (II.7) : Réponses impulsionnelles en position et en vitesse du bras chargé

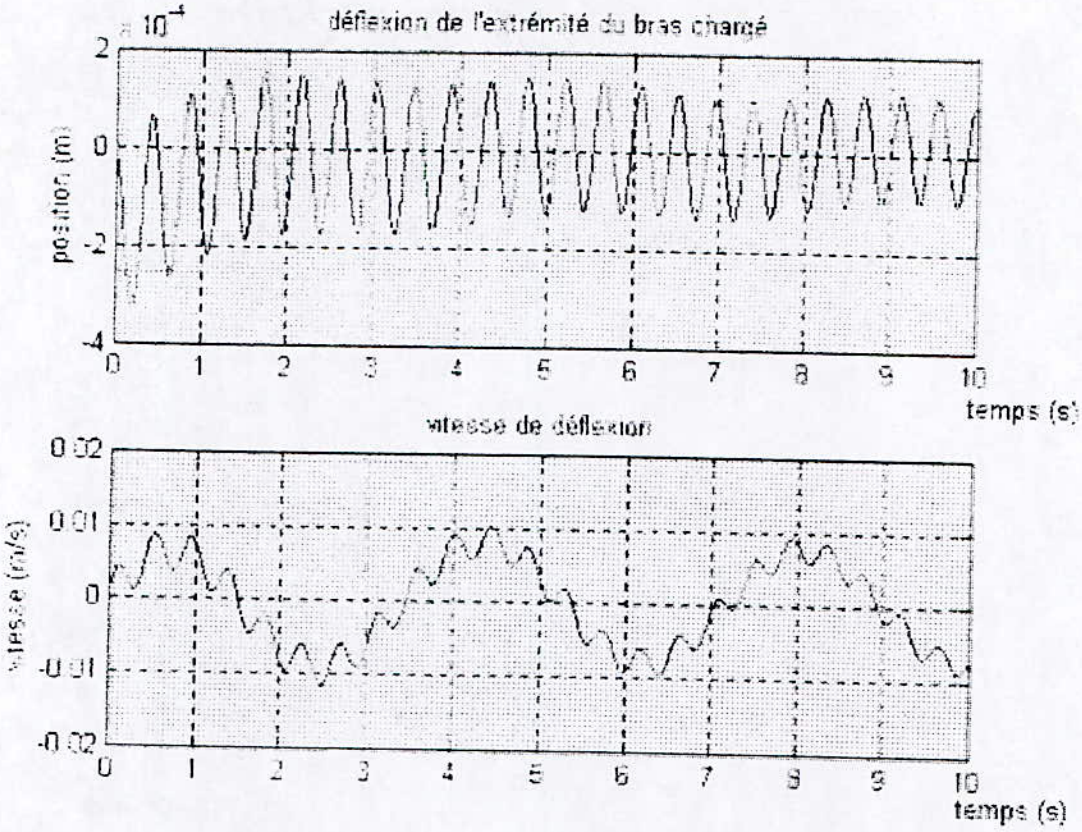


Fig (II,8) : Déflexion et vitesse de déflexion de l'extrémité du bras chargé

II.4.3. Réponses indicielles

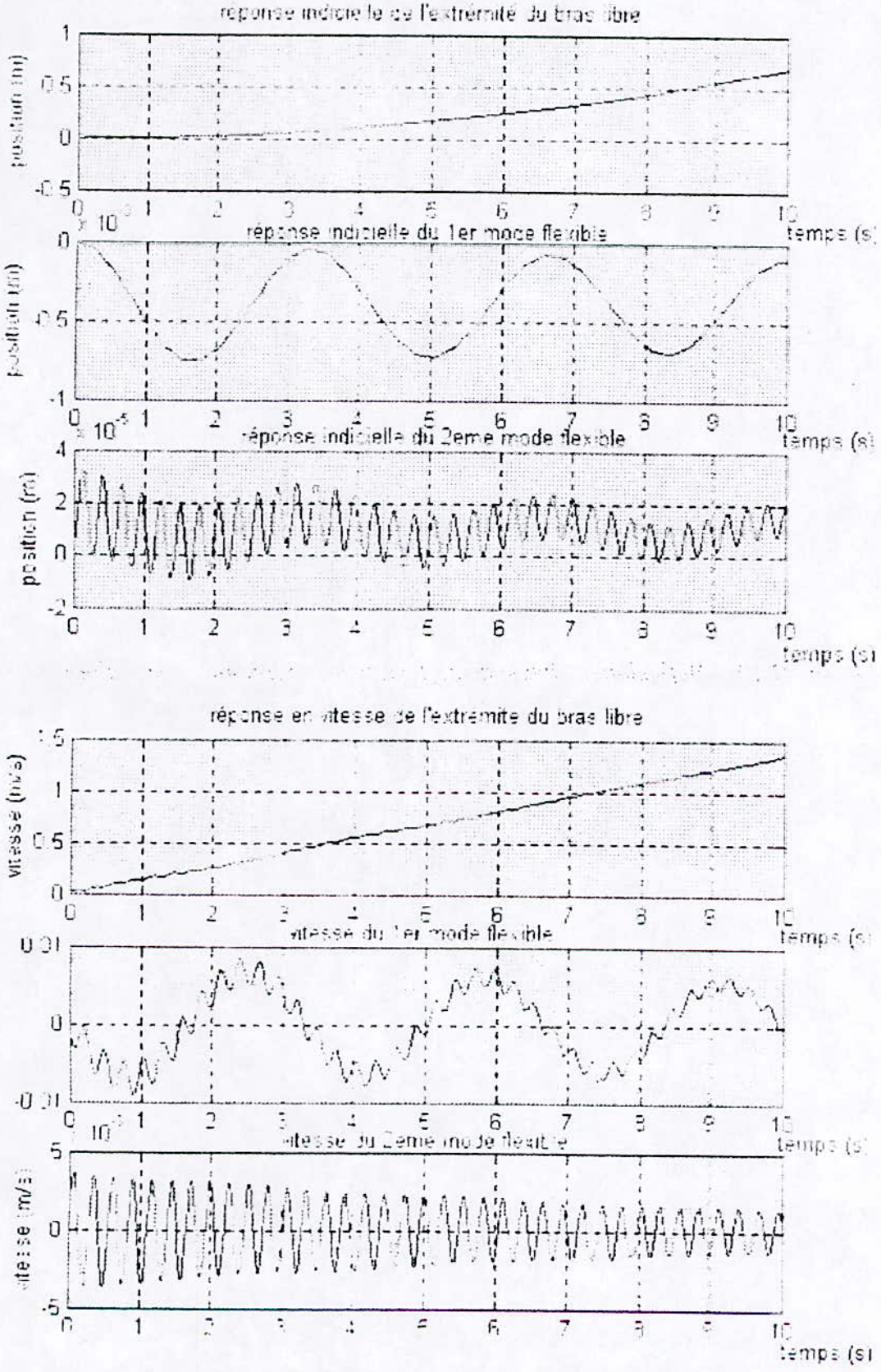


Fig (II.9) : Réponses indicielles en position et en vitesse du bras libre

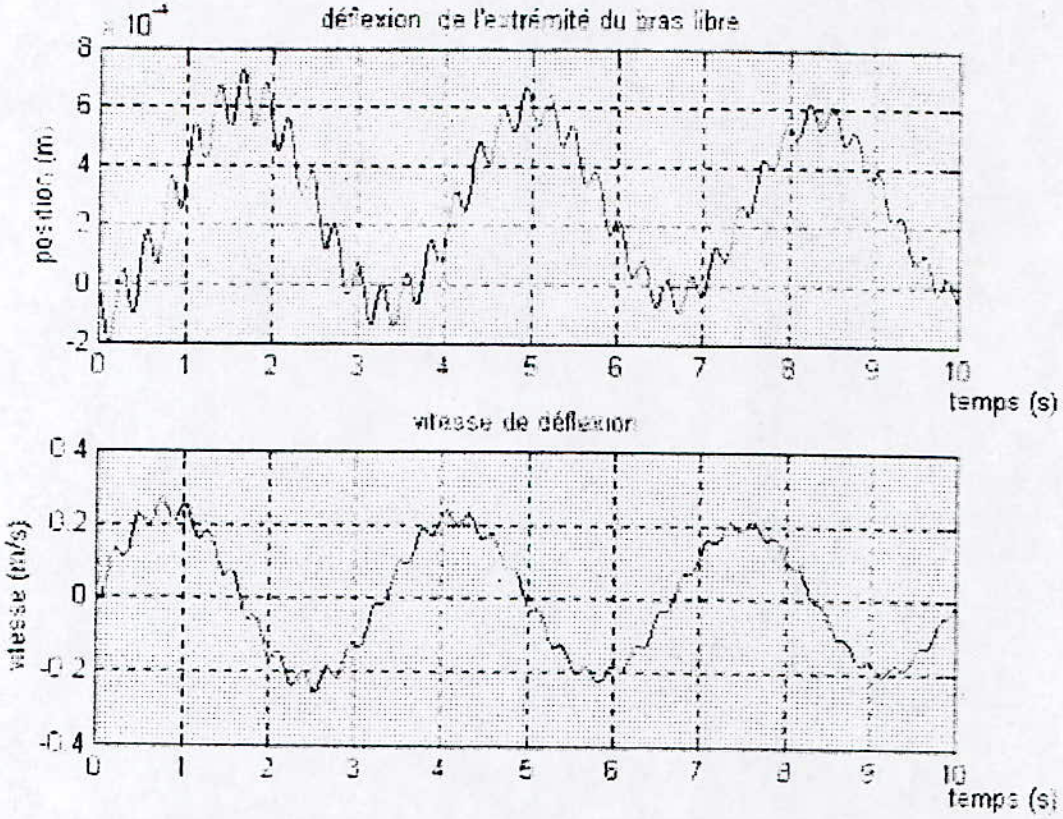


Fig (II.10) : Déflexion et vitesse de déflexion de l'extrémité du bras libre

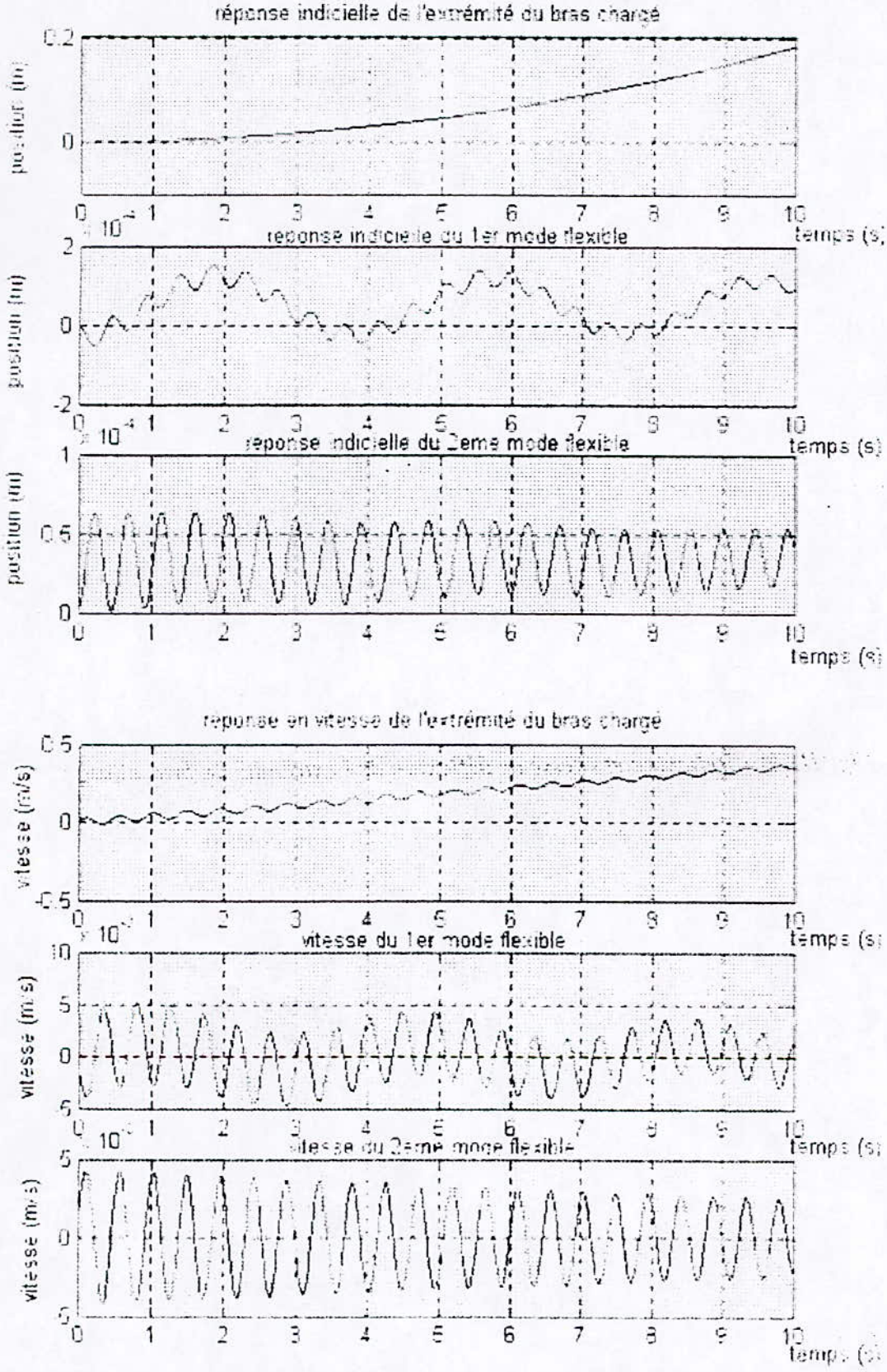


Fig (II.11) : Réponses indicielles en position et en vitesse du bras chargé

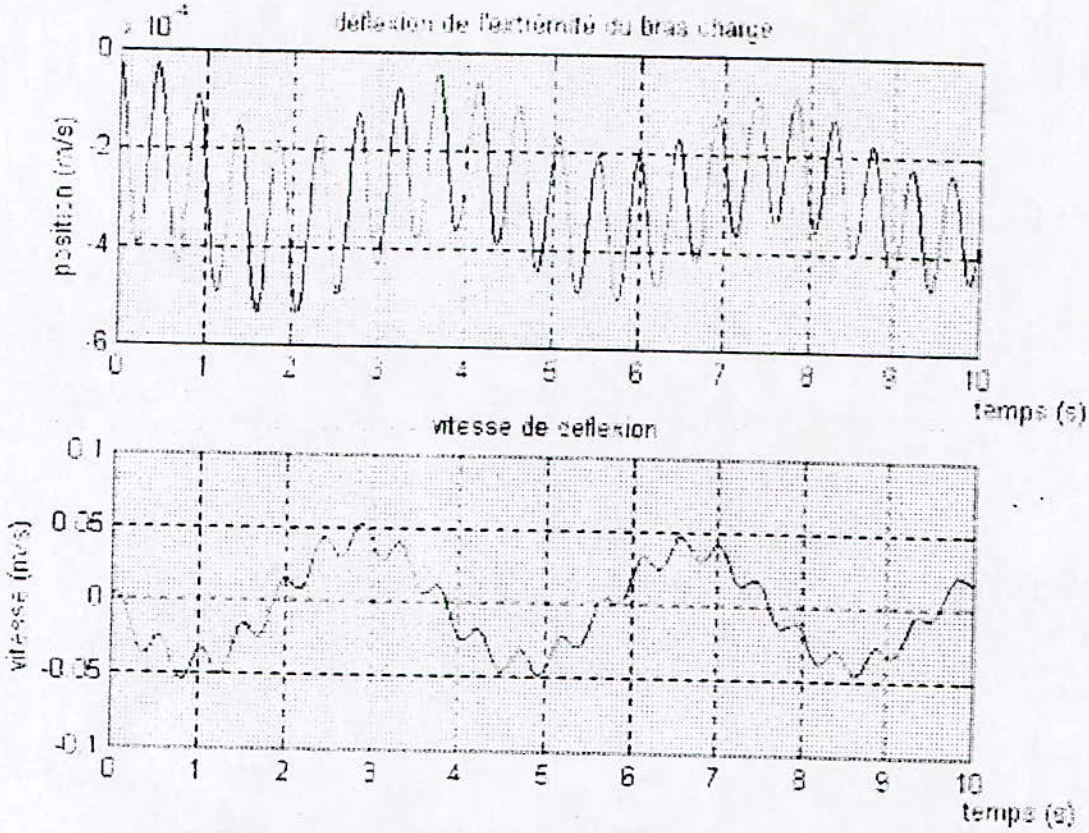


Fig (II,12) : Déflexion et vitesse de déflexion de l'extrémité du bras chargé

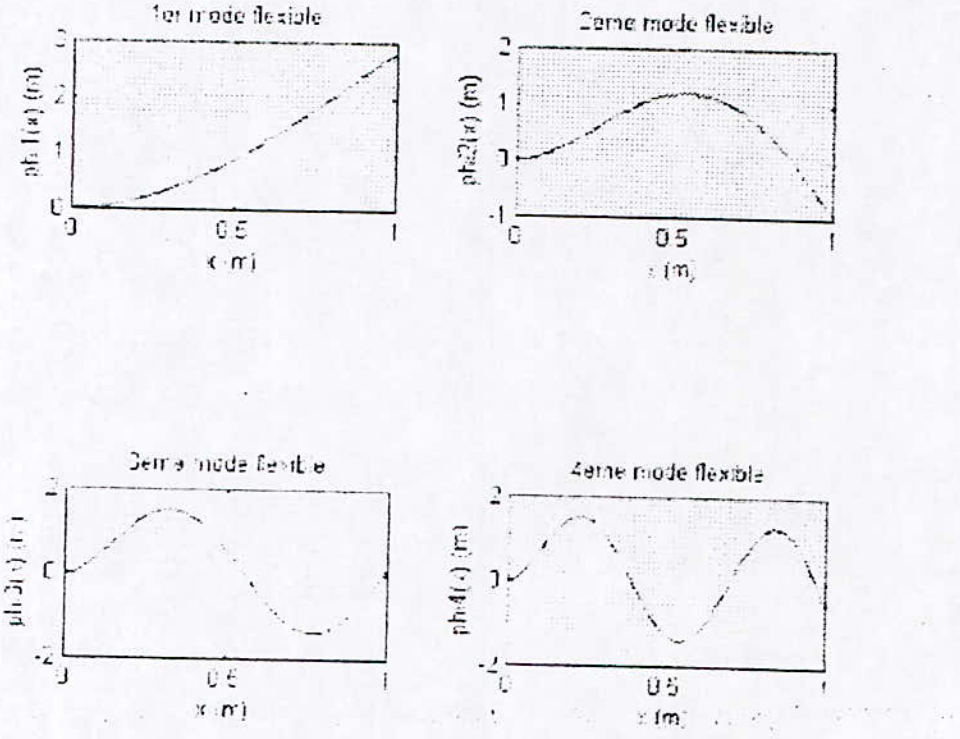


Fig (II.13) : Formes des quatre premières déformées modales

Conclusion

L'approche modale consiste à rapprocher le déplacement par deux fonctions, l'une caractérisant l'espace dite déformée modale et l'autre caractérisant le temps dite coordonnée modale ayant un nombre fini et suffisant de modes.

Le modèle dynamique explicite et complet du robot à une liaison flexible est obtenu en associant l'approche modale au formalisme Lagrangien, la liaison étant considérée comme une poutre bidimensionnelle d'Euler-Bernoulli.

Les simulations en position et en vitesse effectuées en boucle ouverte sur le bras, nous ont permises de mettre en évidence l'état vibratoire de ce dernier, et de voir l'influence de la charge transportée et de la structure du bras souple, sur son extrémité de préhension.

3

Méthodologie du contrôle flou

Introduction

La commande floue est l'application floue de la logique floue. Ces algorithmes sont à base de règles linguistiques du type « Si...Alors ». Elle sert à prendre une décision même si l'on ne peut estimer les entrées/sorties qu'à partir e prédicats vagues ou lorsque les entrées/sorties sont entachées d'erreurs.

Ce chapitre nous permet de comprendre la méthodologie de contrôle d'une telle commande. Pour cela des connaissances en logique floues sans indispensables et feront l'objet d'étude de la première partie de ce chapitre. La seconde partie étant consacré au principe du contrôle flou.

III.1. Bases de la logique floue

Introduction

Afin de manipuler des connaissances imparfaites, la logique floue intervient comme un outil performant. Des citations conditionnelles linguistiques du types « Si – Alors » sont utilisées pour résoudre des problèmes de décision (contrôle), ou pour décrire le comportement dynamique d'un système inconnu ou mal défini (identification).

La théorie des ensembles et des algorithmes flous développés dans les années *60* peut être judicieusement utilisée pour évaluer les expressions linguistiques imprécises et incertaines à l'aide d'un calculateur.

III.1.1 Généralités sur la logique floue

La logique floue (LF) s'utilise dans les cas suivants :

- Systèmes complexes dans lesquels la modélisation s'avère difficile, voir impossible.
- Systèmes multi entrées ou multi sorties
- Systèmes à réponses non linéaires
- l'observation humaine est à l'origine du contrôle du système

Les différents domaines de sont utilisation sont :

- Gestion de projet
- Modélisation de prix, analyse démographique des marchés
- Analyse de rentabilité pour l'achat de compagnie
- Détection de fraude à la protection sociale, identification de criminels
- Gestion des aspirateurs, des systèmes de ventilation et de régulation thermique
- Pilotage d'un système d'autofocus d'appareil photo
- Système d'approche d'une station orbitale pour la navette spatiale américaine
- Lecture automatique, reconnaissance de caractères
- Traitement d'image

La puissance des processeurs disponibles aujourd'hui, dont certains dédiés à la logique floue, ne pose plus de problème pour l'implémentation des techniques de contrôle ou de commande basées sur cette logique. Techniques peu utilisées, mais extrêmement efficaces dans les asservissement des processus non linéaires ou complexes. Leur seul inconvénient étant la difficulté de démontrer la stabilité des systèmes l'utilisant.

Un des exemples des familles de processeur est le WARP (Weight Associative Rul Processor) avec 16 entrées, 16 sorties et 256 règles traitées

Les algorithmes flous sont caractérisés par :

- la facilité d'implémentation
- la robustesse vis-à-vis des incertitudes
- la non nécessité de modélisation des processus complexes
- la possibilité d'intégration du savoir de l'expert

III.1.2. Principe de la logique floue

soit U une collection continue ou discrète d'objet généralement désigné par u.
U est appelé *univers de discours* et sera considéré comme le domaine de fonctionnement du processus.
u représente *l'élément générique* de U

III.1.2.1. ensemble flou et fonction d'appartenance

soit un univers de discours U. Un ensemble flou F est caractérisé par une fonction d'appartenance μ_F prenant des valeurs dans l'intervalle [0,1] qui détermine le degré d'appartenance, aussi appelé possibilité ou coefficient d'appartenance, de l'élément u à l'ensemble F.
Le degré d'appartenance représente la possibilité pour que la variable u ait la qualité associée au sous- ensemble U_i de U.
F est représenté par un ensemble de paires ordonnées (élément générique, degré d'appartenance)

$$\mu_F : \begin{matrix} U \to [0,1] \\ u \to \mu_F(u) \end{matrix} \qquad \text{(III.1)}$$

$$F = \{(u, \mu_F(u)) / u \in U\}$$

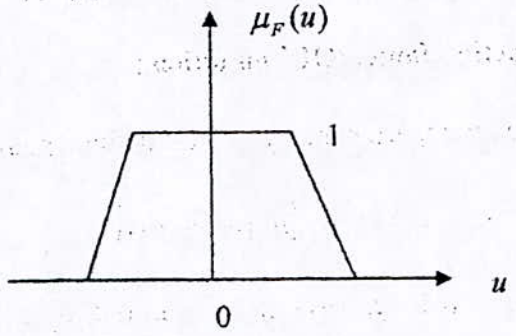


Figure (III.1) : fonction d'appartenance trapézoïdale

III.1.2.2. caractéristiques d'un ensemble flou :

- le support, qu'est l'ensemble des points u dans U, tels que $\mu_F(u) \geq 0$

$$\text{supp(F)} = \{u \in U / \mu_F(u) \geq 0\} \qquad \text{(III.2)}$$

- le noyau, qu'est l'ensemble des points u de U tels que $\mu_F(u) = 1$

$$\text{noy(F)} = \{u \in U / \mu_F(u) = 1\} \qquad \text{(III.3)}$$

- L'ensemble flou vide est noté Φ, il est défini par $\mu_\Phi(u) = 0 \quad \forall u \in U$ (III.4)

- Le plus grand ensemble flou sur U est noté 1_u, il est défini par

$$\mu_{1_U}(u) = 1 \quad \forall u \in U \qquad \text{(III.5)}$$

- Le point de commutation est le cas particulier où l'élément u de U est tel que

$$\mu_F(u) = 0.5 \qquad \text{(III.6)}$$

- Un singleton a une fonction d'appartenance telle que

$$\begin{cases} \mu_F(u) = 1 & qd \quad u = u_0 \\ \mu_F(u) = 0 & qd \quad u \neq u_0 \end{cases} \qquad \text{(III.7)}$$

c'est un ensemble flou correspondant à une variable exacte u_0.

Remarque :

Les fonctions d'appartenance peuvent avoir diverses formes selon leur définition :

- Triangulaire, trapézoïdale
- Gaussienne
- Sigmoïdale...

II.1.2.3. Opérations sur les ensembles flous :

Les opérations d'union, d'intersection et de complémentation dans les ensembles flous sont définies à l'aide de leur fonction d'appartenance.

La disjonction floue, 'OU' ou union :

La fonction d'appartenance $\mu_{A\cup B}$ est définie, pour tout $u \in U$, par :

Mamdani : $\mu_{A\cup B}(u) = \max\{\mu_A(u), \mu_B(u)\}$ (III.8)

Sugeno : $\mu_{A\cup B}(u) = \mu_A(u) + \mu_B(u) - \mu_A(u)\cdot\mu_B(u)$ (III.9)

La conjonction floue, 'ET' ou intersection :

La fonction d'appartenance $\mu_{A\cap B}$ est définie, pour tout $u \in U$, par :

Mamdani : $\mu_{A\cap B}(u) = \min\{\mu_A(u), \mu_B(u)\}$ (III.10)

Sugeno : $\mu_{A\cap B}(u) = \mu_A(u)\cdot\mu_B(u)$ (III.11)

La complémentation, négation ou inverse

La fonction d'appartenance $\mu_{\bar{A}}$ du complément d'un ensemble A est définie, pour tout $u \in U$, dans les deux cas par

$$\mu_{\bar{A}} = 1 - \mu_A(u) \qquad \text{(III.12)}$$

Produit cartésien

Si $F_1, F_2, \ldots, F_n$ sont des ensembles flous dans $U_1, U_2, \ldots, U_n$ respectivement, le produit cartésien de $F_1, F_2, \ldots, F_n$ est un ensemble flou dans l'espace produit $U_1 \times U_2 \times \cdots \times U_n$ ayant pour fonction d'appartenance :

$$\mu_{F_1,\cdots,F_n}(u_1, u_2, \cdots, u_n) = \min\left\{\mu_{F_1}(u_1), \mu_{F2}(u_2), \cdots, \mu_{F_n}(u_n)\right\} \quad \text{(III.13)}$$

ou bien,

$$\mu_{F_1,\cdots,F_n}(u_1, u_2, \cdots, u_n) = \prod_{i=1}^{n} \mu_{F_i}(u_i) \quad \text{(III.14)}$$

Relation floue

Une relation floue d'ordre n est un ensemble flou dans $U_1 \times U_2 \times \cdots \times U_n$ et est exprimé comme

$$R_{U_1 * U_2 * \cdots * U_n} = \left\{\left((u_1, u_2, \cdots, u_n), \mu_R(u_1, u_2, \cdots, u_n)\right) / (u_1, u_2, \cdots, u_n) \in U_1 \times U_2 \times \cdots \times U_n\right\} \quad \text{(III.15)}$$

Composition « Sup-Star »

Si R et S sont deux relations floues dans $U \times V$ et $V \times W$ respectivement, la composition de R et S est une relation floue dénotée par $R \circ S$ et définie par

$$R \circ S = \left\{\left((u, w), \sup_V\left(\mu_R(u, v) * \mu_S(v, w)\right) / u \in U, v \in V, w \in W\right\}\right. \quad \text{(III.16)}$$

III.1.2.4. Variable linguistique et ensembles flous

- Un ensemble flou F dans un univers de discours U est un ensemble flou normal et convexe

Normal : $\max_{u \in U} \mu_F(u) = 1$ (III.17)

Convexe : $\mu_F(\lambda u_1 + (1-\lambda)u_2) \geq \min\left(\mu_F(u_1), \mu_F(u_2)\right)$ (III.18)
Tel que $u_1, u_2 \in U, \lambda \in [0,1]$

Ou bien, si dans U on a $u_1 \leq u_2 \leq u_3$ alors $\mu_F(u_2) \geq \min\left(\mu_F(u_1), \mu_F(u_3)\right)$ (III.19)

- Les ensembles flous sont utilisés afin de représenter des variables linguistiques
Une variable linguistique peut être vue ;

  - Soit comme une variable dont la valeur est un nombre flou
  - Soit comme une variable dont la valeur est définie en termes linguistiques (mots/phrases)

- Une variable linguistique est caractérisée par un 5-uple $(u, T(u), U, G, M)$ dans lequel :

  - u Est le nom de la variable
  - $T(u)$ Est l'ensemble des termes de u, c'est-à-dire, l'ensemble des noms des valeurs linguistiques de u, chacune étant un nombre flou défini sur U.
  - G est une règle syntaxique pour la génération des noms des valeurs de u
  - M est une règle syntaxique qui associe à chaque valeur son sens

III.2. Contrôleur flou

Introduction

La commande floue est l'application floue de la logique floue. Ces algorithmes sont à base de règles linguistiques du type « Si…Alors ». Elle sert à prendre une décision même si l'on ne peut estimer les entrées/sorties qu'à partir e prédicats vagues ou lorsque les entrées/sorties sont entachées d'erreurs.

La résolution d'un problème par la logique floue comprend trois étapes :

- En premier, vient la quantification 'floue' des entrées du système
- En second, l'établissement des règles liant les sorties aux entrées à base des règles floues
- En troisième, la combinaison de ces règles pour la génération des sorties floues

III.2.1. Structure classique d'un contrôleur flou

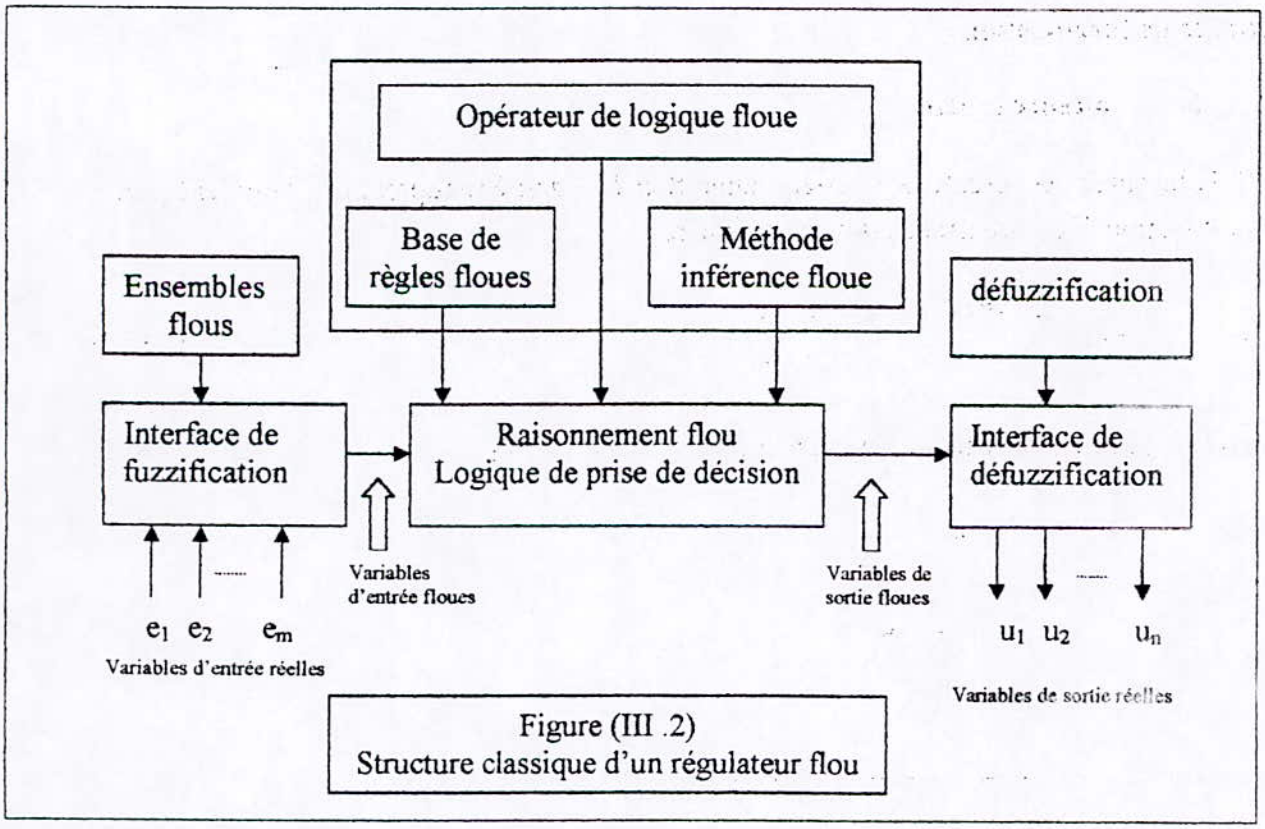


Figure (III .2)
Structure classique d'un régulateur flou

III.2.2. Principe du contrôleur flou

1. procéder à la partition en sous ensembles floues des différents univers de discours que le système impose, et transformer les variables réelles en variables floues (fuzzification)
2. déterminer la base de règle qui va caractériser le fonctionnement désiré du système
3. utiliser les variables floues dans un mécanisme d'inférence qui crée et détermine les variables floues de sortie, en utilisant des opérations sur les fonctions d'appartenance
4. opérer à la défuzzification qui consiste à extraire une valeur réelle de sortie de la fonction d'appartenance du sous-ensemble flou de sortie établi par le mécanisme d'inférence

III.2.2.1. Interface de fuzzification

Un opérateur de fuzzification convertit une valeur numérique en un singleton flou à l'intérieur d'un certain univers de discours
Il convertit des données probabilistes en nombres flous, donc associe à une mesure de la variable u_0 une fonction d'appartenance $\mu_{u_0}(u)$

Méthode de fuzzification :

1. **mesure exacte**

Si la mesure u_0 est exacte, le sous-ensemble flou U_0 doit être représenté par un fait précis. La fonction d'appartenance est alors définie par

$$\mu_{U_0} : U \to U; \begin{cases} \mu_{u_o}(u) = 1 & si \quad u = u_0 \\ \mu_{u_o}(u) = 0 & si \quad u \neq u_0 \end{cases} \qquad \text{(III.20)}$$

L'aspect de cette fonction d'appartenance est donné ci après

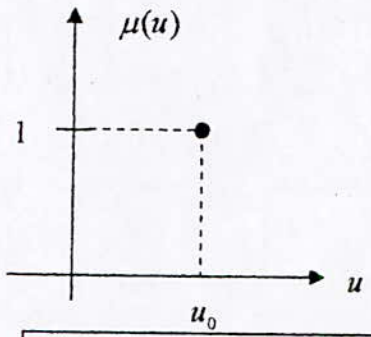


Figure (III.3) : Fonction d'appartenance d'un singleton

2. mesure incertaine

Dans ce cas, le sous-ensemble flou U_0 doit être représenté par un fait imprécis. On utilise alors une méthode de fuzzification qui associe à la variable mesurée u_0 une fonction d'appartenance :

a. $$\mu_{u_0}(u) = \max\left\{0; \frac{|u - u_0|}{\varepsilon}\right\} \qquad \text{(III.21)}$$

dont la représentation est sous forme de triangle de base ε fonction de l'importance relative des erreurs de mesures. Plus elle sont importantes plus la mesure de la variable u_0 devient imprécise, et donc plus la base du triangle doit s'élargir

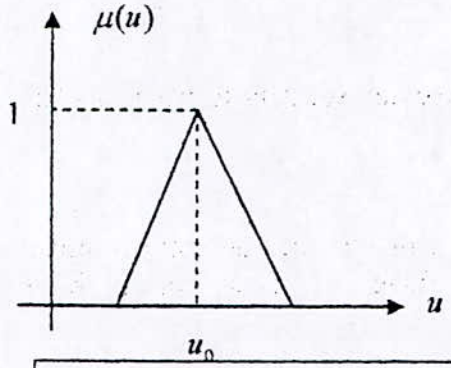


Figure (III.4) : Fonction d'appartenance triangulaire

Ce triangle isocèle est utilisée dans le cas d'un fait incertain tel que u est à peu près égale à u_0

b. dans le cas d'une affirmation de u est à peu près compris entre u_1 et u_2, la forme de la fonction d'appartenance est trapézoïdale

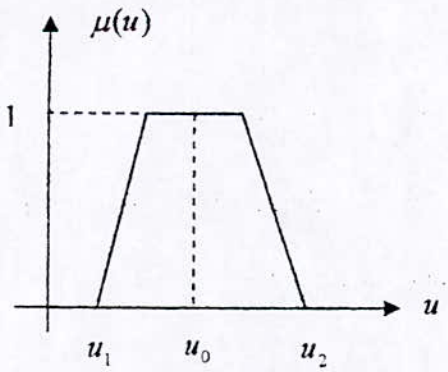
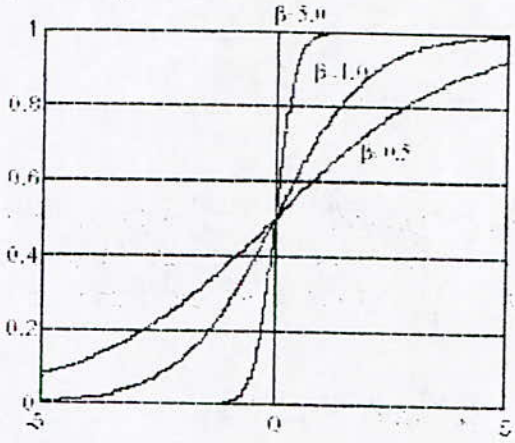


Figure (III.5) : Fonction d'appartenance trapézoïdale

c. la forme sigmoïde et couramment la plus utilisée et est donnée par

$$\mu(u) = \frac{1}{1+\exp(-\beta u)} \qquad \text{(III.22)}$$

Figure (III.6) : Fonction d'appartenance sigmoïdale



d. la forme gaussienne, sera explicité et utilisée dans le prochain chapitre.

III.2.2.2. Base de règle et définitions

C'set un bloc d'existence virtuelle qui regroupe l'ensemble des définitions utilisées dans le contrôle floue (univers de discours, partitions floues,...), ainsi que la base de règle « Si...Alors »de la stratégie du contrôle de l'expert.

Partition floue

La partition floue d'un univers de discours U consiste à définir n sous-ensembles flous F_i de façon à recouvrir U. C'est-à-dire que pour tout élément u de U, il faut assurer une appartenance minimale ε à l'union des F_i :

$$\bigcup F_i \supseteq U_\varepsilon = \left\{u \in U; \mu_{U_\varepsilon}(u) = \varepsilon\right\} \qquad \text{(III.23)}$$

En terme de fonction d'appartenance :

$$\forall u \in U, \mu_{F_1}(u) \cup \mu_{F_2}(u) \cup \cdots \cup \mu_{F_n}(u) \geq \varepsilon \qquad \text{(III.24)}$$

L'ensemble de sous-ensembles flous à définir dans une partition d'un univers de discours est fixé par l'expert.
Les ensembles flous (termes linguistiques) ont habituellement un sens tel :

NG Négatif grand
NM Négatif moyen
NP Négatif petit
EZ Environ zéro
PP Positif petit
PM Positif moyen
PG Positif grand

Les prédicats « petit », « moyen », « grand » ou «négatif », « positif », « proche de zéro » servent à la régulation, les autres servent à la poursuite.

En contrôle classique cinq prédicats peuvent suffire, dans le cas extrême on peut aller jusqu'à sept.

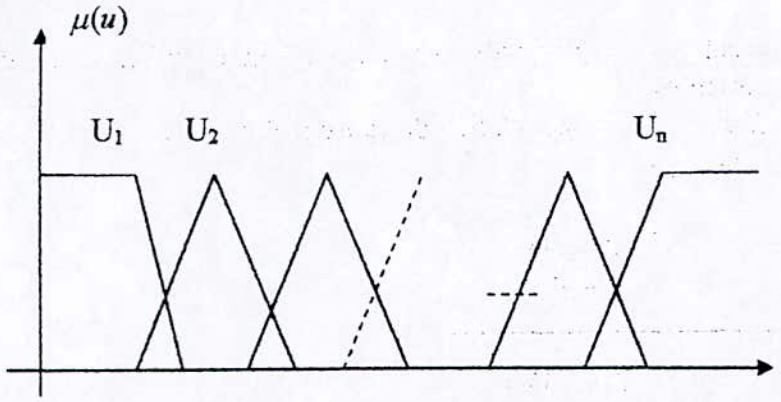


Figure (III.7)
Univers de discours U

Base de règles

Les relations entre classes d'évènement possibles en entrée et des commandes correspondantes sont caractérisées par une base de règle « Si…Alors…»:
Par conséquent, si l'on considère n univers de discours U_i pour les prémisses de règles floues et si pour chaque univers U_i on définit une partition en m_i sous-ensembles flous, le nombre maximum de règles r_{max} est de

$$r_{max} = \prod_{i=1}^{n} m_i \qquad \text{(III.25)}$$

Remarque

- Le nombre de règles définies par l'expert peut être inférieur à r_{max}, c'est le cas s'il existe des configuration impossible à obtenir
- Le nombre de sous-ensembles flous, définissant la partition de l'univers de discours de la commande, n'est pas forcément égale au nombre de règles. Des configurations différentes peuvent aboutir aux mêmes conclusions

L'implication floue

L'implication floue la plus utilisée en commande floue est celle définie par Mamdani, donnée par

$$\forall u_1 \in U_1, \forall u_2 \in U_2, R(u_1,u_2) = \min(\mu_1(u_1), \mu_2(u_2)) \qquad \text{(III.26)}$$

Le raisonnement approximatif

Concept de base : la représentation de propositions par des formules affectant des ensembles flous comme valeurs aux variables.
Soient deux variables $x \in X$ et $y \in Y$, et une relation de cause à effet entre x et y, $y = f(x)$
Alors, on peut effectuer l'inférence :

Prémisse	$y = f(x)$
Fait	$x = x'$
Conséquence	$y = f(x')$

(III.27)

La plupart du temps, on ne connaît le lien de cause à effet f entre x et y qu'en certaines valeurs x particulières. On a une base de règle :

$$R_i : si \quad x = x_i \quad alors \quad y = y_i \,,\ i = \overline{1,n} \qquad \text{(III.28)}$$

Il faut alors, connaissant $x' \in X$, trouver $y' \in Y$ correspondant à x', conformément à la base de règles.
Le problème de base du raisonnement approximatif est de trouver la fonction d'appartenance de la conséquence C d'une base de règles $R_i, \overline{i = 1,n}$, et le fait « x *est* x' »

Mécanisme d'inférence

C'est l'application du raisonnement approximatif. Il permet de calculer le sous-ensemble flou $\mu(u_0)$ relatif à la commande du système, à partir de la base de règles fournie par l'expert et l'entrée fuzzifiée u_0 telle que $u_0 = [u_{0,1}, u_{0,2}, \ldots, u_{0,n}]^T$

Ces règles floues lient les sorties aux entrées par des conditions linguistiques. La sortie est obtenue par déduction floue.

Dans ce contexte, on distingue deux sortes de règles d'inférence :

- Le « Modus Ponens Généralisé » (MPG), utilisé dans le raisonnement par chaînage avant (déduction successive de faits) et est particulièrement utile dans la commande plane
- Le « Modus Tollens Généralisé » (MTG), utilisé en chaînage d'inférence arrière (vérification d'hypothèses) plutôt utilisé en diagnostic

Dans les inférences ou implications floues interviennent les opérateurs *ET* et *OU*. L'opérateur *ET* s'applique aux variables à l'intérieur d'une règle, tandis que l'opérateur *OU* lie les différentes règles.

Afin de traiter numériquement une inférence floue, plusieurs méthodes existent. Les plus importantes sont :

- Méthode d'inférence min-max
- Méthode d'inférence max-prod

Le tableau suivant résume ces deux principales méthodes

Min-max	Max-prod
ET ⇔ min	*ET* ⇔ min
OU ⇔ max	*OU* ⇔ max
Alors ⇔ min	*Alors* ⇔ *prod*(×)
Combinaison des règles activées *OU* ⇔ max	

Tableau -1- règle d'inférence floue

Qualitativement, ces règles donnent sensiblement les mêmes résultats
La plus utilisée est la première méthode qu'est de Mamdani :
- Détermination de la conséquence → min .
- Agrégation des règles → max

Remarque :

Afin de résumer les règles floues, une « table de décision » ou « table d'inférence », anti-diagonale a été adoptée. Elle a pour forme :

sortie inférée ↘	entrées floues
Entrées floues	

Tableau -2- table de décision

III.2.2.3. Interface de défuzzification

Les méthodes d'inférences fournissent une fonction d'appartenance résultante pour la variable de sortie. C'st une information floue qu'il convient de transformer en une grandeur de commande précise : c'est l'étape de défuzzification.

Il existe plusieurs techniques dont nous citons :

- Méthode du maximum
- Méthode du centre de gravité
- Méthode de la moyenne pondérée
- Méthode des surfaces
- Méthode des hauteurs

Les trois premières citées ci-dessus, sont les plus utilisées.

Méthode du maximum : elle consiste à ne considérer pour chaque sortie que la règle présentant le maximum de validité.

Méthode de la moyenne pondérée : elle considère comme valeur de sortie la moyenne des valeurs préconisées par chaque règle, pondérées par leur degré respectif de validité.

Méthode du centre de gravité : C'est la plus performante. Elle consiste à tracer sur le même diagramme, les différentes zones correspondantes à chacune des règles, et à calculer le centre de gravité de la zone considérée.
Si $\mu_{res}(u)$ est la fonction d'appartenance résultante, l'abscisse du cintre de gravité u_0 peut être déterminé à l'aide de la relation générale suivante :

- Dans le cas d'un univers de discours continu :

$$u_0^* = \frac{\int_a^b u\mu_{res}(u)du}{\int_a^b \mu_{res}(u)du} \text{ sur l'intervalle } [a,b] \qquad \text{(III.29)}$$

- Dans le cas d'un univers de discours discret

$$u_0^* = \frac{\sum_{i=1}^{n} \mu_u(w_i).w_i.S_i}{\sum_{i=1}^{n} \mu_u(w_i).S_i} \qquad \text{(III.30)}$$

n : nombre des niveaux de quantification de la sortie du contrôleur
S_i : surface de la $i^{ème}$ fonction d'appartenance u de la variable de sortie
w_i : valeur de la commande locale pour laquelle la fonction d'appartenance atteint la valeur maximale $\mu_u(w_i)$

Dans le cas où les fonctions d'appartenance sont de forme symétrique et de distribution uniforme l'équation devient :

$$u_0^* = \frac{\sum_{i=1}^{n} \mu_u(w_i).w_i}{\sum_{i=1}^{n} \mu_u(w_i)} \qquad \text{(III.31)}$$

Conclusion

Ce chapitre a servi pour monter les différentes bases de la logique floue ainsi que le principe de fonctionnement d'un contrôleur à base de cette logique.

L'inférence floue dite de Mamdani, sera explicité et utilisé dans le quatrième chapitre pour l'obtention du modèle neuro-flou.

Nous traitons dans ce qui suit les étapes de conception d'une commande neuro- floue et son application au bras de robot manipulateur flexible tout en la simulant à l'aide du logiciel MATLAB. En premier, dans le quatrième chapitre, l'algorithme d'apprentissage utilisé est celui de back propagation. Le problème des minima locaux rencontré lors du processus d'optimisation des algorithmes d'apprentissage des neurones nous a mené à considérer d'autres méthodes d'optimisation dites globales. De ce fait, en second, dans le cinquième chapitre, une méthode d'estimation globale a été utilisée en incluant les algorithmes évolutionnistes dont les algorithmes génétiques, qui nous permettent de soulever le problème des minima locaux.

4

Commande neuro-floue

Introduction

Un organe de commande à base de logique floue est constitué d'une table de décision reliant les informations mesurées sur le système à la variable de commande. Chaque case de la table est interprétée comme une règle d'inférence floue. Le chevauchement des ensembles flous correspondant aux parties conditions des règles permet, lorsqu'une erreur précise est traitée par le régulateur, de réaliser une interpolation entre les conclusions des règles. On peut ainsi reconstituer une loi de commande continue à partir d'un jeu de règles de régulation symbolique.

Les algorithmes neuronaux sont fondés sur une approche totalement différente : il ne s'agit plus d'imiter un raisonnement humain mais, au contraire, de reproduire des « réflexes » en entraînant un système sur divers exemples.

Un neurone, élément de base de la connaissance, est modélisé par une fonction non linéaire appliquée à une combinaison linéaire pondérée des sorties d'autres neurones. L'interconnexion entre un grand nombre de ces modules élémentaires permet de modéliser une fonction non linéaire complexe sans avoir à l'écrire explicitement.
L'apprentissage d'un réseau revient, en général, à optimiser un critère de ressemblance entre sa sortie et le comportement désiré, par rapport aux poids portés par les liens entre les différents neurones.

Les méthodes utilisées actuellement, permettent une approche non linéaire de la modélisation et de la commande. La difficulté réside dans le fait qu'il n'y a pas de procédure systématique pour décider combien de couches et combien de neurones par couche sont nécessaires pour bien approcher une fonction.

La mise d'un système d'inférence floue sous forme d'un réseau neuronal a donné naissance à la commande neuro-floue.

Le présent chapitre est consacré à l'élaboration d'un réseau neuro-flou en utilisant l'inférence floue de Mamdani, l'algorithme d'apprentissage étant celui de Back propagation.

IV.1. Réseaux de neurones artificiels

Introduction

Les réseaux de neurones reposent sur des bases mathématiques solides qui permettent d'envisager des applications industrielles à grande échelle, notamment dans le domaine de la modélisation, étant une approche efficace et générique des problèmes non linéaires par apprentissage.

McCulloch et **Pitts** étaient les premiers en 1943 à proposer un modèle neuronal et inventent le **neurone formel** qui portera leurs noms. Ce n'était qu'en 1949, que le mécanisme d'apprentissage a été mis en œuvre sous forme d'une règle de modification des connexions par **Hebb.**

Le premier **réseau de neurone artificiel (RNA)** est apparu en 1958 grâce aux travaux de **Rosenblatt** qui conçoit le « perceptron ». Les réseaux à perceptrons sont des modèles limités et incapables de résoudre des problèmes non linéaires, **Hopfield** montre, en 1982, l'intérêt d'utiliser des réseaux récurrents pour la compréhension et la modélisation des processus.

En parallèle, **Werbos** conçoit son algorithme de rétro- propagation ("back-propagation") de l'erreur, qui offre un mécanisme d'apprentissage, pour les réseaux multicouches de type Perceptron (appelés MLP pour Multi-layer Perceptron), fournissant ainsi un moyen simple d'entraîner les neurones des couches cachées.

IV.1.1. Le Neurone biologique

Le neurone biologique (*figure IV.1*) est une cellule vivante spécialisée dans le traitement des signaux électriques.
Les neurones sont reliés entre eux par des liaisons appelées axones. Ces axones vont eux-mêmes jouer un rôle important dans le comportement logique de l'ensemble. Ils conduisent les signaux électriques de la sortie d'un neurone vers l'entrée (synapse) d'un autre neurone.
Les neurones font une sommation des signaux reçus en entrée et en fonction du résultat obtenu vont fournir un courant en sortie.

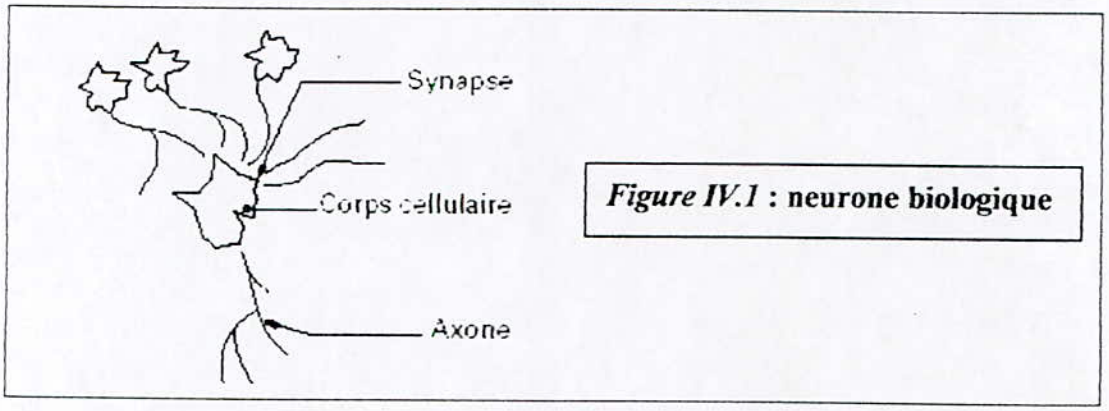


***Figure IV.1* : neurone biologique**

IV.1.2. Le Neurone artificiel (formel)

Le neurone artificiel (fig*ure IV.2*) image d'un neurone biologique est un processeur élémentaire. Il reçoit un nombre variable d'entrées en provenance de neurones appartenant à un niveau situé en aval. À chacune de ces entrées est associé un poids w représentatif de la force de la connexion.
Chaque processeur élémentaire (neurone) est doté d'une sortie unique, qui se ramifie pour alimenter un nombre variable de neurones appartenant à un niveau situé en amont.

IV.1.3. Présentation d'un RNA

Un RNA est un ensemble de neurones formels (d'unités de calcul simples, de noeuds processeurs) associés en couches (ou sous-groupes) et fonctionnant en parallèle.

Dans un réseau, chaque sous-groupe fait un traitement indépendant des autres et transmet le résultat de son analyse au sous-groupe suivant. L'information donnée au réseau va donc se propager couche par couche, de la couche d'entrée à la couche de sortie, en passant soit par aucune, une ou plusieurs couches intermédiaires (dites couches cachées). Il est à noter qu'en fonction de l'algorithme d'apprentissage, il est aussi possible d'avoir une propagation de l'information à reculons ("back propagation").

Les RNA ont la capacité de stocker de la connaissance empirique et de la rendre disponible à l'usage. Les habiletés de traitement (et donc la connaissance) du réseau vont être stockées dans les poids synaptiques, obtenus par des processus d'adaptation ou d'apprentissage.

Pour un RNA, l'apprentissage peut être considéré comme le problème de la mise à jour des poids des connexions au sein du réseau, afin de réussir la tâche qui lui est demandée, selon un algorithme ou procédure.

L'apprentissage est la caractéristique principale des RNA et il peut se faire de différentes manières et selon différentes règles.

IV.1.4. Les types d'apprentissage

- Le mode supervisé

L'apprentissage supervisé consiste, étant donné un vecteur d'apprentissage, à déterminer le vecteur des poids des neurones capables de prédire le même vecteur de sortie à partir du même vecteur d'entrée. Dans cette sorte d'apprentissage, le réseau s'adapte par comparaison entre le résultat qu'il a calculé, en fonction des entrées fournies, et la réponse attendue en sortie. Ainsi, le réseau va se modifier jusqu'a ce qu'il trouve la sortie désirée.

- Le renforcement

Le renforcement est en fait une sorte d'apprentissage supervisé. Dans cette approche le réseau doit apprendre la corrélation entrée/sortie via une estimation de son erreur, c'est-à-dire du rapport échec/succès. Le réseau va donc tendre à maximiser un index de performance qui lui est fourni, appelé *signal de renforcement*. Le système étant capable ici, de savoir si la réponse qu'il fournit est correcte ou non, mais il ne connaît pas la bonne réponse.

- Le mode non supervisé (ou auto-organisationnel)

L'apprentissage est qualifié de non supervisé lorsque seules les valeurs d'entrée sont disponibles. Dans ce cas, l'apprentissage est basé sur des probabilités. Le réseau va se modifier en fonction des régularités statistiques de l'entrée et établir des catégories, en attribuant et en optimisant une valeur de qualité, aux catégories reconnues.

- Le mode hybride

Le mode hybride reprend en fait les deux autres approches, puisque une partie des poids va être déterminée par apprentissage supervisé et l'autre partie par apprentissage non supervisé.

IV.1.5. Les règles d'apprentissage

- Règle de correction d'erreurs

Cette règle s'inscrit dans le paradigme ou modèle d'apprentissage supervisé, c'est-à-dire, dans le cas où on fournit au réseau, une entrée et la sortie correspondante. Si on considère *y*, la sortie calculée par le réseau et *d*, la sortie désirée, le principe de cette règle est d'utiliser l'erreur (d-y), afin de modifier les connexions et de diminuer ainsi l'erreur globale du système. Le réseau va donc s'adapter jusqu'à ce que y soit égal à d.

- Apprentissage de Boltzmann

Ce qu'il faut savoir tout d'abord, c'est que les réseaux de Boltzmann sont des réseaux symétriques récurrents (voir I.6.2) et qu'ils possèdent deux sous-groupes de cellules, le premier étant relié à l'environnement (cellules dites visibles) et le second ne l'étant pas (cellules dites cachées). Cette règle d'apprentissage est de type stochastique (qui relève partiellement du hasard) et elle consiste à ajuster les poids des connexions, de telle sorte que l'état des cellules visibles satisfasse une distribution probabiliste souhaitée.

- Règle de Hebb

Cette règle, basée sur des données biologiques, modélise le fait que si des neurones, de part et d'autre d'une synapse, sont activés de faon synchrone et répétée, la force de la connexion synaptique sera croissante. Dans ce cas l'apprentissage est localisé, c'est-à-dire que la modification d'un poids synaptique *wij* ne dépend que de l'activation d'un neurone *i* et d'un autre neurone *j*.

- Règle d'apprentissage par compétitions

La particularité de cette règle, est que l'apprentissage ne concerne qu'un seul neurone.
Le principe de cet apprentissage est de regrouper les données en catégories. Les patrons similaires vont donc être rangés dans une même classe, en se basant sur les corrélations des données, et seront représentés par un seul neurone, on parle de « winner-take-all ».
Dans un réseau à compétition simple, chaque neurone de sortie est connecté aux neurones de la couche d'entrée, aux autres cellules de la couche de sortie (connexions inhibitrices) et à elle-même (connexion excitatrice). La sortie va donc dépendre de la compétition entre les connexions inhibitrices et excitatrices.

IV.1.6. Les différents types de RNA

IV.1.6.1. Les réseaux "feed-forward"

IV.1.6.1.a. Les Perceptrons

- Le Perceptron monocouche

C'est un réseau simple, il ne se compose que d'une couche d'entrée et d'une couche de sortie. Il a été conçu dans un but premier de reconnaissance des formes. Cependant, il peut aussi être utilisé pour faire de la classification et pour résoudre des opérations logiques simples (telle "ET" ou "OU"). Sa principale limite est qu'il ne peut résoudre que des problèmes linéairement séparables. Il suit généralement un apprentissage supervisé selon la règle de correction de l'erreur (ou selon la règle de Hebb).

- Le Perceptron multicouches (PMC)

C'est une extension du précédent, avec une ou plusieurs couches cachées entre l'entrée et la sortie. Chaque neurone dans une couche est connecté à tous les neurones de la couche précédente et de la couche suivante (excepté pour les couches d'entrée et de sortie) et il n'y a pas de connexions entre les cellules d'une même couche.. Il peut résoudre des problèmes non-linéairement séparables et des problèmes logiques plus compliqués, et notamment le fameux problème du *XOR*. Il suit un apprentissage supervisé selon la règle de correction de l'erreur.

IV.1.6.1.b. Les réseaux à fonction radiale

On les nomme aussi RBF (Radial Basic Functions). Leur architecture est la même que pour les PMC. Cependant, les fonctions de base utilisées ici sont des fonctions Gaussiennes. Les RBF sont donc employés dans les mêmes types de problèmes que les PMC à savoir, la classification et l'approximation de fonctions. L'apprentissage le plus utilisé pour les RBF est le mode hybride en utilisant la règle de correction de l'erreur ou d'apprentissage par compétition.

IV.1.6.2. Les réseaux "feedback" ou à connexion récurrente

Un réseau de ce type signifie qu'une ou plusieurs sorties de neurones d'une couche aval sont connectées aux entrées des neurones de la couche amont ou de la même couche. Ces connexions récurrentes ramènent l'information en arrière par rapport au sens de propagation défini dans un réseau multicouche.
Les réseaux à connexions récurrentes sont des réseaux puissants car ils sont séquentiels plutôt que combinatoires. La rétroaction de la sortie vers l'entrée permet à un réseau de ce type de présenter un comportement temporel.

On y trouve par exemple :

- Les cartes auto- organisatrices de Kohonen

Ce type de réseau, est un réseau à apprentissage non supervisé

- Les réseaux de Hopfield :

Ce sont des réseaux récurrents et entièrement connectés. Ils fonctionnent comme une mémoire associative non linéaire. L'application principale des réseaux de Hopfield est la résolution de problèmes d'optimisation. Le mode d'apprentissage utilisé est le mode non supervisé.

- Les ART : les réseaux ART (Adaptative Resonnance Theorie) sont des réseaux à apprentissage par compétition.

On trouve dans [TOU92], le principe du fonctionnement da chaque réseau présenté brièvement ici.

IV.1.6.3. Réseau à connexions complexes

Chaque neurone est connecté à tous les neurones du réseau y compris lui-même, c'est la structure d'interconnexion la plus générale.

IV.1.7. domaine d'application

- **La classification de données**

Les RNA sont capables d'apprendre l'association entrée/ sortie. Du fait de cette capacité, les RNA vont donc être utilisés par exemple dans, la reconnaissance de caractères [GOS96], la reconnaissance vocale, la classification de cellules sanguines ou encore la carte descriptive d'un circuit imprimé.

- **La catégorisation de données**

Il s'agit de rassembler des données similaires en une même catégorie. Les RNA à apprentissage non supervisé réalisent ce type de tâche et donc s'emploient dans la compression de données ou l'exploration d'analyse de données.

- **L'approximation de fonctions**

Les capacités de classification et de catégorisation des RNA sont encore exploitées ici, puisqu'ils servent parfois à approximer une fonction inconnue (par contre dont on connaît les paires d'entrés/sorties) par une fonction connue. C'est un problème fréquent dans le domaine de la modélisation.

- **Problèmes de prédiction/prévision**

Les RNA sont également utilisés lorsqu'il faut, à partir de données recueillies à un temps t, prédire les données à un temps $t + 1$. Les RNA intéressent donc les domaines de la science et de l'ingénierie mais aussi l'économie de marchés ou la météorologie.

- **L'optimisation**

Pour faire de l'optimisation, le problème est soit, de maximiser le succès soit, de minimiser l'erreur. Encore une fois, les réseaux à apprentissage non supervisé sont particulièrement adaptés pour ce type de tâche et vont donc intéresser des domaines très différents : les mathématiques, les statistiques, l'ingénierie, la médecine ou l'économie.

- **Le contrôle**

Les intérêts essentiels des réseaux neuronaux dans la commande neuronale sont leur capacité naturelle à réaliser une commande adaptative et à synthétiser des modèles non linéaires, dans les systèmes dynamiques.

IV.1.8. Réseaux artificiels neuronaux adaptés à la commande automatique

Les réseaux artificiels neuronaux permettent de représenter les relations fonctionnelles complexes nécessitées par les systèmes de régulation modernes. Les informations enregistrées sont envoyées aux neurones d'un réseau et mémorisées grâce aux facteurs de pondération. L'apprentissage des réseaux s'effectue sur la base d'exemples réels et si possible en ligne.
Étant donné que l'on peut entraîner ces réseaux de neurones à acquérir le comportement souhaité, ils constituent des modules universels faits pour être utilisés là où il s'avère nécessaire de décrire par l'exemple, un comportement complexe et de l'illustrer à l'aide d'une représentation fonctionnelle.
Aussi les réseaux trouvent-ils leur utilisation dans l'élaboration de modèles décrivant les caractéristiques de transfert pour des systèmes non linéaires, très difficiles à décrire. Ils assurent des tâches de prédiction de signaux, de régulation adaptative, de filtrage adaptatif, de classification, de surveillance et de diagnostic des systèmes [SOR96].

IV.1.9. Modélisation neuronale et réseaux multicouches (MLP)

Le perceptron multicouches est le modèle neuronal élémentaire, en particulier pour la modélisation et l'identification des systèmes.

IV.1.9.1. Sélection de l'architecture du réseau

Les méthodes de sélection de l'architecture optimale pour un problème d'estimation particulier sont des techniques itératives. Elles sont appliquées à la sélection d'entrées, de neurones cachés ou de poids individuels. Elles peuvent être divisées en trois groupes :

- Sélection ascendante, qui ajoute des neurones si besoin et les paramètres correspondants à un modèle.
- Sélection descendante, qui supprime les paramètres les moins importants.
- Régression pas à pas, qui combine les deux approches.

Sélection d'architecture et estimation des paramètres sont fortement lié en se referant au premier groupe

IV.1.9.2. Estimation des paramètres

Pour les MLP, les méthodes d'apprentissage et donc d'estimation des paramètres sont nombreuses.
Les méthodes locales, du premier ou second ordre, sont basées sur le calcul du Gradient.
Les méthodes d'optimisation globales, ou stochastiques (aléatoires) incluent en particulier les algorithmes évolutionnistes. De tels algorithmes sont utilisés non seulement pour l'estimation des paramètres, mais aussi pour la détermination de l'architecture et l'adaptation des règles d'apprentissage par exemple les techniques hybrides qui combinent les méthodes du gradient et algorithmes évolutionnistes.

Les performances des algorithmes d'apprentissage pour les MLP sont sensibles aux facteurs suivants :

- Le critère d'erreur, qui peut être simplement quadratique, robuste aux valeurs aberrantes
- Le principe d'initialisation des poids, qui influence considérablement le nombre d'itérations
- le critère d'arrêt de l'apprentissage
- les paramètres spécifiques aux différentes méthodes

IV.1.9.3. Caractéristiques des réseaux multicouches (MLP)

Les réseaux à perceptron multicouches sont des modèles non linéaires à capacité d'approximation universelle, flexibles et parcimonieux :

- approximateurs universel : ils peuvent approximer n'importe quelle fonction non linéaire continu d'un espace de dimension fini dans un autre, avec une précision arbitraire fixée, en les développant en série de fonction paramétrées.

- Parcimonieux : ces approximateurs nécessitent moins de paramètres ajustables à estimer que les développements en série de fonctions fixes pour atteindre les mêmes degrés d'approximation.

- flexibles : plus la relation à modéliser est complexe, plus nombreux sont les neurones et les paramètres dans le modèle neuronale sans tout de même changer la forme globale du modèle.

Les principaux problèmes qui se posent, sont la rencontre de minima locaux de la fonction de coût, ce qui conclut prématurément la phase d'apprentissage, ainsi qu'une convergence relativement lente de l'algorithme de rétro- propagation.

IV.2. Développement des réseaux multicouches

IV.2.1. Le neurone formel

Le neurone formel (*figure 2*) est un processeur élémentaire qui réalise une somme pondérée des signaux qui lui parviennent. La valeur de cette sommation est comparée à un seuil et la sortie du neurone est une fonction ***non linéaire*** du résultat :

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

$$y = \varphi(u) \qquad \text{(IV.2)}$$

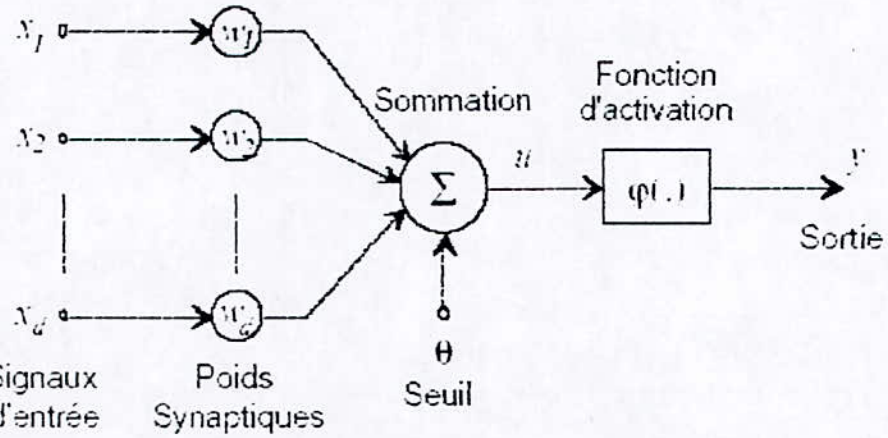


***Figure IV.2* : le neurone formel**

En prenant la convention de noter par

- $X = [-1 \quad x_1 \quad \cdots \quad x_d]^T$, le vecteur d'entrées augmenté
- $W = [\theta \quad w_1 \quad \cdots \quad w_d]^T$, le vecteur de poids augmenté

Il vient $$y = \varphi(W^T X) \qquad \text{(IV.3)}$$

IV.2.2. Le perceptron

Architecture du perceptron

Le perceptron (*figure IV.3*) est la forme la plus simple de réseau de neurones, et permet de classifier correctement des objets appartenant à deux classes linéairement séparables. Il consiste en un seul neurone qui possède un seuil ainsi qu'un vecteur de poids synaptiques ajustables

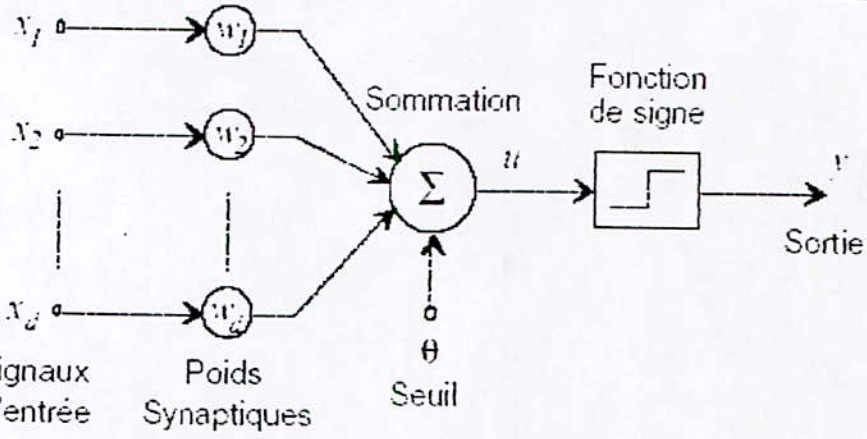


***Figure IV.3* : Le perceptron**

Le perceptron associe à chaque classe une fonction linéaire qui s'exprime

$$\Phi_i(X) = W_i^T X \qquad \text{(IV.4)}$$

où

$W_i = [w_0 \quad w_1 \quad \cdots \quad w_n]^T$, est le vecteur de coefficient de pondération

$X = [-1 \quad x_1 \quad \cdots \quad x_d]^T$, est le vecteur de caractéristiques augmenté d'un objet classifié

la fonction utilisée par les neurones est la fonction signe :

$$sng(u) \overset{\Delta}{=} \begin{cases} 1 & si \quad u > 0 \\ -1 & \text{sinon} \end{cases} \qquad \text{(IV.5)}$$

Les paramètres du perceptron, c'est-à-dire les poids synaptiques des neurones, peuvent être déterminés grâce à un entraînement supervisé. La règle d'apprentissage du perceptron, développée originellement par Rosenblatt, est assurée de converger si les données sont linéairement séparables. Il ne converge toutefois pas lorsque les objets à classifier ne sont pas linéairement séparables. Un autre algorithme peut alors être utilisé, tel le critère des moindres carrés de l'erreur.

IV.2.3. Architecture du Réseau multicouches

La mise en cascade de perceptrons conduit à ce qu'on appelle le perceptron multicouches *(figure IV.4)* cependant la fonction utilisée est une fonction continue.
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.

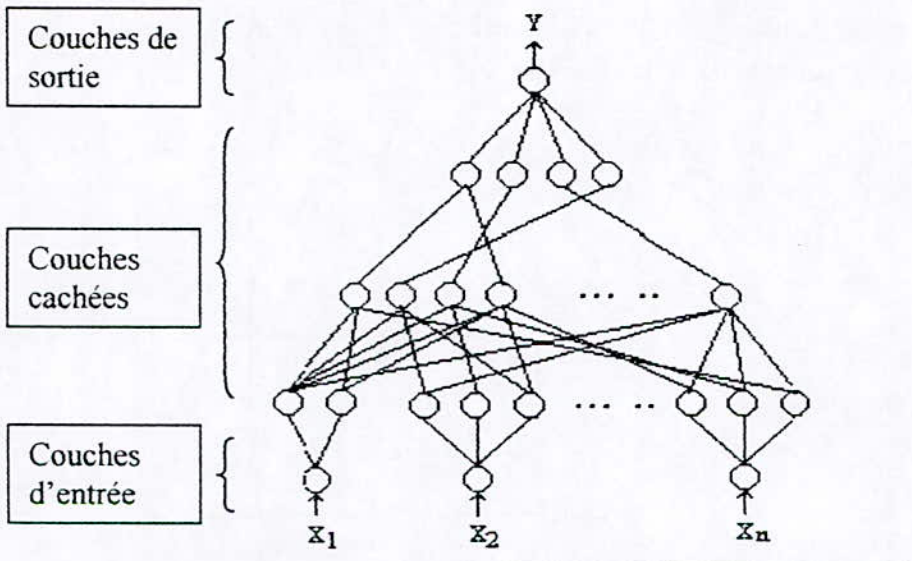


***Figure IV.4* : architecture du réseau multicouches**

Les fonctions réalisées par un réseau multicouches sont de la forme :

$$\Phi_i(X) = \varphi_{l,i}\left(-w_{l,i0} + \sum_{j=1}^{h_l} w_{l,ij}\varphi_{l,j}\left(-w_{l-1,i0} + \sum_{q=1}^{n} w_{l-1,jq}x_q\right)\right) \qquad \text{(IV.6)}$$

où

$\varphi_{l,i}(\cdot)$ représente la fonction d'activation du neurone i de la couche l

$W_{l,i} = \begin{bmatrix} w_{l,i0} & w_{l,i1} & w_{l,i2} & \cdots & w_{l,ih_{l-1}} \end{bmatrix}^T$ est le vecteur de poids augmenté de ce même neurone

h_l vaut le nombre de neurones de la couche l

$X = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^T$ est le vecteur de caractéristiques présenté à l'entrée du réseau

La fonction d'activation des neurones doit absolument être non linéaire, car sans elle, le perceptron multicouches ne ferait qu'implanter une série de transformations linéaires consécutives, qui pourraient dès lors se réduire à une seule.

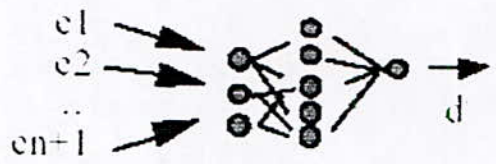
IV.3. Le modèle neuro-flou

IV.3.1. Architecture du modèle

Le modèle neuro-flou est obtenu en modélisant un système d'inférence flou par un réseau à perceptron multicouches. Le réseau de neurones est utilisé pour approximer l'implication floue à partir d'exemples d'inférence floue et ainsi, nous obtenons l'inférence floue connexionniste.

Si g est la fonction établissant la correspondance entre les coefficients d'incertitude des prémisses et celui de la règle avec les coefficients d'incertitude de la conclusion (*figure IV.5*), les exemples d'apprentissage à soumettre au réseau seront des vecteurs de la forme $(e_1, e_2, \cdots, e_n, e_{n+1}, d)$ avec $d = g(e_1, e_2, \cdots, e_n, e_{n+1})$. Les valeurs de $(e_1, e_2, \cdots, e_n)$ sont celles des prémisses. La valeur de e_{n+1} est celle du coefficient d'incertitude de la règle, d est la valeur du coefficient d'appartenance de la conséquence.

L'apprentissage permet d'adapter la fonction g réalisée par le réseau de neurones au comportement décrit par les exemples d'apprentissage.



Base d'exemples
((e1, e2, ... en+1), d
avec
d=g(e1, e2, ... en+1

***Figure IV.5* : apprentissage de l'inférence floue par réseaux multicouches**

IV.3.2. Modèle d'inférence floue de Mamdani et Sugeno

La représentation de connaissance en logique floue suit deux classes. La première (classe A), suggérée par Takagi et Sugeno, représente la classe générale des systèmes non linéaires statiques ou dynamiques. Elle est basée sur la partition de l'espace de l'entrée en sous espaces linéaires, la sortie est une combinaison linéaire des entrées, et est représenté par la règle suivante :

$$R^i : if \quad x_1 \quad is \quad A_1^i \; and \quad x_2 \quad is \quad A_2^i, \cdots, and \quad x_m \quad is \quad A_m^i$$
$$then \quad y^i = a_0^i + a_1^i x_1 + \cdots + a_m^i x_m \qquad \text{(IV.7)}$$

où $R^i (i = 1,2,\cdots,c)$ indique la *ième* règle floue, et $x_j (j = 1,2,\cdots,m)$ est l'entrée et y^i est la sortie de la règle floue R^i. $A_1^i, A_2^i, \cdots, A_m^i (i = 1,2,\cdots,c)$ sont les fonctions d'appartenance floues.

Cette approche d'approximation de fonction non linéaire, combinant linéairement plusieurs sous systèmes linéaires, en décomposant l'espace d'entrée en sous espaces floues, et chaque

espace floue est représenté par une fonction linéaire, ne permet pas à la sortie de se décrire linguistiquement. L'implémentation du modèle obtenu est très difficile, le programme de l'optimisation paramétrique étant non-linéaire.

La seconde classe (classe B), dans la modélisation floue a été développée par Mamdani et utilisée par Sugeno et d 'autres.
Les règles sont présentées comme suit

$$R^i : If \quad x_1 \quad is \quad A_1^i \; and \quad x_2 \quad is \quad A_2^i, \quad \cdots \quad , and \quad x_m \quad is \quad A_m^i$$
$$then \quad y^i \quad is \quad B^i \qquad \text{(IV.8)}$$

où les $A_1^i, A_2^i, \cdots, A_m^i, B^i (i = 1,2,\cdots c)$ sont des fonctions d'appartenance floues.

Cette approche a quelques avantages en dépit de la première. Les conséquences des règles sont présentées linguistiquement et aide à mieux comprendre la structure du modèle. Le modèle de cette approche est facile à implémenter.

C'est ce dernier modèle qui sera utilisé dans la suite du travail.

IV.3.3. Inférence floue connexionniste de Mamdani

IV.3.3.1. Architecture du modèle

La (*figure IV.6*), représente l'architecture du perceptron multicouches considéré afin de modéliser le système d'inférence flou de Mamdani.

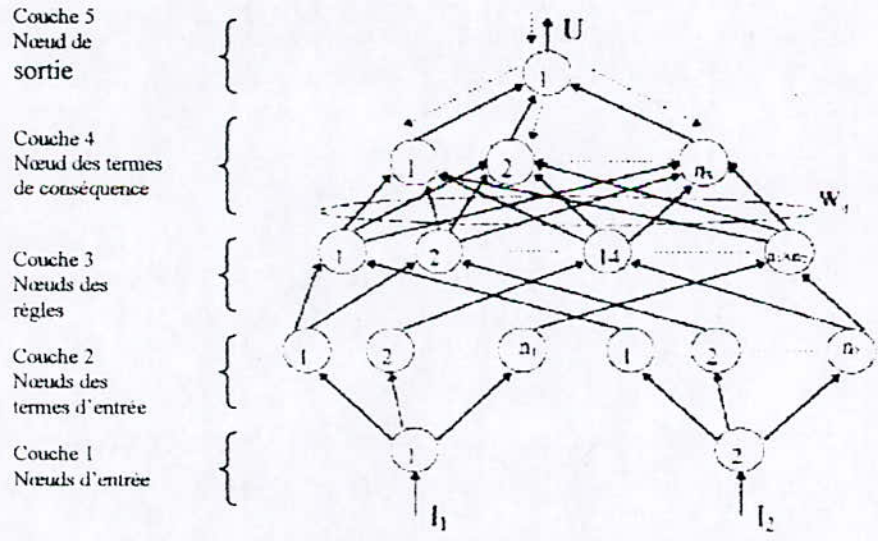


***Figure IV.6* : modèle multicouche connexionniste de l'inférence de Mamdani**

Le modèle a un total de cinq couches comme représenté par Lin et Lee [FAR98]. On considère deux entrées et une sortie, et donc deux nœuds en entrée et un seul nœud en sortie. Les nœuds de la première couche sont les *nœuds d'entrée*, ils transmettent le signal d'entrée directement aux autres nœuds de la couche en aval. La couche 5 est la couche de sortie.

Les nœuds de la deuxième et quatrième couche sont des *nœuds de conditions*. Ils fonctionnent comme des fonctions de transfert afin d'exprimer les entrées/sorties linguistiques floues.

Les fonctions adaptées dans ce réseau sont les fonctions cloches ou Gaussiennes, caractérisées par leur moyenne m et leur variance σ, qui correspondent aux paramètres à ajuster par l'algorithme d'apprentissage.

Chaque base floue, de la première et la seconde entrées de la première couche, consiste en n_1 et n_2 bases floues respectivement. Les termes linguistiques considérés sont P, EZ, N ou $PG, PM, PP, EZ, NP, NM, NG$ classés par ordre décroissant dans le réseau.

Par conséquent, $n_1 + n_2$ nœuds et n_3 nœuds sont inclus dans la couche 2 et 4 respectivement.

Chaque nœud de la troisième couche représente « *le nœud de la règle flou* ». Une règle pour un nœud. Au total, il y en a $n_1 \times n_2$ nœuds dont chaque règle est appliquée à deux variables floues d'entrée.

Les liens de la troisième et quatrième couche définissent respectivement les prés- conditions et les conséquences de la règle. Ceux dans 4 sont variables afin d'ajuster les sorties (réponse du noeud). Cependant, ceux de la deuxième et cinquième couche sont constants de part et d'autre des nœuds voisins.

Donc le modèle neuro-flou peut ajuster les règles floues en modifiant les liens de la couche 4 et donc en modifiant les paramètres des fonctions floues dans les nœuds de la deuxième et quatrième couche.

IV.3.3.2. Modèle connexionniste

Nous adoptons les notations suivantes :

net_i^L : la valeur d'entrée du *ième* nœud dans la couche L

O_i^L : la valeur de la sortie du *ième* nœud de la couche L

m_i^L, σ_i^L : la moyenne et la variance de la fonction d'appartenance du *ième* nœud de la couche L

$w_{i,j}$: le poids du lien qui lie la sortie du nœud j de la couche 3 à l'entrée du nœud i de la couche 4

Couche 1 :

$$O_1^1 = I_1$$
$$O_2^1 = I_2 \qquad \text{(IV.9)}$$

Couche 2 :

$$net_i^2 \begin{cases} O_1^1 & pour \quad i = \overline{1, n_1} \\ O_2^1 & pour \quad i = \overline{n_1 + 1, n_1 + n_2} \end{cases} \tag{IV.10}$$

$$O_i^2 = \exp\left[-\left(\frac{net_i^2 - m_i^2}{\sigma_i^2}\right)^2\right] \text{ pour } i = \overline{1, n_1 + n_2} \tag{IV.11}$$

Les poids des liens de cette couche sont fixés à l'unité.

Couche 3 :

Chaque nœud de cette couche reçoit deux entrées de la couche en aval. La règle d'inférence utilisée ici est celle min-max. La sortie du nœud étant déterminée par la conjonction floue.
Par conséquent, cette couche a pour modèle :

$$net_i^3 = \min(O_j^2, O_k^2) \quad , i = (j-1) + (k - n_2) \tag{IV.12}$$

$$\text{pour } j = \overline{1, n_1} \text{ et } k = \overline{n_1 + 1, n_1 + n_2}$$

$$O_i^3 = net_i^3 \text{ pour } i = 1, 2, \ldots, n_1 \times n_2$$

Les poids des liens de cette couche sont aussi fixés à l'unité.

Couche 4 :

Chaque nœud de cette couche accomplie une disjonction floue, afin de réunir les règles pour une même variable floue.

Cette couche est modélisée comme suit :

$$net_i^4 = \sum_{j=1}^{n_1 * n_2} w_{ij} O_j^3 \tag{IV.13}$$

$$O_i^4 = \min(1, net_i^4) \text{ pour } i = \overline{1, n_3} \tag{IV.14}$$

$w_{ij} = 0 \quad ou \quad 1$: il exprime le poids de l'association de la *jème* règle avec la *ième* sortie.

Couche 5 :

Cette couche a pour fonction la défuzzification (par centre de gravité). Le nœud de la cinquième couche calcul la sortie du signal de commande du modèle connexionniste flou.
Le centre de gravité de la partie défuzzifiée est simulé par :

$$net_1^5 = \sum_{j=1}^{n_3} m_j^4 \sigma_j^4 O_j^4 \tag{IV.15}$$

et $$O_1^5 = \frac{net_1^5}{\sum_{j=1}^{n_3} \sigma_j^4 O_j^4} \qquad \text{(IV.16)}$$

où $$w_{5,j} = m_j^4 \sigma_j^4 \qquad \text{(IV.17)}$$

Remarque :

Dans une commande neuro-floue, le régulateur est d'abord représenté sous forme neuronale puis ses paramètres sont déterminés par un algorithme d'apprentissage.

IV.4. utilisation de la méthode d'apprentissage locale

IV.4.1. Introduction

L'algorithme d'apprentissage du perceptron multicouches, connu sous le nom d'*algorithme de rétro-propagation*, nécessite toutefois que les fonctions d'activations des neurones soient continues et dérivables.

Le principe utilisé par la rétropropagation ("backpropagation" en anglais) de gradient est la minimisation d'une fonction dépendante de l'erreur.
Une perception intuitive de cet algorithme consiste à considérer l'apprentissage comme la recherche sur la surface de coût de la position de coût minimal. A chaque configuration de poids correspond un coût. Le gradient est une estimation locale de la pente de la surface. La minimisation du gradient permet de parcourir cette surface orthogonalement aux courbes de niveau d'un pas fixé (*figure IV.7*). Les problèmes rencontrés durant l'apprentissage résultent des zones très plates et des minima locaux.

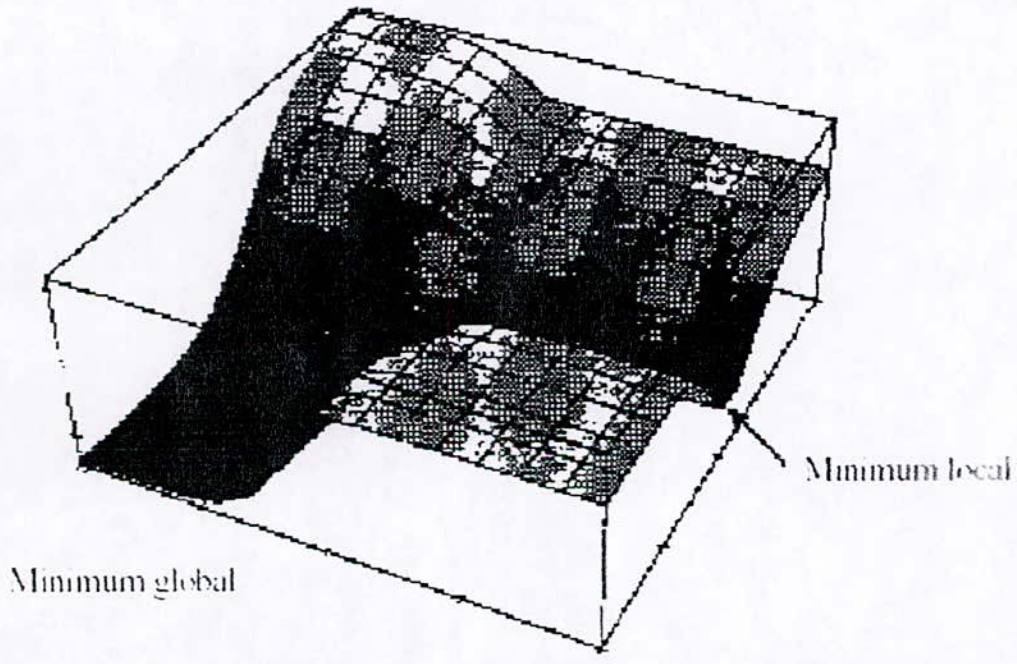


***Figure IV.7* : Recherche de la zone de coût minimal**

IV.4.2. La rétro- propagation du gradient

Pour la méthode de rétro- propagation du gradient deux phases sont à considérer :

- une phase de propagation, qui consiste à présenter une configuration d'entrée au réseau puis à propager cette entrée de la couche d'entrée à la couche de sortie en passant par les couches cachées.

- Une phase de rétro- propagation, qui consiste à minimiser l'erreur commise sur l'ensemble des exemples présentés. L'erreur considérée comme une fonction des poids synaptiques, et représentée par la somme des différences au carrée entre les réponses calculées et celles désirées pour tous les membres contenus dans l'ensemble d'apprentissage.

Soit $Y_l^{(k)} = \begin{bmatrix} -1 & y_{l,1}^{(k)} & \cdots & y_{l,h_l}^{(k)} \end{bmatrix}^T$ la notation du vecteur augmentés des sorties de la couche l lorsque le vecteur $X_k = \begin{bmatrix} x_{k1} & x_{k2} & \cdots & x_{kd} \end{bmatrix}^T$ est présenté à l'entrée du réseau, et par $W_{l,i} = \begin{bmatrix} w_{l,i0} & w_{l,i1} & \cdots & w_{l,ih_{l-1}} \end{bmatrix}^T$ le vecteur augmentés des poids du neurone i de la couche l

Le gradient est donné par

$$W_{l,ij}(\tau+1) = w_{l,ij}(\tau) - \alpha \frac{\partial E^{(k)}}{\partial w_{l,ij}} \tag{IV.18}$$

où

- α est le taux d'apprentissage, $\alpha > 0$

- $$\frac{\partial E^{(k)}}{\partial w_{l,ij}} = \delta_{l,i}^{(k)} y_{l-1,j}^{(k)} \tag{IV.19}$$

avec,

pour un neurone de la couche de sortie : $\delta_{l,i}^{(k)} = \dot{\varphi}_{l,i}\left(W_{l,i}^T Y_{l-1}^{(k)}\right)\left(y_{l,i}^{(k)} - t_i^{(k)}\right)$ (IV.20)

et pour un neurone d'une couche cachée : $\delta_{l,i}^{(k)} = \dot{\varphi}_{l,i}\left(W_{l,i}^T Y_{l-1}^{(k)}\right)\sum_{q=1}^{h_{l+1}} w_{l+1,qi}\delta_{l+1,q}^{(k)}$ (IV.21)

où $\dot{\varphi}_{l,i}(u^{(k)}) = \left.\frac{\partial \varphi}{\partial u}\right|_{u=u^{(k)}}$ (IV.22)

Le calcul de $\frac{\partial E^{(k)}}{\partial w_{l,ij}}$ est effectué comme suit :

Les sorties sont calculées par

Pour $i > 0$ $\quad y_{l,j}^{(k)} = \varphi\left(W_{l,i}^T Y_{l-1}^{(k)}\right) = \varphi(u_{l,i}^{(k)})$ (IV.23)

en posant $u_{l,i}^{(k)} = W_{l,i}^T Y_{l-1}^{(k)}$ (IV.24)

Le processus de calcul est initialisé en posant $Y_0^{(k)} = X_k$

Le gradient de l'erreur peut s'exprimer

$$\frac{\partial E^{(k)}}{\partial w_{l,ij}} = \frac{\partial E^{(k)}}{\partial u_{l,i}^{(k)}} \frac{\partial u_{l,i}^{(k)}}{\partial w_{l,ij}} \tag{IV.25}$$

en posant $$\frac{\partial E^{(k)}}{\partial u_{l,j}^{(k)}} = \delta_{l,i}^{(k)} \tag{IV.26}$$

où $\delta_{l,i}^{(k)}$ est appelé *gradient local* de l'erreur.

En tenant compte de (IV.24), on a :

$$\frac{\partial u_{l,i}^{(k)}}{\partial w_{l,ij}} = y_{l-1,j}^{(k)} \tag{IV.27}$$

Il vient ainsi,

$$\frac{\partial E^{(k)}}{\partial w_{l,ij}} = \delta_{l,i}^{(k)} y_{l-1,j}^{(k)} \tag{IV.28}$$

le gradient local peut se développer selon

$$\frac{\partial E^{(k)}}{\partial u_{l,i}^{(k)}} = \sum_{q=1}^{h_{l+1}} \frac{\partial E^{(k)}}{\partial u_{l+1,q}^{(k)}} \frac{\partial u_{l+1,q}^{(k)}}{\partial u_{l,i}^{(k)}} \tag{IV.29}$$

ou encor

$$\frac{\partial E^{(k)}}{\partial u_{l,i}^{(k)}} = \sum_{q=1}^{h_{l+1}} \frac{\partial E^{(k)}}{\partial u_{l+1,q}^{(k)}} \frac{\partial u_{l+1,q}^{(k)}}{\partial y_{l,i}^{(k)}} \frac{\partial y_{l,i}^{(k)}}{\partial u_{l,i}^{(k)}} \tag{IV.30}$$

de (IV.26) $$\frac{\partial E^{(k)}}{\partial u_{l+1,q}^{(k)}} = \delta_{l+1,q}^{(k)} \tag{IV.31}$$

de (IV.24) $$\frac{\partial u_{l+1,q}^{(k)}}{\partial y_{l,i}^{(k)}} = w_{l+1,qi} \tag{IV.32}$$

de (IV.23) $$\frac{\partial y_{l,i}^{(k)}}{\partial u_{l,i}^{(k)}} = \dot{\varphi}_{l,i}(u_{l,i}^{(k)}) \tag{IV.33}$$

Il vient

$$\delta_{l,i}^{(k)} = \dot{\varphi}_{l,i}(u_{l,i}^{(k)}) \sum_{q=1}^{h_{l+1}} w_{l+1,qi} \delta_{l+1,q}^{(k)} \tag{IV.34}$$

Le gradient local doit être calculé en premier pour le neurone de la couche de sortie en substituant l par L.

IV.4.3. La Phase d'Apprentissage

L'apprentissage du perceptron multicouches est supervisé, et consiste à adapter les poids des neurones de manière à ce que le réseau soit capable de réaliser une transformation donnée, représentée par un ensemble d'exemples constitué d'une suite de N vecteurs d'entrées $X_k = [x_{k1} \quad x_{k2} \quad \cdots \quad x_{kd}]^T$ associée à une autre suite de vecteurs de sorties désirées $T^{(k)} = [t_1^{(k)} \quad t_2^{(k)} \quad \cdots \quad t_{h_l}^{(k)}]^T$

Le critère des moindres carrés de l'erreur peut être utilisé pour définir la fonction coût non linéaire à minimiser, exprimée comme suit :

$$E = \frac{1}{2}\sum_{k=1}^{N}\sum_{i=1}^{h_L}\left(y_{L,i}^{(k)} - t_i^{(k)}\right)^2 \qquad \text{(IV.35)}$$

où

N est le nombre d'exemple d'apprentissage ;

L est le nombre de couche du réseau ;

h_l vaut le nombre de neurones que contient la couche l ;

$y_{l,i}^{(k)}$ désigne la sortie du neurone i de la couche l lorsque le vecteur X_k est présenté à l'entrée du réseau ;

$t_i^{(k)}$ représente la valeur de sortie désirée pour le neurone i de la dernière couche lorsque le vecteur X_k est présenté à l'entrée du réseau ;

IV.4.4. Règle d'apprentissage

La règle d'apprentissage du perceptron multicouches se résume comme suit :

1. initialiser l'ensemble des vecteurs de poids synaptiques $W_{l,i}$ à des valeurs aléatoires ;
2. appliquer un exemple d'apprentissage X_k à l'entrée du perceptron multicouches et évaluer l'erreur quadratique commise par le réseau selon $E^{(k)} = \sum_{i=1}^{h_l}\left(y_{l,i}^{(k)} - t_i^{(k)}\right)^2$
3. corriger les poids selon $W_{l,ij}(\tau+1) = w_{l,ij}(\tau) - \alpha\dfrac{\partial E^{(k)}}{\partial w_{l,ij}}$
4. Incrémenter τ et k et reprendre au point 2

Remarque :

Le gradient local $\delta_{l,i}^{(k)}$ d'un neurone est en effet calculé à partir des gradients locaux $\delta_{l+1,i}^{(k)}$ des neurones de la couche ultérieure. Le calcul des gradients commence donc par la dernière couche, et est ensuite propagé progressivement de celle-ci vers la première couche du réseau.

IV.5. Application et simulation de la méthode d'estimation locale au bras flexible

Dans ce qui suit, nous allons synthétiser un régulateur neuro-flou de type Mamdani à sept classes d'appartenance, puis l'appliquer au bras flexible présenté dans chapitre II.

Voici la structure du contrôleur neuro-flou :

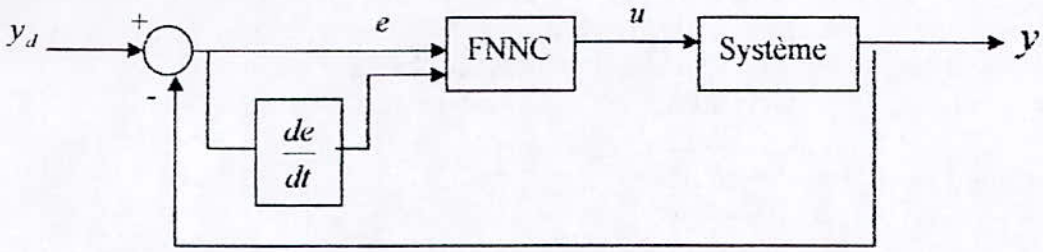


***Figure IV.8* : Contrôleur neuro-flou**

Nous considérons l'erreur et sa dérivée comme entrées du régulateur neuro-flou, et les règles décisionnelles du contrôle sont données par la table de décision suivante :

Δe / e	u	N	Z	P
N		N	N	Z
Z		N	Z	P
P		Z	P	P

***Tableau IV.I* : table de décision de à 3 fonctions d'appartenance**

Modèle connexionniste :

$O_1^1 = e$

$O_1^2 = \Delta e$

$n_1 = 3 \quad et \quad n_2 = 3 \quad et \quad n_3 = 9$

Le gradient :

Les paramètres se trouvent dans la dernière couche, de ce fait le gradient est donné par :

$$W_{5,j}(\tau+1) = w_{5,j}(\tau) - \alpha \frac{\partial E^{(k)}}{\partial w_{5,j}} \tag{IV.36}$$

où

$$\frac{\partial E^{(k)}}{\partial w_{5,j}} = \frac{\sum_{j=1}^{n_4} O_j^4}{\sum_{j=1}^{n_4} \sigma_j^4 O_j^4} \left(O_1^{5(k)} - t^{(k)} \right) \tag{IV.37}$$

IV.5.1. Analyse des performances

Les essais en simulation effectués sur notre bras ont montré l'insuffisance d'une telle commande pour l'élimination totale des vibrations. Les principaux problèmes qui se posent, sont la rencontre de minima locaux de la fonction de coût, ce qui conclut prématurément la phase d'apprentissage, ainsi qu'une convergence relativement lente de l'algorithme de rétro propagation (***figures IV9, IV10***). Une des astuces d'apprentissage proposée est l'utilisation d'un terme de moment pouvant permettre d'accomplir plus aisément cette phase d'apprentissage et d'utiliser d'autre algorithmes améliorés dis 'accéléré'.
De plus le mode d'apprentissage choisi influence l'évolution de cette dernière.

IV.5.1.1. Utilisation d'un Terme de Moment

La formule d'adaptation des poids synaptiques, fournie par l'algorithme de rétro propagation, est souvent modifiée par l'ajout d'un terme de moment. Dans ce cas, la valeur d'un poids synaptique n'est plus seulement adaptée proportionnellement à la dérivée de la fonction de coût par rapport à ce poids, mais est également modifiée en fonction de la correction appliquée à l'instant précédent. Sous forme mathématique, la formule d'adaptation des poids synaptiques s'écrit alors:

$$W_{l,ij}(\tau+1)-W_{l,ij}(\tau)=-\alpha\frac{\partial E^{(k)}}{\partial w_{l,ij}}+\beta\left(W_{l,ij}(\tau)-W_{l,ij}(\tau-1)\right) \qquad \text{(IV.38)}$$

Où β, $0 \le \beta \le 1$, est un paramètre qui est appelé terme de moment

Les valeurs des paramètres et doivent âtre déterminées empiriquement, de manière à limiter la fréquence d'apparition de deux phénomènes qui sont opposés et qui conduisent à un ralentissement de l'évolution de l'apprentissage.

IV.5.1.2. Les Différents Modes d'Apprentissage

Il existe deux modes principaux d'apprentissage, selon la façon dont les vecteurs de poids synaptiques sont adaptés.

L'Apprentissage En Ligne

Il consiste à modifier les valeurs de ces poids synaptiques immédiatement après la présentation d'un objet. Dans ce cas, seul le gradient instantané de la fonction de coût est utilisé pour l'adaptation des paramètres du système. Sous la condition que les objets soient présentés au réseau de neurones de manière aléatoire, l'apprentissage en-ligne rend la recherche du minimum de la fonction de coût stochastique en nature, ce qui rend moins probable, pour l'algorithme de rétro-propagation, de tomber dans un minimum local.

L'Apprentissage Hors-ligne

Le second mode principal d'apprentissage consiste, quant à lui, à accumuler les gradients instantanés consécutifs, et à n'effectuer l'adaptation des poids synaptiques que lorsque l'ensemble des objets d'apprentissage ont été présentés au perceptron multicouches. Cette dernière méthode permet de mieux estimer le gradient réel de la fonction de coût, puisqu'il est à présent calculé à partir d'un ensemble d'objets, plutôt qu'à partir d'un seul.

L'efficacité relative des modes d'apprentissage en-ligne et hors-ligne dépend essentiellement du problème considéré. L'apprentissage en-ligne présente cependant l'avantage que, pour une seule présentation de l'ensemble de la base de données, il implique de multiples phases d'adaptations des poids synaptiques lorsque des données similaires se représentent, ce qui se produit fréquemment pour des bases de données très étendues.

IV.5.1.3. Algorithmes Accéléré

D'autre algorithmes d'apprentissage dis 'accéléré' sont proposés pour l'augmentation du temps d'apprentissage Les deux principaux sont l'Algorithme de **Sanossian & Evans**, et l'Algorithme de **Vogl**.

L'Algorithme de Sanossian & Evans

Comme l'algorithme de rétro-propagation classique utilise une même valeur du taux d'apprentissage pour tous les poids synaptiques, ce dernier n'est optimal dans aucun des cas, et la convergence de l'algorithme demeure relativement lente. La solution proposée par Sanossian et Evans est de choisir une valeur du taux d'adaptation, indépendamment pour chaque poids synaptique, et qui décroît lorsque l'amplitude du module de la dérivée de l'erreur par rapport au poids considéré augmente. Ces valeurs du taux d'adaptation sont déterminées à l'avance et tabulées.
Par rapport à l'algorithme de rétro-propagation classique, il apparaît une diminution du nombre d'itérations nécessaires. La durée globale de l'apprentissage se trouve également réduite, du fait que l'algorithme de Sanossian & Evans ne requiert pratiquement pas de calculs supplémentaires par itération.
L'inconvénient majeur de cette méthode d'apprentissage, est que la phase d'optimisation des nombreux paramètres de l'algorithme requiert beaucoup de temps lorsque le problème étudié est de grandes dimensions.

L'Algorithme de Vogl

La méthode d'apprentissage développée par Vogl se rapproche plus de l'algorithme de rétro-propagation initial. Le taux d'adaptation des poids synaptiques est identique pour tous ceux-ci, mais peut lui-même être modifié d'une itération à l'autre.

- tant que la valeur de la fonction de coût décroît, le taux d'adaptation est augmenté:

$$E(\tau) < E(\tau - 1) \qquad \alpha(\tau + 1) = \alpha(\tau).(1 + \varepsilon_a) \tag{IV.39}$$

- il demeure par contre inchangé lorsque l'erreur n'augmente que légèrement:

$$E(\tau - 1) \le E(\tau) < (1 + \rho).E(\tau - 1) \qquad \alpha(\tau + 1) = \alpha(\tau) \tag{IV.40}$$

- et il est diminué si l'erreur augmente de manière significative:

$$E(\tau - 1).(1 + \rho) \le E(\tau) \qquad \alpha(\tau + 1) = \alpha(\tau).(1 - \varepsilon_d) \tag{IV.41}$$

Dans ce dernier cas, les poids synaptiques sont en outre réinitialisés à leur dernière valeur ayant permis d'obtenir une erreur qui n'a pas nécessité de diminution du taux d'apprentissage. Cette méthode apparaît surtout intéressante par le fait que la valeur du taux d'adaptation se modifie automatiquement, tout au long de l'apprentissage, en fonction des circonstances rencontrées. Le choix de la valeur initiale du taux d'adaptation apparaît dès lors moins

critique, et une phase d'optimisation n'est *à priori* vraiment nécessaire que pour les trois paramètres principaux de cet algorithme, à savoir:

- le taux d'accroissement ε_a ;
- le taux de décroissance ε_d;
- le seuil de tolérance de dépassement de l'erreur ρ.

Tous ces facteurs sont des réels positifs inférieurs à l'unité.

C'est le mode d'utilisation en-ligne de l'algorithme d'apprentissage qui apparaît comme étant le plus intéressant. En outre, la modification du taux d'adaptation au fur et à mesure de l'apprentissage permet de mieux suivre l'évolution de ce dernier, et d'atteindre ainsi une vitesse de convergence élevée. Le recours à un terme de moment ne permet pas d'améliorer la rapidité de convergence de l'algorithme, et s'avère donc ici inutile.

L'algorithme qui apparaît alors être le plus avantageux est celui proposé par **Vogl**. Il permet une mise au point automatique du taux d'adaptation tout au long de la phase d'apprentissage.

IV.5.1.4. Testes et performances

Nous avons effectué différents tests sur le bras et qui sont :

- la poursuite de trajectoire
- le lâché de masse
- le test de charges

La trajectoire principale est une sigmoïde et a pour équation :

$$\theta(t) = \frac{\Delta}{2\pi}\left(\omega t - \sin(\omega t)\right) \quad ; t_0 \le t \le t_f \qquad \text{(IV.41)}$$

Avec $\omega = \dfrac{2\pi}{t_f}$ et $\Delta = \theta_{t_f} - \theta_{t_0}$ et $t_f = 10.(1+\theta_{t_f})$

Les trajectoires choisies ont pour caractéristiques :

$\theta_0 = 0 \quad rad$

Trajectoire1 : $\theta_{tf} = 0.1 \quad rad$

Trajectoire2 : $\theta_{tf} = 0.5 \quad rad$

Trajectoire3 : $\theta_{tf} = 1 \quad rad$

Trajectoire4 : $\vartheta_{tf} = \begin{cases} \theta_t(t) \quad rad \quad pour \quad 0 \le t \le t_f \\ \theta_{tf} \quad pour \quad t > t_f \end{cases}$

Trajectoire5 : $\vartheta_{tf} = \begin{cases} \theta_t(t) \quad rad \quad pour \quad 0 \le t \le t_f \\ 0 \quad pour \quad t > t_f \end{cases}$

Trajectoire6 : $\vartheta_{tf} = \begin{cases} \theta_t(t) \quad rad \quad pour \quad 0 \le t \le t_f \\ t_f = 10.(1+\theta_{t_f}) \quad et \quad t_{simulation} = 2 * t_f \end{cases}$

Les facteurs ont pour valeur :

- le taux d'accroissement $\varepsilon_a = 0.01$
- le taux de décroissance $\varepsilon_d = 0.09$
- le seuil de tolérance de dépassement de l'erreur $\rho = 0.001$

Le test de charge a été effectués sur trois masses : 0.5, 1 et 1.4 Kg pour une réponse à la poursuite de la trajectoire2. Les résultats sont présentés sur la ***figure (IV.16).*** Nous remarquons le manque de robustesse de cette commande vis-à-vis la variation de charge.

La ***figure (IV.12)*** nous montre la robustesse de la commande vis-à-vis le lâché de charge à la sixième seconde de la poursuite de la trajectoire2.

Les figures ***(IV.9, 10, 11, 13, 14, 15)*** montrent la réponse en position du bras flexible à la poursuite des trajectoires présentées ci-dessus.

Dans la ***figure (IV.14)***, pour poursuite de la trajectoire5, il faut signaler que la commande n'est pas nulle entre 0 et 5 secondes, mais que c'est de l'ordre de 0.01 N.m.

L'algorithme de Vogl nous a permit d'accélérer relativement le processus d'apprentissage mais n'a pas contribué à l'annulation totale de l'erreur si ce n'est l'amélioration des performances.

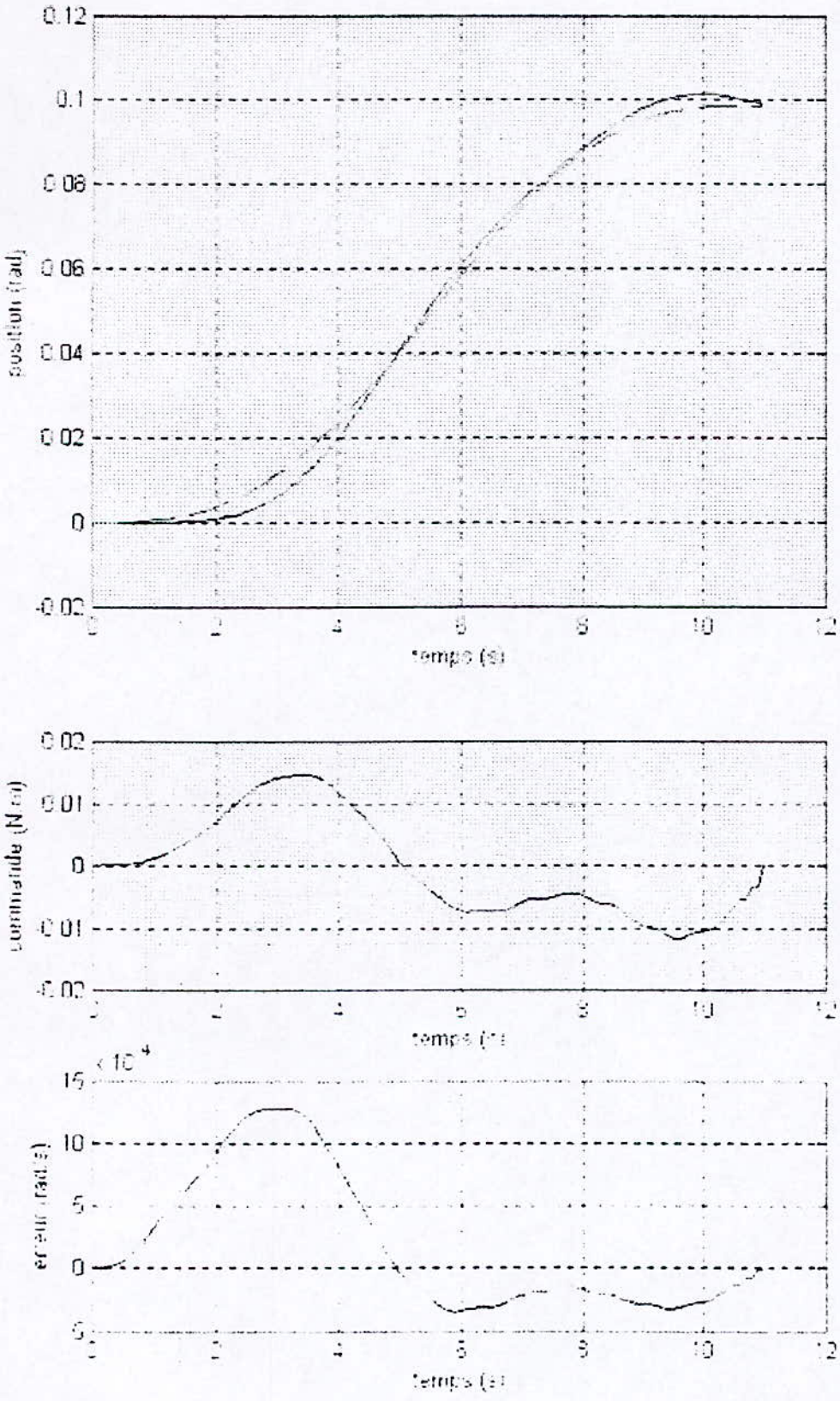


Fig (VI.9) : ***réponse en position du bras flexible, commande et erreur correspondantes à la poursuite de la trajectoire1***

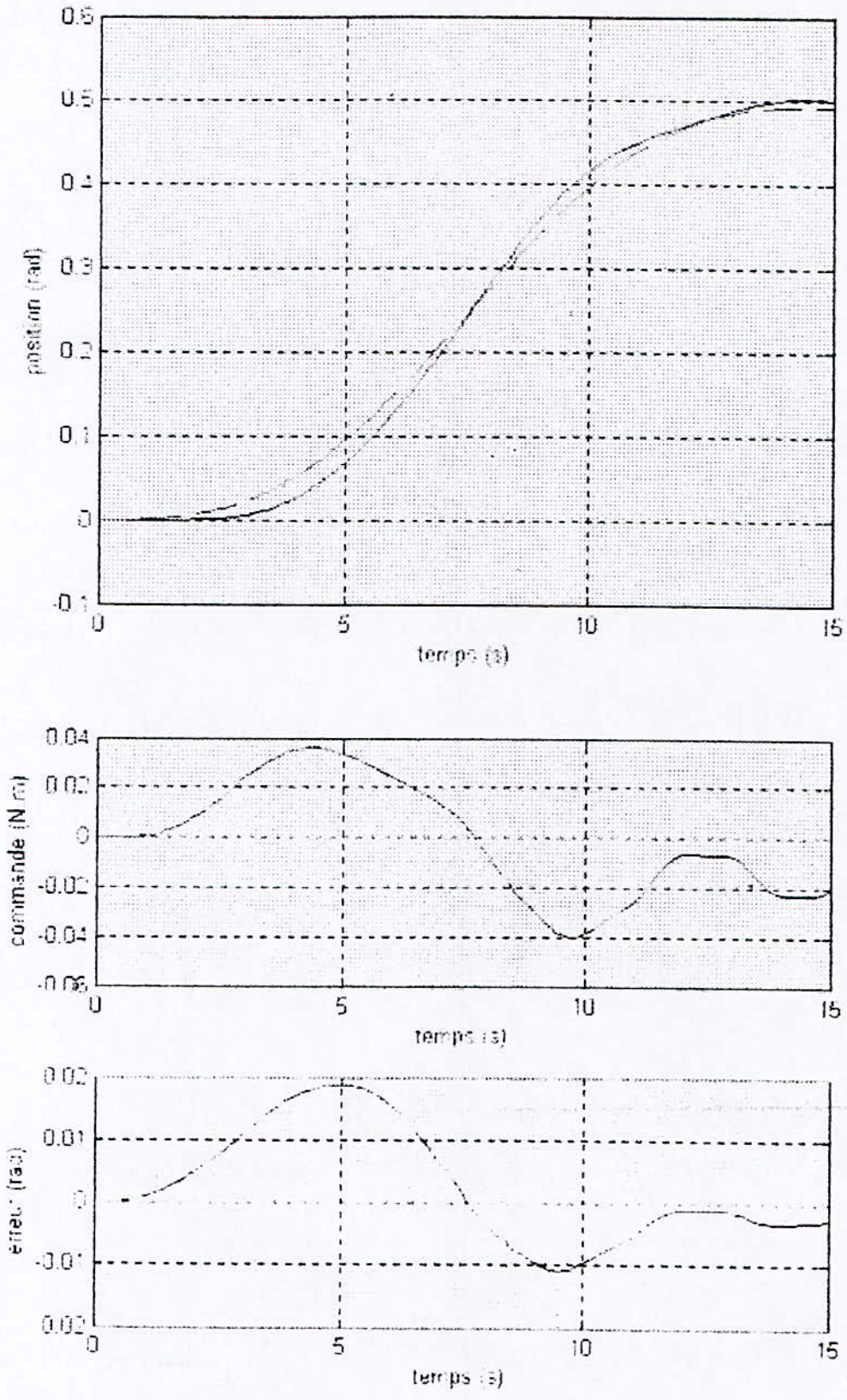


Fig (VL10) : ***réponse en position du bras flexible, commande et erreur correspondantes à la poursuite de la trajectoire2***

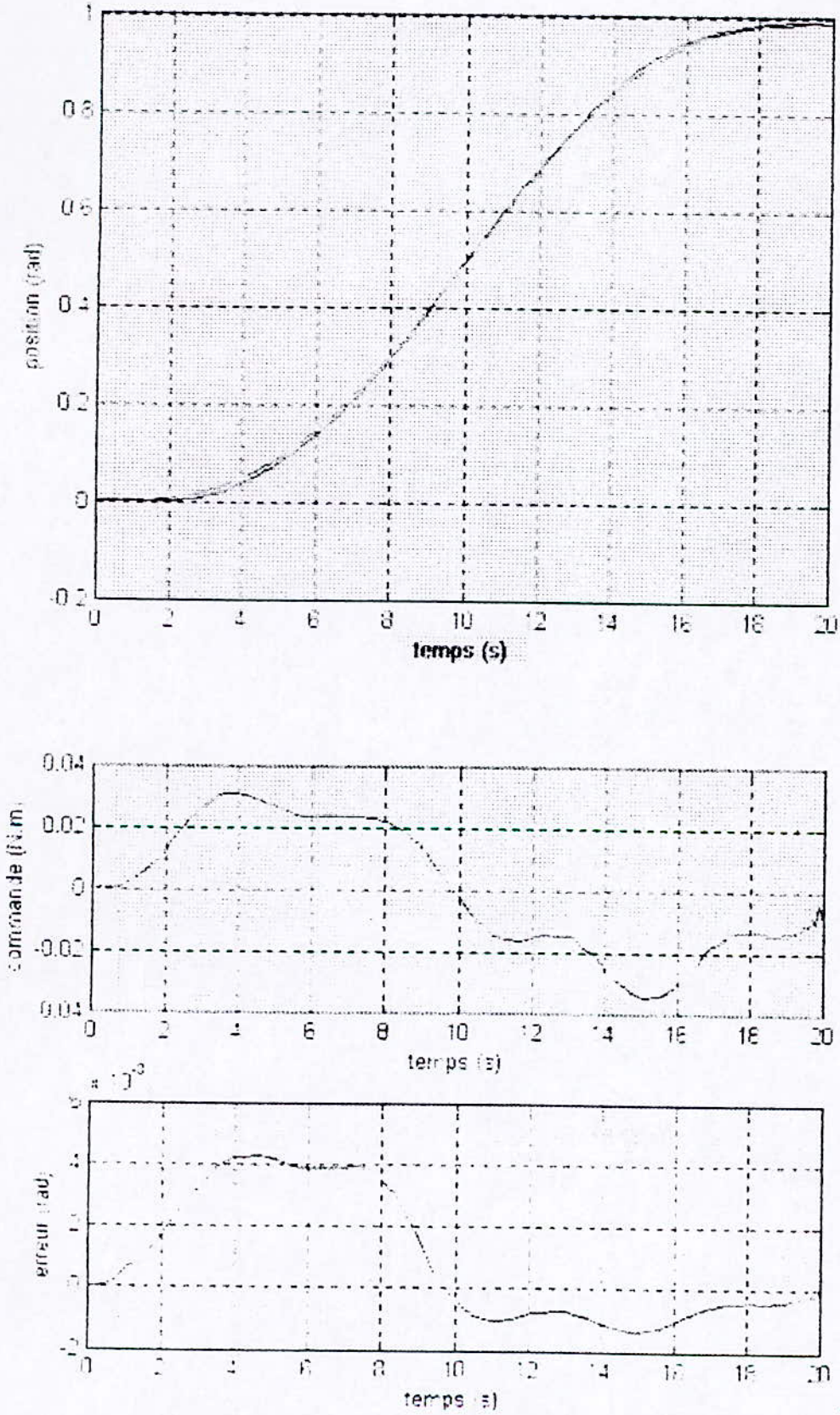
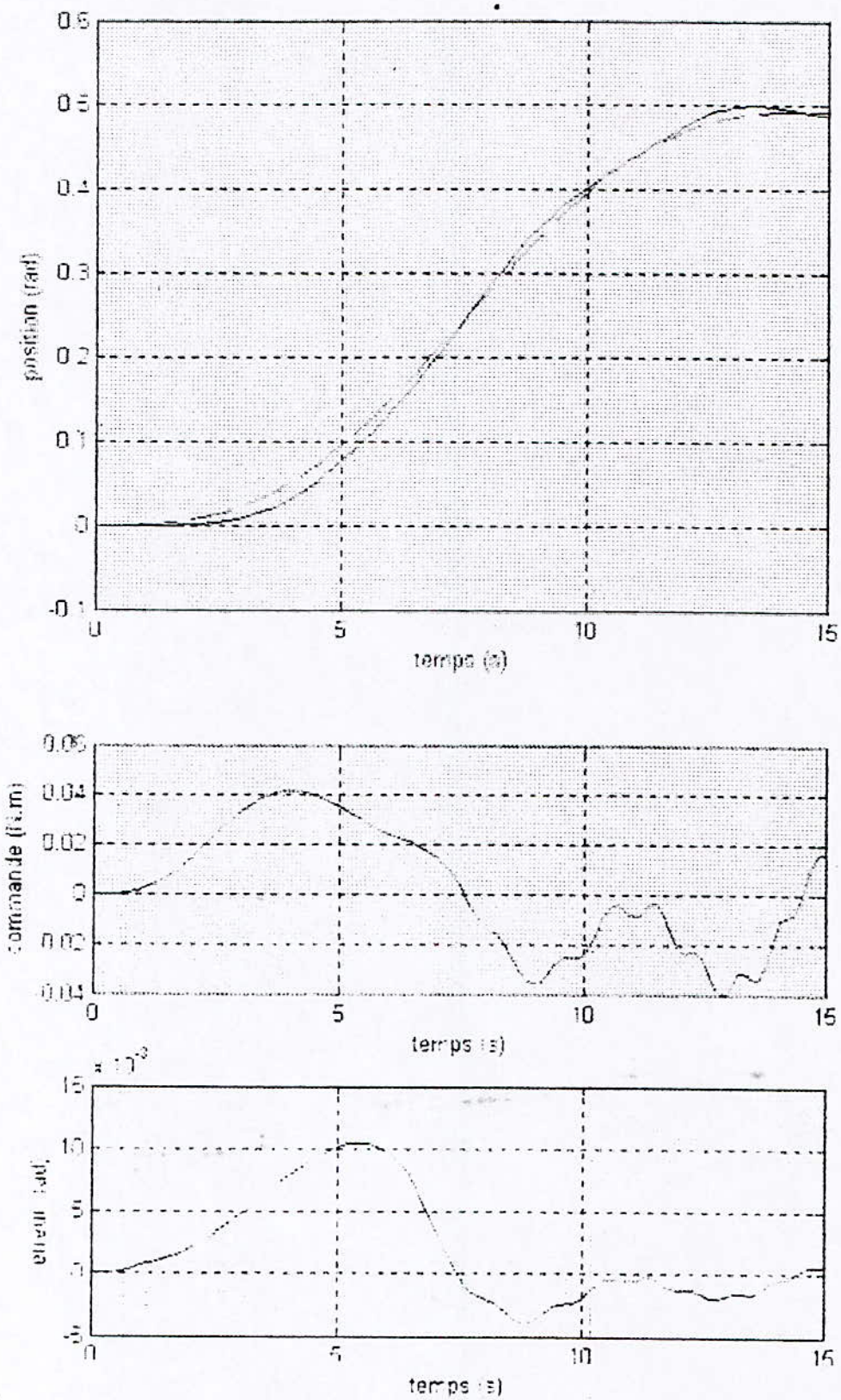


Fig (VI.11) : ***réponse en position du bras flexible, commande et erreur correspondantes à la poursuite de la trajectoire3***



Fig(VI.12) : ***Position, commande et erreur de poursuite de la trajectoire2 pour un lâché de masse à la 6ème seconde***

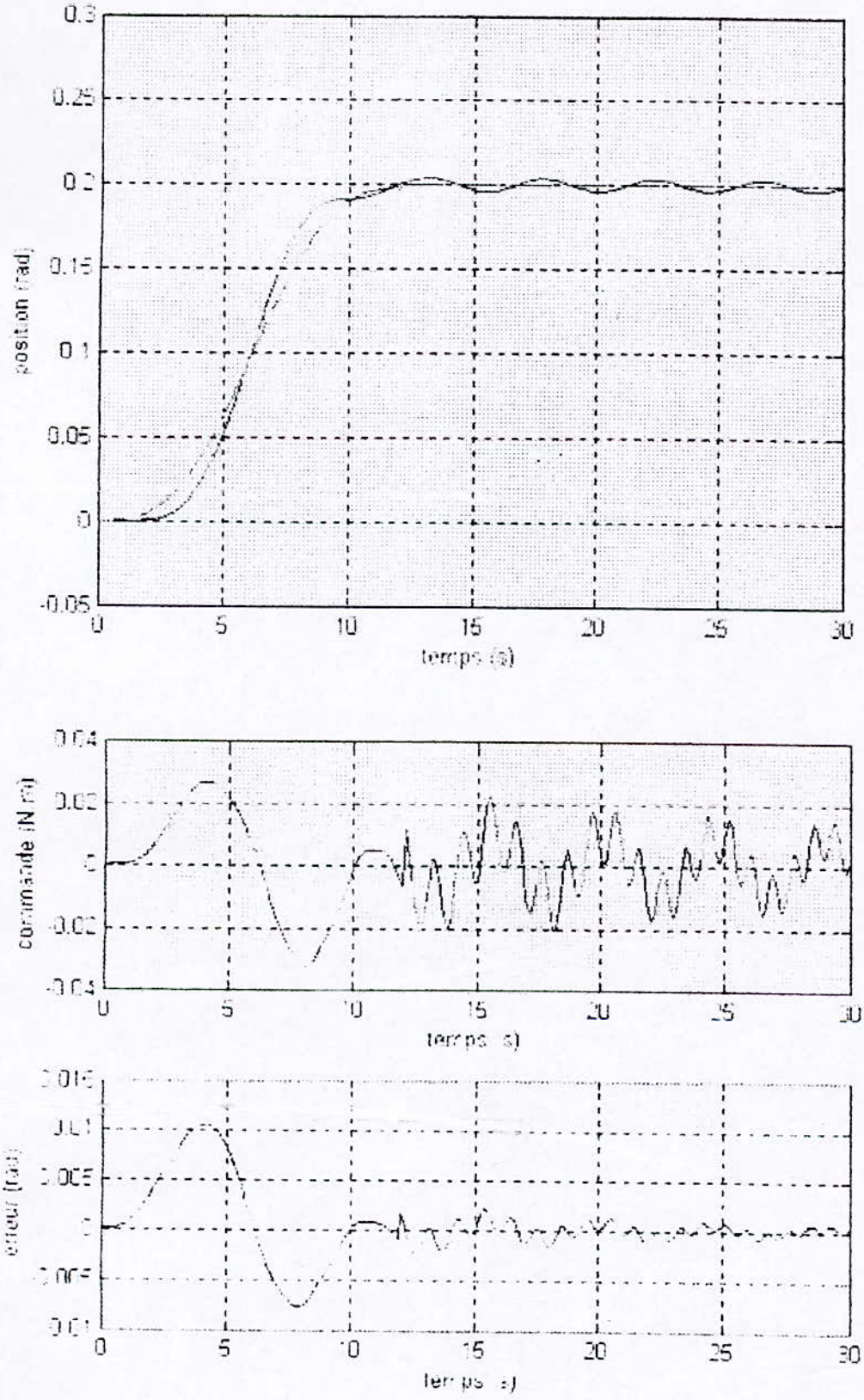


Fig (IV.13) réponse en position du bras flexible, commande et erreur correspondantes à la poursuite de la trajectoire4

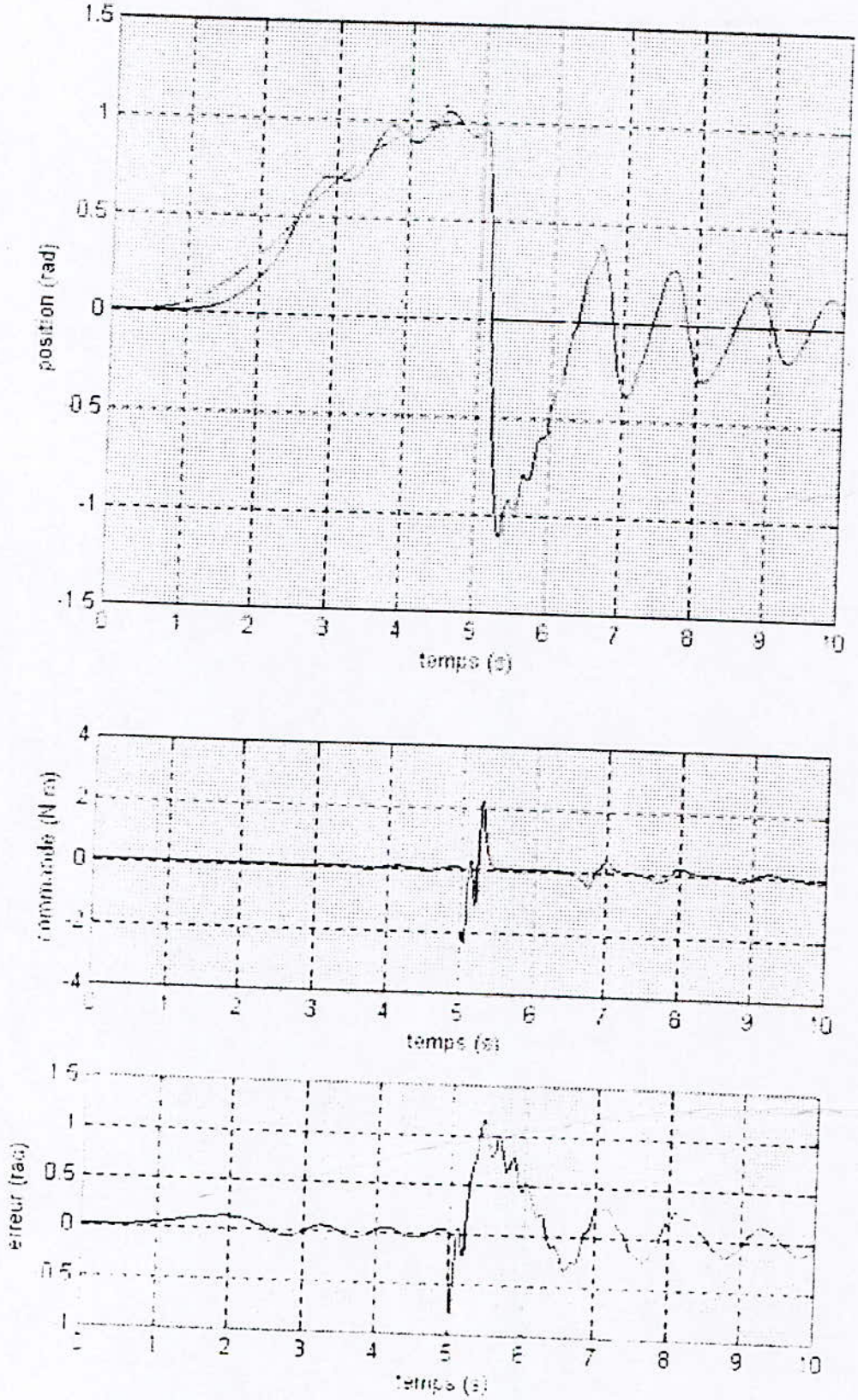


Fig (IV.14) réponse en position du bras flexible, commande et erreur correspondantes à la poursuite de la trajectoire5

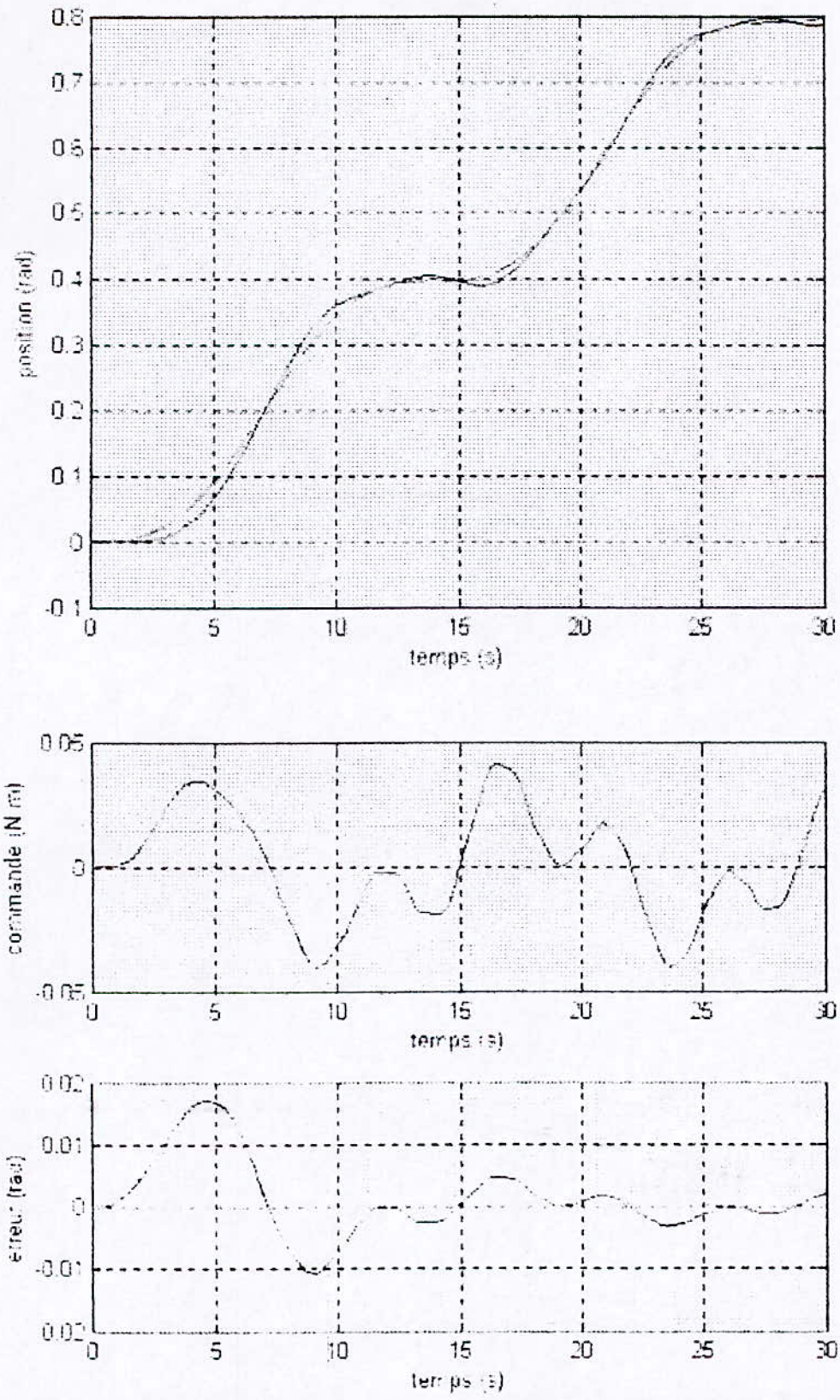


Fig (IV.15) réponse en position du bras flexible, commande et erreur correspondantes à la poursuite de la trajectoire6

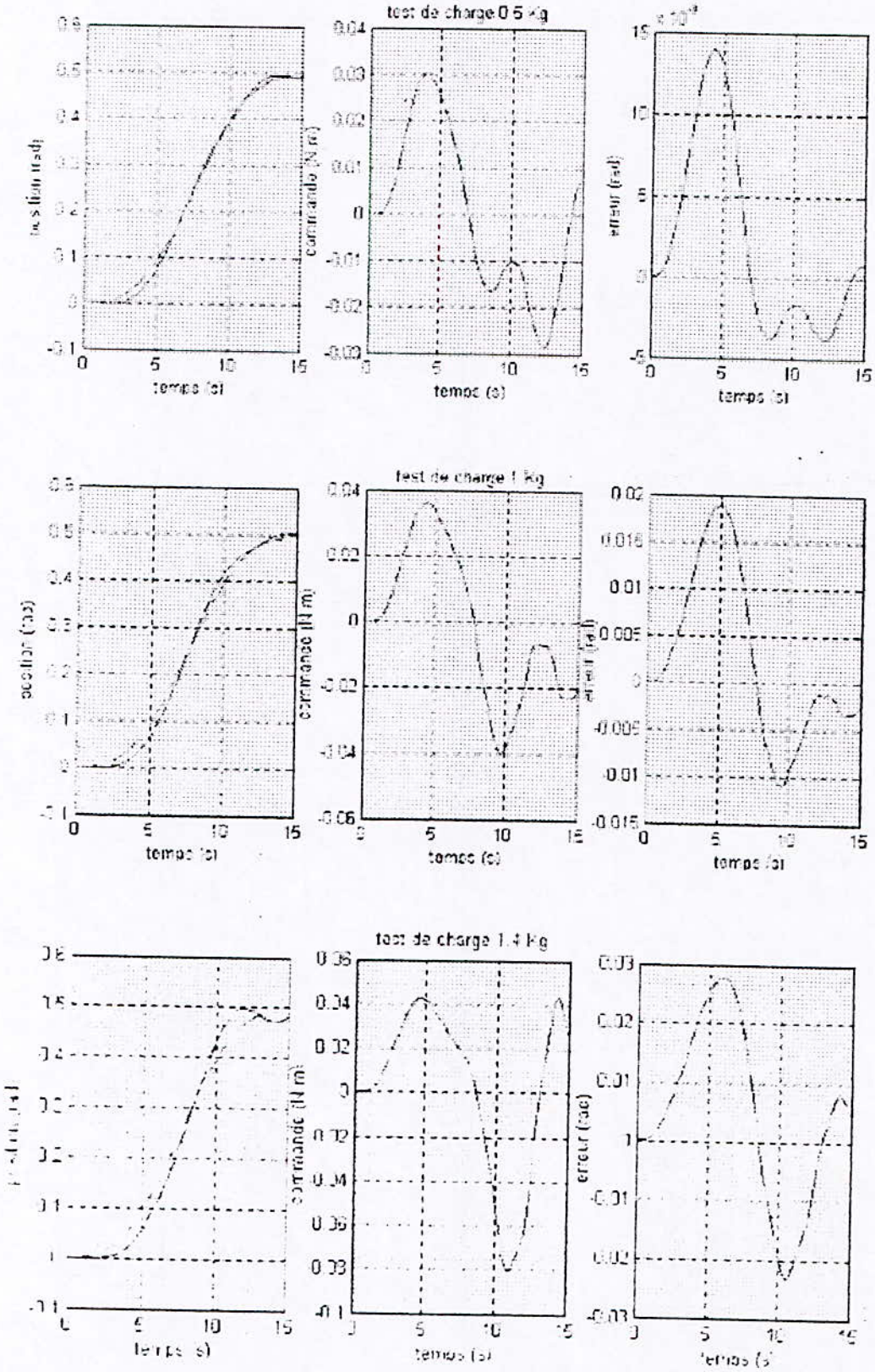


Fig (IV.16) réponse en position du bras flexible, commande et erreur correspondantes au test de charge

Conclusion

Ce chapitre a été consacré à l'élaboration d'un réseau neuro-flou en se basant sur l'inférence floue de Mamdani.

L'utilisation de l'algorithme de back propagation seul mène à la rencontre de minima locaux, ce qui arrête prématurément la recherche du minimum global et les méthodes d'amélioration de ce dernier étant des méthodes d'optimisation conventionnelle sont à utilité limitée.

De nouvelles classes de méthodes d'optimisation sont apparues, inspirées de la sélection et évolution biologique, et sont appliquées en commande.

L'architecture multicouche obtenue sera utilisée dans le chapitre suivant avec un algorithme d'apprentissage global, ayant vu le besoin d'améliorer l'algorithme d'apprentissage du réseau proposé.

5

Commande neuro-floue Hybride

Introduction

Nous avons constaté lors de notre étude dans le chapitre précédent, l'insuffisance de la commande neuro-floue à base d'un algorithme d'optimisation locale à la commande en position d'un bras flexible voyant les performances atteintes. Pour cela faire appel à d'autres algorithmes et méthodes semble nécessaire.

Le présent chapitre est consacré à la mise en œuvre d'une commande neuro-floue en y ajoutant une méthode d'optimisation globale incluant en particulier les algorithmes évolutionnistes. L'algorithme utilisé est sous le nom de l'algorithme d'apprentissage hybride établit en trois phases d'apprentissage.

Les AG utilisés ici ne sont pas statiques, mais plutôt dynamiques. Quelques uns de leurs paramètres changent avec le changement de la configuration du problème. Il s'agit du MRD-GA (multiresolutional dynamic genetic algorithm) *utilisé pour optimiser les paramètres des fonctions d'appartenances en ligne.*

V.1. Algorithme d'apprentissage hybride [FAR98,ZHO00]

Cet algorithme est composé de trois phases d'apprentissage :

1. phase d'initialisation des fonctions d'appartenance
2. phase d'élaboration des règles floues
3. phase d'optimisation des paramètres des fonctions d'appartenance.

Dans la première et deuxième phase, les algorithmes utilisés sont non supervisés. L'algorithme est supervisé dans la troisième phase.

V.1.1. Première étape d'apprentissage :

Les valeurs des centres (ou moyennes) et des largeurs (ou variance) initiales des fonctions sont déterminés par une technique d'apprentissage auto- organisatrice dite de KOHONEN.

V.1.1.1. La Carte Auto-Organisatrice

V.1.1.1.a. Architecture du Réseau

La carte auto-organisatrice est un ensemble structuré d'unités de traitement (neurones), qui sont disposées en une seule couche de laquelle émane une topologie définie par une notion de voisinage des cellules. En général c'est un carrée.

V.1.1.1.b Phase d'Apprentissage

Tout réseau de neurones requiert une phase d'apprentissage, laquelle a pour but d'adapter les valeurs des poids synaptiques.
Le modèle d'entraînement de la carte consistera donc à :

- sélectionner le neurone de sortie à réponse minimale correspondant le mieux à un signal d'entrée donné;
- induire dans un voisinage de noeuds une région d'intense activité.

V.1.1.1.c. Règles d'apprentissage

L'apprentissage est *non-supervisé* et consiste à répéter les étapes suivantes:

1. Initialiser aléatoirement les poids, et présenter un vecteur x à l'entrée du réseau
2. rechercher le neurone q^* dont le vecteur de poids est le plus poche, au sens de la distance euclidienne, du vecteur d'entrée :

$$q^* : \min_q \tfrac{1}{2}(x - W_q)^T (x - W_q) \qquad \text{(V.1)}$$

3. adapter le vecteur de poids de ce neurone ainsi que ceux de ses voisins de manière à ce qu'ils se rapprochent davantage du vecteur d'entrée :

$$W_q(\tau+1) = W_q(\tau) + \alpha(x - W_q(\tau)) \quad \text{si } q \in V(q^*) \qquad \text{(V.2)}$$
$$W_q(\tau+1) = W_q(\tau) \quad \text{sinon}$$

où V(q*) désigne le voisinage du neurone q^* dans lequel les vecteurs de poids sont adaptés, et α, $(\alpha > 0)$, est le taux d'apprentissage décroissant.

Le terme quadratique de la formule (V.1) peut être omis car il est constant pour tous les neurones. La recherche de celui dont le vecteur de poids synaptiques est le plus proche du vecteur d'entrée s'effectue alors selon:

$$q^* : \max_q \left(W_q^T x - \tfrac{1}{2} W_q^T W_q \right) \qquad \text{(V.3)}$$

Remarque :
Pour la convergence de l'algorithme de Kohonen, il faut conditionner α , on peut la prendre égale à $\frac{1}{t}$. C'est une bonne approximation.

V.1.1.2. application de l'algorithme de Kohonen au calcul des moyennes et de la variance des fonctions d'appartenance

- $\|x(t) - M_v(t)\| = \min_{1 \le i \le k} \{\|x(t) - m_i(t)\|\}$ (V.4)
- $m_v(t+1) = m_v(t) + \alpha(t)[x(t) - m_v(t)]$ (V.5)
- $m_i(t+1) = m_i(t)$ pour $m_i \neq m_v$ (V.6)
- $k = |V(x)|$, $V(x)$ voisinage de x (V.7)

Une fois les moyennes des fonctions calculées, leur largeur est déterminée en optimisant E donné par

$$E = \tfrac{1}{2} \sum_{i=1}^{N} \left[\sum_{j \in V_i} \left(\frac{m_i - m_j}{\sigma_i} \right)^2 - r \right]^2 \qquad \text{(V.8)}$$

où r est un paramètre à valeur dans [1,2]

Quand la troisième phase optimises les paramètres des fonctions d'appartenance, la largeur peut être simplement déterminée par

$$\sigma_i = \frac{|m_i - m_v|}{r} \qquad \text{(V.9)}$$

V.1.2. Deuxième phase d'apprentissage :

Pour la règle O_{ki}^3 de la couche 3, et la sortie des nœuds de la quatrième couche O_{ki}^4, nous voulons décider parmi les n_3 conséquences obtenues, celles considérées correctes afin de trouver les $n_1 \times n_2$ règles.

Un nouvel algorithme a été mis en œuvre. Le MMFA (maximum matching factor algorithm). Il est décrit en trois étapes:

Première étape :

Pour chaque règle d'un noeud de la troisième couche, on construit un facteur d'égalité, égale à n_3.

Dans ce cas, nous avons $n_1 \times n_2 \times n_3$ facteurs. Chaque facteur mis à égalité est noté M_{ij} où i est l'index du nœud de la règle $i = \overline{1, n_1 \times n_2}$ et j est l'index de la variable linguistique de sortie $j = \overline{1, n_3}$.

Deuxième étape :

M_{ij} est calculé selon le pseudo code suivant :

pour $i = \overline{1, n_1 \times n_2}$

  pour $j = \overline{1, n_3}$

    pour $k = 1, N$ *(nombre dexemple)*

$$M_{ij} = \begin{cases} M_{ij} + O_{ki}^3 & si \quad O_{ki}^4 \text{ , est le maximum parmi les } O_k^4 \\ M_{ij} & \text{sinon} \end{cases}$$

    fin

  fin

fin

Troisième étape :

Après avoir calculer les M_{ij}, les conséquences peuvent être déterminées à partir des facteurs utilisés selon le pseudo code suivant :

Pour $i = \overline{1, n_1 \times n_2}$

- *trouver le maximum des facteurs* $M_{\max}$ *parmi la série* $M_i \left(M_{ij}, i = \overline{1, n_3} \right)$
- *trouver le terme correspondant à l'index* $j_{\max}$ *de* $M_{\max}$
- *Eliminer tous les liens dans la quatrième couche en relation avec les nœuds de la troisième couche, excepté celui connecté au nœud indexé* $j_{\max}$

Fin

V.1.3. Troisième phase d'apprentissage :

Le réseau neuronal est établi, une fois les règles floues trouvées. Cette troisième phase d'apprentissage est considérée dans le but ajuster les paramètres des fonctions d'appartenance tout en les optimisant.

En général, optimiser revient à trouver une solution optimale parmi une famille de solutions acceptable selon un critère donné. Trouver l'optimum global n'est pas toujours garanti.
Une nouvelle approche supervisée est proposée dans cette phase pour l'optimisation des paramètres des fonctions d'appartenance en garantissant de trouver l'optimum global, et cela en utilisant les algorithmes génétiques.

Le MRD-GA (multiresolutional dynamic genetic algorithm) change son espace de recherche de l'optimum avec le changement de la configuration du problème et le modèle flou est dynamiquement adapté.

Nous allons consacrer une partie de cette phase aux algorithmes génétiques afin de bien les assimiler et comprendre leur utilisation et apport par rapport à l'optimisation.

V.1.3.1. Optimisation par des algorithmes génétiques

Introduction

L'évolution biologique a engendré des systèmes vivants autonomes extrêmement complexes qui peuvent s'adapter continuellement à un environnement complexe, incertain et en constante transformation.

Les différentes situations auxquelles les êtres vivants se sont adaptés laissent penser que le processus de l'évolution est capable de résoudre de nombreuses classes de problèmes. Il se caractérise par sa robustesse.

L'évolution est à base d'un mécanisme qui repose essentiellement sur la sélection des individus les plus adaptés à leur milieu, en leur assurant une descendance et une reproduction.

Les possibilités espérées de ces mécanismes ont conduit quelques chercheurs des années *50* à vouloir les simuler pour les appliquer à l'ingénierie. Des contraintes d'insuffisance de connaissances de la génétique et pour des raisons de faibles performances des calculateurs, ont bloqué l'avancement de ces recherches.

A partir des années *60*, le monde a été révolutionné par de nouvelles méthodes de calcul, développées avec l'apparition de l'ADN et des théories de génétiques.

Ainsi, naissent les calculs évolutionnaires, qui représentent les méthodes numériques, basées sur des « populations » de points pour résoudre des problèmes complexes et qui sont classés d'après [RAD05] comme suit :

- Les algorithmes génétiques (AG) développés par J. Holland en *197*
- La stratégie d'évolution (ES) développée principalement par H. P. Schwefel en *1981* et utilisée en optimisation par T. Bach en *1995*
- La programmation évolutionnaire (EP) de L. J. Fogel en *1962*
- La programmation génétique de J. Kozd en *1994*
- L'optimisation statique de Baluja en*1994* et Muhlenbein en 1999

Dans ce qui suit, nous allons présenter en premier les fondement des AG et en second les paramètres de tels algorithmes. En final, sera donné une description des étapes d'un algorithme génétique.

V.1.3.1.1. Fondements d'un algorithme génétique

Les AG sont basés sur une succession d'opération, mais ne sont pas déterministes. Un problème pouvant âtre résolu à l'aide d'un AG possède un ensemble de solution, la solution n'étant pas unique.

Le rôle des algorithmes génétiques est la sélection des solutions les plus optimales afin de former un ou plusieurs sous-ensembles [DAB05]

V.1.3.1.2. codage des variables

La première étape consiste à définir et coder convenablement le problème (dispositif). A chaque variable d'optimisation x_i correspond un ***gène.*** Un ***chromosome*** est un ensemble de gènes. Chaque gène prend un certain nombre de valeur dénommées ***allèles***.

Chaque dispositif est représenté par un ***individu*** doté d'un ***génotype*** constitué d'un ou plusieurs chromosomes. Ce génotype est exprimé par la fonction dévaluation pour obtenir son ***phénotype***. Celui-ci indique l'aptitude de l'individu à subsister dans son environnement (à se reproduire). Une ***population*** est un ensemble de N individus.

D'un point de vu informatique, différentes méthodes de codage existent, en l'occurrence le codage binaire. Dans ce cas, un gène est un entier long (32 bits), un chromosome est un tableau de gènes, un individu est un tableau de chromosomes et la population est un tableau d'individu.

Un des avantages du codage binaire est la facilité de coder toutes sortes d'objets : des réels, des entiers, des valeurs booléennes, des chaînes de caractères... il suffit d'utiliser des fonctions de codage et décodage pour passe d'une représentation l'autre.

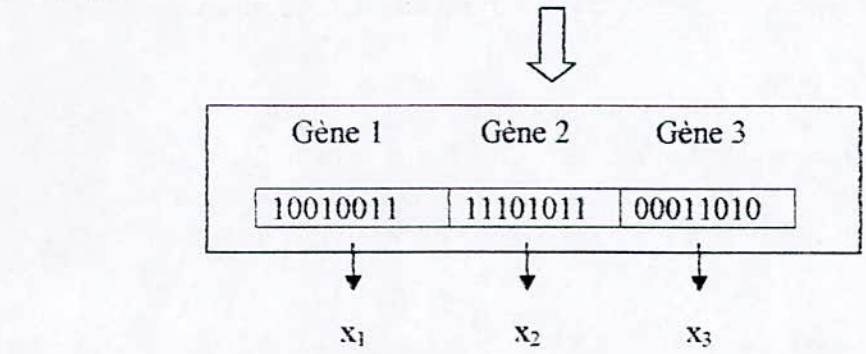


***Figure V.1* : illustration schématique du codage des variables d'optimisation**

V.1.3.1.3. Genèse de la population

La première étape de l'algorithme est la genèse de la population, c'est-à-dire le choix des individus de départ que nous allons faire évoluer. On pourrait prendre les individus régulièrement répartis dans l'espace. Néanmoins, une initialisation aléatoire est plus simple à réaliser.

V.1.3.1.4. Fonction d'évaluation

La fonction d'évaluation ou d'adaptation associe un coût à chaque individu. Elle prend en argument un individu et agit en deux temps :

- Décodage de l'individu (interprétation de la chaîne de bits dans le cas d'un codage binaire), cela peut être vu comme l'exhibition du phénotype de l'individu.
- Calcul de la valeur de ce phénotype comme une solution au problème fournissant la performance de l'individu

Ne connaissante pas la solution du problème, la fonction d'évaluation doit guider l'algorithme vers l'optimum en réalisant implicitement une pression de sélection dans cette direction.

V.1.3.2. Opérateurs génétiques

La reproduction chez les êtres vivants se fait par la recombinaison des gènes parentaux pour former des descendants aux potentialités nouvelles.

La simulation de ce processus pas les AG se fait pendant la phase de reproduction durant laquelle de nouvelles solutions sont générées à partir des solutions courantes. Ce sont les opérateurs génétiques qui définissent la manière dont les individus se recombinent et s'agencent pendant cette phase de reproduction.

Les opérateurs considérés dans les AG sont le ***crossover***, ***mutation***, et la ***sélection***.

V.1.3.2.a. *Le crossover (croissement)*

C'est un opérateur de reproduction qui prend deux individus et combine leurs gènes. Donc il agit sur des paires d'individus. Plusieurs variantes de cet opérateur existent. La plus utilisée consiste à couper en un ou plusieurs points deux individus (aux mêmes endroits) et à y échanger les gènes.

Cet opérateur permet de créer de nouvelles combinaisons des paramètres (allèles) des composants (gènes) et favorise l'exploitation de l'espace de recherche.

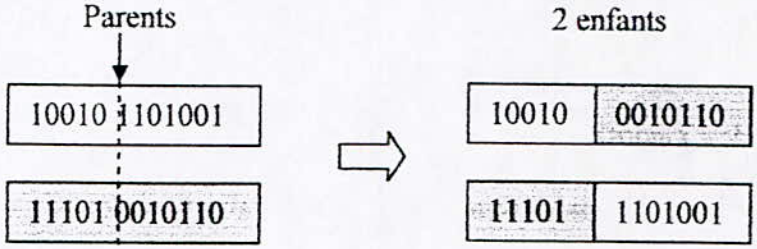


Figure V.2 **: représentation schématique du croissement**

V.1.3.2.b. *La mutation*

C'est l'inversion d'un allèle dans un chromosome (inversion d'un bit). Cela revient à modifier aléatoirement la valeur d'un paramètre du dispositif. Les mutations jouent le rôle d'un bruit et empêchent l'évolution de se figer. Elles permettent d'assurer une recherche aussi bien globale que locale selon le poids et le nombre des bits mutés.

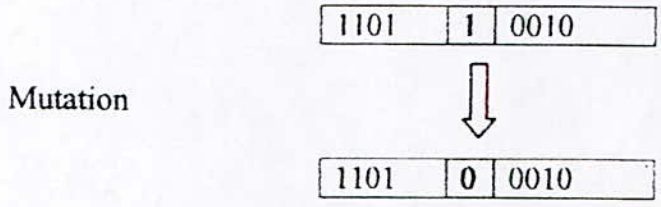


Figure V.3 **: représentation schématique d'une mutation**

V.1.3.2.c. *La sélection*

La phase de sélection est en général considérée comme un opérateur de sélection. Les individus les mieux adaptés tendent à fournir la descendance la plus nombreuse.

Aussi la phase de sélection spécifie les individus de la population qui doivent survivre et auxquels les opérateurs génétiques du crossover et de la mutation s'appliquent.

Lors de la sélection, une population temporaire est formée contenant des copies des génotypes des parents. Chaque parent contribue aléatoirement par un nombre de copies en accord avec sa performance ; les meilleurs individus tendent à donner un plus grand nombre de copies que les plus mauvais.

Plusieurs méthodes de sélection sont proposées pour les AG. La méthode la plus utilisée est celle de *la roue de loterie* (figure 4). Elle assure la proportionnalité entre le nombre de descendance d'un individu et sa performance en attribuant une section de surface à chaque individu proportionnelle à sa performance.

Par cette méthode, le processus de sélection naturelle est reproduit en faisant en sorte que seul les individus les mieux adaptés survivent et se reproduisent, ceci en ayant plus de chance de tomber sur une grande surface dans la roulette.

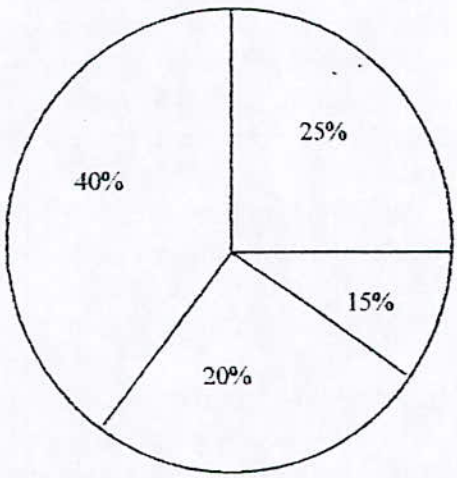


***Figure V.4* : la sélection par la roue de loterie**

V.1.3.3. Paramètre d'un algorithme génétique

Les algorithmes génétiques peuvent être réglés à l'aide de différents paramètres déterminés à l'avance et dont dépend fortement la convergence de l'algorithme.

V.1.3.3.a. La taille de la population

Les conditions de convergence changent si la taille de la population change. Si la taille de la population est petite, la probabilité de s'attarder sur des minima locaux est grande. En revanche, si la taille de la population est importante, la recherche s'effectue de façon redondante et l'efficacité de l'algorithme est globalement affectée.

V.1.3.3.b. Le taux de crossover

L'opérateur de crossover est appliqué avec une probabilité P_c. Plus cette valeur est grande, plus sont introduites de nouvelles structures dans la nouvelle génération.
Si ce taux est trop élevé, les structures performantes sont trop fréquemment détruites. Par contre, si il est trop faible, la population n'évolue pas assez vite.

V.1.3.3.c. Le taux de mutation

Comme pour l'opérateur du crossover, la mutation est appliquée avec une probabilité P_m. Plus cette probabilité est grande, plus grand est le risque de suivre une fausse piste car la recherche devient purement aléatoire. Et si ce taux est petit, la mutation est trop faible, la population est moins diversifiée et il y a risque d'une stagnation prématurée.

Le taux de mutation dans les algorithmes génétiques est généralement pris faible. Des études empiriques conseillent pour l'obtention de bons résultats une fréquence qui se situe autour d'une mutation tout les 1000 bits.

V.1.3.3.d. Le fossé des générations

Ce paramètre contrôle la portion de la population qui doit être remplacée à chaque génération. Si ce taux est de 1, l'ensemble de la population est remplacé. K. De Jong a introduit la notion d'écart entre les générations et a proposé un nombre entre 0 et 1 afin de ne pas remplacer tout un ensemble mais plutôt les parents qui doivent être remplacés par leurs enfants.

Du moment où l'on conserve certains individus par rapport à d'autre, la question du choix des individus à sauvegarder apparaît. Ce lui ci est généralement basé sur la performance des individus : plus l'individu est performant plus il a la chance de survivre. La sélection étant stochastique, le meilleur individu peut ne pas survivre d'une génération à une autre.

Dans l'art actuel de la théorie des AG, il semble que trouver des valeurs pour ses paramètres est encor plus un art qu'une science. Il n'y a aucun jeu de paramètres qui serait universel pour tous les problèmes considérés du fait que ces valeurs dépendent étroitement du type de problème à résoudre. Néanmoins, les valeurs suivantes obtenues expérimentalement peuvent être appliquées à une multitude de problèmes :

- Probabilité d'application du crossover : $P_c = 0.25$
- Probabilité d'application de la mutation : $P_m = 0.001$
- Population de taille relativement modérée entre 25 et 100 individus, selon la taille d'un individu.

V.1.3.4. Description des étapes d'un algorithme génétique

Un algorithme génétique part d'une population initiale d'individu. La population évolue alors en effectuant à chaque génération le cycle de la sélection/reproduction.
En voici un pseudo code du cycle :

Début
    $t = 0$
    Initialiser $P(t) = P(0)$
Evaluer $P(0)$
Tant que (condition non terminée) faire :
    $t = t + 1$, incrémenter la génération
    Sélectionner $P(t)$ parmi $P(t-1)$
    Recombiner $P(t)$, utilise l'opérateur génétique (crossover, mutation)
    Evaluer $P(t)$
    Fin tant que
Fin

Cet algorithme est un algorithme de base et est appelé « algorithme génétique simple ».

La sélection :

Le procédé de cette méthode consiste à dupliquer chaque individu de la population de manière proportionnelle à son adaptation :

- *Calculer l'adaptation* $Eval(V_i)$ *pour chaque chaine* V_i, $i = \overline{1,N}$ *(N étant la taille de la population)*
- *Calculer l'adaptation totale de la population* $Som = \sum_{i=1}^{N} Eval(V_i)$
- *Calculer la probabilité de sélection de chaque individu* $P_i = Eval(V_i) / Som(t), i = \overline{1,N}$
- *Calculer la probabilité cumulative* q_i *pour chaque individu* $q_i = \sum_{j=1}^{i} P_j$
- *La sélection d'un individu se fait en tournant la roue de loterie, et ceci en générant un nombre réel r compris entre 0 et 1 :*

Si $r < q_1$ *l'individu* v_1 *n'est pas sélectionné*
Si $q_{i-1} < r < q_i$ *l'individu* v_i *est sélectionné*

Cette opération se répète N fois pour sélectionner entièrement la population intermédiaire sur laquelle les opérateurs du crossover et de mutation s'appliquent.

Le crossover :

$N \times P_c$ individus subissent en moyenne un crossover selon le processus suivant :

Pour chaque individu de la nouvelle population
Générer un réel $r \in [0,1]$
si $r < P_c$ *alors l'individu est marqué pour un crossover*

Les individus sont pris deux à deux. Pour chaque paire, on applique le crossover en générant aléatoirement le(s) point(s) de coupure. Les deux descendants remplacent leurs parents.

La mutation :

Elle est appliquée sur chaque bit des chaînes binaires des chromosomes de la population avec une probabilité P_m. Le nombre de bits modifiés lors de l'application de l'opérateur de mutation est en moyenne égale à $N \times L \times P_m$ où L est la taille de l'individu et N celle de la population. Le processus se présente comme suit

Pour chaque individu de la population
Pour chaque bit de l'individu
Générer aléatoirement un réel $r \in [0,1]$
Si $r < P_m$ *alors inverser le bit*

V.1.3.5. troisième phase d'apprentissage : le MRD-GA

Le MRD-GA utilise les séries d'entiers décimaux afin de coder les paramètres du modèle flou. Cette représentation permet l'utilisation des séries de petite taille. Le nombre des allèles est déterminé par le nombre total de bases floues utilisées pour partitionner l'espace entrées/sorties des variables floues. Dans le cas de notre modèle, nous avons $n_1 + n_2 + n_3 = n_4$ fonctions d'appartenance. Chacune d'elle a deux paramètres, la moyenne et la variance, ce qui fait $n_4 \times 2$ paramètres à optimiser.
Donc la largeur du gène est de $n_4 \times 2$ allèles, et chaque allèle peut prendre des valeurs dans[1,2,...,9]. Afin de convertir la valeur de l'allèle à une valeur d'une moyenne ou d'une variance, la procédure suivante est à utiliser :

Etape 1

Les valeurs initiales de la moyenne et de la variance du contrôleur flou doivent être entrées dans le programme de l'algorithme génétique. Soient m_{i0} et σ_{i0} pour $i = \overline{1,9}$

Etape 2

Les nouveaux centres et largeurs sont calculés par :

$$m_i = m_{i0} + (s_i - 5) \cdot \delta_m \qquad *$$
$$\sigma_i = \sigma_{i0} + (s_{(i+n_4)} - 5) \cdot \delta_\sigma \qquad **$$

où s_i est la valeur du *ième* allèle dans la série. δ_m et δ_σ sont les offset du centre et la largeur respectivement. Il est recommandé pour des raisons de convergence de l'AG de prendre très petites ces dernières valeurs, autour de 0.001.

Etape 3

Si la valeur de l'allèle s_i de n'importe quel centre est égale à cinq alors il n'y aurait aucun changement dans la fonction d'appartenance. Et s'il est plus grand que cinq la valeur du centre augmentera sinon il diminuera.

Chaque chromosome a une fonction d'évaluation. Dans le cas du MRD-GA, nous considérons la moyenne de l'erreur quadratique (mean squared errir (MSE)), calculée par :

$$MSE = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

Où y_i est la valeur de la sortie du contrôleur, et $\hat{y}_i$ est la valeur de la sortie estimée du contrôleur flou.

Pour chaque génération τ , la valeur de l'offset diminue selon les équations suivantes :

$$\delta_m = \delta_m \times \theta_m, 0 < \theta_m < 1$$
$$\delta_\sigma = \delta_\sigma \times \theta_\sigma, 0 < \theta_\sigma < 1$$

où θ_m et θ_σ sont les facteurs de modification de la moyenne et de la variance respectivement. Elles peuvent être des fonctions exponentielles.

La condition d'arrêt du programme devrait évidemment être le nombre de génération à exécuter

Le pseudo code de cet algorithme est donné ci-dessous :

Début
Initialiser $P(t) = P(0)$
Initialiser la fonction d'évaluation $F(t) = 0$
Evaluer $P(0)$
Chercher $F(t)$ *de* $P(0)$ *puis assigner* $F(t)$ *à* $P(0)$
Tan que (condition non fin pas terminée) faire
Début
$t = t+1$ *incrémentation de la génération*
Si $\mathrm{mod}(t/\tau) = 0$ *alors modifier (diminuer)* δ_m *et* δ_σ
Sélectionner $P(t)$ *parmi* $p(t-1)$ *utilisant 'la roue de lotterie'*
Recombiner $P(t)$ *utilisant le crossover et la mutation*
Evaluer $P(t)$
Chercher la fonction d'évaluation de $P(t)$ *et comparer avec* $F(t)$,
si plus grand faire
Début
Assigner la nouvelle fonction d'évaluation à $F(t)$
Adapter le centre et la moyenne m_{t0} *et* σ_{t0} *selon * et ***
Fin
Fin
Fin

Le MRD-GA a plusieurs avantages en dépit des AGS. Cet algorithme permet d'augmenter la dynamique dans la résolution de l'espace recherche et cela en diminuant δ_m et δ_σ au fur et à mesure que les paramètres du modèle approchent leurs valeurs optimales. Il est dynamique par le fait qu'il adapte la moyenne et la variance continuellement ce qui augment la chance des AG à converger.

V.2. test et analyse des performances

Les résultats obtenus par cette commande sont meilleurs que ceux de la commande précédente par rapport au temps d'exécution du processus d'apprentissage et les performances atteintes.
Nous nous somme arrêté à 18 générations de l'algorithme génétique pour les paramètres suivants : Pm= 0.3; Pc= 0.99; $\delta_m = 0.5$; $\delta_\sigma = 0.9$; $\theta_m = 0.9$; $\theta_\sigma = 0.98$.

Voici la structure du contrôleur considéré :

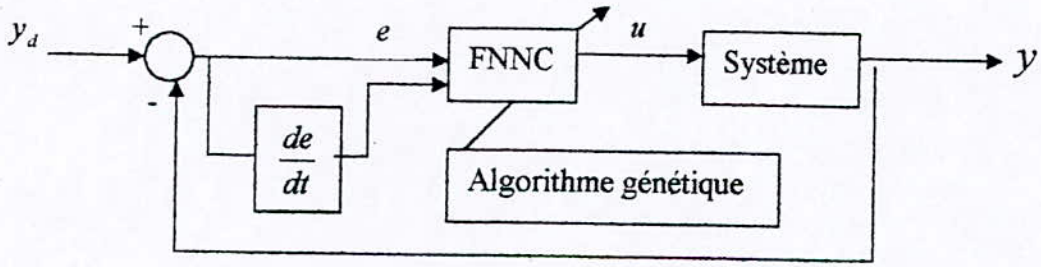


***Figure (V.9)* : Contrôleur neuro-flou à base d'AG**

Les trajectoires sont les mêmes que pour le chapitre précédent.
Les résultats de poursuite de trajectoires obtenus sont présentés sur les ***figures (V. 10, 11, 12, 14 , 15, 16).*** L'erreur est pratiquement nulle et est de l'ordre de 10^{-6}.
Pour ce qui est du lâché de masse, le résultat est donné par la ***figure (V.13).***
Le test de charge effectué sur le bras pour la poursuite de la trajectoire3 est donné par la ***figure (V.17).***
Nous avons atteint les performances souhaitées, pour uniquement 1 itération de la boucle d'apprentissage par rapport à 5 obtenues par la commande précédente.
Les résultats obtenus nous permettent donc de conclure que cette commande est robuste.

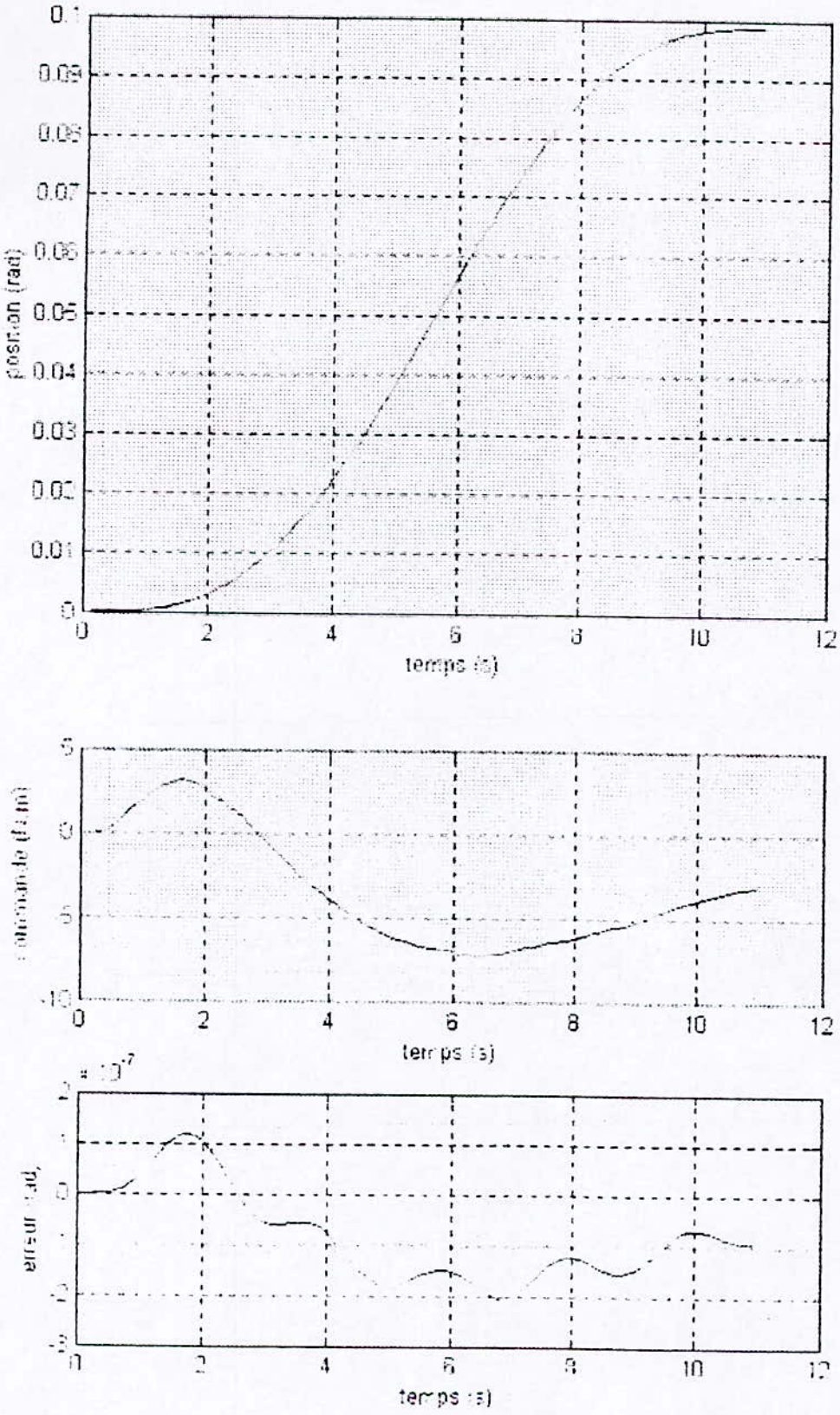


Fig (V.10) : ***réponse en position du bras flexible, commande et erreur correspondantes à la poursuite de la trajectoire1***

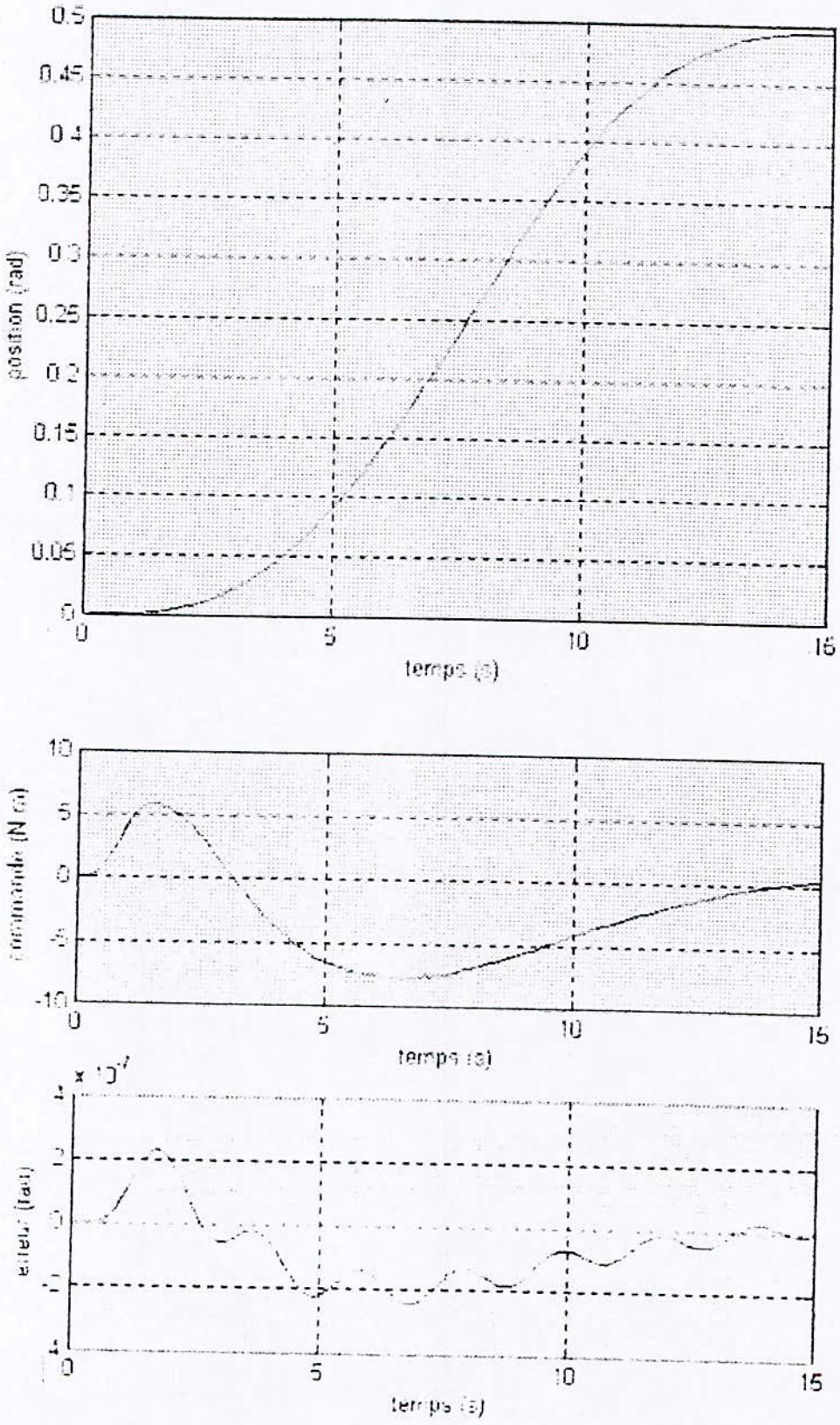


Fig (V.11) : ***réponse en position du bras flexible, commande et erreur correspondantes à la poursuite de la trajectoire2***

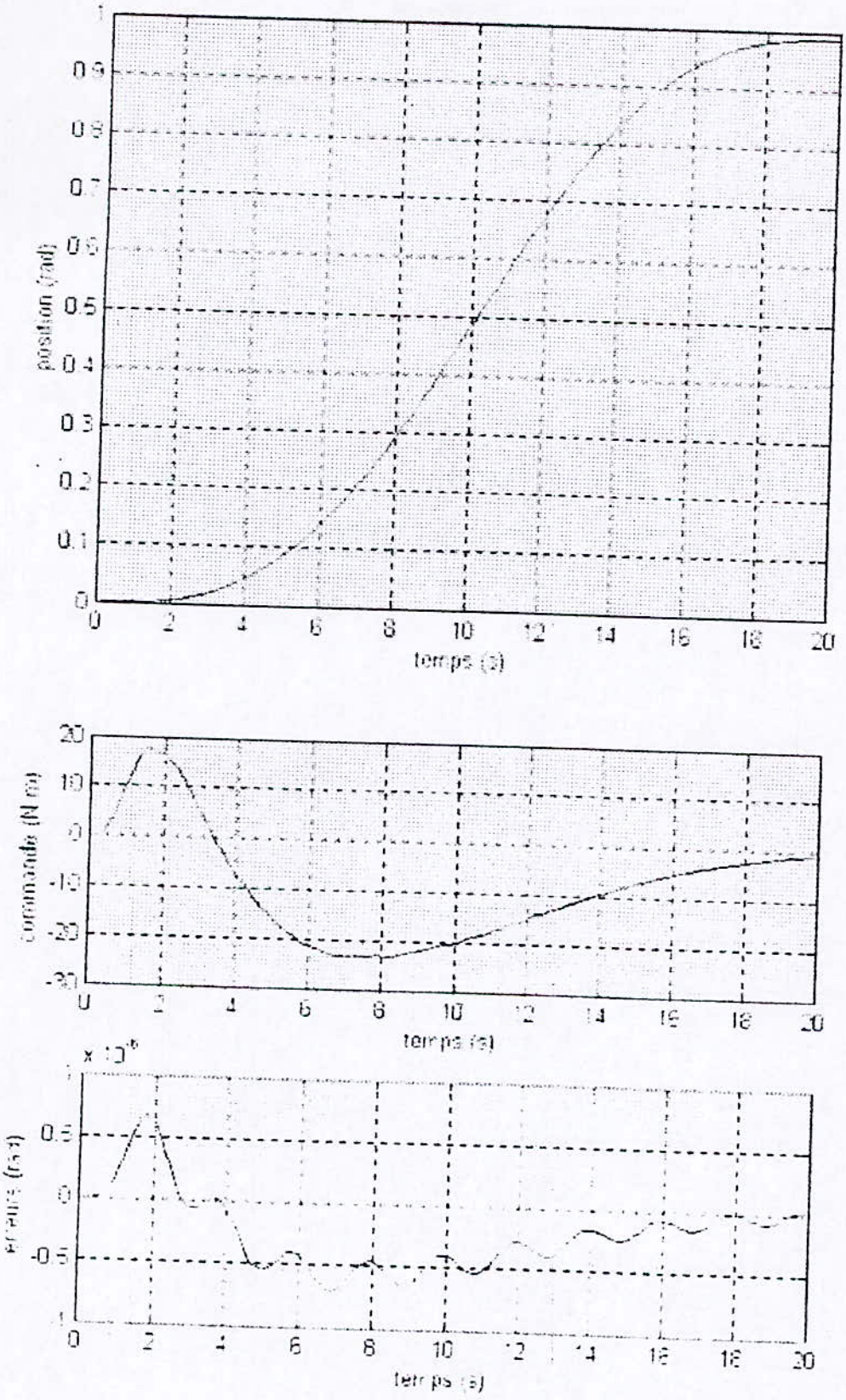
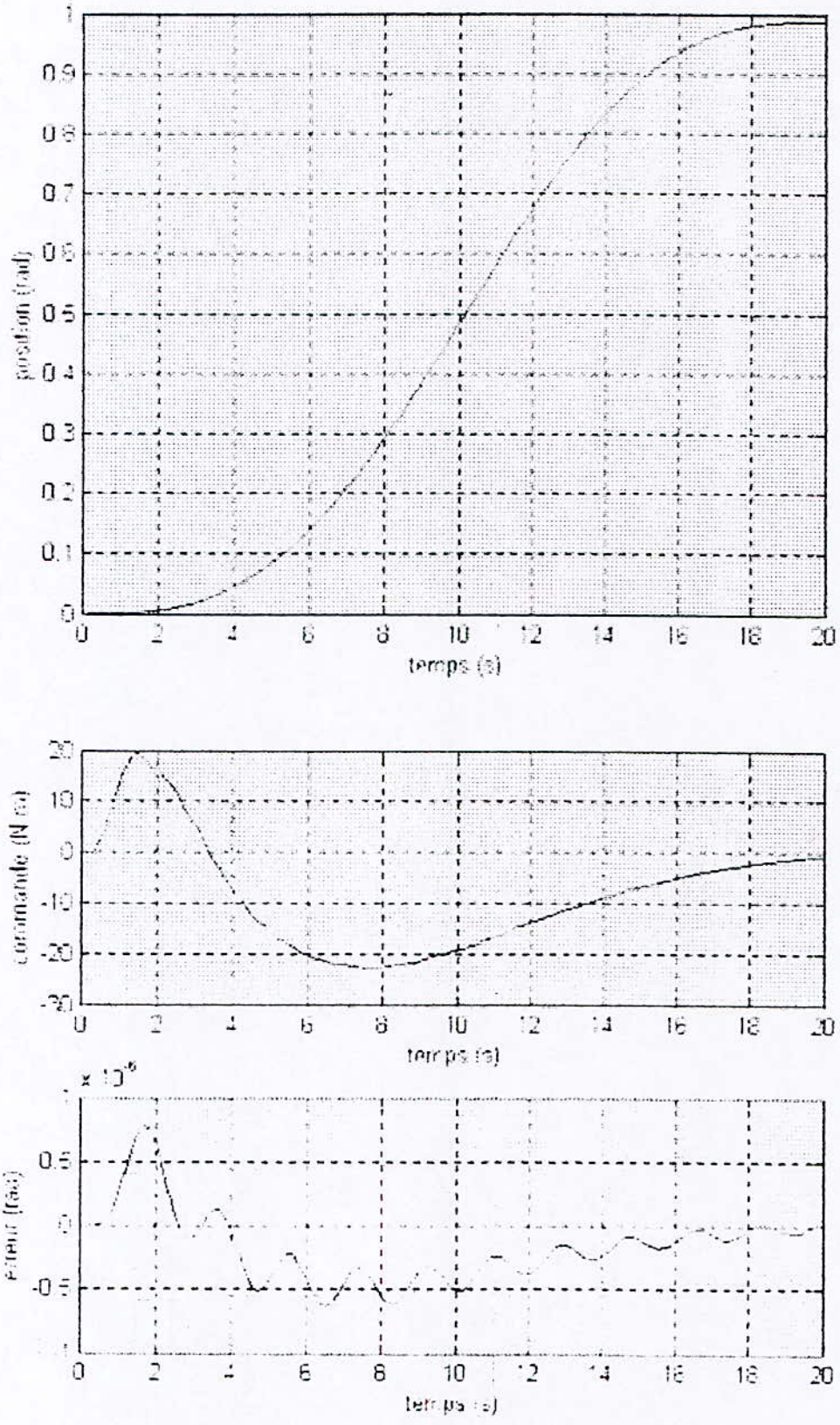


Fig (V.12) : ***réponse en position du bras flexible, commande et erreur correspondantes à la poursuite de la trajectoire3***



Fig(V.13) : ***Position, commande et erreur de poursuite de la trajectoire3 pour un lâché de masse à la 8ème seconde***

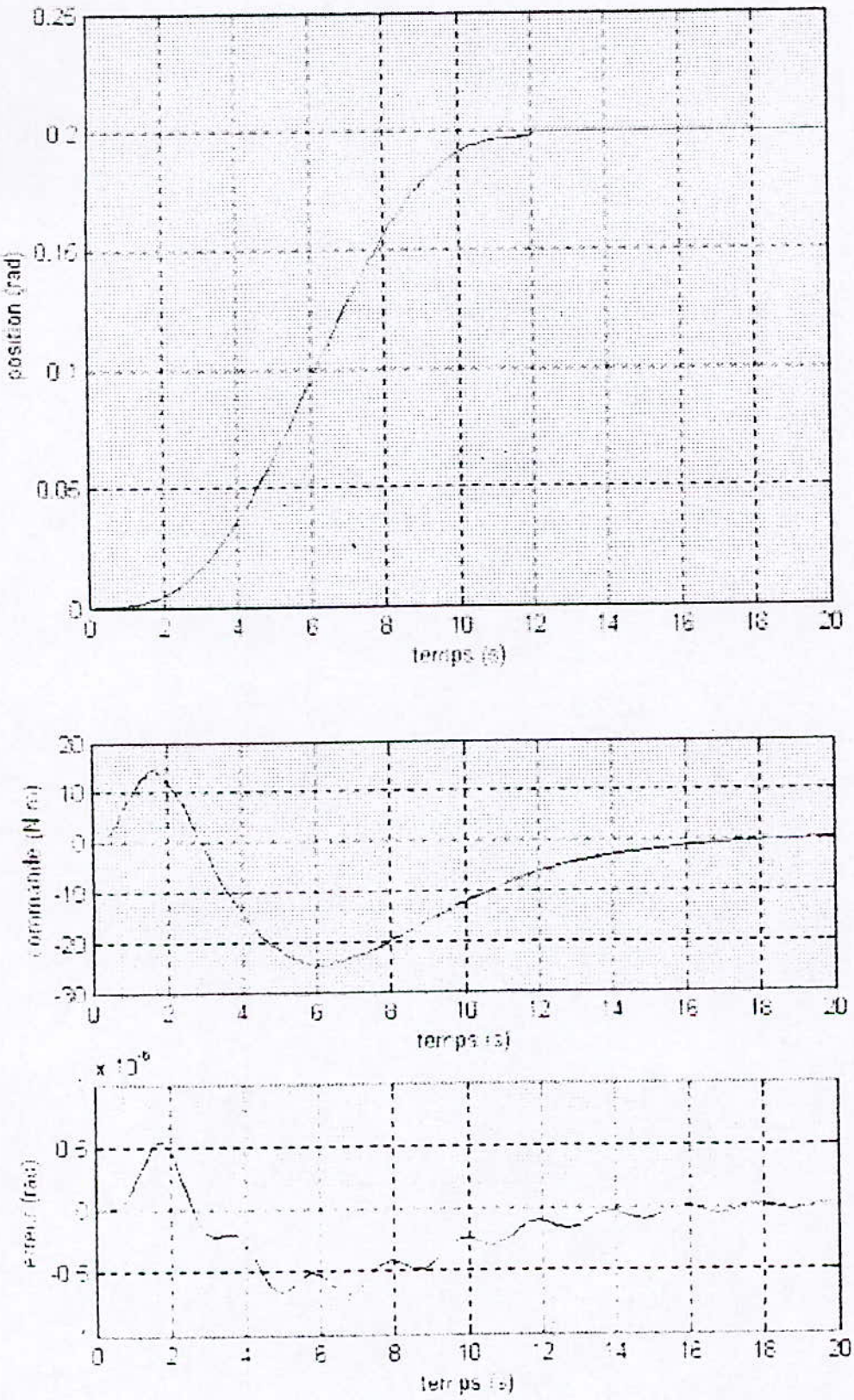


Fig (V.14) : ***réponse en position du bras flexible, commande et erreur correspondantes à la poursuite de la trajectoire4***

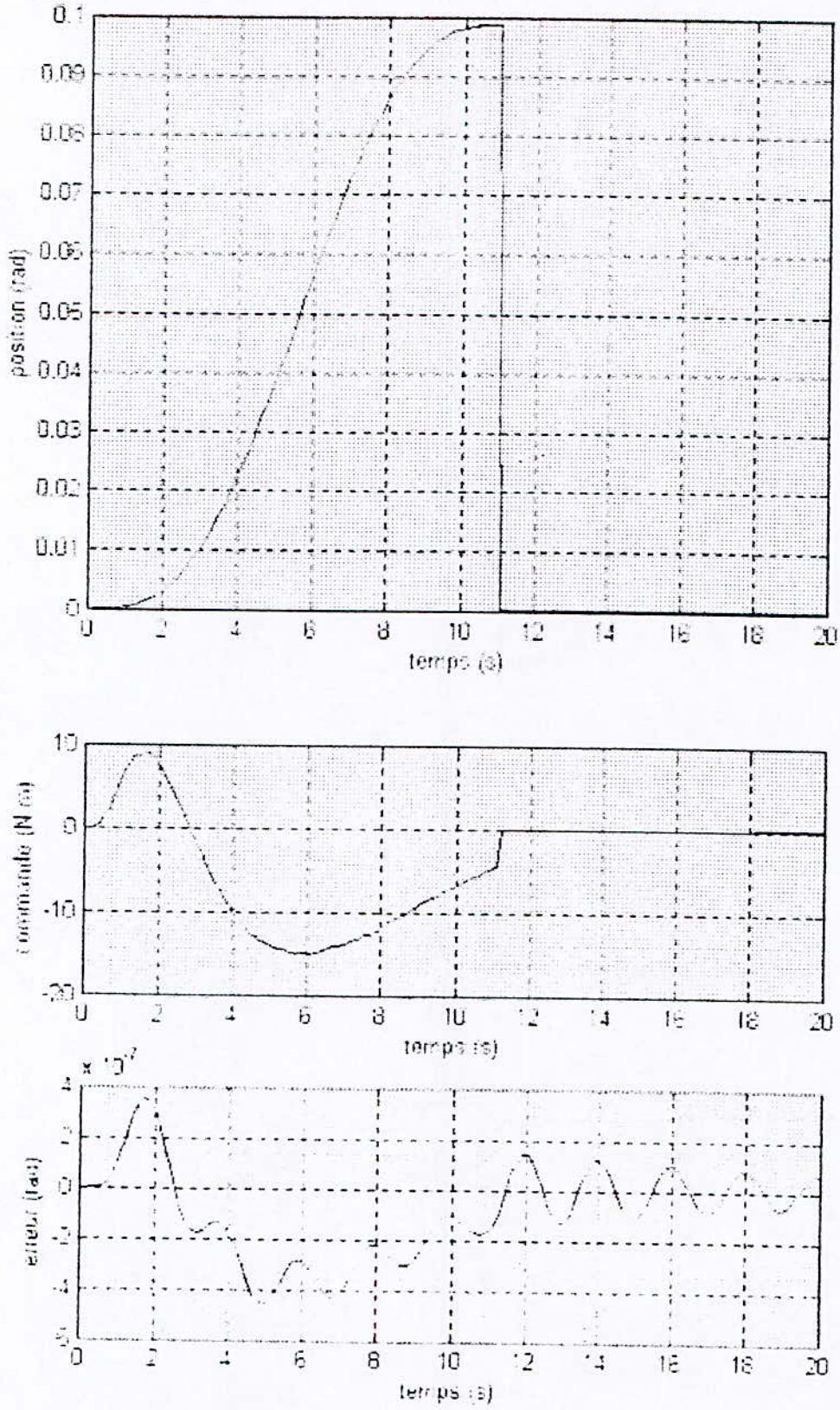


Fig (V.15) : ***réponse en position du bras flexible, commande et erreur correspondantes à la poursuite de la trajectoire5***

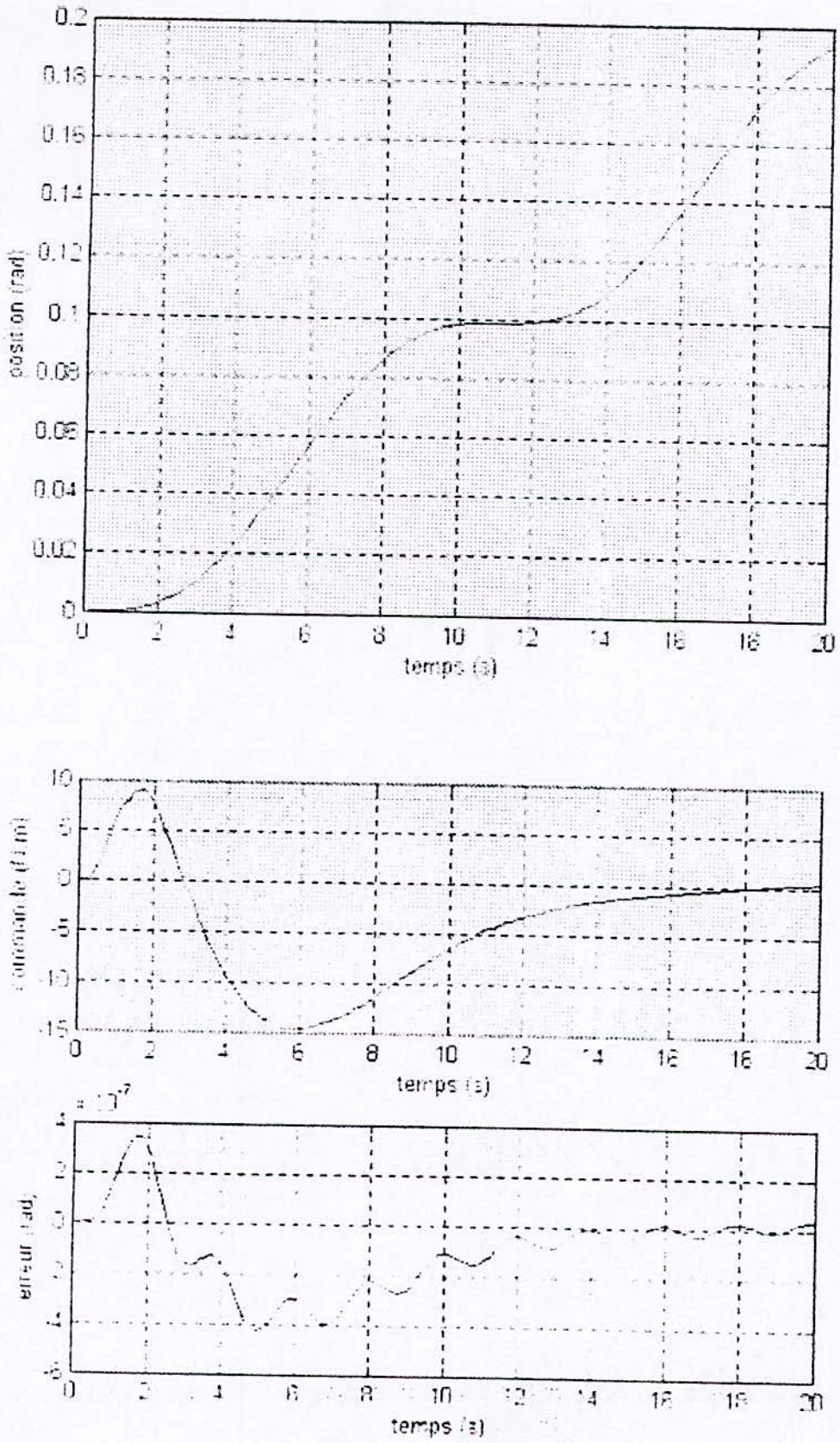


Fig (V.16) : ***réponse en position du bras flexible, commande et erreur correspondantes à la poursuite de la trajectoire6***

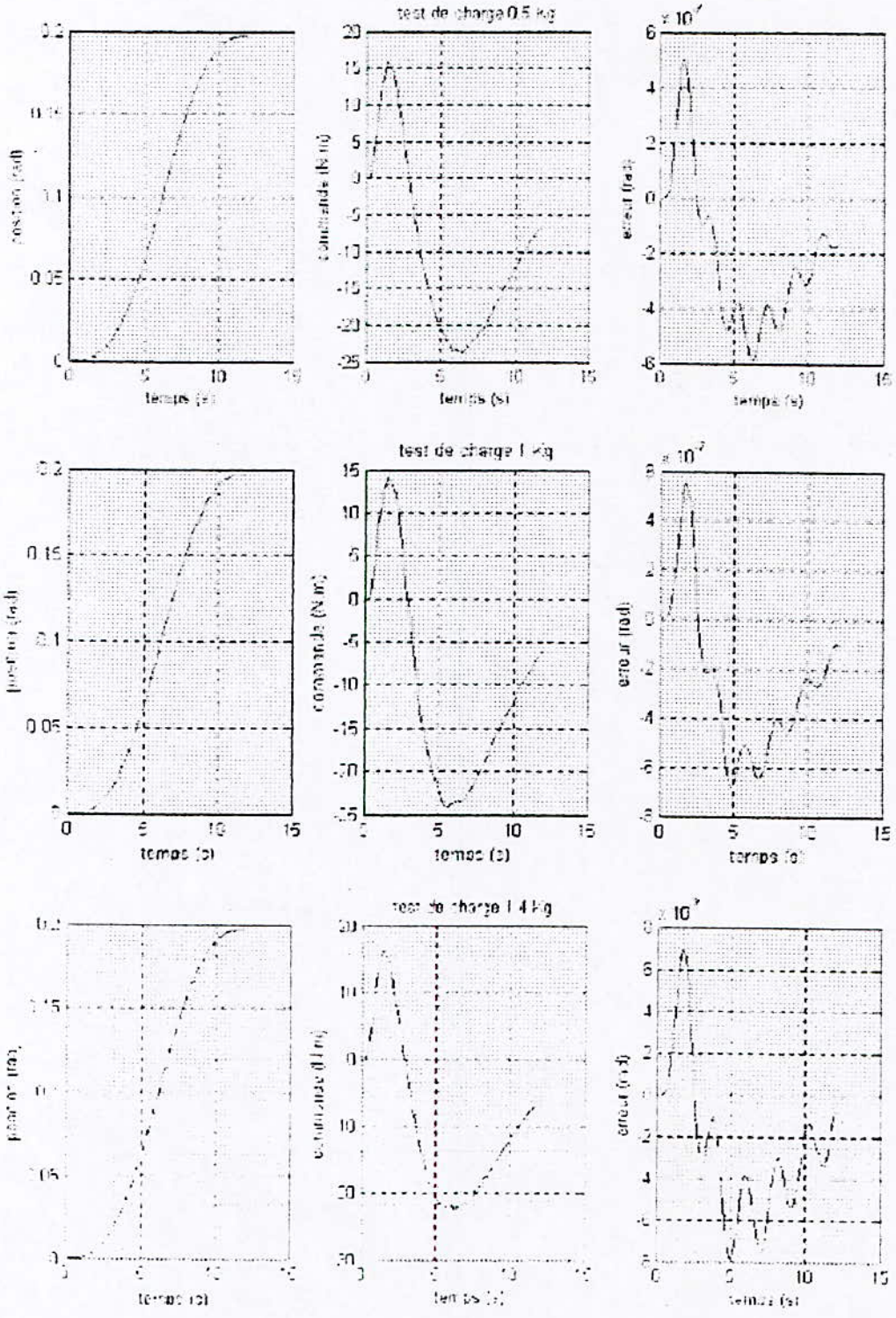


Fig (V.17) réponse en position du bras flexible, commande et erreur correspondantes au test de charge

Conclusion

La commande hybride présentée dans ce chapitre est appropriée pour la commande en position d'un bras de robot manipulateur à une liaison flexible voyant la haute performance atteinte.

L'apprentissage en ligne du réseau neuro-flou a été réalisé en optimisant les paramètres des fonctions d'appartenances par un algorithme génétique dynamique.

L'algorithme de Kohonen non supervisé n'a été utilisé que pour l'initialisation des fonctions d'appartenance et cela à partir de n'importe quels paramètres initiaux de ces dernières.

Sans hybridation des deux méthodes d'apprentissage, locale et globale, la réponse du bras de robot est lente. La rétro- propagation du gradient accélère le processus d'apprentissage au fur et à mesure que l'algorithme génétique optimise les différents paramètres.

Conclusion générale

Ce travail a porté sur l'étude d'une commande intelligente appliquée à une structure flexible.

La commande neuro-floue, obtenue par l'association de la logique floue aux réseaux de neurones a été appliquée au modèle non linéaire du robot manipulateur à une liaison flexible, obtenu à base de l'approche modale. Les résultats en simulation obtenus par cette méthode ne sont pas satisfaisants en plus du temps d'exécution lent de l'algorithme d'apprentissage, dû principalement à la rencontre des minima locaux de ce dernier. Les méthodes conventionnelles utilisées pour l'amélioration de cet algorithme sont à performances limitées.

Améliorer les performances de cette commande revient à considérer des algorithmes d'optimisation globale en introduisant la méthode auto organisatrice de Kohonen pour l'initialisation des fonctions d'appartenance.

L'algorithme génétique (AG) est donc utilisé afin d'optimiser les paramètres des fonctions d'appartenance considérées par la logique floue, au fur et à mesure que l'algorithme de Back propagation accélère le processus d'apprentissage du réseau.

Les testes de simulation ont montrés la robustesse d'une telle commande vis-à-vis des variations de charges et de la poursuite de trajectoires.

Il faut tout de même mentionner la difficulté de mettre en œuvre une commande neuro-floue par programmation MATLAB. Il est impossible de réaliser une inférence floue sur un réseau neuronal directement par la boitte outils 'Neural Network'. Il a fallu programmer ligne par ligne les algorithmes proposés.

Ce travail gagnerait par être enrichi par les axes suivants :

- Extension de la commande intelligente (multivariable) à un bras manipulateur à plusieurs liaisons flexibles

- Le Recuit simulé (RS), qu'est un processus d'optimisation similaire au processus physique d'échauffement d'un solide jusqu'à fondement suivi par son refroidissement jusqu'à cristallisation, est le plus approprié pour gagner de l'espace mémoire et donc de la vitesse d'exécution. Un des inconvénients de l'utilisation des AG étant le grand espace mémoire utilisé, une des méthodes à proposer serait l'hybridation de ces algorithmes avec le recuit simulé, qui permettrai de diminuer cet espace et garder les performances de la commande grâce aux AG.

-Mettre en œuvre une commande neuro-floue à base de l'inférence de Mamdani n'est point évidente par programmation. Il existe un outils 'neuro-fuzzy' nommé NEFCON, qui peut être associé à MATLAB Simulink. Il serait plus facile de travailler avec cet outil.

Comme toute conclusion, nous dirons que c'est la fin d'un début,
et ce n'est qu'un début...

Références bibliographiques

[AKB 98] **M. R. Akbarzadeh-T, E. Tunstel, K. Kumbla, M. Jamshidi**, « *Soft computing paradigm for hybrid fyzzy controlers: expermintals applications* », conférence IEEE, pp. 1205, 1998.

[ALA 99] **Daniel ALAZARD, Christelle CUMER, Pierre APKARIAN, Michel GAUVRIT, Gilles FERRERES,** «*Robustesse et commande optimale* », CEPADUES, déc 1999.

[ALA 01] **Daniel ALAZARD**, « *Méthodologie de synthèse de commande de vol d'un avion souple* », APII-JESA. Commande robuste. Volume – n°35/2001, pp.169-189, 2001.

[BAR 96] **J. P. BARRAT, M. BARRAT, Y. LECLUSE,** « *Application de la logique floue : commande de la température d'un four* », technique de l'Ingénieur R 7428, 1996.

[BLO 02] **Gérard BLOCH, Calogero BELTRAMI**, « *un réseau de neurones pour le contrôle du rapport air-carburant dans les moteurs à essence* », conf. on Neural System, Ecole Supérieure des Sciences et Technologies de Nancy, France, 2002

[BOI 88] **Jean-Daniel Boissonnat, Bertrand Faverjon, Jean-Pierre Marlet**, « *Technique de la robotique, Architecture et commande* », HERMES, Paris, 1988

[BOU 02] **Bernadette BOUCHON-MENNIER**, « *La logique floue : principe, aide à la décision* », Hèrmes sciences, Paris, France, 2002

[CAV 97] **David J. Cavuto**, «*An Exploration and development of current artificiel neural network theory and application with emphasis on artificiel life* », These de doctorat, The Cooper Union Albert Nerken school of Engineering, New York, USA, May 1997

[CHI 94] **M. Chiaberge, J. J. Merelo, L. M. Reyneri, A. Prieto, L. Zocca**, « *A comparison of neural network, lineair controllers, genetic algorithms and simulated annealing for real time control* », Conf. on Integrated Neural system for Robotic Applications, The Inter-University Cooperation between Italy and Spain,1994

[COU 02] **Rémi Coulom**, "*Apprentisage par renforcement utilisant des réseaux de neurones, avec des applications au contrôle de moteur* », thèse doctorat, Institut National Polytechnique de Grenoble, France, juin 2002

[DAB 05] **P. Dabrowski**, « *Introduction aux algorithmes génétiques* »,Conf. On genetic algorithm, EIVD, 23 fev 2005

[FAR 98] **Wael A. FARAG, Victor H. QUINTANA, Germano Lambert- Torres**, "*A genetic- based neuro-fuzzy approach for modelling and control of dynamical systems*", IEEE Trans. Neural Networks, Vol. 9, N°. 5, pp. 756-767, Sept 1998

[FAU 98] **Alain Faure**, « *Cybernétique des réseaux neuronaux, commande et perception* ».Hermes, 1998.

[FOS 93] **Wendy Foslien, Tariq Samad,** "*fuzzy controller synthesis with neural networks process models*", IEEE Int. Symposium on Intelligent control, pp. 370-375, Chicago, USA, Aug 1993

[GEN 97] **Sylviane GENTIL,** « *intelligence artificielle appliquée à l'automatique* », Technique de l'ingénieur, R7215, 1997

[GOM96] **S. C .P . GOMES**, « *Précision de la transmission du couple par un moto – réducteur électrique : modélisation et commande d'un bras rigide ou flexible avec compensation du frottement* », thèse de doctorat en automatique, Ecole Nationale Supérieure de l'aéronautique et de l'espace, Toulouse, France, 1992

[GOS 96] **Bernard Gosselin**, « *application de réseaux de neurones artificiels à la reconnaissance automatique de caractère manuscrits* », thèse doctorat, faculté polytechnique de Mons, France, 1996

[GUI 99] **A. Guillemin, N. Morel**, « *A self adaptive and smart system for blinds control* » rapport technique, EPFL, Lausanne, may1999

[HAY 98] **I. Hayachi, M. Umano, T. Maeda, A. Bastian, L. C. Jain**, "*Acquisition of fuzzy knowledge by NN and GA*", IEEE 2nd Intl. Conf. on knowledge-Based Intelligent Electronic Systems, pp. 21-23, Apl 1998

[HEI 92] **Jochen Heistermann,** "*A mixed genetic approach to the optimization of neural controller*", IEEE, pp. 459-464, 1992

[JEA 99] **Matthieu JEANNEAU, Daniel ALAZARD, Philipe MOUYON,** « *Contrôle semi-adaptatif pour structures flexibles* », JDA'99 Journées doctorales d'Automatique, GdR, Nancy, France, 21-23 Sep 1999

[JIK 97] **Yi JIKAI, Yan Hongping Hongtao, Hou Yuanbin,** « *Fuzzy control technique based on genetic algorithms optimizing and its application*", IEEE Int Conf. on Intelligent Processing Systems, pp.329-333, Oct 1997

[KHA 01] **F.Khaber, A. Hamzaoui, K. Zehar**, « *Commande adaptative floue pour des systèmes non linéaires incertains multi variables* », rapport technique, Labo. QUERE, Université Ferhat ABBAS, Sétif, Algérie, 2001

[KIM 94] **kim Chawee Ng, Yun Li**, "*Design of sophisticated fuzzy logic controllers using genetic algorithms*", IEEE Intl. conf. on Fuzzy Systems, pp. 1708-1712, 1994

[KRO 96] **Ben Cröse, Patrick van Der Smagt**, "*An introduction to Neural networks*", Edition Eighth, Nov 1996

[LAM 01] **H. K. Lam, S. H. Ling, F.H.F Leung, P. K. S. Tam**, "*Optimal and stable fuzzy controllers for nonlinear systems subject to parameter uncertainties using genetic algorithm*", IEEE Intl. conf. on Fuzzy Systems, pp. 908-911, 2001

[LOU 97] **M. LOUDINI**, «*Modélisation, Analyse et Méthodologie de Commande linguistique Floue d'un Bras manipulateur de Robot flexible* », Thèse de Magister en automatique, Ecole Nationale Polytechnique, Alger, Algérie 1997

[MAH 04] **MAHI,** « commande Hinf d'un bras flexible », thèse d'ingénieur, école polytechnique, Alger, Algérie, 2004.

[MAM 93] **E. H. Mamdani**, "*Twenty years of fuzzy control: experiences gained and lessons learnt*", IEEE Intl. conf. on fuzzy systems, pp. 339-344, 1993

[MAR 00] **A. T. Marcio, K. Marcani**, « *Optimization of Takagi-Sugeno Fuzzy controllers, using a genetic algorithm*", IEEE Intl. conf. on Fuzzy Systems, pp. 30-35, 2000

[POP 96] **D.POPOVIC, Ning HIONG,** « *Design of flexible structured fuzzy controllers using genetic algorithms* », IEEE, pp.1682-1686, 1996

[RIV 95] **I. Rivals, L. Personnaz, G. Dreyfus, J. L. Ploix,** « *Modélisation, classification et commande par réseaux de neurones : principes fondamentaux, méthodologie de conception et illustrations industrielles* », article dans « les réseaux de neurones pour la modélisation et la conduite des procédés », Ecole Supérieure de Physique et de Chimie industrielles, Paris, France, 1995

[RYU 00] **Jee-Hwan Ryu, Dong-Soo Kwon, Youngjin Park**, « *A robust controller design method for a flexible manipulator with a large time varying payload and parameter uncertainties*", journal of intelligent and robotic systems 27, pp.345-361, 2000

[SOR 99] **Fabrice SORIN, Lionel BROUSSARD, Pierre ROBLIN,** « *Régulation d'un processus industriel par réseaux de neurones* », technique de l'ingénieur, S 7 582, 1999

[TUN 95] **E. Tunstel, M. R. Akbarzadeh-T, K. Kumbla, M. Jamshidi**, "*Hybrid fuzzy control shemes for robotic systems*", IEEE Intl. conf. on Fuzzy Systems, pp. 171-176, 1995

[VAN 00] **Vanden BERGHEN, Hugues BERSINI**, « *Régulation directe adaptative et prédictive sur plusieurs pas de temps pour processus à plusieurs entrées et plusieurs sorties* », thèse doctorat, chapitre4, pp.139-153, Université Libre de Bruxelles, 2000.

[VIN96] **S. VINCENT**, «*Etude de la complémentarité d'actionneurs pour la commande active des structures flexibles* », thèse doctorat en automatique, Ecole Nationale Supérieure de l'aéronautique et de l'espace, Toulouse, France, 1996

[ZHO 00] **Z. J. ZHOU, Z. Y. Mao, Peter K. S. Tam**, « *On Designing optimal fuzzy Neural network controller using genetic algorithms* », IEEE Proceeding of the 3rd World Congress on Intelligent Control and Automation, pp. 391-395, Jun28-Jul2, 2000.

Annexes A

Valeurs des paramètres physiques du système étudié [LOU97]

Sous système	grandeur	valeur	Unité
Articulation+moteur	J_a	0.4	Kg.m^2
Bras	L	1	M
	ρ	1	Kg/m
	EI	1	N.m^2
	J_b	$\frac{1}{3}$	Kg.m^2
Charge	m_c	1	Kg
	J_c	5.10^{-3}	Kg.m^2

Mode	β	ω
1	1.2479	1.5572
2	4.0271	16.2171
3	7.1136	50.6037
4	10.1958	103.9535

Résumé : Les commandes conventionnelles appliquées à un système non linéaire, complexe et instable comme le robot manipulateur à une liaison flexible étudié dans cette thèse ne sont pas performantes en plus des difficultés de les mettre en œuvre.
Les contrôleurs à base de commandes non conventionnelles dont la commande neuro-floue optimisée par les algorithmes génétiques dynamiques, ont été appliqués à de tels bras en considérant leur modèle non linéaire.
La logique floue manipule les connaissances imparfaites et l'imprécision implicite. Elle est donc utilisée pour le contrôle et l'identification. Les réseaux de neurones permettent et facilitent la modélisation des fonctions non linéaires. Leur mode d'apprentissage est optimisé par un algorithme d'optimisation globale ce qui améliore considérablement les performances.

Mots clés : système dynamique, logique floue, algorithme génétique, modélisation non linéaire, bras flexible, réseau de neurones.

Abstract: The conventional commands applied on nonlinear complex irregular systems such as flexible single link manipulators for modelling and control aren't useful.
A genetic-based neuro-fuzzy approach for modelling and control of dynamical systems, one of the non conventional commands, is used in this thesis to evaluate the performance of the proposed modelling approach and has shows a good one.
The genetic algorithm plays a central role in the optimisation of the neuro-fuzzy approach, which combines the merit of the fuzzy logic theory to neural networks to build a linguistic model easy to control and identify.

Index terms: Dynamic control, fuzzy logic, genetic algorithm, non linear modelling, flexible robots neural networks.

ملخص: تقنيات التحكم التقليدية (العادية) المستعملة عادة على النظام اللاخطي، المعقد وغير المستقر مثل اليد الآلية ذات ارتباط أحادي لين والمدروسة في هذه الأطروحة ليست فعالة، زيادة على صعوبة استخدامها.
إن آلات المراقبة ذات تقنيات التحكم الحديثة كالتي تستعمل المنطق الغامض للمراقبة وتحديد الهوية، والشبكة العصبونية لتسهيل تصميم النماذج المعقدة، قد أعطت نتائج فعالة باستعمالها للخوارزميات الجينية (الوراثية) كالتي طبقت على اليد الآلية.

كلمات المفتاح: المنطق الغامض، التحكم النشيط، الخوارزميات الجينية، التصميم المعقد، الشبكات العصبونية، اليد الآلية اللينة.

Résumé : Les commandes conventionnelles appliquées à un système non linéaire, complexe et instable comme le robot manipulateur à une liaison flexible étudié dans cette thèse ne sont pas performantes en plus des difficultés de les mettre en œuvre.
Les contrôleurs à base de commandes non conventionnelles dont la commande neuro-floue optimisée par les algorithmes génétiques dynamiques, ont été appliqués à de tels bras en considérant leur modèle non linéaire.
La logique floue manipule les connaissances imparfaites et l'imprécision implicite. Elle est donc utilisée pour le contrôle et l'identification. Les réseaux de neurones permettent et facilitent la modélisation des fonctions non linéaires. Leur mode d'apprentissage est optimisé par un algorithme d'optimisation globale ce qui améliore considérablement les performances.

Mots clés : système dynamique, logique floue, algorithme génétique, modélisation non linéaire, bras flexible, réseau de neurones.

Abstract: The conventional commands applied on nonlinear complex irregular systems such as flexible single link manipulators for modelling and control aren't useful.
A genetic-based neuro-fuzzy approach for modelling and control of dynamical systems, one of the non conventional commands, is used in this thesis to evaluate the performance of the proposed modelling approach and has shows a good one.
The genetic algorithm plays a central role in the optimisation of the neuro-fuzzy approach, which combines the merit of the fuzzy logic theory to neural networks to build a linguistic model easy to control and identify.

Index terms: Dynamic control, fuzzy logic, genetic algorithm, non linear modelling, flexible robots neural networks.

ملخص: تقنيات التحكم التقليدية (العادية) المستعملة عادة على النظام اللاخطي، المعقد وغير المستقر مثل اليد الآلية ذات ارتباط أحادي لين والمدروسة في هذه الأطروحة ليست فعالة، زيادة على صعوبة استخدامها.
إن آلات المراقبة ذات تقنيات التحكم الحديثة كالتي تستعمل المنطق الغامض للمراقبة وتحديد الهوية، والشبكة العصبونية لتسهيل تصميم النماذج المعقدة، قد أعطت نتائج فعالة باستعمالها للخوارزميات الجينية (الوراثية) كالتي طبقت على اليد الآلية.

كلمات المفتاح: المنطق الغامض، التحكم النشيط، الخوارزميات الجينية، التصميم المعقد، الشبكات العصبونية، اليد الآلية اللينة.