

16/97

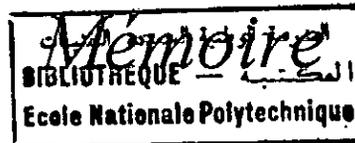
الجمهورية الجزائرية الديمقراطية الشعبية
REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

وزارة التعليم العالي و البحث العلمي
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

المدرسة الوطنية المتعددة التقنيات
ECOLE NATIONALE POLYTECHNIQUE

DER de Génie Electrique et Informatique

FILIERE : AUTOMATIQUE



En vue d'obtenir le diplôme
d'Ingénieur d'Etat en AUTOMATIQUE

THÈME :

**COMMANDE FLOUE ET NEURO-
FLOUE DÉCENTRALISÉE
APPLICATION EN ROBOTIQUE**

Proposé par :
Mr D. BOUKHETALA
Mr F. BOUDJEMA

Fait par :
Mr MELAHI Ahmed
Mr BOULAHIA Abdelmalek

Dirigé par :
Mr D. BOUKHETALA
Mr F. BOUDJEMA

Promotion 1997
E.N.P. 10 Avenue Sassen Badi El-Sarrach Alger

AVANT-PROPOS

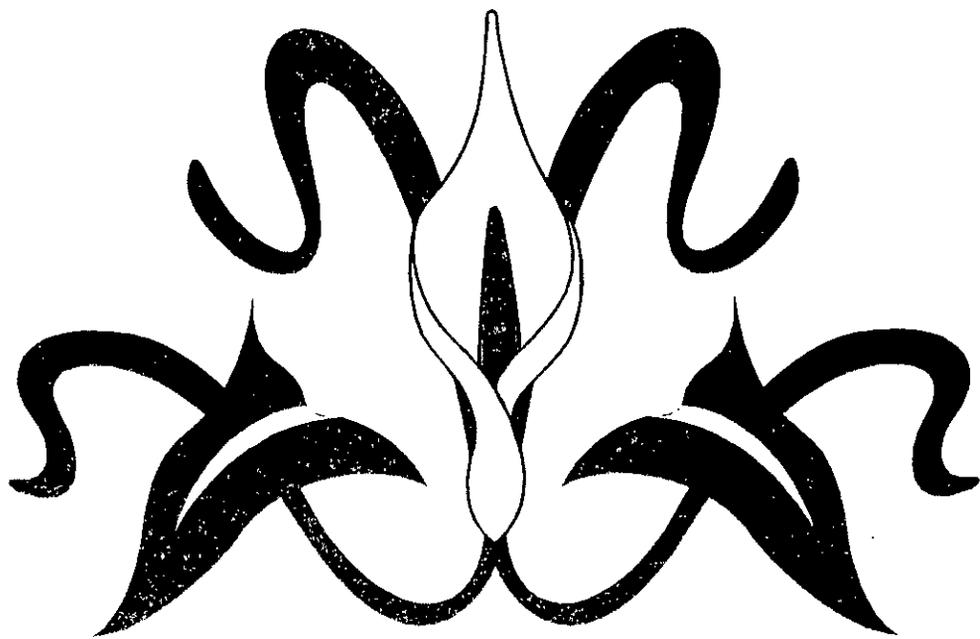
Nous tenons à exprimer notre reconnaissance à nos promoteurs Monsieur D.Boukhetala et Monsieur F.Boudjema pour l'aide qu'ils nous ont apportée pour réaliser ce mémoire.

Nous tenons également à remercier Monsieur H.Chekireb, qui nous a introduits dans le domaine de la commande par logique floue .

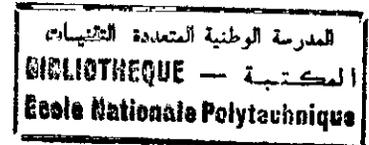
Nous adressons nos sincères remerciements à Messieurs les membres de jury, qui nous ont fait l'honneur de juger ce travail.

Sans oublier d'exprimer nos remerciements à tous nos amis, et tous ceux qui nous ont aidé de près ou de loin. En particulier, Ahmed, Abderahmane, Youcef et Bilal.

Nous remercions aussi les agents de la bibliothèque, Salah, Krimeo,... Que les enseignants, qui ont contribué à notre formation, reçoivent notre gratitude.



SOMMAIRE



INTRODUCTION GENERALE

CHAPITRE I. MODELISATION

I.1. Introduction.....	1
I.2. Présentation du robot.....	1
I.3. Transformations de Denavit-Hartenberg.....	2
I.4. Modélisation géométrique.....	4
I.5. Modélisation cinématique.....	5
I.6. Génération de trajectoires.....	6
I.6.1. Trajectoire cycloïdale.....	6
I.6.2. Trajectoire de Viviani.....	6
I.7. Modélisation dynamique.....	7
I.8. Conclusion.....	9

CHAPITRE II. COMMANDE PAR LOGIQUE FLOUE

II.1. Introduction.....	11
II.2. Sous-ensembles flous.....	11
II.2.1. Variables linguistiques.....	12
II.2.2. Opérations sur les sous-ensembles flous.....	12
II.2.2.1. Egalité.....	12
II.2.2.2. Inclusion.....	12
II.2.2.3. Complément.....	12
II.2.2.4. Intersection.....	13
II.2.2.5. Union.....	13
II.2.2.6. produit cartésien.....	13
II.2.2.7. Relation floue.....	13
II.3. Logique floue.....	13
II.4. Règles floues.....	14
II.5. Raisonnement et prise de décisions.....	14
II.6. Commande par logique floue.....	15
II.6.1. Structure générale d'un régulateur flou.....	16
II.6.1.1. Fuzzification.....	16
II.6.1.2. Inférence.....	16
II.6.1.3. Defuzzification.....	17
II.6.1.3.1. Méthode du maximum.....	17
II.6.1.3.2. Méthode de la moyenne des maxima.....	17
II.6.1.3.3. Méthode du centre de gravité.....	17
II.6.1.4. Bloc de règles.....	17
II.6.2. Types de régulateurs flous.....	18
II.6.2.1. Régulateur du type de Mamdani.....	18
II.6.2.2. Régulateur du type de Sugeno.....	18
II.6.2.3. Régulateur du "deuxième ordre".....	18
II.7. Conclusion.....	19

CHAPITRE III. COMMANDE PAR RESEAUX NEURO-FLOUS

III.1. Introduction.....	20
III.2. Réseaux adaptatifs généralisés.....	20
III.2.1. Définition.....	20
III.2.2. Architectures des réseaux adaptatifs généralisés.....	21
III.2.2.1. Les nœuds.....	21
III.2.2.2. Les liens.....	21
III.2.2.3. Propriétés.....	21
III.2.3. Catégories des RAN.....	22
III.2.4. Représentation.....	22
III.2.4.1. Représentation en couche.....	22
III.2.4.2. Représentation en ordre topologique.....	23
III.2.5. Exemples.....	23
III.3. Réseau neuro-flou.....	24
III.3.1. Méthode de Mamdani.....	25
III.3.1.1. Description.....	25
III.3.1.2. Apprentissage.....	27
III.3.2. Méthode de Sugeno.....	30
III.3.2.1. Description.....	30
III.3.2.2. Apprentissage.....	32

CHAPITRE IV. APPLICATION

IV.1. Introduction.....	35
IV.2. Décentralisation.....	36
IV.3. Décentralisation, modèle dynamique du robot.....	37
IV.4. Structure de la commande décentralisée par logique floue.....	37
IV.5. Régulateur du type de Mamdani.....	38
IV.5.1. Introduction.....	38
IV.5.2. Structure des régulateurs.....	39
IV.5.3. Synthèse des régulateurs.....	39
IV.5.3.1. Régulateurs à trois classes.....	40
IV.5.3.2. Régulateurs à cinq classes.....	41
IV.5.3.3. Régulateurs à sept classes.....	42
IV.5.4. Etude comparative.....	43
IV.5.4.1. Remarques sur l'aspect général de la réponse.....	43
IV.5.4.2. Comparaison et justification.....	44
IV.5.5. Performances et tests de robustesse.....	44
IV.5.5.1. Variations paramétriques.....	44
IV.5.5.2. Défaillance du capteur.....	45
IV.5.5.3. Rupture de la commande.....	45
IV.5.6. Conclusion.....	45
IV.6. Régulateur du type de Sugeno.....	56
IV.6.1. Introduction.....	56
IV.6.2. Détermination des régulateurs.....	56
IV.6.2.1. Synthèse du régulateur de type de Sugeno par identification.....	56
IV.6.2.2. Synthèse du régulateur de type de Sugeno par approximation.....	57

IV.6.2.3. Régulateur de type de Sugeno à trois classes.....	60
IV.6.2.4. Régulateur de type de Sugeno à deux classes.....	60
IV.6.3. Etude comparative.....	61
IV.6.3.1. Aspect général de la réponse.....	61
IV.6.3.2. Comparaison et justification.....	61
IV.6.4. Tests de robustesse.....	62
IV.6.4.1. Variation paramétrique.....	62
IV.6.4.2. Capteur défaillant.....	62
IV.6.4.3. Rupture de commande.....	63
IV.6.5. Conclusion.....	63
IV.7. Régulateurs neuro-flous.....	73
IV.7.1. Introduction.....	73
IV.7.2. Synthèse des régulateurs.....	73
IV.7.2.1. Apprentissage par TBP.....	74
IV.7.2.2. Résultat de l'apprentissage.....	78
IV.7.3. Etude comparative.....	80
IV.7.3.1. Aspect général de la réponse.....	80
IV.7.3.2. Comparaison et justification.....	80
IV.7.4. Tests de robustesse.....	80
IV.7.4.1. Variation paramétrique.....	80
IV.7.4.2. Capteur défaillant.....	80
IV.7.4.3. Rupture de commande.....	80
IV.7.5. Conclusion.....	80

CONCLUSION GENERALE

ANNEXE I . Fonctions d'appartenance.....	A.1
ANNEXE II . Normes et conormes triangulaires.....	A.2
ANNEXE III . Régulateurs par logique floue.....	A.3

BIBLIOGRAPHIE

المدرسة الوطنية المتعددة التقنيات
BIBLIOTHEQUE — المكتبة
Ecole Nationale Polytechnique

INTRODUCTION GENERALE

"Combien de royaumes nous ignorons!"
PLAISE PASCAL

INTRODUCTION GENERALE

Vu la complexité croissante des systèmes abordés de jour en jour, de nouvelles techniques s'avèrent nécessaires pour l'analyse et la synthèse de commandes. Parmi les techniques obtenues on cite, la commande décentralisée, la commande par logique floue,...

L'analyse des systèmes complexes présente de nombreux problèmes. Parmi ces problèmes, on trouve le grand nombre de données à traiter simultanément, ainsi que les limites physiques des calculateurs actuels. Pour pallier à ces problèmes, on procède par la décentralisation sous laquelle le système sera vu composé de plusieurs sous systèmes interconnectés.

L'apparition de la commande par logique floue a permis d'attaquer les systèmes mal définis, mal modélisés voire n'ayant aucun modèle mathématique. Cette technique utilise les connaissances des experts formulées sous forme de règles. Elle opère comme le raisonnement humain pour générer la commande.

Ces dernières années, on a constaté l'adaptation des techniques d'apprentissage, développées pour les réseaux de neurones artificiels, aux systèmes d'inférence floue. Cette adaptation a permis l'obtention des règles floues et les améliorer pour avoir de bonnes performances.

Ainsi, notre travail, commande floue et neuro-floue décentralisée, combine ces trois techniques. Il a pour objectif, la synthèse des régulateurs à base de connaissances ayant les performances de rapidité, de simplicité, de précision et en plus la capacité de l'apprentissage par l'algorithme TBP.

D'abord, des régulateurs flous de type de Mamdani sont synthétisés, car ils sont simples à avoir en utilisant la matrice de Macvicar-Whelan.

Afin de préparer l'application de la technique du neuro-flou, il nous a fallu passer aux régulateurs de type de Sugeno qui sont simples en formules mais difficiles à avoir à partir des connaissances de l'expert. Pour cela, nous avons donné une méthode simple et efficace pour passer des régulateurs de type de Mamdani à ceux de type de Sugeno (méthode d'approximation).

Une fois les régulateurs de type de Sugeno sont synthétisés, l'application de l'algorithme TBP, nous a permis de bien placer les fonctions d'appartenance pour avoir de meilleures performances.

Le présent mémoire organisé en quatre chapitres, la commande floue et neuro-floue décentralisée et son application aux systèmes dynamiques interconnectés (robot).

Le premier chapitre est consacré à la modélisation géométrique, cinématique et dynamique du robot manipulateur et au problème de génération de trajectoires.

Dans le second chapitre, on présente les bases mathématiques de la logique floue, ainsi que son application dans la commande floue des systèmes. Aussi, on présente les différents types de régulateurs flous (Mamdani, Sugeno).

Le troisième chapitre est une introduction à une nouvelle technique, la commande neuro-floue, qui adapte les systèmes d'inférence floue pour pouvoir appliquer les techniques d'apprentissage utilisées pour les réseaux de neurones artificiels.

Le quatrième chapitre est consacré aux applications des différentes approches (floues (Mamdani, Sugeno), neuro-floue). Des comparaisons sont données au fur et à mesure, ainsi que des tests de robustesse.

Enfin, une conclusion générale et des perspectives sont présentées.

CHAPITRE I

MODELISATION

*"Il n'y a pas de problèmes, il n'y a que des solutions
l'esprit de l'homme invente ensuite les problèmes"*
ANDRE GIDE

Chapitre I : MODELISATION**I.1. INTRODUCTION:**

En général, la synthèse d'une commande d'un processus donné, passe toujours par une phase primordiale: c'est la modélisation. Elle consiste à établir un ensemble d'équations mathématiques entre les différents composants de ce dernier. Comme cas particulier, en robotique, elle consiste à établir plusieurs modèles: Géométriques, Cinématiques et Dynamiques; qu'il s'agit d'un modèle directe ou inverse, que nous allons illustrer dans ce chapitre.

I.2. PRESENTATION DU ROBOT:

Dans notre travail, nous avons opté pour un robot de classe 4, choisi parmi les 8 classes proposées par D.P. Stoten [17]. Il se caractérise par deux translations et une rotation.

Pour faciliter la modélisation du robot, on suppose que:

- 1- Les frottements sont de nature visqueuse et linéaires par rapport aux vitesses généralisées.
- 2- Les différentes liaisons sont rigides.
- 3- Les capteurs ont un gain unitaire et une dynamique négligeable.
- 4- Les actionneurs sont idéaux (la force généralisée est directement proportionnelle au signal de commande).

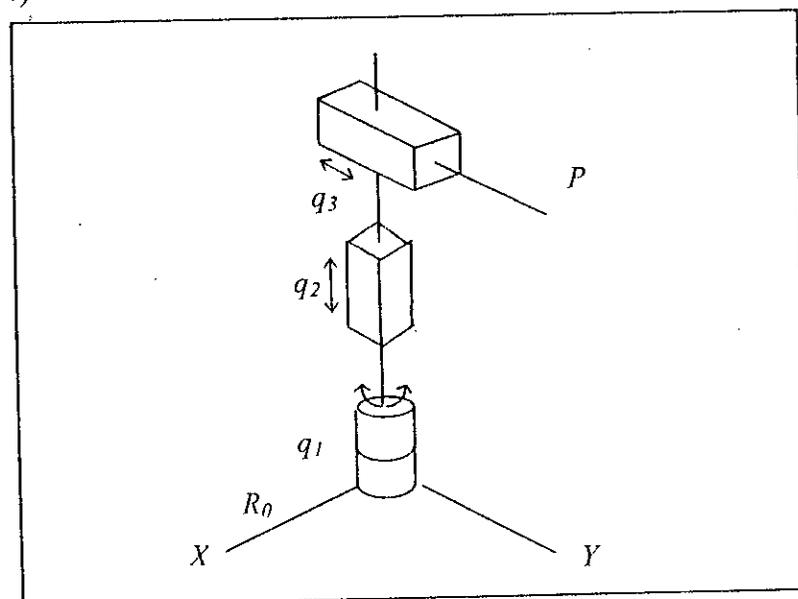


Figure (I-1). SCHEMA DU ROBOT DE CLASSE 4

1.4. MODELISATION GEOMETRIQUE :

Les transformations de DENAVIT-HARTENBERG permettent d'exprimer la position de l'organe terminal, par rapport au repère $R_0 (x_0, y_0, z_0)$ lié à la base, sous forme d'une fonction.

$$\underline{r}(R_0) = \underline{F}(\underline{q}); \quad \underline{r}^T = [x \quad y \quad z] \quad ; \quad \underline{q}^T = [q_1 \quad q_2 \quad q_3] \quad (I-5)$$

$$\underline{F}(\underline{q}) = T_3^0 \cdot \underline{r}_3^3 \quad (I-6)$$

$$\underline{r} = T_0^3 \cdot \underline{r}_3^3 = [-d_3 \sin \theta_1 \quad d_3 l_1 \quad d_2 \quad 1]^T \quad (I-7)$$

$$MGD = \begin{cases} x = -d_3 \sin \theta_1 \\ y = d_3 \cos \theta_1 \\ z = d_2 \end{cases} \quad (I-8)$$

Les équations (I- 8) forment le Modèle Géométrique Direct (MGD).

Le MGD permet, à partir des coordonnées articulaires, de trouver les coordonnées de l'organe terminal dans l'espace opérationnel du robot.

On tire, du MGD, les équations

$$MGI = \begin{cases} d_2 = z \\ \theta_1 = \text{Arctg}\left(\frac{-x}{y}\right) \\ d_1 = \frac{y}{\cos \theta_1} \end{cases} \quad \text{Pour } y \neq 0 \text{ et } \theta \neq \frac{\pi}{2} + k\pi ; k \in \mathbb{Z} \quad (I-9)$$

Les équations (I-9) forment le Modèle Géométrique Inverse (MGI).

1.5. MODELISATION CINEMATIQUE:

Pour remédier à la singularité du modèle géométrique inverse, on utilise le modèle cinématique, qui lie les variations des coordonnées articulaires et celles des coordonnées opérationnelles. Ce modèle est obtenu par linéarisation du modèle géométrique directe.

$$F_j(\underline{q}) = \sum_{k=0}^{\infty} \frac{1}{k!} \left[\sum_{i=1}^n (q_i - q_{i0}) \frac{\partial}{\partial q_i} \right]^k F_j(\underline{q}) \Big|_{\underline{q}_0} \quad ; j = 1, \dots, 4 \quad (I-10)$$

En se limitant au développement du 1 ordre dans la série de TAYLOR,

$$F_j(\underline{q}) = F_j(\underline{q}_0) + \sum_{i=1}^n \frac{\partial F_j(\underline{q})}{\partial q_i} \Big|_{\underline{q}_0} (q_i - q_{i0}) \quad (I-11)$$

avec :

$$\begin{aligned} \underline{r} - \underline{r}_0 &= [J](\underline{q} - \underline{q}_0) \quad , \\ \underline{q} - \underline{q}_0 &= [J]^{-1}(\underline{r} - \underline{r}_0) \end{aligned}$$

et

$$[J] = \begin{pmatrix} \frac{\partial F_1(\underline{q})}{\partial q_1} & \frac{\partial F_1(\underline{q})}{\partial q_2} & \frac{\partial F_1(\underline{q})}{\partial q_3} \\ \frac{\partial F_2(\underline{q})}{\partial q_1} & \frac{\partial F_2(\underline{q})}{\partial q_2} & \frac{\partial F_2(\underline{q})}{\partial q_3} \\ \frac{\partial F_3(\underline{q})}{\partial q_1} & \frac{\partial F_3(\underline{q})}{\partial q_2} & \frac{\partial F_3(\underline{q})}{\partial q_3} \end{pmatrix} \quad (I-12)$$

d'où :

$$\begin{pmatrix} \theta_1(t_i) - \theta_1(t_{i-1}) \\ d_2(t_i) - d_2(t_{i-1}) \\ d_3(t_i) - d_3(t_{i-1}) \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ -\cos[\theta_2(t_{i-1})] & -\sin[\theta_2(t_{i-1})] \\ d_3(t_{i-1}) & d_3(t_{i-1}) \\ -\sin[\theta_1(t_{i-1})] & d_3(t_{i-1})\cos[\theta_1(t_{i-1})] \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \begin{pmatrix} x(t_i) - x(t_{i-1}) \\ y(t_i) - y(t_{i-1}) \\ z(t_i) - z(t_{i-1}) \end{pmatrix} \quad (I-13)$$

1.6. GENERATION DE TRAJECTOIRES :

Afin de tester l'habilité de notre commande, nous imposerons au robot certaines trajectoires à poursuivre. Nous nous limiterons à deux : la trajectoire cycloïdale qui présente des continuités en position, en vitesse et en accélération; et la fenêtre de Viviani qui présente des discontinuités en vitesse et en accélération.

1.6.1. TRAJECTOIRE CYCLOIDALE:

Elle se présente sous la forme suivante:

$$r(t) = a + b.t + c.\sin kt \tag{I-14}$$

Les paramètres a, b, c et k seront déterminés comme suit :

$$\begin{cases} \dot{r}(0) = 0 \\ \dot{r}(t_f) = 0 \end{cases} \quad \text{et} \quad \begin{cases} r(0) = 0 \\ r(t_f) = 1 \end{cases} \Rightarrow \begin{cases} a = 0 \\ b = 1/t_f \\ c = -1/2\pi \\ k = 2\pi/t_f \end{cases} \tag{I-15}$$

Enfin, nous tirons :

$$r(t) = \frac{t}{t_f} - \frac{1}{2\pi} \sin \frac{2\pi}{t_f} t \tag{I-16}$$

$$\dot{r}(t) = \frac{1}{t_f} - \frac{1}{t_f} \cos \frac{2\pi}{t_f} t \tag{I-17}$$

1.6.2. FENETRE DE VIVIANI :

Cette trajectoire est l'intersection d'un cylindre et d'une sphère, décrits par les équations suivantes :

$$x^2 + \left(y - \frac{R}{2}\right)^2 = \left(\frac{R}{2}\right)^2 \quad \text{et} \quad x^2 + y^2 + z^2 = R^2 \tag{I-18}$$

En posant :

$$\begin{cases} x = \rho \cos \theta \\ y = \rho \sin \theta \end{cases} \quad \text{d'où:} \quad \begin{cases} \rho = R \sin \theta \\ z = |R \cos \theta| \end{cases} \tag{I-19}$$

I.7. MODELISATION DYNAMIQUE:

Dans cette section, on exposera une approche systématique pour résoudre le problème de la modélisation du robot [30] [18] [17]. Celle-ci consiste à utiliser le formalisme d' Euler-Lagrange qui s'énonce comme suit:

$$\frac{d}{dt} \left(\frac{\partial E_c}{\partial \dot{q}_i} \right) - \frac{\partial E_p}{\partial q_i} + \frac{\partial E_d}{\partial \dot{q}_i} = \eta_i \quad i = \overline{1, n} \quad (I-20)$$

avec :

- E_c : Energie cinétique.
- E_p : Energie potentielle.
- E_d : Energie de dissipation.
- n : Degré de liberté.

et

$$E_p = (m_1 + m_3 + m_0)q_2g \quad (I-21)$$

$$E_c = \frac{1}{2} \left\{ (m_1 + m_3 + m_0) \dot{q}_2^2 + (m_3 + m_0) \dot{q}_3^2 + \left(J_1 + J_3 + m_3 \left(q_3 - \frac{l_2}{2} \right)^2 + m_0 \dot{q}_3^2 \right) \dot{q}_1^2 \right\} \quad (I-22)$$

$$E_d = \frac{1}{2} \left[f_2 \dot{q}_1^2 + f_4 \dot{q}_2^2 + f_3 \dot{q}_3^2 \right] \quad (I-23)$$

En appliquant (I-20), on trouve :

$$\eta_1 = \left(J_1 + J_3 + m_3 \left(q_3 - \frac{l_2}{2} \right)^2 + m_0 q_3^2 \right) \ddot{q}_1 + 2.m_3 \left(q_3 - \frac{l_2}{2} \right) \dot{q}_1 \dot{q}_3 + 2.m_0 q_3 \dot{q}_1 \dot{q}_3 + f_4 \dot{q}_1$$

$$\eta_2 = (m_1 + m_3 + m_0) (\ddot{q}_2 + g) + f_4 \dot{q}_2 \quad (I-24)$$

$$\eta_3 = (m_3 + m_0) \ddot{q}_3 - m_3 \left(q_3 - \frac{l_2}{2} \right) \dot{q}_1^2 - m_0 q_3 \dot{q}_1^2 + f_3 \dot{q}_3$$

Le modèle sous la forme d'état est le suivant :

$$\begin{aligned} \dot{\underline{X}} &= A\underline{X} + B\underline{U} + \underline{D} \\ \underline{Y} &= C\underline{X} \end{aligned} \tag{I-25}$$

avec :

$$\underline{X}^T = [x_1 \ x_2 \ x_3 \ x_4 \ x_5 \ x_6] = \begin{bmatrix} q_2 & \dot{q}_2 & q_1 & \dot{q}_1 & q_3 & \dot{q}_3 \end{bmatrix}$$

$$\underline{U}^T = [u_1 \ u_2 \ u_3]: \text{Vecteur de commande.}$$

$$\underline{Y}^T = [y_1 \ y_2 \ y_3] = [q_1 \ q_2 \ q_3]$$

D = Bloc non linéaire .

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & \frac{-f_1}{m_1 + m_3 + m_0} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \frac{-f_2}{J_S} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & \frac{-f_3}{m_3 + m_0} \end{pmatrix}; \quad B = \begin{pmatrix} 0 & 0 & 0 \\ \frac{K_1}{m_1 + m_3 + m_0} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & \frac{K_2}{J_S} & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \frac{K_3}{m_3 + m_0} \end{pmatrix}$$

$$C = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}; \quad J_S = J_1 + J_3 + m_3 \left(x_5 - \frac{l_2}{2} \right)^2 + m_0 x_5^2$$

$$\underline{D}^T = \left[0 \quad -g \quad 0 \quad -2 \cdot \frac{m_3}{J_S} \left(x_5 - \frac{l_2}{2} \right) x_4 x_6 - \frac{2}{J_S} m_0 x_4 x_5 x_6 \quad 0 \quad \frac{m_3}{m_3 + m_0} \left(x_5 - \frac{l_2}{2} \right) x_4^2 + \frac{m_0 x_4^2 x_5}{m_3 + m_0} \right]$$

Les caractéristiques du robot sont :

$m_1 = 20 \text{ kg}$	$m_3 = 10 \text{ kg}$	$l_2 = 1.5 \text{ m}$	$g = 9.81 \text{ m.s}^{-1}$
$f_1 = 30 \text{ N.s.m}^{-1}$	$f_2 = 7.825 \text{ Nm.s.rd}^{-1}$	$f_3 = 20 \text{ N.s.m}^{-1}$	
$K_1 = 100 \text{ N.V}^{-1}$	$K_2 = 10 \text{ N.V}^{-1}$		$K_3 = 10 \text{ N.V}^{-1}$
$J_1 = 0.2 \text{ kg.m}^2$		$J_3 = 2 \text{ kg.m}^2$	

TABLEAU (I-2). Les caractéristiques du robot

En utilisant un pas de 10^{-3} sec, la simulation en boucle ouverte par RK45 a été effectuée.

Les résultats de La simulation sont représentés par les figures de la page suivante .

1.8. CONCLUSION

- La modélisation du robot est une phase indispensable pour la commande, car elle permet de voir les différents couplages entre ses grandeurs et leurs influences sur son comportement dynamique.
- Le formalisme d ' EULER-LAGRANGE a donné un modèle non linéaire.
- La simulation en boucle ouvert nous a permis de voir le comportement dynamique du robot.

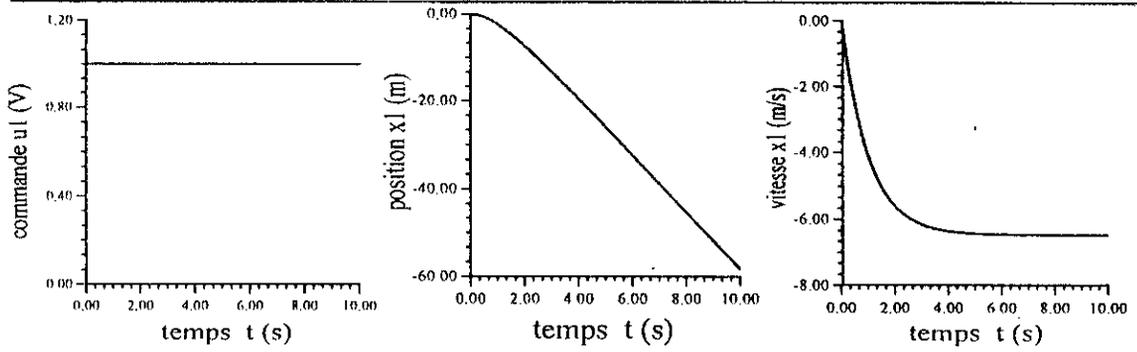


Figure (I-3.a) . Réponse indicielle de l'articulation 2.

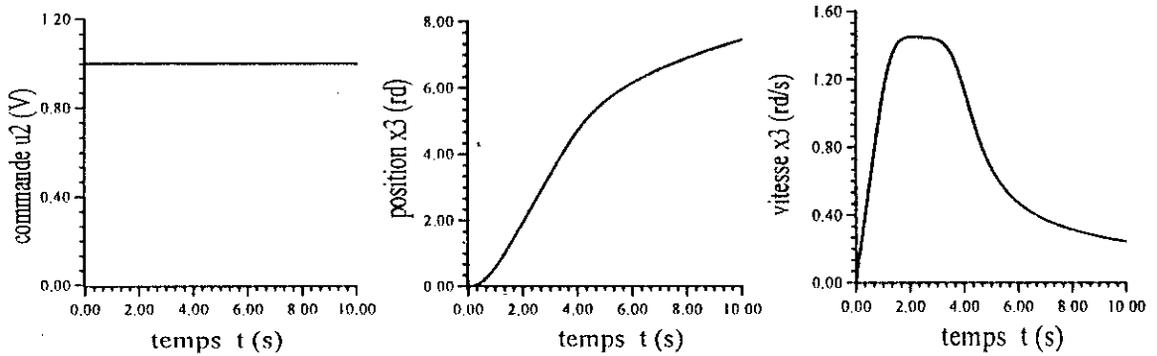


Figure (I-3.b) . Réponse indicielle de l'articulation 1.

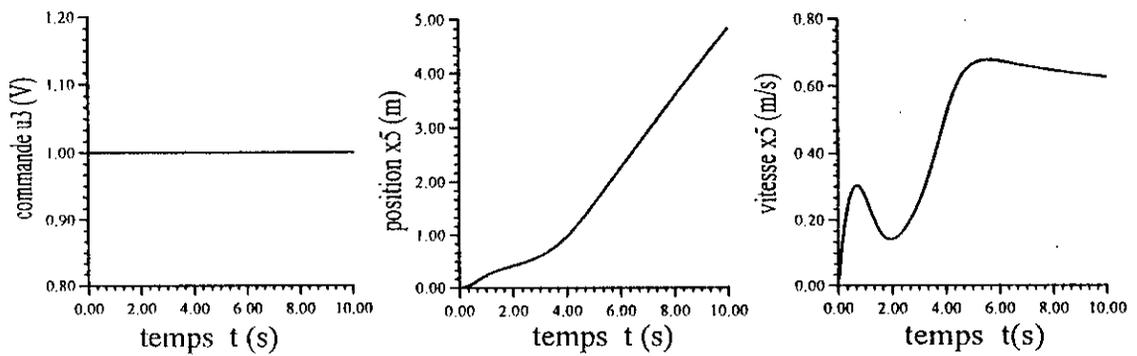


Figure (I-3.c) . Réponse indicielle de l'articulation 3.

CHAPITRE II

COMMANDE PAR LOGIQUE FLOUE

"L'evidence paralyse la demonstration"
PIERE REVERDR

II.2.1. VARIABLE LINGUISTIQUE :

L'idée de base des sous-ensembles flous est d'exprimer, en termes mathématiques, les attributs qualificatifs du langage commun (petit, grand,...)[6]. Dans ce but, on introduit la notion des variables linguistiques. Contrairement à une variable numérique, dont la valeur est un nombre, la valeur d'une variable linguistique est un terme du langage humain, qu'on appelle valeur floue. La représentation mathématique d'une valeur floue est réalisée par l'application de la notion des sous-ensembles flous. Une valeur floue peut être représentée par une fonction d'appartenance. La fonction d'appartenance peut être interprétée comme un moyen d'exprimer le degré d'incertitude du langage humain. Cependant, il ne s'agit pas d'une incertitude sur une grandeur, mais d'une incertitude sur la valeur floue que peut prendre la variable linguistique associée à cette grandeur.

II.2.2. OPERATIONS SUR LES SOUS-ENSEMBLES FLOUS :

Afin de manipuler les sous-ensembles flous, il est nécessaire de définir certaines opérations élémentaires[1][2][3][23].

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

II.2.2.1. EGALITE :

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

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

II.2.2.2. INCLUSION:

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

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

II.2.2.3 COMPLEMENT:

Le complément A^c de A est défini par

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

II.2.2.4 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 général, l'opération Min. C'est une norme triangulaire.

II.2.2.5 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 général, l'opération Max. C'est une conorme triangulaire.

II.2.2.6 PRODUIT CARTESIEN:

Soient 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, X_2, \dots, X_n , ayant pour fonction d'appartenance:

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

II.2.2.7 RELATION FLOUE:

Soient 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 comme suit:

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

II.3 LOGIQUE FLOUE:

Elle n'est autre que l'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 notre exposé, on donne quelques définitions [3]:

Soient X une variable floue et A une caractéristique.

A) LA PROPOSITION:

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

B) LA CONJONCTION:

La conjonction de deux propositions est réalisée par l'opérateur et. Par exemple,

"X₁ est A₁ et X₂ est A₂"

C) LA DISJONCTION:

La disjonction de deux propositions est réalisée par l'opérateur ou. Par exemple,

"X₁ est A₁ ou X₂ est A₂"

D) L'IMPLICATION:

L'implication est définie comme suit :

"si X₁ est A₁ alors X₂ est A₂"

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 POSITIVE".

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 DECISIONS:

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
 μ_A μ_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
 μ_A μ_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:

L'objectif de la commande par logique floue consiste à piloter un processus en utilisant les connaissances et l'expérience d'un opérateur humain qualifié "EXPERT". Les connaissances de l'expert sont schématisées à l'aide d'un ensemble de règles de conduite basées sur un vocabulaire symbolique[7][8].

Dans ce qui suit, on montre comment un régulateur, dont la loi de commande est basée sur des règles linguistiques, peut être synthétisé.

II.6.1. STRUCTURE GENERALE D'UN REGULATEUR FLOU:

Comme le montre la figure (II-1), le régulateur flou se compose de quatre blocs :

- Fuzzification
- Inférence
- Base de règles
- Defuzzification

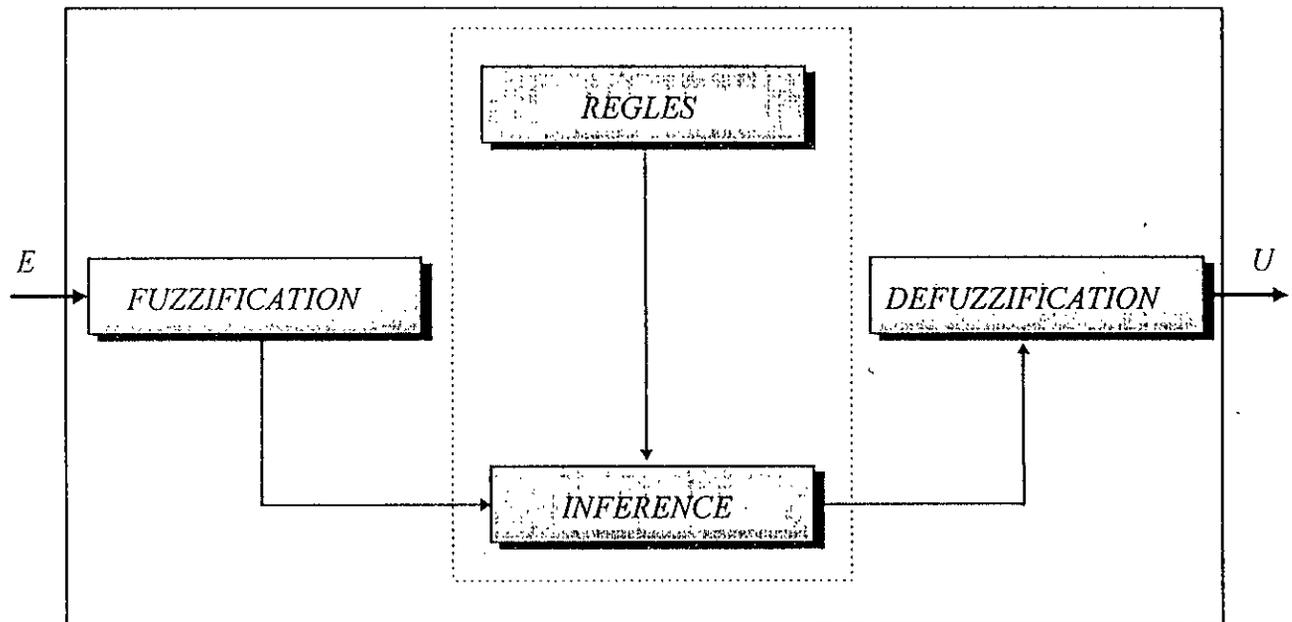


Figure (II-1) . STRUCTURE GENERALE D'UN REGULATEUR FLOU.

II.6.1.1. FUZZIFICATION:

C'est l'étape qui permet de transformer les grandeurs mesurées sur le processus en un ensemble de valeurs floues[5].

II.6.1.2. INFERENCE:

C'est un mécanisme de décision qui, en manipulant le bloc des règles, établit une décision. Il existe plusieurs stratégies d'inférence[5].

Exemple:

Mamdani	(Max. , Min.)
Larsen	(Max. , Prod.)
Sugeno	(Somme , Prod.)

II.6.1.3.DEFUZZIFICATION:

Elle consiste à obtenir les valeurs numériques de la commande à partir de l'ensemble flou obtenu par le moteur d'inférence. Il existe plusieurs stratégies de defuzzification [1].

II.6.1.3.1.METHODE DU MAXIMUM:

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

II.6.1.3.2.METHODE DE LA MOYENNE DES MAXIMA:

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

II.6.1.3.3.METHODE DU CENTRE DE GRAVITE:

Elle donne comme valeur le centre de gravite de l'ensmble flou.

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

II.6.1.4.BLOC DE REGLES:

Un système est caractérisé par un ensemble de règles, dont les sources sont:

- l'expertise et l'expérience des opérateurs formulées en termes de règles,
- l'observation et la modélisation du comportement et des actions des opérateurs,
- l'utilisation des méthodes d'apprentissage notamment avec l'introduction de la technique neuronale qui est l'objet du chapitre suivant.

Cependant, pour des problèmes élémentaires de régulation et de poursuite, l'écriture des règles devient plus facile en se basant sur :

- une analyse quantitative du comportement dans le plan de phase des trajectoires position-vitesse,
- une extension du principe de la commande par mode de glissement.

II.6.2. TYPES DE REGULATEURS FLOUS:

Il existe plusieurs types de régulateurs, parmi eux on cite ceux de Mamdani, de Sugeno, de Larsen et de Tsukamoto...

II.6.2.1. REGULATEUR FLOU DU TYPE DE MAMDANI:

Les régulateurs de type de Mamdani utilisent des règles à prémisses et conclusions symboliques, l'inférence (max., min.) et la défuzzification par centre de gravité. Exemple:

si "e est NG" et "de est PS" alors "la commande est GR"

II.6.2.2. REGULATEUR DU TYPE DE SUGENO:

Les régulateurs de type de sugeno reposent sur des règles, dont les prémisses sont symboliques, tandis que les conclusions sont numériques[29]. Exemple:

Si "e est NG" et "de est NG" alors $U=1+3e-2de$

II.6.2.3. REGULATEUR DU "DEUXIEME ORDRE":

Les méthodes décrites ci-dessus ont été développées pour le cas général d'un régulateur multi-entrées. Théoriquement le nombre d'entrées n'est pas limité[7][8]. En pratique, cependant, il n'est pas rationnel d'utiliser plus de trois variables d'entrée puisque la détermination des règles devient trop complexe. Dans le cas d'un processus d'ordre élevé, on utilisera plutôt une technique qui consiste à décomposer le régulateur en blocs de commande à deux entrées (décentralisation de la commande).

Cette technique sera utilisée dans le cadre du quatrième chapitre, où on appliquera la commande floue à un robot de classe 4. Ici on présente un modèle d'un bloc de commande à deux entrées, que l'on rencontre souvent dans le domaine de la logique floue.

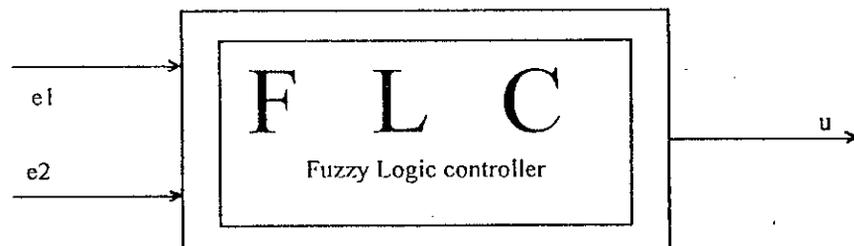


Figure (II-2) .REGULATEUR DU DEUXIEME ORDRE.

Comme vocabulaire des variables linguistiques on peut choisir par exemple un vocabulaire à sept termes (classes) pour e_1, e_2 et U : NB, NM, NS, Z, PS, PM, PB. L'étape suivante consistera à définir les sous-ensembles flous par leurs fonctions d'appartenances et à fixer les gains de normalisation [13][14][15].

REMARQUES:

- Les régulateurs de deuxième ordre utilisent souvent l'erreur e et sa dérivée \dot{e} .
- Si on applique directement la sortie du régulateur au processus, la commande sera dite de type PD flou.
- Si on fait passer la sortie du régulateur dans un intégrateur, la commande sera dite PI flou.

II.7 CONCLUSION:

la commande par logique floue est riche en approches, en types de régulateurs et en types d'actions. Ceci permet à l'automaticien de choisir en fonction de son cahier des charges (régulation, poursuite, rapidité, robustesse) le régulateur adéquat.

CHAPITRE III

COMMANDE NEURO-FLOU

*"Ce qui n'est pas connu se laisse comprendre par ce
qui est connu"*

ABOU ALI HOSSAYN IBN SINA

III.1. INTRODUCTION:

Les réseaux de neurones artificiels (RNA) jouent un grand rôle dans la commande et le diagnostique, car ils peuvent représenter, d'une manière approximative, n'importe quelle fonction non linéaire. Ces réseaux ne sont qu'un cas particulier des réseaux dits Réseaux Adaptatifs Non linéaires (RAN). Ces derniers sont inspirés du domaine de la biologie (cerveau,...)[34]. Les réseaux de neurones sont des approximateurs universels de fonctions, mais on ne possède aucun critère sur leurs tailles et topologies (nombres de couches, nombres de neurones, ...).

Dans l'autre partie du traitement distribué de l'information, on trouve la logique floue (LF) qui gagne des propriétés du raisonnement approché. Comme avec les RNA, on peut représenter, d'une manière approchée, des fonctions par des règles floues, en faisant un bon choix des sous-ensembles flous et des fonctions d'appartenance associées. Malgré cela, on ne sait pas encore si les approximateurs flous sont universels ou non [5].

Les recherches actuelles sont orientées vers l'utilisation des RAN et de la LF ensemble, en les associant dans un tout appelé Réseau Neuro-Flou. Ceci, afin de tirer profit des capacités de l'apprentissage des RAN et de celles du raisonnement approché de la LF, et de minimiser les inconvénients de chaque méthode utilisée seule.

Certains chercheurs ont étudié la réalisation du matériel qui fonctionne comme régulateur neuro-flou, en combinant le matériel déjà réalisé pour les régulateurs par logique floue RLF et pour les réseaux de neurones RNA.

III.2. RESEAUX ADAPTATIFS GENERALISES:**III.2.1. DEFINITION:[34]**

Les réseaux adaptatifs non linéaires RAN sont des systèmes constitués d'un grand nombre d'unités élémentaires appelées cellules telles que:

- ces unités forment un réseau qui traite l'information de manière parallèle et distribuée.
- ces unités interagissent de manières non linéaires et compétitives dans un certain environnement.

- des opérateurs modifient l'état des unités et leurs interactions de façon que le système globale s'adapte progressivement à l'environnement qui n'est pas forcément fixe.

III.2.2. ARCHITECTURES DES RESEAUX ADAPTATIFS GENERALISES:

Dans le cas général, un réseau adaptatif se présente comme un ensemble de nœuds et de liens.[16]

III.2.2.1. LES NŒUDS:

Les nœuds réalisent des fonctions, c'est à dire, leurs sorties sont des fonctions en leurs entrées. Ces fonctions peuvent être à paramètres ajustables ou sans paramètres ajustables.

Les nœuds, dont les paramètres sont ajustables, seront représentés par des carrés, alors que les autres par des cercles. En changeant les paramètres des nœuds, on change les caractéristiques du réseau adaptatif. Généralement, les fonctions des nœuds sont différentiables.

III.2.2.2. LES LIENS:

Ils sont, tout simplement, des arcs orientés donnant le sens d'écoulement de l'information. Ils ne possèdent pas de paramètres de pondération comme dans le cas des RNA.

III.2.2.3. PROPRIETES:

Les paramètres des RAN sont localisés dans les nœuds. Chaque nœud possède un ensemble local de paramètres, ajustables ou non. L'union de ces ensembles locaux de paramètres est, par définition, l'ensemble des paramètres du RAN.

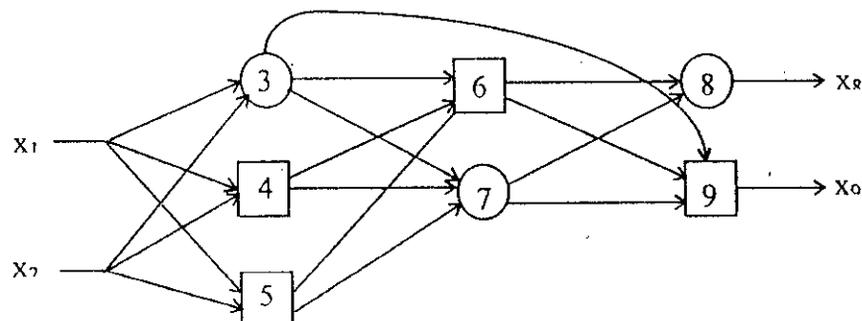


Fig. (III-1) . Exemple d'un RAN.

III.2.3. CATEGORIES DES RAN:

On trouve deux catégories :

FEED FORWARD : pas de flèches (arcs orientés) d'un nœud vers les nœuds précédents.

RECURRENT : il existe au moins une flèche orientée d'une couche à une autre couche précédente.

III.2.4. REPRESENTATION:

On distingue deux types de représentation:

III.2.4.1. REPRESENTATION EN COUCHE:

Les sorties d'une couches, dans ce type de représentation, sont connectées aux entrées des couches qui la suivent. Cette représentation est préférée car:

- les noeuds de chaque couche possèdent la même fonction aux paramètres près.
- les entrées des nœuds d'une couche sont le résultat du même niveau d'abstraction sur l'entrée initiale.

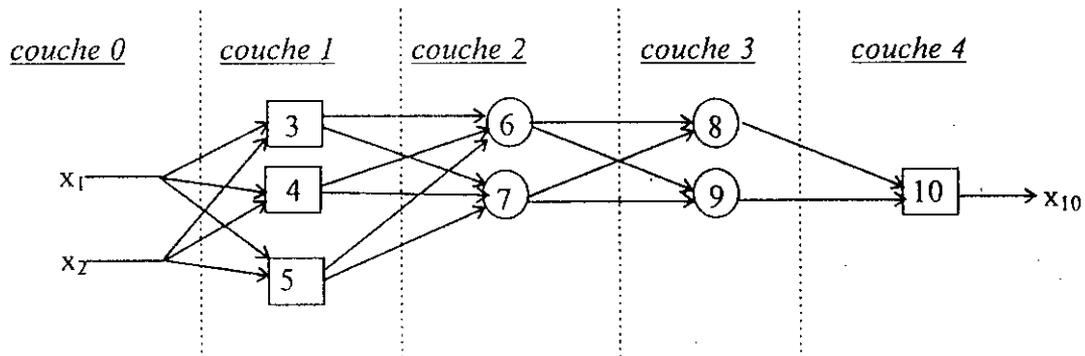


Fig. (III-2). Représentation en couche.

III.2.4.2. REPRESENTATION EN ORDRE TOPOLOGIQUE:

Dans cette représentation les nœuds sont ordonnés en séquence 1,2,..., et elle possède les propriétés suivantes :

- il n'y a pas de lien du nœud i au nœud j tant que $i \geq j$,
- l'algorithme d'apprentissage est facile à obtenir.

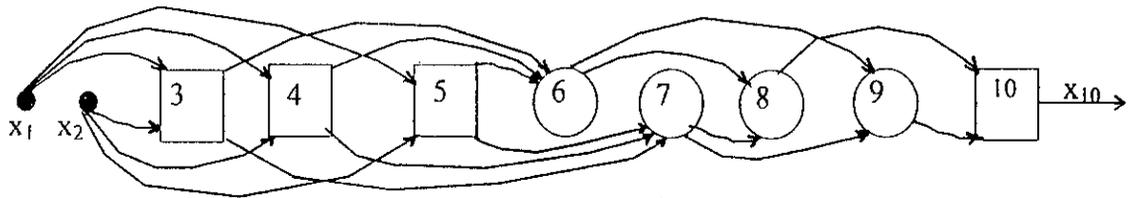


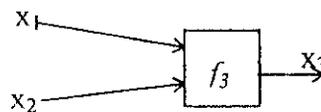
Fig. (III-3). Représentation en ordre topologique.

III.2.5. EXEMPLES:

1. Soit la fonction suivante

$$x_3 = f_3(x_1, x_2; a_1, a_2, a_3) = a_1x_1 + a_2x_2 + a_3 \tag{III-1}$$

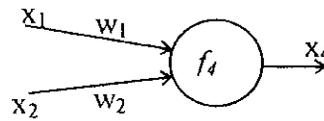
On peut représenter cette fonction comme suit



avec

f_3 la fonction du nœud dont les paramètres sont a_1, a_2, a_3 .

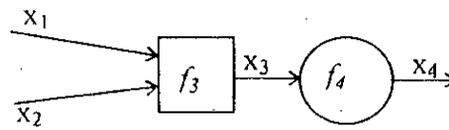
2. Soit le neurone artificiel suivant



On a

$$x_4 = f_4\left(\sum_{i=1}^2 w_i x_i\right) , \quad f_4(x) = \frac{1}{1 + e^{-x}} \quad (\text{III-2})$$

On peut représenter ce neurone artificiel comme suit



avec

$$x_3 = f_3(x_1, x_2; w_1, w_2) = w_1 x_1 + w_2 x_2 , \quad x_4 = f_4(x_3) = \frac{1}{1 + e^{-x_3}} \quad (\text{III-3})$$

w_1, w_2 : paramètres

On voit bien, que les RN sont un sous-ensemble des RAN.

III.3. RESEAU NEURO-FLOU:

Dans ce qui suit, on va présenter la mise d'un système d'inférence floue sous la forme d'un réseau adaptatif non linéaire. La représentation, ainsi obtenue, est appelée système d'inférence floue basé sur les réseaux adaptatifs (ANFIS : Adaptive Network-based Fuzzy Inference System).

Il existe plusieurs approches pour réaliser un système d'inférence floue. Chaque approche a son image dans la représentation par des RAN. On va présenter deux cas:

- approche de Mamdani,
- approche de Sugeno

Une description sera donnée, suivie par l'algorithme d'apprentissage qui n'est rien d'autre qu'une version adaptée de l'algorithme de Back-propagation.

III.3.1. METHODE DE MAMDANI [24]:

C'est la plus simple sous sa forme de règles, mais elle fastidieuse à mettre sous forme de RAN.

III.3.1.1. DESCRIPTION:

Comme le montre la figure (III-4), il y a cinq couches. Chaque couche représente une partie du système d'inférence floue.

couche1: Elle est formée de nœuds fixes. Les nœuds ont la même fonction

$$O_{1i} = f_1(x_{1i}) = x_{1i} \quad (\text{III-4})$$

O_{1i} : sortie du nœud (i) de la couche 1

x_{1i} : entrée du nœud (i) de la couche 1

f_1 : fonction des nœuds de la couche 1.

Cette couche distribue l'information de l'entrée.

couche2: Elle est constituée par des nœuds à paramètres ajustables. Ces nœuds représentent les fonctions d'appartenance associées aux valeurs floues prises par les variables linguistiques, qui apparaissent dans les prémisses. Chaque nœud a comme fonction

$$y = f_2(x; m, \sigma) = e^{-\left\{\frac{x-m}{\sigma}\right\}^2} \quad (\text{III-5})$$

couche3: Les nœuds de cette couche sont fixes, et réalisent des fonctions simulant le moteur d'inférence. Chaque nœud possède la fonction

$$y = f_3(x_1, x_2, \dots, x_n) = \min(x_1, x_2, \dots, x_n) \quad (\text{III-6})$$

couche4: Cette couche est un peu spéciale, car elle fonctionne en deux modes:

mode1: Elle réalise l'agrégation, donc chaque nœud possède la fonction

$$y = f_4(x_1, x_2, \dots, x_n) = \min\left(1, \sum_{i=1}^n x_i\right) \quad (\text{III-7})$$

mode2: Elle sert pour l'apprentissage, et elle reçoit ses entrées de la couche 5.

couche5: Il existe deux types de nœuds dans cette couche.

type1: Les nœuds de ce type réalisent la défuzzification et possèdent la fonction:

$$y = f_5((x_1, x_2, \dots, x_n); (m_1, m_2, \dots, m_n); (\sigma_1, \sigma_2, \dots, \sigma_n)) = \frac{\sum_{i=1}^n m_i \sigma_i x_i}{\sum_{i=1}^n m_i \sigma_i} \quad (\text{III-8})$$

type2: Les nœuds de ce type servent à l'apprentissage. Ils reçoivent les sorties désirées et les transmettent à la couche 4.

Ils possèdent la fonction

$$y = f_5(y_d) = y_d \quad (\text{III-9})$$

Avec

- y : sortie du noeud
- x_1, \dots, x_n : entrées du noeud
- y_d : sortie désirée
- $m, \sigma, m_1, \sigma_1, \dots, m_n, \sigma_n$: paramètres

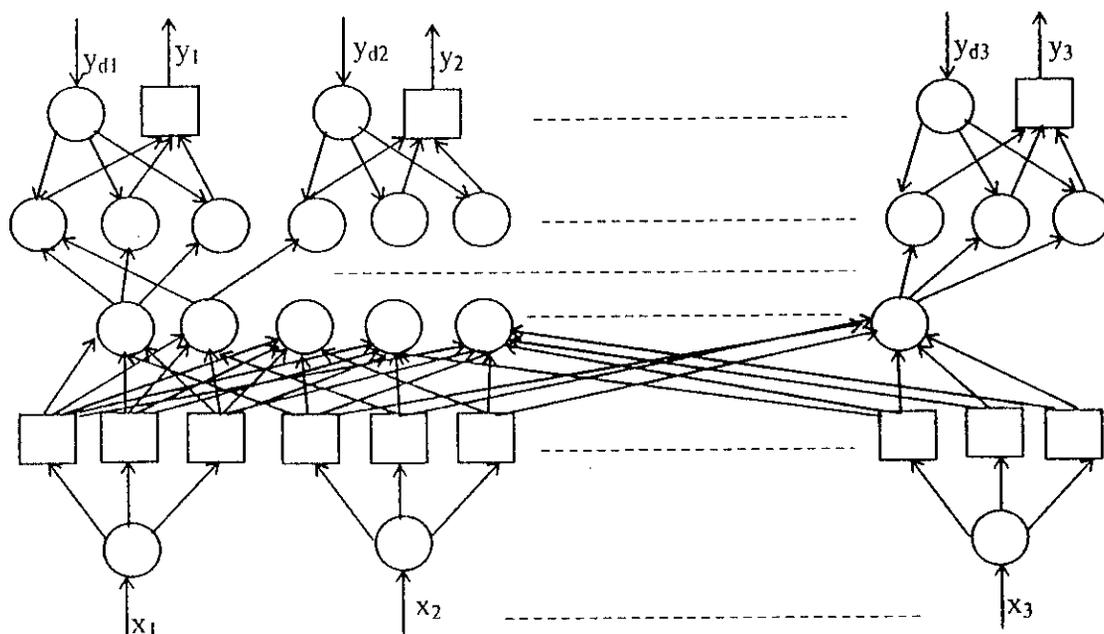


Fig. (III-4). Système de règles floues sous la forme d'un RAN (Mamdani).

III.3.1.2. APPRENTISSAGE:

Après la classification des données de l'apprentissage, on utilise l'algorithme de back-propagation, qui consiste à minimiser l'erreur

$$E = \frac{1}{2} (y_d - y)^2 \tag{III-10}$$

avec

y_d : sortie désirée

y : sortie réelle

L'algorithme est le suivant:

$$E = \frac{1}{2}(y_d - y)^2 \quad (\text{III-11})$$

couche 5:

$$m_i(t+1) = m_i(t) + \eta(y_d - y) \frac{\sigma_i x_i}{\sum_{j=1}^n \sigma_j x_j}, \quad i = \overline{1, k}$$

$$\sigma_i(t+1) = \sigma(t) + \eta(y_d - y) \frac{m_i x_i \left(\sum_{j=1}^n \sigma_j x_j \right) - \left(\sum_{j=1}^n m_j \sigma_j x_j \right) x_i}{\left(\sum_{j=1}^n \sigma_j x_j \right)^2}, \quad i = \overline{1, k} \quad (\text{III-12})$$

$$\delta^5 = (y_d - y)$$

couche 4:

$$\delta^4_i = (y_d - y) \frac{m_i \sigma_i \left(\sum_{j=1}^n \sigma_j x_j \right) - \left(\sum_{j=1}^n m_j \sigma_j x_j \right) \sigma_i}{\left(\sum_{j=1}^n \sigma_j x_j \right)^2} \quad (\text{III-13})$$

couche 3:

$\delta_i^3 = \delta_i^4$ pour une seule sortie (SISO),

$$\delta_i^3 = \sum_k \delta_k^4 \text{ pour plusieurs sorties (MIMO).} \quad (\text{III-14})$$

couche 2:

$$\frac{\partial E}{\partial f_i} = \sum_k q_k \quad (\text{III-15})$$

La sommation s'effectue sur les nœuds de règles alimentés par f_i^2 .

$$q_k = \begin{cases} \delta_k^3 & \text{si } f_i \text{ est le minimum à l'entrée d'un nœud (k) de la couche (3),} \\ 0 & \text{sinon.} \end{cases}$$

L'ajustement des paramètres:

$$\begin{aligned} m_{ij}(t+1) &= m_{ij}(t) - \eta \frac{\partial E}{\partial f_i} f_i \frac{2(x_i - m_{ij})}{\sigma_{ij}^2} \\ \sigma_{ij}(t+1) &= \sigma_{ij}(t) - \eta \frac{\partial E}{\partial f_i} f_i \frac{2(x_i - m_{ij})^2}{\sigma_{ij}^3} \end{aligned} \quad (\text{III-16})$$

avec δ^i : erreur à transmettre de la couche (i) vers la couche (i-1)

$$\delta^i = -\frac{\partial E}{\partial f_i} \quad (\text{III-17})$$

f_i : fonction du nœud de la couche (i).

REMARQUE:

x_i est l'entrée du nœud (i) d'une couche, elle est calculée par le passage en avant de l'algorithme.

III.3.2. METHODE DE SUGENO [12][16]:

L'obtention des règles floues sous la forme de Sugeno est difficile. Mais, la représentation sous la forme d'un RAN est très simple.

III.3.2.1. DESCRIPTION:

Le réseau est formé de cinq couches. On montre la forme d'un tel réseau par l'exemple de la figure (III-5). Cet exemple présente les deux règles floues, sous la forme de Sugeno, suivantes :

SI ((x est A_1) ET (y est B_1)) ALORS ($z = p_1x + q_1y + r_1$)
 SI ((x est A_2) ET (y est B_2)) ALORS ($z = p_2x + q_2y + r_2$)

couche 1: Chaque nœud de cette couche possède des paramètres ajustables. La fonction du nœud est identique à la fonction d'appartenance à un sous-ensemble flou de l'univers de discours des entrées.

$$O_i^1 = f_i^1(x) = \mu_{A_i}(x) = w_i^1 \quad (\text{III-18})$$

avec

x : l'entrée du nœud

A_i : valeur linguistique associée à μ_{A_i}

On prend souvent

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

ou

$$\mu_{A_i}(x) = \frac{1}{1 + \left[\left(\frac{x - c_i}{a_i} \right)^2 \right]^{b_i}} \quad (\text{III-20})$$

avec

$\{a_i, b_i, c_i\}$: paramètres du nœud (1,i).

couche 2: Les nœuds de cette couche sont fixes, et ils possèdent les fonctions

$$O_i^2 = f_i^2(O_i^1, O_{i+2}^1) = f_i^2(w_i^1, w_{i+2}^1) = w_i^1 \cdot w_{i+2}^1 = w_i^2 \quad (\text{III-21})$$

couche 3: Les nœuds de cette couche sont fixes, et ils possèdent les fonctions

$$O_i^3 = f_i^3(O_i^2, O_2^2) = f_i^3(w_i^2, w_2^2) = \frac{w_i^2}{w_i^2 + w_2^2} = w_i^3 \quad (\text{III-22})$$

couche 4: Chaque nœud de cette couche est ajustable, et a pour fonction

$$O_i^4 = f_i^4(O_i^3, O_2^3; x, y) = f_i^4(w_i^3, w_2^3; x, y) = w_i^3 \cdot (p_i x + q_i y + r_i) = w_i^4 \quad (\text{III-23})$$

avec $\{p_i, q_i, r_i\}$: paramètres ajustables

couche 5: C'est un nœud unique, fixe et sert comme sommateur.

$$O_1^5 = f_1^5(O_1^4, O_2^4) = f_1^5(w_1^4, w_2^4) = w_1^4 + w_2^4 = w_1^5 = z \quad (\text{III-24})$$

z : la sortie

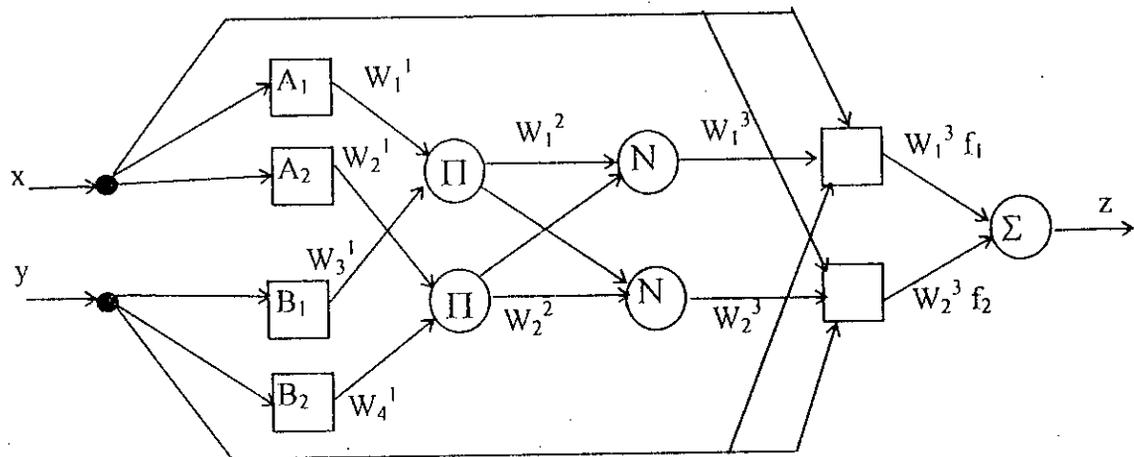


Fig. (III-5). Système de règles floues sous la forme d'un RAN (Sugeno).

III.3.2.2. APPRENTISSAGE:

Supposons qu'on a un RAN avec L couches, et n_k nœuds dans la couche k . Soit O_i^k la fonction et aussi la sortie du nœud (k, i) . La sortie d'un nœud dépend des entrées de ce nœud ainsi que de ses paramètres,

$$O_i^k = f_i^k(O_1^{k-1}, \dots, O_{n_{k-1}}^{k-1}; a, b, \dots, c) \quad (\text{III-25})$$

a, b, \dots, c : paramètres du nœud (k, i) .

L'apprentissage se fait par l'algorithme de back-propagation en minimisant l'erreur E ,

$$E = \sum_{p=1}^P E_p \tag{III-26}$$

$$E_p = \sum_{m=1}^{n_L} (T_{m,p} - O_{m,p}^L)^2, \quad 1 \leq p \leq P$$

avec

- P : nombres d'exemples à apprendre,
- $T_{m,p}$: composante de la sortie désirée du nœud (L, m) pour l'exemple (p) ,
- $O_{m,p}^L$: composante de la sortie réelle du nœud (L, m) pour l'exemple (p) .

La procédure de descente du gradient est la suivante:

- Pour chaque nœud, calculer

$$\frac{\partial E_p}{\partial O} \tag{III-27}$$

- Pour un nœud (L, i) , $i=1, \dots, n_L$,

$$\frac{\partial E_p}{\partial O_{i,p}^L} = -2(T_{i,p} - O_{i,p}^L) \tag{III-28}$$

- Pour les nœuds internes (k, i) , $1 \leq k \leq L-1, \quad i=1, \dots, n_k$

$$\frac{\partial E_p}{\partial O_{i,p}^k} = \sum_{m=1}^{n_{k+1}} \frac{\partial E_p}{\partial O_{m,p}^{k+1}} \cdot \frac{\partial O_{m,p}^{k+1}}{\partial O_{i,p}^k} \tag{III-29}$$

- Si α est un paramètre ajustable, alors on aura

$$\frac{\partial E_p}{\partial \alpha} = \sum_{O^* \in S} \frac{\partial E_p}{\partial O^*} \cdot \frac{\partial O^*}{\partial \alpha} \quad (\text{III-30})$$

avec

S : l'ensemble des nœuds dont les sorties dépendent de α de manière directe.

- On tire

$$\frac{\partial E}{\partial \alpha} = \sum_{p=1}^P \frac{\partial E_p}{\partial \alpha} \quad (\text{III-31})$$

- L'ajustement de α

$$\alpha(t+1) = \alpha(t) - \eta \frac{\partial E}{\partial \alpha} \quad (\text{III-32})$$

avec

η : le pas d'apprentissage

- Le pas η peut être calculé à partir de

$$\eta = \frac{s}{\sqrt{\sum_{\alpha} \left(\frac{\partial E}{\partial \alpha} \right)^2}} \quad (\text{III-33})$$

avec

s : coefficient choisi pour varier la vitesse de convergence.

CHAPITRE IV

APPLICATION

"Tout est dit... tout reste à faire"
LUDWIG VAN BEETHOVEN

Chapitre IV: APPLICATION*IV.1. INTRODUCTION:*

La conception et la synthèse d'une commande d'un système doivent répondre à un certain nombre de critères de performances et d'objectifs à atteindre à savoir la rapidité, la précision, ...

Ce travail a pour objectif, la synthèse d'un régulateur à base de connaissances, qui peut apprendre le comportement du système, afin de maîtriser sa conduite.

Cet objectif est atteint en suivant certaine direction bien choisie qui nous amène pas à pas au régulateur voulu. Ce régulateur doit répondre aux exigences et critères suivants:

- taille du régulateur réduite,
- rapidité du calcul de la commande,
- bonne précision,
- possibilité de l'apprentissage .

Ces critères se présentent ou bien qualitativement ou bien quantitativement.

La taille réduite du régulateur se présente comme étant le nombre de règles, dans la base de connaissances, qui doit être réduit.

La rapidité de calcul de la commande est assurée par le choix du type du régulateur.

Dans notre cas, nous avons choisi le régulateur du type de Sugeno.

La bonne précision est obtenue par l'ajustement des paramètres du régulateur, à savoir les gains de normalisation, les fonctions d'appartenance, ...

Enfin, l'apprentissage améliore les performances . L'algorithme utilisé est celui de la rétropropagation temporelle (Temporal Back-Propagation TBP).

IV.2. DECENTRALISATION:

La commande par logique floue, ainsi que la commande par réseaux neuro-flous, se basent sur la notion de base de connaissances (système expert à chaînage avant). Cette dernière est à la fois difficile à formuler, lourde à appliquer dans notre application (processus robotique). Pour cela, nous effectuons une décomposition du système en trois sous-systèmes. Chaque sous-système est commandé par une station locale de commande qui utilise des grandeurs locales pour générer la commande de ce sous-système. A titre d'exemple, le tableau suivant présente la difficulté d'avoir la base de connaissance dans le cas centralisé.

<i>nombre de classes d'appartenance</i>	<i>nombre de règles cas centralisé</i>	<i>nombre de règles cas décentralisé</i>	<i>rapport cent./déc.</i>
3	729	27	27
5	15625	75	208.33
7	117649	147	800.33

Tableau (IV-1). Nombre de règles, cas centralisé, cas décentralisé.

Comme nous pouvons le voir sur le tableau, le cas centralisé présente les difficultés suivantes:

- grand nombre de règles à déterminer,
- si nous ne voulons pas déterminer toutes les règles, une autre difficulté apparaît, qui se présente comme suit : - Quelles sont les règles à déterminer et quel est le critère de choix pour avoir ces règles? Ce qui est très difficile à répondre,
- dans le cas où nous pouvons déterminer ces règles, deux contraintes se présentent:
 - temps de calcul de la commande sera grand,
 - taille de la base de connaissances sera grande, d'où une difficulté en réalisation pratique.

Le cas décentralisé présente les facilités suivantes:

- temps de calcul réduit,
- la commande du système est partagée entre les trois régulateurs, qui n'ont aucune relation entre eux,
- la taille des trois régulateurs est simple,
- possibilité de la réalisation pratique.

Dans notre cas, la décentralisation de la commande n'est pas un choix mais une obligation, car sinon la commande sera irréalisable.

Mais cela n'exclue pas la possibilité d'avoir les règles, dans le cas centralisé, auprès d'un expert, qui connaît bien le système. Le problème, dans ce cas, sera le temps de calcul de la commande. Pour pallier à ce problème, A.E. Gegov et P.M. Frank [28] présentent une méthode de décentralisation par la décomposition active des lois de commandes.

IV.3. DECENTRALISATION, MODELE DYNAMIQUE DU ROBOT:

En examinant le modèle dynamique du robot, nous remarquons que:

- La deuxième articulation est parfaitement découplée des autres. Cette articulation forme le premier sous-système.

- La première et la troisième articulations sont fortement couplées. Malgré cela, nous allons prendre chaque articulation comme étant un sous-système à part.

Notre but sera donc de commander ces trois sous-systèmes en n'utilisant que des mesures locales et de tester les performances des régulateurs ainsi synthétisés.

REMARQUES:

Le modèle n'est utilisé, que pour la simulation, car nous n'avons pas besoin du modèle pour la synthèse du régulateur dans la pratique.

IV.4. STRUCTURE DE LA COMMANDE DECENTRALISEE PAR LOGIQUE

FLOUE:

La figure (IV-1) présente la structure décentralisée de la commande du robot. Chaque sous-système est commandé par un régulateur par logique flou, dont les entrées sont l'erreur et la variation de l'erreur de chaque sous-système.

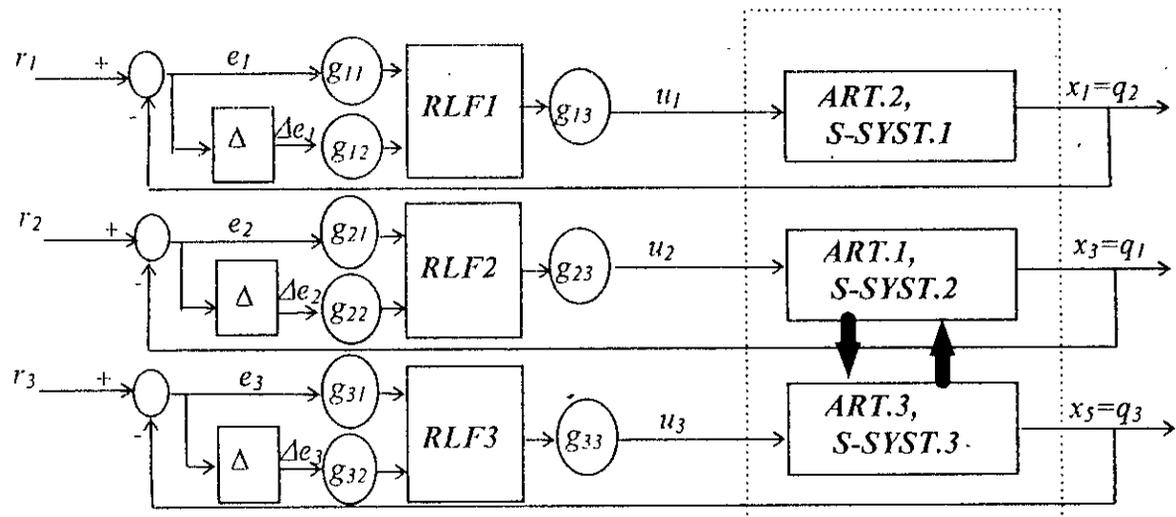


figure (IV-1). structure de la commande décentralisée par logique floue

IV.5. REGULATEUR DU TYPE DE MAMDANI:

IV.5.1. INTRODUCTION:

En 1974, E.H. MAMDANI [9] a présenté, pour la première fois, la technique de réglage par logique floue. Celle-ci consiste à déterminer un ensemble de règles qui, appliquées au système, peuvent le contrôler. L'obtention de ces règles est facile auprès des experts qui connaissent bien le système.

Après, Macvicar et Whelan [35] ont observé que les bases de règles proposées par King et Mamdani étaient incomplètes. Ils ont fait une analyse détaillée sur ces bases de règles et proposé une matrice de règles qui possède 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.

<i>E</i> \ ΔE	<i>NB</i>	<i>NS</i>	<i>ZE</i>	<i>PS</i>	<i>PB</i>
<i>NB</i>	<i>NB</i>	<i>NB</i>	<i>NB</i>	<i>NS</i>	<i>ZE</i>
<i>NS</i>	<i>NB</i>	<i>NB</i>	<i>NS</i>	<i>ZE</i>	<i>PS</i>
<i>ZE</i>	<i>NB</i>	<i>NS</i>	<i>ZE</i>	<i>PS</i>	<i>PB</i>
<i>PS</i>	<i>NS</i>	<i>ZE</i>	<i>PS</i>	<i>PB</i>	<i>PB</i>
<i>PB</i>	<i>ZE</i>	<i>PS</i>	<i>PB</i>	<i>PB</i>	<i>PB</i>

Tableau (IV-2). Matrice de Macvicar-Whelan.

E: erreur, ΔE : variation de l'erreur

NB: negative big, NS: negative small, ZE: zero, PS: positive small, PB: positive big

IV.5.2. STRUCTURE DES REGULTEURS:

Les régulateurs de type de Mamdani, (ref. Ch.II), utilisés se forment des blocs:

- Fuzzification: Celle-ci est réalisée par des fonctions d'appartenance triangulaires.
- Base de règles: Elle contient les règles floues.
- Moteur d'inférence: Celui-ci combine les entrées après fuzzification et les règles floues pour tirer des conclusions. Cette opération se compose de:
 - Conjonction des prémisses: Elle est réalisée par l'opération *Min*.
 - Implication floue: Elle est réalisée, aussi, par l'opération *Min*.
 - Agrégation des règles: Elle est assurée par l'opération *Max*.
- Defuzzification: Dans notre application, nous avons choisi celle du centre de gravité.

IV.5.3. SYNTHÈSE DES REGULTEURS:

En utilisant la matrice de Macvicar-Whelan, nous déterminons la base de règles pour chaque régulateur. Nous commençons par le cas le plus simple, qui est celui d'un régulateur du type de Mamdani (3x3).

Les trois sous-systèmes possèdent le même type de régulateur, et les mêmes fonctions d'appartenance. La différence réside dans les gains de normalisation.

IV.5.3.1. REGULATEUR A TROIS CLASSES:

- La base de règles:

<i>E</i> \ ΔE	<i>N</i>	<i>Z</i>	<i>P</i>
<i>N</i>	<i>N</i>	<i>N</i>	<i>Z</i>
<i>Z</i>	<i>N</i>	<i>Z</i>	<i>P</i>
<i>P</i>	<i>Z</i>	<i>P</i>	<i>P</i>

Tableau (IV-3). Base de règles, RLF (3x3)

(*N*: négative, *Z*: zéro, *P*: positive).

- Fonctions d'appartenance:

Les fonctions d'appartenance sont de la forme triangulaire. Elles sont présentées par la figure (IV-2).

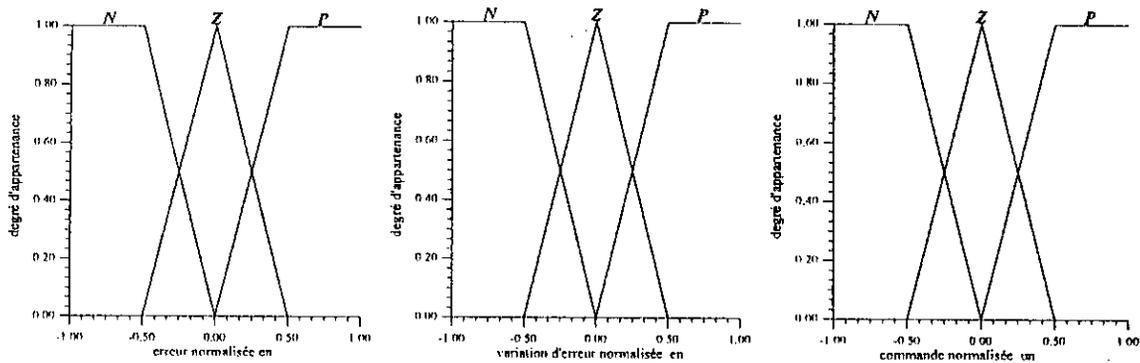


Figure (IV-2). Fonctions d'appartenance aux classes *N*, *Z* et *P*.

- Gains de normalisation:

Les gains de normalisations pour l'erreur et la variation de l'erreur, en cas de poursuite, sont pris faibles. Une règle pratique consiste à prendre le gain correspondant à l'erreur, proche de la valeur de l'erreur désirée, et le gain correspondant à la variation de l'erreur égal au dixième du gain précédant.

Dans notre cas, nous avons pris :

- gain-erreur : $g_1=0.005$,
- gain-variation de l'erreur : $g_2=0.0005$,

Le gain correspondant à la commande dépend du sous-système, et nous avons:

- sous-système 1: $g_3=4$
- sous-système 2: $g_3=7$
- sous-système 3: $g_3=8$

IV.5.3.2. REGULATEUR A CINQ CLASSES:

- La base de règle:

<i>E</i> \ ΔE	<i>NB</i>	<i>NS</i>	<i>ZE</i>	<i>PS</i>	<i>PB</i>
<i>NB</i>	<i>NB</i>	<i>NB</i>	<i>NB</i>	<i>NS</i>	<i>ZE</i>
<i>NS</i>	<i>NB</i>	<i>NB</i>	<i>NS</i>	<i>ZE</i>	<i>PS</i>
<i>ZE</i>	<i>NB</i>	<i>NS</i>	<i>ZE</i>	<i>PS</i>	<i>PB</i>
<i>PS</i>	<i>NS</i>	<i>ZE</i>	<i>PS</i>	<i>PB</i>	<i>PB</i>
<i>PB</i>	<i>ZE</i>	<i>PS</i>	<i>PB</i>	<i>PB</i>	<i>PB</i>

Tableau (IV-4). Table de règles. RLF (5x5)

- Fonctions d'appartenance:

Elles sont de la forme triangulaire, et sont présentées par la figure (IV-3).

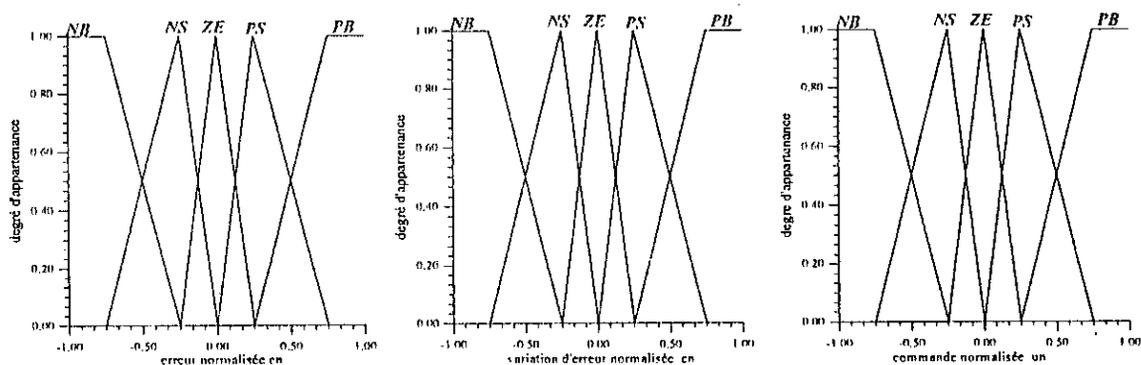


Figure (IV-3). Fonctions d'appartenance aux classes NB, NS, ZE, PS et PB.

- Gains de normalisation:
 Comme précédemment, nous avons:
 - gain-erreur : $g_1=0.005$
 - gain-variation de l'erreur : $g_2=0.0005$
- Le gain correspondant à la commande :
 - sous-système 1: $g_3=3$
 - sous-système 2: $g_3=4$
 - sous-système 3: $g_3=20$

IV.5.3.3. REGULATEUR À SEPT CLASSES:

- La 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 (IV-5). table de règles, RLF (7x7),

(NM: negative medium, PM: positive medium).

- pour les gains de normalisation nous avons constaté que:

A/ si nous fixons d'abord les gains associés aux entrées d'un régulateur, et nous variaisons le gain associé à la commande :

1/ gain associé à la commande faible conduit à une commande faible donc à une erreur grande.

2/ gain associé à la commande grand conduit à une commande qui présente de fortes oscillations.

B/ si, pour avoir une erreur faible et une commande non oscillante, nous diminuons les gains, le problème sera la commande qui ne possède pas l'énergie nécessaire pour assurer la conduite du robot.

IV.5.4.2. COMPARAISON ET JUSTIFICATION:

Nous remarquons, d'après la figure (IV-11.a,b,c), que le RLF(3x3) est le meilleur, et que l'augmentation du nombre de règles ne signifie pas l'obtention de meilleures performances, car il y a plusieurs facteurs qui entrent en jeu, comme les fonctions d'appartenance, les gains de normalisation, les opérateurs de conjonction, ...

En plus, l'augmentation du nombre de règles rend difficile l'obtention de ces règles, et rend aussi plus lent le calcul de la commande.

Voici un tableau qui donne les valeurs approchées de k_p et k_d pour chaque sous-système:

		<i>s-syst1</i>	<i>s-syst2</i>	<i>s-syst3</i>
RLF(3x3)	k_p	1773	1862	2128
	k_d	10640	18620	21280
RLF(5x5)	k_p	792	1056	5280
	k_d	7920	10560	52800
RLF(7x7)	k_p	622.8	830.4	4152
	k_d	6228	8304	41520

Tableau (IV-6). Valeurs approchées de k_p et de k_d .

IV.5.5. PERFORMANCES ET TESTS DE ROBUSTESSE:

Pour voir la robustesse de notre commande, nous avons fait les tests suivants:

IV.5.5.1. VARIATIONS PARAMETRIQUES:

Dans ce test, nous avons varié les paramètres suivants: les masses, les coefficients de frottement, les longueurs, de 50% de leurs valeurs initiales. Les résultats de ce test sont présentés à la figure (IV-12.a,b,c).

Nous avons constaté que les erreurs en position restent faibles, malgré l'apparition d'une erreur statique au niveau de la deuxième articulation, que le régulateur n'a pas pu éliminer. La commande est toujours en relation quasi-linéaire avec l'erreur.

Avec ce test, nous pouvons conclure que la commande est robuste par rapport aux paramètres.

IV.5.5.2. DÉFAILLANCE DU CAPTEUR:

Dans ce test, nous avons simulé la défaillance du capteur n°3. Les résultats de ce test sont présentés sur la figure (IV-13.a,b,c).

Nous avons constaté que la réponse de la deuxième articulation n'est pas perturbée par la défaillance de ce capteur, car cette articulation est totalement découplée des deux autres. Alors que la première présente une réponse un peu perturbée à cause du couplage qui existe entre les articulations 1 et 3, malgré cela, les erreurs restent faibles.

En ce qui concerne l'articulation n°3, son contrôle n'est pas assuré. Nous avons remarqué que la commande reste bornée dans les limites du fonctionnement ce qui évite le danger d'endommager le robot.

IV.5.5.3. RUPTURE DE LA COMMANDE:

Dans ce test, nous avons simulé une défaillance au niveau régulateur-actionneur, qui se présente comme s'il n'y a pas de passage de la commande du régulateur vers l'actionneur. Nous avons fait la rupture sur l'articulation n°1. Les résultats sont présentés sur la figure (IV-14.a,b,c).

Nous avons constaté que la deuxième articulation n'est pas affectée par cette rupture à cause du découplage. La première articulation ne suit pas la trajectoire, alors que, malgré le couplage, la troisième articulation ne présente pas une perturbation apparente.

IV.5.6. CONCLUSION:

Dans cette partie du travail, nous avons pu voir la simplicité des régulateurs par logique floue du type de Mamdani, en synthèse et en formulation. Nous avons pu synthétiser des régulateurs qui donnent de bons résultats, en erreur et en robustesse. L'inconvénient reste le temps de calcul de la commande qui est dû au nombre de règles.

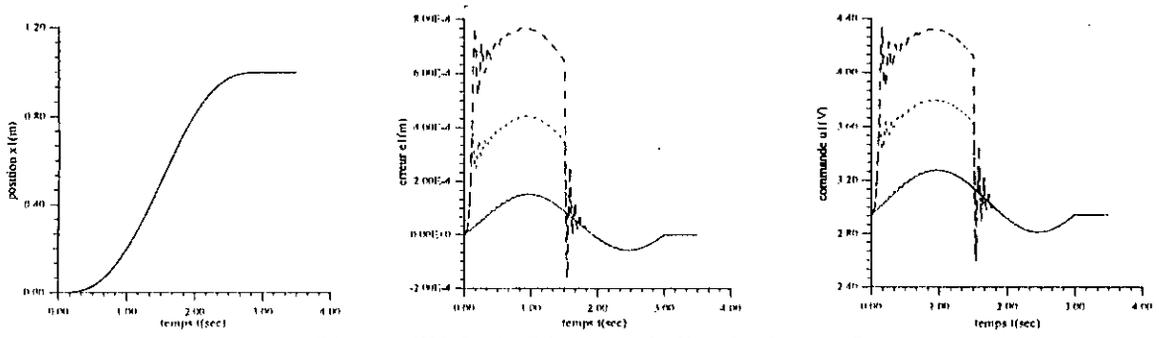


Figure (IV-5.a). Réponse de l'articulation 2
position, erreur et commande
RLF type Mamdani 3x3

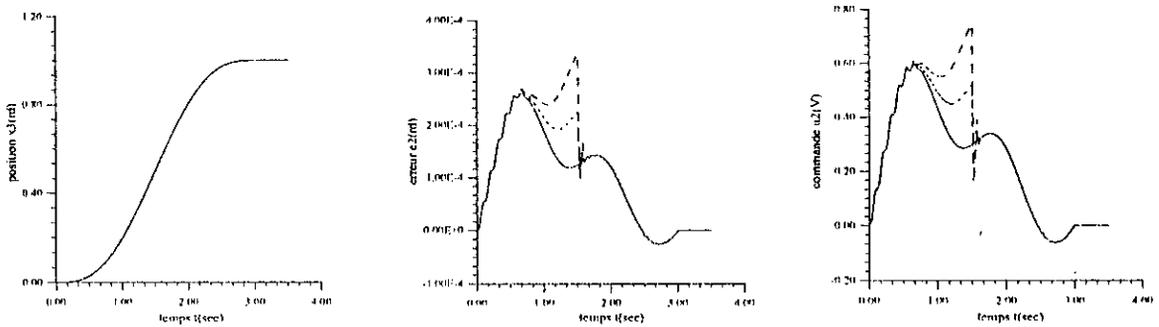


Figure (IV-5.b). Réponse de l'articulation 1
position, erreur et commande
RLF type Mamdani 3x3

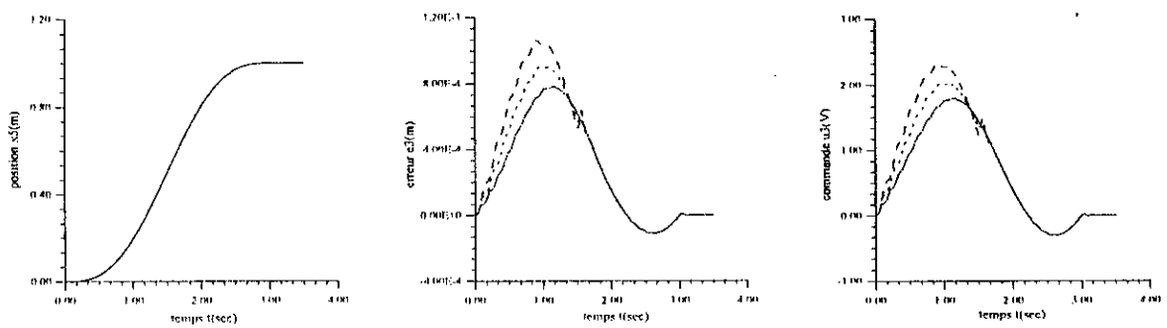


Figure (IV-5.c). Réponse de l'articulation 3
position, erreur et commande
RLF type Mamdani 3x3

— Référence
 - - - - - $m_0 = 50\%$ de m_3
 - - - - - $m_0 = 0$ kg
 - - - - - $m_0 = 100\%$ de m_3

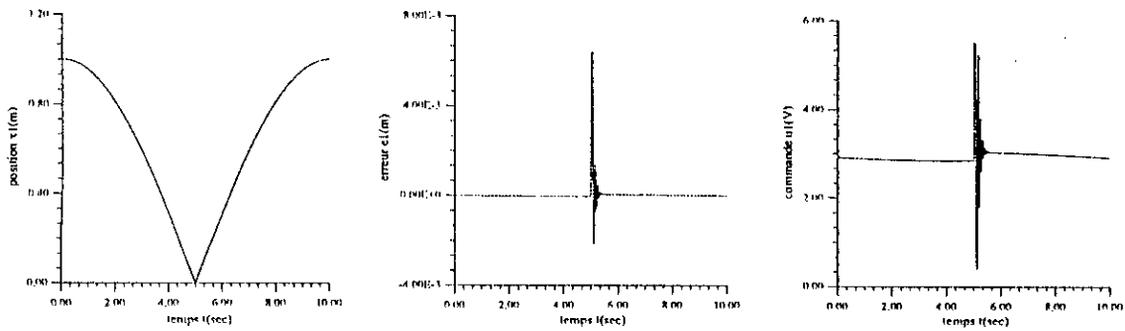


Figure (IV-6.a). Fenêtre de Viviani, Art. 2
position, erreur, commande
RLF type Mamdani 3x3

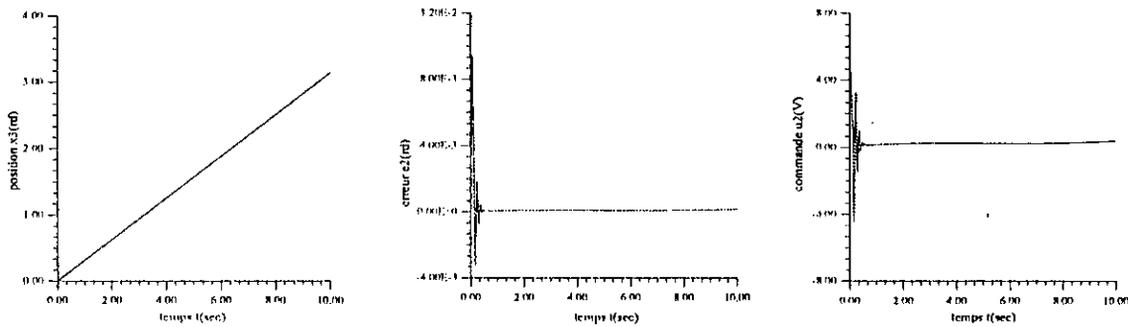


Figure (IV-7.b). Fenêtre de Viviani, Art. 1
position, erreur, commande
RLF type Mamdani 3x3

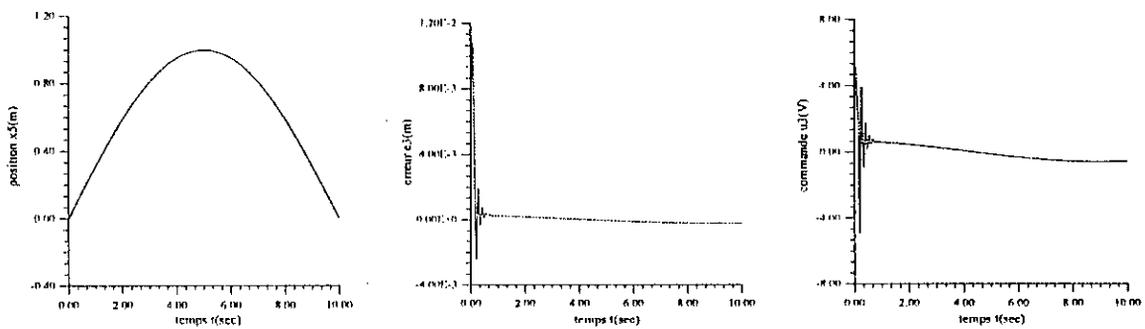


Figure (IV-6.c). Fenêtre de Viviani, Art. 3
position, erreur, commande
RLF type Mamdani 3x3

— Référence

— Réponse

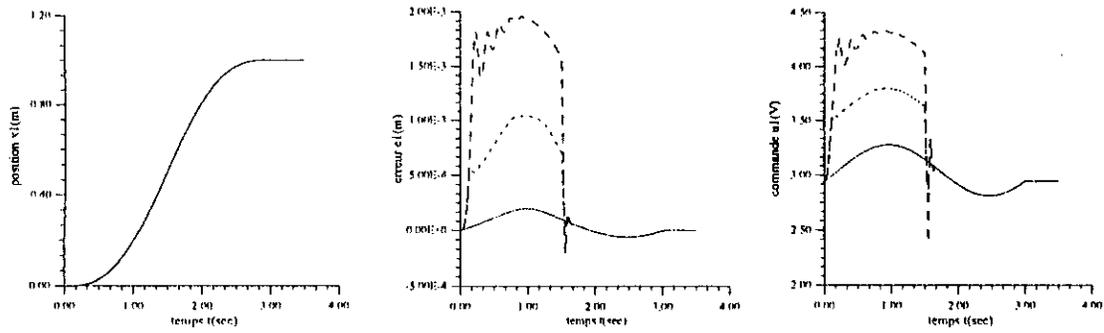


Figure (IV-7.a). Réponse de l'articulation 2
position, erreur et commande
RLF type Mamdani 5x5

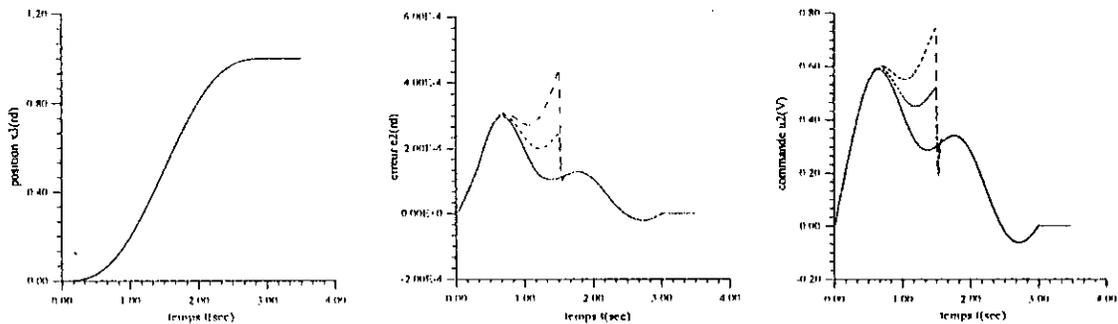


Figure (IV-7.b). Réponse de l'articulation 1
position, erreur et commande
RLF type Mamdani 5x5

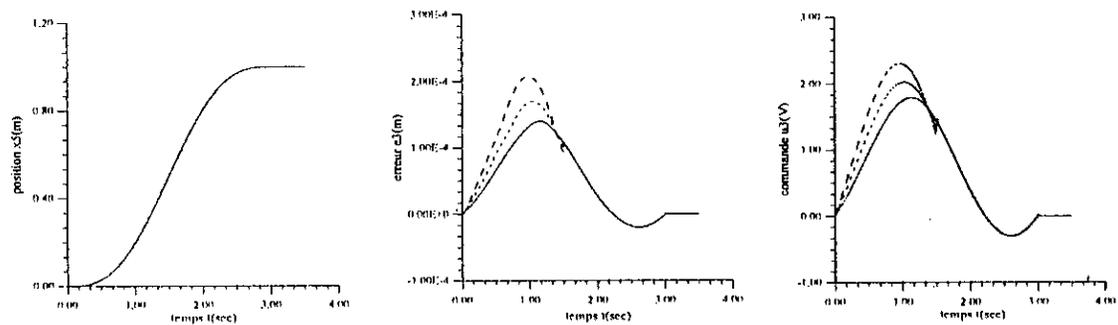


Figure (IV-7.c). Réponse de l'articulation 3
position, erreur et commande
RLF type Mamdani 5x5

——— Référence
 $m_0=0$ kg
 - - - - $m_0=50\%$ de m_3
 - - - - $m_0=100\%$ de m_3

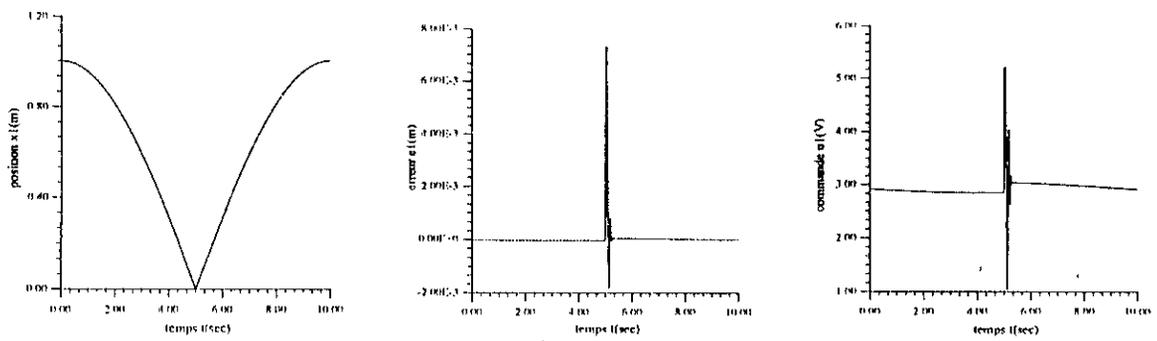


Figure (IV-8.a). Fenêtre de Viviani, Art.2
position, erreur, commande
RLF type Mamdani 5x5

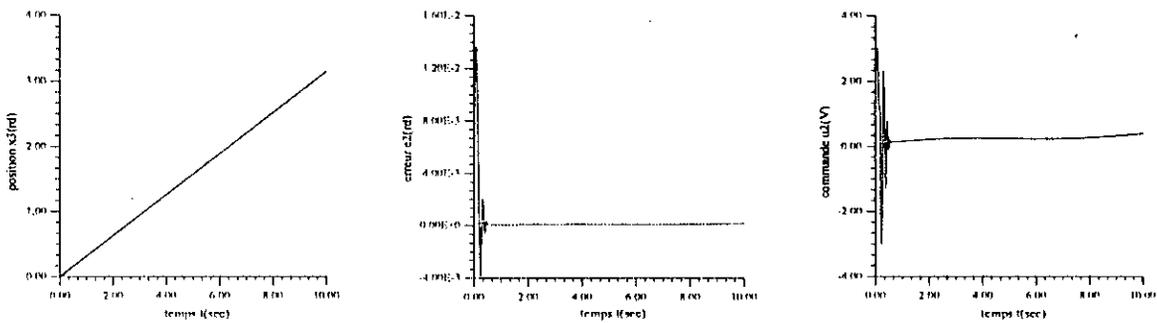


Figure (IV-8.b). Fenêtre de Viviani, Art.1
position, erreur, commande
RLF type Mamdani 5x5

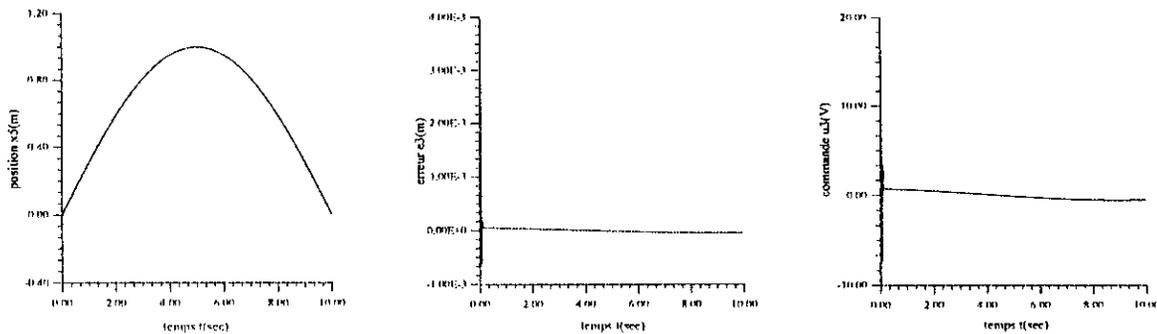


Figure (IV-8.c). Fenêtre de Viviani, Art.3
position, erreur, commande
RLF type Mamdani 5x5

— Référence

----- Réponse

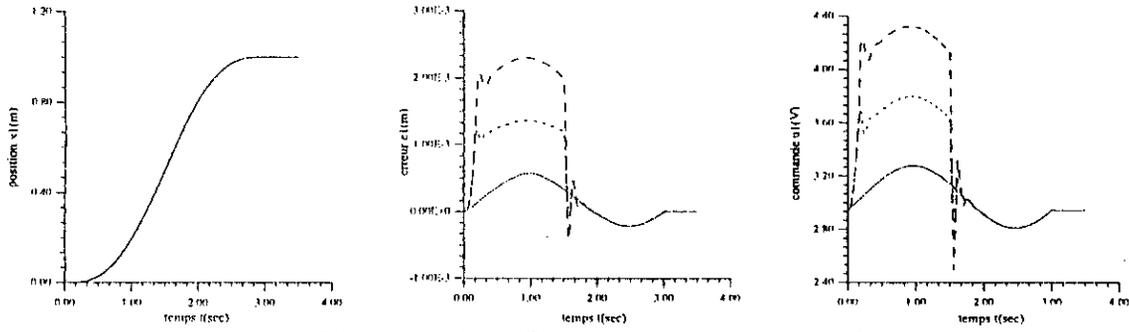


Figure (IV-9.a). Réponse de l'articulation 2
position, erreur et commande
RLF type Mamdani 7x7

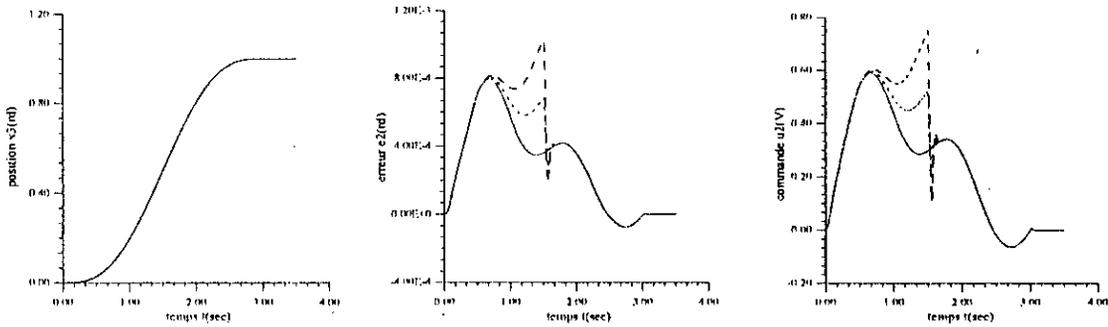


Figure (IV-9.b). Réponse de l'articulation 1
position, erreur et commande
RLF type Mamdani 7x7

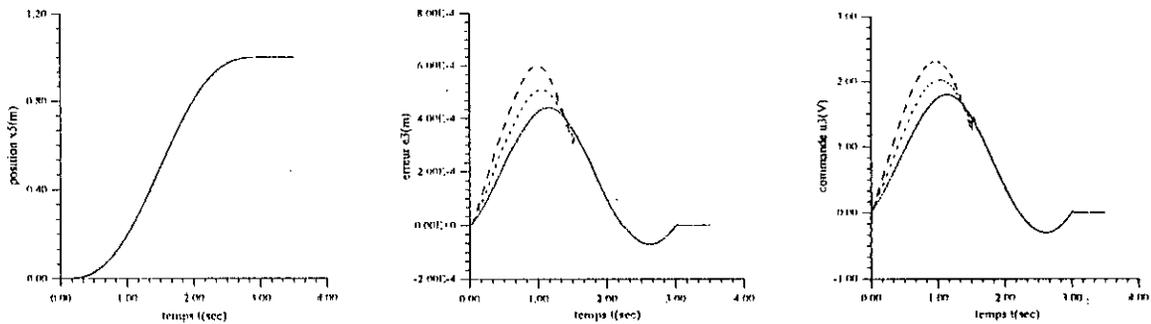


Figure (IV-9.c). Réponse de l'articulation 3
position, erreur et commande
RLF type Mamdani 7x7



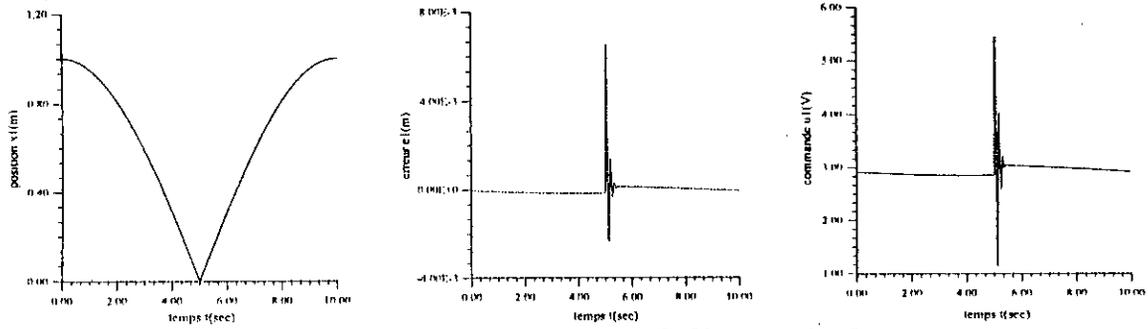


Figure (IV-10.a). Fenêtre de Viviani, Art.2
position, erreur, commande
RLF type Mamdani 7x7

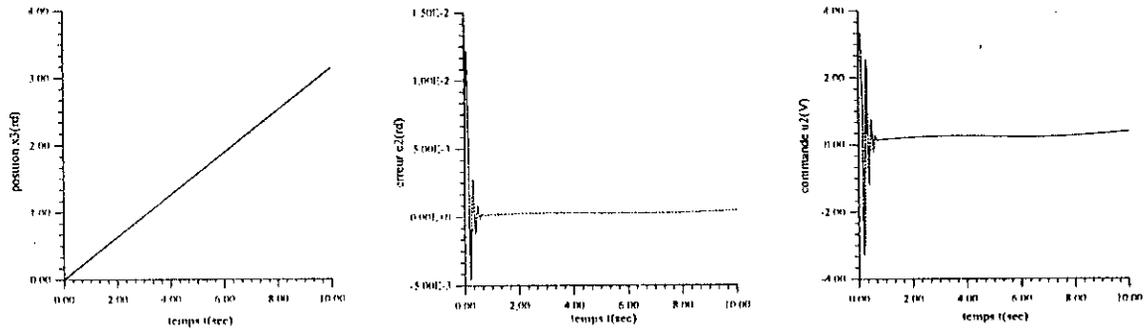


Figure (IV-10.b). Fenêtre de Viviani, Art.1
position, erreur, commande
RLF type Mamdani 7x7

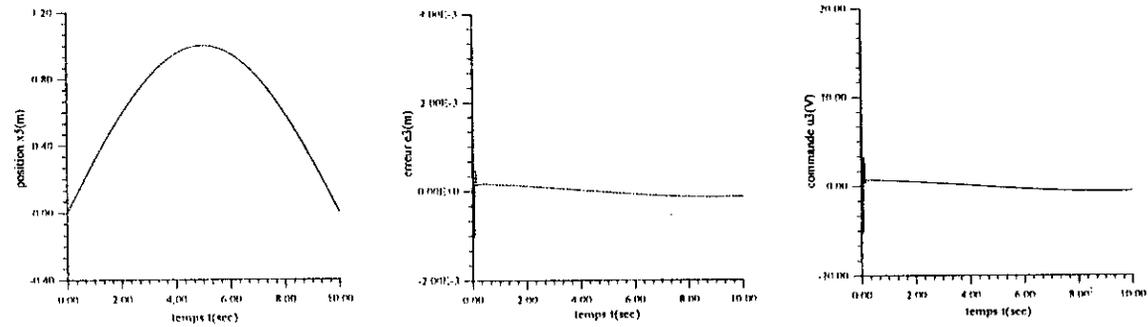


Figure (IV-10.c). Fenêtre de Viviani, Art.3
position, erreur, commande
RLF type Mamdani 7x7

— Référence

— Réponse

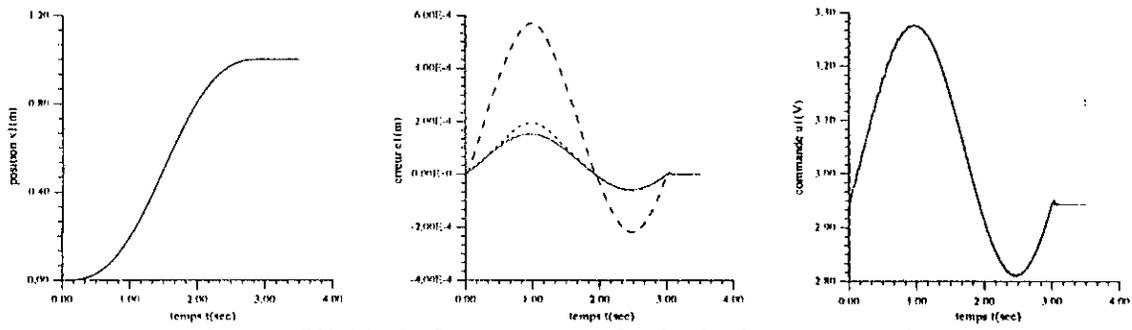


Figure (IV-11.a). Comparaison des 3 régulateurs pour l'art. 2

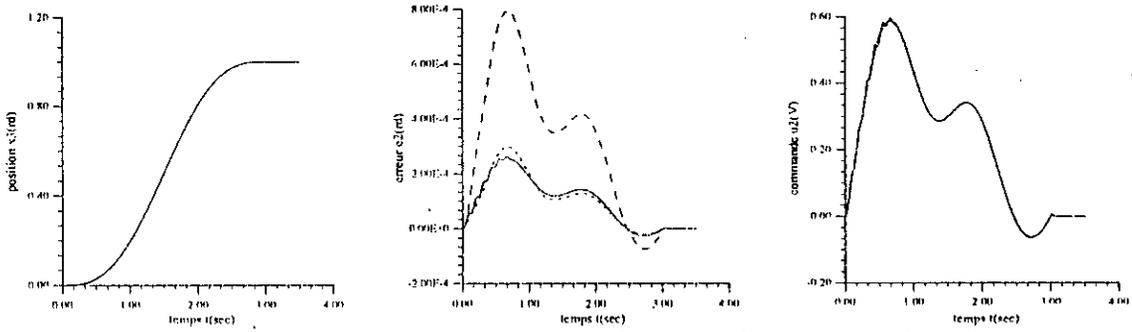


Figure (IV-11.b). Comparaison des 3 régulateurs pour l'art. 1

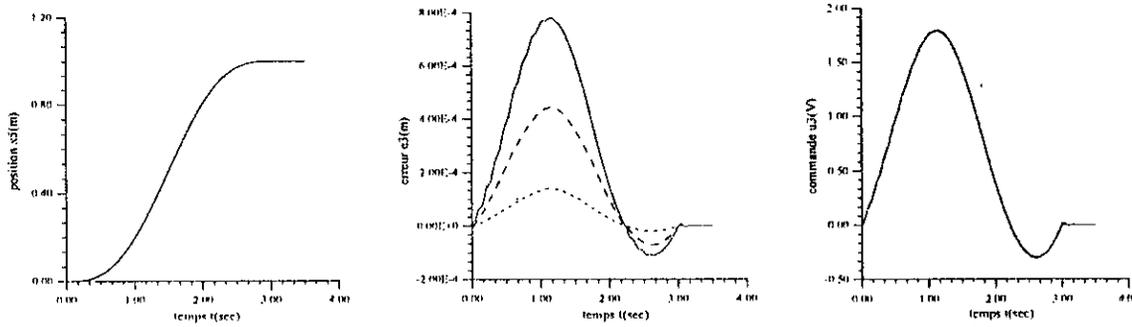


Figure (IV-11.c). Comparaison des 3 régulateurs pour l'art. 3

——— Référence
 RLF type Mamdani 5x5
 ——— RLF type Mamdani 3x3
 - - - - RLF type Mamdani 7x7

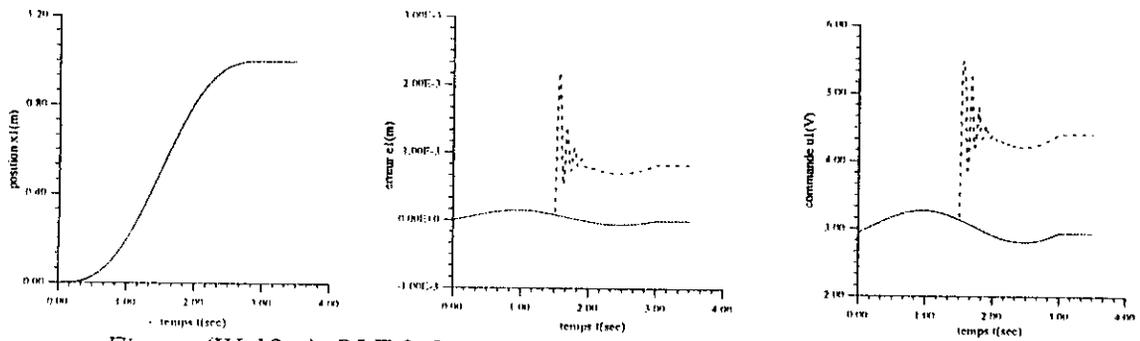


Figure (IV-12.a). RLF 3x3 type Mamdani, Variations des paramètres, Art.2

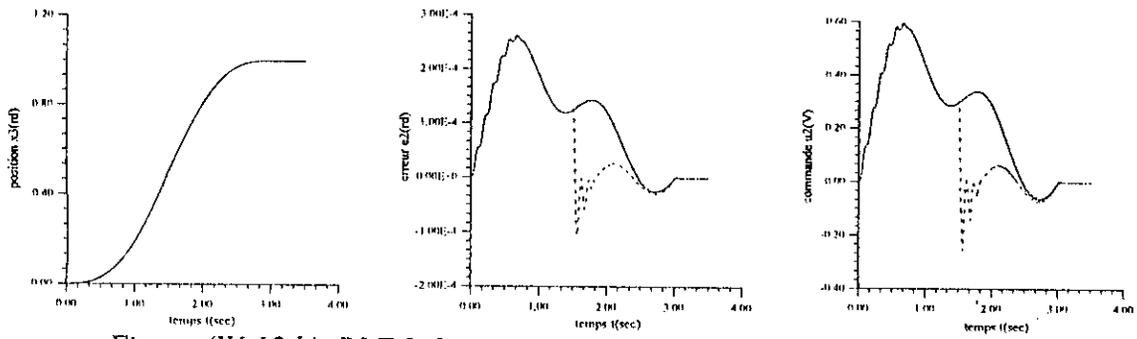


Figure (IV-12.b). RLF 3x3 type Mamdani, Variations des paramètres, Art.1

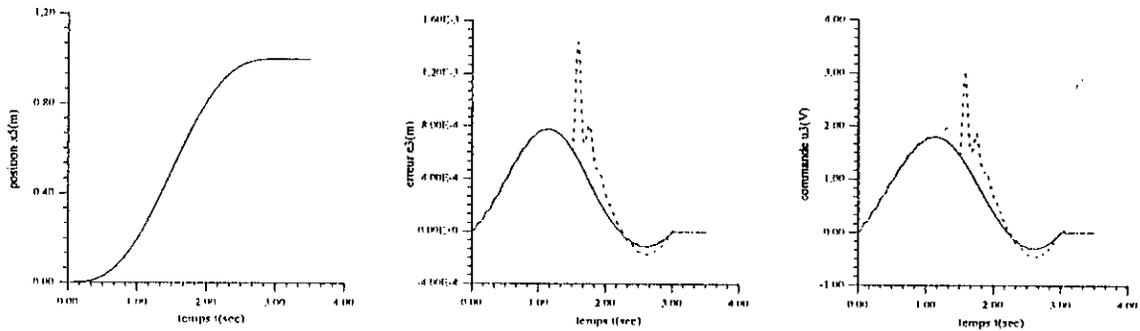


Figure (IV-12.c). RLF 3x3 type Mamdani, Variations des paramètres, Art.3

— Référence

— Sans variations de paramètres

--- Avec variations de paramètres

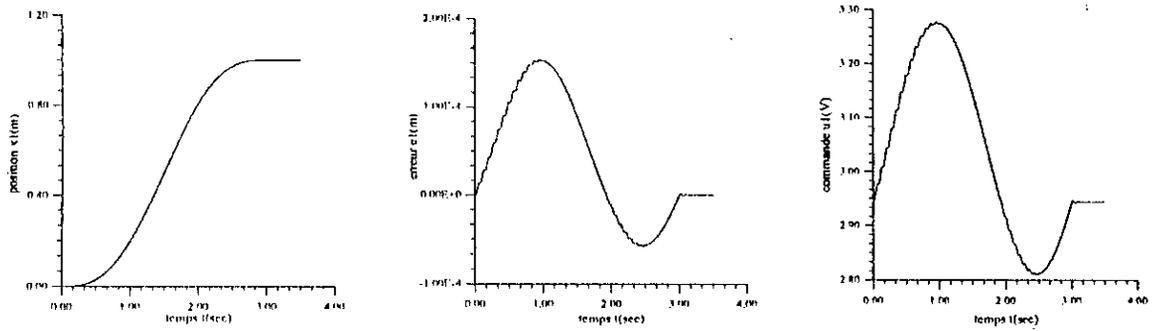


Figure (IV-13.a). RLF 3x3 type Mamdani, capteur n°3 défaillant, art.2

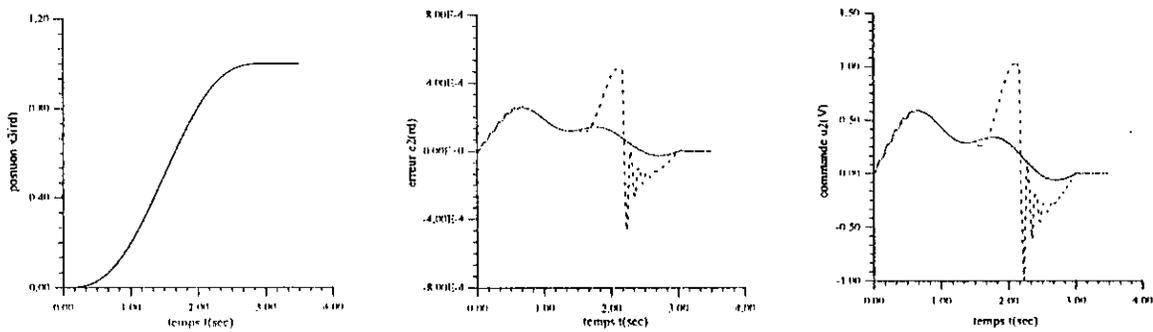


Figure (IV-13.b). RLF 3x3 type Mamdani, capteur n°3 défaillant, art.1

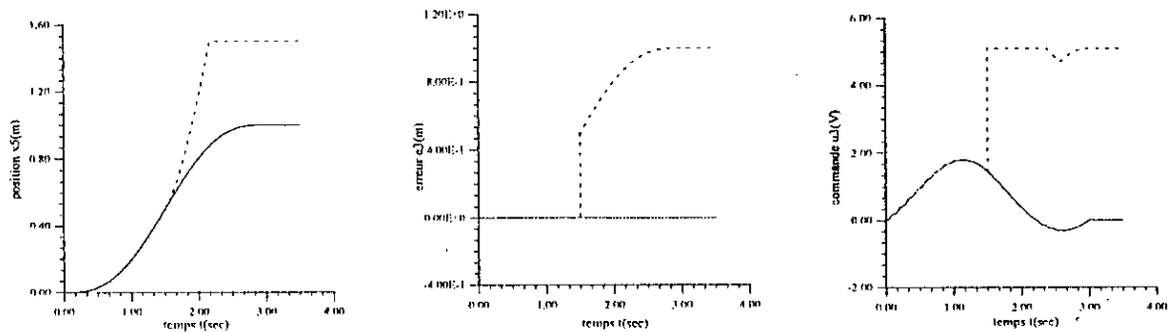


Figure (IV-13.c). RLF 3x3 type Mamdani, capteur n°3 défaillant, art.3

————— Référence
 ————— Capteur n°3 non défaillant
 - - - - - Capteur n°3 défaillant

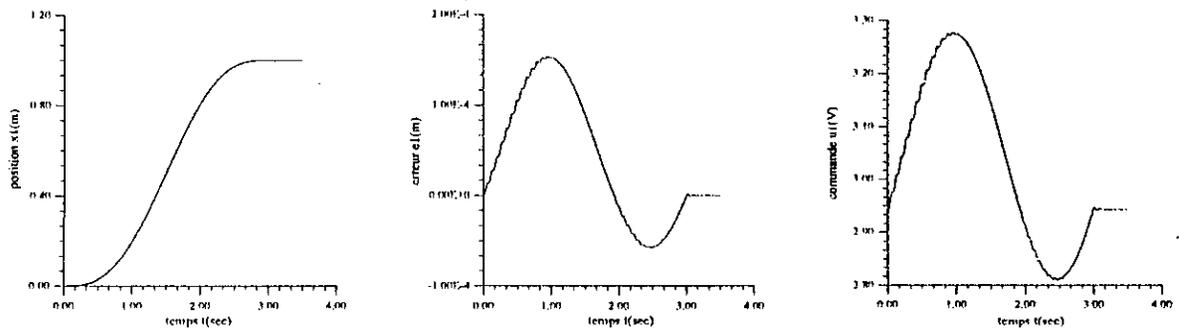


Figure (IV-14.a). RLF 3x3 type Mamdani, rupture de la commande de l'art.1, réponse de l'art.2

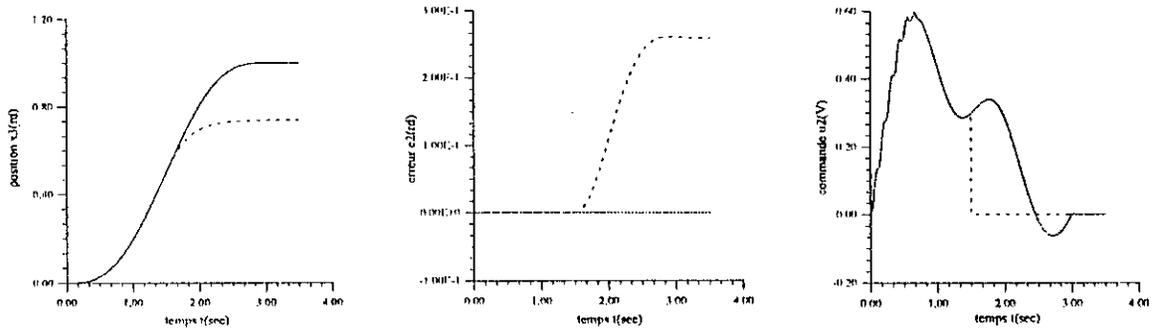


Figure (IV-14.b). RLF 3x3 type Mamdani, rupture de la commande de l'art.1, réponse de l'art.1

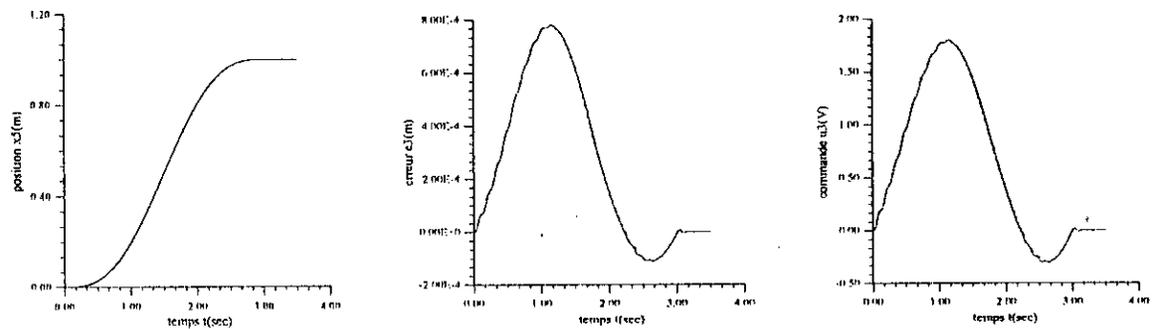


Figure (IV-14.c). RLF 3x3 type Mamdani, rupture de la commande de l'art.1, réponse de l'art.3

Référence
 Sans rupture de la commande
 Avec rupture de la commande

IV.6. REGULATEURS DE TYPE DE SUGENO:**IV.6.1. INTRODUCTION:**

Le régulateur de type de Sugeno a été proposé par M. Sugeno, Il est difficile à déterminer par des connaissances vagues et imprécises.

IV.6.2. DETERMINATION DES REGULATEURS:

En utilisant le régulateur de type de Mamdani, nous pouvons déterminer celui de type de Sugeno. Le passage peut se faire par deux méthodes. Ce passage est justifié par le temps de calcul qui se réduit énormément pour le régulateur de type de Sugeno par rapport à celui de type de Mamdani car l'étape de défuzzification est émergeée dans l'inférence.

IV.6.2.1. SYNTHÈSE DU REGULATEUR DE TYPE DE SUGENO PAR IDENTIFICATION:

On impose au régulateur de type de Mamdani un ensemble d'entrées, pour construire un fichier E/S qu'on va utiliser pour identifier celui de type de Sugeno.

On a:

$$u_j(e_j, \Delta e_j) = \sum_{i=1}^n \overline{w}_i f_{ij}$$

$$f_{ij} = p_i e_j + q_i \Delta e_j + r_i$$

$$j = \overline{1, N} \quad \text{(IV-1)}$$

n : nombre de règles

N : nombre de mesures

\overline{w}_i : coefficient de pondération

Par exemple, pour $n=4$, on a:

$$u_j = \overline{w}_1 f_{1j} + \overline{w}_2 f_{2j} + \overline{w}_3 f_{3j} + \overline{w}_4 f_{4j} \quad \text{(IV-2)}$$

avec $j=1, \dots, N$.

On peut écrire

$$\begin{aligned}
 \underline{U} &= A\theta \\
 \underline{U} &= [u_1, u_2, \dots, u_N]^T \\
 \underline{\theta} &= [p_1, q_1, r_1, p_2, q_2, r_2, p_3, q_3, r_3, p_4, q_4, r_4]^T \\
 A &= \begin{bmatrix} \overline{w_1 e_1} & \overline{w_1 \Delta e_1} & \overline{w_1} & \dots & \overline{w_4 e_1} & \overline{w_4 \Delta e_1} & \overline{w_4} \\ \overline{w_1 e_2} & \overline{w_1 \Delta e_2} & \overline{w_1} & \dots & \overline{w_4 e_2} & \overline{w_4 \Delta e_2} & \overline{w_4} \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \overline{w_1 e_N} & \overline{w_1 \Delta e_N} & \overline{w_1} & \dots & \overline{w_4 e_N} & \overline{w_4 \Delta e_N} & \overline{w_4} \end{bmatrix} \quad (IV-3)
 \end{aligned}$$

Par la méthode des moindres carrés on tire:

$$\hat{\underline{\theta}} = [A^T A]^{-1} A^T \underline{U} \quad (IV-4)$$

IV.6.2.2. SYNTHÈSE DU RÉGULATEUR DE TYPE DE SUGENO PAR APPROXIMATION:

Cette méthode consiste à approcher la surface $u_n(e_n, \Delta e_n)$ par des plans, (similaire à l'approximation d'une courbe par des segments de droites). Pour cela, On divise le domaine $(e_n, \Delta e_n)$ en $n \times m$ sous-domaines, avec n et m les nombres de classes associées respectivement à e_n et à Δe_n . On dispose, ainsi, de $n \times m$ règles. Chaque sous-domaine représente une règle qui possède trois paramètres à déterminer. Ces derniers déterminent un plan dans l'espace $(e_n, \Delta e_n, u_n)$. C'est ce plan qui donne une approximation de la surface $u_n = u_n(e_n, \Delta e_n)$ dans ce sous-domaine.

Il suffit d'avoir trois points pour déterminer un plan. Soient A, B et C les trois points dont les coordonnées sont $(e_{n1}, \Delta e_{n1}, u_{n1}), (e_{n2}, \Delta e_{n2}, u_{n2})$ et $(e_{n3}, \Delta e_{n3}, u_{n3})$, respectivement. Le plan est donné par une équation de la forme

$$u = pe_n + q\Delta e_n + r$$

et on a:

$$u_i = pe_{ni} + q\Delta e_{ni} + r$$

avec $i = 1, 2, 3$

sous forme matricielle

$$\underline{U} = A\underline{\theta}$$

$$\text{avec } \underline{U} = [u_1, u_2, u_3]^T \text{ et } \underline{\theta} = [p, q, r]^T \quad (\text{IV-5})$$

$$\text{et } A = \begin{bmatrix} e_{n1} & \Delta e_{n1} & 1 \\ e_{n2} & \Delta e_{n2} & 1 \\ e_{n3} & \Delta e_{n3} & 1 \end{bmatrix}$$

et on tire enfin:

$$\underline{\theta} = A^{-1}\underline{U}$$

Ainsi on détermine, pour chaque règle, ses paramètres.

C'est cette méthode que nous avons utilisée, pour la détermination du régulateur de type de Sugeno.

REMARQUE: La méthode de détermination par l'identification est très lourde à appliquer, mais elle est plus précise, alors que l'autre méthode est très simple et moins précise.

IV.6.2.3. RÉGULATEUR DE TYPE DE SUGENO A TROIS CLASSES:

A partir du régulateur de type de Mamdani ayant sept classes, nous avons déterminé le régulateur de type de Sugeno ayant seulement trois classes (9 règles), en utilisant la méthode de détermination par l'approximation. Nous avons trouvé

- Base de règles:

$$u_n = p e_n + q \Delta e_n + r$$

<i>n° règle</i>	<i>e_n</i>	<i>Δe_n</i>	<i>p</i>	<i>q</i>	<i>r</i>
1	N	N	0.1515	0.1515	-0.4035
2	N	Z	0.4818	0.4818	-0.0671
3	N	P	0.3552	0.3552	0
4	Z	N	0.4818	0.4818	-0.0671
5	Z	Z	0.6591	0.6591	0
6	Z	P	0.4818	0.4818	0.0671
7	P	N	0.3552	0.3552	0
8	P	Z	0.4818	0.4818	0.0671
9	P	P	0.1515	0.1515	0.4035

Tableau (IV-7). Base de règles, RLF(3x3), Sugeno.

- Fonctions d'appartenance:

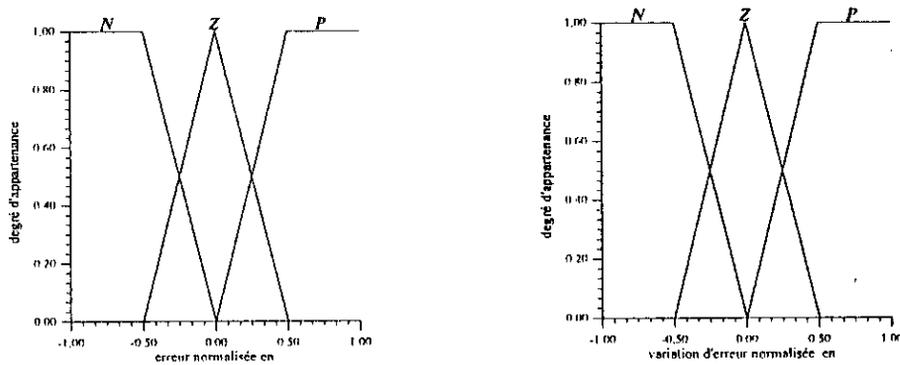


Figure (IV-15). fonctions d'appartenance, (l'erreur et sa variation).

- Gains de normalisation:

Sous-système 1:

$$g_1=0.005, g_2=0.0005, g_3=4$$

Sous-système 2:

$$g_1=0.005, g_2=0.0005, g_3=7$$

Sous-système 3:

$$g_1=0.005, g_2=0.0005, g_3=20$$

IV.6.2.4. REGULATEUR DE TYPE DE SUGENO A DEUX CLASSES:

Pour minimiser, encore, le temps de calcul de la commande, nous avons supprimé cinq règles de la base de connaissances du régulateur précédant. Nous avons, donc, laissé quatre règles seulement. Les règles supprimées sont ceux dont les prémisses contiennent la classe Z. Nous avons, aussi, changé les fonctions d'appartenance aux classes qui restent : *N* et *P*.

- Base de règles:

$$u_n = p e_n + q \Delta e_n + r$$

<i>n</i> ^o règle	<i>e_n</i>	Δe_n	<i>p</i>	<i>q</i>	<i>r</i>
1	<i>N</i>	<i>N</i>	0.1515	0.1515	-0.4035
2	<i>N</i>	<i>P</i>	0.3552	0.3552	0
3	<i>P</i>	<i>N</i>	0.3552	0.3552	0
4	<i>P</i>	<i>P</i>	0.1515	0.1515	0.4035

Tableau (IV-8). Base de règles, RLF(2x2), Sugeno.

- Fonctions d'appartenance:

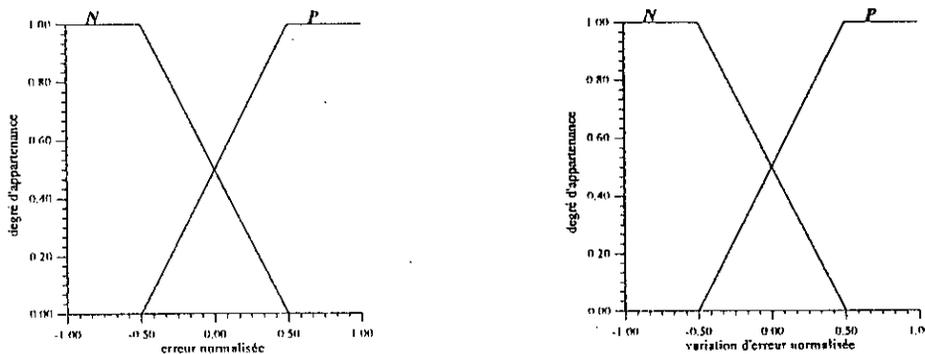


Figure (IV-16). fonctions d'appartenance, (l'erreur et sa variation).

- Gains de normalisation:

Sous-système 1:

$$g_1=0.005, g_2=0.0005, g_3=4$$

Sous-système 2:

$$g_1=0.005, g_2=0.0005, g_3=7$$

Sous-système 3:

$$g_1=0.005, g_2=0.0005, g_3=20$$

IV.6.3. ETUDE COMPARATIVE:

D'après les simulations, nous allons maintenant tirer quelques conclusions et remarques sur la réponse et sur les performances des régulateurs. Les résultats sont présentés par les figures (IV-17.a,b,c),..., (IV-24.a,b,c).

IV.6.3.1. ASPECT GENERAL DE LA REPONSE:

Les résultats sont présentés par les figures (IV-17.a,b,c),..., (IV-20.a,c,b). Comme dans le cas des régulateurs de type de Mamdani, nous remarquons la relation quasi-linéaire entre l'erreur, la variation de l'erreur et la commande. Les efforts appliqués par les actionneurs augmentent avec la charge. Nous remarquons aussi la bonne poursuite dans le cas de la fenêtre de Viviani (erreur faible, son max. est 0.8% du max. de la consigne).

IV.6.3.2. COMPARAISON ET JUSTIFICATION:

Les résultats sont présentés par les figures (IV-21.a,b,c).

En ce qui concerne les deux régulateurs de type de Sugeno, RLF(3x3) et RLF(2x2), nous avons constaté que:

- **Remarques:**

- les deux RLF donnent de bons résultats, mais la précision avec le RLF(3x3) est meilleure qu'avec le RLF(2x2).

- les commandes générées par les deux RLF sont très proches, voire identiques.

- quand la charge est lâchée, des oscillations au niveau de la commande apparaissent. Ces oscillations sont atténuées plus rapidement avec le RLF(3x3) qu'avec le RLF(2x2).

- Le RLF(3x3) est meilleur que le RLF(2x2) en précision, mais le dernier est le plus rapide.

- *Justification:*

- les deux RLF de type de Sugeno sont des versions approchées de ceux de type de Mamdani précédents.

- en éliminant quelques règles (ici, 5 règles parmi 9 sont éliminées), le RLF perd de l'information sur le système, ce qui se traduit par une dégradation de la maîtrise du comportement du système.

IV.6.4. TESTS DE ROBUSTESSE:

Les tests de robustesse sont appliqués sur le RLF(3x3).

IV.6.4.1. VARIATION PARAMETRIQUE:

Les résultats de ce test sont présentés par les figures (IV-22.a,b,c). L'application d'une variation paramétrique, comme dans le cas précédent, perturbe le système. Mais l'effet de cette perturbation est très faible en général. Nous avons constaté l'existence d'une erreur statique, dans la réponse de l'articulation n°2, que le RLF(3x3) n'a pas pu éliminer, mais elle reste faible. Les deux autres articulations changent leurs comportements, mais elles ne présentent pas d'erreurs statiques.

IV.6.4.2. CAPTEUR DEFAILLANT:

Nous avons supposé que, le capteur n°3 tombe en panne, pendant le fonctionnement du robot. Les résultats sont présentés par les figures (IV-23;a,b,c). La deuxième articulation n'est pas perturbée à cause du découplage. Nous avons constaté la perte de contrôle de l'articulation n°3. L'articulation n°1 est affectée par ce test, mais son régulateur réagit rapidement pour assurer son bon fonctionnement.

Nous avons remarqué que la commande de l'articulation n°3 augmente rapidement pour atteindre des valeurs qui peuvent endommager l'actionneur n°3. Ceci est dû au fait que nos RLF(3x3) de type de Sugeno possèdent des fonctions d'appartenance aux classes N et P qui ne s'annulent pas en dehors de l'intervalle $[-1,1]$. Pour pallier à ce

problème, nous proposons de faire, soit une limitation de la commande, soit un choix des fonctions d'appartenance qui s'annulent en dehors de l'intervalle $[-1,1]$.

Pour le cas des régulateurs de type de Mamdani, nous n'avons pas remarqué ce problème, car il est, tout simplement, éliminé par l'étape de défuzzification.

IV.6.4.3. RUPTURE DE COMMANDE:

Pendant le fonctionnement du robot, la commande du sous-système n°1 est annulée pour simuler une défaillance au niveau RLF-actionneur. Les résultats sont présentés par les figures (IV-24;a,b,c). Nous avons remarqué la perte de contrôle de l'articulation n°1. Les deux autres articulations fonctionnent normalement, à cause du découplage pour l'articulation n°2, et de la décentralisation pour l'articulation n°3.

IV.6.5. CONCLUSION:

Nous avons ainsi réalisé des RLF de type de Sugeno qui, comme le présentent les figures (IV-25.a,b,c), ne sont pas meilleurs que ceux de type de Mamdani en ce qui concerne la précision, mais la différence est très faible. Dans l'autre partie, nous avons obtenu des régulateurs qui, au point de vue simplicité et vitesse, sont formidables. Quatre règles suffisent pour commander chaque sous-système.

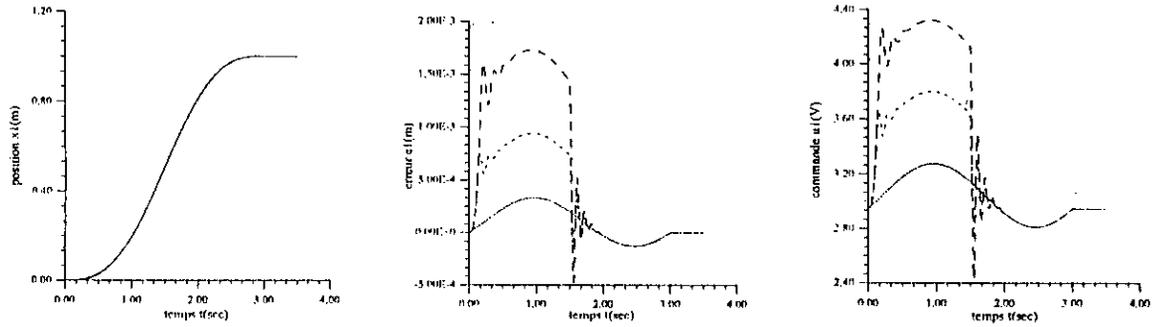


Figure (IV-17.a). Réponse de l'articulation 2
position, erreur et commande
RLF type Sugeno 3x3

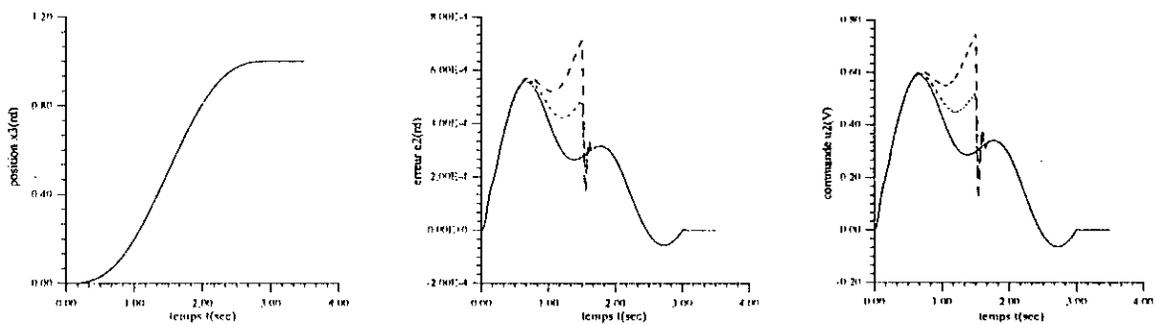


Figure (IV-17.b). Réponse de l'articulation 1
position, erreur et commande
RLF type Sugeno 3x3

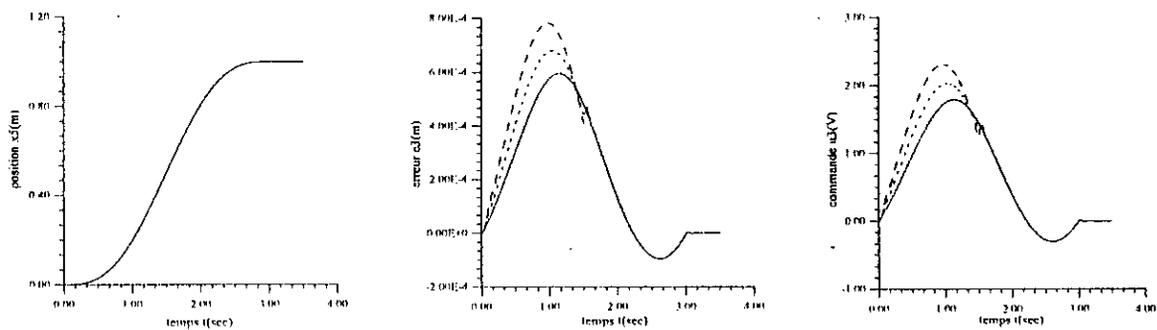


Figure (IV-17.c). Réponse de l'articulation 3
position, erreur et commande
RLF type Sugeno 3x3

— Référence
 $m_0=50\%$ de m_3
 - - - - $m_0=100\%$ de m_3
 - - - - $m_0=0$ kg

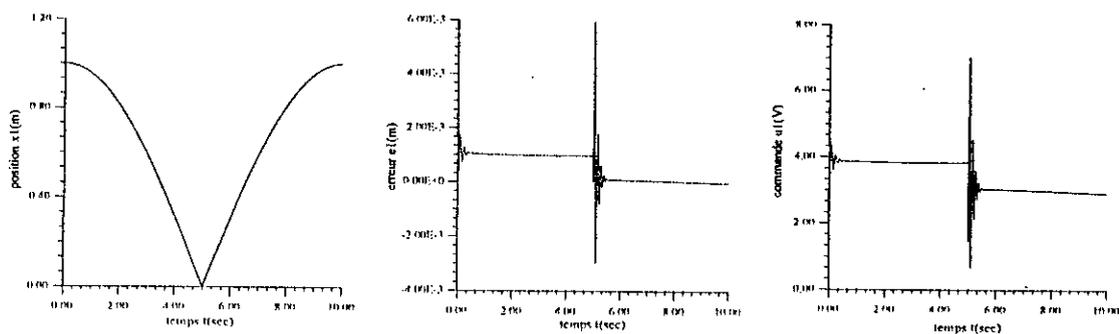


Figure (IV-18.a). Fenêtre de Viviani, Art.2
position, erreur, commande
RLF type Sugeno 3x3

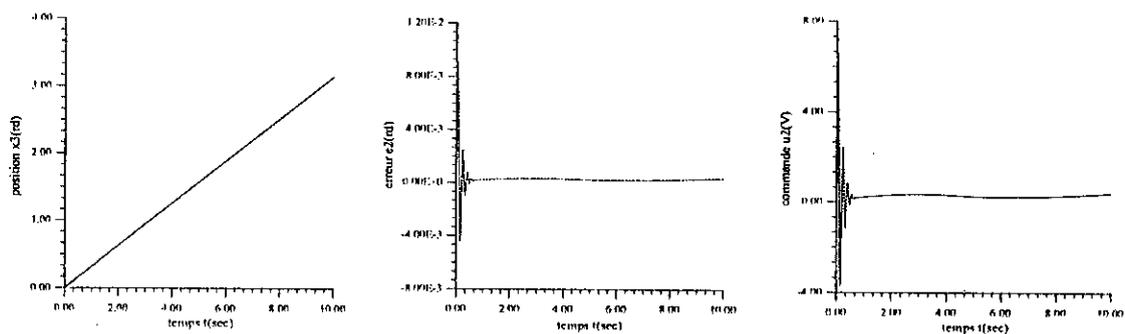


Figure (IV-18.b). Fenêtre de Viviani, Art.1
position, erreur, commande
RLF type Sugeno 3x3

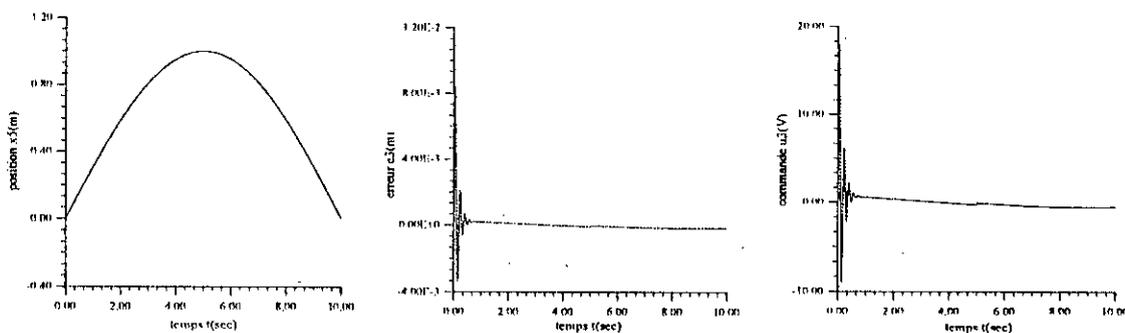


Figure (IV-18.c). Fenêtre de Viviani, Art.3
position, erreur, commande
RLF type Sugeno 3x3

— Référence

--- Réponse

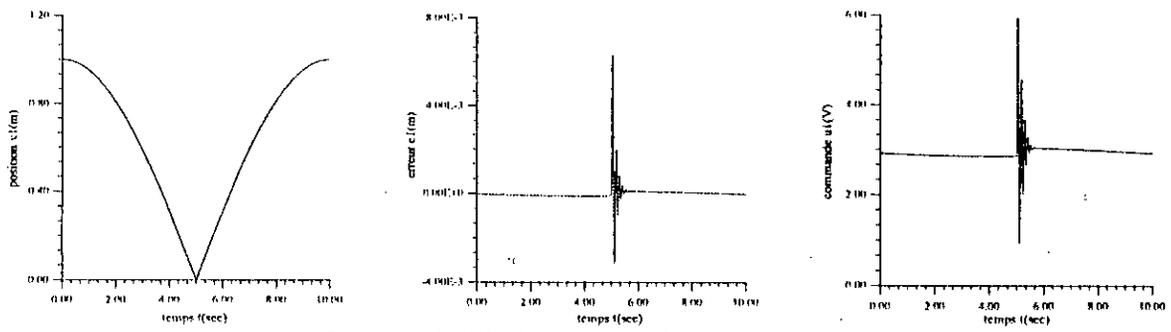


Figure (IV-20.a). Fenêtre de Viviani, Art.2
position, erreur, commande
RLF type Sugeno 2x2

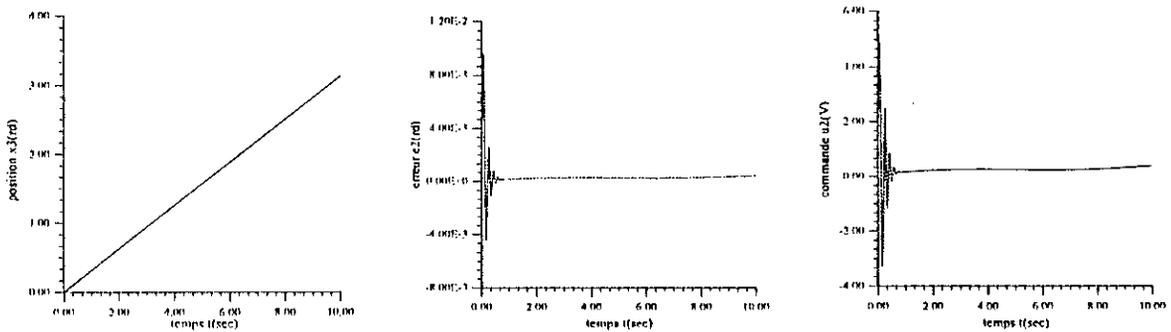


Figure (IV-20.b). Fenêtre de Viviani, Art.1
position, erreur, commande
RLF type Sugeno 2x2

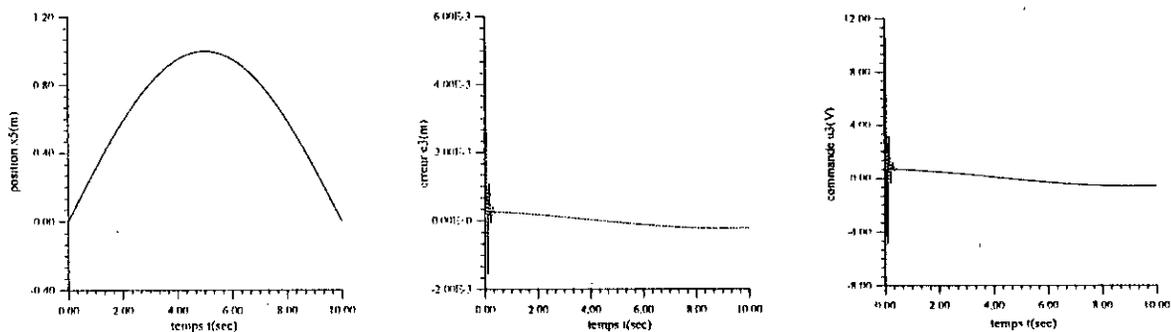


Figure (IV-20.c). Fenêtre de Viviani, Art.3
position, erreur, commande
RLF type Sugeno 2x2

— Référence

— Réponse

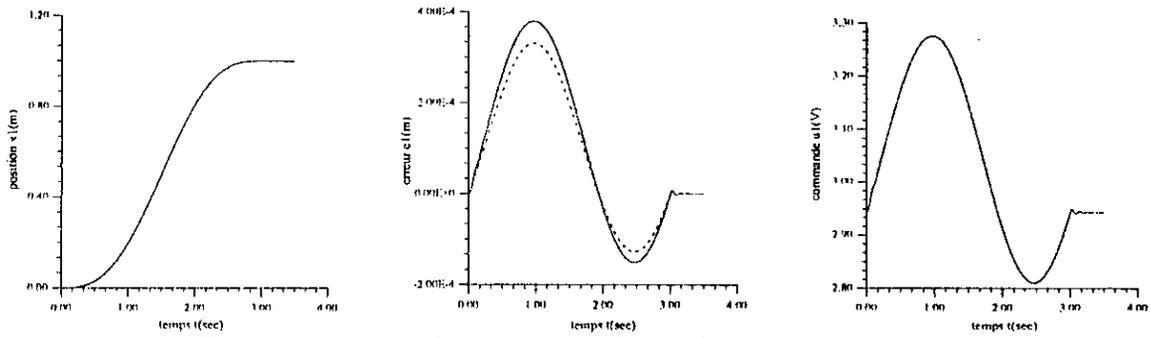


Figure (IV-21.a). Comparaison des 2 régulateurs pour l'art.2

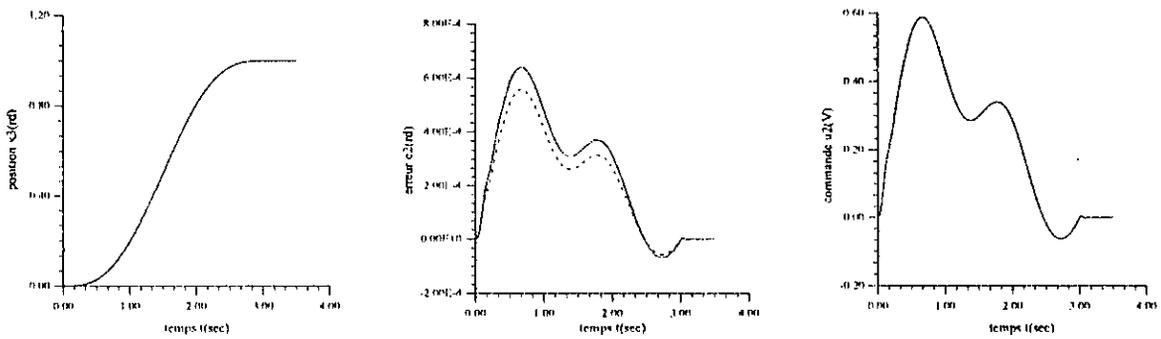


Figure (IV-21.b). Comparaison des 2 régulateurs pour l'art.1

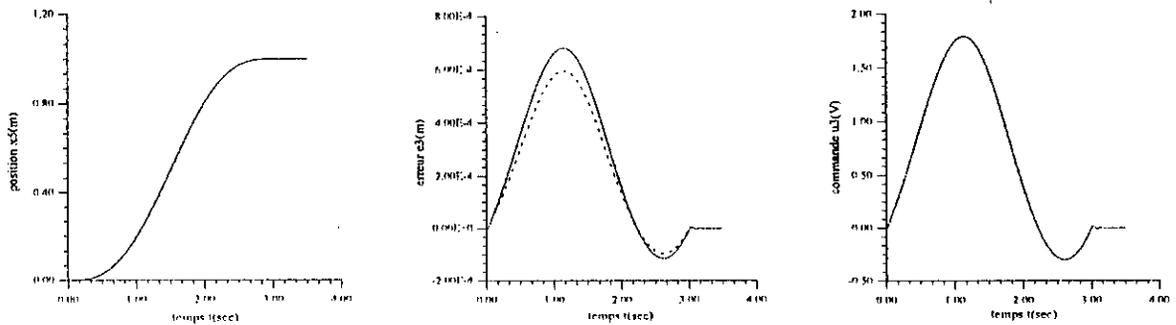


Figure (IV-21.c). Comparaison des 2 régulateurs pour l'art.3

Référence
 RLF type Sugeno 2x2
 RLF type Sugeno 3x3

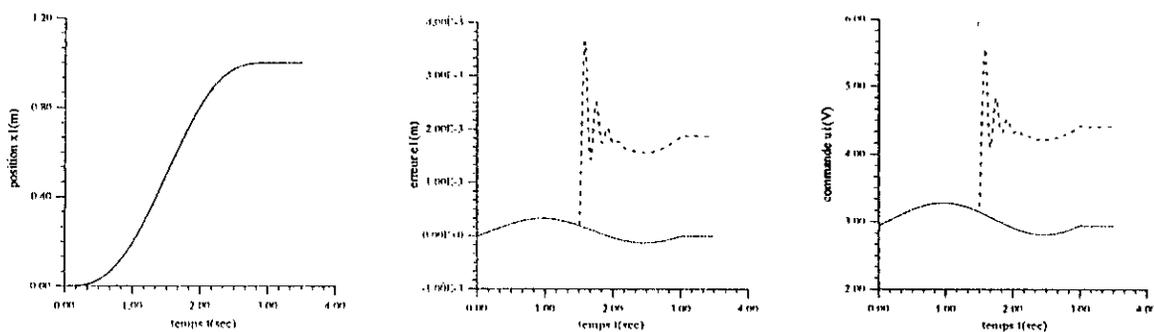


Figure (IV-22.a). RLF 3x3 type Sugeno, Variations des paramètres, Art. 2

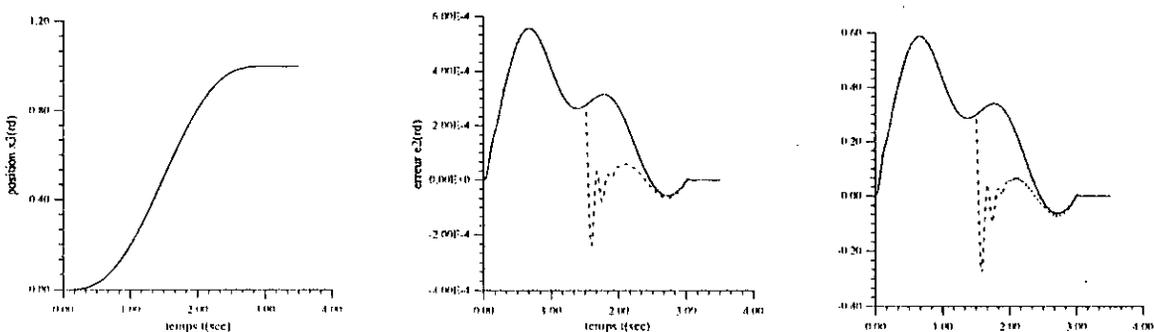


Figure (IV-22.b). RLF 3x3 type Sugeno, Variations des paramètres, Art. 1

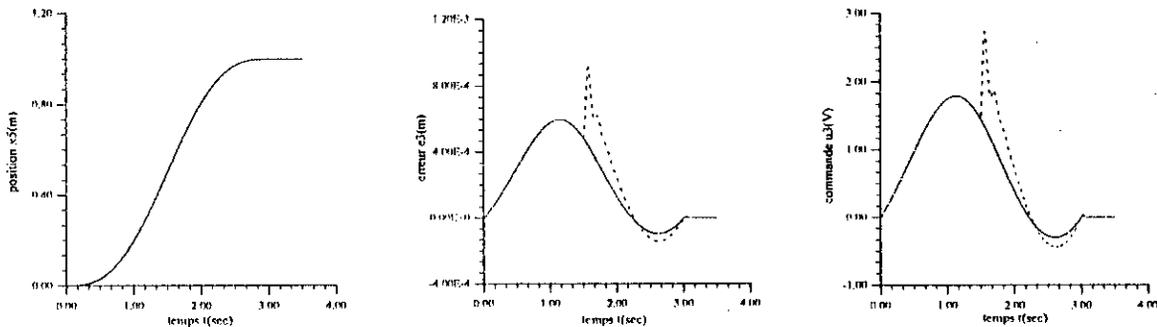


Figure (IV-22.c). RLF 3x3 type Sugeno, Variations des paramètres, Art. 3

——— Référence - - - - - Sans variations de paramètres
 - - - - - Avec variations de paramètres

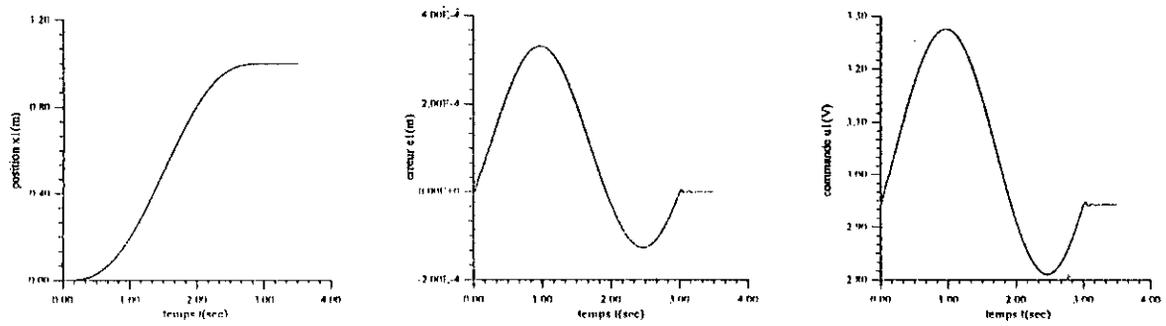


Figure (IV-23.a). RLF 3x3 type Sugeno, capteur n°3 défaillant, art.2

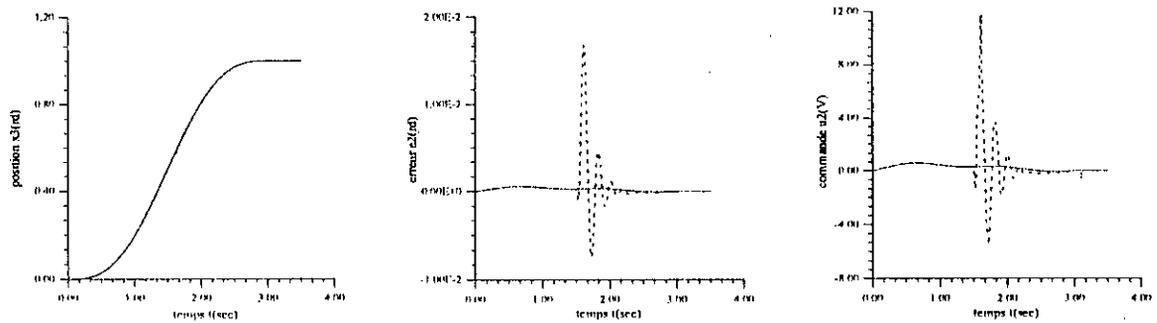


Figure (IV-23.b). RLF 3x3 type Sugeno, capteur n°3 défaillant, art.1

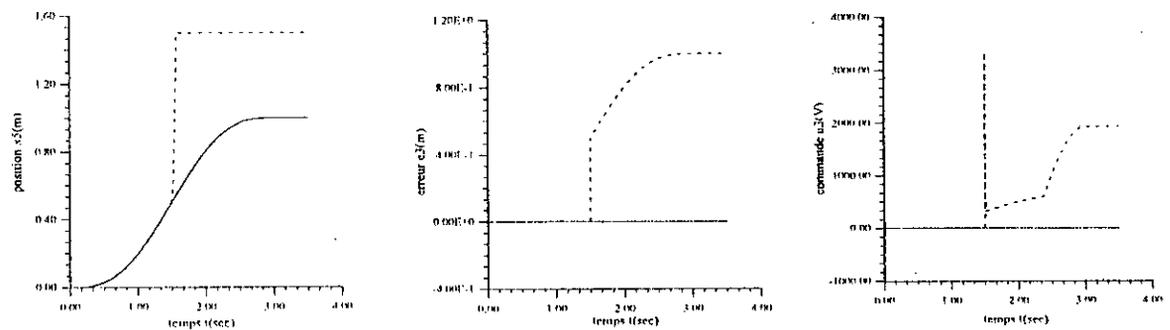


Figure (IV-23.c). RLF 3x3 type Sugeno, capteur n°3 défaillant, art.3

Référence
 Capteur n°3 non défaillant
 Capteur n°3 défaillant

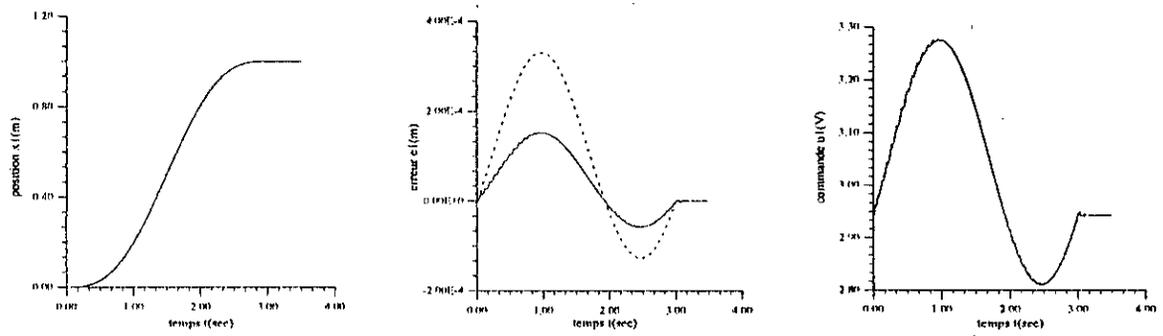


Figure (IV-25.a). Comparaison entre le RLF 3x3 type Mamdani et le RLF 3x3 type Sugeno, Art. 2

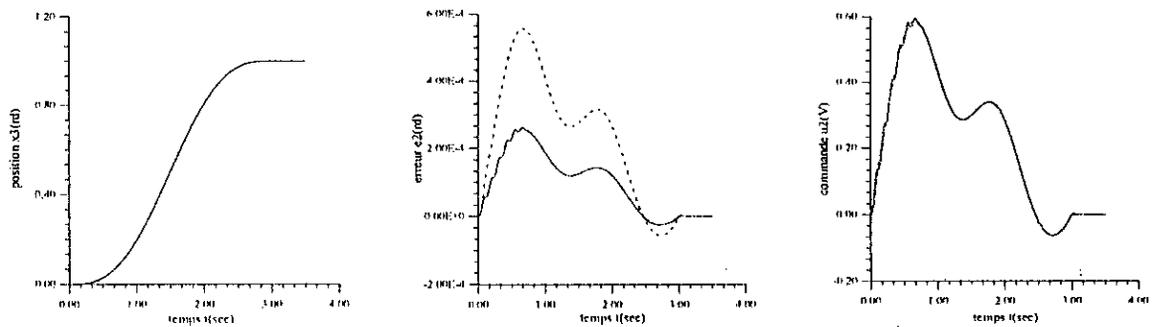


Figure (IV-25.b). Comparaison entre le RLF 3x3 type Mamdani et le RLF 3x3 type Sugeno, Art. 1

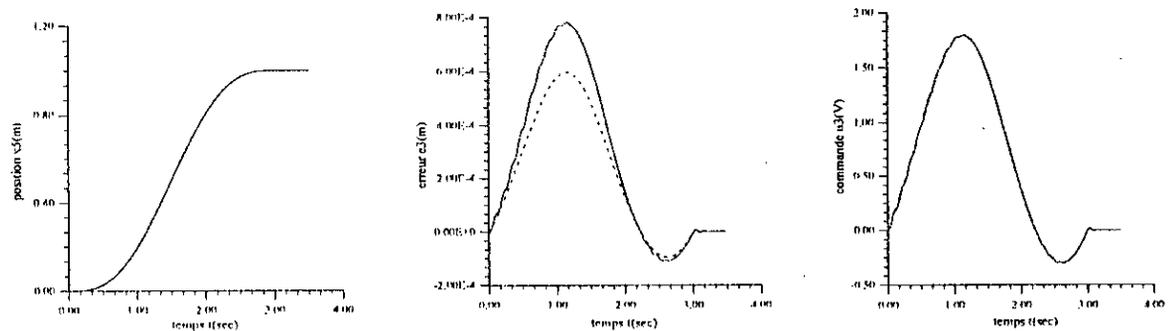


Figure (IV-25.c). Comparaison entre le RLF 3x3 type Mamdani et le RLF 3x3 type Sugeno, Art. 3

— Référence
 - - - - - RLF 3x3 type Mamdani
 RLF 3x3 type Sugeno

IV.7. REGULATEURS NEURO-FLOUS:

IV.7.1. INTRODUCTION:

Les techniques d'apprentissage réalisées pour les réseaux de neurones peuvent être utilisées avec les réseaux neuro-flous. Dans cette partie, nous allons utiliser une parmi elles, celle connue sous le nom de Temporal Back-Propagation, TBP.

IV.7.2. SYNTHESE DES REGULATEURS:

Après avoir ajusté les gains de normalisation (Mamdani), et trouvé les coefficients des conclusions (Sugeno), nous allons, maintenant, nous intéresser aux fonctions d'appartenance.

La structure de réglage par la méthode de Sugeno (RLF 2x2) reste comme précédemment, seulement nous allons ajouter un critère à minimiser.

L'apprentissage par TBP minimise ce critère en ajustant les paramètres des fonctions d'appartenance, qui sont choisies dérivables (gaussiennes).

Nous prenons les paramètres, des fonctions d'appartenance, obtenus après l'apprentissage et les implémenterons dans les RLF 2x2 de Sugeno.

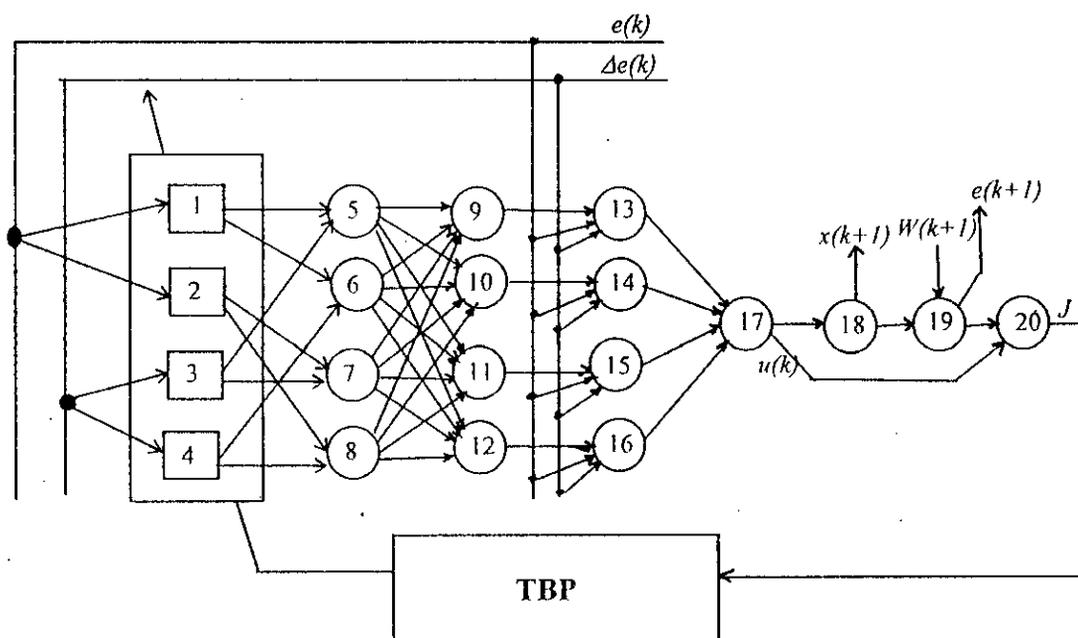


Figure (IV-26). Structure de réglage par réseau neuro-flou, représentation en couches.

VI.7.2.1. APPRENTISSAGE PAR TBP:

Pour pouvoir tirer l'algorithme d'apprentissage, nous utiliserons la représentation en ordre topologique, donnée par la figure (IV-27), pour un seul sous-système. Pour les autres sous-systèmes, l'approche est similaire.

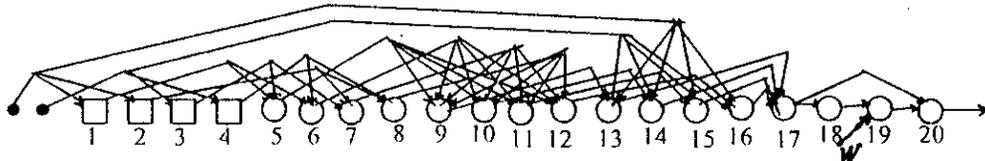


Figure (IV-27). Représentation en ordre topologique, Approche TBP.

L'approche TBP considère chaque fonction dans le régulateur, et le sous-système correspondant comme étant des nœuds, ainsi la structure globale aura la forme d'un réseau (figure(IV-26)).

Le sous-système est représenté par le nœud (18), l'erreur est calculée par le nœud (19) et le critère est donné par le nœud (20).

- *feedforward:*

D'abord, l'erreur normalisée et la variation de l'erreur normalisée sont calculées. Après, elles sont injectées dans le réseau, dont les équations sont données ci-après:

O_j : sortie du nœud (j),

e_n : erreur normalisée à l'instant k ,

de_n : variation de l'erreur normalisée à l'instant k ,

e_p : erreur précédente.

$$\left\{ \begin{array}{l} e_n = \frac{O_{19}}{g_1} \\ de = O_{19} - e_p \\ de_n = \frac{de}{g_2} \\ e_p = O_{19} \end{array} \right. \quad (IV-6)$$

La couche (1) contient les nœuds correspondant aux fonctions d'appartenance.

c_i, s_i : paramètres des fonctions d'appartenance.

$$\text{Couche 1} \left\{ \begin{array}{l} O_1 = \exp \left\{ - \left(\frac{e_n - c_1}{s_1} \right)^2 \right\} \\ O_2 = \exp \left\{ - \left(\frac{e_n - c_2}{s_2} \right)^2 \right\} \\ O_3 = \exp \left\{ - \left(\frac{de_n - c_3}{s_3} \right)^2 \right\} \\ O_4 = \exp \left\{ - \left(\frac{de_n - c_4}{s_4} \right)^2 \right\} \end{array} \right. \quad (\text{IV-7})$$

La couche (2) fait la conjonction des prémisses.

$$\text{Couche 2} \left\{ \begin{array}{l} O_5 = O_1 \cdot O_3 \\ O_6 = O_1 \cdot O_4 \\ O_7 = O_2 \cdot O_3 \\ O_8 = O_2 \cdot O_4 \end{array} \right. \quad (\text{IV-8})$$

La couche (3) normalise les coefficients de vérité associés aux conclusions.

$$\text{Couche 3} \left\{ \begin{array}{l} O_9 = \frac{O_5}{O_5 + O_6 + O_7 + O_8} \\ O_{10} = \frac{O_6}{O_5 + O_6 + O_7 + O_8} \\ O_{11} = \frac{O_7}{O_5 + O_6 + O_7 + O_8} \\ O_{12} = \frac{O_8}{O_5 + O_6 + O_7 + O_8} \end{array} \right. \quad (\text{IV-9})$$

La couche (4) calcule les conclusions et les multiplie par les coefficients de vérité normalisés (coefficients de pondération) correspondants.

$$\text{Couche 4} \left\{ \begin{array}{l} O_{13} = O_9 \cdot f_1 \\ O_{14} = O_{10} \cdot f_2 \\ O_{15} = O_{11} \cdot f_3 \\ O_{16} = O_{12} \cdot f_4 \end{array} \right. \quad (\text{IV-10})$$

$$\text{avec } f_i = p_i \cdot e_n + q_i \cdot de_n + r_i \quad i = \overline{1,4}$$

La couche (5) calcule la commande. O_{17} est la commande.

$$\text{Couche 5} \left\{ O_{17} = g_3 \cdot (O_{13} + O_{14} + O_{15} + O_{16}) \right. \quad (\text{IV-11})$$

La couche (6) représente le sous-système lui-même. O_{18} est la sortie du sous-système (position).

$$\text{Couche 6 } \{O_{18} = S(O_{17}) \quad (\text{IV-12})$$

La couche (7) calcule l'erreur à l'instant $k+1$. W est la consigne.

$$\text{Couche 7 } \{O_{19} = W - O_{18} \quad (\text{IV-13})$$

La couche (8) calcule le critère à minimiser. λ est un coefficient de pondération.

$$\text{Couche 8 } \{O_{20} = O_{19}^2 + \lambda O_{17}^2 \quad (\text{IV-14})$$

- **algorithme TBP:**

Le critère à minimiser est le suivant:

$$J = e^2 + \lambda u^2 \quad (\text{IV-15})$$

$$J = O_{19}^2 + \lambda O_{17}^2$$

En utilisant la représentation en ordre topologique, la rétropropagation du critère devient facile à calculer. Le calcul est basé sur la notion de dérivée ordonnée.

$$\varepsilon_i = \frac{\partial^+ J}{\partial O_i} = \frac{\partial J}{\partial O_i} + \sum_{i < j \leq 19} \frac{\partial^+ J}{\partial O_j} \cdot \frac{\partial O_j}{\partial O_i} \quad i = \overline{19,1} \quad (\text{IV-16})$$

Le calcul donne:

$$\varepsilon_{19} = 2 \cdot O_{19}$$

$$\varepsilon_{18} = -\varepsilon_{19}$$

$$\varepsilon_{17} = 2 \cdot \lambda \cdot O_{17} + \varepsilon_{18} \cdot \frac{\Delta x}{\Delta u} \quad \text{avec } \frac{\Delta x}{\Delta u} \approx \frac{\partial S}{\partial u}$$

$$\varepsilon_{16} = \varepsilon_{17} \cdot g_3$$

$$\varepsilon_{15} = \varepsilon_{17} \cdot g_3$$

$$\varepsilon_{14} = \varepsilon_{17} \cdot g_3$$

$$\varepsilon_{13} = \varepsilon_{17} \cdot g_3$$

$$\varepsilon_{12} = \varepsilon_{16} \cdot f_4$$

$$\varepsilon_{11} = \varepsilon_{15} \cdot f_3$$

$$\varepsilon_{10} = \varepsilon_{14} \cdot f_2$$

$$\varepsilon_9 = \varepsilon_{13} \cdot f_1$$

(IV-17)

$$\begin{aligned} \varepsilon_8 &= \frac{O_5 \cdot (\varepsilon_{12} - \varepsilon_9) + O_6 \cdot (\varepsilon_{12} - \varepsilon_{10}) + O_7 \cdot (\varepsilon_{12} - \varepsilon_{11})}{[O_5 + O_6 + O_7 + O_8]^2} \\ \varepsilon_7 &= \frac{O_5 \cdot (\varepsilon_{11} - \varepsilon_9) + O_6 \cdot (\varepsilon_{11} - \varepsilon_{10}) + O_8 \cdot (\varepsilon_{11} - \varepsilon_{12})}{[O_5 + O_6 + O_7 + O_8]^2} \\ \varepsilon_6 &= \frac{O_5 \cdot (\varepsilon_{10} - \varepsilon_9) + O_7 \cdot (\varepsilon_{10} - \varepsilon_{11}) + O_8 \cdot (\varepsilon_{10} - \varepsilon_{12})}{[O_5 + O_6 + O_7 + O_8]^2} \\ \varepsilon_5 &= \frac{O_6 \cdot (\varepsilon_9 - \varepsilon_{10}) + O_7 \cdot (\varepsilon_9 - \varepsilon_{11}) + O_8 \cdot (\varepsilon_9 - \varepsilon_{12})}{[O_5 + O_6 + O_7 + O_8]^2} \end{aligned} \tag{IV-18}$$

$$\begin{aligned} \varepsilon_4 &= O_1 \cdot \varepsilon_6 + O_2 \cdot \varepsilon_8 \\ \varepsilon_3 &= O_1 \cdot \varepsilon_5 + O_2 \cdot \varepsilon_7 \\ \varepsilon_2 &= O_3 \cdot \varepsilon_7 + O_4 \cdot \varepsilon_8 \\ \varepsilon_1 &= O_3 \cdot \varepsilon_5 + O_4 \cdot \varepsilon_6 \end{aligned} \tag{IV-19}$$

avec ε_i la dérivée à transmettre du nœud (i+1) vers le nœud (i).

• *ajustement des paramètres:*

Les paramètres sont ajustés par la formule classique du gradient.

Si α est un paramètre alors il est ajusté suivant la formule suivante:

$$\Delta\alpha = -\eta \frac{\partial^+ J}{\partial \alpha} \tag{IV-20}$$

η est le pas d'apprentissage, il est pris variable suivant la formule:

$$\eta = \frac{SS}{\sqrt{\sum_{\alpha} \left(\frac{\partial^+ J}{\partial \alpha} \right)^2}} \tag{IV-21}$$

SS : step size, pris fixe et faible.

Pour ajuster les paramètres, nous avons

$$\frac{\partial^+ J}{\partial \alpha} = \varepsilon_i \cdot \frac{\partial O_i}{\partial \alpha} \tag{IV-22}$$

Puisque nous avons utilisé deux classes d'appartenance pour chaque entrée des régulateurs, et puisque nous avons constaté une symétrie dans les tables de règles de Mamdani et de Sugeno, nous continuons à admettre cette symétrie. En utilisant cette symétrie, nous

n'allons pas ajuster les paramètres de toutes les fonctions d'appartenance, mais, seulement ceux des classes N . Le nombre de paramètres à ajuster se réduit donc à quatre par régulateur.

Les paramètres des conclusions sont pris fixes, et les autres paramètres des fonctions d'appartenance suivent les relations de symétrie données en bas.

$$\begin{aligned} \frac{\partial^+ J}{\partial c_1} &= \varepsilon_1 \cdot \frac{\partial O_1}{\partial c_1} \\ \frac{\partial^+ J}{\partial c_3} &= \varepsilon_3 \cdot \frac{\partial O_3}{\partial c_3} \\ \frac{\partial^+ J}{\partial s_1} &= \varepsilon_1 \cdot \frac{\partial O_1}{\partial s_1} \\ \frac{\partial^+ J}{\partial s_3} &= \varepsilon_3 \cdot \frac{\partial O_3}{\partial s_3} \end{aligned} \tag{IV-23}$$

Les dérivées des fonctions d'appartenance par rapport aux paramètres sont données par

$$\begin{aligned} \frac{\partial O_i}{\partial c_i} &= 2 \cdot \frac{v_i - c_i}{s_i^2} \cdot O_i \\ \frac{\partial O_i}{\partial s_i} &= 2 \cdot \frac{(v_i - c_i)^2}{s_i^3} \cdot O_i \\ i &= 1,3 \\ v_1 &= e_n \\ v_3 &= de_n \end{aligned} \tag{IV-24}$$

Les relations de symétrie

$$c_2 = -c_1, s_2 = s_1, c_4 = -c_3, s_4 = s_3 \tag{IV-25}$$

IV.7.2.2. RESULTAT DE L'APPRENTISSAGE:

Les résultats de l'apprentissage sont donnés par le tableau suivant:

	c_1	s_1	c_3	s_3
<i>avant app.</i>	-0.5	0.6	-0.5	0.6
<i>après app. s-s1</i>	-0.667	0.395	-0.652	0.430
<i>après app. s-s2</i>	-0.599	0.489	-0.589	0.513
<i>après app. s-s3</i>	-0.574	0.518	-0.568	0.536

Tableau (IV-9). Paramètres des fonctions d'appartenance, avant et après apprentissage.

Ainsi, les figures (IV-28,29) présentent les formes des fonctions d'appartenance avant et après l'apprentissage.

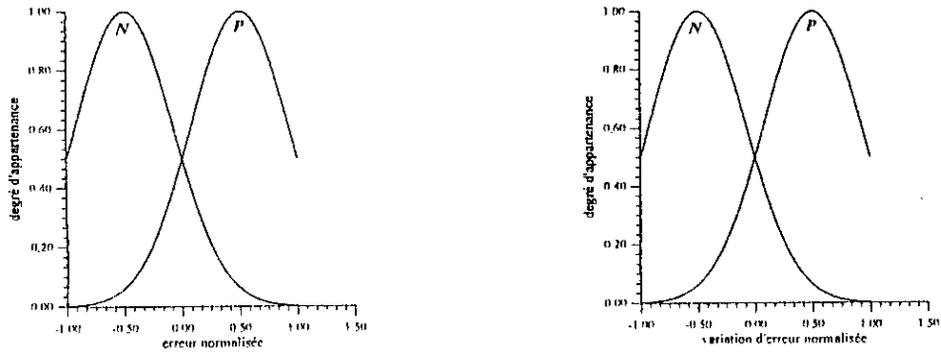
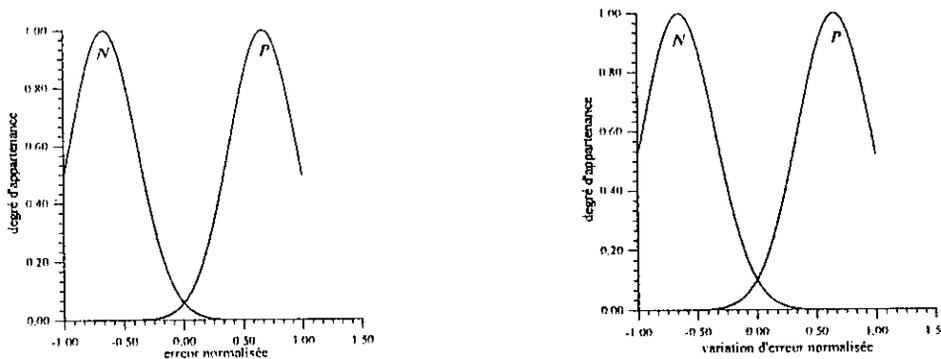
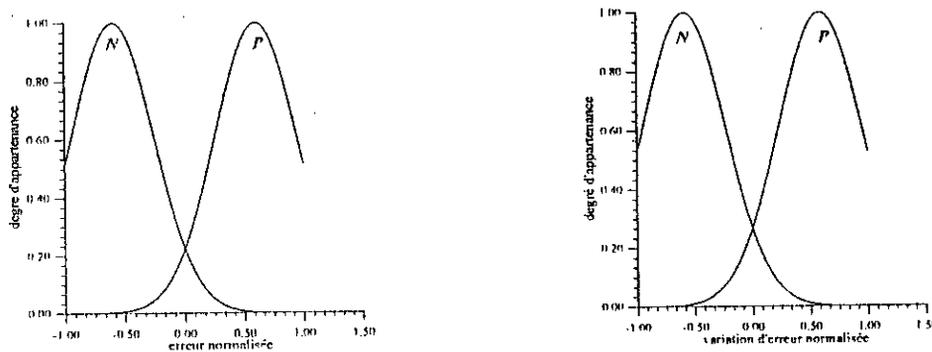


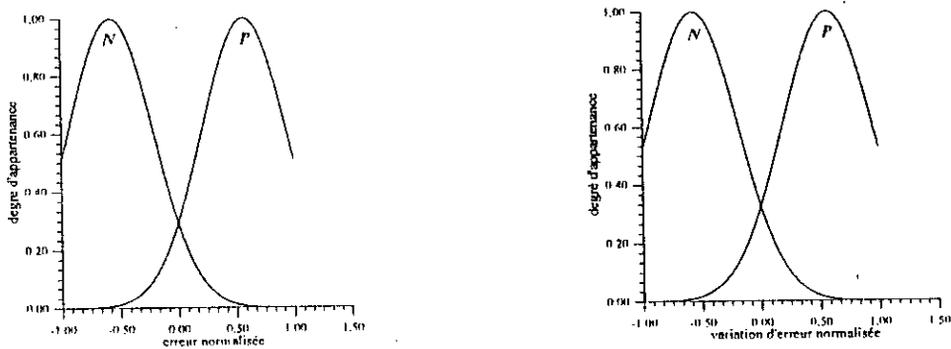
Figure (IV-28). Fonctions d'appartenance, avant l'apprentissage, tous les sous-systèmes.



a) sous-système n°1



b) sous-système n°2



c) sous-système n°3

Figure (IV-29). Fonctions d'appartenance, après l'apprentissage.

IV.7.3. ETUDE COMPARATIVE:**IV.7.3.1. ASPECT GENERAL DE LA REPONSE:**

Les résultats sont présentés par les figures (IV-30,a,b,c), (IV-31.a,b,c). Nous remarquons une bonne poursuite. Les erreurs sont encore plus faibles et l'aspect général de la proportionnalité entre l'erreur et la commande est conservé.

IV.7.3.2. COMPARAISON ET JUSTIFICATION:

Les figures (IV-35.a,b,c) montrent que les RLF 2x2 de type de Sugeno obtenus après l'apprentissage sont meilleurs.

Par rapport aux RLF 3x3 de type de Sugeno, nous observons une grande amélioration, car les erreurs sont devenues plus faibles. En plus, il y a quatre règles par régulateur au lieu de neuf.

Par rapport aux RLF 3x3 de type de Mamdani, le temps de calcul de la commande est réduit énormément.

IV.7.4. TESTS DE ROBUSTESSE:

Les tests de robustesse donnent les résultats représentés par les figures (IV-32.a,b,c), (IV-33.a,b,c), (IV-34.a,b,c).

IV.7.4.1. VARIATION PARAMETRIQUE:

Les résultats de ce test sont comparables aux résultats du cas précédent. Ainsi, nous remarquons l'existence d'une erreur statique au niveau de l'articulation n°2. (figure(IV-32.a,b,c)).

IV.7.4.2. CAPTEUR DEFAILLANT:

Le choix des fonctions gaussiennes a éliminé le problème de limitation de la commande. Nous constatons l'effet de la décentralisation, et la réponse de l'articulation n°1 est peu affectée par la défaillance du capteur n°3. (figure(IV-33.a,b,c)).

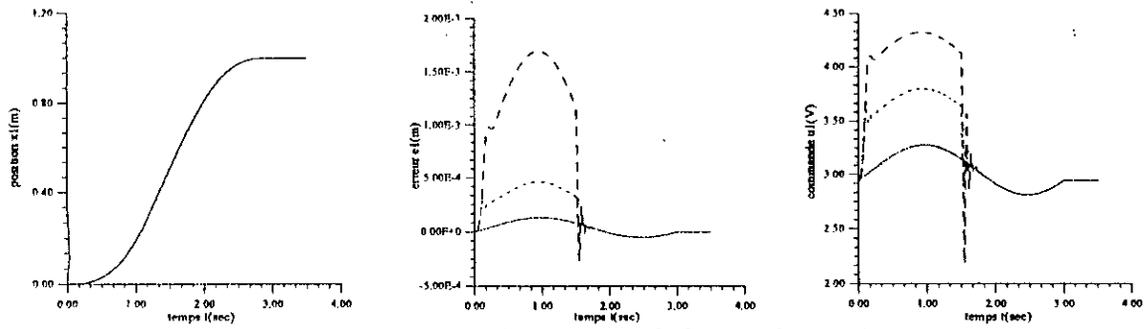
IV.7.4.3. RUPTURE DE LA COMMANDE:

Comme il est montré par les figures (IV-34.a,b,c), ce test a un effet sur l'articulation n°1 seulement. Les deux autres suivent les trajectoires correspondants, sans aucune perturbation pour l'articulation n°2 car elle est parfaitement découplée, et sans une perturbation apparente pour l'articulation n°3, à cause de la décentralisation.

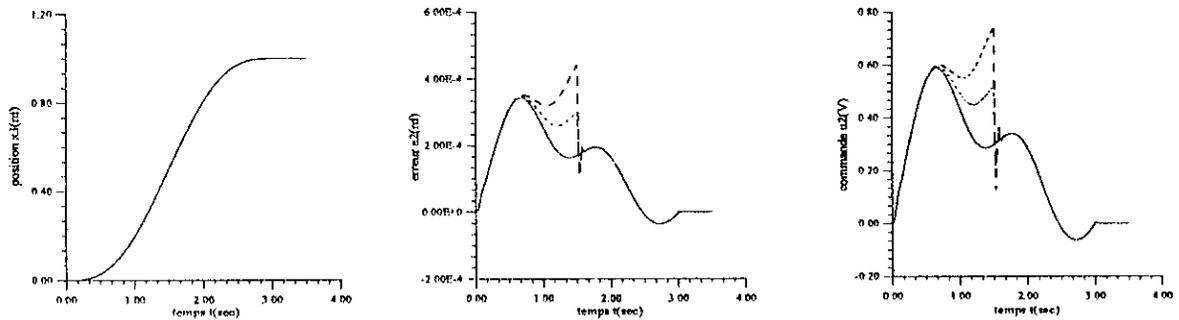
IV.7.5. CONCLUSION:

Nous avons ainsi obtenu des régulateurs, à la fois, rapides, simples, réduits et plus précis.

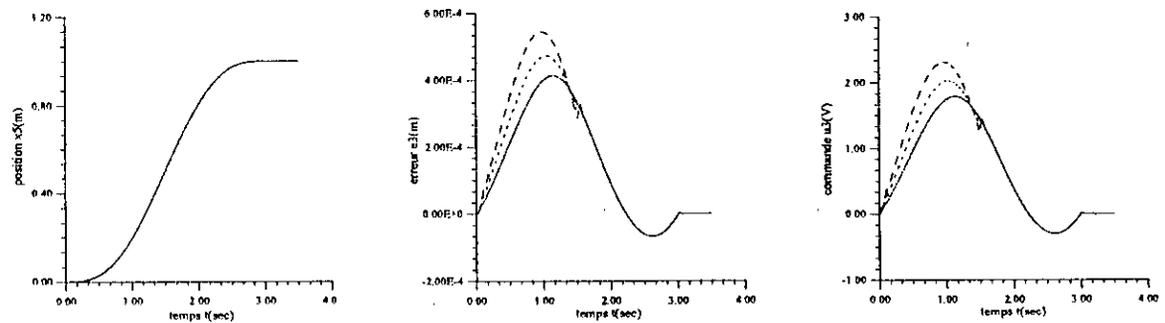
Les tests de robustesse ont montré l'efficacité des régulateurs.



Figure(IV-30.a). Réponse de l'articulation 2
position, erreur et commande
RLF type Sugeno 2x2 après apprentissage



Figure(IV-30.b). Réponse de l'articulation 1
position, erreur et commande
RLF type Sugeno 2x2 après apprentissage



Figure(IV-30.c). Réponse de l'articulation 3
position, erreur et commande
RLF type Sugeno 2x2 après apprentissage

——— Référence ——— $m_0=0$ kg
 - - - - $m_0=50\%$ de m_3 - - - - $m_0=100\%$ de m_3

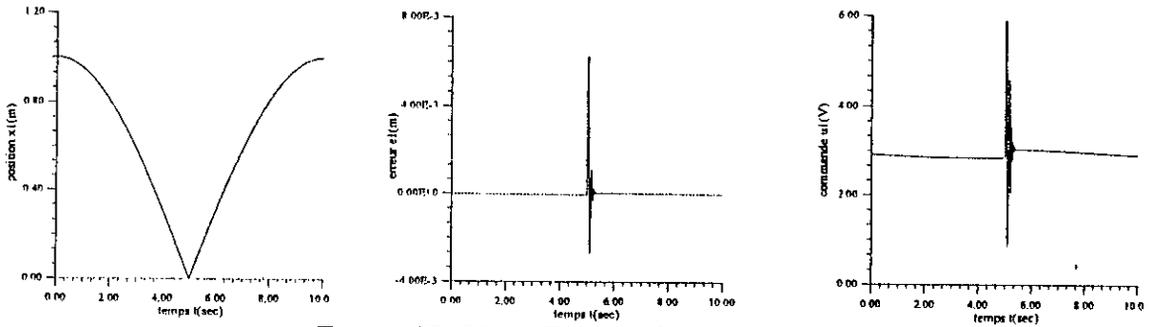


Figure (IV-31.a). Fenêtre de Viviani, Art.2
position, erreur, commande
RLF type Sugeno 2x2 après apprentissage

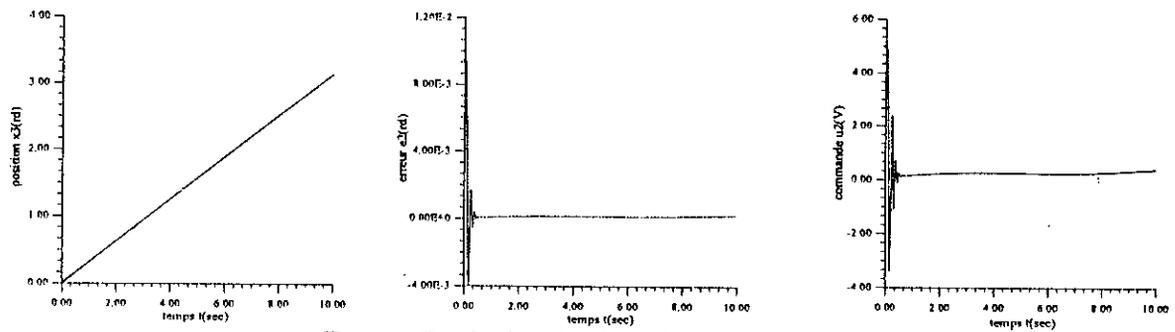


Figure (IV-31.b). Fenêtre de Viviani, Art.1
position, erreur, commande
RLF type Sugeno 2x2 après apprentissage

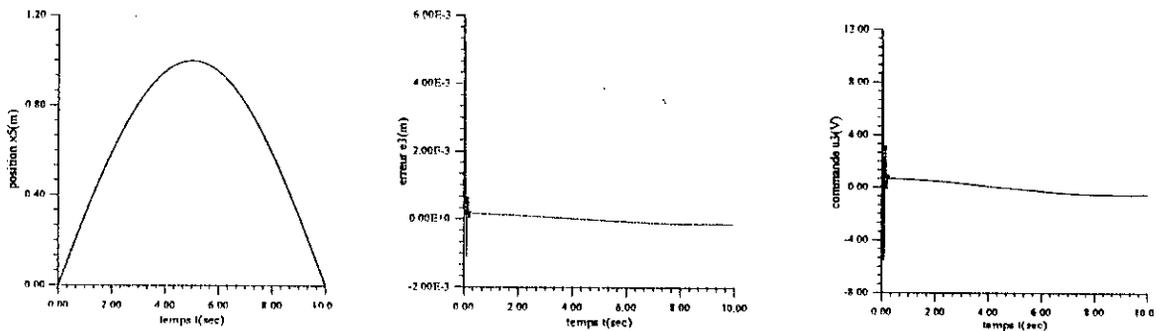


Figure (IV-31.c). Fenêtre de Viviani, Art.3
position, erreur, commande
RLF type Sugeno 2x2 après apprentissage

— Référence

— Réponse

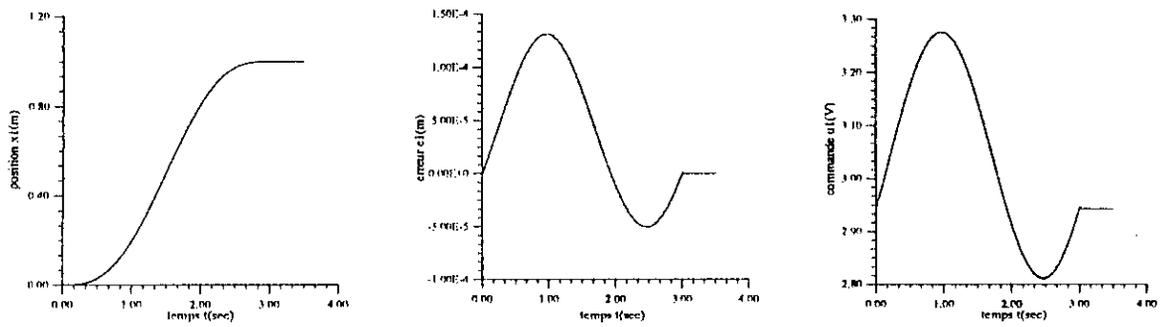


Figure (IV-33.a). Art.2, capteur n°3 défaillant, RLF type Sugeno 2x2 après apprentissage

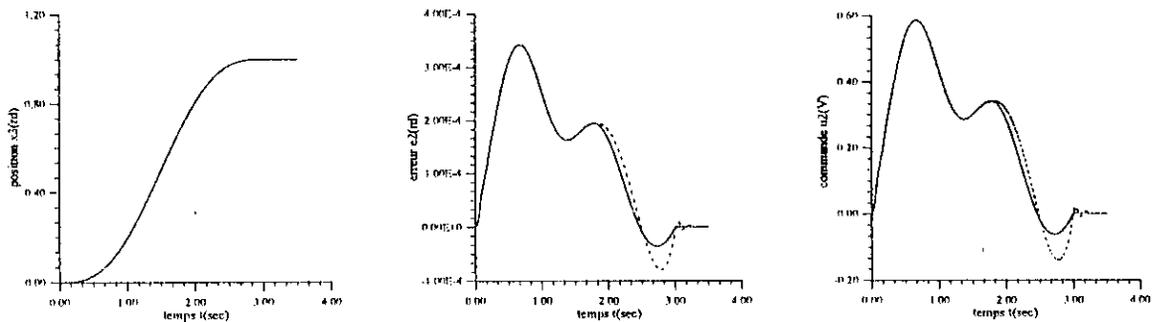


Figure (IV-33.b). Art.1, capteur n°3 défaillant, RLF type Sugeno 2x2 après apprentissage

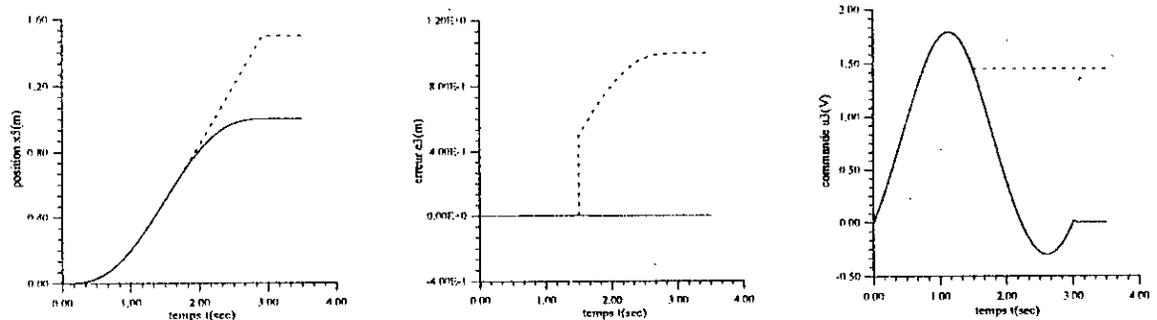


Figure (IV-33.c). Art.3, capteur n°3 défaillant, RLF type Sugeno 2x2 après apprentissage

— Référence

— Capteur n°3 non défaillant

- - - - - Capteur n°3 défaillant

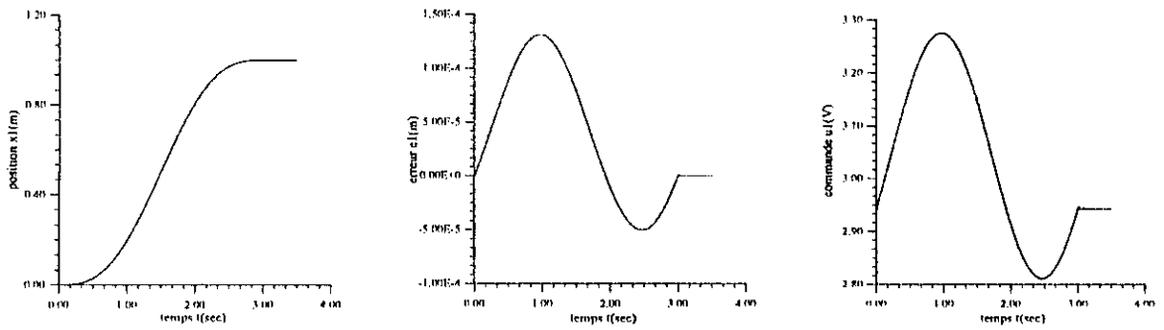


Figure (IV-34.a). Art. 2, rupture de la commande de l'art. 1, RLF 2x2 type Sugeno après apprentissage

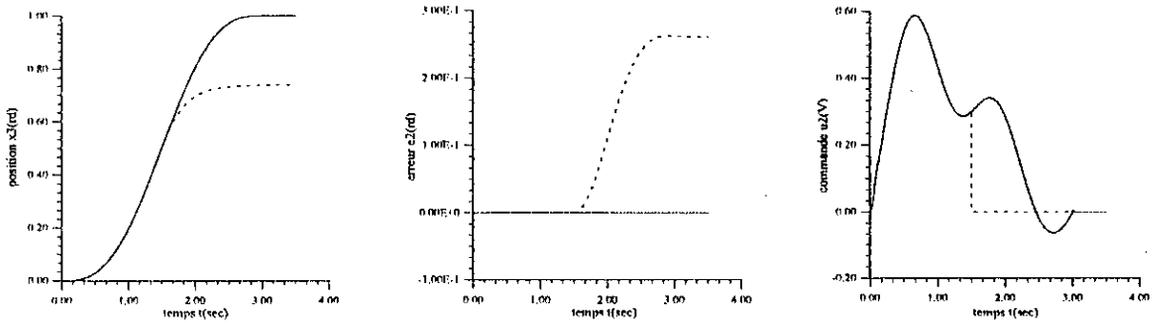


Figure (IV-34.b). Art. 1, rupture de la commande de l'art. 1, RLF 2x2 type Sugeno après apprentissage

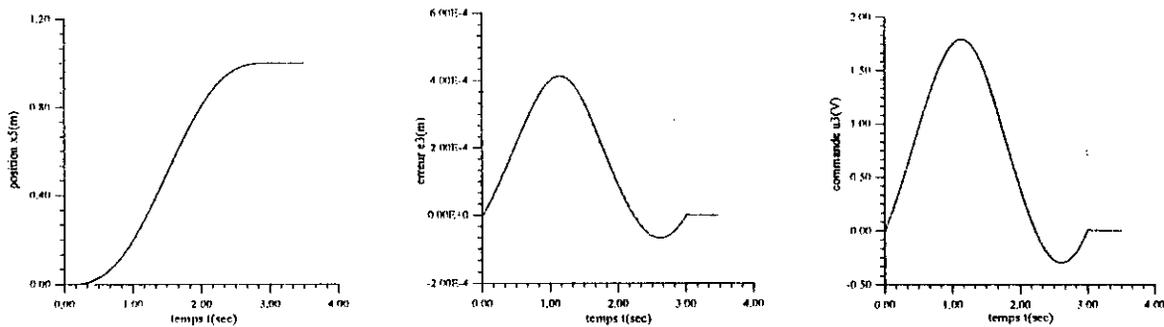


Figure (IV-34.c). Art. 3, rupture de la commande de l'art. 1, RLF 2x2 type Sugeno après apprentissage

——— Référence
 ——— Sans rupture de la commande
 - - - - Avec rupture de la commande

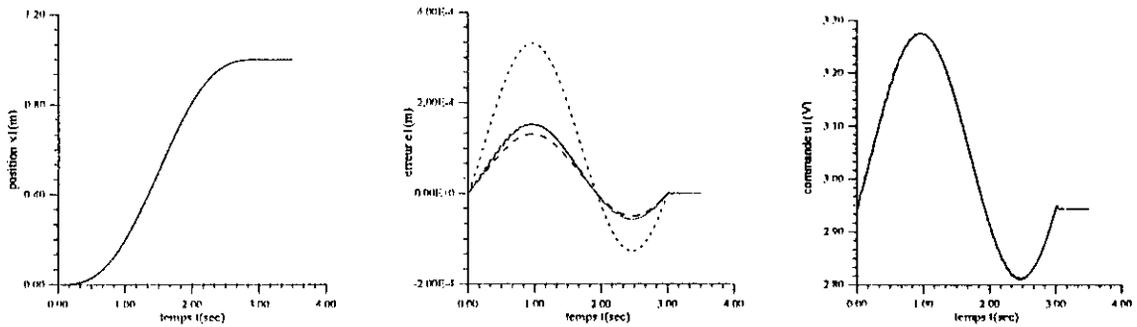


Figure (IV-35.a). Comparaison entre: RLF 3x3 type Mamdani, RLF 3x3 type Sugeno et RLF 2x2 type Sugeno après l'apprentissage. Art. 2.

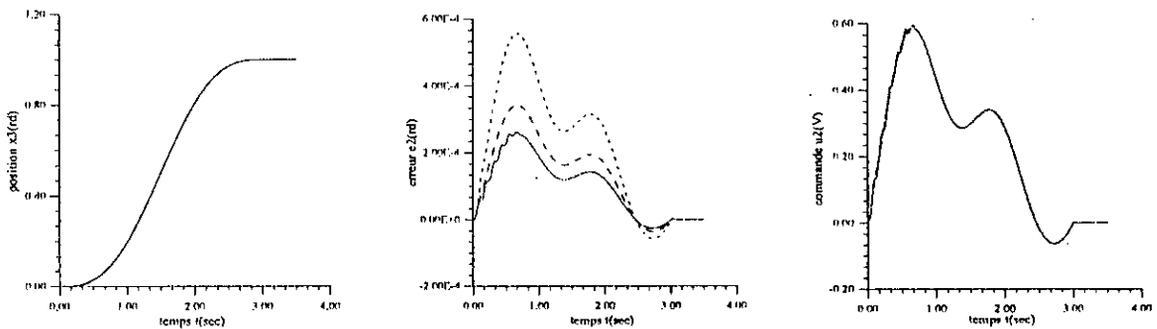


Figure (IV-35.b). Comparaison entre: RLF 3x3 type Mamdani, RLF 3x3 type Sugeno et RLF 2x2 type Sugeno après l'apprentissage. Art. 1.

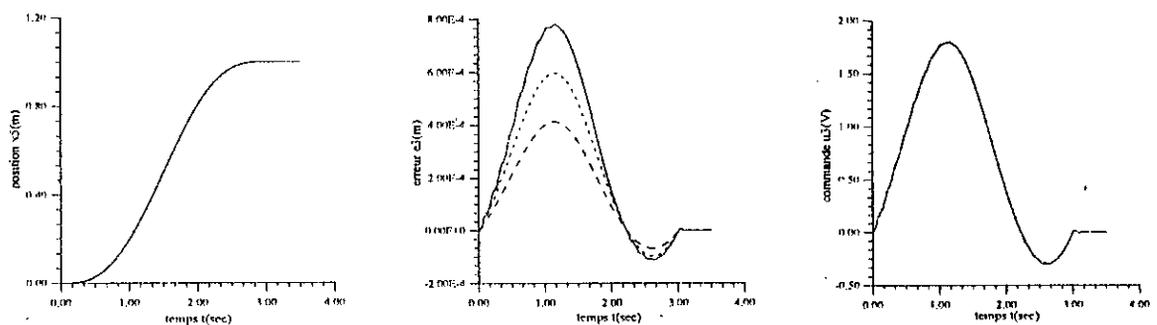


Figure (IV-35.c). Comparaison entre: RLF 3x3 type Mamdani, RLF 3x3 type Sugeno et RLF 2x2 type Sugeno après l'apprentissage. Art. 3.

——— Référence
 - - - - RLF 3x3 type Sugeno
 - - - - RLF 3x3 type Mamdani
 - - - - RLF 2x2 type Sugeno après l'apprentissage

CONCLUSION GENERALE

"la bêtise consiste à vouloir conclure"
GUSTAVE FLAUBERT

CONCLUSION GENERALE

Dans ce mémoire, nous avons appliqué la commande floue et neuro-floue décentralisée à un robot manipulateur. La synthèse des régulateurs a suivi une démarche bien déterminée et bien choisie. Celle-ci nous a amenés, pas à pas, vers les régulateurs voulus. Aussi, nous avons donné une méthode de passage des régulateurs de type de Mamdani à ceux de types de Sugeno, et présenté l'algorithme TBP comme étant un bloc à part ajouté à la structure de réglage par la méthode de Sugeno.

Pour chaque approche (Mamdani, Sugeno, neuro-flou), nous avons présenté les résultats de simulation et de tests de robustesse. Nous avons trouvé de meilleurs résultats en précision, en simplicité, en rapidité et en robustesse. Des études comparatives entre ces approches ont été effectuées et des remarques et résultats ont été tirés.

- Vu la richesse de la commande par logique floue en approche, en types de régulateurs,..., des applications dans le domaine des systèmes complexes, mal modélisés, voire sans modèles mathématiques, nous semblent très intéressantes.

- La méthode d'approximation, nous semble générale, car il suffit de donner la caractéristique d'un régulateur pour réaliser son homologue de sugeno. Même le régulateur de type de Sugeno nous semble être un cas particulier d'une famille de régulateurs ayant comme loi de commande:

$$u = \langle \underline{f} | \underline{g} \rangle, \quad \langle | \rangle \text{ produit scalaire}$$

$$\underline{f} = [f_1 \quad f_2 \quad \dots \quad f_n]^T$$

$$\underline{g} = [g_1 \quad g_2 \quad \dots \quad g_n]^T$$

avec:

f_i fonctions bornées, de même forme et prennent leurs valeurs dans $[0,1]$,

g_i fonctions de même forme.

Pour cela, nous proposons l'étude de ce type de régulateurs.

- Une étude plus détaillée de la commande neuro-floue parait nécessaire, car, dans notre travail, nous l'avons seulement adaptée à notre cas, afin d'avoir un travail homogène ayant des idées et des applications qui se succèdent.

- L'algorithme TBP n'est pas applicable en ligne car il y aura un risque résultant du calcul de $\frac{\Delta x}{\Delta u}$ qui peut être infini.

- En plus, nous proposons l'application de notre travail à d'autres systèmes, car nous avons constaté le caractère général de la démarche suivie, et le modèle du robot n'a été utilisé que pour la simulation.

- Une comparaison avec les autres techniques, montre que le réglage par logique floue utilise un nombre réduit de mesures (de capteurs), et donne de bons résultats.

ANNEXE I

FONCTIONS D'APPARTENANCE

On distingue plusieurs types de fonctions d'appartenance, parmi lesquelles on cite:

- Fonctions triangulaires:

$$tr = \max\left(\min\left(\left(\frac{x-a}{b-a}\right), 1, \left(\frac{c-x}{c-b}\right)\right), 0\right)$$

- Fonctions trapézoïdales:

$$tra = \max\left(\min\left(\left(\frac{x-a}{b-a}\right), 1, \left(\frac{d-x}{d-c}\right)\right), 0\right)$$

- Fonctions gaussienne:

$$gauss = \exp\left\{-\left(\frac{x-c}{s}\right)^2\right\}$$

- Fonctions cloches:

$$clo = \frac{1}{1 + \left(\frac{x-c}{a}\right)^{2h}}$$

- Fonctions sigmoïdes:

$$sig = \frac{1}{1 + e^{-a(x-c)}}$$

ANNEXE II

NORMES ET CONORMES TRIANGULAIRES

Il existe plusieurs types de normes et de conormes triangulaires. Les plus utilisées sont:

- Zadeh:

T-norme : $\min(x, y)$

T-conorme : $\max(x, y)$

- Lukasiewicz:

T-norme : $\max(x + y - 1, 0)$

T-conorme : $\min(x + y, 1)$

- Probabiliste:

T-norme : $x \cdot y$

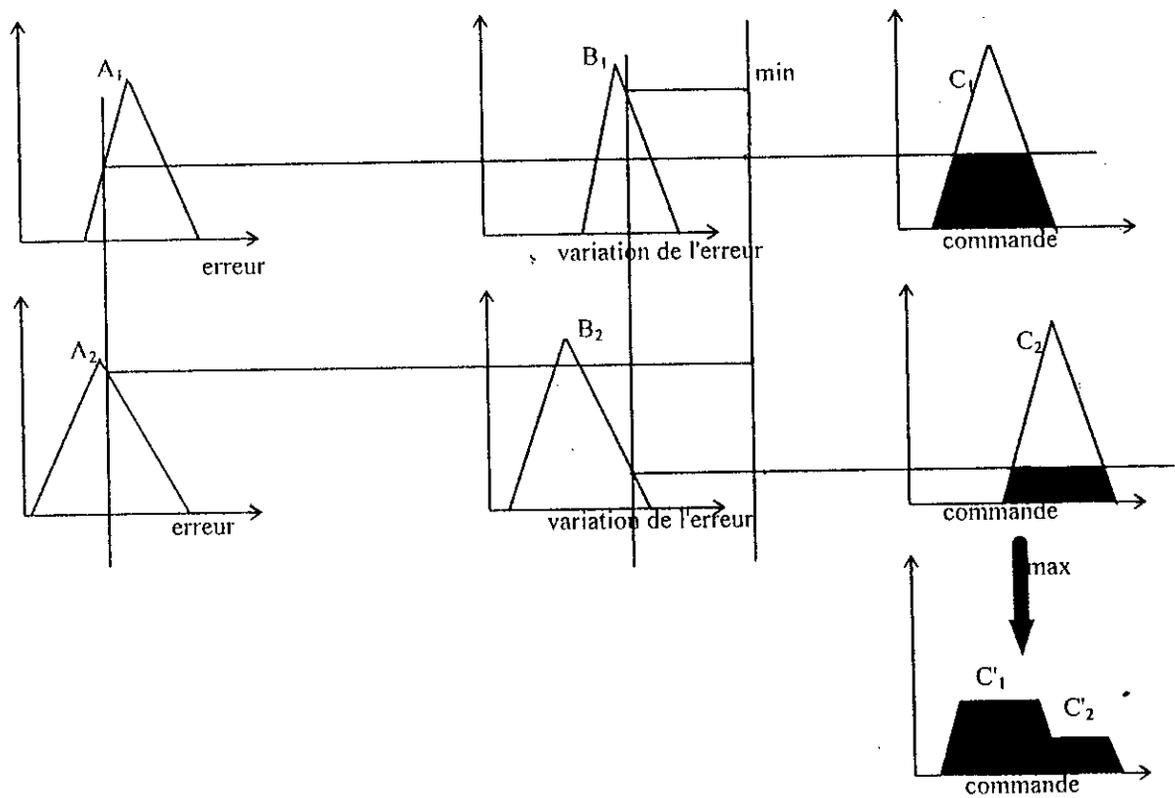
T-conorme : $x + y - x \cdot y$

ANNEXE III

REGULATEURS PAR LOGIQUE FLOUE

• Régulateur de Mamdani :

- R₁: Si x est A₁ et y est B₁ Alors z est C₁
 R₂: Si x est A₂ et y est B₂ Alors z est C₂

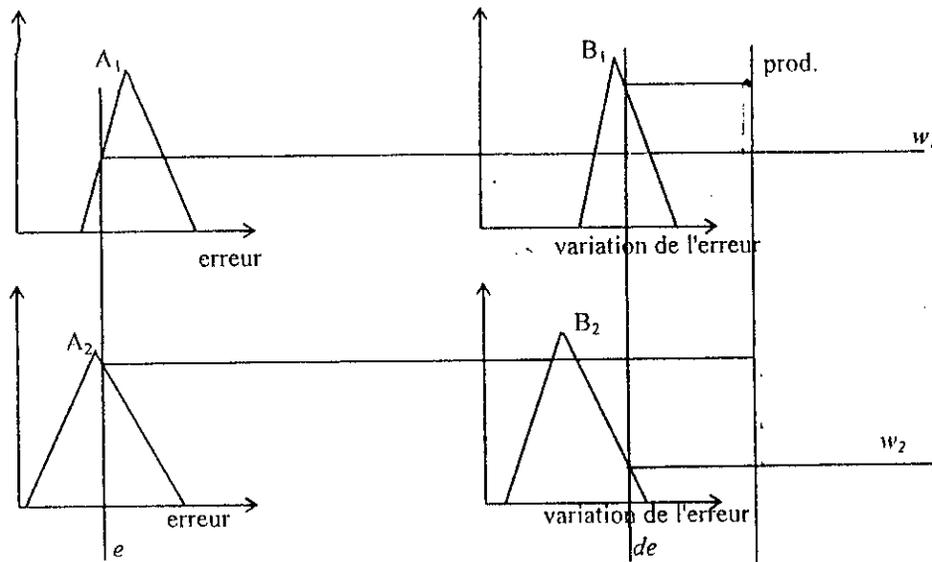


On calcule la commande par la méthode du centre de gravité.

• Régulateur de Sugeno :

R₁: Si e est A₁ et de est B₁ Alors f₁=p₁e + q₁de + r₁

R₂: Si e est A₂ et de est B₂ Alors f₂=p₂e + q₂de + r₂



$$\overline{w_1} = \frac{w_1}{w_1 + w_2}$$

$$\overline{w_2} = \frac{w_2}{w_1 + w_2}$$

$$f = \overline{w_1} \cdot f_1 + \overline{w_2} \cdot f_2$$

BIBLIOGRAPHIE

- [1] D. H. Hellendoorn et M. Reinfrank, *An Introduction to Fuzzy Control*. Springer Verlag, 1993.
- [2] B. Bouchon-Meunier, *Logique Floue*. PUF, 1993.
- [3] A. Kauffman, *Nouvelle Logique pour l'Intelligence Artificielle*. Hermès, 1987.
- [4] D. Dubois et H. Prade, *Théorie des Possibilités: Application à la Représentation des Connaissances en Informatique*. Masson, 2 ed., 1987.
- [5] J. Mendel, "Fuzzy logic system for engeneering. A tutorial," in *Proc. IEEE*, vol 83, no3, pp.345-377, Mars, 1995.
- [6] L. A. Zadeh, "Fuzzy logic," *IEEE Comput. Mag.*, pp. 83-93, 1988.
- [7] C. C. Lee, " Fuzzy logic in control systems : fuzzy logic çontroller-Part I," *IEEE Trans., syst., Man. and Cybern.*, vol. 20, pp. 404-418, Feb. 1990.
- [8] C. C. Lee, " Fuzzy logic in control systems : fuzzy logic controller-Part II," *IEEE Trans. syst., Man. and Cybern.*, vol. 20, pp. 419-435, Feb. 1990.
- [9] E. H. Mamdani, " Application of fuzzy algorithms for simple dynamic plant, " in *Proc. IEE* ,vol. 121, pp. 1585-1588, 1974.
- [10] B. Kosko, *Neural Network and Fuzzy Systems: A Dynamical Systems Approach*. Englewood Cliffs, NJ: Printice Hall, 1992.
- [11] J. -S. R. Jang, "Rule extraction using generalized neural networks," in *Proc. 4th IFSA World Congress*, July.1991, pp. 82-86 (volume for Artificial Intelligence).
- [12] J. -S. R. Jang, "Self-learning fuzzy controller based on temporal back-propagation, " *IEEE Trans. Neural Networks.*, vol. 3, pp. 714-723, Sept. 1992.
- [13] T. -S. Liu et J. C. Wu, "A model for rider-motorcycle system using fuzzy control," *IEEE Trans., Syst., Man. and Cybern.*, vol. 23, no. 1, Jan./feb. 1993.
- [14] A. Boscolo, C. Manglavacchi, F. Drius, et M. Golak, "Fuzzy controller for generally loaded DC electric motor," in *IFAC (Intelligent Component and Instruments for Control Applications)*, Malaga, Spain, 1992.
- [15] R. Hreero, J. Landaluze, C. F. Nicolas, et R. Reyero, "A highly non linear fuzzy control algorithm for servo-system positioning," in *IFAC (Intelligent Component and Instruments for Control Applications)*, Malaga, Spain, 1992.
- [16] J. -S. R. Jang et C. T. Sun, "Neuro-fuzzy modeling and control," in *Proc. IEEE*, vol. 83, no3, Mars, 1995, pp. 378-405.
- [17] D. P. Stoten, " Generalized manipulator dynamics, with regard to model reference adaptive control," *INT. J. Control*, vol.50, no.6, pp.2245-2268, 1989.
- [18] L. Guenfaf, *Etude de Différentes Stratégies de la Commande Adaptative*. Thèse de magister, ENP, 1995.
- [19] P. H. Coiffet, *Robot Habilis, Robot Sapien*. Hermès, 1993.
- [20] J. P. Lallemand, S. Zeghloul, *Robotique: Aspects Fondamentaux*. Masson, 1994.
- [21] H. Bühler, *Réglage par Logique Floue* . PPR., 1994.
- [22] H. Scholten, *Logique Floue et Régulation PID: Théorie et Pratique de la Régulation Active avec Interface à Réaliser Soi-même et Programme d'Expérimentation*. Publitronic, 1995.
- [23] R. Ketata, *Méthodologie de Régulation Numérique Incluant la Logique Floue*. Thèse de doctorat, no 199, INSA, Toulouse, 1992.
- [24] C. -T. Lin, G. Lee, "Neural-Network-Based Fuzzy logic control and decision system, " *IEEE Trans. Computers*, vol.40, no.12, Dec.1991.

- [25] E. M. Scharf, N. J. Mandic, "The application of fuzzy controller to the control of the multi-degree-of-freedom robot arm," in *Industrial Application of Fuzzy Control*. M. Sugeno, Ed. Amsterdam: North Holland, 1985, pp. 41-62.
- [26] S. Shao, "Fuzzy self-organizing controller and application for dynamic processes," *Fuzzy Sets. Syst.*, vol.26, pp.151-164, 1988.
- [27] C. C. Lee, "A self-learning rule-based controller employing approximate reasoning and neural net concepts," *INT. J. Intelligent Systems*, vol.6, pp.71-93, 1991.
- [28] A. E. Gegov et P. M. Frank, "Decentralized fuzzy control of multivariable systems by active decomposition of control laws," *INT. J. Control*. vol.62, no.4, pp.781-798, 1995.
- [29] T. Takagi et M. Sugeno, "Derivation of fuzzy control rules from human operator's control actions," in *IFAC Fuzzy Information*, Marseille, France, 1983.
- [30] C. Vibet, *Robotique: Principes et Contrôle*, Ellipses, 1987.
- [31] Y. Koren, *La Robotique: Pour les Ingénieurs*, McGraw-Hill, 1986.
- [32] P. J. Werbos, "Back-propagation through time: what it does and how to do it," in *Proc IEEE*, vol.78, no.10, Oct. 1990.
- [33] H. Ushida, T. Yamaguchi, K. Goto et T. Takagi, "Fuzzy-Neuro control using associative memories and its applications," in *Control Eng., Practice*, Vol.2, no.1, pp.129-145, 1994.
- [34] J. M. Randers, *Algorithmes Génétiques et Réseaux de Neurones*, Hermès, 1995.
- [35] S. Tzafestas, "Incremental Fuzzy PID Control," *IEEE Trans. Industrial Electronics*, vol. 37, no 5, Oct. 1990.

ملخص

هذا العمل يتناول في تطبيق التحكم بواسطة المنطق الغامض و العصبي الغامض على نظام معقد (روبوت). الطابع اللامركزي للتحكم ضروري و بدونه سيكون من المتعذر مراقبة النظام. في أول الأمر، تم تشكيل منظم بواسطة المنطق الغامض من صنف ممداني لسهولة، ثم اقترحت طريقة سهلة و فعالة للحصول على منظم بواسطة المنطق الغامض من صنف سوجينو (طريقة التقريب). هذا الأخير سيخضع لخوارزمية التعلم TBP التي تعيد توزيع دوال الإنتماء من أجل الحصول على نتائج جيدة من حيث الدقة و سرعة الحساب و البساطة.

كلمات مفتاحية: تحكم لامركزي، منطق غامض، عصبي غامض، قاعدة معارف، خوارزمية تعلم TBP.

Résumé :

Ce travail consiste en l'application de la commande par logique floue et neuro-floue à un système complexe (Robot). Le caractère décentralisé de la commande s'avère nécessaire et sans lequel le contrôle du système sera impossible. En premier temps, un régulateur par logique floue de type de Mamdani est synthétisé pour sa simplicité . Après, une méthode simple et efficace est proposée pour avoir un régulateur de type de Sugeno (méthode d'approximation). Ce dernier lui est appliqué l'algorithme d'apprentissage TBP qui ajuste la répartition des fonctions d'appartenance pour avoir de bonnes résultats en précision, en rapidité et en simplicité.

Mots clés :

commande décentralisée, logique floue, neuro-floue, base de connaissances, algorithme d'apprentissage TBP.

Abstract :

This work consists of the application of fuzzy and neuro-fuzzy control to a complex system (Robot). The control decentralization is necessary and without it the system control will be impossible. First, a fuzzy controller of Mamdani type is synthesized for its simplicity. After, a simple and efficient method is proposed to obtain a fuzzy controller of Sugeno type (approximation method). On the later, the learning algorithm TBP is applied to adjust the membership functions in order to obtain good results in precision, speed computing and simplicity.

Key words :

decentralized control, fuzzy logic, neuro-fuzzy, knowledge base, TBP learning algorithm