

20/98

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

ECOLE NATIONALE POLYTECHNIQUE

D.E.R de Génie Electrique & Informatique

FILIERE : AUTOMATIQUE

Mémoire

En vue d'obtenir le diplôme
D'ingénieur d'Etat en AUTOMATIQUE

المدرسة الوطنية المتعددة التقنيات
المكتبة — BIBLIOTHEQUE
Ecole Nationale Polytechnique

THEME

**APPLICATION DE LA COMMANDE
FLOUE ET NEURO-FLOUE A
STRUCTURE DECENTRALISEE AU
ROBOT PUMA 560**

Proposé et dirigé par :

Mr D. BOUKHETALA
Mr M.S. BOUCHERIT

Fait par :

M.A. ZEROUAL

PROMOTION 1998

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR
ET DE LA RECHERCHE SCIENTIFIQUE

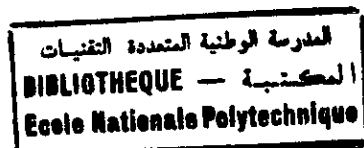
ECOLE NATIONALE POLYTECHNIQUE

D.E.R de Génie Electrique & Informatique

FILIERE : AUTOMATIQUE

Mémoire

En vue d'obtenir le diplôme
D'ingénieur d'Etat en AUTOMATIQUE



THEME

APPLICATION DE LA COMMANDE
FLOUE ET NEURO-FLOUE A
STRUCTURE DECENTRALISEE AU
ROBOT PUMA 560

Proposé et dirigé par :

Mr D. BOUKHETALA
Mr M.S. BOUCHERIT

Fait par :

M.A. ZEROUAL

PROMOTION 1998

ملخص: قمنأ فف هءا العمل بتطبق التحكم اللامركزي الغامض و العصبي الغامض على ربوت PUMA 560. فف أول الأمر تم تشكيل ثلاث منظماء من صنف ممدني (ثلاث، خمس وسبع مجالات للإتماء) وتم اعتماد المنظم دو ثلاث مجالات للإتماء لبساطته. ثم استعملت حوارزمية التعلم BP للحصول على منظم من صنف سوجينو الذي يعطي نتائج جيدة خلال زمن مختصر .

كلمات مفتاحية: التحكم اللامركزي ، المنطق الغامض ، العصبي الغامض ، قاعدة معارف ، حوارزمية تعلم BP

Résumé :

Ce travail consiste en l'application de la commande décentralisée floue et neuro-floue au robot PUMA560 . En premier temps, trois régulateurs de type MAMDANI (trois, cinq et sept classes d'appartenance) sont synthétisés, celui à trois classes est retenu pour sa simplicité. Après, l'algorithme d'apprentissage BP est utilisé pour synthétiser le régulateur de type SUGENO qui donne de bonnes performances avec un temps de calcul réduit.

Mots clés :

Commande décentralisée, logique floue, neuro-floue, base de connaissances, algorithme d'apprentissage BP.

Summary :

This work consists of the application of the decentralized fuzzy and neuro-fuzzy control to the robot PUMA560 . In first time, three controllers of typical MAMDANI (three, five and seven classify membership) are synthesized, that to three classify membership is retained for its simplicity. After, the learning algorithm BP is used to synthesize the controller of typical SUGENO that gives good performances with a time of calculation reduced.

Key words :

Decentralized control, fuzzy logic, neuro -fuzzy, basis of knowledge, learning algorithm BP.



المدرسة الوطنية المتعددة الفنون
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

Dédicace

A la mémoire de mon regretté père

A ma très chère mère, qu'elle reçoive toute ma gratitude

et ma reconnaissance.

A mon cher frère « Nouri ».

A mes frères et sœurs

A « Dekka » et tout mes amis.

Je dédie ce travail

Aduane





REMERCIEMENTS

Au terme de ce travail, je tiens à remercier vivement mes promoteurs, Mr D. Boukhetala et Mr M.S. Boucherit, et je profite de l'occasion qui m'est offerte pour leurs exprimer toute ma gratitude pour leurs apport scientifique précieux et pour les conseils judicieux et l'encouragement constant qu'ils n'ont cessé de me prodiguer tout au long de ce travail.

Je suis très reconnaissant aux honorables membres de jury pour l'intérêt qu'il ont témoigné pour mon travail en acceptant de le lire et de le juger en y apportant leurs compétences.

Toute ma reconnaissance et mes remerciements à tout ceux qui m'ont aidé et contribué à la réalisation de ce mémoire.

SOMMAIRE



INTRODUCTION GENERALE

Chapitre I : MODELISATION DU ROBOT PUMA560

| | |
|---|----|
| I.1 Introduction | 1 |
| I.2 Modélisation cinématique | 2 |
| I.2.1 Paramétrage de DENAVITT & HARTENBERG | 2 |
| I.2.2 Présentation du robot PUMA 560 | 4 |
| I.2.2.1 Modélisation géométrique | 5 |
| I.2.2.2 Définition des indicateurs de configuration | 6 |
| I.3 Modélisation dynamique | 7 |
| I.3.1 Introduction | 7 |
| I.3.2 Formalisme de LAGRANGE-EULER | 7 |
| I.3.3 Modèle dynamique du robot puma 560 | 12 |
| I.4. Génération de trajectoire | 16 |
| I.5. conclusion | 18 |

Chapitre II : LA COMMANDE PAR LOGIQUE FLOUE

| | |
|--|----|
| II.1 Introduction | 19 |
| II.2 Ensembles flous | 20 |
| II.2.1 Fonction d'appartenance | 20 |
| II.2.2 Variable linguistique | 20 |
| II.2.3 Opérations sur les sous-ensembles flous | 20 |
| II.3 Logique floue | 22 |
| II.4 Règles floues | 22 |
| II.5 Raisonnement et prise de décision | 22 |
| II.6 Commande par logique floue | 23 |
| II.6.1 Structure du réglage par logique floue | 23 |
| II.6.2 Configuration interne d'un régulateur par logique floue | 24 |
| II.6.2.1 Fuzzification | 25 |

| | |
|--|----|
| II.6.2.2 Inférence..... | 25 |
| II.6.2.3 Défuzzification..... | 25 |
| II.7 Types de régulateurs flous | 25 |
| II.7.1 Régulateur flou de type MAMDANI | 26 |
| II.7.2 Régulateur flou de type SUGENO | 26 |
| II.8 Etapes pour la conception d'un réglage par logique floue..... | 26 |
| II.9 Propriétés d'un réglage par logique floue | 27 |
| II.10 Application de la commande floue au robot puma560 | 27 |
| II.10.1 Décentralisation..... | 27 |
| II.10.2 Commande floue du robot..... | 28 |
| II.10.2.1 Régulateur flou de MAMDANI..... | 28 |
| II.10.2.2 Loi de commande | 29 |
| II.10.2.3 La matrice des règles de MACVICAR et WHELAN | 29 |
| II.10.2.4 Les gains de normalisation..... | 30 |
| II.10.2.5 La trajectoire test..... | 30 |
| II.10.2.6 Synthèse des régulateurs | 31 |
| II.10.3 Régulateur à trois classes | 31 |
| II.10.4 Régulateur flou à cinq classes..... | 32 |
| II.10.5 Régulateur flou à sept classes..... | 33 |
| II.10.6 Résultats de simulations | 34 |
| II.11 conclusion..... | 35 |



Chapitre III : COMMANDE NEURO-FLOUE

| | |
|--|----|
| III.1 Introduction..... | 49 |
| III.2 Architecture des réseaux de neurones adaptatifs | 50 |
| III.3 Exemples des réseaux adaptatifs..... | 51 |
| III.4 L'algorithme de back- propagation..... | 52 |
| III.5 La méthode de gradient..... | 53 |
| III.6 Le réseau neuro-flou | 54 |
| III.6.1 Architecture..... | 54 |
| III.6.2 Détermination du régulateur | 56 |
| III.7 Application de la commande neuro-floue au robot puma560..... | 57 |
| III.7.1 Introduction..... | 57 |

| | | |
|---|--|----|
| III.7.2 Synthèse du régulateur..... | المدرسة الوطنية المتعددة التقنيات BIBLIOTHEQUE المكتبة Ecole Nationale Polytechnique | 57 |
| III.7.3 Algorithme d'apprentissage..... | | 59 |
| III.7.4 Valeurs initiales des paramètres..... | | 60 |
| III.7.5 Résultats..... | | 60 |
| III.8 Conclusion..... | | 62 |

CONCLUSIONS ET PERSPECTIVES

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

INTRODUCTION GENERALE

INTRODUCTION GENERALE

L'analyse et la commande d'un système complexe pose de nombreux problèmes, parmi lesquels le grand nombre de données à traiter simultanément ainsi que les limites physiques des calculateurs. Pour pallier ces problèmes, on procède par la décentralisation sous laquelle le système sera vu composé de plusieurs sous systèmes interconnectés, chaque sous système sera commandé par une unité locale. Le robot est un de ces systèmes complexes où s'impose la décentralisation pour de la synthèse des lois de commande.[SER 89]

L'emploi des méthodes conventionnelles de commande, s'appuie crucialement sur une modélisation du processus à commander, or celle ci n'est pas toujours évidente, surtout lorsqu'il s'agit d'un système complexe et non linéaire tel que le robot.

D'autres problèmes dus à la complexité numérique des algorithmes d'identification et à la quantité d'information acquise limitent les performances de la commande. Pour résoudre de tels problèmes, de nouvelles stratégies de commande ont été élaborées, une des plus attrayantes est la commande par logique floue[LEE 90a][BUH 94], qui se base sur l'imitation des aspects approximatifs et qualitatifs du raisonnement humain. Des citations conditionnelles linguistiques de type « 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).

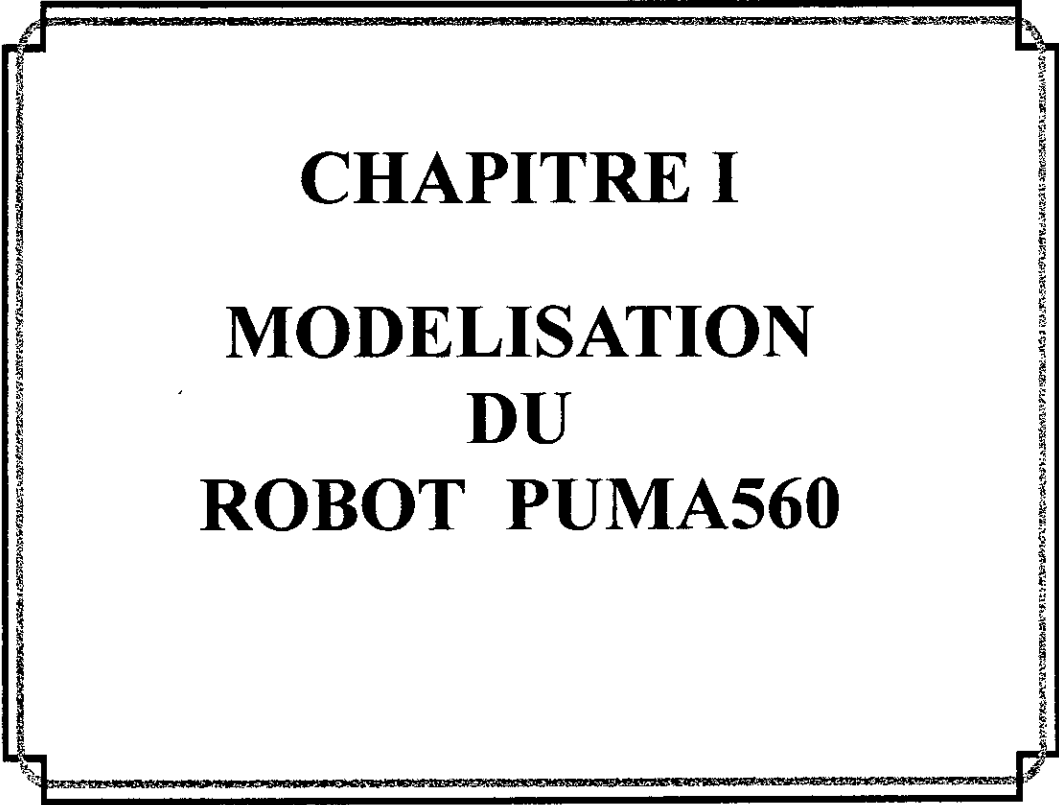
Durant les dernières années on a constaté l'adaptation des techniques d'apprentissage développées pour les réseaux de neurones artificiels aux systèmes d'inférences floue[SHI 95]. Cette adaptation a permis l'obtention des règles floues et leurs amélioration pour avoir de bonnes performances.

Le présent travail se compose en trois chapitres.

Le premier chapitre présente les différentes modélisations du robot *PUMA560*, ainsi que son comportement dynamique en boucle ouverte obtenu par simulation.

Le deuxième chapitre est la partie dans laquelle on présente la théorie de la commande par logique floue et ses différentes approches. En fin de ce chapitre, on expose les performances et les tests de robustesse effectués pour différents schémas de commande , avec une étude comparative.

Dans le dernier chapitre, nous avons exploité les capacités d'apprentissage des réseaux adaptatifs pour synthétiser un régulateur de type Sugeno en vue de simplifier le régulateur et réduire le temps de calcul de la commande. Les performances de ce régulateur pour la commande du robot *PUMA560* sont exposées en fin de chapitre.



CHAPITRE I

MODELISATION

DU

ROBOT PUMA560

CHAPITRE I

MODELISATION DU ROBOT PUMA560

I.1 INTRODUCTION :

Le mot « robot » a été créé en 1920 par l'écrivain tchèque Karel Capek, dans une de ses pièces de théâtre, pour dénommer un androïde construit par un savant et capable d'accomplir tous les travaux normalement exécutés par un homme.

Un robot se compose d'un socle, d'au moins un bras muni d'organes de préhension (généralement des pinces, parfois des ventouses ou des électroaimants), d'actionneurs pneumatiques, hydrauliques ou électriques, de capteurs de position, de pression, etc., et d'organes de traitement de l'information. C'est la nature de ces deux derniers types d'organes qui permet de classer les robots en quatre catégories de complexité croissante.

Les robots les plus simples et les plus nombreux sont des automatismes séquentiels à cycle simple ou multiple, fixe ou modifiable d'après la nature des pièces à manipuler. Lorsque le trajet de la « main » est complexe, on peut « enseigner » au robot les mouvements d'un opérateur humain en les enregistrant sur une bande magnétique. Le troisième niveau est constitué par les robots à commande numérique, dont la programmation nécessite l'emploi de langages spéciaux. Enfin, les robots « intelligents » sont capables de prendre des décisions d'après leur état et celui de leur environnement, et sont dotés pour cela de capteurs évolués et d'une grande capacité de traitement de l'information. Ils savent reconnaître la forme et l'orientation des objets sur un écran de télévision et sont capables de réagir à la prononciation de mots appartenant à un certain vocabulaire.

I.2 MODELISATION CINEMATIQUE :

I.2.1 Paramétrage de DENAVITT & HARTENBERG :

Un manipulateur série consiste en une chaîne de liaisons connectées par des articulations (charnières ou glissières).

Un manipulateur à n degrés de liberté compte n liaisons et n articulations. Les liaisons permettent de maintenir une relation fixe entre les articulations du manipulateur à chaque extrémité de la liaison.

La base du manipulateur est la liaison 0 et ne fait pas partie des n liaisons ; la liaison 1 est reliée à la liaison base par l'articulation 1, et il n'y a pas d'articulation au bout de la dernière liaison.

Chaque liaison est caractérisée par son propre repère. Utilisant les transformations homogènes, on peut trouver les matrices de passage d'un repère à un autre.

DENAVITT et HARTENBERG [PAU 83] ont établi une transformation basée sur un paramétrage particulier. On résume ce paramétrage dans les étapes suivantes (figure I-1).

Etape 1: Numéroté chaque liaison et articulation, en commençant de la base (liaison 0) jusqu'à l'élément terminal (liaison n). Le mouvement de la liaison i est une rotation si l'articulation i est un pivot (charnière) et est une translation si l'articulation i est prismatique (glissière).

Etape 2: Etablir le repère R_i de chaque liaison i (articulation) $R_i(O_i, X_i, Y_i, Z_i)$, tel que :

- L'axe Z_i est choisi le long de l'axe de l'articulation $i+1$.
- L'axe X_i peut être n'importe quel axe normal à Z_{i-1} , son sens le long de cette normale va de l'articulation i vers l'articulation $i+1$.
- Dans le cas où Z_{i-1} et Z_i ne sont pas parallèles, on choisit l'axe X_i parallèle ou antiparallèle au vecteur $Z_{i-1} \wedge Z_i$. Ce choix implique que l'axe X_i est dirigé suivant la normale commune entre Z_{i-1} (articulation i) et Z_i (articulation $i+1$), et l'axe Y_i est choisi de façon à former un trièdre droit.

Etape 3 : On définit deux groupes de paramètres :

- deux paramètres de localisation de l'axe Z_i dans la liaison i .
 a_i = distance entre Z_{i-1} et Z_i , le long de X_i .
 α_i = angle (Z_{i-1}, Z_i).
- deux paramètres de mouvement (rotation et /ou translation) de la liaison.
 d_i = distance (X_{i-1}, X_i) mesurée suivant Z_{i-1} et qui représente les coordonnées de O_i dans R_{i-1} le long de Z_{i-1} .
 θ_i = angle(X_{i-1}, X_i) obtenu par rotation de X_{i-1} vers X_i autour de Z_{i-1} .

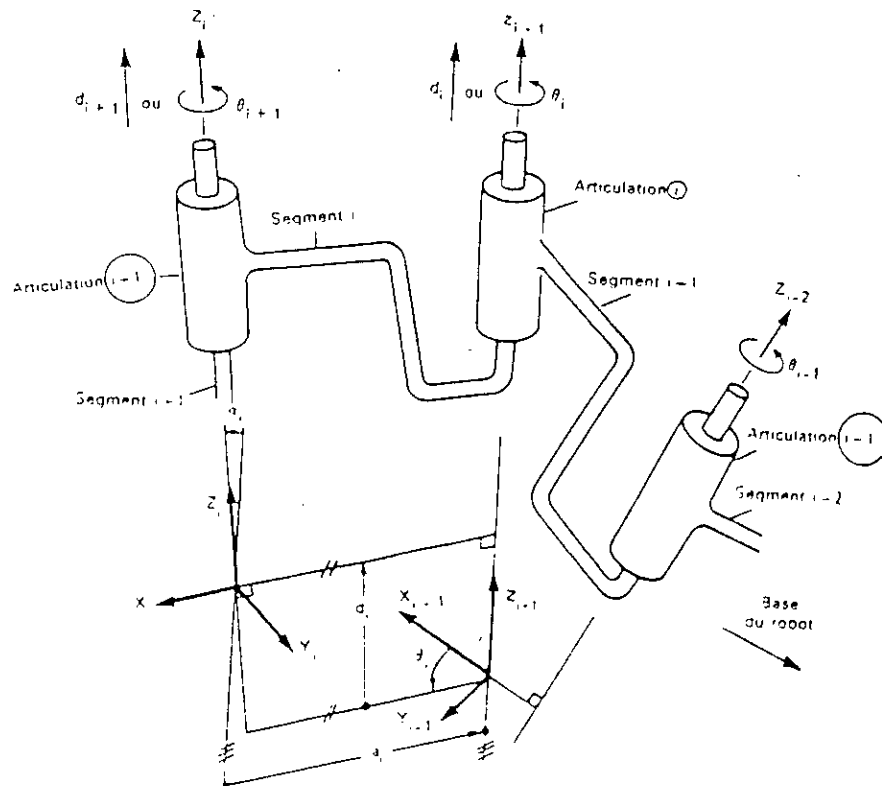


Figure (I.1) : Paramètres de DENAVITT et HARTENBERG

Soit T_{i-1}^i la matrice de transformation de coordonnées homogènes qui permet le passage du repère R_{i-1} au repère R_i . Ce dernier se fait suivant quatre mouvements simples :

1. Rotation autour de Z_{i-1} d'un angle θ_i ;
2. Translation le long de Z_{i-1} , de distance d_i ;
3. Translation le long de X_i (X_i =rotation X_{i-1}), de distance a_i ;
4. Rotation autour de X_i , d'un angle α_i .

La matrice de passage T_{i-1}^i est donc le produit de quatre transformations homogènes :

$$T_{i-1}^i = \text{rot}(Z_{i-1}, \theta_i) \cdot \text{trans}(Z_{i-1}, d_i) \cdot \text{trans}(X_i, a_i) \cdot \text{rot}(X_i, \alpha_i).$$

où : rot \equiv rotation, trans \equiv translation.

Ce qui donne :

$$T_{i-1}^i = \begin{bmatrix} C_{i-1}^i & d_{i-1}^i \\ 0 & 1 \end{bmatrix} \tag{I.1}$$

où $d_{i-1}^i = [a_i \cos \theta_i \quad a_i \sin \theta_i \quad d_i]^T$ est le vecteur droit pour la translation ;

$$\text{et } C_{i-1}^i = \begin{bmatrix} \cos \theta_i & -\sin \theta_i \cdot \cos \alpha_i & \sin \theta_i \cdot \cos \alpha_i \\ \sin \theta_i & \cos \theta_i \cdot \cos \alpha_i & -\cos \theta_i \cdot \sin \alpha_i \\ 0 & \sin \alpha_i & \cos \alpha_i \end{bmatrix} \tag{I.2}$$

est la matrice rotation.

D'où :

$$T_{i-1}^i = \begin{bmatrix} \cos \theta_i & -\sin \theta_i \cdot \cos \alpha_i & \sin \theta_i \cdot \sin \alpha_i & a_i \cos \theta_i \\ \sin \theta_i & \cos \theta_i \cdot \cos \alpha_i & -\cos \theta_i \cdot \sin \alpha_i & a_i \sin \theta_i \\ 0 & \sin \alpha_i & \cos \alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (I.3)$$

I.2.2 Présentation du robot PUMA 560 : [MAD 97] [AMS 86]

La figure(I.2) représente le robot PUMA 560 qui présente six articulations. On s'arrêtera dans ce travail aux trois premières articulations qui sont rotationnelles, la première évoluant sur un plan horizontal, et les deux autres évoluant sur des plans verticaux. On supposera donc les trois autres articulations fixes. Le bras de robot étudié est donc à trois degrés de libertés.

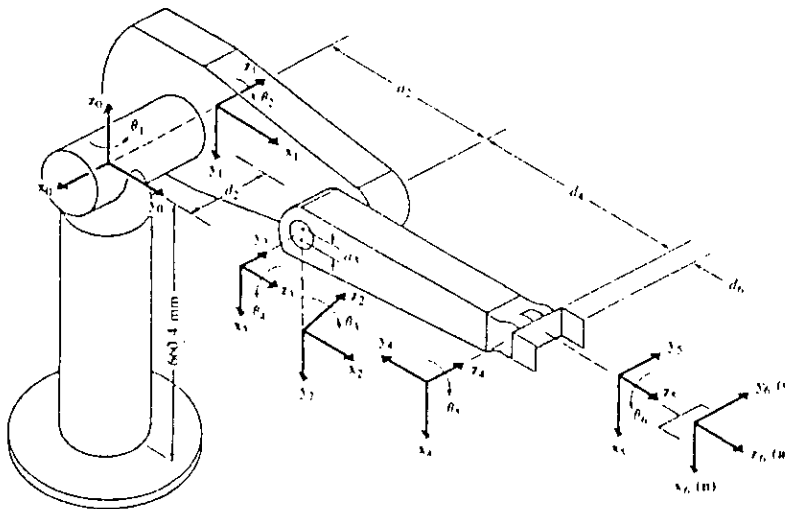


Figure (I.2) : Représentation du robot PUMA560

L'application de la paramétrage de DENAVITT et HARTENBERG sur le PUMA 560 (figure(I.2)) donne lieu aux paramètres suivants (tableau(I.1)).

| N° DE LA LIAISON | VARIABLE | θ_i | A_i | D_i | α_i |
|------------------|----------|------------|-------|-------|-------------|
| 1 | Q_1 | θ_1 | 0 | 0 | -90° |
| 2 | Q_2 | θ_2 | L_2 | d_2 | 0 |
| 3 | Q_3 | θ_3 | L_3 | 0 | 0 |

Tableau (I.1) : paramétrage de D&H du PUMA 560

Le repère R_3 est sur le centre de masse de l'effecteur (fin de la liaison 3).

Le vecteur des coordonnées généralisées est :

$$q = [\theta_1 \ \theta_2 \ \theta_3]^T \quad (I.4)$$

Les matrices de transformations associées à ce robot sont alors :

$$T_0^1 = \begin{bmatrix} C1 & 0 & -S1 & 0 \\ S1 & 0 & C1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T_1^2 = \begin{bmatrix} C2 & -S1 & 0 & L_2C2 \\ S2 & C2 & 0 & L_2S2 \\ 0 & 0 & 1 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T_2^3 = \begin{bmatrix} C3 & -S3 & 0 & L_3C3 \\ S3 & C3 & 0 & L_3S3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (I.5)$$

On aura alors :

$$T_0^3 = T_0^1 \cdot T_1^2 \cdot T_2^3 = \begin{bmatrix} C1 \cdot C23 & -C1 \cdot S23 & -S1 & C1(L_2C2 + L_3C23) - d_2S1 \\ S1 \cdot C23 & -S1 \cdot S23 & C1 & S1(L_2C2 + L_3C23) + d_2C1 \\ -S23 & -C23 & 0 & -(L_2S2 + L_3S23) \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (I.6)$$

avec : $C_i = \cos\theta_i$, $C_{ij} = \cos(\theta_i + \theta_j)$
 $S_i = \sin\theta_i$, $S_{ij} = \sin(\theta_i + \theta_j)$

I.2.2.1 Modélisation géométrique :

a) Modèle géométrique direct :

Les transformations précédentes permettent de déterminer l'expression de la position de l'organe terminal $r(R_0)$ à partir des positions articulaires q_i .

$$r(R_0) = F(q) = T_0^3 r(R_3) \quad (I.7)$$

$$\text{avec : } \begin{cases} r(R_3) = [0 & 0 & 0 & 1]^T \\ r(R_0) = [x & y & z & 1]^T \end{cases} \quad (I.8)$$

Des équations (I.6), (I.7) et (I.8), on obtient :

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} C1(L_2C2 + L_3C23) - d_2S1 \\ S1(L_2C2 + L_3C23) + d_2C1 \\ -(L_2S2 + L_3S23) \end{bmatrix} \quad (I.9)$$

L'équation (I.9) constitue le modèle géométrique direct du PUMA 560.

b) Modèle géométrique inverse : (MGI)

Il s'agit maintenant d'établir la relation $q=F(r)$. Pour se faire, il existe plusieurs solutions :

- Transformation inverse , par Paul & al (1981)
- Approche géométrique par Lee & Ziegler (1984)
- Méthode des matrices duelles, par Denavit (1956)
- Quaternions dual , par Young & Arden Stein (1964)

Pour établir le modèle géométrique inverse du PUMA 560, on procédera par approche géométrique :

On présente d'abord l'approche géométrique pour le calcul du MGI du PUMA 560 (à 6 degrés de liberté) .

En se basant sur la géométrie du bras humain , on peut définir les indicateurs de configurations : arm(bras), elbow(coude) associés aux trois premières articulations et wrist(poignet) associé quant à lui aux trois dernières articulations du PUMA 560 .

Il existe pour le PUMA quatre solutions différentes pour les trois premières articulations, et à chacune d'elles correspondent deux solutions possibles et différentes pour les trois dernières.

Les deux premiers indicateurs servent à choisir une seule solution pour les trois premières articulations et l'indicateur wrist sert à choisir une solution pour les trois dernières.

I.2.2.2 Définition des indicateurs de configuration :

Les indicateurs de configuration sont préspecifiés par l'utilisateur en vue de trouver la solution inverse .Dans notre cas, on les définira comme suit :

$$arm = \begin{cases} +1 & \text{right arm} \\ -1 & \text{left arm} \end{cases} \quad elbow = \begin{cases} +1 & \text{above arm} \\ -1 & \text{below arm} \end{cases} \quad wrist = \begin{cases} +1 & \text{wrist down} \\ -1 & \text{wrist up} \end{cases}$$

Avec ces paramètres et pour les trois premières articulations, on retrouve effectivement le même MGD déjà trouvé (équation (I.9)) [NED 96].

En effectuant des projections sur les plans (X_{i-1}, Y_{i-1}) des différentes articulations , en utilisant quelques règles trigonométriques simples et en s'arrêtant aux trois premières articulations, on obtient le modèle géométrique inverse du PUMA 560 [NED 96] :

$$\theta_1 = \arctg \left\{ \frac{-arm.y.\sqrt{x^2 + y^2 - d_2^2} - x.d_2}{-arm.x.\sqrt{x^2 + y^2 - d_2^2} - y.d_2} \right\} \quad -\pi \leq \theta_1 \leq \pi \quad (I.10.a)$$

$$\theta_2 = \arctg \left\{ \frac{\sin \alpha . \cos \beta + arm.elbow . \cos \alpha}{\cos \alpha . \sin \beta - arm.elbow . \sin \alpha . \cos \beta} \right\} \quad -\pi \leq \theta_2 \leq \pi \quad (I.10.b)$$

$$\theta_3 = \phi - \frac{\pi}{2} \quad -\pi \leq \theta_3 \leq \pi \quad (I.10.c)$$

avec :

$$\cos \alpha = \frac{-arm \cdot \sqrt{x^2 + y^2 - d_2^2}}{\sqrt{x^2 + y^2 + z^2 - d_2^2}}, \quad \sin \alpha = \frac{-z}{\sqrt{x^2 + y^2 + z^2 - d_2^2}}$$

$$\sin \beta = \frac{x^2 + y^2 + z^2 - d_2^2 + L_2^2 - L_3^2}{2a_2 \cdot \sqrt{x^2 + y^2 + z^2 - d_2^2}}$$

$$\cos \phi = \frac{a_2^2 + L_3^2 - x^2 - y^2 - z^2 + d_2^2}{2L_2 L_3}, \quad \sin \phi = arm \cdot elbow \cdot \sqrt{1 - (\cos \phi)^2}$$

I.3 MODELISATION DYNAMIQUE :

I.3.1 Introduction :

On s'intéresse maintenant aux efforts actionneurs produits par les mouvements du manipulateur. Il s'agit d'établir les équations différentielles qui relient les efforts actionneurs

$T_i(t)$ aux positions, vitesses et accélérations articulaires ($q_i(t)$, $\dot{q}_i(t)$, $\ddot{q}_i(t)$ respectivement).

L'ensemble de ces équations constitue le modèle dynamique du manipulateur.

Deux exploitations du modèle sont possibles :

- Résoudre les équations différentielles pour obtenir les mouvements $q_i(t)$ pour des $T_i(t)$ donnés dans le but d'une simulation du comportement dynamique. On parle alors de modèle dynamique direct (MDD).
- Calculer les efforts $T_i(t)$ pour des mouvements $q_i(t)$ connus, donnés à l'avance ou mesurés. Cette approche constitue le modèle dynamique inverse (MDI).

Pour obtenir le modèle dynamique, on peut utiliser :

1. Le formalisme des théorèmes généraux de la dynamique (méthode de Newton-Euler).
2. Le formalisme de D'Alembert qui fait appel aux principes des puissances virtuelles.
3. Le formalisme des équations de Lagrange-Euler pour établir les modèles dynamiques des robots manipulateurs.

I.3.2 Formalisme de Lagrange-Euler : [MAD 97] [PAU 83]

Les équations de Lagrange-Euler sont :

$$\frac{d}{dt} \frac{\partial T}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} + \frac{\partial E_D}{\partial q_i} = T_i \quad i=1, \dots, n \quad (I.11)$$

L étant le lagrangien , défini par :

$$L = E_c - E_p \quad (I.12)$$

Où :

- E_c : énergie cinétique totale du robot manipulateur ;
- E_p : énergie potentielle totale du robot manipulateur ;
- E_D : énergie de dissipation ;
- T_i : force généralisée à la $i^{\text{ème}}$ articulation ;
- n : degrés de liberté du robot manipulateur (nombre de liaisons ou articulations) ;
- q_i : $i^{\text{ème}}$ coordonnée généralisée ;
- \dot{q}_i : $i^{\text{ème}}$ vitesse généralisée .

♦ **Energie cinétique :**

Commençons par trouver l'expression de la vitesse à l'articulation i .

$$V_0^i = \frac{dr_0^i}{dt} \quad (I.13)$$

avec :

$$r_0^i = T_0^i r_i^i$$

Où r_i^i est la $i^{\text{ème}}$ coordonnée homogène exprimée dans le repère R_i . Comme les liaisons sont toutes rigides, on a $\frac{dr_i^i}{dt} = 0$, donc :

$$V_0^i = \sum_{j=1}^i \frac{\partial T_0^i}{\partial q_j} \frac{dq_j}{dt} r_i^i \quad (I.14)$$

ou encore :

$$V_0^i = \sum_{j=1}^i U_{ij} \dot{q}_j r_i^i \quad (I.15)$$

où :

$$U_{ij} = \begin{cases} 0 & j > i \\ T_0^{j-1} Q_j T_{j-1}^i & j \leq i \end{cases} \quad (I.16)$$

l'équation (I.15) n'est qu'une forme compacte de l'équation (I.14), avec :

$$Q_j = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (I.17)$$

si la liaison j est rotationnelle, et :

$$Q_j = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad (I.18)$$

Si la liaison j est translationnelle, L'énergie cinétique de l'élément i (dans la liaison i) sera alors :

$$dE_{ci} = \frac{1}{2} \text{trace}(V_i V_i^T) dm \quad (I.19)$$

En remplaçant V_i de l'équation (I.15) dans l'équation (I.19), on trouve :

$$dE_{ci} = \frac{1}{2} \text{trace} \left(\sum_{j=1}^i \sum_{k=1}^i U_{ij} (r_i^i r_i^{iT} dm) U_{ik}^T \dot{q}_j \dot{q}_k \right) \quad (I.20)$$

D'où l'expression de l'énergie cinétique de la liaison i :

$$E_{ci} = \frac{1}{2} \text{trace} \left(\sum_{j=1}^i \sum_{k=1}^i U_{ij} J_i U_{ik}^T \dot{q}_j \dot{q}_k \right) \quad (I.21)$$

Avec :

$$J_i = \begin{bmatrix} \int x_i^2 dm & \int x_i y_i dm & \int x_i z_i dm & \int x_i dm \\ \int x_i y_i dm & \int y_i^2 dm & \int y_i z_i dm & \int y_i dm \\ \int x_i z_i dm & \int y_i z_i dm & \int z_i^2 dm & \int z_i dm \\ \int x_i dm & \int y_i dm & \int z_i dm & \int dm \end{bmatrix} \quad (I.22)$$

l'énergie cinétique des actionneurs est définie par :

$$E_{ca} = \frac{1}{2} \sum_{i=1}^n I_i \dot{q}_i^2 \quad (I.23)$$

où : I_i est un moment d'inertie dans le cas d'une rotation et une masse dans le cas d'une translation.

L'énergie cinétique totale est alors :

$$E_c = E_{ca} + \sum_{i=1}^n E_{ci} \quad (I.24)$$

♦ *Energie potentielle :*

Soit m_i la masse de la liaison i. L'énergie potentielle est définie par :

$$E_p = \sum_{i=1}^n -m_i g^T r_0^i \quad (I.25)$$

De l'équation (I.19) on déduit :

$$E_p = -\sum_{i=1}^n m_i g^T T_0^i r_i^i \quad (I.26)$$

avec :

$$g^T = [0 \quad 0 \quad -|g| \quad 1] \quad (I.27)$$

où g est la gravité.

♦ **Energie de dissipation :**

Elle est donnée par la relation suivante :

$$E_D = \frac{1}{2} \sum_{i=1}^n f_{vi} \dot{q}_i^2 \quad (I.28)$$

où :

f_{vi} est le coefficient de frottement visqueux dû à la liaison i .

Des équations (I.24), (I.26), (I.28) on déduit l'expression générale du Lagrangien.

$$L = \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^i \sum_{k=1}^i \text{trace}(U_{ij} J_i U_{ik}^T) \dot{q}_j \dot{q}_k + \sum_{i=1}^n m_i g^T T_0^i r_i^i \quad (I.29)$$

Où J_i , U et g^T sont définis par les équations (I.22), (I.16) et (I.27) respectivement.

En remplaçant la valeur de L trouvée (I.28) dans l'équation de Lagrange (I.11), on trouvera pour la liaison i :

$$T_i = \sum_{j=1}^n \sum_{k=1}^j \text{trace}(U_{jk} J_j U_{ji}^T) \ddot{q}_k + \sum_{j=1}^n \sum_{k=1}^j \sum_{l=1}^j \text{trace}(U_{jkl} J_j U_{ji}^T) \dot{q}_k \dot{q}_l + f_{vi} \dot{q}_i - \sum_{j=1}^n m_j g^T U_{ji} r_j^j \quad (I.30)$$

avec :

$$U_{ij} = \begin{cases} T_0^k Q_k T_{k-1}^{j-1} Q_j T_{j-1}^i & k \leq j \leq i \\ T_0^{j-1} Q_j T_{j-1}^{k-1} Q_k T_{k-1}^i & j \leq k \leq i \\ 0 & j < i < k \end{cases}$$

En posant :

$$M_{ij}(q) = \sum_{k=\max(i,j)}^n \text{trace}(U_{kj} J_k U_{ki}^T), \quad i=1, n, \quad j=1, n \quad (I.31)$$

$$G_i(q) = -\sum_{j=i}^n m_j g^T U_{ji} r_j^j \quad (I.32)$$

$$N_{ijk}(q) = \sum_{l=\max(i,j,k)}^n \text{trace}(U_{ljk} J_l U_{li}^T), \quad i=1,n, \quad j=1,n, \quad k=1,n \quad (I.33)$$

$$H_i(\dot{q}_i) = f_{vi} \dot{q}_i \quad (I.34)$$

L'équation (I.30) devient alors :

$$T_i = \sum_{j=1}^n M_{ij}(q) \ddot{q}_j + \sum_{j=1}^n \sum_{k=1}^n N_{ijk}(q) \dot{q}_j \dot{q}_k + G_i(q) + H_i(\dot{q}_i) \quad (I.35)$$

En réécrivant l'équation (I.35) sous forme matricielle on obtient :

$$T = M(q)\ddot{q} + N(q,\dot{q}) + G(q) + H(\dot{q}) \quad (I.36)$$

avec :

$q(t), \dot{q}(t), \ddot{q}(t) \in R^n$ représentent respectivement les positions, vitesses et accélérations articulaires.

$M(q) \in R^{n \times n}$: matrice d'inertie symétrique définie positive.

$N(q,\dot{q}) \in R^n$: vecteur représentant les efforts de coriolis et centrifuge.

$G(q) \in R^n$: vecteur des efforts gravitationnels.

$H(\dot{q}) \in R^n$: vecteur représentant les frottements visqueux.

T : vecteur de forces et/ou couples moteurs.

La détermination de $N(q,\dot{q})$ de l'équation (I.33) peut s'avérer longue. Une méthode rapide pour le calcul de $N(q,\dot{q})$ revient à calculer les matrices centrifuges et Coriolis indépendamment, et qui seront multipliés ensuite par leurs vecteurs.

La somme des vecteurs obtenus donne alors le vecteur $N(q,\dot{q})$. On obtient ces deux matrices par dérivation de la matrice d'inertie (principe de la conservation de l'énergie).

L'équation (I.36) devient alors :

$$T = M(q)\ddot{q} + D(q)[\dot{q}\dot{q}] + C(q)[\dot{q}]^2 + G(q) + H(\dot{q}) \quad (I.37)$$

où :

$D(q) \in R^{n \times \frac{n(n-1)}{2}}$: matrice des couples de Coriolis ;

$C(q) \in R^{n \times n}$: matrice des couples centrifuges ;

$[\dot{q}\dot{q}] = [\dot{q}_1 \dot{q}_2 \cdots \dot{q}_1 \dot{q}_n \quad \dot{q}_2 \dot{q}_3 \cdots \dot{q}_2 \dot{q}_n \cdots \dot{q}_{n-1} \dot{q}_n] \in R^{\frac{n(n-1)}{2}}$

$[\dot{q}]^2 = [\dot{q}_1^2 \cdots \dot{q}_n^2] \in R^n$

Avec :

$$D_{ij} = 2\beta^{i,kl}$$

$$C_{ij} = \beta^{ij}$$

β étant le symbole de Christoffel [MAD 97],[AMS 86] défini par :

$$\beta^{i,jk} = \frac{1}{2} \left[\frac{\partial M_{ij}}{\partial q_k} + \frac{\partial M_{ik}}{\partial q_j} - \frac{\partial M_{jk}}{\partial q_i} \right] \quad (I.38)$$

La matrice d'inertie M étant symétrique définie positive, on a alors :

$$\begin{cases} \frac{\partial M_{ij}}{\partial q_k} = \frac{\partial M_{ji}}{\partial q_k} & \forall i, j, k \\ \frac{\partial M_{ij}}{\partial q_k} = 0 & \text{pour } i \geq k, j \geq k \end{cases} \quad (I.39)$$

Le vecteur représentant les frottements $H(q)$ sera plus généralement donné par la relation :

$$H(q) = f_{vi} \dot{q}_i + f_{si} \operatorname{sgn}(\dot{q}_i) \quad i = 1, n \quad (I.40)$$

Où f_{vi} et f_{si} sont respectivement les coefficients des frottements visqueux et secs de la $i^{\text{ème}}$ articulation.

On remarquera aussi que l'entrée de commande est indépendante pour chaque articulation du robot manipulateur, ceci est dû à la supposition de la non flexibilité des articulations et liaisons du manipulateur.

I.3.3. Modèle dynamique du robot puma 560 : [MAD 97] [AMS 86]

Le robot manipulateur considéré assure trois mouvements rotationnels ; la première articulation évoluant sur un plan horizontal et les deux autres évoluant sur des plans verticaux . Le modèle dynamique associé à ce manipulateur est :

$$T = M(q)\ddot{q} + N(q, \dot{q}) + G(q) + T_{m0} \quad (I.41)$$

avec :

$$M(q) = \begin{bmatrix} I_1 + I_2 C23^2 + I_3 C2^2 + I_4 C2C23 & I_5 S23 + I_6 S2 & I_5 S23 \\ I_5 S23 + I_6 S2 & I_7 + I_4 C3 & I_8 + 0.5 I_4 C3 \\ I_5 S23 & I_8 + 0.5 I_4 C3 & I_9 \end{bmatrix} \quad (I.42)$$

$$N(q, \dot{q}) = \begin{bmatrix} -(2(I_3 S_2 C_2 + I_2 S_2 C_2) + I_4 (C_2 S_2 + S_2 C_2)) \dot{q}_1 \dot{q}_2 - (2I_2 S_2 C_2 + I_4 C_2 S_2) \dot{q}_1 \dot{q}_3 \\ + (I_6 C_2 + I_5 C_2) \dot{q}_2^2 + (2I_5 C_2) \dot{q}_2 \dot{q}_3 + (I_5 C_2) \dot{q}_3^2 \\ (I_3 C_2 S_2 + I_2 C_2 S_2 + 0.5 I_4 (S_2 C_2 + C_2 S_2)) \dot{q}_1^2 - (I_4 S_3) \dot{q}_2 \dot{q}_3 - (0.5 I_4 S_3) \dot{q}_3^2 \\ (I_2 S_2 C_2 + 0.5 I_4 C_2 S_2) \dot{q}_1^2 + (0.5 I_4 S_3) \dot{q}_2^2 \end{bmatrix} \quad (I.43)$$

$$G(q) = \begin{bmatrix} 0 \\ -(m_3 L_2 + 0.5 m_2 L_2) g C_2 - 0.5 m_3 L_3 g C_2 \\ -0.5 m_3 L_3 g C_2 \end{bmatrix} \quad (I.44)$$

$$T = [T_1 \quad T_2 \quad T_3]^T \quad (I.45)$$

Où :

$$\begin{aligned} C_i &= \cos \theta_i & C_{ij} &= \cos(\theta_i + \theta_j) \\ S_i &= \sin \theta_i & S_{ij} &= \sin(\theta_i + \theta_j) \end{aligned}$$

avec :

$$\begin{aligned} I_1 &= I_{yy1} + I_{xx2} + m_2 d_2 (d_2 + e) + m_3 d_2^2 + I_{xx3} + I_{xxt} + m_t d_2^2 + I_{M1} \\ I_2 &= I_{yy3} - I_{xx3} + I_{yyt} - I_{xxt} + m_t L_3^2 \\ I_3 &= I_{yy2} - I_{xx2} + (m_3 + m_t) L_2^2 \\ I_4 &= (m_3 + 2m_t) L_2 L_3 \\ I_5 &= (0.5 m_3 + m_t) L_3 d_2 \\ I_6 &= 0.5 m_2 L_2 (d_2 + e) + (m_3 + m_t) d_2 L_2 \\ I_7 &= I_{zz2} + I_{zz3} + m_3 L_2^2 + I_{zzt} + m_t (L_2^2 + L_3^2) + I_{M2} \\ I_8 &= I_{zz3} + I_{zzt} + m_t L_3^2 \\ I_9 &= I_8 + I_{M3} \end{aligned}$$

I_{Mi} moments d'inertie des différents moteurs

$I_{xxt}, I_{yyt}, I_{zzt}$ moments d'inertie totale par rapport aux principaux axes de l'effecteur.

Le vecteur des couples additifs T_{m0} représente l'effet de la charge, il est calculé par la matrice jacobienne, cette dernière étant la dérivée du vecteur position de l'effecteur.

$$J_i(q) = \frac{\partial p}{\partial q_i} \quad (I.46)$$

pour le PUMA 560, et en remplaçant p (la position de l'effecteur) par son expression donnée par l'équation (I.9)

$$J(q) = \begin{bmatrix} -S_1(L_2 C_2 + L_3 C_2) - d_2 C_1 & -C_1(L_2 S_2 + L_3 S_2) & -C_1 L_3 S_2 \\ C_1(L_2 C_2 + L_3 C_2) - d_2 S_1 & -S_1(L_2 S_2 + L_3 S_2) & -S_1 L_3 S_2 \\ 0 & -(L_2 C_2 + L_3 C_2) & -L_3 C_2 \end{bmatrix} \quad (I.47)$$

Le couple dû au port de la charge est alors :

$$T_{m0} = m_0 J^T(q) \left[J(q)\ddot{q} + J(q,\dot{q})\dot{q} + g \right] \quad (\text{I.48})$$

avec :

$$g = [0 \quad 0 \quad 9.81]^T$$

Les paramètres réels de ce robot sont :

- Masses des différentes liaisons :

$$\begin{array}{lll} m_2 = 17.40 \text{ kg} & m_3 = 5.04 \text{ kg} & m_4 = 0.82 \text{ kg} \\ m_5 = 0.35 \text{ kg} & m_6 = 0.09 \text{ kg} & m_t = m_4 + m_5 + m_6 = 1.26 \text{ kg} \end{array}$$

- Paramètres géométriques :

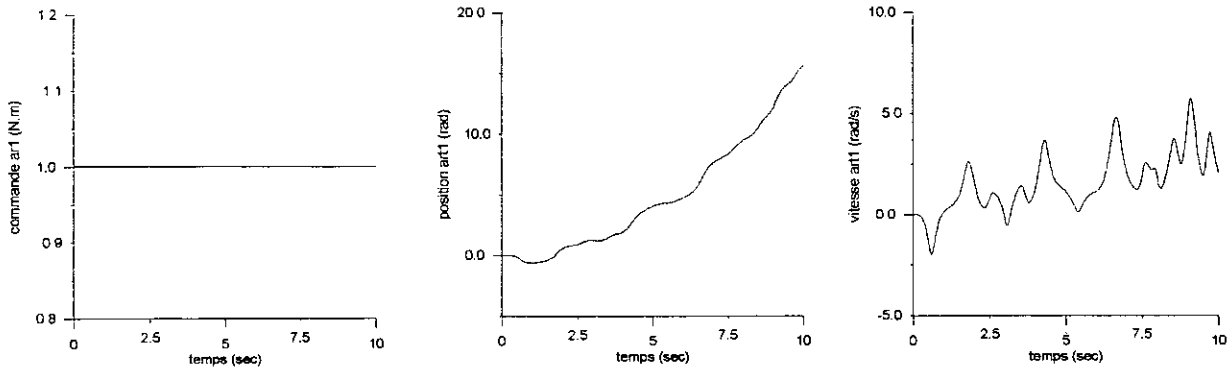
$$d_2 = 149.09 \text{ mm} \quad L_2 = 431.8 \text{ mm} \quad L_3 = 433.07 \text{ mm}$$

- Paramètres d'inertie :

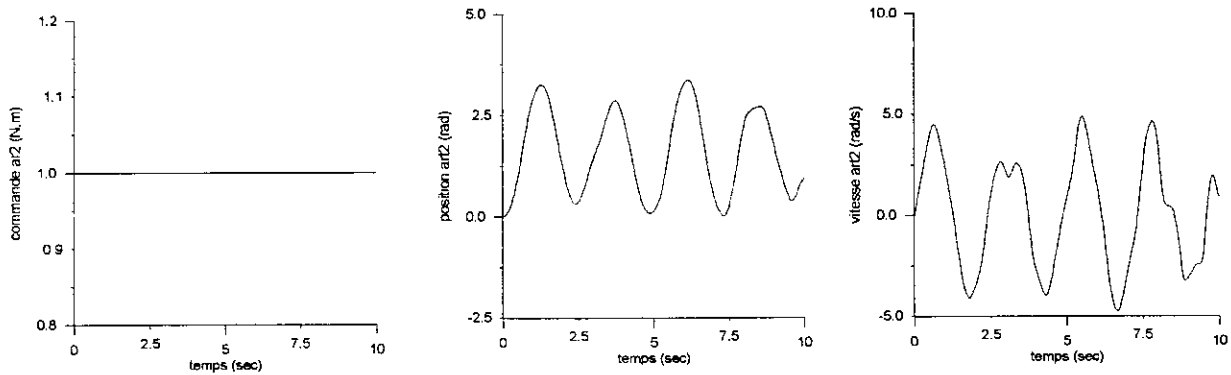
| N° de la liaison | $I_{xxi} [\text{kg m}^2]$ | $I_{yyi} [\text{kg m}^2]$ | $I_{zzi} [\text{kg m}^2]$ | $I_{Mi} [\text{kg m}^2]$ |
|------------------|---------------------------|---------------------------|---------------------------|--------------------------|
| 1 | - | $350 \cdot 10^{-3}$ | - | 1.14 |
| 2 | $130 \cdot 10^{-3}$ | $524 \cdot 10^{-3}$ | $539 \cdot 10^{-3}$ | 4.71 |
| 3 | $192 \cdot 10^{-3}$ | $15.4 \cdot 10^{-3}$ | $212 \cdot 10^{-3}$ | 0.83 |
| 4 | $1.30 \cdot 10^{-3}$ | $1.80 \cdot 10^{-3}$ | $1.80 \cdot 10^{-3}$ | - |
| 5 | $0.30 \cdot 10^{-3}$ | $0.30 \cdot 10^{-3}$ | $0.40 \cdot 10^{-3}$ | - |
| 6 | $0.04 \cdot 10^{-3}$ | $0.15 \cdot 10^{-3}$ | $0.15 \cdot 10^{-3}$ | - |
| 4+5+6 | $1.64 \cdot 10^{-3}$ | $2.25 \cdot 10^{-3}$ | $2.35 \cdot 10^{-3}$ | - |

Tableau (I.2) : paramètres d'inertie du PUMA 560

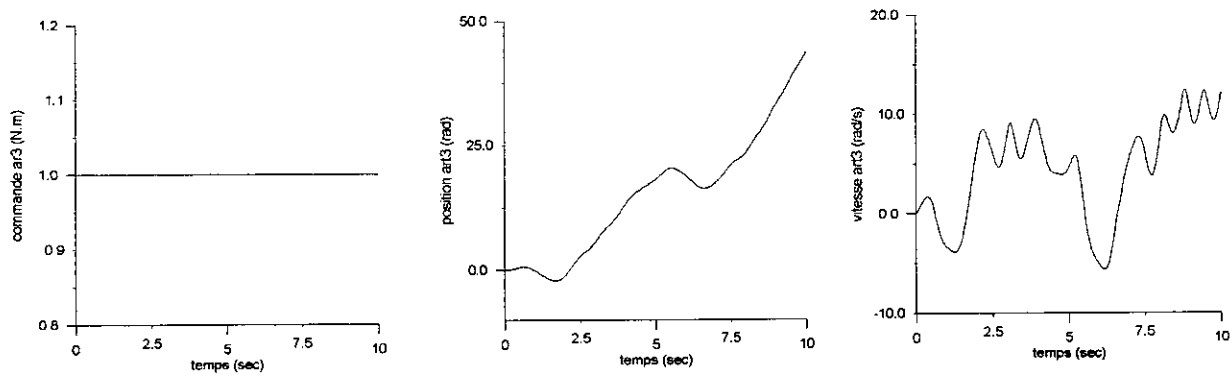
Les réponses des articulations à un échelon unité sur une intervalle suffisante de temps sont données par les figures (I.4), (I.5) et (I.6)



Figure(I.4) : Réponse de la 1^{ère} articulation Position et vitesse (en boucle ouverte)



Figure(I.5) : Réponse de la 2^{ème} articulation Position et vitesse (en boucle ouverte)



Figure(I.6) : réponse de la 3^{ème} articulation Position et vitesse (en boucle ouverte)

I.4. GENERATION DE TRAJECTOIRE :

La dynamique du robot exige d'imposer des trajectoires réalisables. La continuité en position, vitesse et accélération offre au robot la possibilité de poursuivre la trajectoire avec des commandes réalisables.

La décomposition de la tâche en plusieurs points intermédiaires nécessite une continuité du premier et du second ordre [BAL 95].

L'imposition d'une position finale constante par exemple, nécessite l'utilisation d'un polynôme du troisième degré permettant une continuité en position et en vitesse.

Cette tâche se traduit par le passage d'un état d'équilibre à un autre état d'équilibre, ces deux états définiront les conditions aux limites du polynôme d'interpolation:

$$P(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3$$

$$\begin{cases} p(0) = \theta_0 & \dot{p}(0) = \dot{\theta}_0 \\ p(t_f) = \theta_f & \dot{p}(t_f) = \dot{\theta}_f \end{cases} \quad (I.49)$$

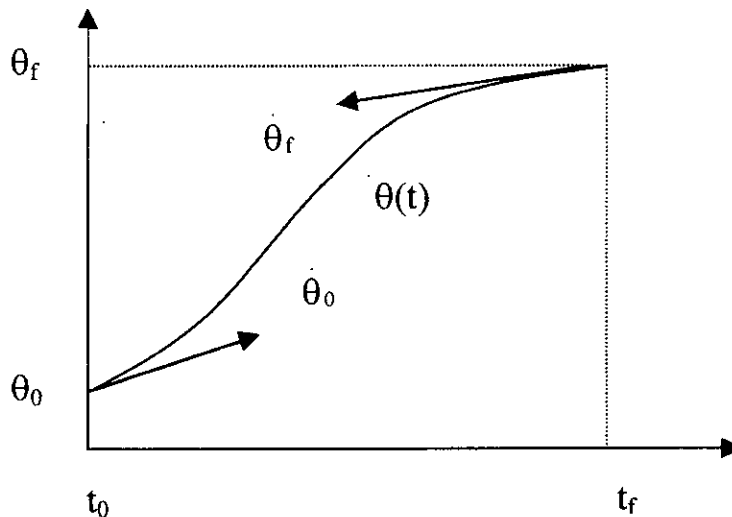


figure (I.7) : Génération d'un polynôme d'interpolation

Généralement $\dot{\theta}_0 = \dot{\theta}_f = 0$

A partir des conditions aux limites (équation I.49), on calcule les différents coefficients du polynôme $p(t)$, on aura alors :

$$\begin{cases} a_0 = \theta_0 \\ a_1 = \dot{\theta} \\ a_2 = \frac{3}{t_f^3}(\theta_f - \theta_0) - \frac{2}{t_f}\dot{\theta}_0 - \frac{1}{t_f}\dot{\theta}_f \\ a_3 = -\frac{2}{t_f^3}(\theta_f - \theta_0) + \frac{1}{t_f^2}(\dot{\theta}_f + \dot{\theta}_0) \end{cases} \quad (I.50)$$

On peut aussi assurer cette continuité par une cycloïde, on dira alors que la trajectoire est cycloïdale, elle sera donnée par l'équation :

$$q_d(t) = \begin{cases} q_d(0) + \frac{D}{2\pi} \left[2\pi \frac{t}{t_f} - \sin\left(2\pi \frac{t}{t_f}\right) \right] & \text{pour } 0 \leq t \leq t_f \\ q_d(t_f) & \text{pour } t > t_f \end{cases} \quad (I.51)$$

avec $D = q_d(t_f) - q_d(0)$ est le déplacement, et t_f est l'instant final du mouvement, et q_d est la trajectoire de référence.

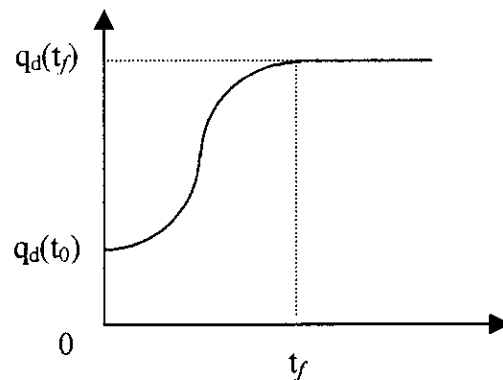


figure (I.8) : trajectoire cycloïdale

Pour le PUMA 560, il existe une trajectoire cycloïdale test proposée par LEAHVY. Les différentes articulations (art1, art2, art3) se déplacent de la position $(-50^\circ, -135^\circ, 135^\circ)$ à la position $(45^\circ, -85^\circ, 30^\circ)$ en un temps de mouvement égal à 1.5 secondes. Cette trajectoire est choisie car elle excite toute la dynamique de ce bras de robot.

Si maintenant la consigne n'est pas constante, pour une trajectoire donnée (trajectoire de l'effecteur du bras de robot), on peut aisément déterminer les déplacements et vitesses dus à chaque articulation $(\theta_i, \dot{\theta}_i)$, ceci en utilisant les équations du modèle géométrique et cinématique inverse des robots.

En effet, à chaque point de coordonnées cartésiennes, on détermine les coordonnées généralisées correspondantes. Suivant la trajectoire de l'effecteur (élément terminal du bras de robot), on trouve une relation donnant les coordonnées généralisées (q_i) qui décrivent cette même trajectoire.

I.5 CONCLUSION

Dans ce chapitre nous avons établi le modèle géométrique direct et inverse du robot *PUMA 560* ainsi que son modèle dynamique. Ce dernier a été utilisé pour la simulation de son comportement dynamique en boucle ouverte, et il sera également utilisé pour la simulation des lois de commande présentées aux chapitres II et III.

Ce modèle dynamique obtenu par le formalisme d'EULER-LAGRANGE a permis de mettre en évidence les couplages qui existent entre les différentes articulations.

Le comportement dynamique du robot manipulateur exige une trajectoire spécifique (continue en position, en vitesse et en accélération), permettant ainsi le contrôle du robot avec des commandes physiquement réalisables.



CHAPITRE II
LA COMMANDE
PAR
LOGIQUE FLOUE

CHAPITRE II

LA COMMANDE PAR LOGIQUE FLOUE

II.1.INTRODUCTION :

La logique floue 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. Mais ce n'est que depuis quelques années que la commande floue a connu, essentiellement au Japon, un essor remarquable ; Elle a été appliquée à des problèmes aussi divers que la purification de l'eau, la fabrication du ciment, la marche automatisée d'une rame de métro, etc. [BAR 96].

L'attitude des automaticiens à l'égard de la commande floue a d'abord été réservée : après avoir pendant des décennies, affirmé la nécessité d'identifier le mieux possible un système pour pouvoir construire une commande ayant des performances satisfaisantes, fallait il faire confiance à cette méthode nouvelle, qui prétendait remplacer les commandes basées sur l'identification, par des techniques s'appuyant sur le savoir faire humain plutôt que sur des équations .

Actuellement, cette attitude a évolué. On peut dire que la commande floue va peu à peu prendre place dans la panoplie de l'ingénieur contemporain, sans supplanter les méthodes traditionnelles, et qu'elle constituera un complément précieux dans le cas des systèmes difficilement identifiables ou dont les paramètres subissent des variations brutales.

II.2 ENSEMBLES FLOUS : [BAR 96]

Si l'on considère une grandeur physique e , dire que e appartient à un certain ensemble flou revient à lui attribuer une propriété de définition imprécise (linguistique) : e est grand, petit voisin de zéro etc. La frontière d'un tel ensemble est mal définie, si bien qu'un élément peut appartenir à la fois à un ensemble et à son complémentaire.

II.2.1 Fonction d'appartenance : [KAU 87] [DUB 87]

A la variable e , dont l'ensemble des valeurs possibles est E , et au sous-ensemble E_1 de E , on associe une fonction $\mu_{E_1}(e)$ comprise entre 0 et 1, appelée fonction d'appartenance (on dit aussi degré d'appartenance) qui représente la possibilité pour que la variable e ait la qualité associée au sous-ensemble E_1 .

Le plus souvent, on utilise pour les fonctions d'appartenance des formes triangulaires ou trapézoïdales. Il s'agit des formes simples composées par des morceaux de droites. Cependant, il existe d'autres formes tel que la forme gaussienne ou la cloche.

On présente les ensembles flous comme suit :

$$A = \{(x, \mu_A(x)) / x \in X\} \quad \text{forme générale} \quad \text{(II.1)}$$

$$A = \int \mu_A(x) / x \quad \text{forme continue} \quad \text{(II.2)}$$

$$A = \sum_{x \in X} \mu_A(x) / x \quad \text{forme discrète} \quad \text{(II.3)}$$

| | |
|------------|---------------------------------------|
| $\mu_A(x)$ | fonction d'appartenance |
| X | ensemble de référence |
| x | la valeur de la variable floue donnée |

NB :

On parle souvent d'univers de discours, qui n'est rien d'autre que l'ensemble flou.

II.2.2 Variable linguistique :

Dans la logique floue, on introduit la notion de la variable linguistique dont la valeur est un terme du langage humain, contrairement à une variable numérique dont la valeur est un nombre. La représentation mathématique d'une valeur floue est réalisée par l'application des notions des ensembles flous. Une valeur floue peut être représentée par une fonction d'appartenance.

II.2.3 Opérations sur les sous-ensembles flous :

Afin de manipuler les ensembles flous, il est nécessaire de définir certaines opérations élémentaires.

Soit A et B deux sous-ensembles flous dans X caractérisés respectivement par $\mu_A(x)$ et $\mu_B(x)$.

♦ **Egalité :**

On dit que A et B sont égaux si et seulement si

$$\mu_A(x) = \mu_B(x) \quad \forall x \in X \quad (\text{II.4})$$

♦ **Inclusion :**

On dit que A est inclus dans B si et seulement si

$$\mu_A(x) \leq \mu_B(x) \quad \forall x \in X \quad (\text{II.5})$$

♦ **Complément :**

Le complément A^c de A est défini par

$$\mu_{A^c}(x) = 1 - \mu_A(x) \quad \forall x \in X \quad (\text{II.6})$$

♦ **Intersection :**

L'intersection de A et B est définie par

$$\mu_{A \cap B}(x) = T(\mu_A(x), \mu_B(x)) \quad \forall x \in X \quad (\text{II.7})$$

Où T est, en générale l'opération Min. C'est une norme triangulaire.

♦ **Union :**

L'union de A et B est définie par

$$\mu_{A \cup B}(x) = \perp(\mu_A(x), \mu_B(x)) \quad \forall x \in X \quad (\text{II.8})$$

Où \perp est, en générale, l'opérateur Max. C'est une conorme triangulaire.

♦ **Produit cartésien :**

Soit A_1, A_2, \dots, A_n des sous-ensembles flous dans X_1, X_2, \dots, X_n respectivement. Le produit cartésien de A_1, A_2, \dots, A_n , est un sous-ensemble flou dans l'espace produit de X_1, \dots, X_n , ayant pour fonction d'appartenance :

$$\mu_{A_1 \times A_2 \times \dots \times A_n}(x_1, x_2, \dots, x_n) = \text{Min}(\mu_{A_1}(x_1), \mu_{A_2}(x_2), \dots, \mu_{A_n}(x_n)) \quad (\text{II.9})$$

♦ **Relation floue :**

Soit U_1, U_2, \dots, U_n des univers de discours. Une relation floue est un sous-ensemble flou dans $U_1 * U_2 * \dots * U_n$ exprimé par :

$$R_{U_1 \times \dots \times U_n} = \{((x_1, \dots, x_n), \mu_R(x_1, \dots, x_n)) / (x_1, \dots, x_n) \in U_1 \times \dots \times U_n\} \quad (\text{II.10})$$

II.3 LOGIQUE FLOUE :

La logique floue est une extension de la logique classique. Elle permet de traduire l'incertitude et l'imprécision du langage humain et attribue un degré de vérité à une proposition donnée. Pour la suite de cet exposé, on donne quelques définitions ;

Soit X une variable floue et A une caractéristique.

➤ La proposition :

Une proposition floue est une description de la variable floue, donnée sous la forme suivante « X est A ».

➤ La conjonction :

La conjonction de deux propositions est réalisée par l'opérateur ET :

« X₁ est A ET X₂ est B »

➤ La disjonction :

La disjonction de deux propositions est réalisée par l'opérateur OU :

« X₁ est A OU X₂ est B »

➤ L'implication :

L'implication est définie comme suit :

« si X₁ est A alors X₂ est B »

II.4 REGLES FLOUES :

Une règle floue est une relation exprimée à l'aide d'une implication. Par exemple :

*SI ERREUR est POSITIVE ET VARIATION D'ERREUR est NEGATIVE
ALORS COMMANDE est GRANDE.*

Cette règle est constituée de trois propositions :

- ◆ La proposition 1 est « ERREUR est GRANDE ».
 - ◆ La proposition 2 est « VARIATION D'ERREUR est NEGATIVE ».
- Ces deux propositions forment les prémisses de la règle.
- ◆ La proposition 3 est « COMMANDE est GRANDE », elle forme la conclusion de la règle.
- La conjonction des prémisses, l'implication et la conclusion forment la règle floue.

II.5 RAISONNEMENT ET PRISE DE DECISION :

Cela consiste à manipuler et utiliser les propositions et les règles floues dans le but d'obtenir une décision. Pour cela on utilise deux modes de raisonnement inspirés de la logique classique qui sont : **MP**, modus ponens et **MT**, modus tollens.

Soient P et C deux propositions logiques. On définit :

- **Le MP** par $P \Rightarrow C$
Si P est vraie alors C est vraie.
- **Le MT** par $\bar{C} \Rightarrow \bar{P}$
Si C est fausse alors P est fausse.

En logique floue, on généralise ces deux modes :

- **Le GMP :**

Règle floue : si X est A alors Y est B

$$\mu_A \qquad \mu_B$$

Fait observé : X est A'

$$\mu_{A'}$$

conclusion : Y est B'

$$\mu_{B'}$$

- **Le GMT :**

Règle floue : si X est A alors Y est B

$$\mu_A \qquad \mu_B$$

Fait observé : Y est B'

$$\mu_{B'}$$

Conclusion : X est A'

$$\mu_{A'}$$

NB : En commande, on utilise le GMP, pour respecter la causalité. Alors que pour les systèmes experts, on utilise les deux.

II.6 COMMANDE PAR LOGIQUE FLOUE.

II.6.1 Structure du réglage par logique floue :

Un domaine d'application de la logique floue qui devient de plus en plus important, est celui du réglage et de la commande des processus industriels. En effet cette méthode permet d'obtenir une loi de réglage souvent très efficace sans devoir faire des études théoriques approfondies. [BUH 94]

La figure (II.1) présente la structure d'un réglage par logique floue.

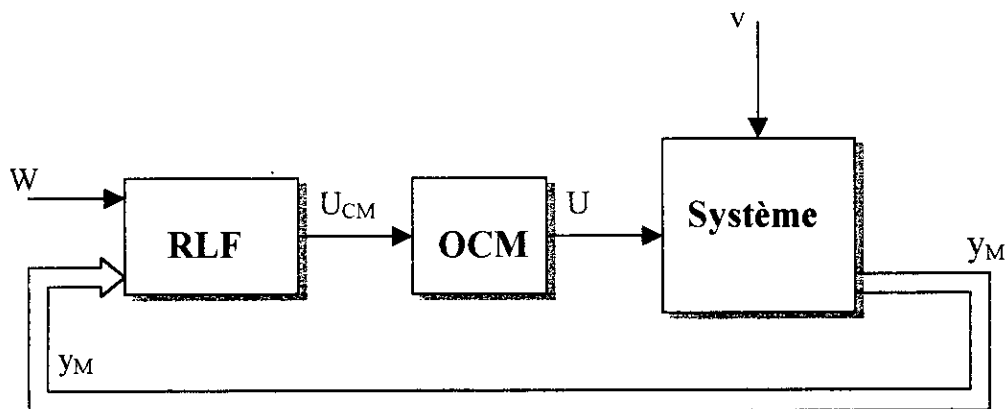


Figure (II.1) : structure d'un réglage par logique floue.

Comme dans les réglages conventionnels, il y a le système à régler et l'organe de commande OCM. Le régulateur par logique floue RLF fournit le signal de commande U_{CM} . Il reçoit à son entrée la grandeur de consigne w et une ou plusieurs grandeurs mesurées, réunies dans le vecteur y_M .

II.6.2 Configuration interne d'un régulateur par logique floue : [BUH 94]

La figure (II.2) montre la configuration interne d'un régulateur par logique floue. On peut distinguer trois parties :

- Fuzzification.
- Inférence.
- Défuzzification.

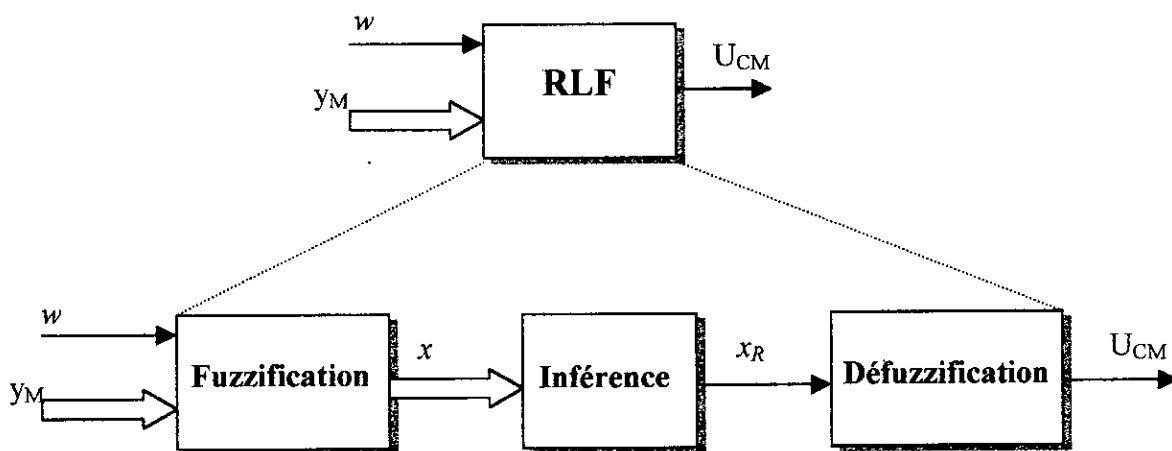


Figure (II.2) : Configuration interne d'un régulateur par logique floue.

II.6.2.1 Fuzzification

Le bloc fuzzification contient en général un traitement de données préliminaire, par exemple la formation de l'écart de réglage $e=w-y$ ou la détermination de la variation d'une certaine grandeur. Ces grandeurs sont alors traitées par des variables linguistiques, ce qui nécessite leur définition par des fonctions d'appartenance. La fuzzification fournit une série de variables floues, réunies par le vecteur x .

II.6.2.2 Inférence

Dans le bloc inférence, les valeurs des variables linguistiques sont liées par plusieurs règles qui doivent tenir compte du comportement statique et dynamique du système à régler ainsi que des buts de réglage envisagés. On obtient ainsi une information floue pour la variable de sortie x_R du régulateur.

Il existe plusieurs stratégies d'inférence.

Exemples :

MAMDANI

LARSEN

SUGENO

II.6.2.3 Défuzzification

Puisque l'organe de commande doit être attaqué avec une valeur bien précise pour le signal de commande U_{CM} , il faut transformer la valeur floue en une valeur déterminée. Cela se fait dans le bloc défuzzification en utilisant plusieurs stratégies dont les plus importantes sont :

1. Méthode du maximum :

Elle donne comme valeur celle qui a le plus grand degré d'appartenance.

2. Méthode de la moyenne des maxima :

Elle donne comme valeur la moyenne des valeurs dont le degré d'appartenance est maximal.

3. Méthode du centre de gravité :

C'est la méthode la plus utilisée, elle donne comme valeur le centre de gravité de l'ensemble flou.

$$Z_0 = \frac{\sum_{j=1}^n \mu_z(w_j) \cdot w_j}{\sum_{j=1}^n \mu_z(w_j)}$$

II.7 TYPES DE REGULATEURS FLOUS [LEE 90a]

Il existe plusieurs types de régulateurs, parmi lesquels on cite ceux de Mamdani, de Sugeno, de Larsen et de Tsukamoto, ceux de Mamdani et Sugeno sont les plus utilisés.

II.7.1 Régulateur flou de type Mamdani

Le régulateur de type Mamdani était proposé comme la première tentative à contrôler une combinaison de chaudière et machine à vapeur par un ensemble de règles linguistiques de contrôle obtenues d'opérateur humain expérimenté. Ce régulateur utilise des règles à prémisses et conclusions symboliques, l'inférence et la défuzzification.

II.7.2 Régulateur flou de type Sugeno

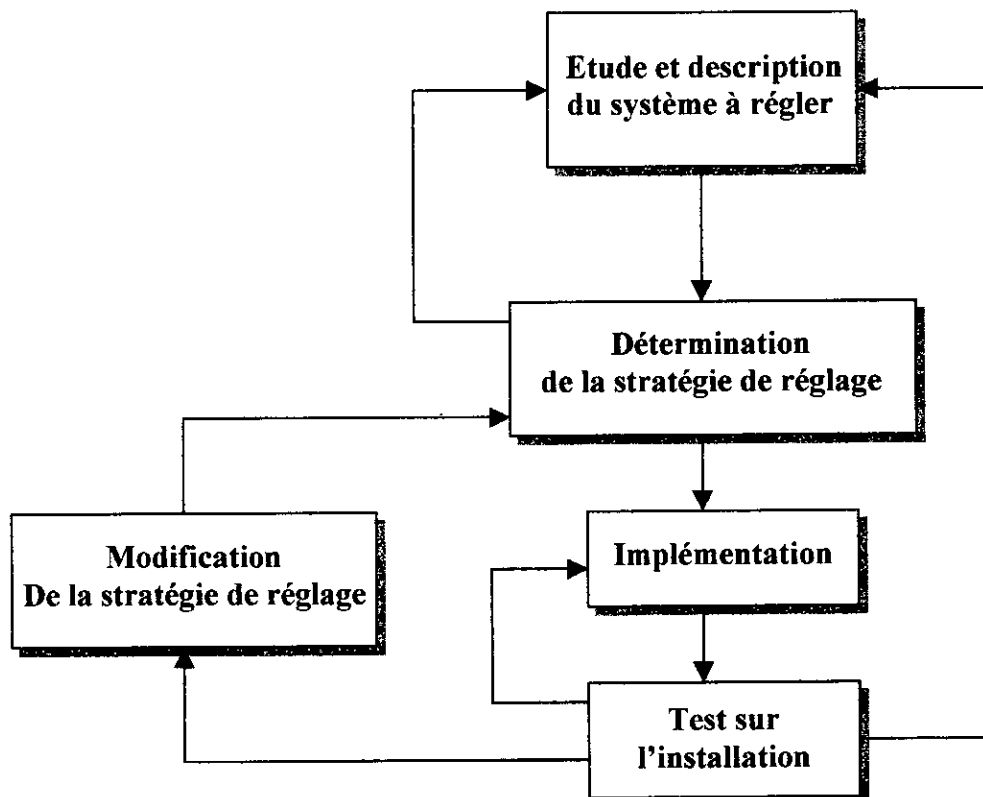
Le modèle de ce régulateur était proposé par Takagi, Sugeno et Kang dans un effort à développer une approche systématique à générer des règles floues à partir d'une contribution d'entrées sorties.

Une règle floue typique dans un régulateur de type Sugeno a la forme :

$$\text{Si } x \text{ est } A \text{ et } y \text{ est } B \text{ alors } z = f(x,y)$$

II.8 ETAPES POUR LA CONCEPTION D'UN REGLAGE PAR LOGIQUE FLOUE.

Le procédé à suivre lors de la conception d'un réglage par logique floue est assez différent de celui d'un réglage conventionnel. La figure (II.3) montre les étapes principales : [BUH 94]



figure(II.3) : Etapes lors de la conception d'un réglage par logique floue.

Il est à noter que dans le cas d'un réglage par logique floue, il n'est pas nécessaire d'établir un modèle. Si, pour un certain système à régler, il existe tout de même un modèle

Il est à noter que dans le cas d'un réglage par logique floue, il n'est pas nécessaire d'établir un modèle. Si, pour un certain système à régler, il existe tout de même un modèle mathématique convenable, on peut l'utiliser pour tester et modifier la stratégie de réglage à l'aide d'une simulation numérique. Cela facilite évidemment la mise en service sur l'installation réelle.

II.9 PROPRIETES D'UN REGLAGE PAR LOGIQUE FLOUE

- Mesure de plusieurs grandeurs : la grandeur à régler et d'autres grandeurs caractérisant le comportement dynamique du système à régler.
- Aptitude à régler convenablement surtout des systèmes à régler avec un comportement dynamique compliqué, dont la modélisation est difficile voir impossible.
- La richesse en approches de commande.

II.10 APPLICATION DE LA COMMANDE FLOUE AU ROBOT PUMA560

II.10.1 Décentralisation

L'application de la commande floue à un système complexe nécessite d'abord de choisir entre la commande centralisée et la commande décentralisée selon les critères suivants :

- La réalisabilité de la commande.
- Son temps de calcul.

Si on opte pour la commande centralisée, on constate par le tableau (II.1) qu'elle comporte un nombre de règles très élevé, ce qui nécessite une grande connaissance du système, et cela n'est pas toujours évident. Le temps de calcul de cette commande est aussi grand, idem pour la taille du régulateur résultant ce qui rend sa réalisation pratique difficile.

Mais si on choisit la commande décentralisée, on palliera au problème du nombre des règles à utiliser, on réduira le temps de calcul de la commande, et le régulateur complexe du cas centralisé se simplifiera en trois régulateurs simples, car dans la commande décentralisée le système global est décomposé en trois sous-systèmes. Chaque sous-système est commandé indépendamment des autres, les interconnexions sont considérées comme étant des perturbations.

| Nombre de classe d'appartenance | Nombre de règles floues | |
|------------------------------------|-------------------------|------------------|
| | Cas centralisé | Cas décentralisé |
| 3 | 729 | 27 |
| 5 | 15625 | 75 |
| 7 | 117649 | 147 |

Tableau (II.1) : nombre de règles, cas centralisé, cas décentralisé.

Donc on appliquera au robot *PUMA560* une commande décentralisée, c'est à dire que chaque articulation sera commandée comme étant un système indépendant des autres. Les interconnexions ainsi que l'effet des commandes des autres articulations seront considérés par le régulateur comme des perturbations externes.

Remarque :

La synthèse de la commande floue ne nécessite pas un modèle du système à régler, mais il est impératif de simuler la commande sur un modèle avant son implémentation.

II.10.2 Commande floue du robot

Sur la figure (II.4) sont présentés les trois régulateurs flous des trois articulations du robot PUMA560, ces derniers sont indépendants l'un de l'autre, chacun à deux entrées qui sont l'erreur et la variation de l'erreur et une sortie qui est la commande à appliquer à l'articulation.

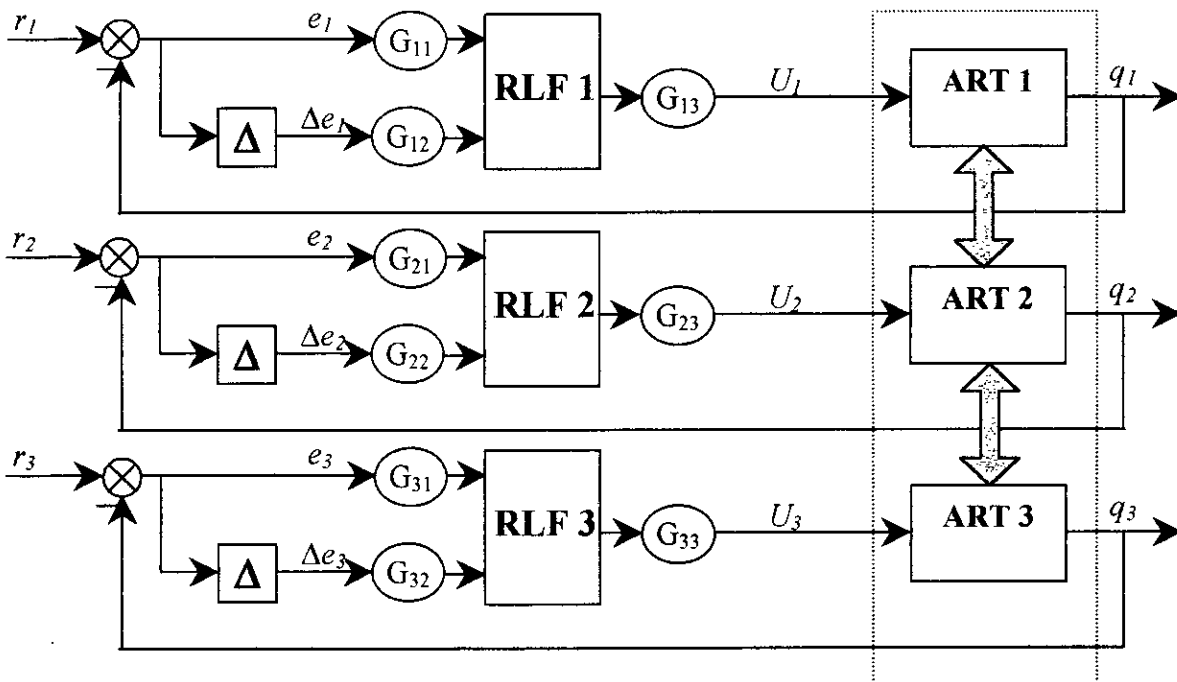


Figure (II.4) : structure de la commande floue du robot

II.10.2.1 Régulateur flou de MAMDANI [LOU 97]

Le schéma du régulateur de type de MAMDANI est composé :

➤ du contrôleur flou comprenant :

- un bloc de calcul de la variation de l'erreur (Δe) au cours du temps.
- les facteurs d'échelle ou gains associés à l'erreur (e), à sa variation (Δe), et à la commande (U).
- un bloc de fuzzification de l'erreur et de sa variation.
- une base de règles de contrôle flou.
- une logique floue utilisée pour l'évaluation des règles de contrôle flou (inférence), pour notre cas c'est la méthode somme-prod qui est utilisée.
- un bloc de défuzzification servant à convertir la commande floue en une valeur numérique.

➤ du processus à contrôler.

II.10.2.2 Loi de commande

La loi adoptée est fonction des entrées choisies pour notre contrôleur, en l'occurrence l'erreur et sa variation sur une période d'échantillonnage T_e :

$$U(kT_e) = f(e(kT_e), \Delta e(kT_e))$$

Par conséquent, l'activation de l'ensemble des règles de décision associées donne la commande U nécessaire et qui constitue donc la sortie de notre contrôleur.

Cette commande est obtenue par une simple lecture d'une table de décision prédéfini hors ligne. Notre choix de la loi de commande s'est porté sur la forme suivante :

$$U(kT_e) = G_u \cdot u_n(kT_e) \quad (\text{II.11})$$

Où G_u est le gain associé à la commande.

II.10.2.3 La matrice des règles de Macvicar et Whelan [TZA 90]

Après que Mamdani[MAM 74] a présenté la technique de réglage par logique floue, en se basant sur des règles obtenues auprès d'un expert, Macvicar et Whelan, ont fait une analyse sur ces bases de règles et proposé une matrice de règles à deux entrées, l'erreur et sa variation en se basant sur les deux principes suivants :

- Si la sortie est égale à la valeur désirée, et la variation de l'erreur est nulle, la commande sera maintenue constante.
- Si la sortie diverge de la valeur désirée, l'action sera dépendante du signe et de la valeur de l'erreur et de sa variation. Si les conditions sont telles que l'erreur peut être corrigée par elle-même alors la commande sera maintenue. Dans le cas contraire la commande sera changée pour avoir des résultats satisfaisants.

| $e \backslash \Delta e$ | NB | NS | ZE | PS | PB |
|-------------------------|----|----|----|----|----|
| NB | NB | NB | NB | NS | ZE |
| NS | NB | NB | NS | ZE | PS |
| ZE | NB | NS | ZE | PS | PB |
| PS | NS | ZE | PS | PB | PB |
| PB | ZE | PS | PB | PB | PB |

Tableau (II.2) : matrice de Macvicar-Whelan.

NB : négative big
NS : négative small
ZE : zero
PS : positive small
PB : positive big

II.10.2.4 Les gains de normalisation

Les gains de normalisation de l'erreur et de la variation de l'erreur ainsi que les gains associés aux commandes jouent un rôle extrêmement important. En effet, ce sont ces derniers qui fixeront les performances de la commande.

Il n'y a aucune méthode systématique qui donne ces paramètres d'emblée ; en fait, il faut procéder par tâtonnement en utilisant des règles empiriques et l'expérience acquise au fil du temps.

II.10.2.5 La trajectoire test

Pour tester les performances des commandes appliquées au robot, on va lui imposer une trajectoire de référence, c'est la trajectoire de LEAHVY qui excite toute sa dynamique et qui consiste à faire varier dans un intervalle de temps de 1.5 secondes la position de :

- ◆ La première articulation entre -50 degrés et 45 degrés.
- ◆ La deuxième articulation entre -135 degrés et -85 degrés.
- ◆ La troisième articulation entre 135 degrés et 30 degrés.

Les figures (II.5.a), (II.5.b), (II.5.c) représentent les trajectoires de référence pour la première, la deuxième et la troisième articulation respectivement.

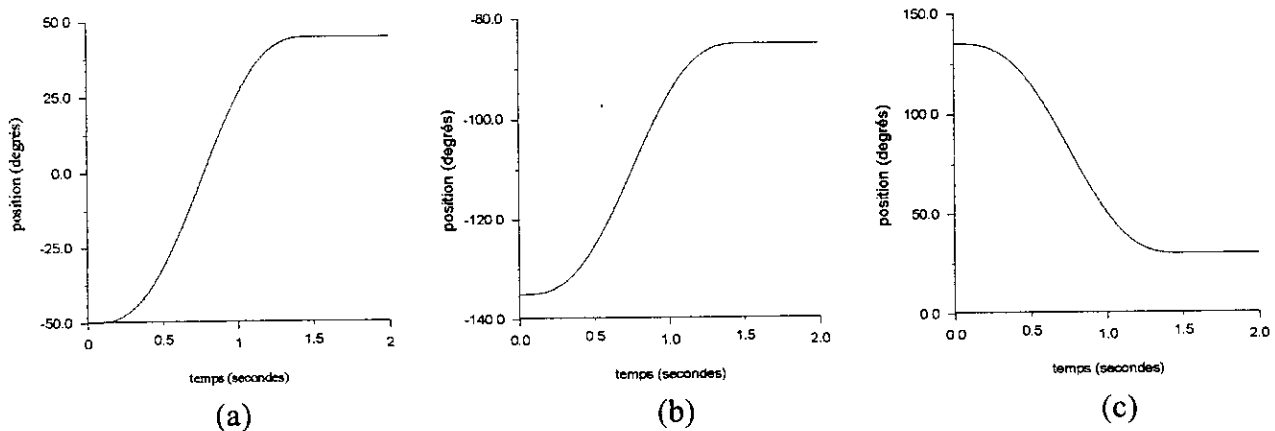


Figure (II.5) : trajectoires de référence

II.10.2.6 Synthèse des régulateurs

La base des règles est obtenue à partir de la matrice de Macvicar-whelan, les régulateurs des trois sous-systèmes se différencient seulement par les gains de normalisation et les gains associés aux commandes, les fonctions d'appartenance sont triangulaires pour les entrées et les sorties.

II.10.3 Régulateur à trois classes

◆ La base des règles :

| | | | |
|-------------------------|----------|----------|----------|
| $e \backslash \Delta e$ | N | Z | P |
| N | N | N | Z |
| Z | N | Z | P |
| P | Z | P | P |

Tableau (II.3) : Base de règles. RLF (3×3)
(N : négative, Z : zéro, P : positive)

◆ Fonctions d'appartenance.

Les fonctions d'appartenance sont représentées par la figure (II.6)

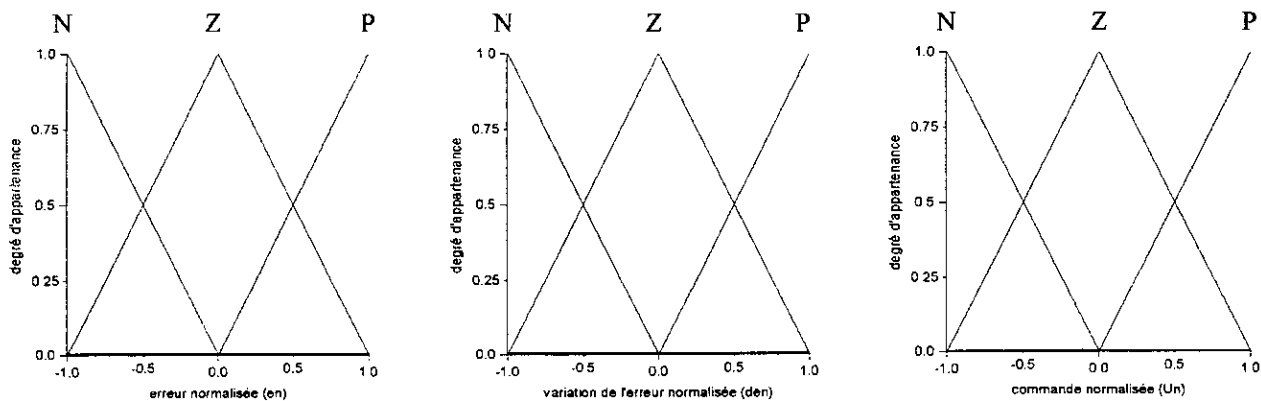


Figure (II.6) :fonctions d'appartenance RLF à trois classes

II.10.4 Régulateur flou à cinq classes

◆ La base des règles :

| $e \backslash \Delta e$ | NB | NS | ZE | PS | PB |
|-------------------------|----|----|----|----|----|
| NB | NB | NB | NB | NS | ZE |
| NS | NB | NB | NS | ZE | PB |
| ZE | NB | NS | ZE | PS | PB |
| PS | NS | ZE | PS | PB | PB |
| PB | ZE | PS | PB | PB | PB |

Tableau (II.4) : Base de règles RLF (5×5)

◆ Fonctions d'appartenance :

Elles ont des formes triangulaires, et sont représentées par la figure (II.7)

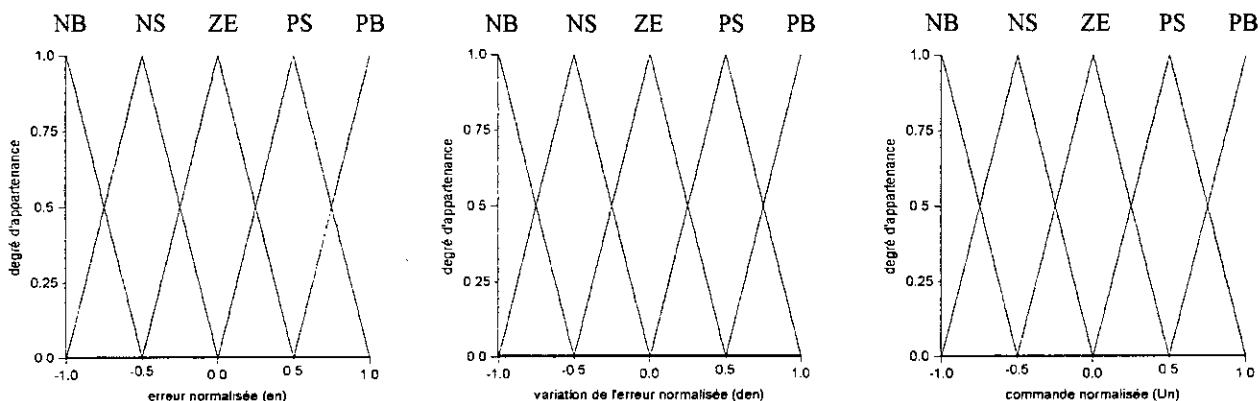


Figure (II.7) : fonctions d'appartenance RLF à 5 classes

II.10.5 Régulateur flou à sept classes

◆ Base de règles

| $e \backslash \Delta e$ | NB | NM | NS | ZE | PS | PM | PB |
|-------------------------|----|----|----|----|----|----|----|
| NB | NB | NB | NB | NB | NM | NS | ZE |
| NM | NB | NB | NB | NM | NS | ZE | PS |
| NS | NB | NB | NM | NS | ZE | PS | PM |
| ZE | NB | NM | NS | ZE | PS | PM | PB |
| PS | NM | NS | ZE | PS | PM | PB | PB |
| PM | NS | ZE | PS | PM | PB | PB | PB |
| PB | ZE | PS | PM | PB | PB | PB | PB |

Tableau (II.5) :base de règles
RLF 7×7

◆ Fonctions d'appartenance :

Elles ont la forme triangulaire, elles sont représentées par la figure (II.8) :

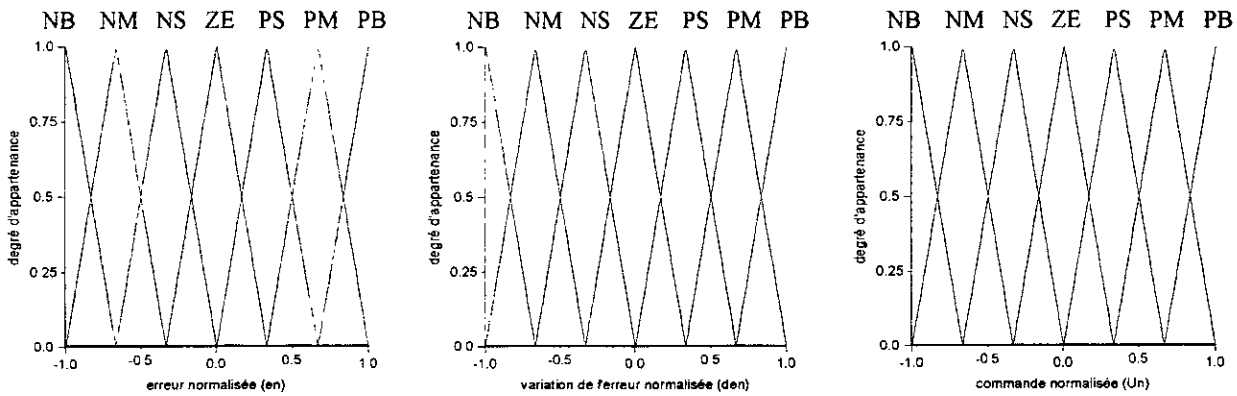


Figure (II.8) :fonctions d'appartenance
RLF à sept classes

II.10.6 Résultats de simulations

Les simulations des lois de commande ont été réalisées en langage FORTRAN, par l'algorithme de résolution Runge-Kutta d'ordre 4, avec une période d'échantillonnage égale à 0.01 secondes.

Un ensemble d'essais nous a permis de trouver les gains de normalisation et les gains des commandes assurant une dynamique globale satisfaisante.

◆ Analyse des performances

Les figures de (II.9) à (II.11) représentent les réponses des trois articulations, pour les trois régulateurs flous (trois, cinq et sept classes d'appartenance.) pour la trajectoire de LEAHVY à vide.

En examinant les figures nous remarquons que:

- Les articulations du robot suivent fermement les trajectoires désirées, avec une valeur maximale de l'erreur qui n'atteint pas 0.01 degrés.
- Les commandes sont lisses, et réalisables.
- Une relation quasi-linéaire existe entre l'erreur et la commande.

◆ Tests de robustesse

Les tests suivants ont été faits pour tester la robustesse des régulateurs :

• Test de la charge :

C'est de faire démarrer le robot avec une charge qui sera lâchée avant qu'il achève son parcours. Pour notre test, nous avons utilisé deux masses respectives de 10 et 4 kg, qui seront lâchées après 0.75 secondes du démarrage.

• Rupture de la commande :

Ce test consiste à simuler une défaillance d'un des trois régulateurs qui commande le robot, qui intervient pendant son mouvement. Le régulateur que nous avons choisi est celui qui commande la deuxième articulation. La défaillance se produit à $t=0.75$ secondes.

Les résultats des tests appliqués aux différentes articulations, sont représentés sur les figures de (II.12) à (II.20).

Pour le premier test, on peut remarquer que les régulateurs fournissent un couple plus grand pour compenser la charge. La chute de cette dernière pendant le mouvement n'influe pas sur le système, grâce à la rapidité de réponse des régulateurs.

Nous remarquons pour les régulateurs à 5 et à 7 classes, dans le test de 10 kg, l'apparition au démarrage, de petites oscillations dans la commande. Ces oscillations sont rapidement atténuées.

Le deuxième test nous a permis de valider la décentralisation de la commande. En effet malgré une défaillance du deuxième régulateur, qui a causé la divergence de la deuxième articulation, les deux autres suivent leurs références. Cela illustre l'avantage de commande décentralisée vis à vis de la commande centralisée, car si nous avions eu cette défaillance dans un régulateur centralisé le comportement du robot ne serait pas maîtrisé.

◆ Etude comparative

Les régulateurs présentés ont donné de bonnes performances, mais cela n'est pas le seul critère pour la désignation d'un régulateur, il faut voir aussi sa simplicité, et sa réalisabilité.

En examinant les résultats déjà présentés, ainsi que la comparaison entre les trois régulateurs présentée dans la figure (II.21), on tire les remarques suivantes :

- Le régulateur à trois classes d'appartenance donne des résultats meilleurs que celles donnés par les régulateurs à 5 et à 7 classes d'appartenance.
- Ces derniers ont des résultats identiques.

Donc, on conclut que :

- L'augmentation de nombre de classes ne garantit pas toujours l'amélioration des performances de la commande.
- Le régulateur à trois classes d'appartenance est le mieux à adopter pour ses performances et sa simplicité.

II.11 CONCLUSION

Dans ce chapitre nous avons présenté la théorie de la logique floue et les différentes démarches de conception d'un régulateur flou. Cette commande présente des avantages vis à vis des méthodes de commande conventionnelles, notamment :

- La simplicité de conception.
- La richesse en approches.
- Elle ne nécessite pas de modèle paramétrique mathématique.
- Le vaste domaine d'application.

L'application de cette commande sur un robot *PUMA560* a permis de mettre en évidence ces bonnes performances et sa robustesse vis à vis des différentes perturbations. Plusieurs régulateurs ont été utilisés, celui de trois classes d'appartenance est retenu pour sa simplicité et ses meilleures performances.

L'inconvénient de cette commande réside dans le tâtonnement des gains de normalisation et des gains associés à la commande. Cela n'empêche pas que la commande floue est une alternative assez valable des méthodes conventionnelles de commandes.

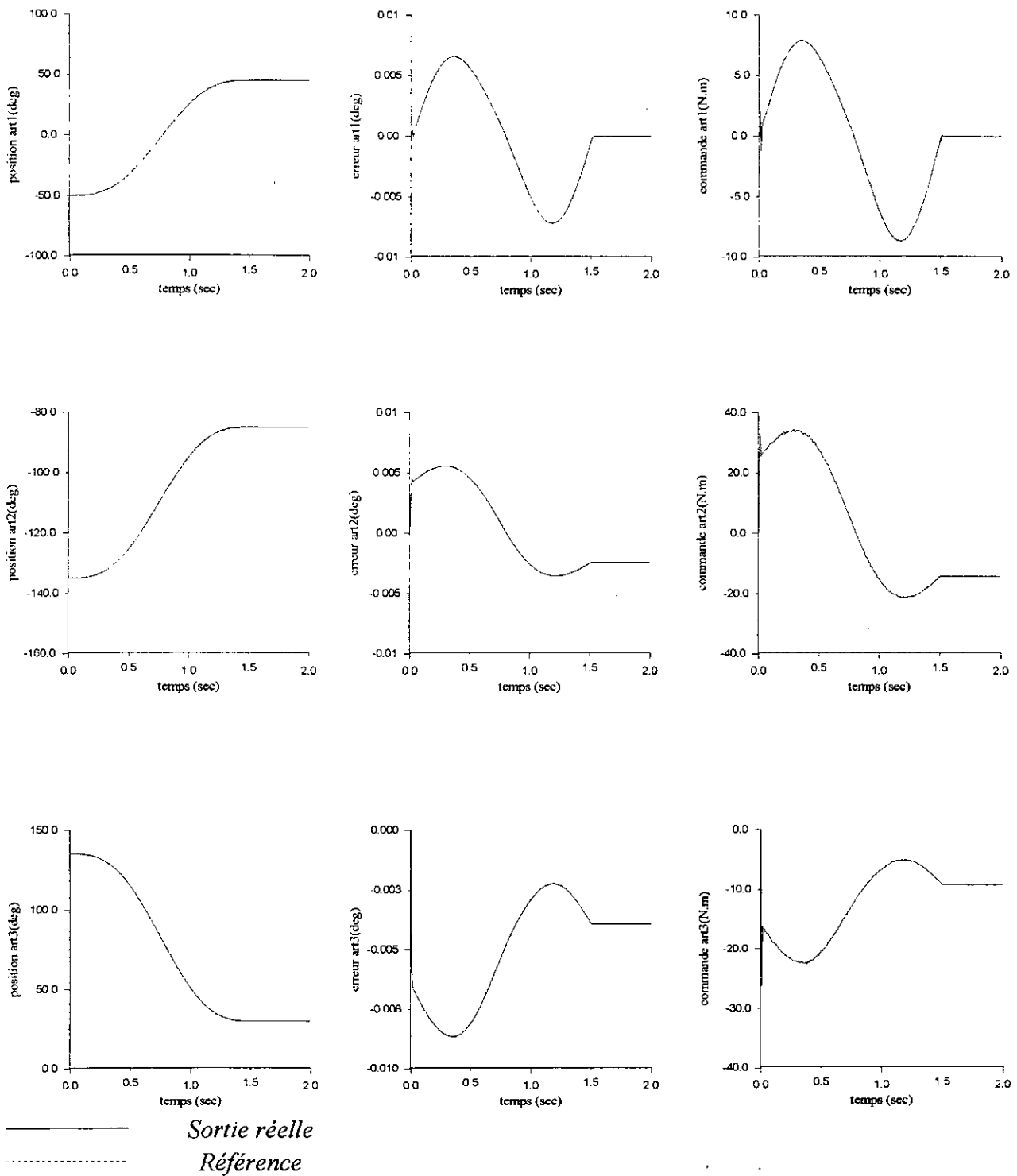


Figure (II.9) : Réponse du robot pour une poursuite à vide.
Régulateur flou à trois classes d'appartenance

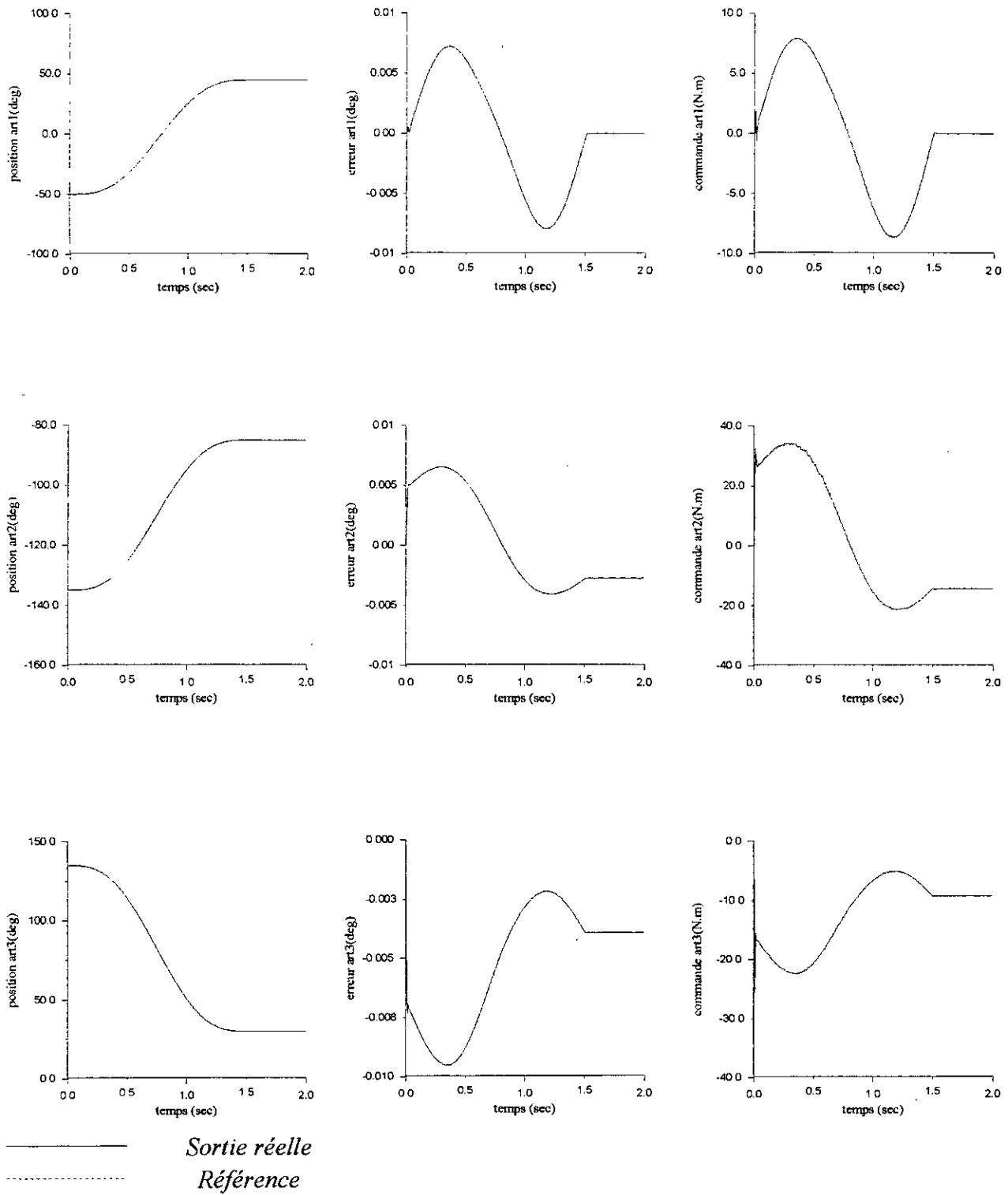


Figure (II.10) : Réponse du robot pour une poursuite à vide.
 Régulateur flou à cinq classes d'appartenance

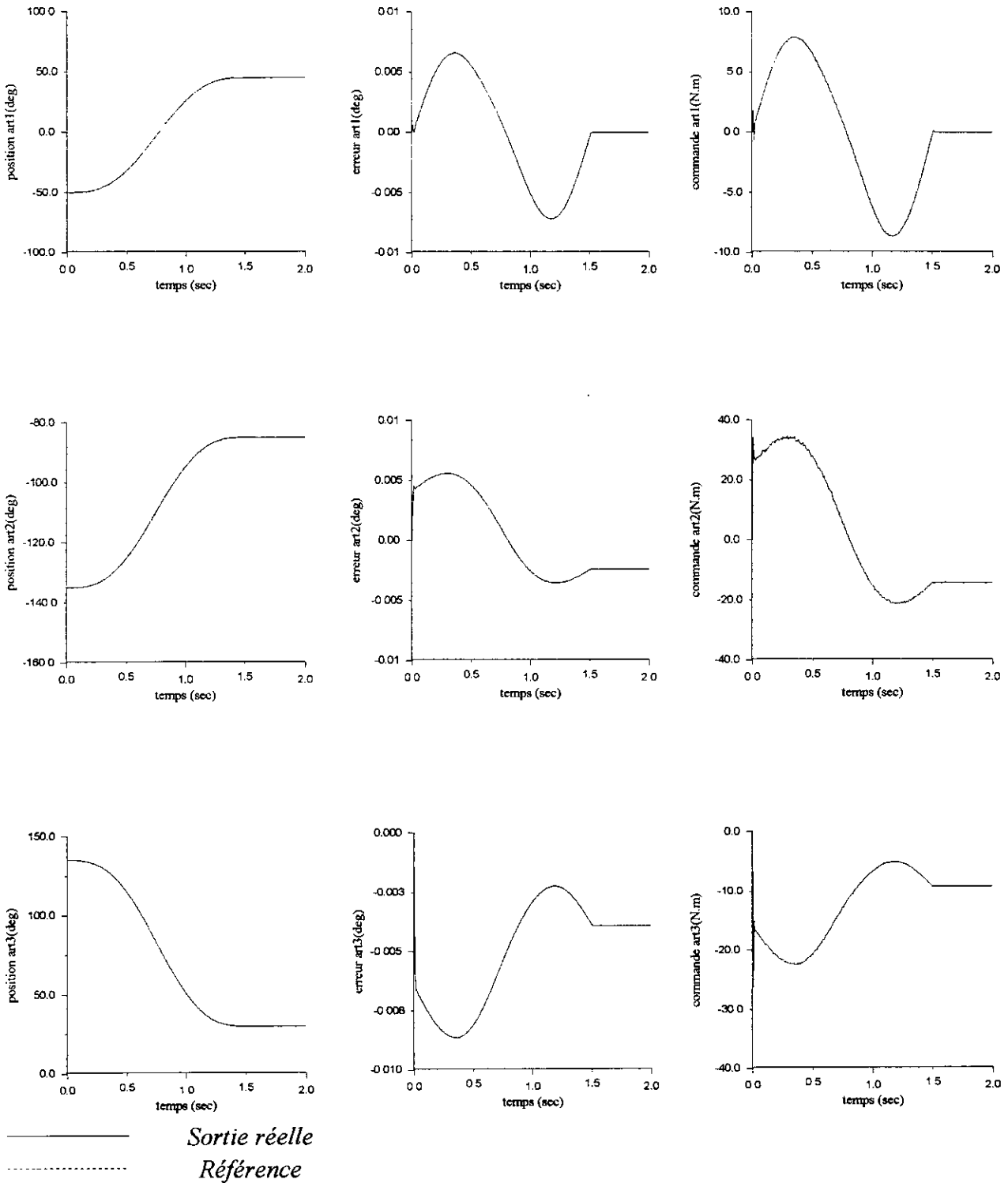


Figure (II.11) : Réponse du robot pour une poursuite à vide.
 Régulateur flou à sept classes d'appartenance

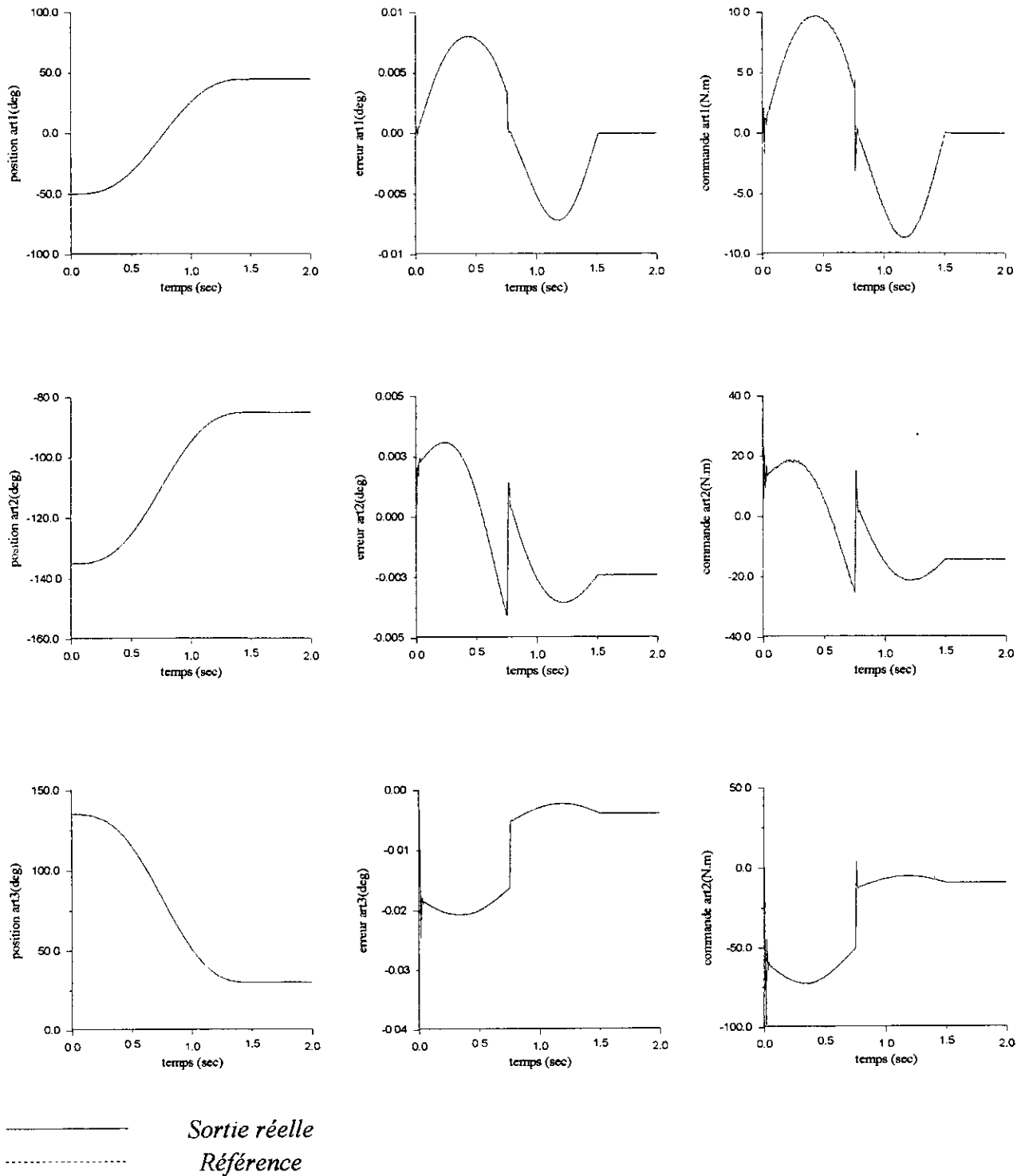
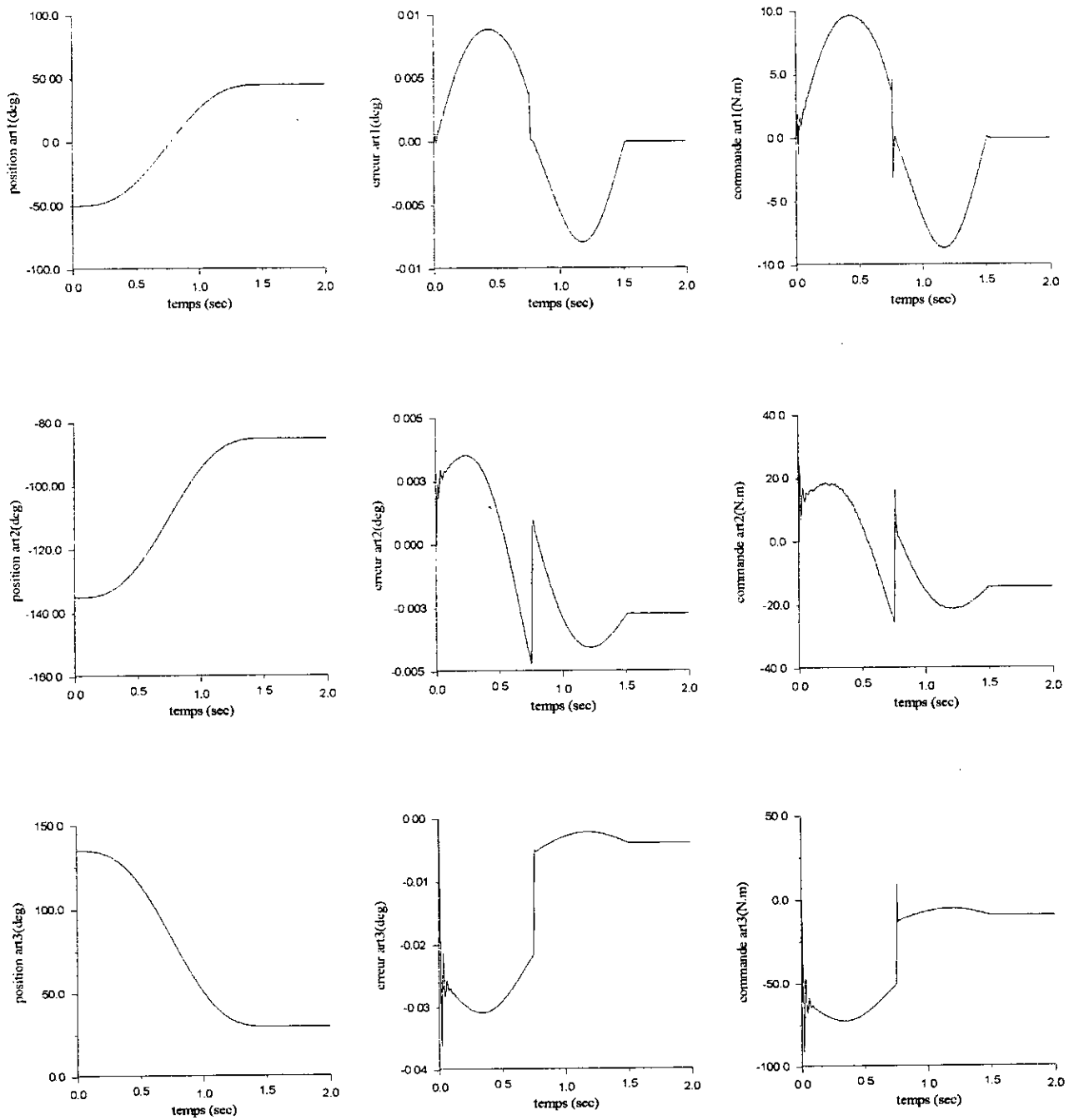


Figure (II.12) : Réponse du robot pour un test de la charge (10kg)
 Régulateur flou à trois classes



————— *Sortie réelle*
 - - - - - *Référence*

Figure (II.13) : Réponse du robot pour un test de la charge (10kg)
 Régulateur flou à cinq classes

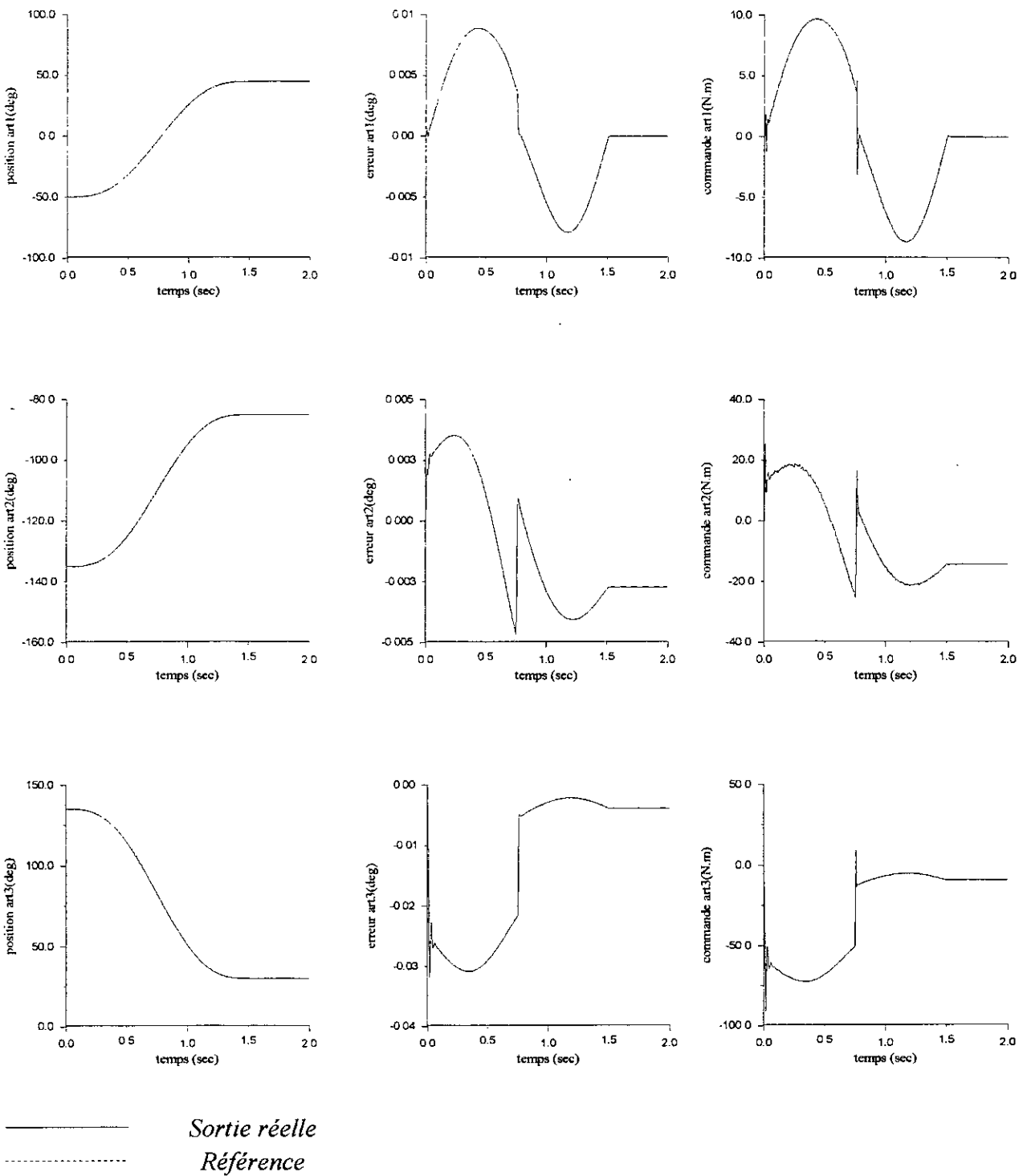


Figure (II.14) : Réponse du robot pour un test de la charge (10kg)
Régulateur flou à sept classes

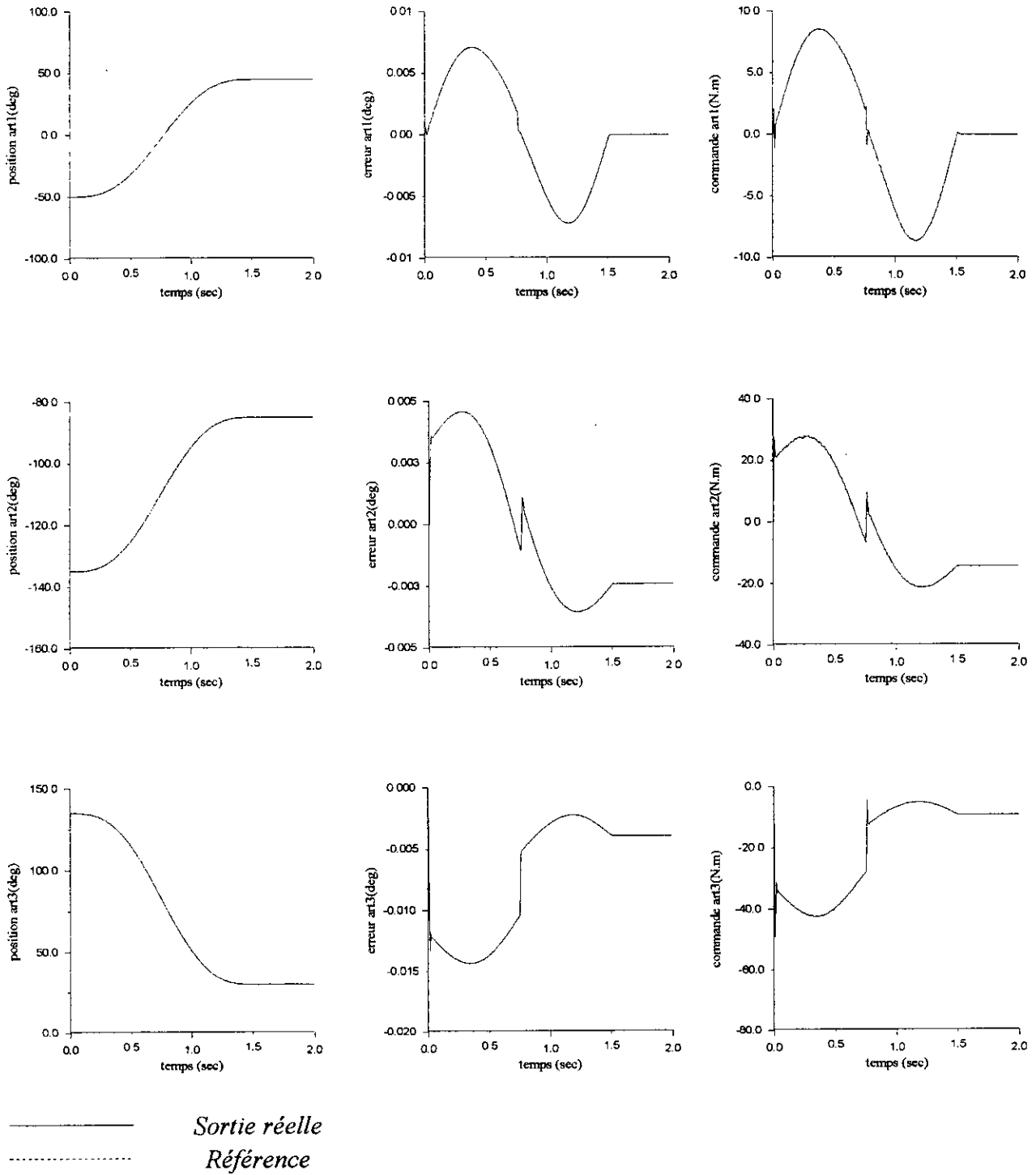


Figure (II.15) : Réponse du robot pour un test de la charge (4kg)
 Régulateur flou à trois classes

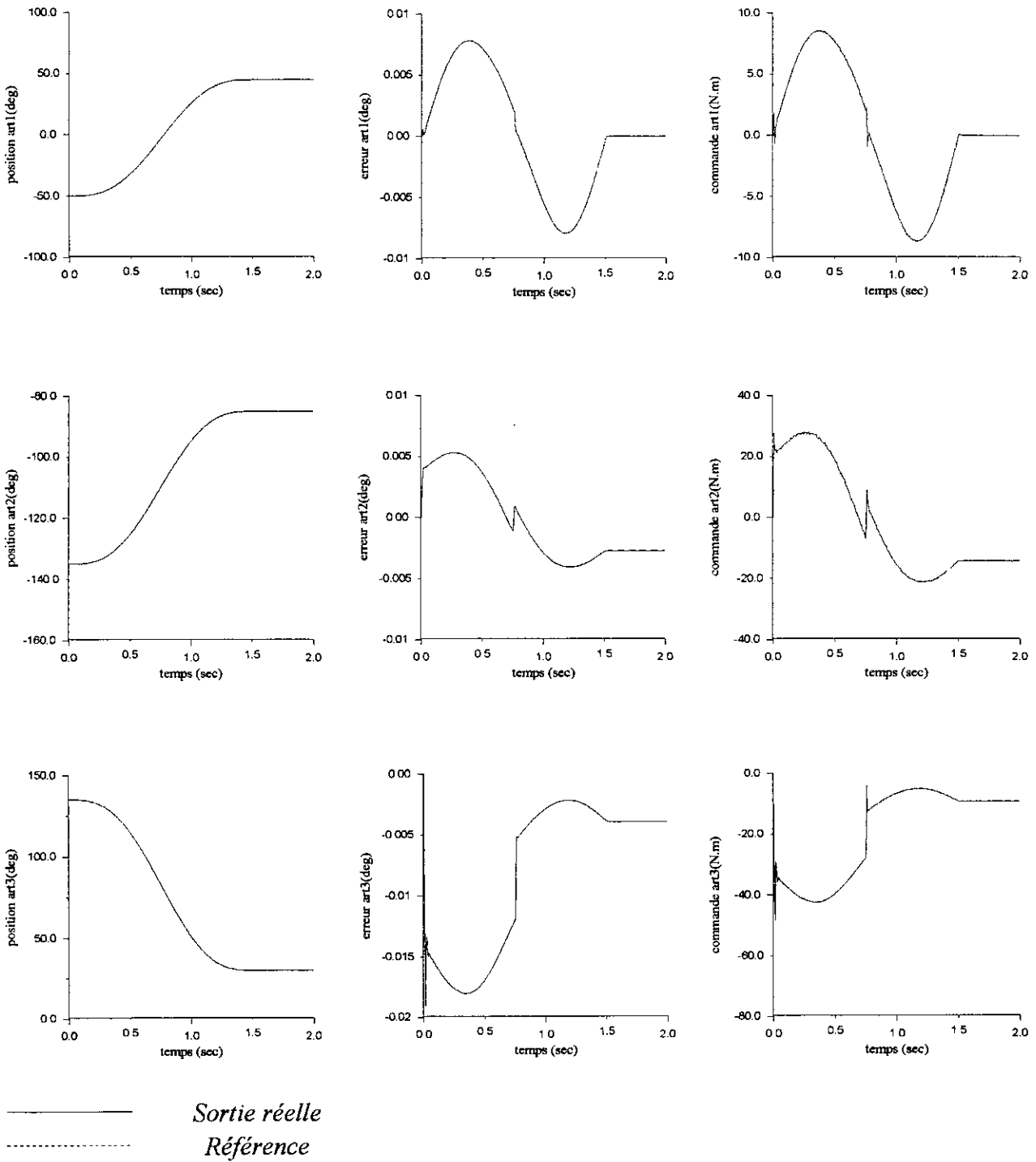


Figure (II.16) : Réponse du robot pour un test de la charge (4kg)
 Régulateur flou à cinq classes

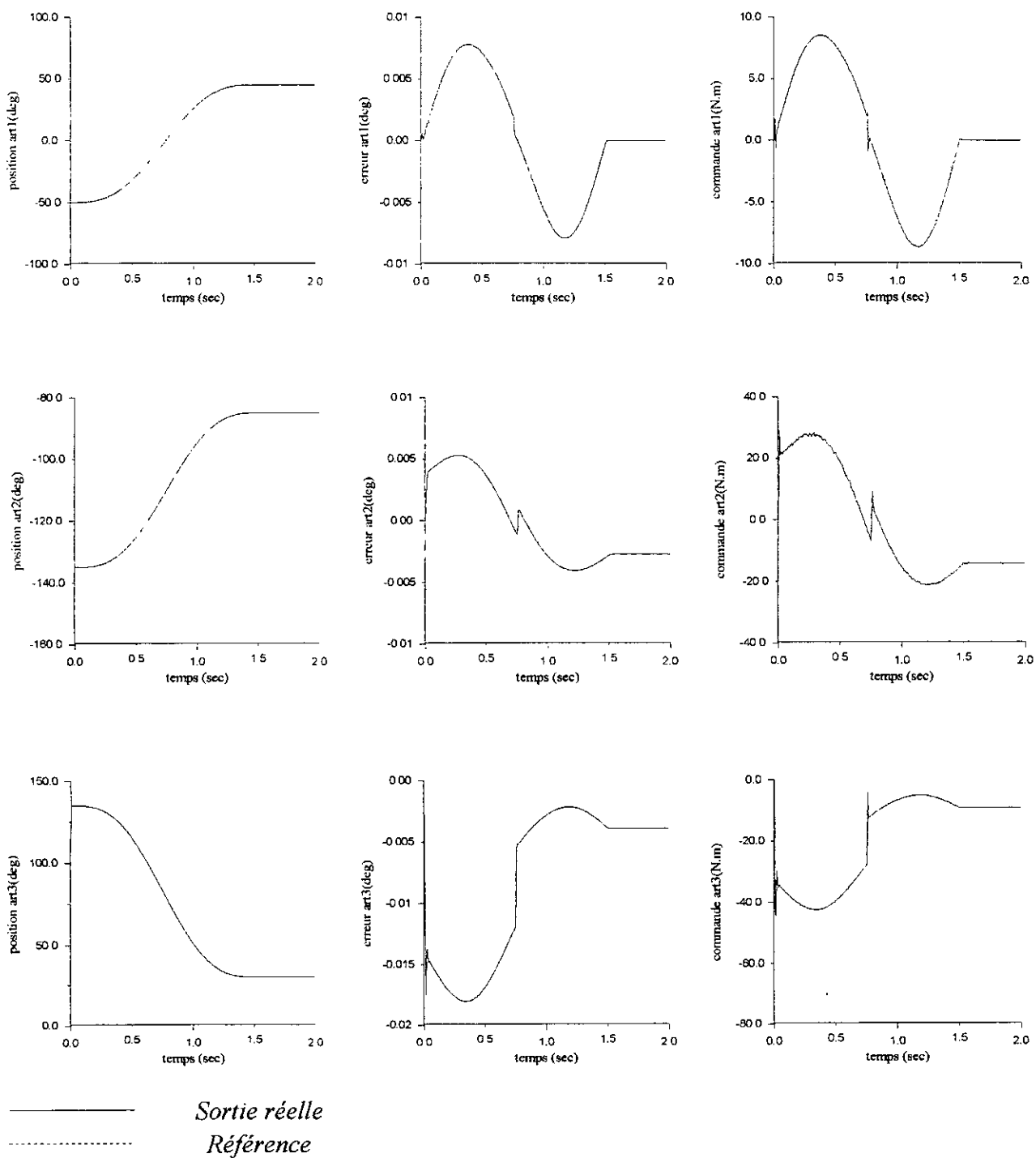


Figure (II.17) : Réponse du robot pour un test de la charge (4kg)
Régulateur flou à sept classes

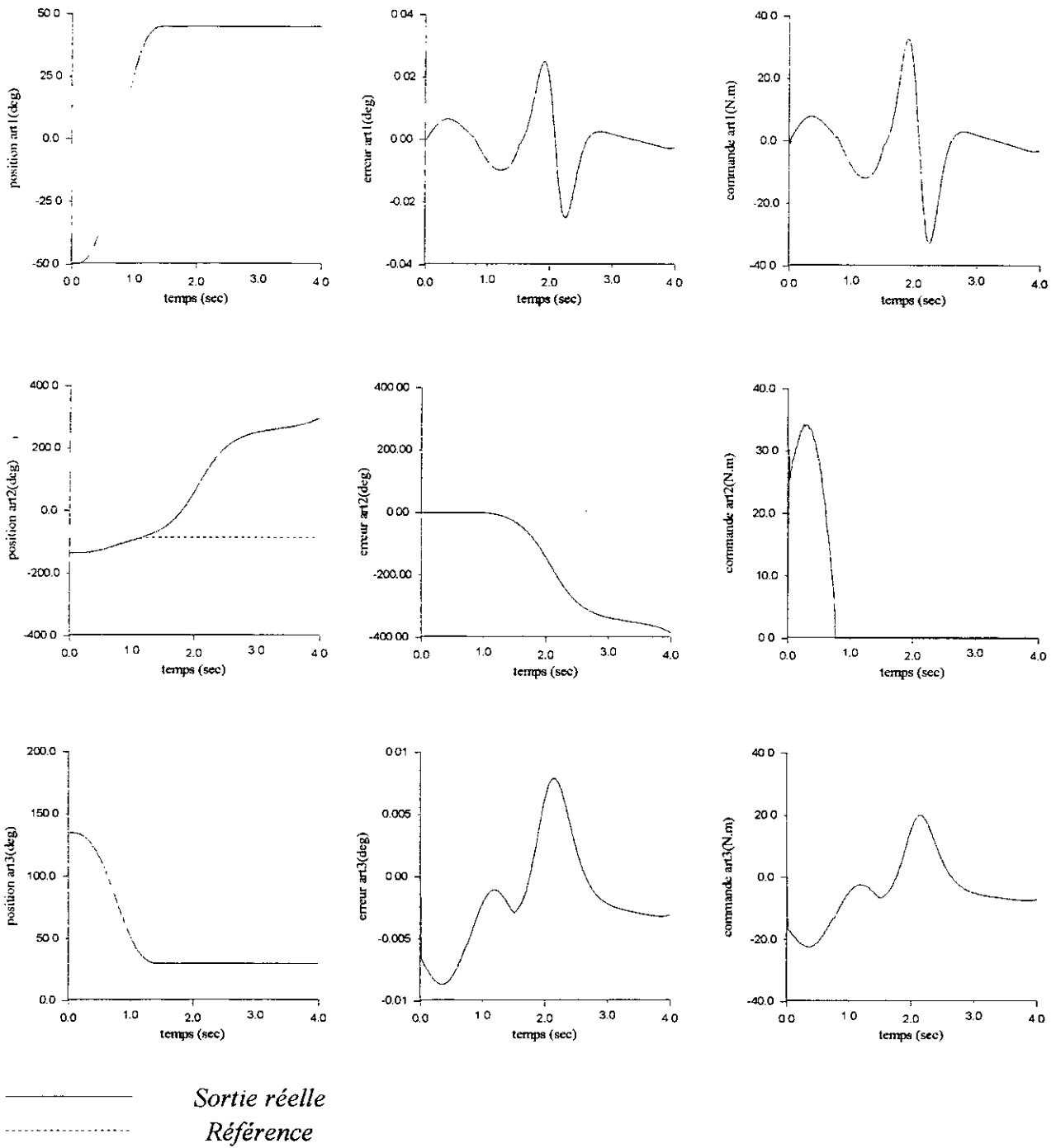


Figure (II.18) : Réponse du robot pour un test de rupture de la commande Régulateur flou à trois classes.

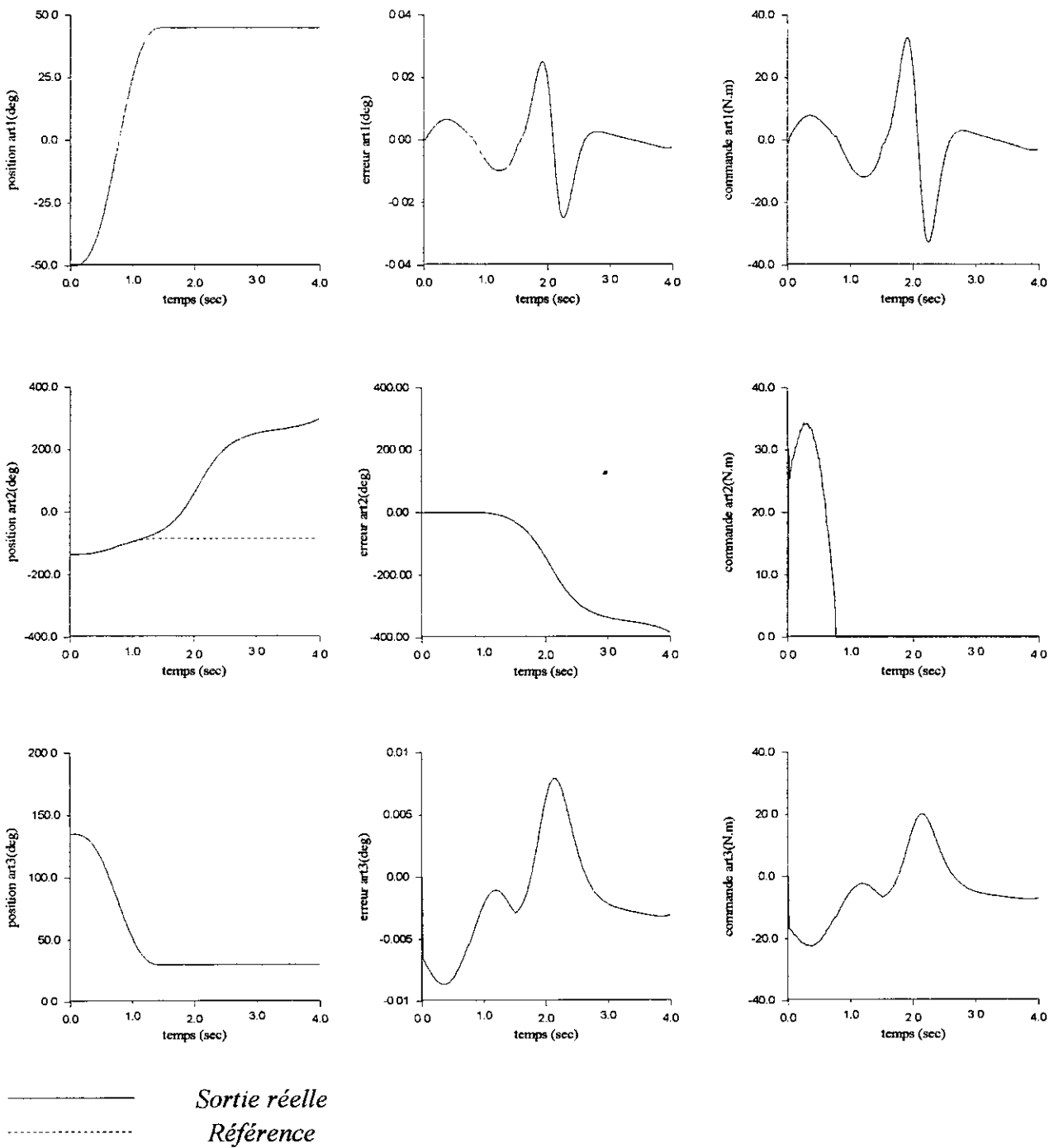


Figure (II.19) : Réponse du robot pour un test de rupture de la commande Régulateur flou à cinq classes.

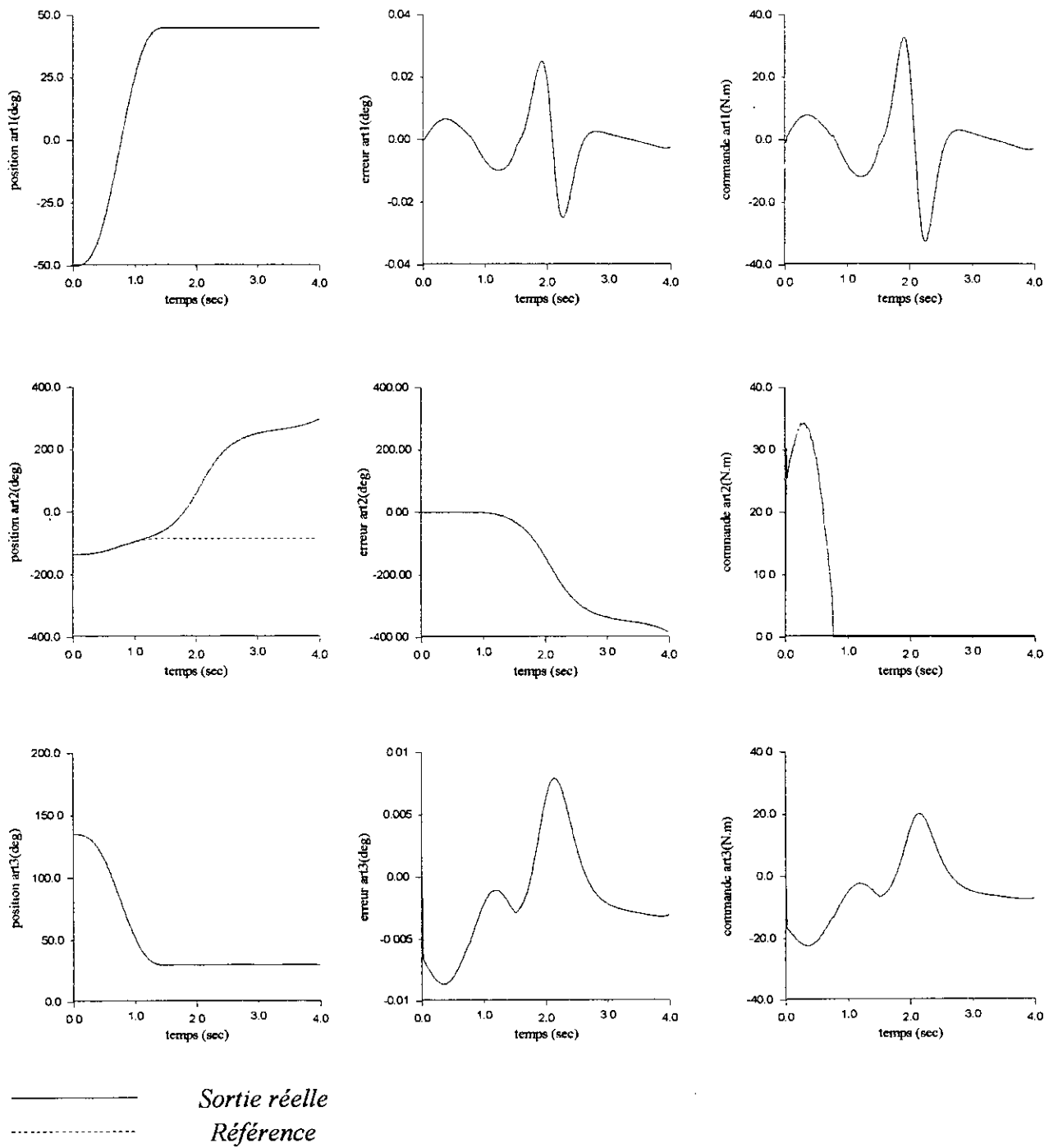
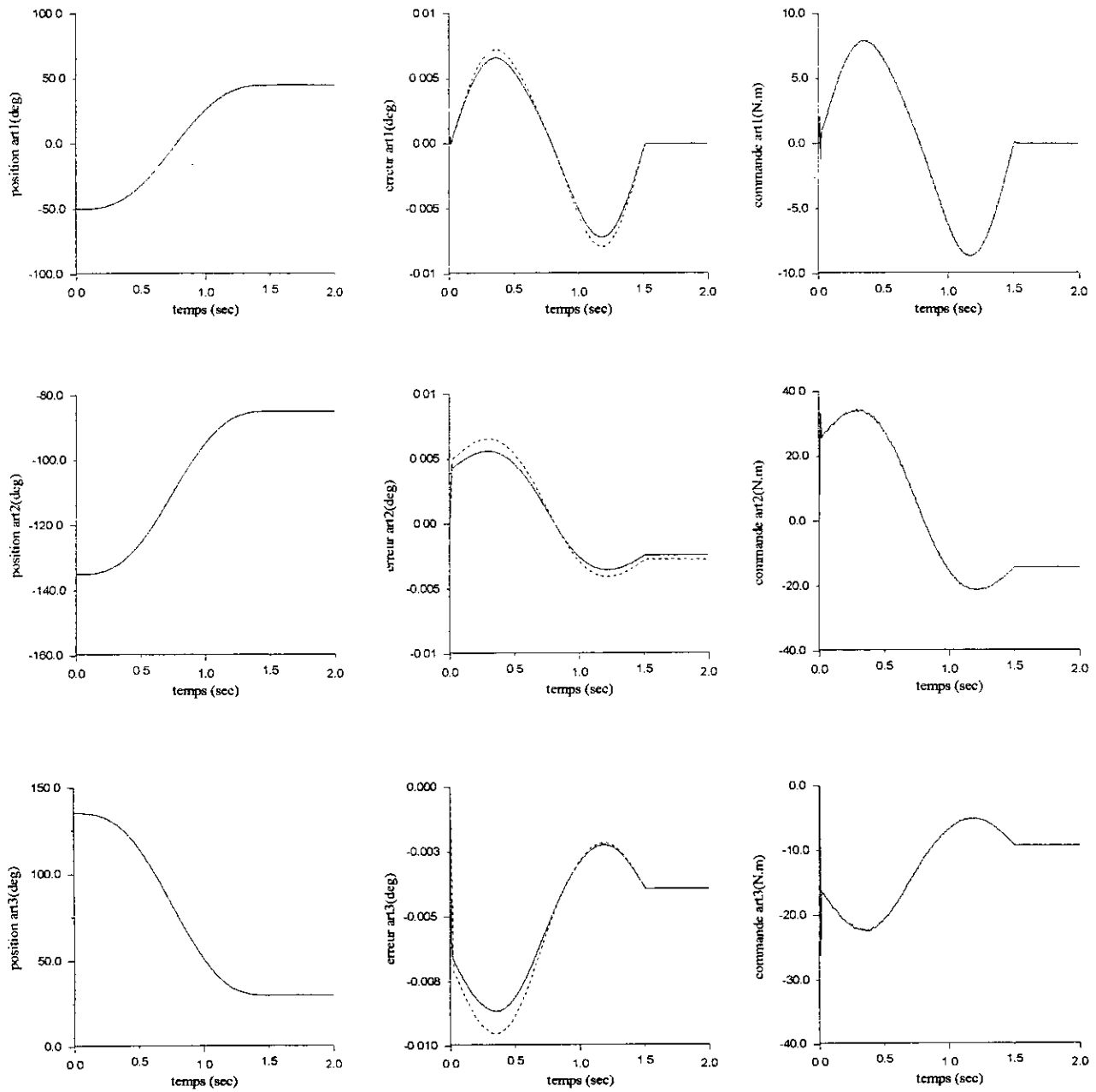


Figure (II.20) : Réponse du robot pour un test de rupture de la commande Régulateur flou à sept classes.



- RLF à 3classes
- - - - - RLF à 5classes
- RLF à 7classes

Figure (II.21) : comparaison des réponses du robot pour les trois régulateurs. (poursuite à vide)



CHAPITRE III

COMMANDE

NEURO-FLOUE

CHAPITRE III

COMMANDE NEURO-FLOUE

III.1 INTRODUCTION

Les réseaux neuronaux sont des dispositifs constitués d'un très grand nombre de processeurs simples identiques, appelés neurones artificiels et fortement interconnectés, qui fonctionnent en parallèle selon des architectures diverses, à l'instar des neurones du cerveau. Soumis à une stimulation externe, le réseau modifie spontanément l'ensemble de ses connexions neuronales jusqu'à un état stable. Après cette phase d'apprentissage, il est capable de reconnaître très rapidement des modèles en réagissant à des signaux identiques. [RAN 95]

Grâce à ce pouvoir d'apprentissage, les réseaux de neurones artificiels jouent un rôle très important dans la résolution des différents problèmes liés à la commande.

Actuellement la tendance des chercheurs est d'associer la commande par logique floue aux réseaux de neurones artificiels, afin de tirer profit des capacités d'apprentissage des RNA et de celles du raisonnement approché de la logique floue, et de minimiser les inconvénients de chaque méthode utilisée seule. Le résultat de cette association est appelé réseau neuro-flou.

III.2 ARCHITECTURE DES RESEAUX DE NEURONES ADAPTATIFS.[SHI 95]

Comme son nom l'indique un réseau de neurones adaptatif est une structure dont le comportement dépend des valeurs d'un nombre de paramètres variables. Plus spécifiquement un réseau de neurones adaptatif est composé d'un nombre de nœuds connectés par des liens, chaque nœud réalise une fonction et chaque lien spécifie la direction de l'information d'un nœud vers un autre.

Généralement la fonction du nœud est une fonction paramétrique dont les paramètres sont ajustables. En changeant ces paramètres on change la fonction du nœud ainsi que le comportement général du réseau adaptatif.

La figure (III.1) montre la structure typique d'un réseau de neurones adaptatif, c'est une représentation en couches.

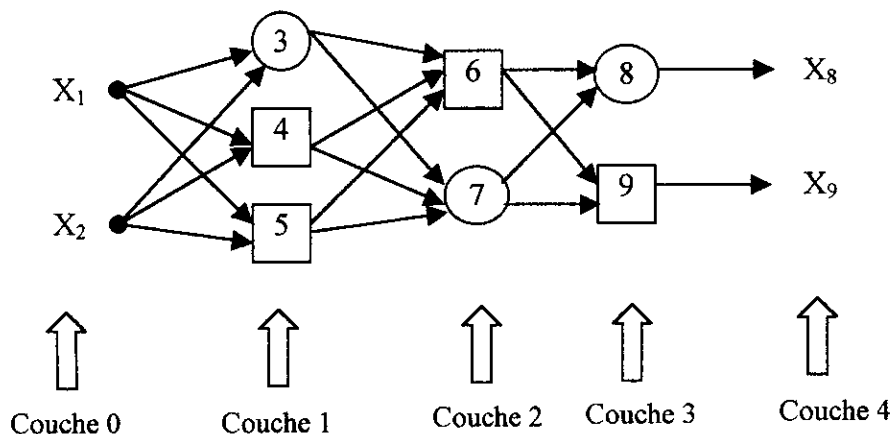


Figure (III.1): structure d'un réseau de neurones adaptatif

Les paramètres d'un réseau sont distribués entre ces nœuds, donc chaque nœud a des paramètres locaux, leur union définit les paramètres du réseau.

Si les paramètres d'un nœud sont variables on représente ce nœud par un carré, ou alors si les paramètres sont fixes on représente le nœud par un cercle. Généralement les réseaux adaptatifs sont classés en deux catégories selon le sens des liens entre les différents nœuds: *feedforward* et *récurrents*. la figure (III.1) est un réseau *feedforward*, la propagation de l'information se fait dans un seul sens uniquement (de gauche vers la droite), s'il existe un lien qui transmet l'information dans le sens inverse en formant un chemin circulaire dans le réseau alors le réseau est récurrent. La figure (III.2) en représente un exemple.

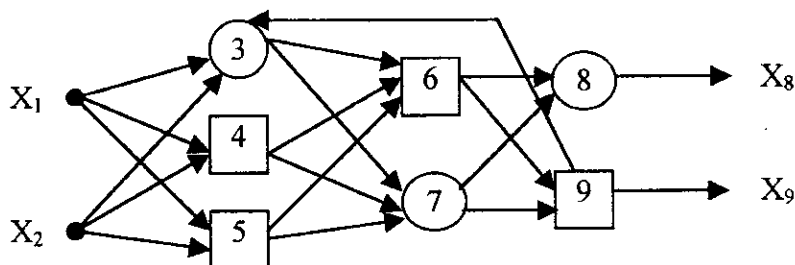


figure (III.2): réseau adaptatif récurrent

Autre représentation du réseau feedforward, c'est la représentation par ordre topologique qui ordonne les nœuds en séquences 1,2,..., en sorte qu'il n'y a pas de liens à partir du nœud i vers le nœud j tant que $i \geq j$, (figure (III.3)). Cette représentation facilite la formulation de l'algorithme d'apprentissage.

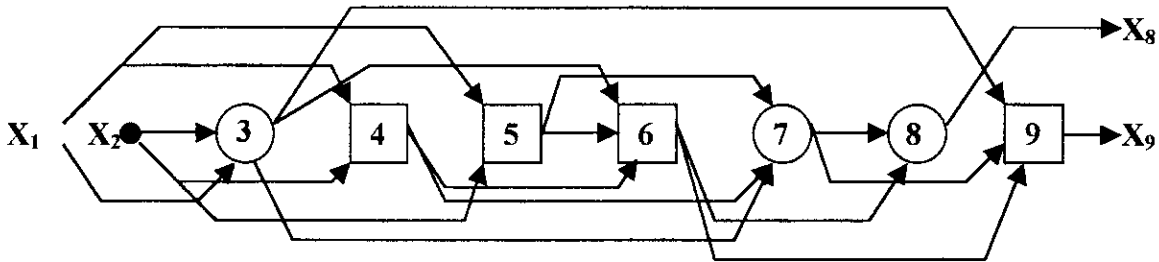


Figure (III.3) : représentation par ordre topologique.

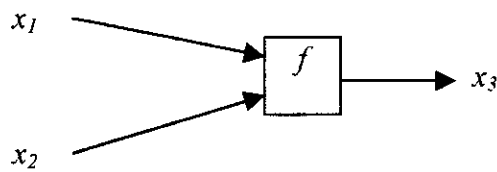
III.3 EXEMPLES DES RESEAUX ADAPTATIFS [SHI 95]

Les nœuds des réseaux adaptatifs réalisent des fonctions linéaires et non linéaires :

1. Soit la fonction suivante :

$$f(x_1, x_2, x_3, a_1, a_2, a_3) = a_1 x_1 + a_2 x_2 + a_3 \tag{III.1}$$

Cette fonction se représente comme suit :

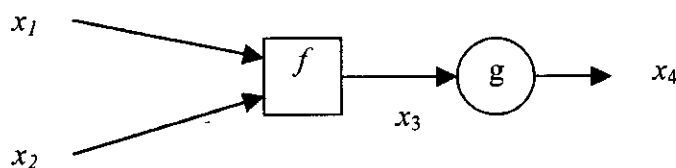


f est la fonction du nœud dont les paramètres sont a_1, a_2, a_3 .

Si on désire que la sortie du réseau ait seulement deux valeurs 0 et 1, on ajoute au nœud précédent un autre nœud dont la fonction est la suivante :

$$x_4 = g(x_3) = \begin{cases} 1 & \text{si } x_3 \geq 0 \\ 0 & \text{si } x_3 < 0 \end{cases} \tag{III.2}$$

Le réseau sera représenté comme suit :



La fonction g peut être non dérivable comme c'est le cas dans l'exemple précédent, ou non linéaire comme la fonction sigmoïde :

$$x_4 = g(x_3) = \frac{1}{1 + e^{-x_3}} \tag{III.3}$$

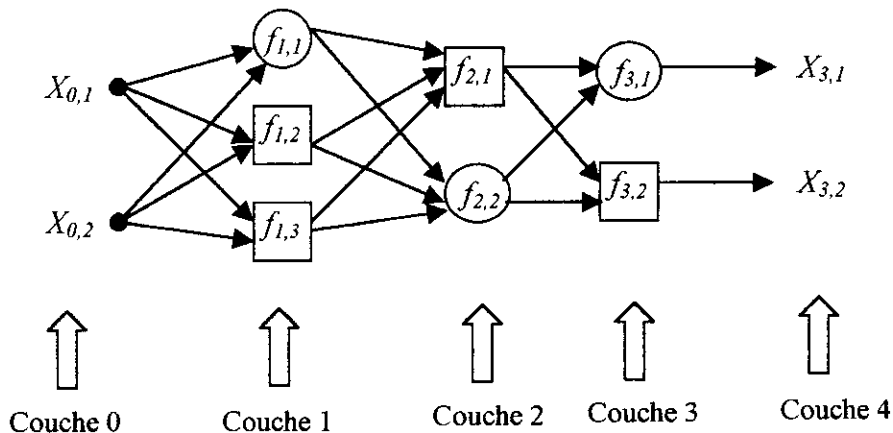
III.4 L'ALGORITHME DE BACK-PROPAGATION

L'algorithme de back-propagation est un algorithme d'apprentissage utilisé pour obtenir récursivement un vecteur dont chaque élément est défini comme la dérivée ordonnée d'une mesure d'erreur relativement à un paramètre. [SHI 95]

Supposant qu'on a un réseau feedforward adaptatif dans la représentation par couches à L couches chaque couche l a $N(l)$ nœuds. Donc les sorties et les fonctions du nœud i ($i=1, 2, \dots, N(l)$) de la couche l peuvent être représentées par $x_{l,i}$ et $f_{l,i}$ respectivement comme c'est montré sur la figure (III.4). l'expression générale de la fonction $f_{l,i}$ est :

$$x_{l,i} = f_{l,i}(x_{l-1,1}, \dots, x_{l-1,N(l-1)}, \alpha, \beta, \gamma, \dots) \tag{III.4}$$

Avec α, β, γ des paramètres internes du nœud.



Figure(III.4) : Représentation d'un réseau feedforward

Supposant qu'on a un vecteur à P entrées de données, on peut définir une erreur de mesure pour la $p^{ème}$ entrée ($1 \leq p \leq P$) comme la somme des erreurs carrées :

$$E_p = \sum_{k=1}^{N(L)} (d_k - x_{L,k})^2 \tag{III.5}$$

Avec d_k est la $k^{ème}$ composante de la $p^{ème}$ sortie désirée et $x_{L,k}$ est la $k^{ème}$ composante du vecteur actuel de sorties du réseau. Evidemment quand E_p est nulle, le réseau est capable de reproduire exactement la $p^{ème}$ sortie désirée. Notre tâche est de minimiser une erreur globale définie comme :

$$E = \sum_{p=1}^P E_p \tag{III.6}$$

➤ **Remarque**

La définition de E_p n'est pas unique, d'autres définitions sont possibles pour des cas spécifiques ou des applications.

III.5 LA METHODE DE GRADIENT

Pour pouvoir utiliser la méthode de gradient, on doit d'abord observer qu'une variation du paramètre α va affecter la sortie du nœud qui contient α ainsi que la couche finale et par conséquent l'erreur de mesure.

On définit le signal d'erreur $\varepsilon_{l,i}$ comme étant la dérivé ordonnée de l'erreur mesurée E_p du nœud i de la couche l , et on le symbolise : [JAN 92]

$$\varepsilon_{l,i} = \frac{\partial^+ E_p}{\partial x_{l,i}} \quad (\text{III.7})$$

Le calcul de la dérivé ordonnée est illustrer par l'exemple suivant :

$$\text{Si on a} \quad \begin{cases} y = f(x) \\ z = g(x, y) \end{cases} \quad (\text{III.8})$$

Alors :

$$\frac{\partial^+ z}{\partial x} = \frac{\partial g(x, f(x))}{\partial x} = \frac{\partial g(x, y)}{\partial x} \Big|_{y=f(x)} + \frac{\partial g(x, f(x))}{\partial y} \Big|_{y=f(x)} \cdot \frac{\partial f(x)}{\partial x} \quad (\text{III.9})$$

pour les nœuds de la dernière couche le signal d'erreur est :

$$\varepsilon_{L,i} = \frac{\partial^+ E_p}{\partial x_{L,i}} = \frac{\partial E_p}{\partial x_{L,i}} \quad (\text{III.10})$$

$$\text{Cela est égal a} \quad \varepsilon_{L,i} = -2(d_i - x_{L,i}) \quad (\text{III.11})$$

pour les nœuds internes on a :

$$\varepsilon_{l,i} = \frac{\partial^+ E_p}{\partial x_{l,i}} = \sum_{m=1}^{N(l+1)} \frac{\partial^+ E_p}{\partial x_{l+1,m}} \times \frac{\partial f_{l+1,m}}{\partial x_{l,i}} = \sum_{m=1}^{N(l+1)} \varepsilon_{l+1,m} \frac{\partial f_{l+1,m}}{\partial x_{l,i}} \quad (\text{III.12})$$

Si α est un paramètre dans le i^{eme} nœud de la couche l , on a :

$$\frac{\partial^+ E_p}{\partial \alpha} = \frac{\partial^+ E_p}{\partial x} \frac{\partial f_{l,i}}{\partial \alpha} = \varepsilon_{l,i} \frac{\partial f_{l,i}}{\partial \alpha} \quad (\text{III.13})$$

s'il intervient dans plusieurs nœuds l'équation (III.13) devient :

$$\frac{\partial^+ E_p}{\partial \alpha} = \sum_{x^* \in S} \frac{\partial^+ E_p}{\partial x^*} \frac{\partial f^*}{\partial \alpha} \quad (\text{III.14})$$

Avec : S est l'ensemble des nœuds où α intervient directement et f^* est la fonction du nœud pour une valeur x^* calculée.

La dérivée ordonnée de l'erreur de mesure globale par rapport à α est :

$$\frac{\partial^+ E}{\partial \alpha} = \sum_{p=1}^P \frac{\partial^+ E_p}{\partial \alpha} \quad (\text{III.15})$$

L'ajustement du paramètre α se fait par la relation :

$$\alpha(t+1) = \alpha(t) - \eta \frac{\partial^+ E}{\partial \alpha} \quad (\text{III.16})$$

Tel que η est le pas d'apprentissage exprimé par :

$$\eta = \frac{k}{\sqrt{\sum_{\alpha} \left(\frac{\partial^+ E}{\partial \alpha} \right)^2}} \quad (\text{III.17})$$

k est un paramètre choisi pour varier la vitesse de convergence.

Il est à noter qu'il existe deux méthodes principales d'apprentissage :

- **Apprentissage off-line** : l'ajustement des paramètres se fait seulement après que tous les exemples seraient passés dans le réseau.
- **Apprentissage on-line** : les paramètres sont ajustés chaque fois qu'une paire d'entrées – sorties soit présente.

III.6 LE RESEAU NEURO-FLOU

La mise d'un système d'inférence floue sous la forme d'un réseau adaptatif est appelée système d'inférence floue basé sur les réseaux adaptatifs (ANFIS : Adaptive Network-based Fuzzy Inference System).[SHI 95]

III.6.1 Architecture

Dans ce qui va suivre, on va s'intéresser au modèle de Sugeno. Pour la simplicité on considère que le système d'inférence floue est à deux entrées et une seule sortie, donc les deux règles floues sont exprimées comme suit :[SHI 95]

Règles 1 : si x est A_1 et y est B_1
Alors $f_1 = p_1 x + q_1 y + r_1$

Règles 2 : si x est A_2 et y est B_2
Alors $f_2 = p_2 x + q_2 y + r_2$

La figure (III.5) illustre le mécanisme de raisonnement pour le modèle de Sugeno, son architecture équivalente sous forme d'un réseau adaptatif est présentée par la figure (III.6), où les nœuds de la même couche ont des fonctions similaires ; (on notera la sortie du nœud i dans la couche l par $O_{l,i}$)

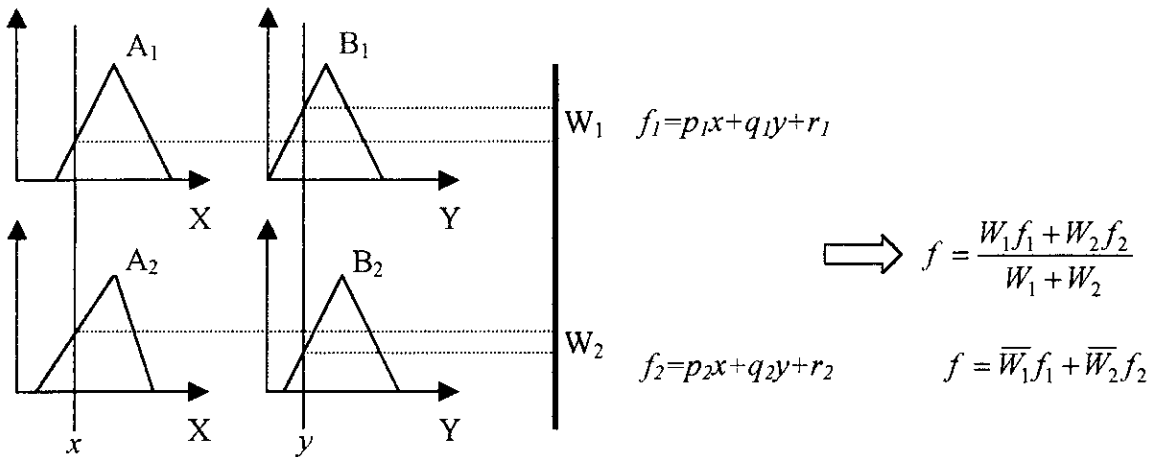


figure (III.5) : Modèle de Sugeno à deux règles

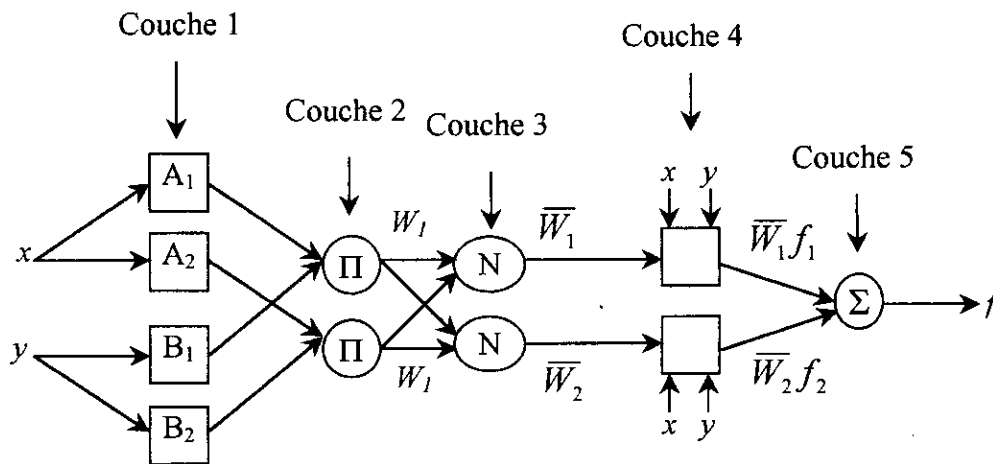


Figure (III.6) : le système de règles floues de Sugeno sous forme d'un RAN

La description du système présenter ci-dessus est la suivante :[SHI 95][JAN 92]

Couche 1 : chaque nœud dans cette couche est à paramètres variables avec la sortie du nœud définie par :

$$\begin{aligned}
 O_{1,i} &= \mu_{A_i}(x), \text{ pour } i=1, 2, \text{ ou} \\
 O_{1,i} &= \mu_{B_{i-2}}(y), \text{ pour } i=3, 4
 \end{aligned}
 \tag{III.18}$$

Où x (ou y) est l'entrée et A_i (ou B_i) est un ensemble flou associé au nœud, c'est à dire que les sorties de cette couche sont les valeurs des fonctions d'appartenance associées aux prémisses.

Les fonctions d'appartenance peuvent être trapézoïdales, triangulaires ou une fonction dérivable comme la fonction gaussienne :

$$\mu_{A_i}(x) = \exp \left\{ - \left[\left(\frac{x - c_i}{a_i} \right)^2 \right]^{b_i} \right\} \quad (\text{III.19})$$

Où $\{a_i, b_i, c_i\}$ est l'ensemble des paramètres à ajuster.

Couche 2 : chaque nœud dans cette couche est un nœud fixe, on le symbolise par (Π) et sa sortie est exprimée par :

$$O_{2,i} = w_i = \mu_{A_i}(x) \times \mu_{B_i}(y), \quad i=1,2 \quad (\text{III.20})$$

Couche 3 : c'est une couche de normalisation, ces nœuds sont fixes et réalisent la fonction suivante :

$$O_{3,i} = \bar{w}_i = \frac{w_i}{w_1 + w_2}, \quad i=1,2 \quad (\text{III.21})$$

Les sorties de cette couche sont appelées les poids normalisés.

Couche 4 : chaque nœud de cette couche est à paramètres variables et a comme fonction :

$$O_{4,i} = \bar{w}_i f_i = \bar{w}_i (p_i x + q_i y + r_i) \quad i=1,2 \quad (\text{III.22})$$

Où \bar{w}_i est la sortie de la couche 3 et $\{p_i, q_i, r_i\}$ est l'ensemble des paramètres à ajuster.

Couche 5 : cette couche a un seul nœud fixe qu'on symbolise par (Σ), il calcule la sortie finale du réseau comme étant la somme de tout les signales entrants :

$$O_{5,i} = \sum_i \bar{w}_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \quad (\text{III.23})$$

Ainsi on a construit un réseau adaptatif qui a exactement la fonction qu'un système de règles floues de Sugeno, il est à noter que la structure de ce réseau adaptatif n'est pas unique, on peut facilement combiner les couches 3 et 4 pour avoir un réseau équivalent à 4 couches seulement.

III.6.2 Détermination du régulateur

D'après l'architecture du réseau adaptatif du modèle flou de la figure (III.6), on peut observer que si les paramètres des prémisses sont fixes, la sortie finale du réseau peut être exprimée comme une combinaison linéaire des paramètres de conséquence, donc f peut être réécrite comme suit :

$$\begin{aligned} f &= \frac{w_1}{w_1 + w_2} f_1 + \frac{w_2}{w_1 + w_2} f_2 \\ &= \bar{w}_1 f_1 + \bar{w}_2 f_2 \end{aligned}$$

$$f = (\bar{w}_1 x) p_1 + (\bar{w}_1 y) q_1 + (\bar{w}_1) r_1 + (\bar{w}_2 x) p_2 + (\bar{w}_2 y) q_2 + (\bar{w}_2) r_2 \quad (\text{III.24})$$

Cette équation est linéaire par rapport aux paramètres p_1 , q_1 , r_1 , p_2 , q_2 et r_2 . La synthèse du régulateur de type Sugeno revient à la détermination de ces paramètres.

III.7 APPLICATION DE LA COMMANDE NEURO-FLOUE AU ROBOT PUMA560

III.7.1 Introduction

Dans ce qui va suivre, nous allons synthétiser un régulateur de type Sugeno à trois classes d'appartenance. ce régulateur présente deux avantages majeurs vis à vis de celui de type Mamdani :

- Temps de calcul de la commande réduit.
- Facilité de mettre sous forme de RNA.

III.7.2 Synthèse du régulateur

L'obtention du régulateur de type Sugeno à partir des connaissances vagues et imprécises est très difficile, donc nous allons construire ce régulateur à partir de celui de type Mamdani en suivant la procédure suivante :

- Construire un fichier E/S à partir des résultats du régulateur de type Mamdani déjà réalisé.
- Mettre le régulateur de type Sugeno sous forme d'un réseau adaptatif, et le soumettre à un algorithme d'apprentissage pour déterminer les paramètres qui donnent les meilleures performances. Le fichier qui sera utilisé dans l'apprentissage est celui du régulateur de type Mamdani.

La représentation du régulateur sous forme d'un réseau adaptatif est illustrée par la figure (III.7) où les nœuds à paramètres variables sont ceux de l'avant dernière couche.

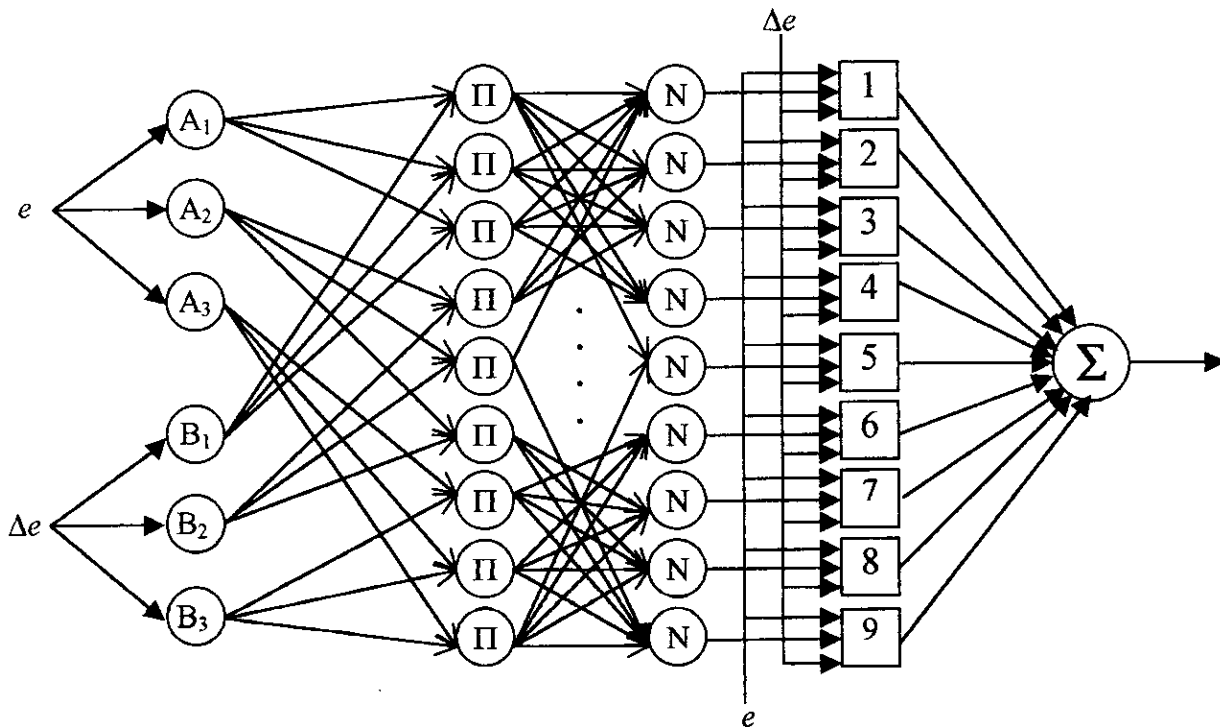


Figure (III.7) : Le régulateur de type Sugeno sous forme de RNA

Couche 1 :

La sortie de chaque nœud dans cette couche :

$$\begin{aligned} O_{1,i} &= \mu_{A_i}(e), \text{ pour } i=1, 3, \text{ ou} \\ O_{1,i} &= \mu_{B_{i-2}}(\Delta e), \text{ pour } i=4, 6 \end{aligned} \quad (\text{III.25})$$

Tel que μ_A et μ_B sont les fonctions d'appartenance associées aux prémisses, triangulaire à paramètres fixes.

Couche 2 :

Les sorties des nœuds de cette couche sont exprimées par :

$$\left\{ \begin{aligned} O_{2,1} &= O_{1,1} \cdot O_{1,4} = w_1 \\ O_{2,2} &= O_{1,1} \cdot O_{1,5} = w_2 \\ O_{2,3} &= O_{1,1} \cdot O_{1,6} = w_3 \\ O_{2,4} &= O_{1,2} \cdot O_{1,4} = w_4 \\ O_{2,5} &= O_{1,2} \cdot O_{1,5} = w_5 \\ O_{2,6} &= O_{1,2} \cdot O_{1,6} = w_6 \\ O_{2,7} &= O_{1,3} \cdot O_{1,4} = w_7 \\ O_{2,8} &= O_{1,3} \cdot O_{1,5} = w_8 \\ O_{2,9} &= O_{1,3} \cdot O_{1,6} = w_9 \end{aligned} \right. \quad (\text{III.26})$$

couche 3 :

La sortie du nœud est exprimée par :

$$O_{3,i} = \bar{w}_i = \frac{w_i}{\sum_{k=1}^9 w_k} \quad (\text{III.27})$$

couche 4 :

Les nœuds de cette couche sont à paramètres ajustables, leurs fonctions sont de la forme :

$$O_{4,i} = \bar{w}_i (p_i \cdot e + q_i \cdot \Delta e + r_i) \quad i=1,9 \quad (\text{III.28})$$

Tel que p , q et r sont les paramètres à ajuster.

Couche 5 :

Cette couche contient un seul nœud qui réalise la somme des sorties des nœuds de la couche précédente :

$$O_5 = \sum_{i=1}^9 O_{4,i} \quad (\text{III.29})$$

Les paramètres des prémisses sont supposés fixes, donc O_5 peut être écrite sous la forme :

$$O_5 = \sum_{i=1}^9 (\bar{w}_i e) p_i + (\bar{w}_i \Delta e) q_i + (\bar{w}_i) r_i \quad (\text{III.30})$$

III.7.3 Algorithme d'apprentissage

L'algorithme que nous allons utiliser est celui de back-propagation, pour minimiser l'erreur suivante :

$$E = \sum_{p=1}^P E_p \quad (\text{III.31})$$

Avec :

$$E_p = \frac{1}{2} (T_p - O_{5,p})^2 \quad 1 \leq p \leq P \quad (\text{III.32})$$

et

P : nombre d'exemples à apprendre.

T_p : composante de la sortie désirée pour l'exemple p .

$O_{5,p}$: composante de la sortie réelle pour l'exemple p .

L'application de cet algorithme dans notre cas est très simple, du fait que les paramètres à ajuster se trouvent à la quatrième couche.

La dérivé ordonnée de E_p par rapport à un paramètre α s'écrit :

$$\frac{\partial^+ E_p}{\partial \alpha} = \frac{\partial E_p}{\partial O_{5,p}} \cdot \frac{\partial O_{5,p}}{\partial \alpha} \quad (\text{III.33})$$

Avec :

$$\frac{\partial E_p}{\partial O_{5,p}} = -(T_p - O_{5,p}) \quad (\text{III.34})$$

A partir de l'équation (III.30) la dérivé de $O_{5,p}$ par rapport aux paramètres p , q et r sont :

$$\begin{cases} \frac{\partial O_{5,p}}{\partial p_i} = \bar{w}_i \cdot e \\ \frac{\partial O_{5,p}}{\partial q_i} = \bar{w}_i \cdot \Delta e \\ \frac{\partial O_{5,p}}{\partial r_i} = \bar{w}_i \end{cases} \quad i=1,9 \quad (\text{III.35})$$

donc :

$$\begin{cases} \frac{\partial^+ E_p}{\partial p_i} = -(T_p - O_{s,p}) \cdot \bar{w}_i \cdot e \\ \frac{\partial^+ E_p}{\partial q_i} = -(T_p - O_{s,p}) \cdot \bar{w}_i \cdot \Delta e \\ \frac{\partial^+ E_p}{\partial r_i} = -(T_p - O_{s,p}) \cdot \bar{w}_i \end{cases} \quad i=1,9 \quad (\text{III.36})$$

et la dérivée de l'erreur globale s'obtient par :

$$\frac{\partial^+ E}{\partial p_i} = \sum_{p=1}^P \frac{\partial^+ E_p}{\partial p_i} \quad i=1,9 \quad (\text{III.37})$$

$$\frac{\partial^+ E}{\partial q_i} = \sum_{p=1}^P \frac{\partial^+ E_p}{\partial q_i} \quad i=1,9 \quad (\text{III.38})$$

$$\frac{\partial^+ E}{\partial r_i} = \sum_{p=1}^P \frac{\partial^+ E_p}{\partial r_i} \quad i=1,9 \quad (\text{III.39})$$

Nous ajustons les paramètres par les relations :

$$p_i(t+1) = p_i(t) - \eta \frac{\partial^+ E}{\partial p_i} \quad i=1,9 \quad (\text{III.40})$$

$$q_i(t+1) = q_i(t) - \eta \frac{\partial^+ E}{\partial q_i} \quad i=1,9 \quad (\text{III.41})$$

$$r_i(t+1) = r_i(t) - \eta \frac{\partial^+ E}{\partial r_i} \quad i=1,9 \quad (\text{III.42})$$

III.7.4 Valeurs initiales des paramètres

Puisque nous n'avons pas une idée précise sur les paramètres p , q et r , nous avons choisi plusieurs paramètres initiaux pour l'apprentissage, ce qui nous a permis de tirer les conclusions suivantes :

- Si les valeurs initiales sont grandes, alors le temps de convergence sera énorme.
- Le temps de convergence se réduit pour de petites valeurs initiales.
- Les paramètres finaux dépendent des paramètres initiaux.

III.7.5 Résultats

Dans ce paragraphe, nous allons présenter les résultats obtenus par l'algorithme d'apprentissage pour la synthèse du régulateur de type Sugeno, et les résultats de son implémentation pour la commande du robot *PUMA560*.

Dans un premier essai, nous avons initialisé les paramètres p , q et r à 0.5, puis à 1 dans un deuxième essai. Les résultats de l'apprentissage pour les deux essais sont respectivement présentés dans les tableaux (III.1) et (III.2).

| i | p_i | q_i | r_i |
|-----|-------|-------|--------|
| 1 | 0.5 | 0.5 | 0.499 |
| 2 | 0.508 | 0.5 | 0.486 |
| 3 | 0.5 | 0.5 | 0.5 |
| 4 | 0.5 | 0.5 | 0.187 |
| 5 | 0.384 | 0.364 | 0.00 |
| 6 | 0.498 | 0.499 | -0.054 |
| 7 | 0.5 | 0.5 | 0.5 |
| 8 | 0.574 | 0.498 | -0.049 |
| 9 | 0.5 | 0.5 | 0.499 |

Tableau (III.1)

| i | p_i | q_i | r_i |
|-----|-------|-------|--------|
| 1 | 1.00 | 1.00 | 0.999 |
| 2 | 0.866 | 1.00 | 0.265 |
| 3 | 1.00 | 1.00 | 1.00 |
| 4 | 1.00 | 1.00 | 0.320 |
| 5 | 0.594 | 0.541 | 0.00 |
| 6 | 0.997 | 0.999 | -0.184 |
| 7 | 1.00 | 1.00 | 1.00 |
| 8 | 1.05 | 0.996 | -0.274 |
| 9 | 1.00 | 1.00 | 0.997 |

Tableau (III.2)

Donc, à partir de deux initialisations différentes nous avons synthétisé deux régulateurs différents de type Sugeno. Ces derniers seront utilisés pour la commande du robot *PUMA560*.

◆ Analyse des performances

D'après les simulations, nous allons tirer quelques conclusions et remarques sur la réponse et les performances des deux régulateurs. Les résultats sont présentés par les figures de (III.8) et (III.9).

Comme dans le régulateur de type Mamdani, les deux régulateurs de type sugeno garantissent une très bonne poursuite, avec une erreur maximale ne dépassant pas 0.01 degrés et les commandes sont lisses et réalisables.

En comparant les résultats de simulation des deux régulateurs nous pouvons observer qu'il sont très proches, voir identiques.

◆ Tests de robustesse.

Les tests de robustesse sont ceux appliqués sur les régulateurs de type Mamdani. Les résultats sont présentés par les figures de (III.10) à (III.15).

▪ Test de la charge

Pour les deux tests utilisés, les deux régulateurs fournissent le couple qui compense la charge, et évite au robot d'être influencé par le lâchage de cette charge.

▪ Rupture de la commande

Comme nous avons conservé le principe de la décentralisation, la rupture de la commande de la deuxième articulation influe seulement sur cette dernière, les deux autres articulations suivent leurs références normalement.

Donc, les performances et la robustesse des deux régulateurs de type Sugeno sont très satisfaisantes, bien que leurs structures soient différentes.

III.8 CONCLUSION

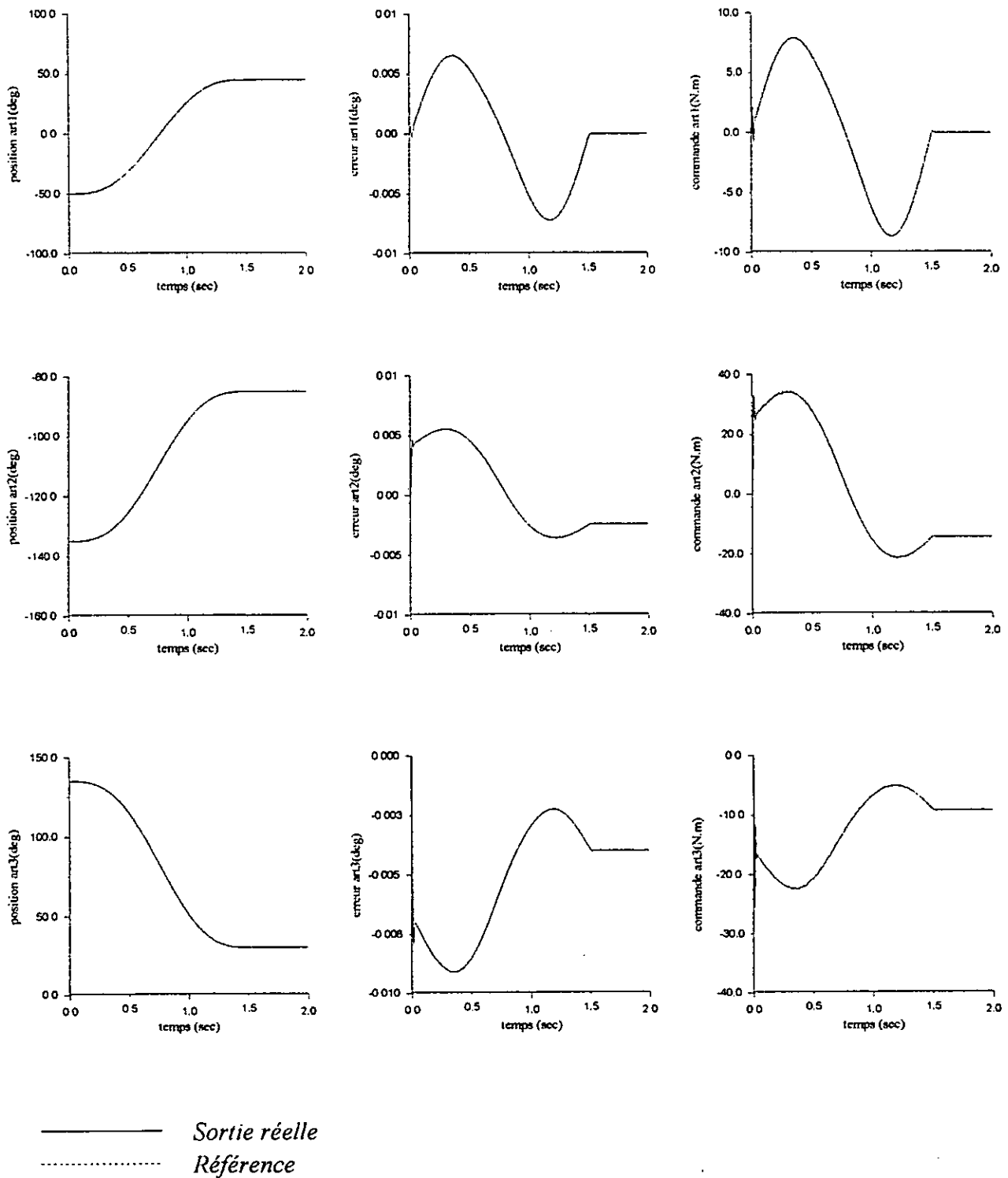
Dans ce chapitre nous avons présenté la théorie des réseaux de neurones adaptatifs, l'algorithme d'apprentissage de back-propagation et la mise d'un système d'inférence floue sous forme d'un RNA.

Pour réduire le temps de calcul de la commande nous avons synthétisé le régulateur de type Sugeno qui a donné les mêmes performances que celles du régulateur de type Mamdani.

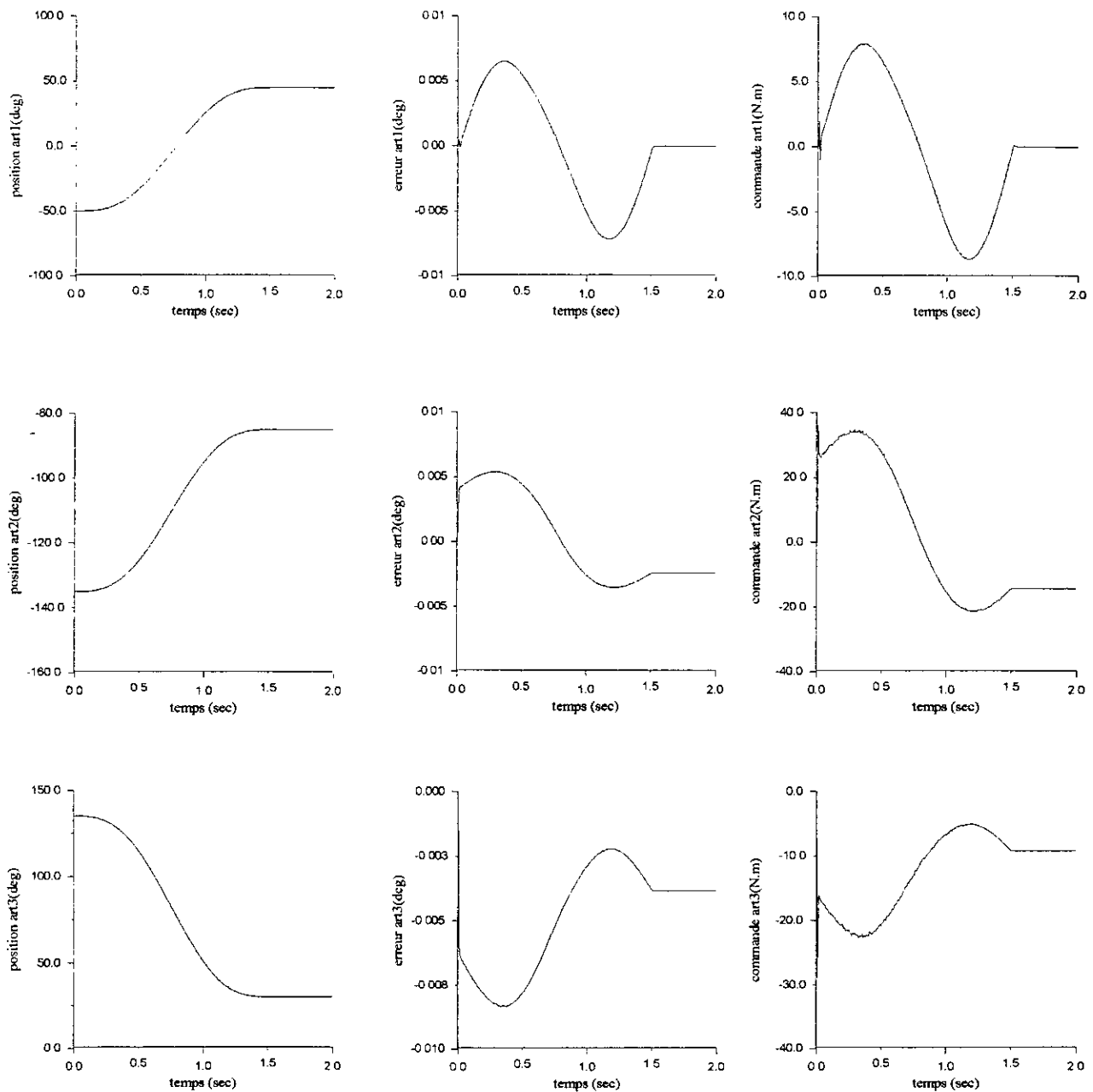
Le régulateur est d'abord représenté sous forme d'un RNA, puis ses paramètres sont déterminés en utilisant l'algorithme de back-propagation. Le choix des paramètres initiaux influe sur le temps de convergence, ainsi que sur la structure du régulateur.

Les paramètres p , q , et r du régulateur de type Sugeno ne sont pas uniques. En effet nous avons utilisé deux ensembles différents de paramètres qui ont donné des résultats très satisfaisants pour la commande du robot *PUMA560*.

Donc, nous pouvons conclure que l'introduction des réseaux de neurones adaptatifs dans la commande floue constitue un outil qui permet d'améliorer les performances de cette commande.

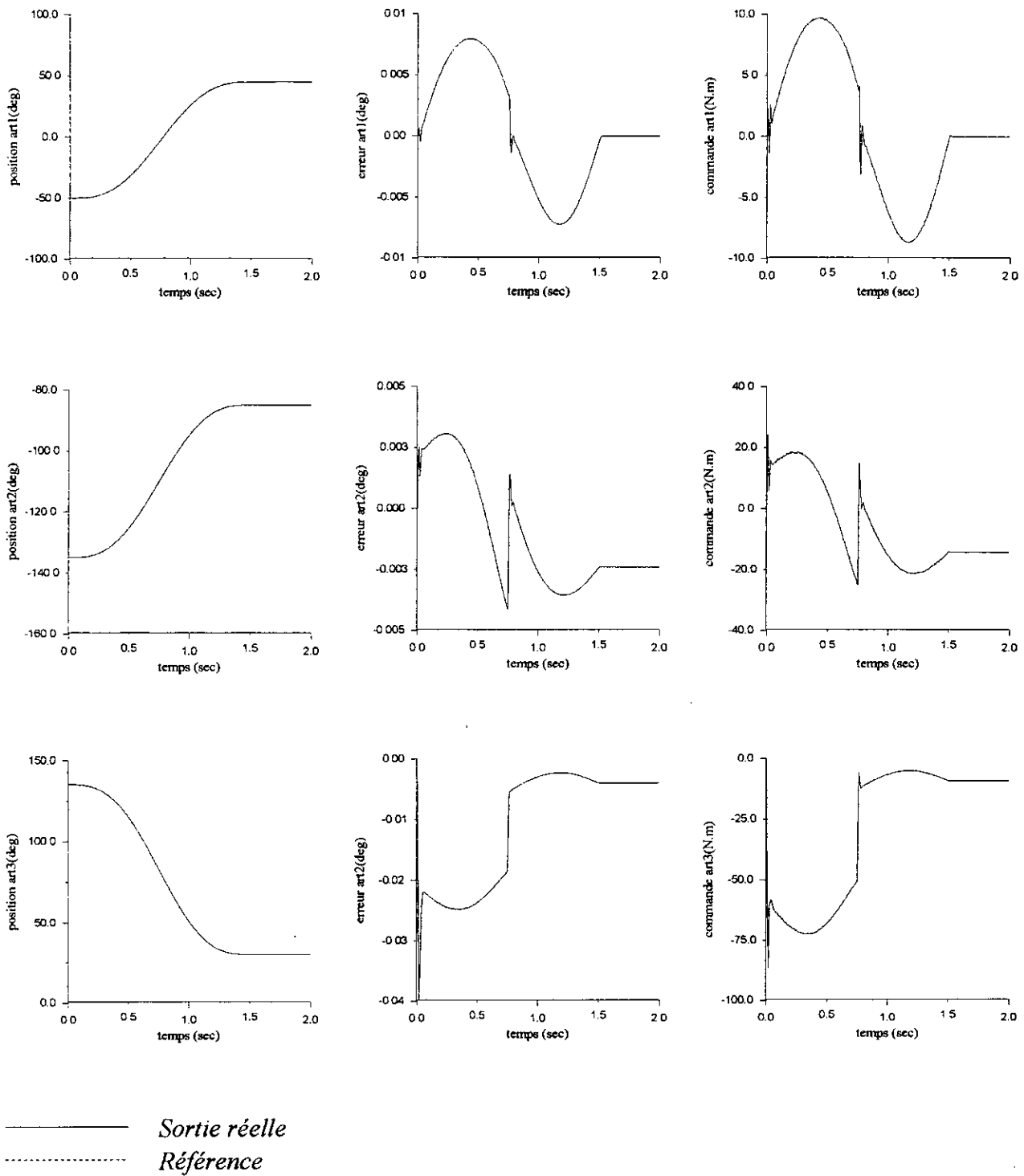


Figure(III.8) : Réponse du robot pour une poursuite à vide. Régulateur de type Sugeno (premier choix des paramètres)

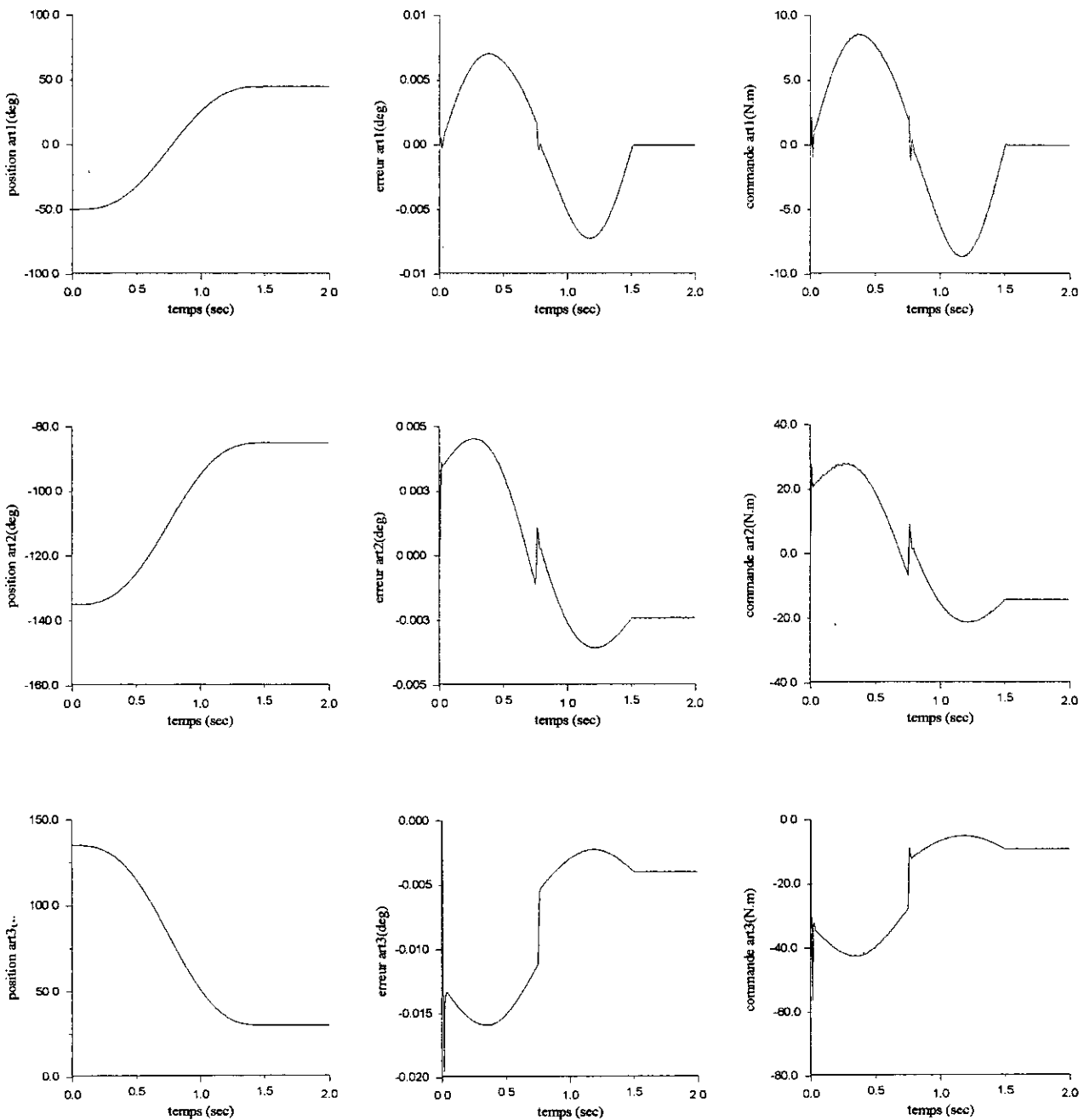


————— Sortie réelle
 Référence

Figure(III.9) : Réponse du robot pour une poursuite à vide.
 Régulateur de type Sugeno (deuxième choix des paramètres)



Figure(III.10) : Réponse du robot pour le test de la charge (10 kg).
 Régulateur de type Sugeno (premier choix des paramètres)



————— Sortie réelle
 Référence

Figure(III.11) : Réponse du robot pour le test de la charge (4 kg).
 Régulateur de type Sugeno (premier choix des paramètres)

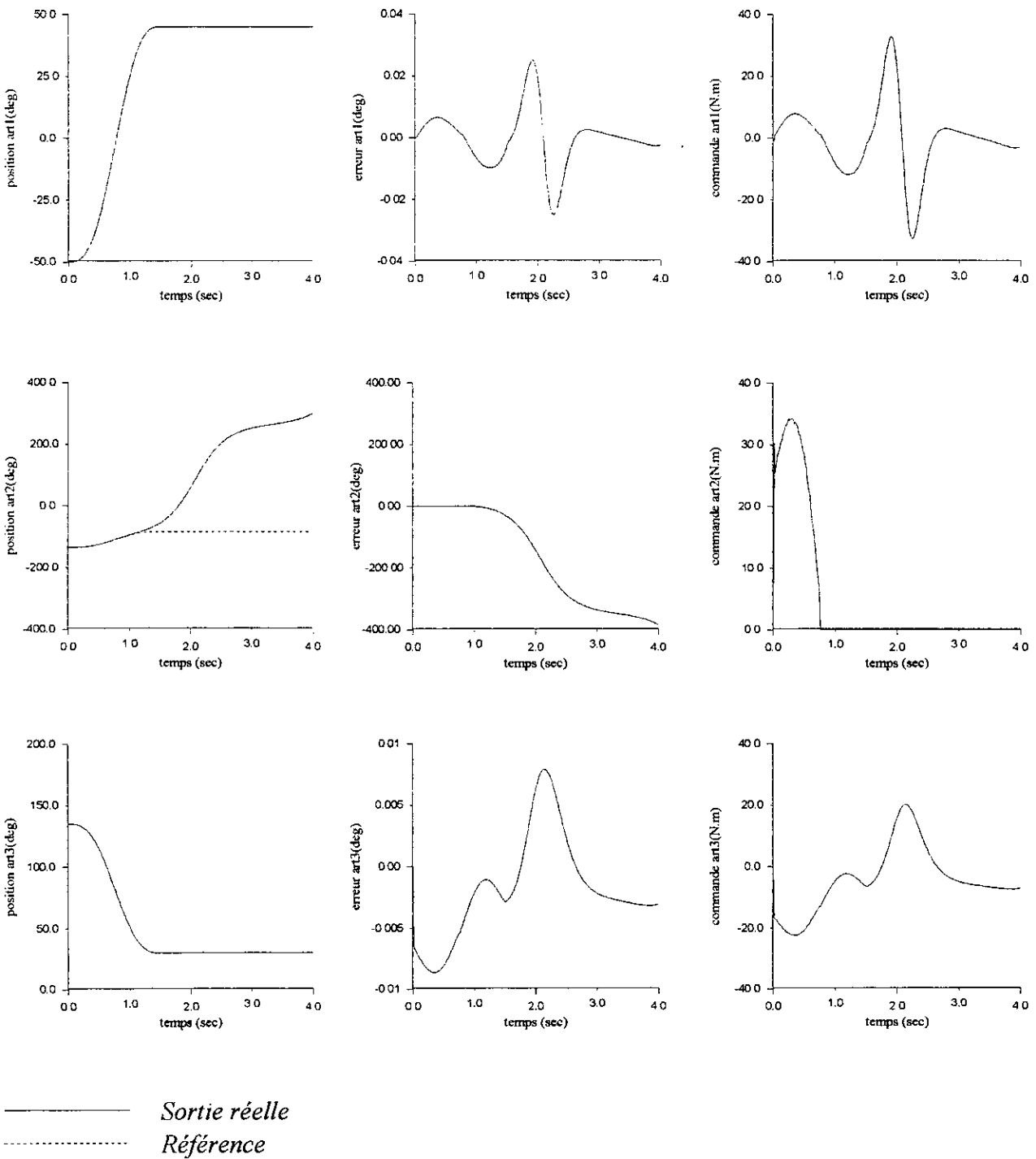
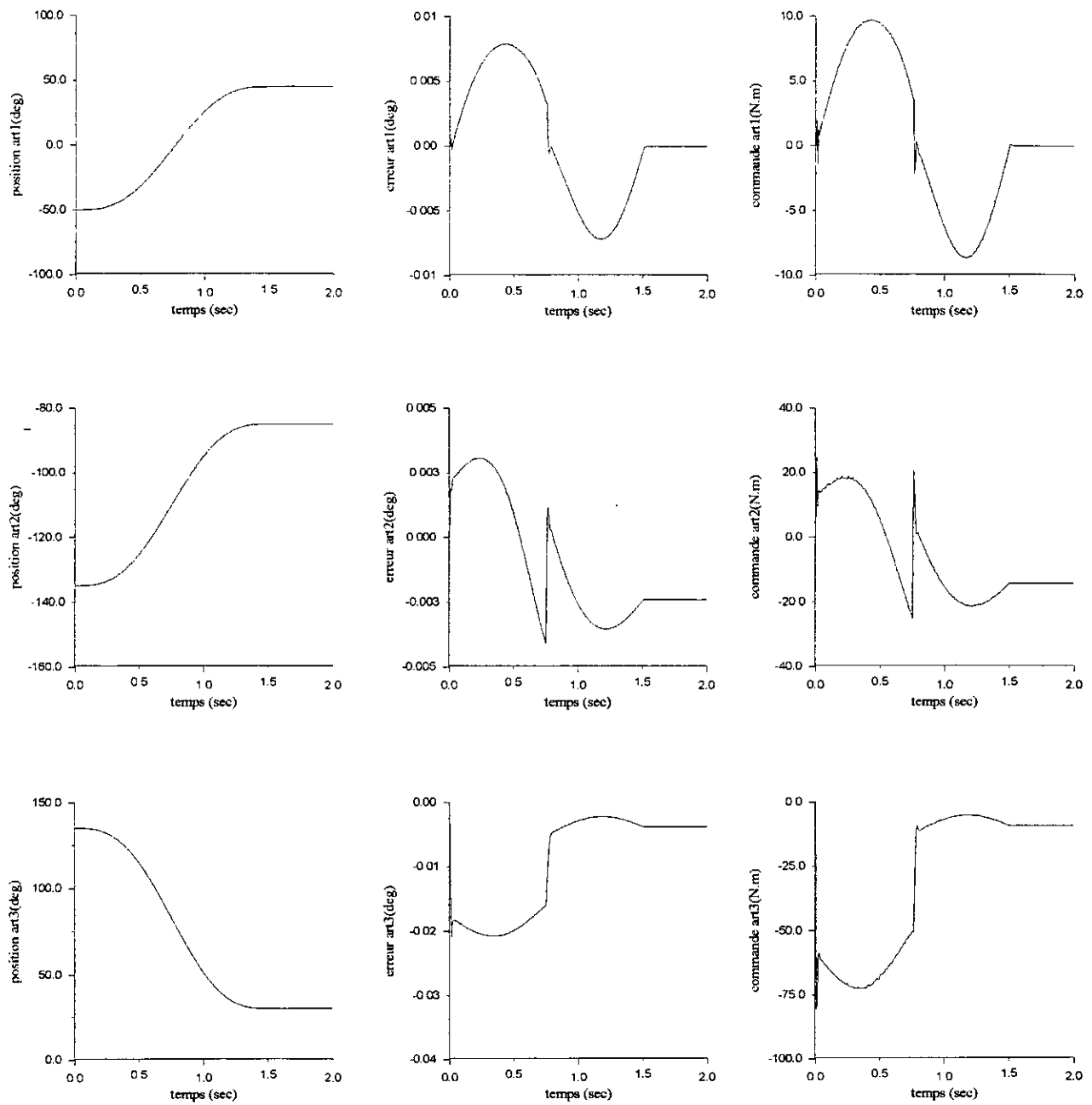
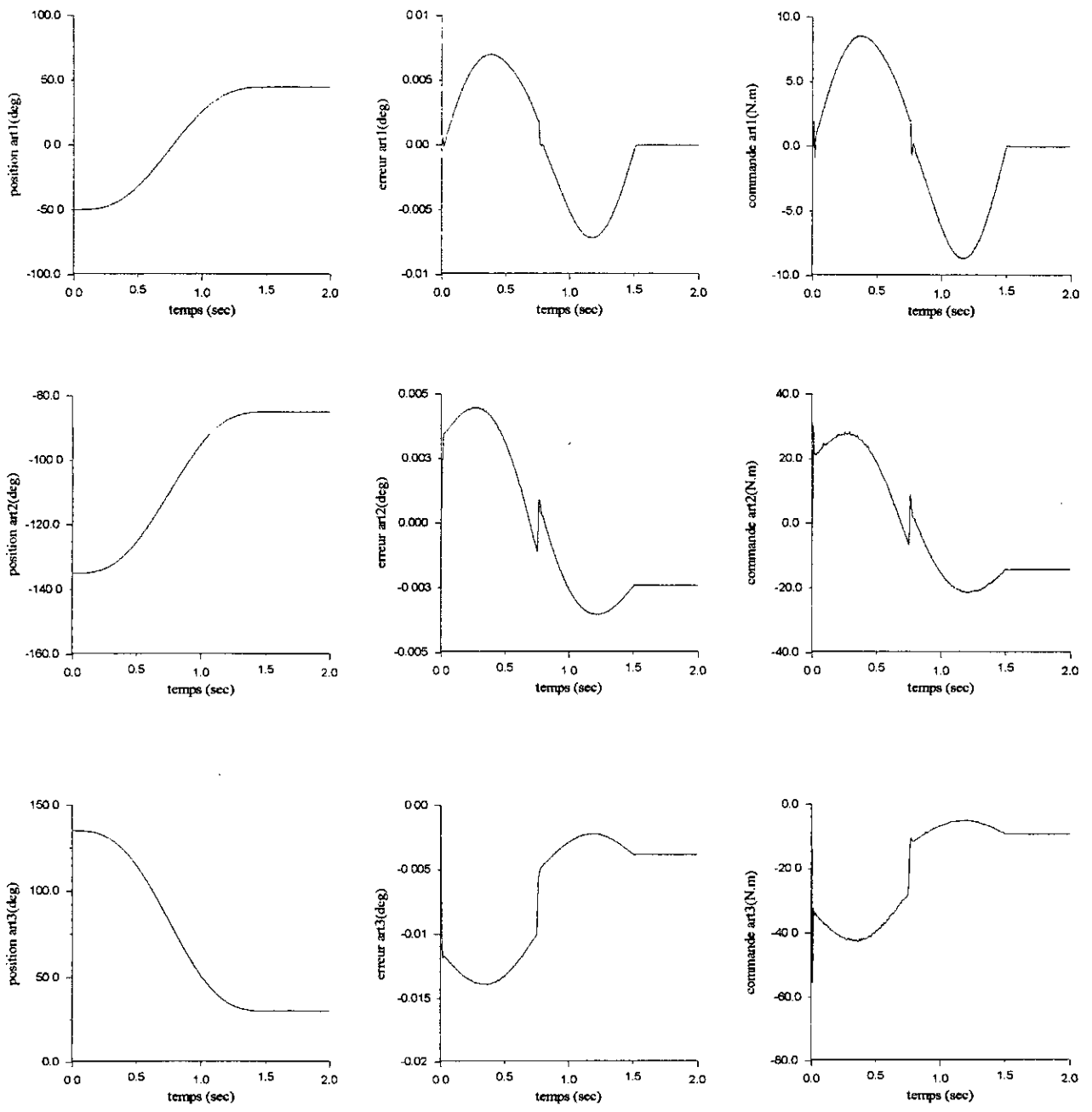


Figure (III.12) : Réponse du robot pour un test de rupture de la commande Régulateur flou de type Sugeno (premier choix des paramètres).



————— Sortie réelle
 - - - - - Référence

Figure(III.13) : Réponse du robot pour le test de la charge (10 kg).
 Régulateur de type Sugeno (deuxième choix des paramètres)



————— *Sortie réelle*
 *Référence*

Figure(III.14) : Réponse du robot pour le test de la charge (4 kg).
 Régulateur de type Sugeno (deuxième choix des paramètres)

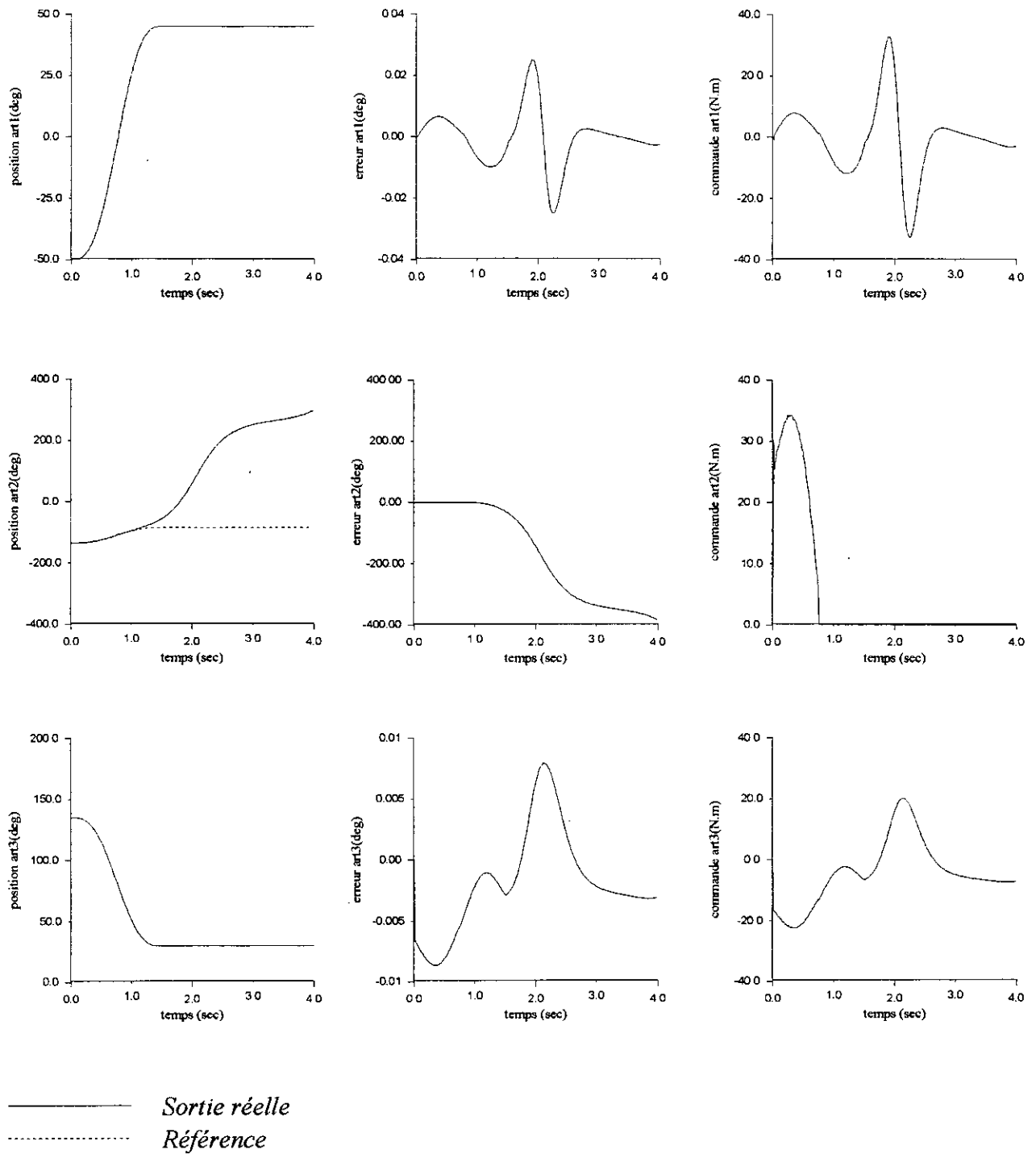


Figure (III.15) : Réponse du robot pour un test de rupture de la commande Régulateur flou de type Sugeno (deuxième choix des paramètres).



**CONCLUSION
ET
PERSPECTIVES**

CONCLUSION

Dans ce mémoire, nous avons appliqué la commande floue et neuro-floue décentralisée au robot *PUMA560*. Trois régulateurs flous de type MAMDANI (trois, cinq et sept classes d'appartenance) ont été d'abord appliqués, des résultats de simulation et de tests de robustesse ont été présentés ainsi qu'une étude comparative entre les trois régulateurs.

Les performances des régulateurs ont été très satisfaisantes. Celui à trois classes d'appartenance est retenu pour sa simplicité et ses bonnes performances.

La technique neuro-floue adoptée par la suite et l'algorithme de back-propagation nous ont permis de synthétiser le régulateur de type Sugeno à trois classes qui améliore la commande en la rendant plus simple à formuler et en réduisant son temps de calcul, tout en gardant ces bonnes performances.

L'utilisation des RNA pour la détermination du régulateur de type Sugeno est facile et peut être appliquée à d'autres types de systèmes.

La commande par logique floue présente un domaine d'application vaste, néanmoins l'inconvénient réside dans le tâtonnement des gains de normalisation et ceux associés aux commandes. Une étude plus approfondie dans le domaine peut ouvrir une bonne voie d'investigation.

L'application de la commande floue et neuro-floue à d'autres types de systèmes complexes tels que les réseaux électriques et les processus pétrochimiques, nous semble être très utile afin de remédier au problème de complexité de la modélisation de ces types de systèmes.

REFERENCES BIBLIOGRAPHIQUES

- [AMS 86] B.AMSTRONG, O. KHATIB and J.BURDICK, « The explicit dynamic model and inertial of the PUMA 560 arm » Proc. International conference on robotics and automation, 1986.
- [BAL 95] N.BALI, « Etudes des performance de la commande prédictive généralisée application au robot PUMA et SCARA. » Thèse de magistère ENP 1995.
- [BAR 96] J.P.BARRAT, M.BARRAT, Y.LECLUSE « exemple d'application de la logique floue : commande de la température d'un four pilote» Techniques de l'ingénieur. R7428, 1996.
- [BUH 94] H.BUHLER . « réglage par logique floue » Presse polytechnique Romande 1994.
- [DUB 87] D.DUBOIS, H.PRADE « Théories des possibilités : App à la représentation des connaissances en informatique. » MASSON 1987.
- [JAN 91] J.S.R JANG « Self learning fuzzy controller based on temporal back-propagation ». IEEE Trans. Neural networks vol3, pp714 -723. Sep 1992.
- [KAU 87] A.KAUFMANN, « Nouvelle logique pour l'intelligence artificielle. » Ed HERMES 1987.
- [LAL 94] J.P.LALLEMAND, S.ZEGHLOUL « Robotique : Aspect fondamentaux. » MASSON 1994.
- [LEE 90a] C.C. LEE « Fuzzy logic in control systems :Fuzzy logic controler .Part I. » IEEE Trans. Syst. man. and Cybern. Vol 20, N°2 pp 419-435. Mar/Avr 1990.
- [LOU 97] M.LOUDINI « Modélisation, analyse et méthodologie de commande linguistique floue d'un bras manipulateur de robot flexible. » Thèse de magistère ENP 1997.
- [MAD 97] T.MADANI « Commande décentralisée à structure variable, Application en robotique. » Thèse de PFE. ENP 1997.
- [MAM 74] E.H. MAMDANI « Application of fuzzy algorithme for the control of a dynamic plant. » Proc. IEEE .vol 121 N°12 pp1585-1588. 1994.
- [NED 96] NEDJARI. BOUKHARI « Commande adaptative décentralisée, application en robotique. » Thèse de PFE ENP 1996.
- [PAU 83] R.P.PAUL « Robots manipulators : Mathematics programing and control. » The MIT Press 1983.
- [RAN 95] J.M.RANDERS « Algorithmes génétiques et réseaux de neurones. » HERMES 1995.

- [SER 89]** H.SERAJI « Decentralized adaptive control of manipulator. Théory, simulation and expérimentation. » IEEE, Trans on robotics and automation, vol 5 N°2 pp 183-201, Avr 1989.
- [SHI 95]** J.Y.H. SHING ROGER JANG « Neuro-fuzzy modeling and control. » PROC. IEEE, vol 83 N°3 pp 378-404, Mar 1995.
- [TZA 90]** S.TZAFTAS « Instrumental fuzzy PID control » . IEEE Trans industrial electronics, vol 37 N°5 Oct 1990.