16/98

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

ECOLE NATIONALE POLYTECHNIQUE

DER Génie électrique et informatique
Filière Automatique

PROJET DE FIN D'ETUDES

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

Thème :

Utilisation de différents types de réseaux de neurones artificiels dans la commande adaptative décentralisée des robots manipulateurs

Proposé et dirigé par:

Mr D. BOUKHETALA
Mr F. BOUDJEMA

Etudié par :

Mr A. MAHROUCHE
Mlle K. GUEMGHAR

- Juin 1998 -

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

ECOLE NATIONALE POLYTECHNIQUE

DER Génie électrique et informatique
Filière Automatique

PROJET DE FIN D'ETUDES

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

Thème :

Utilisation de différents types de réseaux de neurones artificiels dans la commande adaptative décentralisée des robots manipulateurs

Proposé et dirigé par:

Mr D. BOUKHETALA
Mr F. BOUDJEMA

Etudié par :

Mr M. MAHROUCHE
Mlle K. GUEMGHAR

- Juin 1998 -

A mes parents adorés qu'ils reçoivent
ma profonde reconnaissance
et tout mon amour ,

A ma sœur Souad que j'affectionne
tout particulièrement ,

A l'ensemble de la chorale « le chœur
polytechnicien » , Samia , Amel ,
au « troupeau » et tout
mes amis ,

je dédie ce travail
G . Kahina

A mes chers parents

à tout mes frères et sœurs

à toute la famille Mahrouche

à mes aimables amis

Je dédie ce travail.

M.Malek

REMERCIEMENTS

Il nous est particulièrement agréable de remercier nos promoteurs Mr D. BOUKHETALA et Mr F. BOUDJEMA pour l'aide précieuse qu'ils nous ont apporté dans la réalisation de ce travail.

Notre sincère reconnaissance à messieurs les membres du jury qui nous ont fait l'honneur d'évaluer ce travail.

Que tous ceux qui ont veillé avec nous à l'élaboration de ce travail, trouvent dans l'aboutissement de celui-ci, l'expression de notre profonde gratitude, en particulier messieurs B. Omar, A. Arab et A. Kader.

Sommaire

INTRODUCTION GENERALE

CHAPITRE I. ***MODELISATION DES ROBOTS MANIPULATEURS***

I.1.Introduction 1
I.2.Modélisation cinématique 1
I.2.1.Paramètrisation de DENAVIT & HARTENBERG 1
I.2.2.Présentation du robot SCARA à 2 ddl 4
I.2.3.Présentation du robot PUMA 560 9
I.3.Modélisation dynamique 13
I.3.1.Introduction 13
I.3.2.Formalisme de Lagrange-Euler 14
I.3.3.Modèle dynamique du robot SCARA 19
I.3.4.Modèle dynamique du robot PUMA 560 22
I.4.Génération de trajectoire 26
I.5.Conclusion 28

CHAPITRE II. ***LES RESEAUX DE NEURONES ARTIFICIELS ET LA COMMANDE***

II.1. Introduction aux réseaux de neurones artificiels 29
II.1.1. Le neurone artificiel 29
II.1.2. Les réseaux de neurones artificiels 30
a- Les réseaux de neurones statiques 31
b- Les réseaux de fonctions à bases radiales 32
c- Les réseaux récurrents 34
d- Les réseaux dynamiques 37
II.2. Les algorithmes d'apprentissages 38
II.3. Les RNA et l'approximation des fonctions 42
II.4. Les réseaux de neurones et la commande 43
II.4.1. L'identification par réseaux de neurones 43
II.4.2. Les stratégies de commandes par réseaux de neurones 44
II.5.Conclusion 46

Introduction générale

Introduction générale

Les bras manipulateurs étant des systèmes fortement couplés et de dynamique non linéaire, leur commande nécessite des régulateurs à paramètres adaptatifs suivant des lois déterminées par les techniques de commande adoptées. Plusieurs travaux de recherches et publications sont consacrées à ce domaine cherchant les meilleures stratégies de commande assurant l'objectif principal qui est la conduite des bras manipulateurs à accomplir les tâches désirées, la robustesse par rapport aux perturbations ainsi qu'aux erreurs de modélisation et aux grandeurs non modélisables, la simplicité dans l'implémentation et la réalisabilité pratique.

Dans le but d'alléger la complexité des algorithmes de commande, on opte généralement pour des lois de commande décentralisées, en plaçant un contrôleur local au niveau de chaque articulation pour générer la commande appropriée en fonction seulement des données disponibles localement.

La commande décentralisée se répand de plus en plus grâce aux avantages qu'elle présente en commande comme la rapidité (nécessite moins de calcul), simplicité, facilité d'implémentation et en pratique, elle assure aussi la sécurité du bras manipulateur en cas de défaillance dans l'un des régulateurs.

L'exploitation des propriétés des réseaux de neurones à savoir l'approximation des fonctions non linéaires, la généralisation ainsi que la souplesse et la simplicité des algorithmes d'adaptation des paramètres internes des réseaux ouvre des perspectives prometteuses dans le domaine de l'identification et la commande des systèmes complexes.

Dans ce travail, nous allons étudier l'apport de différentes structures de réseaux de neurones dans la commande adaptative décentralisée de deux bras manipulateurs PUMA 560 et SCARA. Il est organisé en trois chapitres.

Le premier chapitre va porter sur la modélisation des bras manipulateurs. Les modèles mathématiques des robots PUMA 560 et SCARA seront établis, à base desquels, des simulations en boucle ouverte seront faites.

Dans le second chapitre, nous présenterons les principes de base des réseaux de neurones artificiels, les différentes architectures que nous allons utiliser par la suite dans la commande à savoir les réseaux statiques, les réseaux récurrents «with self feedback », les réseaux récurrents

«feedforward NN », les réseaux de fonctions à base radiale et les réseaux dynamiques, ainsi que les algorithmes d'apprentissage correspondants à chaque architecture.

Le troisième chapitre sera consacré à la présentation de la technique de commande appliquée pour la conduite des bras manipulateurs, les algorithmes d'adaptation des paramètres des différentes architectures des réseaux de neurones considérés, assurant la stabilité du système global, en se basant sur la théorie de stabilité de Lyapaunov. A partir des résultats de simulation appliquée sur les deux modèles de bras de robots modélisés, une étude comparative des performances des différentes architectures des réseaux utilisés sera présentée.

Un rappel sur la théorie de l'échantillonnage et de la reconstruction fera l'objet d'une annexe et qui sera présentée à la fin de ce mémoire.

Chapitre I

Modélisation des robots manipulateurs

Chapitre I

MODELISATION DES ROBOTS MANIPULATEURS

I .1. Introduction :

Pour le sens commun, un robot est un dispositif mécanique articulé, capable d'imiter certaines fonctions humaines telles que la manipulation d'objets ou la locomotion, dans le but de se substituer à l'homme pour la réalisation de certaines tâches matérielles. Pour cela, un tel dispositif comporte une partie ''opérationnelle'' ou ''opérative'' qui réalise la tache et une partie ''décisionnelle'' ou ''commande'' qui contrôle la partie opérationnelle.

L'appellation « robot » a pour origine le mot grec « robota » qui signifie travail. La première vraie définition de ce qu'est un robot a été donnée par les Japonais vers 1978 : « un robot est une machine polyvalente (effectuant des tâches différentes), capable d'agir sur son environnement (manipulateur) et adaptable à un environnement changeant (programmable) ».
On traite dans ce travail deux modèles types de robots :

- Robot PUMA560 à 3 degrés de liberté ;
- Robot SCARA à 2 degrés de liberté.

I .2. Modélisation cinématique:

I .2.1. Paramètrisation de DENAVIT & HARTENBERG :

Un manipulateur série consiste en une chaîne de liaisons connectées par des articulations (charnières ou glissières).
Un manipulateur à n degrés de liberté compte n liaisons et n articulations. Les liaisons permettent de maintenir une relation fixe entre les articulations du manipulateur à chaque extrémité de la liaison.
La base du manipulateur est la liaison 0 elle ne fait pas partie des n liaisons ; la liaison 1 est reliée à la liaison base par l'articulation 1 et il n'y a pas d'articulation au bout de la dernière liaison.
Chaque liaison est caractérisée par son propre repère. Utilisant les transformations homogènes, on peut trouver les matrices de passage d'un repère à un autre.

DENAVIT et HARTENBERG [Paul 83] ont établi une transformation basée sur un paramétrage particulier, figure (I.1) On résume cette paramétrisation dans les étapes suivantes.

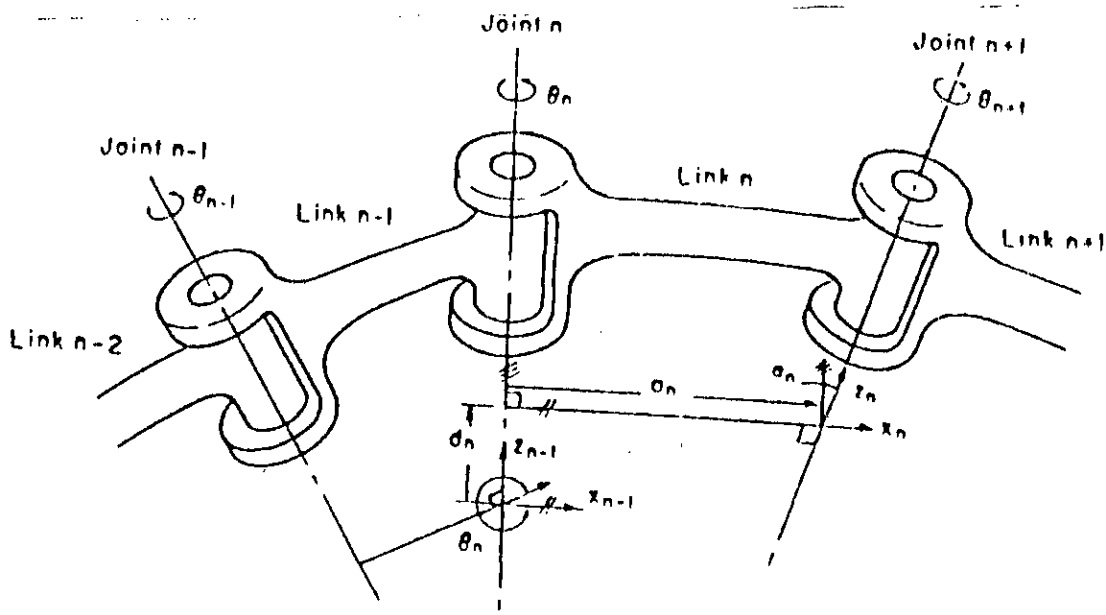

Figure I .1. ***Présentation des référentiels de DENAVIT - HARTEMBERG***

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

ETAPE 2 : Etablir le repère Ri de chaque liaison i (articulation) *Ri(Oi,Xi,Yi,Zi)*, tel que :

- L'axe Z_i est choisi le long de l'axe de l'articulation $i+1$;
- L'axe X_i peut être n'importe quel axe normal à Z_{i-1}, son sens le long de cette normale va de l'articulation i vers l'articulation $i+1$.

Dans le cas où Z_{i-1} et Z_i ne sont pas parallèles, on choisit l'axe Xi parallèle ou antiparallèle au vecteur $Z_{i-1} \wedge Z_i$. Ce choix implique que l'axe X_i est dirigé suivant la normale commune entre Z_{i-1} (articulation i) et Z_i (articulation $i+1$).

L'axe Y_i est choisi de façon à former un trièdre droit.

ETAPE 3 :On définit deux groupes de paramètres :

- deux paramètres de localisation de l'axe Z_i dans la liaison i.

a_i = distance entre Z_{i-1}etZ_i, le long de X_i.

α_i =angle (Z_{i-1}, Z_i).

- deux paramètres de mouvement (rotation et /ou translation) de la liaison.

d_i = distance (X_{i-1} , X_i) mesurée suivant Z_{i-1}.

= coordonnées de O_i dans R_{i-1} le long de Z_{i-1}.

θ_i= angle(X_{i-1} , X_i) obtenu par rotation de X_{i-1} vers X_i autour de Z_{i-1}.

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

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

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

T_{i-1}^{i}= rot(Z_{i-1},θ_i) . trans(Z_{i-1},d_i) . trans(X_i,a_i) . rot(X_i,α_i) .

où : rot ≡ rotation ;

trans ≡ translation .

Ce qui donne :

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

où :

$$d_{i-1}^{i} = [a_i \cos\theta_i \quad a_i \sin\theta_i \quad d_i]^T \tag{I.2}$$

est le vecteur droit pour la translation ;

et

$$C_{i-1}^{i} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i.\cos\alpha_i & \sin\theta_i.\cos\alpha_i \\ \sin\theta_i & \cos\theta_i.\cos\alpha_i & -\cos\theta_i.\sin\alpha_i \\ 0 & \sin\alpha_i & \cos\alpha_i \end{bmatrix} \tag{I.3}$$

est la matrice rotation.

D'où :

$$T_{i-1}^{i} = \begin{bmatrix} \cos\theta_i & -\sin\theta_i.\cos\alpha_i & \sin\theta_i.\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i.\cos\alpha_i & -\cos\theta_i.\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{I.4}$$

I .2.2. Présentation du robot SCARA à 2 ddl : [Bali 95] [Madani 97]

Le robot SCARA à deux degrés de liberté est caractérisé par deux articulations rotationnelles θ_1, θ_2 évoluant sur un espace horizontal (figure I.2).

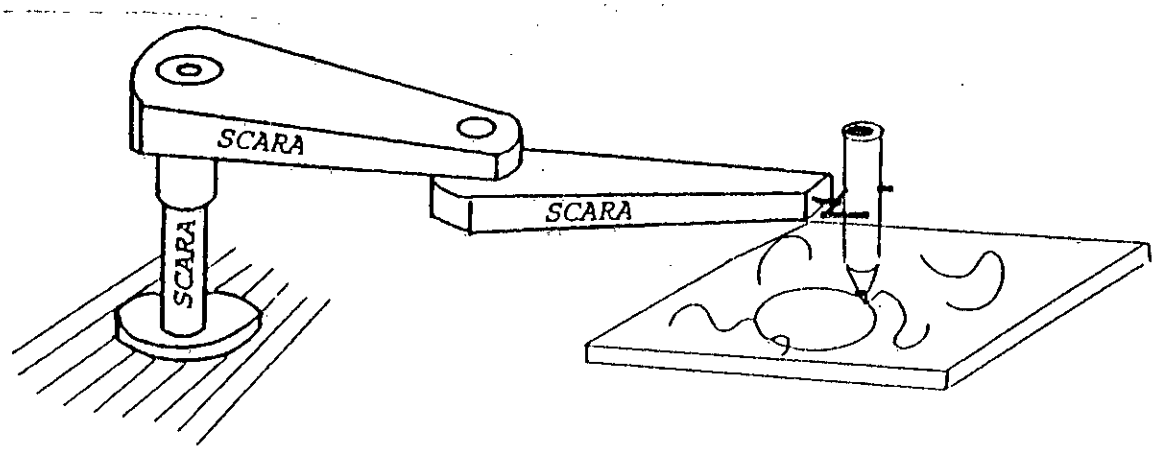

Figure I .2. ***présentation du robot SCARA***

L'application de la paramètrisation de DENAVIT et HARTENBERG (D&H) dans le cas du robot SCARA à deux degrés de liberté (fig.I.3) donne lieu aux paramètres : (tableau I.1).

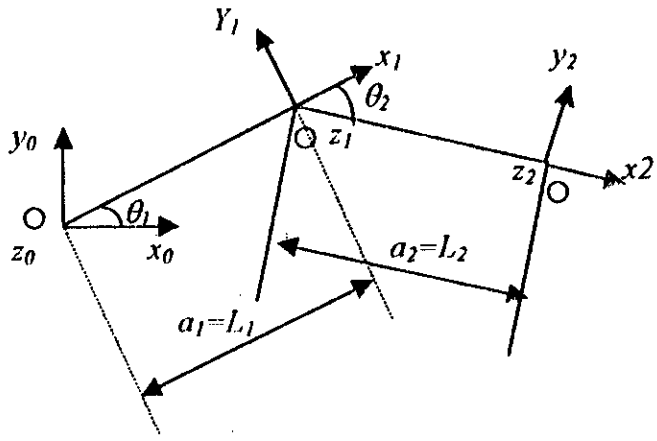

Figure I .3. ***paramétrisation de D-H pour le robot SCARA à deux degrés de liberté***

N° de la liaison	Variables	α_i	a_i	d_i	θ_i
1	q_1	0	L_1	0	θ_1
2	q_2	0	L2	0	θ_2

Tableau1.1. ***paramètrisation de D&H du robot SCARA .***

Le vecteur des coordonnées généralisées :

$$q = [\theta_1 \quad \theta_2]^T \tag{I.5}$$

Les matrices de transformations associées à ce modèle :

$$T_0^1 = \begin{bmatrix} C_1 & -S_1 & 0 & L_1C_1 \\ S_1 & C_1 & 0 & L_1S_1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad , \quad T_1^2 = \begin{bmatrix} C_2 & -S_2 & 0 & L_2C_2 \\ S_2 & C_2 & 0 & L_2S_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{I.6}$$

D'où la matrice de transfert du repère R_0 au repère R_2 est :

$$T_0^2 = T_0^1 . T_1^2 = \begin{bmatrix} C_{12} & -S_{12} & 0 & L_1C_1 + L_2C_{12} \\ S_{12} & C_{12} & 0 & L_1S_1 + L_2S_{12} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{I.7}$$

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

a . Modélisation géométrique :

- ***Modèle géométrique direct :(MGD)***

Les transformations précédentes permettent d'exprimer la position de l'organe terminal r par rapport au repère R_0 : $r(R_0)$ à partir des variables articulaires q_i .

$$r(R_0) = F(q) \tag{I.8}$$

$F(q)$ est une fonction vectorielle, avec :

$$\begin{cases} q^T = [q_1 \quad q_2] = [\theta_1 \quad \theta_2] \\ r^T = [x \quad y \quad z \quad 1] \\ r(R_0) = F(q) = T_0^2 . r_2^2 \end{cases} \tag{I.9}$$

où r_2^2 : coordonnées de l'élément terminal dans R_2. Pour le robot SCARA :

$$r_2^2 = [0 \quad 0 \quad 0 \quad 1]^T \tag{I.10}$$

De l'équation (I.9) on trouve :

$$\begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} L_1C_1 + L_2C_{12} \\ L_1S_1 + L_2S_{12} \\ 0 \\ 1 \end{bmatrix} \tag{I.11}$$

L'équation (I.11) représente le modèle géomètrique direct du robot SCARA. Les transformations de DENAVIT&HARTENBERG ne sont pas indispensables pour tirer le modèle géométrique d'un robot à faibles degrés de liberté. En effet, de la figure (I.4), nous pouvons aisément trouver le modèle géométrique direct du robot SCARA à deux degrés de liberté.

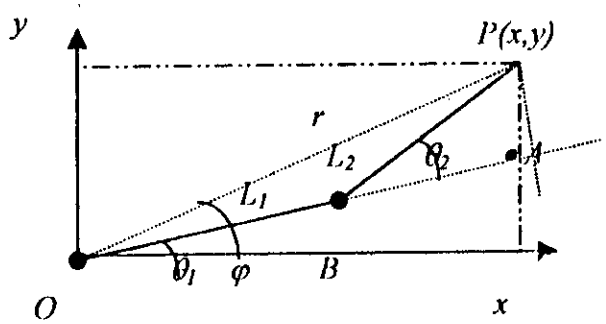

Figure I .4. ***Evolution du bras de robot SCARA***

$$\begin{cases} x = L_1C_1 + L_2C_{12} \\ y = L_1S_1 + L_2S_{12} \\ z = 0 \end{cases} \tag{I.12}$$

Le système d'équations (I.12) est le même que (I.11).

- ***Modèle géométrique inverse : (MGI)***

A partir de l'équation (I.12), on peut tirer les coordonnées généralisées q_i en fonction des coordonnées opérationnelles par rapport au repère R_0, ce qui constitue le modèle géométrique inverse.

Remarquons qu'il est difficile de le tirer analytiquement. On déterminera alors θ_1, θ_2 géométriquement en fonction des coordonnées cartésiennes x, y ou polaires r, φ , avec

$$\begin{cases} r^2 = x^2 + y^2 \\ \varphi = Atg(y/x) \end{cases} \tag{I.13}$$

De la figure(I.4), du triangle OAP, on tire :

$$r^2=(L_1+L_2C_2)^2 +(L_2S_2)^2 \tag{I.14}$$

Pour :

$$\begin{cases} L_1 = L_2 = L \\ r < 2L \end{cases} \tag{I.15}$$

ce qui est toujours vérifié :

$$\theta_2 = Arc\cos\left[\frac{1}{2}\left(\frac{r}{L}\right)^2 - 1\right] \tag{I.16}$$

On a aussi du triangle OAP :

$$Sin(\varphi-\theta_1)=\frac{L_2 \sin\theta}{r} \tag{I.17}$$

Pour

$$\begin{cases} L_1 = L_2 = L \\ et \\ |L_2 \sin\theta_2| < r \end{cases} \tag{I.18}$$

ce qui est toujours vérifié :

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

$$\theta_1 = \varphi - Arc\sin\left[\frac{L}{r}\sin\theta_2\right] \tag{I.19}$$

Donc

$$\begin{cases} \theta_2 = Arc\cos\left[\frac{1}{2}\left(\frac{r}{L}\right)^2 - 1\right] \\ \theta_1 = \varphi - Arc\sin\left[\frac{L}{r}\sin\theta_2\right] \end{cases} \quad (I.20)$$

r et φ définis dans l'équation (I.13).

b . Modèle cinématique :

La cinématique complète la modélisation géométrique en établissant les relations entre les vitesses des paramètres articulaires $q^T = [\theta_1 \ \ \theta_2]$ et celles des paramètres opérationnels selon qu'on travaille en coordonnées cartésiennes $X = [x \ y]^T$ ou en coordonnées polaires (r, φ).

- ***Modèle cinématique direct :***

Il s'agit de déterminer $x = f(q)$.
En dérivant l'équation (I.12) , on obtient :

$$\dot{X} = \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = \begin{pmatrix} -L(S_1 + S_{12}) & -L \cdot S_{12} \\ L(C_1 + C_{12}) & L \cdot C_{12} \end{pmatrix} \cdot \begin{pmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \end{pmatrix} \quad (I.21)$$

L'équation (I.21) constitue le modèle cinématique direct du robot SCARA .

- ***Modèle cinématique inverse :***

Il s'agit maintenant d'établir la relation $\dot{q} = f(\dot{x})$
L'équation (I.20) est équivalente à :

$$\begin{cases} \theta_2 = Arc\cos\left[\frac{1}{2}\left(\frac{r}{L}\right)^2 - 1\right] \\ \theta_1 = \varphi - \varphi_1 \\ \varphi = Arctg\left(\frac{y}{x}\right) \\ \varphi_1 = Arc\sin\alpha \\ \alpha = \frac{L}{r}\sin\theta_2 \end{cases} \quad (I.22)$$

En dérivant l'équation (I.22), on obtient le modèle cinématique inverse :

$$\begin{cases} \dot{\theta}2 = \dfrac{-\dot{r}}{L\sqrt{1-\dfrac{1}{4}\left(\dfrac{r}{L}\right)^2}} \\ \dot{\theta}_1 = \dot{\varphi} - \dot{\varphi}_1 \\ \dot{\varphi} = \dfrac{\dot{y}.x - \dot{x}.y}{r^2} \\ \dot{\varphi}_1 = \dfrac{\dot{\alpha}}{\sqrt{1-\alpha^2}} \\ \dot{\alpha} = \dfrac{L}{r}\theta_2 \cos\theta_2 - \dfrac{\dot{r}}{r^2} L \cdot \sin\theta_2 \end{cases} \tag{I.23}$$

I .2.3. Présentation du robot PUMA 560 : [Madani 97] [Armstrong 86]

La figure(I.5) représente le robot PUMA 560 qui présente six articulations. On se limitera dans ce travail aux trois premières articulations qui sont rotationnelles, la première évoluant sur un plan horizontal, et les deux autres évoluants sur des plans verticaux. On supposera donc les trois autres articulations fixes.

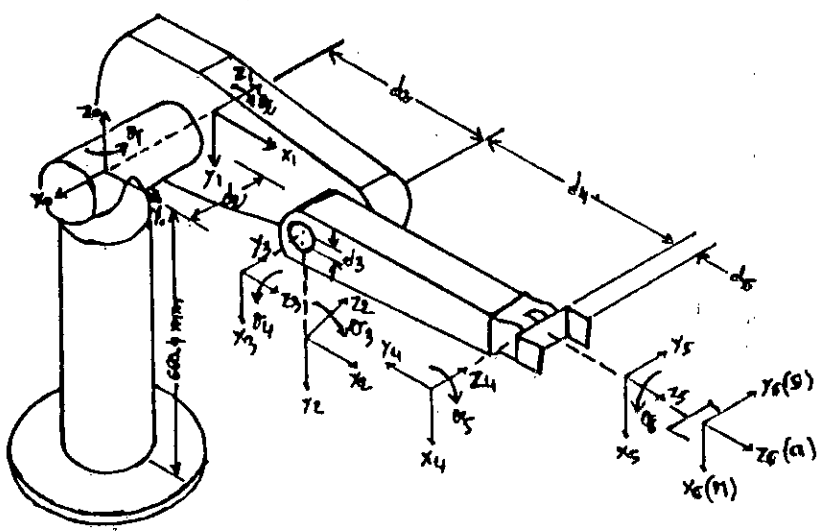

Figure I. 5. ***Présentation du Bras de robot PUMA 560***

Le bras de robot étudié est donc à trois degrés de liberté.
L'application de la paramétrisation de DENAVIT et HARTENBERG sur le robot PUMA 560, (figure I.5) donne lieu aux paramètres suivants (tableau I.2) .

N° de la liaison	Variable	θ_i	a_i	d_i	α_i
1	q_1	θ_1	0	0	-90°
2	q_2	θ_2	L_2	d_2	0
3	q_3	θ_3	L3	0	0

Tableau I .2. ***paramétrisation de D&H du robot PUMA 560***

Le repère R_3 est sur le centre de masse de l'effecteur (fin de la liaison 3) .
Le vecteur des coordonnées généralisées est :

$$q = [\theta_1 \quad \theta_2 \quad \theta_3]^T \tag{I.24}$$

Les matrices de transformations associées à ce robot sont alors :

$$T_0^1 = \begin{bmatrix} C_1 & 0 & -S_1 & 0 \\ S_1 & 0 & C_1 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad T_1^2 = \begin{bmatrix} C_2 & -S_2 & 0 & L_2C_2 \\ S_2 & C_2 & 0 & L_2S_2 \\ 0 & 0 & 1 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

et

$$T_2^3 = \begin{bmatrix} C_3 & -S_3 & 0 & L_3C_3 \\ S_3 & C_3 & 0 & L_3C_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{I.25}$$

On aura alors :

$$T_0^3 = T_0^1 \cdot T_1^2 \cdot T_2^3 = \begin{bmatrix} C_1 \cdot C_{23} & -C_1 \cdot S_{23} & -S_1 & C_1(L_2C_2 + L_3C_{23}) - d_2S_1 \\ S_1 \cdot C_{23} & -S_1 \cdot S_{23} & C_1 & S_1(L_2C_2 + L_3C_{23}) + d_2C_1 \\ -S_{23} & -C_{23} & 0 & -(L_2S_2 + L_3S_{23}) \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{I.26}$$

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

- **Modélisation géométrique :**

- ***Modèle géométrique direct :***

$$r(R_0) = F(q) = T_0^3 r(R_3) \tag{I.27}$$

avec :

$$\begin{cases} r(R_3) = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T \\ r(R_0) = \begin{bmatrix} x & y & z & 1 \end{bmatrix}^T \end{cases} \tag{I.28}$$

Des équations (I.26) , (I.27) et (I.28) , on obtient :

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} C_1(L_2C_2 + L_3C_{23}) - d_2S_1 \\ S_1(L_2C_2 + L_3C_{23}) + d_2C_1 \\ -(L_2S_2 + L_3S_{23}) \end{bmatrix} \tag{I.29}$$

L'équation (I.29) constitue le modèle géométrique inverse du robot PUMA 560 .

- ***Modèle géométrique inverse : (MGI)***

Il s'agit maintenant d'établir la relation $q=F(r)$.
Il existe plusieurs solutions pour l'établir :

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

Comme pour le robot SCARA , on procédera par approche géométrique pour l'établissement du modèle géométrique inverse du robot PUMA 560 .
On présente d'abord l'approche géométrique pour le calcul du MGI du robot PUMA 560(à 6 degrés de liberté) .
En se basant sur la géométrie du bras humain , on peut définir les indicateurs de configurations : arm(bras), elbow(coude) associés aux trois premières articulations et wrist(poignet) associé quant à lui aux trois dernières articulations du robot PUMA 560 .
Il existe pour le robot PUMA quatre solutions différentes pour les trois premières articulations, et à chacune d'elles correspondent deux solutions possibles et différentes pour les trois dernières .
Les deux premiers indicateurs servent à choisir une seule solution pour les trois premières articulations et l'indicateur wrist sert à choisir une solution pour les trois dernières.

- Définition des indicateurs de configuration :

Les indicateurs de configuration sont préspécifiés par l'utilisateur en vue de trouver la solution inverse. Dans notre cas, on les définira comme suit : (se reporter à la figure I.5)

Right arm : θ_2 positif fait mouvoir le poignet du manipulateur dans le sens des Z_0 positifs, alors que la troisième articulation est inactive.

Left arm : θ_2 positif fait mouvoir le poignet du manipulateur dans le sens des Z_0 négatifs, alors que la troisième articulation est inactive.

Above arm : la position du poignet dans la configuration right arm(respectivement left arm) par rapport au référentiel de l'épaule a une coordonnée négative(respectivement positive) le long de l'axe y_2 .

Below arm : la position du poignet dans la configuration right arm(respectivement left arm) par rapport au référentiel de l'épaule a une coordonnée positive(respectivement négative le long de l'axe y_2 .

Wrist down : le vecteur S (figure I.6) et le vecteur y_5 ont un produit scalaire positif .

Wrist up :le vecteur S et le vecteur y_5 ont un produit scalaire négatif .

Arm = +1 pour une configuration **right arm** ;
-1 pour une configuration **left arm** ;
elbow = +1 pour une configuration **above arm** ;
-1 pour une configuration **below arm** ;
wrist = +1 pour une configuration **wrist down** ;
-1 pour une configuration **wrist up** .

Avec ces paramètres et pour les trois premières articulations, on retrouve effectivement le même MGD déjà établi à l'équation (I.29) [Nedjari 96].

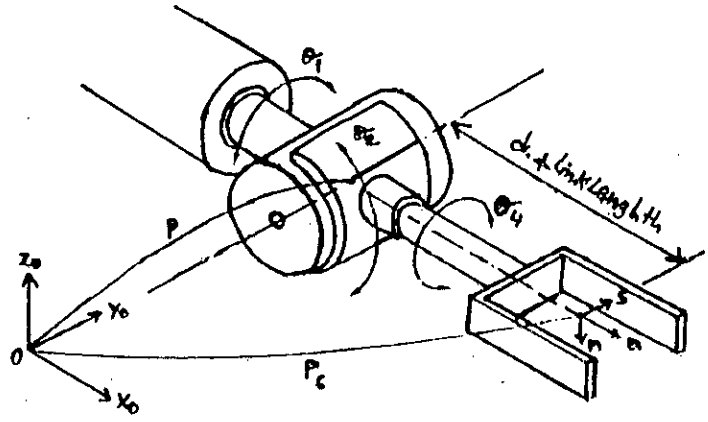

Figure I.6. ***indicateurs de configuration.***

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

$$\theta_1 = tg^{-1}\left\{\frac{-arm.y.\sqrt{x^2+y^2-d_2{}^2}-x.d_2}{-arm.x.\sqrt{x^2+y^2-d_2{}^2}-y.d_2}\right\} \qquad -\pi \le \theta_1 \le \pi$$

$$\theta_2 = tg^{-1}\left\{\frac{\sin\alpha.\cos\beta + arm.elbow.\cos\alpha}{\cos\alpha.\sin\beta - arm.elbow.\sin\alpha.\cos\beta}\right\} \qquad -\pi \le \theta_2 \le \pi \qquad (1.26)$$

$$\theta_3 = \phi - \frac{\pi}{2} \qquad -\pi \le \theta_3 \le \pi$$

avec :

$$\cos\alpha = \frac{-arm.\sqrt{x^2+y^2-d_2{}^2}}{\sqrt{x^2+y^2+z^2-d_2{}^2}} \qquad \sin\alpha = \frac{-z}{\sqrt{x^2+y^2+z^2-d_2{}^2}}$$

$$\sin\beta = \frac{x^2+y^2+z^2-d_2{}^2+L_2{}^2-L_3{}^2}{2a_2.\sqrt{x^2+y^2+z^2-d_2{}^2}} \qquad (1.27)$$

$$\cos\phi = \frac{a_2{}^2+L_3{}^2-x^2-y^2-z^2+d_2{}^2}{2L_2L_3} \qquad \sin\phi = arm.elbow.\sqrt{1-(\cos\phi)^2}$$

I .3. Modélisation dynamique :

I .3.1. Introduction :

On s'intéresse maintenant aux efforts actionneurs produits par les mouvements du manipulateur .Il s'agit d'établir les équations différentielles qui relient les efforts actionneurs $T_i(t)$ aux positions, vitesses et accélérations articulaires($(q_i(t), \dot{q}_i(t), \ddot{q}(t)$ respectivement) .

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

Deux exploitations du modèle sont possibles :

- Résoudre les équations différentielles pour obtenir les mouvements $q_i(t)$ pour des $T_i(t)$ donnés dans le but d'une simulation du comportement dynamique. On parle alors de modèle dynamique(MDD) .

- Calculer les efforts $T_i(t)$ pour des mouvements $q_i(t)$ connus, donnés à l'avance ou mesurés in situ . Cette approche constitue le modèle dynamique inverse(MDI).

Pour obtenir le modèle dynamique, on peut utiliser :

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

I .3.2. Formalisme de Lagrange-Euler : [Madani 97] [Lallemand 94] [Paul 83]

Les équations de Lagrange-Euler sont :

$$\frac{d}{dt}\frac{\partial T}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} + \frac{\partial E_D}{\partial \dot{q}_i} = T_i \qquad i=1,\dots,n \tag{I.32}$$

L étant le lagrangien, défini par :

$$L=E_c-E_p \tag{I.33}$$

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

- **Energie cinétique :**

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

$$V_0^i = \frac{dr_0^i}{dt} \tag{I.34}$$

avec :

$$r_0^i = T_0^i r_i^i \tag{I.35}$$

où r_i^i est la $i^{ème}$ coordonnée homogène exprimée dans le repère R_i . Comme les liaisons sont toutes rigides, on a :

$$\frac{dr_i^i}{dt} = 0 \tag{I.36}$$

donc :

$$V_0^i = \sum_{j=1}^{i} \frac{\partial T_0^i}{\partial q_j} \frac{dq_j}{dt} r_i^i \tag{I.37}$$

ou encore :

$$V_0^i = \sum_{j=1}^{i} U_{ij}\, \dot{q}_j\, r_i^i \tag{I.38}$$

où :

$$U_{ij} = \begin{cases} 0 & j > i \\ T_0^{j-1} Q_j T_{j-1}^i & j \leq i \end{cases} \tag{I.39}$$

l'équation (I.38) n'est qu'une forme compacte de l'équation (I.37), avec :

$$Q_j = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{I.40}$$

si la liaison j est rotationnelle, et :

$$Q_j = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{I.41}$$

si la liaison j est translationnelle.

L'énergie cinétique de l'élément i (dans la liaison i) sera alors :

$$dE_{ci} = \frac{1}{2} trace\left(V_i V_i^T\right) dm \tag{I.42}$$

En remplaçant V_i de l'équation (I.38) dans l'équation (I.42), on trouve :

$$dE_{ci} = \frac{1}{2} trace\left(\sum_{j=1}^{i} \sum_{k=1}^{i} U_{ij} \left(r_i^i r_i^{i\,T} dm \right) U_{ik}^T\, \dot{q}_j\, \dot{q}_k \right) \tag{I.43}$$

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

$$E_{cl} = \frac{1}{2} trace\left(\sum_{j=1}^{i} \sum_{k=1}^{i} U_{ij} J_i U_{ik}^T \dot{q}_j \dot{q}_k \right) \tag{I.44}$$

Avec :

$$J_i = \begin{bmatrix} \int x_i^2 dm & \int x_i y_i dm & \int x_i z_i dm & \int x_i dm \\ \int x_i y_i dm & \int y_i^2 dm & \int y_i z_i dm & \int y_i dm \\ \int x_i z_i dm & \int y_i z_i dm & \int z_i^2 dm & \int z_i dm \\ \int x_i dm & \int y_i dm & \int z_i dm & \int dm \end{bmatrix} \tag{I.45}$$

l'énergie cinétique des actionneurs est définie par :

$$E_{ca} = \frac{1}{2} \sum_{i=1}^{n} I_i \dot{q}_i^2 \tag{I.46}$$

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

L'énergie cinétique totale est alors :

$$E_c = E_{ca} + \sum_{i=1}^{n} E_{ci} \tag{I.47}$$

- **Energie potentielle :**

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

$$E_p = \sum_{i=1}^{n} - m_i g^T r_0^i \tag{I.48}$$

De l'équation (I.35) on déduit :

$$E_p = -\sum_{i=1}^{n} m_i g^T T_0^i r_i^i \tag{I.49}$$

avec :

$$g^T = \begin{bmatrix} 0 & 0 & |g| & 1 \end{bmatrix} \tag{I.50}$$

où g est la gravité.

- **Energie de dissipation :**

Elle est donnée par la relation suivante :

$$E_D = \frac{1}{2}\sum_{i=1}^{n} f_{vi}\,\dot{q}_i^{\,2} \tag{I.51}$$

où f_{vi} est le coefficient de frottement visqueux dû à la liaison i.
Des équations (I.47),(I.49),(I.51) on déduit l'expression générale du Lagrangien.

$$L = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{i}\sum_{k=1}^{i} trace\left(U_{ij} J_i U_{ik}^{\,T}\right)\dot{q}_j\,\dot{q}_k + \sum_{i=1}^{n} m_i g^T T_0^i r_i^i \tag{I.52}$$

Où J_i, U et g^T sont définis par les équations (I.45),(I.39) et (I.50) respectivement.
En remplaçant la valeur de L trouvée(I.52) dans l'équation de Lagrange (I.32), on trouvera pour la liaison i :

$$T_i = \sum_{j=1}^{n}\sum_{k=1}^{j} trace\left(U_{jk} J_j U_{ji}^{\,T}\right)\ddot{q}_k + \sum_{j=1}^{n}\sum_{k=1}^{j}\sum_{l=1}^{j} trace\left(U_{jkl} J_j U_{ji}^{\,T}\right)\dot{q}_k\,\dot{q}_l + \mathrm{f}_{vi}\,\dot{q}_i - \sum_{j=1}^{n} m_i g^T U_{ji} r_j^j \tag{I.53}$$

avec :

$$U_{ij} = \begin{cases} T_0^k Q_k T_{k-1}^{j-1} Q_j T_{j-1}^i & k \le j \le i \\ T_0^{j-1} Q_j T_{j-1}^{k-1} Q_k T_{k-1}^i & j \le k \le i \\ 0 & j < i < k \end{cases} \tag{I.54}$$

En posant :

$$M_{ij}(q) = \sum_{k=\max(i,j)}^{n} trace\left(U_{kj} J_k U_{ki}^{\,T}\right), \qquad \begin{matrix} i = 1,n \\ j = 1,n \end{matrix} \tag{I.55}$$

$$G_i(q) = -\sum_{j=i}^{n} m_j g^T U_{ji} r_j^j \tag{I.56}$$

$$N_{ijk}(q) = \sum_{l=\max(i,j,k)}^{n} trace\left(u_{ijk} J_l U_{li}^{T}\right), \qquad \begin{matrix} i=1,n \\ j=1,n \\ k=1,n \end{matrix} \tag{I.57}$$

$$H_i(\dot{q}_i) = f_{vi}\,\dot{q}_i \tag{I.58}$$

L'équation (I.53) devient alors :

$$T_i = \sum_{j=1}^{n} M_{ij}(q)\ddot{q}_j + \sum_{j=1}^{n}\sum_{k=1}^{n} N_{ijk}(q)\dot{q}_j\dot{q}_k + G_i(q) + H_i(\dot{q}_i) \tag{I.59}$$

En réécrivant l'équation (I.59) sous forme matricielle on obtient :

$$T = M(q)\ddot{q} + N(q,\dot{q}) + G(q) + H(\dot{q}) \tag{I.60}$$

avec

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

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

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

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

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

T : vecteur de forces et/ou couples moteurs.

La détermination de $N(q,\dot{q})$ de l'équation (I.57) peut s'avérer longue .Une méthode rapide pour le calcul de $N(q,\dot{q})$ revient à calculer les matrices centrifuges et Coriolis indépendamment, et qui seront multipliés ensuite par leurs vecteurs.

La somme des vecteurs obtenus donne alors le vecteur $N(q,\dot{q})$.

On obtient ces deux matrices par dérivation de la matrice d'inertie (principe de la conservation de l'énergie).

L'équation (I.60) devient alors :

$$T = M(q)\ddot{q} + D(q)[\dot{q}\dot{q}] + C(q)[\dot{q}^2] + G(q) + H(\dot{q}) \tag{I.61}$$

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

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

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

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

Avec $D_{ij} = 2\beta^{ikl}$

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

β étant le symbole de Christoffel [Madani 97],[Armstrong 86] défini par :

$$\beta^{i,jk} = \frac{1}{2}\left[\frac{\partial M_{ij}}{\partial q_k} + \frac{\partial M_{ik}}{\partial q_j} - \frac{\partial M_{jk}}{\partial q_i}\right] \tag{I.62}$$

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

$$\begin{cases} \dfrac{\partial M_{ij}}{\partial q_k} = \dfrac{\partial M_{ji}}{\partial q_k} & \forall i,j,k \\ \dfrac{\partial M_{ij}}{\partial q_k} = 0 & pour\ i \geq k\ ,\ j \geq k \end{cases} \tag{I.63}$$

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

$$H(q) = f_{vi}\,\dot{q}_i + f_{si} sgn(\dot{q}) \quad i = 1,n \tag{I.64}$$

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

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

I .3.3. Modèle dynamique du robot SCARA :

Avant d'établir le modèle dynamique du robot SCARA, plusieurs hypothèses sont à prendre en considération : [Madani 97]

- Les frottements sont secs et visqueux ;
- Les différentes liaisons sont rigides ;
- Les actionneurs sont des moteurs à courant continu (le couple moteur est directement proportionnel au signal de commande) ;
- Les capteurs ont des gains unitaires et sont de dynamique négligée.

Les moteurs sont inclus dans le modèle dynamique ; les commandes calculées sont les couples à appliquer par les moteurs à chaque charge de ces derniers.

Selon l'hypothèse sur les actionneurs, les commandes peuvent être obtenues en terme de tension en divisant les couples calculés par ($n_i k_{mi}$) où k_{mi} est la constante de couple du moteur i et n_i est le $i^{ème}$ rapport de réduction.

On obtient le modèle dynamique du robot SCARA en utilisant le formalisme de Lagrange [Madani 97]. Il est donné par [Benallegue 91] :

$$T = (M(q) + I_m)\ddot{q} + N(q,\dot{q}) + G(q) + H(\dot{q}) + T_{m0} \tag{I.65}$$

avec :

$$M(q) = \begin{bmatrix} I_1 + I_2 + m_2 L_1^2 + m_1 L_1 L_2 C_2 & I_2 + \frac{1}{2} m_2 L_1 L_2 C_2 \\ I_2 + \frac{1}{2} m_2 L_1 L_2 C_2 & I_2 \end{bmatrix} \tag{I.66}$$

$$I_m = \begin{bmatrix} n_1^2 I_{m1} & 0 \\ 0 & n_2^2 I_{m2} \end{bmatrix} \tag{I.67}$$

$$N(q,\dot{q}) = \begin{bmatrix} -m_2 L_1 L_2 S_2 \left(\dot{q}_1 \dot{q}_2 + \frac{\dot{q}_2^2}{2} \right) \\ m_2 L_1 L_2 S_2 \frac{\dot{q}_1^2}{2} \end{bmatrix} \tag{I.68}$$

$$H(\dot{q}) = \begin{bmatrix} n_1^2 (f_{v1} \dot{q}_1 + f_{s1} \operatorname{sgn}(\dot{q}_1)) \\ n_2^2 (f_{v2} \dot{q}_2 + f_{s2} \operatorname{sgn}(\dot{q}_2)) \end{bmatrix} \tag{I.69}$$

$$T = [T_1 \quad T_2] \tag{I.70}$$

$$g(q) = [0 \quad 0]^T \tag{I.71}$$

et :

$$\begin{cases} I_1 = \frac{1}{3} m_1 L_1^2 \ , \ I_2 = \frac{1}{3} m_2 L_2^2 \\ C_i = \cos q_i \ , \ S_i = \sin q_i \end{cases} \tag{I.72}$$

Où I_1 et I_2 sont les inerties par rapport aux axes de rotation Z_0 et Z_1 des deux articulations qui sont supposées cylindriques et uniformes.

Le vecteur des couples additifs T_{m0} représente l'effet de la charge, il est calculé par la matrice jacobienne. Cette dernière étant la dérivée du vecteur position de l'effecteur [Madani 97].

$$J_i(q) = \frac{\partial P}{\partial q_i} \tag{I.73}$$

Avec

$$P = [x \quad y]^T \tag{I.74}$$

En dérivant l'équation (I.12), on obtient pour le robot SCARA

$$J(q) = \begin{bmatrix} -l_1 S_1 - l_2 S_{12} & -l_2 S_{12} \\ l_1 C_1 + l_2 C_{12} & l_2 C_{12} \end{bmatrix} \tag{I.75}$$

Le couple dû au port de la charge est alors :

$$T_{m0} = m_0 J^T(q)\left[J(q)\dot{q} + \dot{J}(q,\dot{q})\dot{q} + g \right] \tag{I.76}$$

avec g défini dans l'équation (I.71).
Les valeurs numériques des différents paramètres du robot sont : [Madani 97]

$m_1 = 15.91$ kg
$m_2 = 11.36$ kg
$L_1 = L_2 = 0.432$ m
$f_{v1} = f_{v2} = 7\times10^{-4}$ N rad^{-1}s
$f_{S1} = f_{S2} = 10^{-3}$ N
$I_{M1} = I_{M2} = 10^{-4}$ kg m^2
$k_{M1} = k_{M2} = 0.18$ NmV^{-1}
$n_1 = 120$
$n_2 = 50$

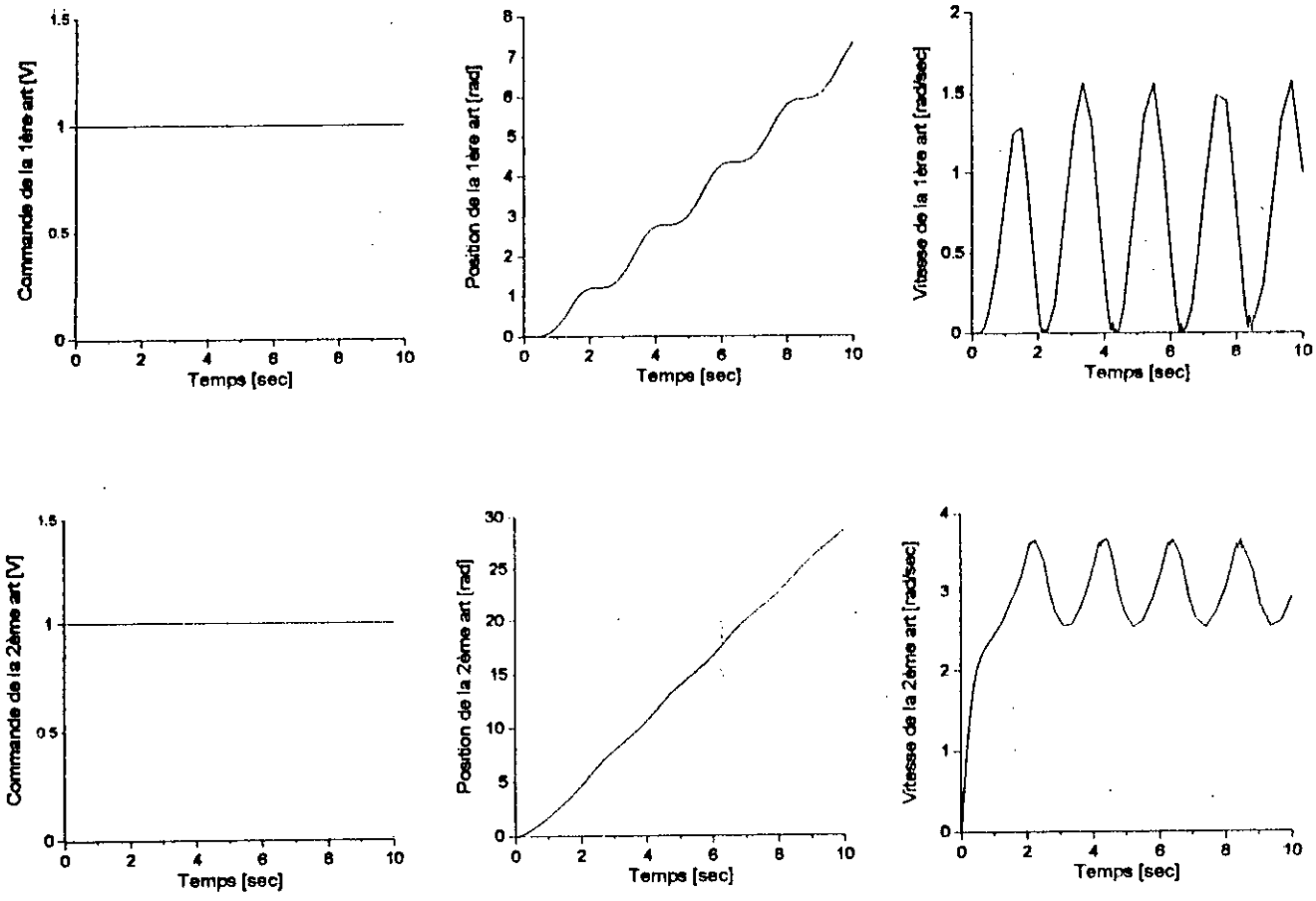

Figure1.7. ***Simulation en boucle ouverte du robot SCARA réponse indicielle en position et en vitesse***

I .3.4. Modèle dynamique du robot PUMA 560 :

Le robot manipulateur considéré assure trois mouvements rotationnels ; la première articulation évoluant sur un plan horizontal et les deux autres évoluant sur des plans verticaux Le modèle dynamique associé à ce manipulateur est :[Madani 97],[Armstrong 86]

$$T = M(q)\ddot{q} + N(q,\dot{q}) + G(q) + T_{m0} \qquad (I.77)$$

avec

$$M(q)=\begin{bmatrix} I_1+I_2C_{23}{}^2+I_3C_2{}^2+I_4C_2C_{23} & I_5S_{23}+I_6S_2 & I_5S_{23} \\ I_5S_{23}+I_6S_2 & I_7+I_4C_3 & I_8+0.5I_4C_3 \\ I_5S_{23} & I_8+0.5I_4C_3 & I_9 \end{bmatrix} \tag{1.78}$$

$$N(q,\dot{q})=\begin{bmatrix} -\left(2(I_3S_2C_2+I_2S_{23}C_{23})+I_4(C_2S_{23}+S_2C_{23})\right)\dot{q}_1\dot{q}_2-\left(2I_2S_{23}C_{23}+I_4C_2S_{23}\right)\dot{q}_1\dot{q}_3 \\ \quad +(I_6C_2+I_5C_{23})\dot{q}_2{}^2+(2I_5C_{23})\dot{q}_2\dot{q}_3+(I_5C_{23})\dot{q}_3{}^2 \\ \left(I_3C_2S_2+I_2C_{23}S_{23}+0.5I_4(S_2C_{23}+C_2S_{23})\right)\dot{q}_1{}^2-(I_4S_3)\dot{q}_2\dot{q}_3-(0.5I_4S_3)\dot{q}_3{}^2 \\ \left(I_2S_{23}C_{23}+0.5I_4C_2S_{23}\right)\dot{q}_1{}^2+(0.5I_4S_3)\dot{q}_2{}^2 \end{bmatrix} \tag{1.79}$$

$$G(q)=\begin{bmatrix} 0 \\ -(m_3L_2+0.5m_2L_2)gC_2-0.5m_3L_3gC_{23} \\ -0.5m_3L_3gC_{23} \end{bmatrix} \tag{1.80}$$

$$T=[T_1 \quad T_2 \quad T_3]^T \tag{1.81}$$

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

Avec

$I_1 = I_{yy1} + I_{xx2} + m_2d_2(d_2+e) + m_3d_2^2 + I_{xx3} + I_{xxt} + m_td_2^2 + I_{M1}$
$I_2 = I_{yy3} - I_{xx3} + I_{yyt} - I_{xxt} + m_tL_3^2$
$I_3 = I_{yy2} - I_{xx2} + (m_3 + m_t)L_2^2$
$I_4 = (m_3 + 2mt)L_2L_3$
$I_5 = (0.5m_3 + m_t)L_3d_2$

$I_6 = 0.5m_2L_2(d_2 + e) + (m_3 + m_t)d_2L_2$
$I_7 = I_{zz2} + I_{zz3} + m_3L_2^2 + I_{zzt} + m_t(L_2^2 + L_3^2) + I_{M2}$
$I_8 = I_{zz3} + I_{zzt} + m_tL_3^2$
$I_9 = I_8 + I_{M3}$

I_{Mi} moments d'inertie des différents moteurs
I_{xxt}, I_{yyt}, I_{zzt} moments d'inertie totale par rapport aux principaux axes de l'effecteur.
Le vecteur des couples additifs T_{m0} représente l'effet de la charge , il est calculé par la matrice jacobienne , cette dernière étant la dérivée du vecteur position de l'effecteur.

$$J_i(q) = \frac{\partial p}{\partial q_i} \tag{I.82}$$

pour le robot PUMA 560 , et en remplaçant p (la position de l'effecteur) par son expression donnée par l'équation (I.29)

$$J(q) = \begin{bmatrix} -S_1(L_2C_2 + L_3C_{23}) - d_2C_1 & -C_1(L_2S_2 + L_3S_{23}) & -C_1L_3S_{23} \\ C_1(L_2C_2 + L_3C_{23}) - d_2S_1 & -S_1(L_2S_2 + L_3S_{23}) & -S_1L_3S_{23} \\ 0 & -(L_2C_2 + L_3C_{23}) & -L_3C_{23} \end{bmatrix} \tag{I.83}$$

Le couple dû au port de la charge est alors :

$$T_{m0} = m_0J^T(q)\left[J(q)\ddot{q} + \dot{J}(q,\dot{q})\dot{q} + g\right] \tag{I.84}$$

avec : $$g = [0 \quad 0 \quad 9.81]^T \tag{I.85}$$

Les paramètres réels de ce robot sont :

- Masses des différentes liaisons :

$m_2 = 17.40\ kg$ $m_3 = 5.04\ kg$ $m_4 = 0.82\ kg$
$m_5 = 0.35\ kg$ $m_6 = 0.09\ kg$ $m_t = m_4 + m_5 + m_6 = 1.26\ kg$

- paramètres géométriques :

$d2 = 149.09\ mm$ $L_2 = 431.8\ mm$ $L_3 = 433.07\ mm$

- Paramètres d'inertie :

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

Tableau I.3. ***paramètres d'inertie du robot PUMA 560***

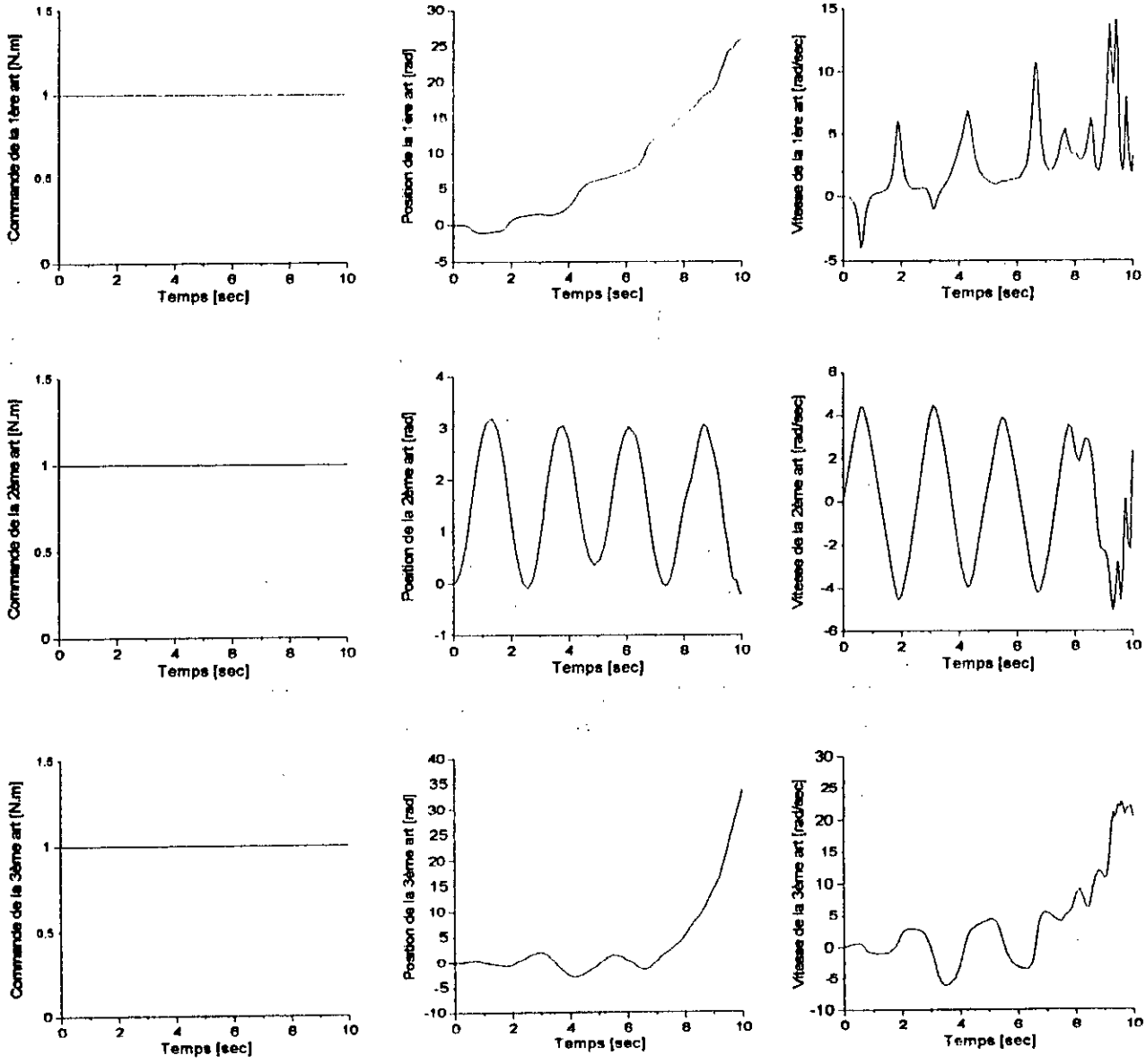

Figure 1.8. ***Simulation en boucle ouverte du PUMA 560***
réponse indicielle en position et en vitesse articulaires

I .4. Génération de trajectoire:

La dynamique du robot exige d'imposer des trajectoires réalisables. La continuité en position, vitesse et accélération offre au robot la possibilité de poursuivre la trajectoire avec des commandes réalisables. La décomposition de la tâche en plusieurs points intermédiaires nécessite une continuité du premier et du second ordre [Bali 95].
L'imposition d'une position finale constante par exemple, peut se faire par le passage par un régime transitoire suivant une trajectoire obtenue par un polynôme du troisième degrés permettant une continuité en position, en vitesse et en accélération pour chaque articulation.
Cette tâche se traduit par le passage d'un état d'équilibre à un autre état d'équilibre, ces deux états définiront les conditions aux limites du polynôme d'interpolation :

$$P(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \tag{I.86}$$

$$\begin{cases} p(0) = \theta_0 & \dot{p}(0) = \dot{\theta}_0 \\ p(t_f) = \theta_f & \dot{p}(t_f) = \dot{\theta}_f \end{cases} \tag{I.87}$$

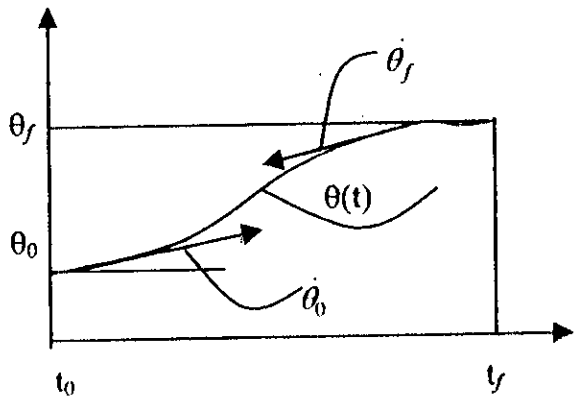

Figure I .9. ***génération d'un polynôme d'interpolation***

Généralement : $$\dot{\theta}_0 = \dot{\theta}_f = 0 \tag{I.88}$$

A partir des conditions aux limites, équation (I.87), on calcule les différents coefficients du polynôme *p(t)*, on aura alors :

$$\begin{cases} a_0 = \theta_0 \\ a_1 = \dot{\theta} \\ a_2 = \dfrac{3}{t_f^{\,3}}\left(\theta_f - \theta_0\right) - \dfrac{2}{t_f}\dot{\theta}_0 - \dfrac{1}{t_f}\dot{\theta}_f \\ a_3 = -\dfrac{2}{t_f^{\,3}}\left(\theta_f - \theta_0\right) + \dfrac{1}{t_f^{\,2}}\left(\dot{\theta}_f + \dot{\theta}_0\right) \end{cases} \tag{I.89}$$

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

$$q_d(t) = \begin{cases} q_d(0) + \dfrac{D}{2\pi}\left[2\pi\dfrac{t}{t_f} - \sin\left(2\pi\dfrac{t}{t_f}\right)\right] & \text{pour } 0 \le t \le t_f \\ q_d(t_f) & \text{pour } t > t_f \end{cases} \tag{I.90}$$

avec : $$D = q_d(t_f) - q_d(0) \tag{I.91}$$

Le terme D dans l'équation (I.91) est le déplacement, t_f est l'instant final du mouvement, et q_d est la trajectoire de référence.

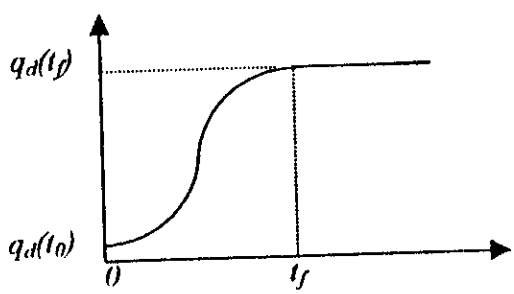

Figure I .10. ***trajectoire cycloïdale***

Pour le robot PUMA 560, il existe une trajectoire cycloïdale test proposé par LEAHVY. Les différentes articulations se déplacent respectivement de la position(-50°, -135°, 135°) à la position(45°, -85°, 30°)en un temps de mouvement égal à 1.5 secondes(rapide). Cette trajectoire est choisie car elle excite toute la dynamique de ce bras de robot.
Si maintenant la consigne n'est pas constante, pour une trajectoire donnée(trajectoire de l'effecteur du bras de robot), on peut aisément déterminer les déplacements et vitesses dus à

chaque articulation $(\theta_i, \dot{\theta}_i)$, ceci en utilisant les équations du modèle géométrique et cinématique inverse des robots.
En effet, à chaque point de coordonnées cartésiennes, on détermine les coordonnées généralisées correspondantes. Suivant la trajectoire de l'effecteur (élément terminal du bras de robot), on trouve une relation donnant les coordonnées généralisées (q_i) qui décrivent cette même trajectoire.

I .5.Conclusion :

La modélisation d'un processus dynamique est la première étape nécessaire et primordiale en vue de sa commande. Il est alors important de considérer le maximum de phénomènes possible lors de l'établissement du modèle, pour que ce dernier représente le plus fidèlement possible le processus, ce qui est généralement très difficile et délicat à obtenir.

Dans ce chapitre, on a étudié la procédure à suivre pour établir le modèle d'un bras de robot. On a alors établi les modèles dynamiques de deux bras manipulateurs fortement non linéaires et complexes : le robot PUMA 560 (en se limitant aux trois premières liaisons) et un robot SCARA (à deux degrés de libertés), ceci en utilisant le formalisme de Larange – Euler.

Quelques trajectoires standards utilisées pour les tests de commande sont présentées en fin de chapitre ; ainsi que les résultats de simulation en boucle ouverte des modèles dynamiques des deux bras manipulateurs considérés. Ces derniers montrent clairement le comportement non linéaire des deux bras de robot.

Chapitre II

Les réseaux de neurones et la commande

Chapitre II

LES RESEAUX DE NEURONES ARTIFICIELS ET LA COMMANDE

II .1. Introduction aux réseaux de neurones artificiels :

Au cours des dernières années, l'une des évolutions les plus marquantes des réseaux de neurones formels était l'abandon de la métaphore biologique au profit de fondements théoriques solides dans le domaine des statistiques [Bouhali 97], [Slotine 92] ; on sait à présent que la propriété fondamentale des réseaux de neurones est l'approximation universelle parcimonieuse. De plus, le développement d'algorithmes performants pour l'entraînement et l'apprentissage des réseaux de neurones bouclés ou non bouclés, leur a ouvert de nouvelles perspectives d'utilisation. En particulier, les réseaux de neurones se sont avérés parfaitement adaptés dans le domaine de l'automatique comme éléments de système de commande de processus dynamiques non linéaires. Des travaux récents ont permis de replacer la mise en œuvre des réseaux de neurones comme modèle de processus ou correcteurs des systèmes dans le cadre général de la commande classique.

Dans ce qui suit, nous allons présenter les architectures de réseaux de neurones artificiels les plus courantes dans le domaine de la commande, à savoir les réseaux statiques, les réseaux de fonctions à bases radiales, les réseaux récurrents et les réseaux dynamiques, ainsi que les algorithmes d'apprentissages correspondants. La deuxième partie de ce chapitre va porter sur les différentes architectures d'identification et de commande de processus dynamiques par réseaux de neurones artificiels.

II .1.1. Le neurone artificiel :

Un neurone artificiel est défini comme une unité fonctionnelle caractérisée par une fonction d'activation f décrivant l'état de sa sortie y en fonction de ses entrées x_i. (figure II.1).

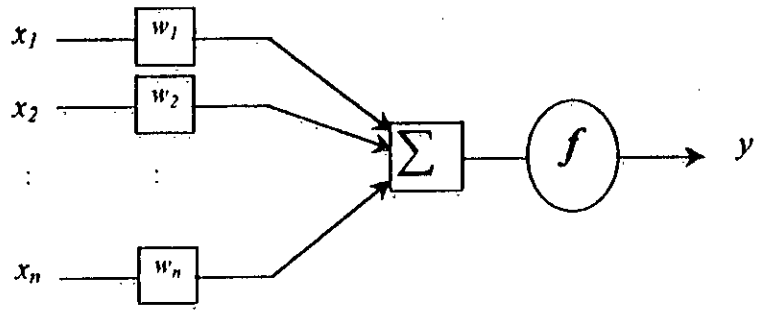

Figure II .1. ***Le modèle artificiel d'un neurone.***

Ainsi, la sortie du neurone sera :

$$y = f\left(\sum_{i=1}^{n} w_i x_i + b\right) \tag{II.1}$$

où w_i sont les poids de pondération des connexions.

b (biais) est la polarisation du neurone.

La fonction non linéaire f permet donc la transformation du signal d'entrée x_i non borné, en un signal de sortie borné. Généralement, cette fonction est une sigmoïde (figure II.2), décrite par l'équation :

$$f(x) = \frac{1}{1+exp(-ax)} \tag{II.2}$$

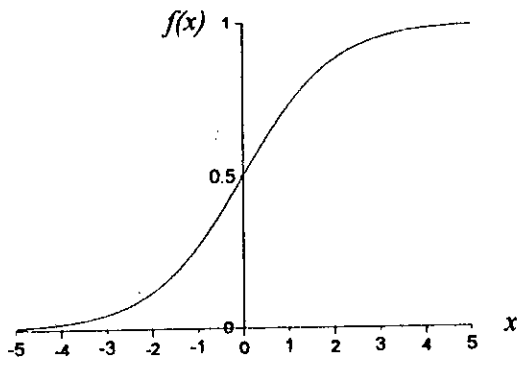

Figure II .2. ***La fonction d'activation sigmoïde***

II .1.2. Les réseaux de neurones artificiels :

Un réseau de neurones est l'interconnexion pondérée de plusieurs neurones considérés comme éléments de base. Ce réseau effectue une application de l'espace des entrées I^m dans l'espace des

sorties R^n d'une façon parallèle et distribuée. On peut décomposer un réseau de neurones en plusieurs champs de neurones, appelés couches. On distingue alors : la couche d'entrée, les couches cachées et la couche de sortie.

Plusieurs architectures de réseaux sont envisagées selon la topologie des connexions des neurones ainsi que la forme des fonctions d'activation des neurones.

a- Les réseaux de neurones statiques :

Un RNA statique est caractérisé par la topologie de ses connexions qui relient les différentes couches du réseau et qui ne contient pas de boucles synaptiques (neuronale) fermées (figure II.3). Les fonctions d'activation des neurones sont généralement sigmoïdales ou de type TANH ; néanmoins, elles peuvent prendre d'autres formes comme la fonction *sat (saturation)*, ou la fonction *sgn (signe)* selon l'utilisation du réseau. Le paramètre "*a*" dans la fonction sigmoïde (équation II.2) peut varier d'un neurone à un autre, ce qui est parfois conseillé pour éviter le problème de l'initialisation des poids.

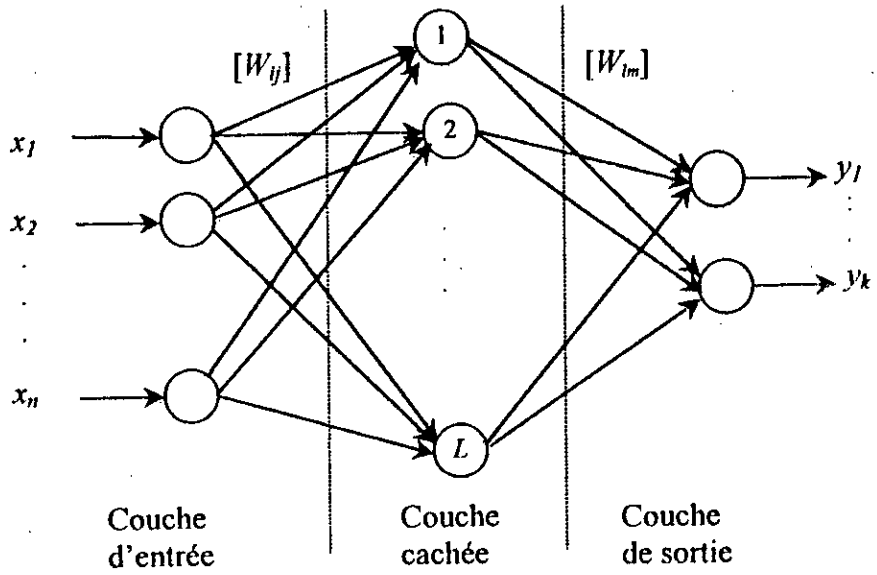

Figure II .3. ***Architecture d'un R.N.A. statique***

La sortie du réseau sera :

$$y_m = \sum_{l=1,L} w_{lm} \cdot f\left(\sum_{\substack{i=1,L \\ j=1,n}} w_{ij} \cdot x_j + b_1 \right) + b_2 \qquad \text{(II.3a)}$$

ou sous une écriture matricielle :

$$y = W_{kL} \cdot f\left(W_{Ln} X_n + b1\right) + b2 \qquad \text{(II.3b)}$$

b- Les réseaux de fonctions à bases radiales :

La particularité de ce type de réseaux consiste en la forme des fonctions d'activations des neurones de la couche cachée qui sont des fonctions gaussiennes (figure II.3a). Ainsi, les neurones de la couche cachée ne réagissent significativement qu'à une partie restreinte de l'espace d'entrée, ce qui permet au réseau de subdiviser cet espace en domaines probabilistes de dimensions égales à la dimension de l'espace d'entrée et équipe le réseau du pouvoir de reconstruction de la sortie correspondante à chaque sous espace d'entrée.

Les fonctions d'activation du neurone i de la couche cachée sont caractérisées par leurs centres $c(i,j)$ et leurs variances ou généralement appelées rayons $v(i\ ,\ j)$, où j désigne la $j^{ème}$ entrée du réseau. On définit aussi le taux de recouvrement des gaussiennes pour chaque entrée. Le choix de ces paramètres n'est pas arbitraire mais basé sur la théorie du signal et les techniques de reconstruction (annexe) [Slotine 92] [Renders 95].

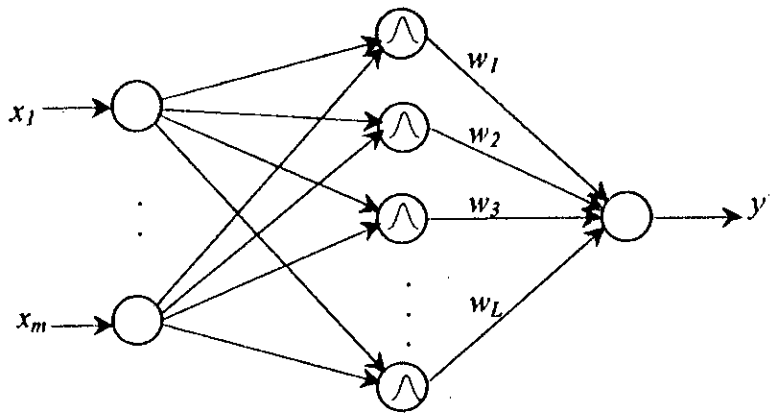

Figure II .3a . ***Architecture d'un réseau RBF***

Ainsi, le réseau va construire la sortie suivante :

$$Y = W \cdot S(X) \tag{II.4a}$$

$$S_i(X) = exp\left(-1/2 \sum_{j=1}^{m} \frac{(x_j - c(i,j))^2}{v(i,j)^2} \right) \quad i = 1,n. \tag{II.4b}$$

avec $\begin{cases} W: \text{ matrice des poids de connexion de la couche cachée.} \\ S(X): \text{ vecteur des sorties de la couche cachée.} \end{cases}$

Chaque neurone de la couche cachée contient donc un nombre de fonctions gaussiennes égal au nombre d'entrées du réseau, chaque fonction représente un sous espace d'une seule entrée, ainsi, la distribution des fonctions de la même entrée à travers les neurones de la couche cachée doit recouvrir tout l'espace de cette entrée (figure II.3b).

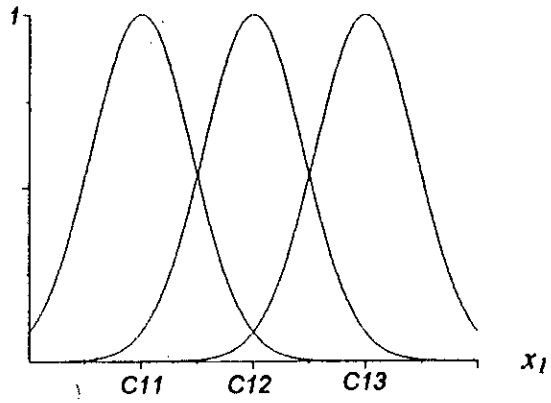

Figure II .3b. ***La subdivision de l'espace d'une seule entrée à travers trois neurones de la couche cachée sous forme de fonctions gausiennes.***

La figure (II.3c) représente le traitement de l'espace des entrées ($m=2$), subdivisé en sous espaces réguliers ($l=3$), au niveau de la couche cachée. Donc, on doit avoir $n=l^m$ gaussiennes (3^2 sous espaces), chaque gaussiènne représente le domaine d'action d'un neurone.

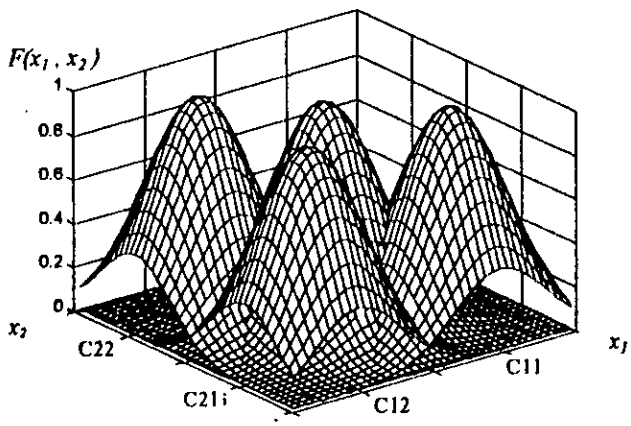

Figure II .3.c. *portion de l'espace de sortie de la couche cachée en fonction des deux entrées.*

Pour un nombre d'entrées supérieur à deux, c'est à nous d'imaginer la subdivision et la forme des sous espaces ou généralement appelés hypercubes, où seront traitées les informations.

Comme on le remarque, le nombre de neurones peut être très élevé selon le nombre d'entrées et la subdivision en sous espaces.

Une comparaison qualitative entre les réseaux statiques et les réseaux de fonction à base radiale est résumée dans le tableau suivant:

	Sigmoïdale	A base radiale
Approximation universelle	Oui	Oui
Dépendance en les poids	Non linéaire	Linéaire
Effet d'un changement paramétrique	Globale	Local
Domaine de définition (support)	Infini	Fini (borné)
Interprétation des paramètres	Non	Possible
Charge de calcul réductible	Non	Oui

Tableau II .1. ***comparaisons des propriétés des réseaux de neurones de fonctions d'activations sigmoidales et à bases radiales.***

Généralement, dans les réseaux à bases radiales, on introduit un terme de normalisation à la sortie du réseau qui dépend des sorties des unités de la couche cachée. Cela se fait dans le but de rendre les variations des paramètres du réseau plus significatives. A la sortie d'un réseau normalisé, on aura :

$$Y = \frac{W \cdot S(X)}{\sum_{i=1}^{n} S_i(X)}. \qquad \text{(II.5.a)}$$

$$S_i(X) = exp\left(-\frac{1}{2} \sum_{j=1}^{m} \frac{\left(x_j - c(i,j)\right)^2}{v_j^{\,2}} \right) \qquad i = 1,n \qquad \text{(II.5.b)}$$

c- Les réseaux récurrents :

Afin d'équiper les réseaux de neurones d'un caractère dynamique, propriété sollicitée dans plusieurs applications, ainsi que d'un pouvoir de mémorisation d'informations lors d'un traitement des données, des connexions ou des boucles fermées s'avèrent nécessaires. Dans ce cadre, nous allons présenter quelques architectures de réseaux récurrents qui diffèrent par leurs topologies d'interconnexions [Kosco 92], [Yang 96].

- **Réseau récurrent « with self feedback »:**

Ce type de réseaux est caractérisé par les boucles de retour sur chaque neurone de la couche cachée (figure II.4a), ce qui permet au réseau de traiter les données présentes en tenant compte des données précédentes.

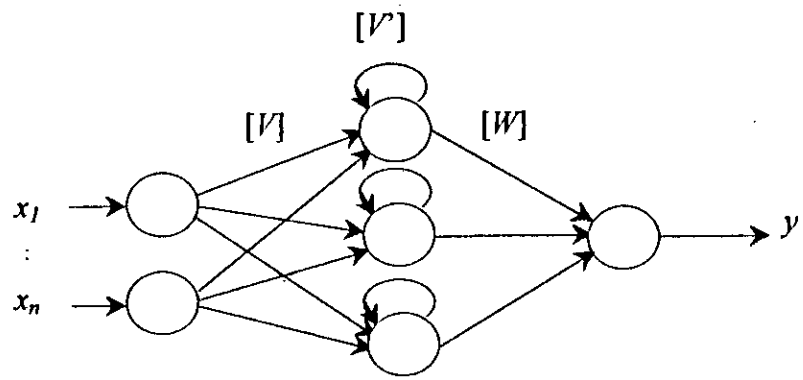

Figure II .4a . ***réseau récurrent « with self feedback »***

La sortie du réseau à l'instant *k* est:

$$y(k) = W.f\left(V.X + V'.S(k-1) + b_1\right) + b_2 \qquad \text{(II.6)}$$

avec
- *V'* : vecteur de poids de connexion de la couche cachée.
- *S(k-1)* : vecteur des sorties de la couche cachée à l'instant *k-1*.

L'utilisation de ce type de réseau a eu un succès considérable dans la modélisation des processus dynamiques notamment en automatique, où ils sont utilisés pour l'identification et la commande des systèmes complexes, ainsi qu'en économie, pour les estimations des modèles prévisionnels économiques.

- **Réseau récurrent complètement connecté «Fully connected recurrent NN»:**

Dans le même but, une autre forme d'interconnexions est proposée, où les neurones de la couche cachée sont complètement connectés (figure II.4b).

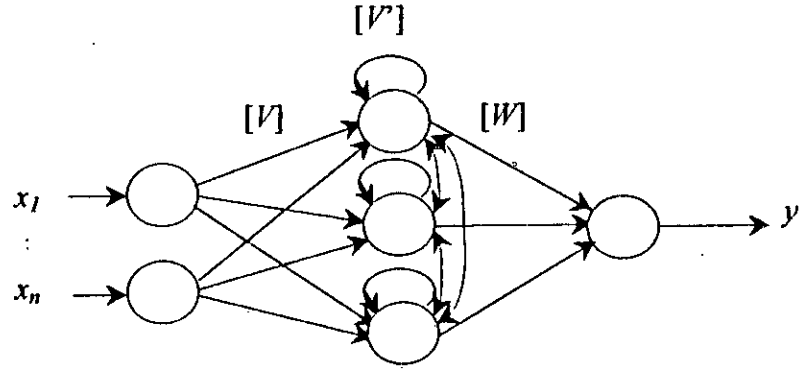

Figure II .4b . ***Fully connected recurrent NN***

La sortie de ce réseau est régit par l'équation suivante :

$$y(k) = W.f(V.X + V'.S(k-1) + b_1) + b_2 \tag{II.7}$$

avec { *V'*: Matrice des poids des connexions de la couche cachée.
S(k-1): Matrice des sorties de la couche cachée à l'instant *k-1*.

- **Réseau récurent « Feed forward récurrent NN» :**

La structure de ce réseau comporte des boucles de retour de la sortie, ce qui permet au réseau de tenir compte de la valeur précédente de la sortie pour le calcul de la nouvelle sortie (figure II.4b).

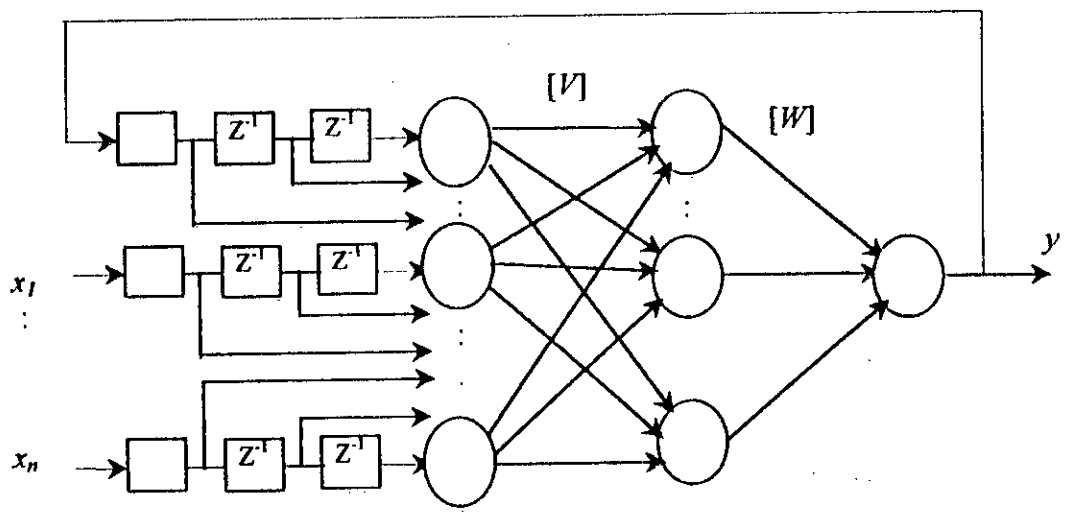

Figure II .4c . ***Feed forward recurrent NN***

La sortie de ce réseau est régie par l'équation suivante:

$$y(k) = W \cdot f(V.X + V'.S(k-1) + b_1) + b_2 \qquad \text{(II.8)}$$

avec $\begin{cases} V' \text{: Matrice des poids des connexions de la couche cachée.} \\ S(k-1) \text{: Matrice des sorties de la couche cachée à l'instant } k-1. \end{cases}$

Les combinaisons de ces structures de réseaux récurrents peuvent donner lieu à d'autres architectures plus complexes, et dont les avantages ne sont pas encore reconnus.

d- Les réseaux dynamiques:

Avec une architecture analogue à celle d'un réseau récurrent complètement connecté, un réseau dynamique, représenté dans la figure (II.5), est régi par le système d'équations d'état suivant :

$$\begin{cases} \dot{X} = -X + W \cdot G(X) + V \cdot Z(k) \\ Y = H \cdot X^* \end{cases} \qquad \text{(II.9a)}$$

Pour l'échantillon k du vecteur d'entrée avec : X^*, le vecteur des points d'équilibres vers lesquels convergent les états x_i des neurones de la couche cachée [Kosco 92] , [Karakasoglu 93]. A l'équilibre ($\dot{X}$ =0), la solution vérifie l'équation :

$$X^* = W.G(X^*) + V.Z(k) \qquad \text{(II.9b)}$$

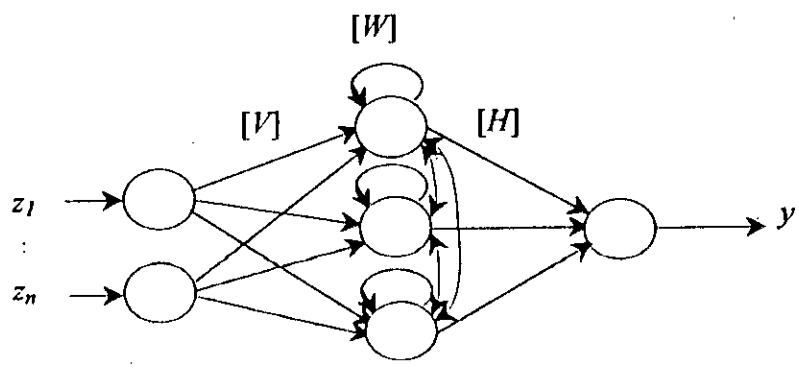

Figure II .5. ***réseau dynamique***

On remarque bien qu'à chaque échantillon d'entrée, le réseau doit établir son équilibre interne pour pouvoir fournir une sortie correspondante à cet échantillon.

II .2. Les algorithmes d'apprentissages:

L'apprentissage des réseaux de neurones est l'ensemble des algorithmes d'ajustement de ses paramètres (les poids, les centres, les variances,...) pour lui permettre une application entre l'espace d'entrée qu'on lui présente et l'espace de sortie correspondant.
Trois classes d'apprentissages sont à distinguer :

- **L'apprentissage supervisé :** disposant de deux espaces entrée/sortie, le réseau peut être entraîné, en ajustant ses paramètres systématiquement, à approximer au mieux les sorties désirées correspondantes à chaque vecteur d'entrée en se basant sur le principe *essai - erreur.*

- **L'apprentissage semi-supervisé:** ce mode d'apprentissage ressemble au précédent par la présence d'un critère de jugement sur l'échec ou le succès de la réponse, mais sans connaître exactement la sortie désirée.

- **L'apprentissage non supervisé:** ce mode d'apprentissage est souvent utilisé lors du traitement des signaux stochastiques, où on ne dispose que de très faibles informations sur l'espace d'entrée. Le but de cet apprentissage est d'équiper le réseau du pouvoir de classification des données, ce qu'on appelle parfois « la séparation aveugle des sources ». Ainsi, les données de la même classe vont, à l'issue de l'apprentissage, se retrouver réunies dans des groupes.

L'algorithme de Backpropagation :

Cet algorithme est l'un des plus utilisé pour l'ajustement des paramètres des réseaux de neurones et conduire ces derniers à savoir reconnaître les sorties désirées, en se basant sur la minimisation du critère quadratique J par la méthode du gradient [Freeman 92].

Avec : $$J = \frac{1}{2} e^2 = \frac{1}{2} (y - \hat{y})^2 \qquad \text{(II.10)}$$

Plusieurs variantes ont été dérivées pour accélérer sa convergence et améliorer ses performances. Dans ce qui suit, nous allons présenter le formalisme de l'algorithme appliqué sur un réseau de neurones à n entrées, m sorties et L couches cachées (figure II.6).

Disposant d'un ensemble d'apprentissage composé de k paires de vecteurs : (X_1,Y_1), (X_2,Y_2),... (X_P,Y_P), avec :
$X_p=(x_{p1},x_{p2},\ldots x_{pn}) \in R^n$ un vecteur d'entrée

$Y_p=(y_{p1},y_{p2},\dots y_{pn})\in R^m$ le vecteur de sortie correspondant désiré.

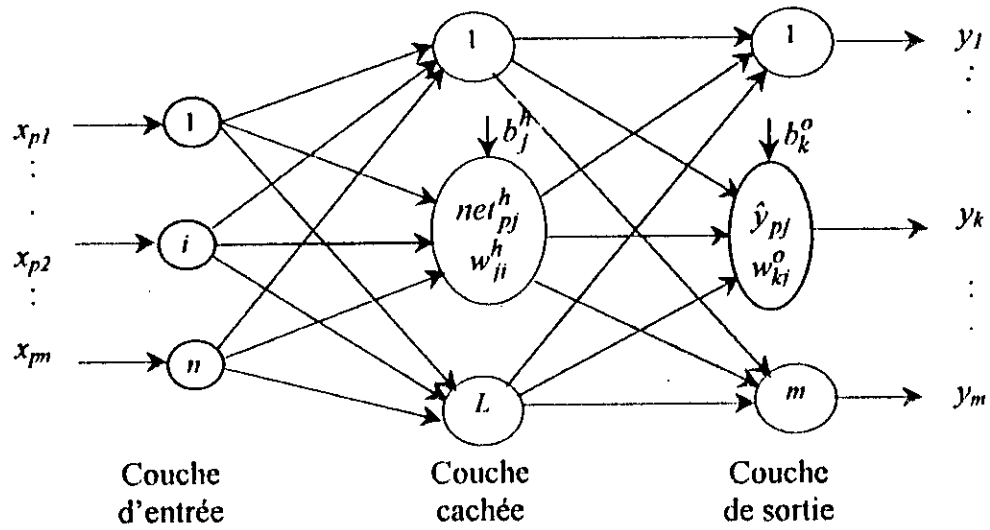

Figure.II.6. ***Backpropagation pour un R.N.A. statique***

W_{ij} : les poids des connexions entre les neurones i et j .

net^h_{pj} : l'entrée totale du neurone j de la couche cachée h pour l 'exemple p.

net^o_{pj} : l'entrée totale du neurone j de la couche de sortie o pour l 'exemple p.

b^h_j : la polarisation du neurone j de la couche cachée h .

f : la fonction d'activation.

μ : le pas d'apprentissage (ou pas d'adaptation).

On veut entraîner le réseau, par l'algorithme de Backpropagation, à approximer la fonction $Y=F(X)$ sur tout l'ensemble d'apprentissage. Pour cela, on procède comme suit :

1- Appliquer un vecteur d'entrée X_p à l'entrée du réseau.

2- Calculer les entrées totales des neurones de la couche cachée :

$$net^h_{pj}=\sum_{i=1}^{n} W^h_{ji}x_{pi}+b^h_j \qquad \text{(II.11)}$$

3- Calculer les sorties des neurones de la couche cachée :

$$I_{pj}=f^h_j(net^h_{pj}) \qquad \text{(II.12)}$$

4- Calculer les entrées totales des neurones de la couche de sortie :

$$net_{pj}^{o} = \sum_{i=1}^{L} W_{kj}^{o} I_{pj} + b_{k}^{o} \qquad \text{(II.13)}$$

5- Calculer les sorties réelles du réseau :

$$\hat{y}_{pk} = f_k^o(net_{pk}^o) \qquad \text{(II.14)}$$

6- Calculer les termes d'erreur sur chaque unité de sortie :

$$e_{pk}^o = (y_{pk} - \hat{y}_{pk}) f_k^{o'}(net_{pk}^o) \qquad \text{(II.15)}$$

7- Calculer les termes d'erreur sur chaque unité cachée :

$$e_{pj}^h = f_j^{h'}(net_{pj}^h) \sum_{k=1}^{m} e_{pk}^o w_{kj}^o \qquad \text{(II.16)}$$

8- Ajuster les poids de la couche de sortie :

$$W_{kj}^o = W_{kj}^o + \mu e_{pk}^o I_{pj} \qquad \text{(II.17)}$$

9- Ajuster les poids de la couche cachée :

$$W_{ji}^h = W_{ji}^h + \mu e_{pj}^h x_{pi} \qquad \text{(II.18)}$$

10- Test d'arrêt

$$Ep = \frac{1}{2} \sum_{k=1}^{m} \left(e_{pk}^o\right)^2 \leq \text{ erreur exigée.} \qquad \text{(II.19)}$$

Notons que cet algorithme n'est pas propre aux réseaux statiques, il est aussi souvent utilisé pour les réseaux récurrents, représentés dans la partie précédente, en traitant l'entrée de chaque unité comme une entrée externe même si elle provient d'une récurrence du réseau.

- **Apprentissage des réseaux à bases radiales:**

Dans la plupart des applications courantes des réseaux RBF, on dispose de centres et de rayons fixés une fois pour toutes, les seuls paramètres ajustables sont les poids w_i de chaque unité cachée.

Dans ce qui suit, on va présenter l'algorithme d'ajustement des poids pour un réseau RBF normalisé en se basant sur la méthode de descente du gradient. Pour les réseaux non normalisés, il suffit d'enlever le terme de normalisation.

Soit le critère à minimiser :

$$J = \frac{1}{2}(y - \hat{y})^2 \tag{II.20}$$

le terme de normalisation et l'ajustement des poids :

$$R(X) = \sum_{i=1}^{n} S_i(X) \tag{II.21}$$

$$\frac{dJ}{dw_i} = (y - \hat{y})\frac{Si(X)}{R(X)} \tag{II.22}$$

$$\Delta w_i = -\mu_w \frac{dJ}{dw_i} \tag{II.23}$$

Dans le cas où la fixation des centres et des rayons a priori ne serait pas possible, ou aussi si le nombre de neurones requis pour l'accomplissement de la tache est trop élevé, la meilleure solution est de poser un nombre restreint de neurones avec des centres et des rayons ajustables. Cela peut se faire comme suit :

$$\frac{dJ}{dc(i,j)} = \frac{dJ}{dS_i}\frac{dS_i}{dc(i,j)} \tag{II.24}$$

$$\frac{dJ}{dv(i,j)} = \frac{dJ}{dS_i}\frac{dS_i}{dv(i,j)} \tag{II.25}$$

$$\frac{dJ}{dS_i} = (y - \hat{y})\left[\frac{w_i R(X) - WS(X)}{R(X)^2}\right] \tag{II.26}$$

$$\frac{dS_i}{dc(i,j)} = S_i \frac{x_i - c(i,j)}{v(i,j)^2} \tag{II.27}$$

$$\frac{dS_i}{dv(i,j)} = S_i \frac{(x_i - c(i,j))^2}{v(i,j)^3} \tag{II.28}$$

ainsi, on aura :

$$\Delta c(i,j) = -\mu_c \frac{dJ}{dc(i,j)} \tag{II.29}$$

$$\Delta v(i,j) = -\mu_v \frac{dJ}{dv(i,j)} \tag{II.30}$$

- **Apprentissage des réseaux dynamiques :**

Pour les réseaux dynamiques à trois couches, p entrées, q unités cachées et une sortie, dont la couche cachée est entièrement connectée, l'algorithme d'apprentissage s'obtient par la méthode de descente du gradient dans le but de minimiser l'erreur instantanée. Il se résume comme suit :

Le critère à minimiser étant :

$$J = (y - \hat{y})^2 = (y - HX^*), \tag{II.31}$$

ce qui donne :

$$\Delta h_i = -\mu_h (y - \hat{y}) x_i^* \qquad i = 1,q \tag{II.32}$$

$$\Delta w_{ij} = \begin{cases} -\mu_w h_i (y - \hat{y}) g_i(x_i^*) & i,j = 1,q \quad si \quad i \neq j \\ 0 & si \quad i = j \end{cases} \tag{II.33}$$

$$\Delta v_{ij} = -\mu_v h_i (y - \hat{y}) z_j \qquad i = 1,q \quad j = 1,p \tag{II.34}$$

Il est à noter que la sélection d'une fonction d'activation qui se sature rapidement permet une stabilité exponentielle des points d'équilibre, ce qui assure une convergence rapide du réseau dynamique. Cet algorithme d'apprentissage permet une capacité d'apprentissage rapide et donc souhaitable en identification et commande des systèmes dynamiques.

II .3. Les réseaux de neurones et l'approximation de fonctions:

Les données présentées au réseau lors de l'apprentissage sont généralement limitées et sous forme d'échantillons représentatifs de l'espace d'entrée (Annexe). Le rôle essentiel du réseau est de pouvoir interpoler et extrapoler ces données, en établissant une fonction non linéaire qui recouvre tout l'espace d'entrée, et savoir reconstruire les liens entre de nouvelles entrées dans le même espace et leurs sorties correspondantes; ce qui est appelé « pouvoir de généralisation ».

Le caractère non linéaire des réseaux de neurones les prédispose à être les candidats naturels pour l'approximation de fonctions non linéaires, mais ils approximent aussi bien les fonctions linéaires.

L'emploi des réseaux de neurones plutôt que des techniques classiques pour l'approximation des fonctions peut se justifier par les avantages suivants :

- simplicité de mise en œuvre (peu d'analyses mathématiques préliminaires).
- capacité d'approximation universelle prouvée.
- possibilité de prendre le point de vue « processus = boite noire ».
- robustesse par rapport à des défaillances internes du réseau.
- capacité d'adaptation.

Parfois, en voulant assurer la capacité d'approximation universelle, on est amené à augmenter le nombre de neurones dans le réseau ; se faisant, on observe une diminution des performances de généralisation à cause du phénomène de sur-paramètrisation. Un compromis est alors toujours à faire. Ceci constitue l'inconvénient principal des réseaux de neurones.

II .4 . Les réseaux de neurones et la commande:

Vu le caractère non linéaire des réseaux de neurones, et grâce au fait qu'ils possèdent la capacité d'auto-organisation et de stockage implicite d'informations ainsi que d'autres propriétés dont on découvrira les avantages au fur et à mesure dans cette partie, l'identification et la commande des systèmes dynamiques ont eu recours à l'utilisation des réseaux de neurones comme des modélisateurs et régulateurs adaptatifs des processus non linéaires.

Dans le cadre de la commande, un RNA est considéré comme un ensemble structuré d'unités agissant d'une façon distribuée et parallèle sur le processus dans le but de lui assurer une conduite souhaitable face à sa structure caractérisée par des non-linéarités, dimensionnalités élevées (multivariables), erreurs de modélisation, bruits de mesures ...

L'apprentissage peut se voir comme un moyen de synthétiser automatiquement une loi de commande « control mapping » sur la base du principe *essais - erreurs.* Si l'on considère plus particulièrement l'aspect *on line* de l'apprentissage, ce sont surtout les facultés d'optimisation des performances et l'adaptation à un environnement incertain qu'il faut souligner.

II .4 .1. L'identification par réseaux de neurones :

L'identification d'un système permet de construire un modèle sur lequel on se base pour concevoir une stratégie de commande. le pouvoir d'approximation de fonctions non linéaires par les réseaux de neurones est exploité dans ce domaine. La figure (II.7) illustre le principe d'identification d'un système représenté par l'équation suivante (II.35) par deux stratégie :

$$y(k)=f(y(k-1),y(k-2),\ldots,y(k-n_y),u(k-1),u(k-2),\ldots,u(k-n_u),e(k-1),e(k-2),\ldots,e(k-n_e)) \qquad (II.35)$$

- le modèle NARX (Non linéaire ARX) qui utilise le prédicteur :

$$\hat{y}(k) = \hat{f}(y(k-1), y(k-2), u(k-1), u(k-2)) \tag{II.36}$$

- le modèle NARMAX(Non linéaire ARMAX) qui utilise le prédicteur :

$$\hat{y}(k) = \hat{f}(y(k-1), y(k-2), u(k-1), u(k-2), \hat{y}(k-1), \hat{y}(k-2)). \tag{II.37}$$

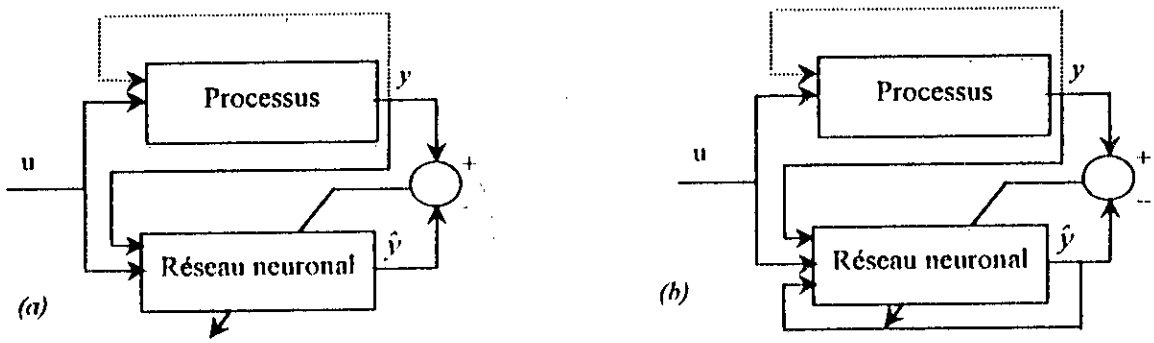

Figure II .7. ***Identification par réseaux de neurones***
(a) modèle NARX ***(b) modèle NARMAX***

Dans ce domaine, on pourrait utiliser les réseaux complètement récurrents dont l'entrée sera u(k) et la sortie $\hat{y}(k)$, cette approche rend le réseau libre de développer sa propre représentation d'état.

Remarque : Les expériences conduites dans le domaine de l'identification par RNA ne permettent de conclure, d'une manière ou d'une autre, la favorisation d'une architecture bien définie en terme d'atténuation des effets dus aux bruits ni de rejet des perturbations, ça dépend plus du système à identifier que du réseau utilisé.

II .4.2. Les stratégies de commandes par réseaux de neurones:

- **La commande supervisée:**

Comme son nom l'indique, cette technique de commande est basée sur l'utilisation d'un régulateur existant déjà (figure II.8). La sortie de ce régulateur sera une fonction à approximer par le réseau. Une fois cette tache accomplie, le réseau sera implémenté à la place de l'ancien régulateur pour conduire le système en imitant son comportement. Cela se fait dans le but de pouvoir généraliser le comportement de la commande pour des cas qui ne se sont pas déjà présentés.

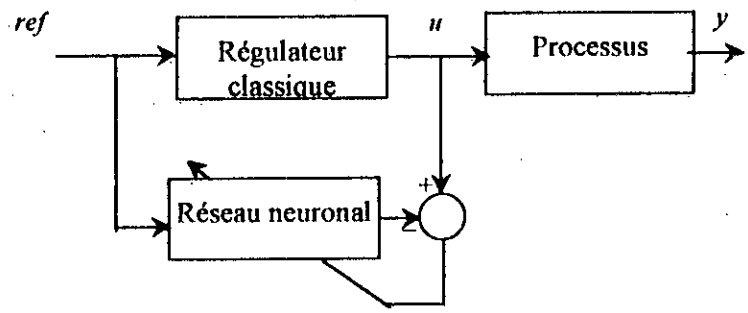

Figure II .8. ***la commande supervisée(phase d'apprentissage)***

- **La commande par apprentissage général:**

Cette approche procède en deux étapes successives; la première consiste à identifier le modèle inverse du processus en injectant la sortie du processus, résultant d'une certaine entrée u(t), comme entrée du réseau qui sera entraîné par rétropropagation afin de produire la sortie la plus proche possible de u(t). Comme étape de commande, le réseau sera placé devant le processus pour le conduire en boucle ouverte (figure II.9).

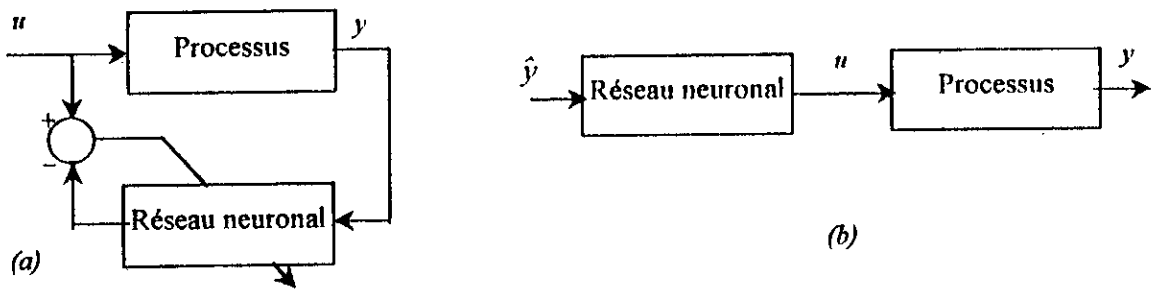

Figure II .9. ***commande par apprentissage général.***
(a) phase d'apprentissage. ***(b) phase de commande.***

- **La commande par identification neuronale du modèle direct:**

C'est une commande indirecte qui consiste à identifier le modèle du processus et puis chercher une stratégie de commande en supposant que ce modèle coïncide parfaitement avec le processus réel « principe d'équivalence certaine ».

- **La commande par apprentissage spécialisé:**

Dans cette approche, le réseau conduit directement le système en utilisant l'erreur de sortie (ou un indice de performance) pour ajuster ses poids en ligne, ce qui est généralement adopté dans la commande adaptative classique. On remarque qu'il ne faut pas nécessairement une phase d'apprentissage préliminaire, l'adaptation est continuelle en se focalisant directement sur le domaine d'intérêt.

II .5.Conclusion :

Dans ce chapitre, nous avons présenté les différentes structures des réseaux de neurones artificiels, les avantages que ces derniers offrent surtout dans le domaine de l'identification et la commande des processus.

Cependant, leur emploi en commande de processus ne se justifie que dans le cas où il est difficile, voire impossible, de concevoir ''classiquement'' un système de commande (avec propriétés fixées) basé sur la connaissance disponible à priori, capable de conduire le système de façon satisfaisante. Ce qui est le cas des processus dynamiques complexes présentant des non linéarités, dimension élevée, objectifs et contraintes de commande multiples, erreurs de modélisation, bruits de mesure, perturbations.

Enfin, en utilisant les réseaux de neurones seuls en commande, on rencontre parfois certaines difficultés dues au manque d'outils théoriques permettant l'analyse préliminaire de la stabilité, ainsi que de méthodes précises de choix des structures convenantes (structure du réseau, nombre optimal de neurones...). Ils sont alors souvent utilisés dans la commande en parallèle avec d'autres régulateurs, de sorte qu'ils effectuent une tâche bien définie comme l'identification ou la supervision.

Chapitre III

Commande adaptative décentralisée par réseaux de neurones

Chapitre III

COMMANDE ADAPTATIVE DECENTRALISEE PAR RESEAUX DE NEURONES

III .1.Introduction :

Les modèles de bras de robots sont fortement non linéaires et variables dans le temps à cause des variations des différents moments d'inertie, le poids de la charge ainsi que la position exacte du centre de masse de l'effecteur. Face à ces obstacles, une commande classique reste impuissante. La commande adaptative est alors la solution la plus adaptée à ce genre de problèmes. Le développement de cette dernière représente un pas important pour la commande très rapide et précise des bras manipulateurs [Seraji 89], [Karakasoglu 93].

Cependant, la commande adaptative standard nécessite l'estimation «on line» de certains termes, ce qui cause une augmentation du temps de calcul. Par ailleurs, certains termes dans le modèle dynamique du robot (les frictions ...) sont non linéaires par rapport aux paramètres, ce qui complique le calcul du régulateur adaptatif.

Pour pallier ces problèmes et réduire le temps de calcul, Sadegh & Horowitz [Sadegh 90] , [Lewis 95] ont proposé une nouvelle approche de commande adaptative appelée ''the desired compensation adaptive law '', soit DCAL.

L'idée est d'utiliser la trajectoire désirée pour le calcul de la commande, ceci en remplaçant les positions et les vitesses réelles pour le calcul de la loi de commande adaptative par les positions et vitesses désirées, de sorte que si la trajectoire désirée est connue à l'avance. La loi adaptative est calculée «off line» sans aucune estimation nécessaire «on line».

A cause de la complexité structurelle des bras manipulateurs, le temps de calcul devient important, on a alors eu recours à la commande décentralisée. Cette commande se répand de plus en plus [Seraji 89], [Lewis 93], [Lewis 95], [Bouhali 96] grâce aux avantages qu'elle présente en commande comme la rapidité (nécessite moins de calcul), simplicité, facilité d'implémentation et des avantages du point de vue de la sécurité.

Le principe de la commande décentralisée est de considérer le système comme un ensemble de plusieurs sous systèmes interconnectés, de sorte que pour chaque sous système sa commande est générée en utilisant uniquement les informations locales, en considérant toutes les interconnexions avec les autres sous systèmes comme des perturbations pour le sous système considéré.

Les réseaux de neurones peuvent être utilisés pour le traitement des systèmes non linéaires, classification de signaux ou encore pour les mémoires associatives. En commande, leur caractère

non linéaire les prédispose à être utilisés en identification pour l'approximation de fonctions non linéaires, ce qui est très intéressant en commande adaptative. Les réseaux de neurones assurent ainsi une simplification de l'implémentation de la commande et une meilleure rapidité.
Cependant, en recherche neuronale, on ne parvient pas toujours à garantir des performances satisfaisantes du système (faible erreur de poursuite et bornage des poids du réseau). Aussi, l'initialisation des poids du réseau est totalement aléatoire. Il est alors souvent nécessaire d'effectuer un apprentissage «off line» du réseau avant de l'implémenter en commande.
Pour pallier tous ces problèmes, on a proposé dans [Lewis 95] un schéma de commande adaptative robuste pour la commande de bras de robots en combinant les théories des RNA et DCAL.
Aucune analyse dynamique préliminaire du bras de robot n'est nécessaire. Le réseau est entraîné directement en temps réel sans aucune phase d'apprentissage préalable ; la stabilité asymptotique globale de l'erreur de poursuite ainsi que le bornage des poids du réseau de neurones sont garantis.
Il est à noter que pour le contrôleur neuronal, on ne peut garantir au mieux que la stabilité locale (bornage des poids du réseau).
Le présent chapitre est organisé en deux parties ; on développe d'abord l'approche centralisée introduit dans [Lewis 95], puis on propose une extension de cette approche au cas décentralisé. Pour cela, on considère que chaque articulation du bras manipulateur est un sous système qui sera commandé indépendamment des autres articulations par une unité de commande locale qui n'a accès qu'aux informations disponibles localement.
Plusieurs réseaux ont été utilisés pour cette commande (des réseaux statiques, des réseaux récurrents « with self feedback », des réseaux récurrents « feedforward NN », des réseaux de fonctions à base radiale et des réseaux dynamiques), et une étude comparative à base de simulations faites sur les bras manipulateurs modélisés au chapitre I est présentée en fin de chapitre.

III .2. Préliminaires mathématiques :

III .2.1. Rappels mathématiques :

Soit A et B des matrices dans $R^{m\times n}$. La norme frobenienne de A est définie par :

$$\|A\|_F^2 = tr\left(A^T A\right) \tag{III.1}$$

où $tr(.)$ est la trace.
De même :

$$\langle AB\rangle_F = tr\left(A^T B\right) \tag{III.2}$$

La norme frobenienne (frobenius norm) est compatible avec la norme 2 de sorte que :

$$\|AX\|_2 \leq \|A\|_F \|X\|_2 \qquad \text{(III.3)}$$

avec : $A \in R^{m \times n}$ *et* $X \in R^n$

III .2.2. Le réseau de neurones :

Pour le cas centralisé, on considère uniquement le cas de l'utilisation d'un réseau de neurones statique.

Soit un réseau de neurones statique à trois couches avec n neurones en entrée, m en sortie et l dans la couche cachée.

La sortie du réseau est définie par :

$$y_i = \sum_{j=1}^{l} \left[w_{ij} \sigma \left[\sum_{k=1}^{n} v_{jk} x_k + \theta_{vj} \right] + \theta_{wi} \right] \qquad i = 1 \ldots m \qquad \text{(III.4)}$$

avec

$\sigma(.)$: la fonction d'activation ;

$V^T = [\, v_{jk}] \in R^{l \times n}$: la matrice des poids d'interconnexion de la couche d'entrée vers la couche cachée ;

$W^T = [w_{ij}] \in R^{m \times l}$: la matrice des poids d'interconnexion de la couche cachée vers la couche de sortie ;

$\theta_v \in R^l$ et $\theta_w \in R^m$: les biais ;

$x \in R^n$ et $y \in R^m$ sont respectivement les vecteur des entrées et des sorties du réseau de neurones.

En augmentant x d'un élément $x_0 = 1$, ($x = [x_0\ x_1 \ldots x_n]$), la première colonne de la matrice V^T représentera alors le vecteur θ_v et la première colonne de la matrice W^T représentera θ_w ; on peut alors réécrire l'équation (III.4) sous forme matricielle :

$$y = W^T \sigma(V^T x) \qquad \text{(III.5)}$$

Une fonction réelle continue $f(x)$ où $x(t) \in R^n$ peut être écrite comme suit :

$$f(x) = W^T \sigma(V^T x) + \varepsilon(x) \qquad \text{(III.6)}$$

où $\varepsilon(x)$ est un vecteur d'erreur de reconstruction de la fonction $f(x)$ par le réseau de neurones.

III .2.3. propriétés des modèles de robots :

Soit un bras manipulateur série à n liaisons rigides ; son modèle peut toujours se ramener à l'écriture suivante :

$$M(q)\ddot{q}+V_m(q,\dot{q})\dot{q}+G(q)+F(\dot{q})+T_d = T \qquad \text{(III.7)}$$

où : $q,\dot{q},\ddot{q}$ dans R^n représentent respectivement les vecteurs de position, vitesse et accélération articulaires ;

$M(q) \in R^{n\times n}$: la matrice d'inertie ;
$G(q) \in R^n$: le vecteur d'effet gravitationnel ;

$V_m(q,\dot{q}) \in R^{n\times n}$: la matrice de coriolis ;
$F \in R^n$: le vecteur de friction ;
$T_d \in R^n$: représente des perturbations additionnelles bornées ;
$T \in R^n$: le vecteur des couples.
Le modèle de l'équation (III.7) satisfait les propriétés suivantes :

Propriété1 :

La matrice d'inertie $M(q)$ est symétrique, définie positive, et satisfait l'équation (III.8).

$$m_1\|y\|^2 \le y^T M(q) y \le m_2\|y\|^2 \qquad \forall y \in R^n \qquad \text{(III.8)}$$

où m_1 et m_2 sont des constantes positives connues et $\|\,.\,\|$ est la norme euclidienne standard.

Propriété 2 :

Les matrices d'inertie et de coriolis (équation III.7) ont la propriété d'antisymetrie suivante :

$$y^T\left(\frac{1}{2}\dot{M}(q)-V_m(q,\dot{q})\right)y = 0 \qquad \forall y \in R^n \qquad \text{(III.9)}$$

où $\dot{M}(q)$ est la dérivée par rapport au temps de la matrice $M(q)$.

Propriété 3 :

On utilise le réseau de neurones pour approximer une fonction fortement non linéaire.

$$f_d(q_d,\dot{q}_d,\ddot{q}_d) = W^T\sigma(V^T x) + \varepsilon(x) \tag{III.10}$$

où
$$f_d(q_d,\dot{q}_d,\ddot{q}_d) = M(q_d)\ddot{q}_d + V_m(q_d,\dot{q}_d)\dot{q}_d + G(q_{dd}) + F(\dot{q}_d) \tag{III.11}$$

$$x = \begin{bmatrix} q_d^T & \dot{q}_d^T & \ddot{q}_d^T \end{bmatrix}^T \tag{III.12}$$

W et V étant constants. On suppose que les position, vitesse et accélération de la trajectoire désirée données respectivement par $q_d,\dot{q}_d,\ddot{q}_d \in R^n$ sont bornées, ce qui garantira le bornage de l'erreur de reconstruction de la fonction $f_d(x)$.

$$\|\varepsilon(x)\| \le \varepsilon_b \tag{III.13}$$

où ε_b est une constante réelle positive.

III .3. Commande adaptative robuste centralisée par RNA des robots : [Lewis 95]

L'objectif de la commande est de réaliser une bonne poursuite de trajectoire pour un bras manipulateur série à n liaisons dont la dynamique, donnée par l'équation (III.7) est mal connue.
On définit le vecteur d'erreur de position $e(t) \in R^n$ ainsi que le vecteur d'erreur filtrée $r(t) \in R^n$ comme suit :

$$e(t) = q_d(t) - q(t) \tag{III.14}$$

$$r = \dot{e} + \Lambda e \tag{III.15}$$

avec

$q_d(t) \in R^n$: le vecteur des positions articulaires de la trajectoire désirée ;
$q(t) \in R^n$: le vecteur des positions articulaires de la trajectoire réelle ;
$\Lambda \in R^{n\times n}$: matrice de gains diagonale définie positive.
En combinant les équations (III.7) et (III.15), on obtient l'équation suivante :

$$M\dot{r} = f(q,\dot{q},q_d,\dot{q}_d,\ddot{q}_d) - V_m r + T_d - T \tag{III.16}$$

la fonction fortement non linéaire $f(q,\dot{q},q_d,\dot{q}_d,\ddot{q}_d)$ est définie par :

$$f(q,\dot{q},q_d,\dot{q}_d,\ddot{q}_d) = M(q)(\ddot{q}_d + \Lambda \dot{e}) + V_m(q,\dot{q})(\dot{q}_d + \Lambda e) + G(q) + F(q) \tag{III.17}$$

On définit également la matrice des poids :

$$Z = diag\{W,V\} \tag{III.18}$$

ainsi que :

$$\begin{cases} \tilde{V} = V - \hat{V} \\ \tilde{W} = W - \hat{W} \\ \tilde{Z} = Z - \hat{Z} \end{cases} \tag{III.19}$$

où $\hat{V},\hat{W}$ obtenus par l'algorithme d'apprentissage du réseau de neurones sont les estimées des valeurs des poids idéaux V et W respectivement.

III .3.1. suppositions et propriétés :

Deux suppositions standards qui sont fréquemment utilisées en robotique et en théorie des réseaux de neurones[Lewis 95] sont considérées :

- Les poids idéaux sont bornés par des constantes positives V_M , W_M , et Z_M .

$$\begin{aligned} \|V\|_F &\le V_M \\ \|W\|_F &\le W_M \\ \|Z\|_F &\le Z_M \end{aligned} \tag{III.20}$$

La trajectoire désirée est bornée.

$$\|x\| \le Q_d \tag{III.21}$$

Où : Q_d est une constante réelle positive et x est défini par l'équation (III.12).
L'erreur de sortie de la couche cachée pour un vecteur d'entrée x est donnée par :

$$\tilde{\sigma} = \sigma - \hat{\sigma} = \sigma(V^T x) - \sigma(\hat{V} x) \tag{III.22}$$

Le développement en série de Taylor de σ dans l'équation (III.22) pour un vecteur d'entrée x est donné par :

$$\sigma(V^T x) = \sigma(\hat{V}^T x) + \sigma'(\hat{V}^T x)\tilde{V}^T x + O(\tilde{V}^T x)^2 \tag{III.23}$$

avec $\sigma' = \left|\frac{d\sigma(z)}{dz}\right|_{z=\hat{z}}$ et $O(z)^2$ correspond aux termes d'ordre élevé.

En notant $\hat{\sigma}' = \sigma'(\hat{V}^T x)$, on obtient :

$$\tilde{\sigma} = \sigma'(\hat{V}^T x)\tilde{V}^T x + O(\tilde{V}^T x)^2 = \hat{\sigma}'\tilde{V}^T x + O(\tilde{V}^T x)^2 \qquad \text{(III.24)}$$

de l'équation (III.24) découle la propriété suivante :

propriété 4 :

pour des fonctions d'activation $\sigma(.)$ telle que $\sigma(.)$ et $\sigma'(.)$ sont bornées (cas des fonctions sigmoidales , tanh , à bases radiales) , les termes de grand ordre dans la série de Taylor, équation (III.24) sont bornés.

$$\left\| O(\tilde{V}^T x)^2 \right\| \le c_1 + c_2 \left\| \tilde{V} \right\|_F \qquad \text{(III.25)}$$

où c_1 , c_2 sont des constantes réelles positives. La preuve de cette propriété est dans[Lewis 95].

III .3.2. structure de la commande :

Soit un réseau de neurones telle que $\varepsilon(x)$ définie dans l'équation (III.10) est bornée (équation (III.13)).

Des propriétés d'approximation des réseaux de neurones et tant que la propriété 3 est satisfaite (x borné), une convergence globale de (III.10) est toujours assurée.

La loi de commande proposée par Lewis [Lewis 95] pour le modèle de l'équation (III.16) est la suivante :

$$T = \hat{W}^T \sigma(\hat{V}^T x) + k_v r + v_R \qquad \text{(III.26)}$$

où $\hat{W}, \hat{V}$ sont les estimés de W et V respectivement (III.10) ; les vecteurs e et r sont définis par les équations (III.14) et (III.15) ; $k_v \in R^{n \times n}$ est une matrice de gains diagonale définie positive ; le terme non linéaire $v_R(t)$ qui sera défini plus tard est utilisé pour compenser les erreurs en vue d'assurer une robustesse aux termes de grand ordre dans les séries de Taylor , aux erreurs de reconstruction de fonctions et aux perturbations T_d bornées. La structure de commande est représentée à la figure (III.1).

x est le vecteur d'entrée du réseau de neurones : $x = (e, \dot{e}, q_d, \dot{q}_d, \ddot{q}_d)$

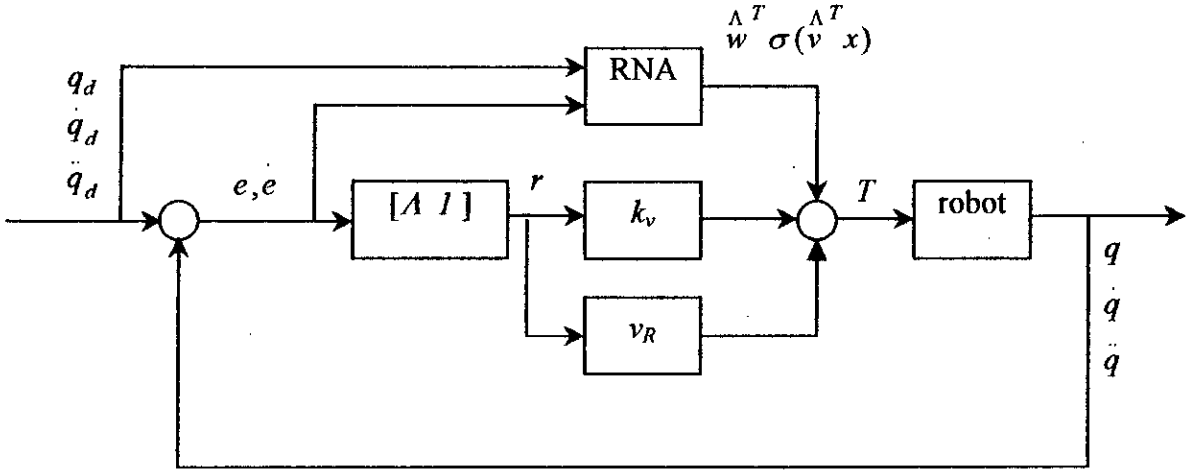

Figure III .1. ***structure d'une commande adaptative robuste des bras de robots***

En remplaçant la loi de commande (III.26) dans l'équation (III.16), on obtient la dynamique du bras de robot en boucle fermée :

$$M\dot{r} = f(q,\dot{q},q_d,\dot{q}_d,\ddot{q}_d) - (k_v + V_m)r - \hat{W}^T\sigma(\hat{V}^T x) - v_R + T_d \qquad \text{(III.27)}$$

En ajoutant et soustrayant $f_d(q_d,\dot{q}_d,\ddot{q}_d)$ défini par l'équation (III.11) et en utilisant la propriété d'approximation des réseaux de neurones (III.10) après développement en série de Taylor de σ (III.24), l'équation (III.27) devient :

$$M\dot{r} = -(k_v + V_m)r + \tilde{W}^T\hat{\sigma} + \hat{W}\hat{\sigma}'\tilde{V}^T x + w - v_R \qquad \text{(III.28)}$$

avec

$$w = f(.) - f_d(.) + \tilde{W}^T\hat{\sigma}'\tilde{V}^T x + W^T O(\tilde{V}^T x)^2 + T_d + \varepsilon(x) \qquad \text{(III.29)}$$

propriété 5 :

Le terme de perturbation $w(t)$ défini dans (III.29) est borné.

$$\|w(t)\| \le \rho(Z,\hat{W},\hat{V}) \qquad \text{(III.30)}$$

telle que la fonction scalaire et positive $\rho(.)$ est définie comme suit :

$$\rho(Z,\hat{W},\hat{V}) = \zeta_0 + \zeta_1\|Z\| + \zeta_2\|Z\|^2 + \zeta_3\left\|\hat{W}\right\|_F + \zeta_4\left\|\hat{V}\right\|_F + \zeta_5\left\|\hat{V}\right\|_F\left\|\hat{W}\right\|_F = S\varphi \qquad \text{(III.31)}$$

où les ζ_i $(i=0...5)$ sont des constantes positives qui dépendent de la trajectoire désirée, des propriétés physiques du robot (masses et longueurs des liaisons, frictions ...), des perturbations et erreurs maximales et de la matrice des gains Λ définie par l'équation (III.15).
Les vecteurs Z,S et φ dans (III.30) et (III.31) sont définis comme suit :

$$\begin{aligned} Z &= \begin{bmatrix} e^T & r^T \end{bmatrix} \\ S &= \begin{bmatrix} 1 & \|Z\| & \|Z\|^2 & \left\|\hat{W}\right\|_F & \left\|\hat{V}\right\|_F & \left\|\hat{W}\right\|_F \times \left\|\hat{V}\right\|_F \end{bmatrix} \\ \varphi &= \begin{bmatrix} \zeta_0 & \zeta_1 & \zeta_2 & \zeta_3 & \zeta_4 & \zeta_5 \end{bmatrix}^T \end{aligned} \qquad \text{(III.32)}$$

Cette propriété est prouvée dans[Lewis 95].
Le terme robuste v_R dans la commande(III.26) est donné par :

$$v_R = \frac{r(S\hat{\varphi})^2}{(S\hat{\varphi})\|r\| + \delta} \qquad \text{(III.33)}$$

avec :

$\dot{\delta} = -\gamma\,\delta$
$\delta(0)$ = constante réelle positive
$\gamma > 0$

Les paramètres de la loi d'adaptation pour φ est donné par :

$$\dot{\hat{\varphi}} = \Gamma S^T\|r\| = -\dot{\tilde{\varphi}} \qquad \text{(III.34)}$$

où Γ est une matrice symétrique définie positive, et $\tilde{\varphi} = \varphi - \hat{\varphi}$. Si la trajectoire désirée est bornée (supposition 2) ; l'algorithme d'apprentissage des poids du réseau de neurones correspondant à la commande développée (III.26), (III.33), (III.34) est le suivant :

$$\begin{aligned} \dot{\hat{W}} &= F\hat{\sigma}r^T \\ \dot{\hat{V}} &= Gx(\hat{\sigma}'^T\hat{W}r)^T \end{aligned} \qquad \text{(III.35)}$$

F et G sont des matrices constantes symétriques définies positives.
Avec la commande développée ci-dessus, équations (III.26), (III.33), (III.34) et (III.35), les erreurs de poursuite $(e,\dot{e})$ tendent vers zéro et les estimées des poids $(\hat{W},\hat{V})$ sont bornées (mais

il est à noter que rien n'assure leur convergence vers les poids idéaux). L'analyse de la stabilité est développée dans [Lewis 95].
Il est intéressant de noter que cette commande est applicable à n'importe quel robot à liaisons rigides sans aucune connaissance nécessaire à priori sur le système (masses et longueurs des différentes liaisons). C'est un grand avantage par rapport à la commande adaptative classique qui requiert une connaissance préliminaire de la dynamique du robot pour établir la loi de commande, ce qui la rend inutilisable pour un autre robot.
Il est aussi très intéressant de voir que le problème de l'initialisation des poids du réseau est complètement écarté ; il suffit de les initialiser tous à 0, le terme robuste et le régulateur proportionnel dérivée stabiliseront le bras de robot le temps que le réseau commence l'apprentissage. On remarquera donc qu'aucun apprentissage «off line» du réseau n'est nécessaire.

III .4. Commande adaptative robuste décentralisée par RNA des robots :

Nous proposons maintenant d'étendre le schéma de commande adaptative robuste par RNA de Lewis au cas décentralisé. Le principe de la décentralisation de la commande consiste à ce que chaque articulation soit commandée séparément en étant considérée comme sous système à part ; sa commande étant calculée d'après les informations locales en ignorant tout de l'état des autres articulations et en considérant les interconnexions avec les autres articulations comme perturbations du sous système (articulation) considéré.

III .4.1. structure de commande :

Considérons un bras manipulateur à n degrés de liberté décrit par l'équation (III.7). La dynamique de la $i^{ème}$ articulation est définie comme suit :

$$m_{ii}\ddot{q}_i + V_{m\,ii}\dot{q}_i + G_i + H_i + T'_{di} = T_i \qquad \text{(III.36)}$$

avec :

$$T'_{di} = T_{di} + \sum_{j\neq i} m_{ij}\ddot{q}_j + \sum_{j\neq i} V_{mij}\dot{q}_j ; \quad i = 1...n \qquad \text{(III.37)}$$

L'équation (III.36) a la même forme que (III.7), il est alors possible d'utiliser la même forme de commande pour le modèle décrit par les équations (III.36) et (III.37) que celle utilisée pour le modèle en commande centralisée (III.7). Le calcul de la commande se fait de la même façon qu'en centralisé.
On propose la structure suivante :

$$m_{ii}\dot{r}_i = f_i(q_i, \dot{q}_i, q_{di}, \dot{q}_{di}, \ddot{q}_{di}) - \frac{1}{2}\dot{V}_{mi} r_i + \dot{T}_{di} - T_i \qquad \text{(III.38)}$$

Où la fonction fortement non linéaire $f_i(q_i, \dot{q}_i, q_{di}, \dot{q}_{di}, \ddot{q}_{di})$ doit être approximée en temps réel et r_i est l'erreur filtrée par articulation où $r_i = \dot{e}_i + \lambda_i e_i$.
De l'équation (III.15), l'équation (III.38) devient :

$$m_{ii}\dot{r}_i = m_{ii}\ddot{q}_{di} - m_{ii}\ddot{q}_i + m_{ii}\lambda_i \dot{e}_i \qquad \text{(III.39)}$$

Or, de l'équation (III.36), on a :

$$m_{ii}\ddot{q}_i = T_i - V_{m\,ii}\dot{q}_i - G_i - H_i - T_{di}' \qquad \text{(III.40)}$$

En remplaçant l'équation (III.40) dans (III.39) et en égalant les deux équations (III.38) et (III.39) on obtient l'expression de la fonction à approximer par le réseau de neurones :

$$f_i(q_i, \dot{q}_i, q_{di}, \dot{q}_{di}, \ddot{q}_{di}) = -\frac{1}{2}\dot{V}_{mii} r_i + H_i + G_i + V_{mii}(\ddot{q}_{di} + \lambda_i \dot{e}_i) + m_{ii}(\dddot{q}_{di} + \lambda_i \ddot{e}_i) \qquad \text{(III.41)}$$

La loi de commande proposée pour la $i^{ème}$ articulation est la suivante :

$$T_i = \hat{W}_i^T \sigma_i(\hat{V}_i^T x_i) + k_{vi} r_i + V_{Ri} \qquad \text{(III.42)}$$

x_i étant le vecteur d'entrée du réseau de neurones de la commande de la $i^{ème}$ articulation.

$$x_i = (e_i, \dot{e}_i, q_{di}, \dot{q}_{di}, \ddot{q}_{di}) \qquad \text{(III.43)}$$

En remplaçant l'expression de la commande dans l'équation (III.38), on obtient la dynamique en boucle fermée pour l'articulation i :

$$m_{ii}\dot{r}_i = f_i(.) - \hat{W}_i^T \sigma_i(\hat{V}_i^T x_i) + V_{Ri} + T_{di}' - (k_{vi} + \frac{1}{2}\dot{V}_{mii}) r_i \qquad \text{(III.44)}$$

En ajoutant et retirant f_{di} (.) au coté droit de l'équation (III.44) et en utilisant les propriétés d'approximation des réseaux de neurones ainsi que le développement en série de Taylor de $\sigma_i(\hat{V}_i^T x_i)$ établie dans l'équation (III.23), on obtient :

$$m_{ii}\dot{r}_i = -(k_{vi} + \frac{1}{2}\dot{V}_{mii}) r_i + \tilde{W}_i^T \hat{\sigma}_i + \hat{W}_i^T \hat{\sigma}_i' \tilde{V}_i^T x_i + w_i - V_{Ri} \qquad \text{(III.45)}$$

où :

$$\omega_i = f_i(.) - f_{di}(.) + \tilde{W}_i^T \hat{\sigma}_i' \tilde{V}_i^T x_i + \tilde{W}_i^T O(\tilde{V}_i^T x_i)^2 + T_{di}' + \varepsilon_i(x_i) \tag{III.46}$$

Le terme ω_i est majoré par :

$$S_i\varphi_i = \zeta_0 + \zeta_1\|Z_i\| + \zeta_2\|Z_i\|^2 + \zeta_3\left\|\hat{W}_i\right\|_F + \zeta_4\left\|\hat{V}_i\right\|_F + \zeta_5\left\|\hat{V}_i\right\|_F\left\|\hat{W}_i\right\|_F \tag{III.47}$$

avec :

$$\begin{aligned} Z_i &= [e_i \quad r_i] \\ S_i &= \left[1 \quad \|Z_i\| \quad \|Z_i\|^2 \quad \left\|\hat{W}_i\right\|_F \quad \left\|\hat{V}_i\right\|_F \quad \left\|\hat{W}_i\right\|_F \times \left\|\hat{V}_i\right\|_F\right] \\ \varphi_i &= [\zeta_0 \quad \zeta_1 \quad \zeta_2 \quad \zeta_3 \quad \zeta_4 \quad \zeta_5]^T \end{aligned} \tag{III.48}$$

Le terme robuste V_{Ri} de la commande de la $i^{ème}$ articulation est donné par :

$$v_{Ri} = \frac{r_i(S_i\hat{\varphi}_i)^2}{(S_i\hat{\varphi}_i)|r_i| + \delta_i} \tag{III.49}$$

où :

$$\begin{aligned} r_i &= e_i + \lambda e_i \\ \dot{\delta}_i &= -\gamma\delta \\ \delta_i(0) &> 0 \text{ et } \gamma > 0 \end{aligned} \tag{III.50}$$

L'algorithme d'adaptation paramétrique de φ_i est :

$$\dot{\hat{\varphi}}_i = \Gamma_i S_i^T |r_i| = -\tilde{\varphi}_i \tag{III.51}$$

III .4.2. Algorithmes d'adaptation des poids des RNA :

Pour chaque articulation, la fonction décrite par l'équation (III.41) sera approximée par un réseau de neurones en utilisant juste les informations disponibles au niveau de cette articulation :

$$y_i = \hat{f}_i(e_i, \dot{e}_i, q_{di}, \dot{q}_{di}, \ddot{q}_{di}) \tag{III.52}$$

L'entrée du réseau de la $i^{ème}$ sera donc le vecteur $x_i = [e_i, \dot{e}_i, q_{di}, \dot{q}_{di}, \ddot{q}_{di}]$ pour générer la sortie y_i.

- **les réseaux statiques :**

La figure (III.2) montre la structure du $i^{ème}$ réseau statique pour l'approximation de la fonction f_i.
La sortie du réseau est :

$$y_i = \hat{f}_i(x_i) = \hat{W}_i^T \hat{\sigma}\left(\hat{V}_i^T x_i\right) \tag{III.53}$$

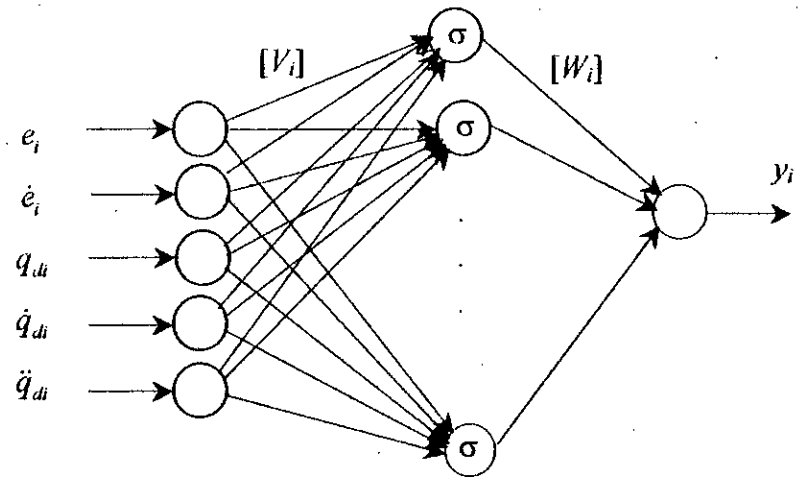

Figure III.2. ***structure du $i^{ème}$ réseau statique***

L'algorithme d'ajustement des poids est donné par :

$$\begin{cases} \dot{\hat{W}}_i = F_i \cdot \hat{\sigma} \cdot r_i^T \\ \dot{\hat{V}}_i = G_i \cdot x_i \cdot (\hat{\sigma}'^T \cdot \hat{W}_i \cdot r_i)^T \end{cases} \tag{III.54}$$

où F_i et G_i sont les pas d'adaptations des matrices poids W_i et V_i respectivement.

- **les réseaux à bases radiales :**

La sortie du réseau à bases radiales à L neurones dans la couche cachée est : (voir II.1.2.b)

$$y_i = \hat{W}_i \cdot S(x_i) \tag{III.55}$$

$$S(x_i) = exp\left(-\frac{1}{2} \sum_{j=1}^{5} \frac{(x_{ij} - c(l,j))^2}{v(l,j)^2} \right) ; \quad l = 1, L \tag{III.56}$$

Pour ce type de réseaux, on garde le même algorithme d'ajustement des poids W_i de sortie de la couche cachée :

$$\dot{\hat{W}}_i = F_i \cdot \hat{\sigma} \cdot r_i^T \tag{III.57}$$

- **les réseaux récurrents** « with self feedback » **:**

En se référant au (§ II.2.c), ce type de réseaux sont régis par l'équation :

$$y_i = \hat{W}_i \hat{\sigma}\left(\hat{V}_i . x_i + \hat{V}_i' . S(x_i)\right) \tag{III.58}$$

l'algorithme d'ajustement des poids est donné par :

$$\begin{cases} \dot{\hat{W}}_i = F_i \cdot \hat{\sigma} \cdot r_i^T \\ \dot{\hat{V}}_i = G_i \cdot x_i \cdot (\hat{\sigma}'^T \hat{W}_i \cdot r_i)^T \\ \dot{\hat{V}}_i' = G_i' \cdot x_i \cdot (\hat{\sigma}'^T \cdot \hat{W}_i \cdot r_i)^T \end{cases} \tag{III.59}$$

où F_i, G_i et G_i' sont les pas d'adaptation des matrices poids W_i, V_i et V_i' respectivement.

- **les réseaux récurrents** « feedforward NN »**:**

Dans ce cas, la sortie du réseau lui sera réinjectée comme une autre entrée (voir II.1.2.c), le vecteur d'entrée est donc : $[x_i \;\; z(y_i)]$ où z est la fonction retard.
le réseau est régi par l'équation :

$$y_i = \hat{W}_i \cdot \hat{\sigma} \cdot \left(\hat{V}_i . [x_i \quad z(y_i)]\right) \tag{III.60}$$

de la même manière, les poids seront ajustés comme suit :

$$\begin{cases} \dot{\hat{W}}_i = F_i . \hat{\sigma} \cdot r_i^T \\ \dot{\hat{V}}_i = G_i \cdot [x_i \quad z(y_i)] \cdot (\hat{\sigma}' \cdot \hat{W}_i \cdot r_i)^T \end{cases} \tag{III.61}$$

avec F_i, G_i et G_i' sont les pas d'adaptation des matrices poids W_i, V_i et V_i' respectivement.

- **les réseaux dynamiques :** [Kosco 92][Karacasoglü 93]

Il est mentionné dans (§.II 1.2.d) que ce type de réseaux nécessite une durée pour l'établissement de son équilibre à chaque fois qu'on lui présente un vecteur d'entrée. Ainsi, une procédure d'échantillonnage est effectuée au niveau des entrées et un filtre de reconstruction sera placé à la sortie de chaque réseau (fig.III.3) (Annexe).

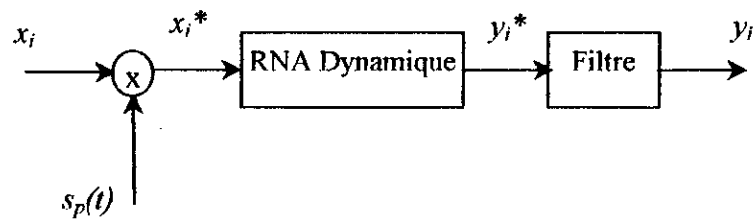

Figure III .3. ***échantillonnage et reconstruction par filtrage.***

Où $s_p(t)$ est le train d'impulsions défini comme suit : $s_p(t) = T \sum_{k=0}^{kT<t} \delta(t - kT)$ (voir annexe).

La dynamique du réseau est régie par les équations :

$$\begin{cases} \dot{E}_i = -E_i + \hat{W}_i \sigma(E) + \hat{V}_i x_i^* \\ y_i^* = \hat{H}_i \overline{E}_i \end{cases} \tag{III.62}$$

avec E_i : vecteur d'état des neurones de la couche cachée.

$\overline{E}_i$: vecteur des états d'équilibre des neurones de la couche cachée.

Pour une période d'échantillonnage T sec, la sortie du réseau sera reconstruite comme suit :

$$y_i(t) = \sum_{k=0}^{kT \le t} y_i^*(k) \, sinc\left[\frac{\pi}{T}(t - kT)\right] \tag{III.63}$$

les réseaux dynamique utilisés contiennent chacun 5 entrées, L neurones dans la couche cachée et une sortie. L'ajustement des poids se fait à chaque période d'échantillonnage selon l'algorithme suivant:

$$H_i(l) = H_i(l) - \mu_h \cdot r_i \cdot \overline{E}_i(l) \qquad l = 1,L \tag{III.64}$$

$$W_i(l,j) = \begin{cases} W_i(l,j) - \mu_w H_i(l) \cdot r_i \cdot \hat{\sigma}_i(\overline{E}_i(l)) & l,j = 1,L \quad si \quad l \neq j \\ -1 \qquad si \qquad l = j \end{cases} \tag{III.65}$$

$$V_i(l,j) = V_i(l,j) - \mu_v \cdot H_i(l) \cdot r_i \cdot x_i^*(l) \quad , \quad l = 1,L \quad j = 1,5 \tag{III.66}$$

III .4.3. Démonstration de la stabilité :

On prouvera la stabilité dans le cas général d'un réseau de neurones dont l'expression de la sortie est donnée par l'équation (III.5) et pour lequel l'algorithme d'adaptation paramétrique est donné par l'équation (III.35).

En effet, pour les réseaux récurrents, il suffit d'augmenter les matrices W et V de sorte à avoir la même écriture pour la sortie du réseau que celle pour le réseau statique (équation III.5), et on peut alors appliquer le même algorithme d'adaptation des poids.

Cette procédure est applicable au réseau dynamique après avoir supposé que le filtre est idéal et qu'à chaque instant, $\overline{E}_i$ est la solution de l'équation différentielle définie dans (III.62), ceci aux instants d'échantillonnage et entre les instants d'échantillonnage de sorte que les équations du réseau (équation III.62) puissent avoir la même forme d'écriture que celle du réseau statique (équation III.5). Cette supposition peut être vérifiée car on a choisi une période d'échantillonnage de *0.001 sec* , ce qui est relativement faible par rapport à la dynamique du bras de robot et donc les valeurs de $\overline{E}_i$ à deux instants d'échantillonnage successifs sont très proches, il est donc possible que la reconstruction de la sortie du réseau soit parfaite.

Considérons la fonction de Lyapunov suivante :

$$L_i = \frac{1}{2} m_{ii} r_i^2 + \frac{1}{2} tr(\tilde{W}_i^T F_i^{-1} \tilde{W}_i) + \frac{1}{2} tr(\tilde{V}_i^T G_i^{-1} \tilde{V}_i) + L_{Ri} \tag{III.67}$$

où :

$$L_{Ri} = \frac{1}{2} \tilde{\varphi}_i^T \Gamma_i^{-1} \tilde{\varphi}_i + \frac{\delta_i}{\gamma_i} \tag{III.68}$$

En dérivant l'équation (III.67) et en utilisant (III.45), on obtient :

$$\dot{L}_i = -k_{vi}r_i^2 + \frac{1}{2}r_i(\dot{m}_{ii} - \dot{V}_{mii}) - r_i V_{Ri} - \delta_i - |r_i| S_i \tilde{\varphi}_i + r_i w_i + tr(\tilde{W}_i^T F^{-1}\dot{\tilde{W}}_i + \tilde{W}_i^T \hat{\sigma}_i' r_i^T) + tr(\tilde{V}_i^T G_i^{-1}\dot{\tilde{V}}_i + \tilde{V}_i^T x_i(\hat{\sigma}_i'^T \hat{W}_i r_i)^T \quad \text{(III.69)}$$

Il est facilement vérifiable sur les deux modèles de bras de robots utilisés dans ce travail (SCARA et PUMA 560) que :

$$\forall i \quad \dot{V}_{mii} = \dot{m}_{ii} \quad \text{(III.70)}$$

Cependant, une généralisation de l'équation (III.70) à tout les modèles de bras de robot n'est pas prouvée.

De l'équation (III.46), et sachant que $\tilde{\varphi}_i = \varphi_i - \hat{\varphi}_i$, on déduit que :

$$r_i w_i - |r_i| S_i \varphi_i \le 0 \quad \text{(III.71)}$$

Il résulte que :

$$\begin{cases} \dot{L}_i \le -\frac{1}{2}k_{vi}r_i^2 + \dot{L}_1 + \dot{L}_2 \\ \dot{L}_1 = |r_i| S_i \hat{\varphi}_i - r_i V_{Ri} - \delta_i \\ \dot{L}_2 = tr(\tilde{W}_i^T F^{-1}\dot{\tilde{W}}_i + \tilde{W}_i^T \hat{\sigma}_i' r_i^T) + tr(\tilde{V}_i^T G_i^{-1}\dot{\tilde{V}}_i + \tilde{V}_i^T x_i(\hat{\sigma}_i'^T \hat{W}_i r_i) \end{cases} \quad \text{(III.72)}$$

En remplaçant V_{Ri} par son expression dans (III.49) on trouve :

$$\dot{L}_1 = \frac{\delta_i |r_i| (S_i \hat{\varphi}_i)^2}{(S_i \hat{\varphi}_i)|r_i| + \delta_i} - \delta_i \le 0 \quad \text{(III.73)}$$

Sachant que $\tilde{W} = W - \hat{W}, \tilde{V} = V - \hat{V}$ et en appliquant les lois d'adaptation paramétrique (III.35) dans l'expression de $\dot{L}_2$, on trouve aisément :

$$L_2 = 0 \tag{III.74}$$

des équations (III.72) , (III.73) , (III.74) on déduit que :

$$\dot{L}_i \leq 0 \tag{III.75}$$

L'équation (III.75) implique que L_i ,r_i , W_i , V_i et φ_i sont tous bornés et e_i , $\dot{e}_i$ tendent tout deux vers 0.

Il est à noter que pour le cas d'un réseau RBF, cette preuve de stabilité n'est plus valable, en raison de l'algorithme d'adaptation paramétrique qui est tout à fait différent de celui considéré dans l'analyse de la stabilité établie ci-dessus.

III .5. Résultats de simulation et interprétation des résultats :

Pour valider le schéma de commande proposé, des simulations ont été effectuées sur deux modèles de bras de robots : PUMA 560 (trois degrés de liberté) et SCARA (deux degrés de liberté) précédemment présentés (chapitre I).

La simulation a entièrement été faite sur le logiciel ' Matlab 4.2 b et 4.2 c'. La résolution numérique des systèmes d'équations différentielles non linéaires a été faite par la méthode de Runge Kutta d'ordre quatre avec un pas d'échantillonnage du temps constant *dt =0.001 sec.*

Les résultats de plusieurs tests de robustesse de la commande aux perturbations externes agissant sur les bras de robots sont également présentés.

a . Le bras de robot PUMA 560 :

Nous avons choisi pour le robot PUMA 560 le test de LEAHVY , test propre au robot PUMA qui consiste en une trajectoire cycloïdale qui excite toute la dynamique du bras. Les articulations se déplacent respectivement de la position {-50° , -135° , 135°} à la position {45° , -85° , 30° }en un temps de mouvement de *1,5 sec* (voir chapitre 1).

Les paramètres de commande sont choisis comme suit :

γ_i=.5, $\delta_i(0)$ =100 k_{vi} = 200, λ_i= 10, i=1...3

Ils seront les même pour tout les réseaux.

Les paramètres des différents réseaux utilisés sont :

- ***Réseaux statiques :***

Les paramètres des réseaux sont :

F1= 0.05, *G1= 0.001*

F2= 0.5, *G2= 0.001*

F3= 0.5, *G3= 0.001*

Chaque réseau est à trois couches, la couche cachée contient dix neurones et les fonctions d'activation sigmoïdales sont données par :

$$\hat{\sigma}_j = \frac{5}{1+exp(-0.1\times j\times net_j)} \qquad j = 1....10 \qquad \text{(III.76)}$$

où : *net* est le vecteur de sortie de la couche cachée.

Les poids initiaux ont tous été pris nuls, ce qui ne pose pas de problèmes à l'instant initial car les terme robuste et proportionnel dérivée agissent le temps que le réseau commence l'apprentissage.

Le vecteur d'entrée du $i^{ème}$ réseau est: $x_i = (e_i, e_i, q_{di}, q_{di}, q_{di})$

- ***Réseaux récurrents 1:*** *« with self feedback »*

Les paramètres des réseaux sont :

F1= 0.05,	*G1= 0.01,*	*G1'=0.02*
F2= 0.5,	*G2= 0.01,*	*G2'=0.02*
F3= 0.5,	*G3= 0.01,*	*G3'=0.02*

Chaque réseau est à trois couches, la couche cachée contient dix neurones, les fonctions d'activation sigmoïdales sont données par l'équation (III.76)et les poids initiaux sont tous nuls.

Le vecteur d'entrée du $i^{ème}$ réseau est: $x_i = (e_i, e_i, q_{di}, q_{di}, q_{di})$

- ***Réseaux récurrents 2 :*** *« feedforward NN »*

Les paramètres des réseaux sont :

F1= 0.05,	*G1= 0.001*
F2= 0.5,	*G2= 0.001*
F3= 0.5,	*G3= 0.001*

Chaque réseau est à trois couches, la couche cachée contient dix neurones, les fonctions d'activation sigmoïdales sont données par l'équation (III.76)et Les poids initiaux des réseaux ont tous été pris nuls.

Le vecteur d'entrée du $i^{ème}$ réseau est: $x_i = (e_i, e_i, q_{di}, q_{di}, q_{di}, y_i)$

- ***Réseaux à bases radiales (RBF):***

Les réseaux sont à trois couches. on a utilisé pour chaque réseau cinq centres fixes, et une variance constante v^2 égale à $\frac{1}{5\pi}$.

On subdivise les espaces normalisés de chacune des 5 entrées en 5 sous espaces réguliers. Ce qui donne un nombre de 5^5=3125 neurones au niveau de la couche cachée de chacun des trois réseaux.

Les paramètres des réseaux sont :

$F1 = 1$ $F2 = 1$ $F3 = 1$

Les poids initiaux des réseaux sont :

$W_1 = 0$, $W_2(i, j) = -0.5$, $W_3(i, j) = 0.5$

Le vecteur d'entrée du $i^{ème}$ réseau est: $x_i = (e_i, \dot{e}_i, q_{di}, \dot{q}_{di}, \ddot{q}_{di})$

- ***Réseaux dynamiques :***

Les paramètres des réseaux sont :

$\mu_{h1} = 0.005$ $\mu_{v1} = 0.001$ $\mu_{w1} = 5 \times 10^{-6}$

$\mu_{h2} = 0.05$ $\mu_{v2} = 0.001$ $\mu_{w2} = 5 \times 10^{-6}$

$\mu_{h3} = 0.05$ $\mu_{v3} = 0.001$ $\mu_{w3} = 5 \times 10^{-6}$

Les réseaux sont à trois couches ; tous les poids sont pris nuls à l'instant initial et les fonctions d'activation sont des sigmoïdes données par l'équation (III.76).

Le vecteur d'entrée du $i^{ème}$ réseau est: $x_i = (e_i, \dot{e}_i, q_{di}, \dot{q}_{di}, \ddot{q}_{di})$

- **Essai à vide :**

- ***Réseaux statiques :*** (figure III.4a)

Les erreurs de poursuite sont faibles, la commande est oscillatoire au début mais reste physiquement réalisable.

- ***Réseaux récurrents 1*** *« with self feedback » :* (figure III.4b)

Les performances de poursuite restent sensiblement les mêmes que pour le cas de l'utilisation de réseaux statiques.

- ***Réseaux récurrents 2*** *« feedforward RNN » :* (figure III.4c)

Les performances de poursuite restent pratiquement les mêmes, mais les commandes sont légèrement plus oscillatoires et d'ordre plus élevé pour les première et deuxième articulations.

- ***Réseaux RBF:*** (figure III.4d)

Les erreurs de poursuite des deuxième et troisième articulations sont plus grandes que lors de l'utilisation des réseaux précédents mais restent acceptables, ceci est probablement dû au nombre réduit de centres (cinq), ce qui n'assure pas une très bonne approximation du réseau de la fonction non linéaire f_i , mais un compromis était à faire, car en augmentant le nombre de centres, l'erreur de validation augmente, et comme l'apprentissage se fait «on line», l'erreur de validation devient l'erreur d'apprentissage du réseau, erreur qui s'accumule dans le temps pour donner les résultats obtenus.
Les commandes quant à elles sont plus douces et d'ordre plus faible que précédemment, ce qui est dû à l'ordre de l'erreur de poursuite toléré.

- ***Réseaux dynamiques:*** (figure III.4e)

Les performances de poursuite restent sensiblement les mêmes que lors de l'utilisation des réseaux statique et récurrents.
Les commandes sont oscillatoires et du même ordre que pour les réseaux statiques.

- **Essai avec lâcher de charge :**

Pour ce test, l'effecteur contient une charge de 4kg. On ajoute une perturbation au robot qui consiste à ce que la masse tombe accidentellement après 1.5 sec.
à faire tomber la masse après 0.75 sec.
Les résultats de la poursuite de trajectoire pour les différents réseaux de neurones utilisés sont présentés aux figures (III.5a , III.5b , III.5c , III.5d et III.5e). Les erreurs de poursuite pendant la charge sont sensiblement les mêmes pour les première et deuxième articulations, mais légèrement plus grandes pour la troisième articulation, ce qui est dû à l'effet de la charge. La commande de la troisième articulation se trouve alors augmentée.
Les commandes agissent au moment du lâcher de la masse pour contrer la perturbation, ce qui prouve le caractère adaptatif et robuste de la commande.
Toutes les remarques données lors de l'essai à vide restent vérifiées pour ce test.

- **Essai avec rupture de commande :**

Pour prouver l'efficacité de la décentralisation de la commande, on propose un test de rupture d'une commande au niveau d'une articulation ; on a opté pour la deuxième articulation, ce qui semble être un cas significatif.
Les résultats de simulation (figures III.6a , III.6b , III.6c , III.6d , III.6e) montrent que malgré la divergence de la deuxième articulation due à la rupture de commande à 0.75 sec, les deux autres articulations continuent à suivre leur trajectoire désirée en réagissant à la rupture de la deuxième commande comme simple perturbation.

Les mêmes remarques que celles données lors de l'essai à vide sont à faire, mais il est à noter que la troisième commande avec réseau dynamique, récurrent ou RBF est bien plus lisse et douce que lors de l'utilisation de réseaux statiques.

- **Etude comparative :**

Dans ce tableau sont présentées les erreurs maximales (en degrés) de chaque articulation du robot PUMA 560 pour les tests effectués précédemment, ainsi qu'une qualification des commandes correspondantes.

N° art	Test	Réseau statique	Réseau récurrent 1	Réseau récurrent 2	Réseau gaussien (RBF)	Réseau dynamique
1	à vide	0.062	0.021	0.021	0.034	0.062
	Décharge	0.023	0.023	0.022	0.038	0.066
	Rupture de la 2ème cde	0.346	0.334	0.318	0.171	0.541
2	à vide	0.075	0.070	0.069	0.465	0.053
	Décharge	0.061	0.058	0.059	0.384	0.047
3	à vide	0.020	0.022	0.022	0.435	0.032
	Décharge	0.056	0.059	0.059	0.809	0.048
	Rupture de la 2ème cde	0.134	0.022	0.022	0.435	0.162
Commandes		Oscilla-toire	Oscilla-toire	Oscilla-toire	Douce	moyennement douce

Tableau III .1. ***étude comparative des différents réseaux pour la commande adaptative robuste du bras de robot PUMA 560***

Le choix du réseau de neurones approprié pour la commande adaptative du bras de robot PUMA 560 est conditionné par le cahier des charges. En effet, si ce dernier tolère des erreurs de poursuite allant jusqu'à 1 degré, le meilleur candidat serait le réseau RBF car il présente des commandes douces ; mais si le robot est utilisé dans une application où une plus grande précision est nécessaire, on optera plutôt pour le réseau dynamique car il donne lieu à des commandes plus douces que celles obtenues avec les réseaux statiques et récurrents ; Cependant, si les moteurs utilisés ont des dynamiques faibles (constantes de temps faibles) , de plus faibles erreurs de poursuite peuvent être obtenues en utilisant les réseaux récurrents.

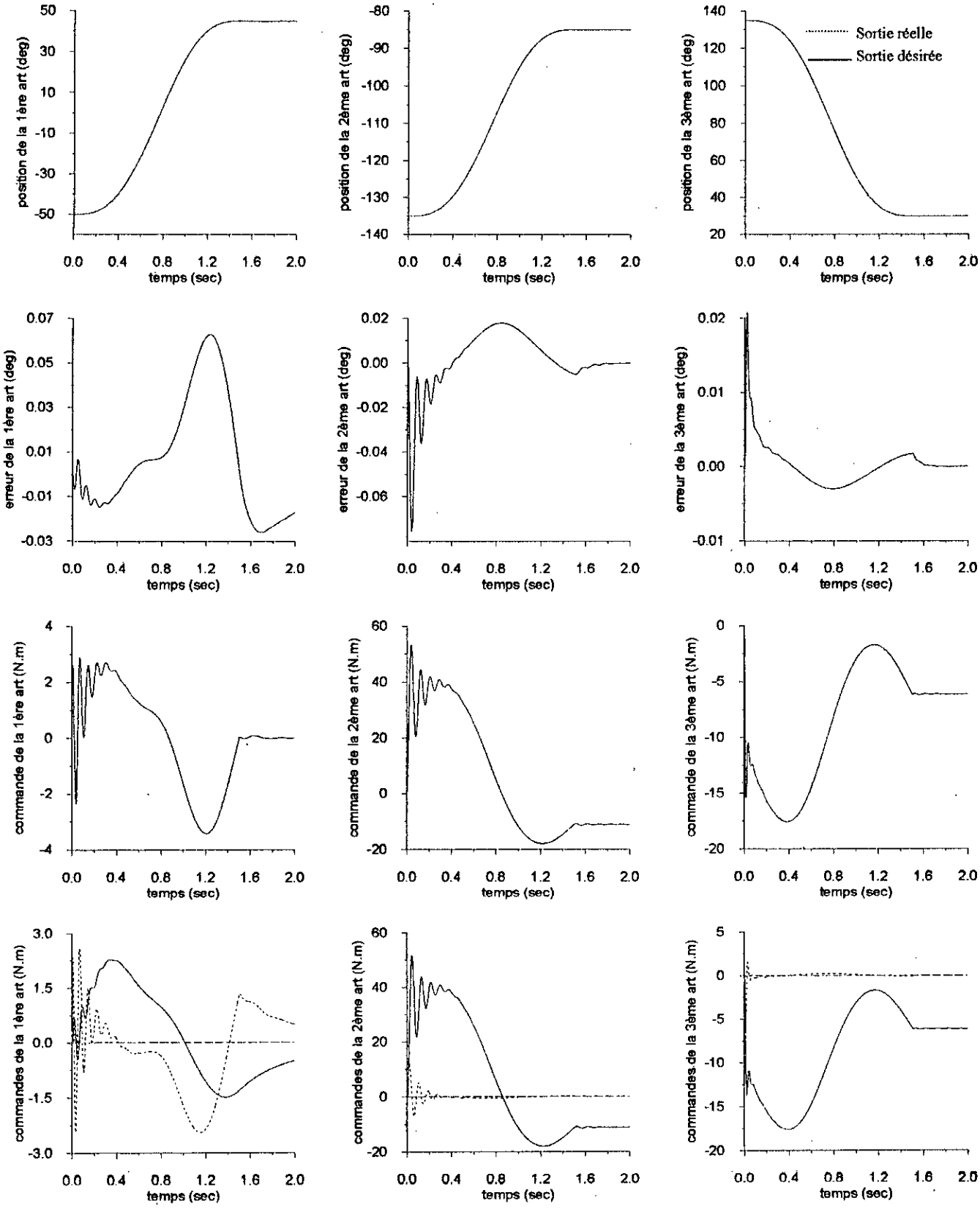

Figure III .4a . ***Poursuite à vide du robot puma 560 (trajectoire de LEAHVY)***
- réseau statique-

——— ***sortie du réseau***
····· ***terme robuste***
—·—·— ***sortie du PD***

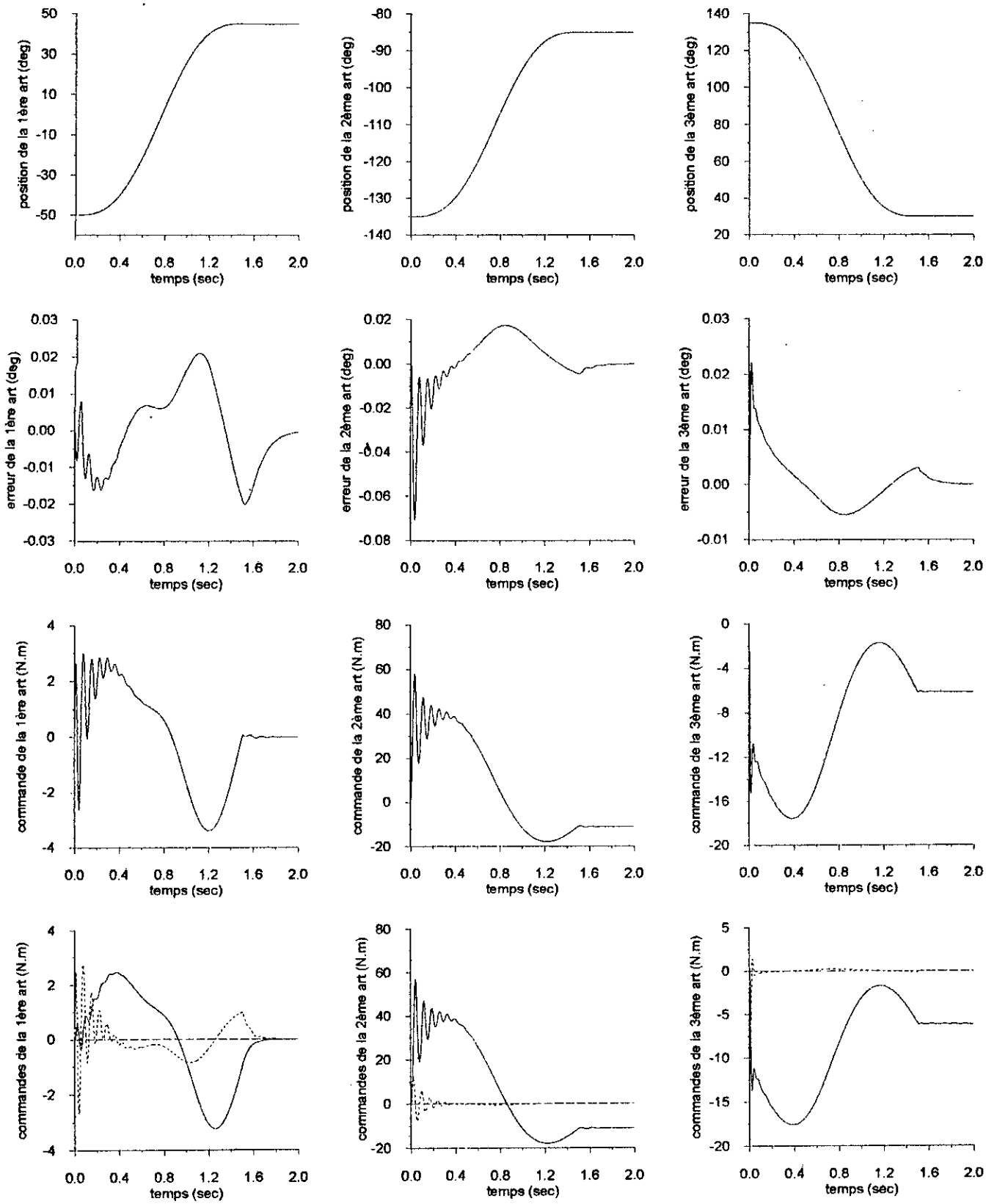

Figure III .4b. ***Poursuite à vide du robot puma 560 (trajectoire de LEAHVY) - réseau réccurent 1 –***

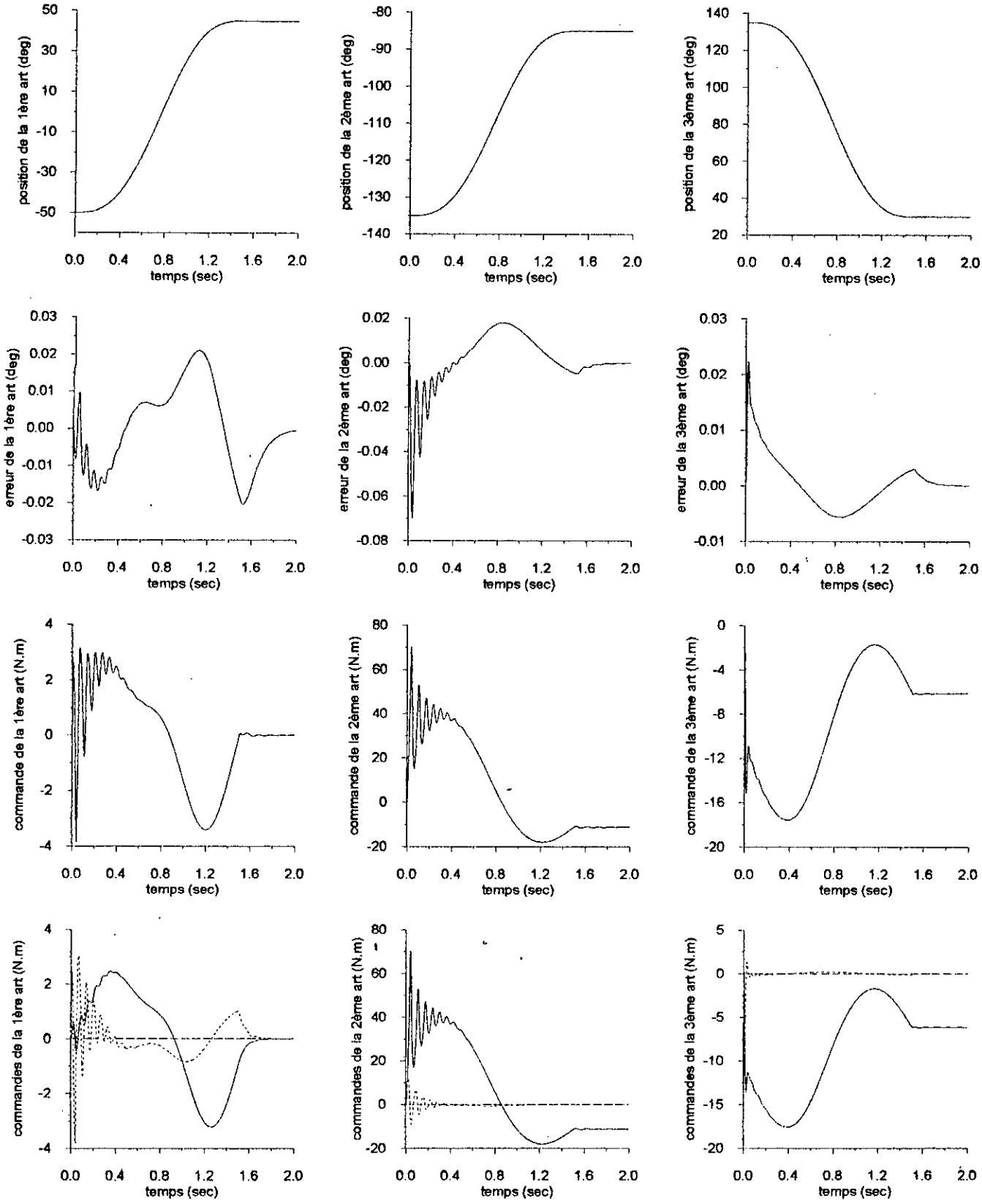

Figure III .4c. ***Poursuite à vide du robot puma 560 (trajectoire de LEAHVY) - réseau réccurent 2 -***

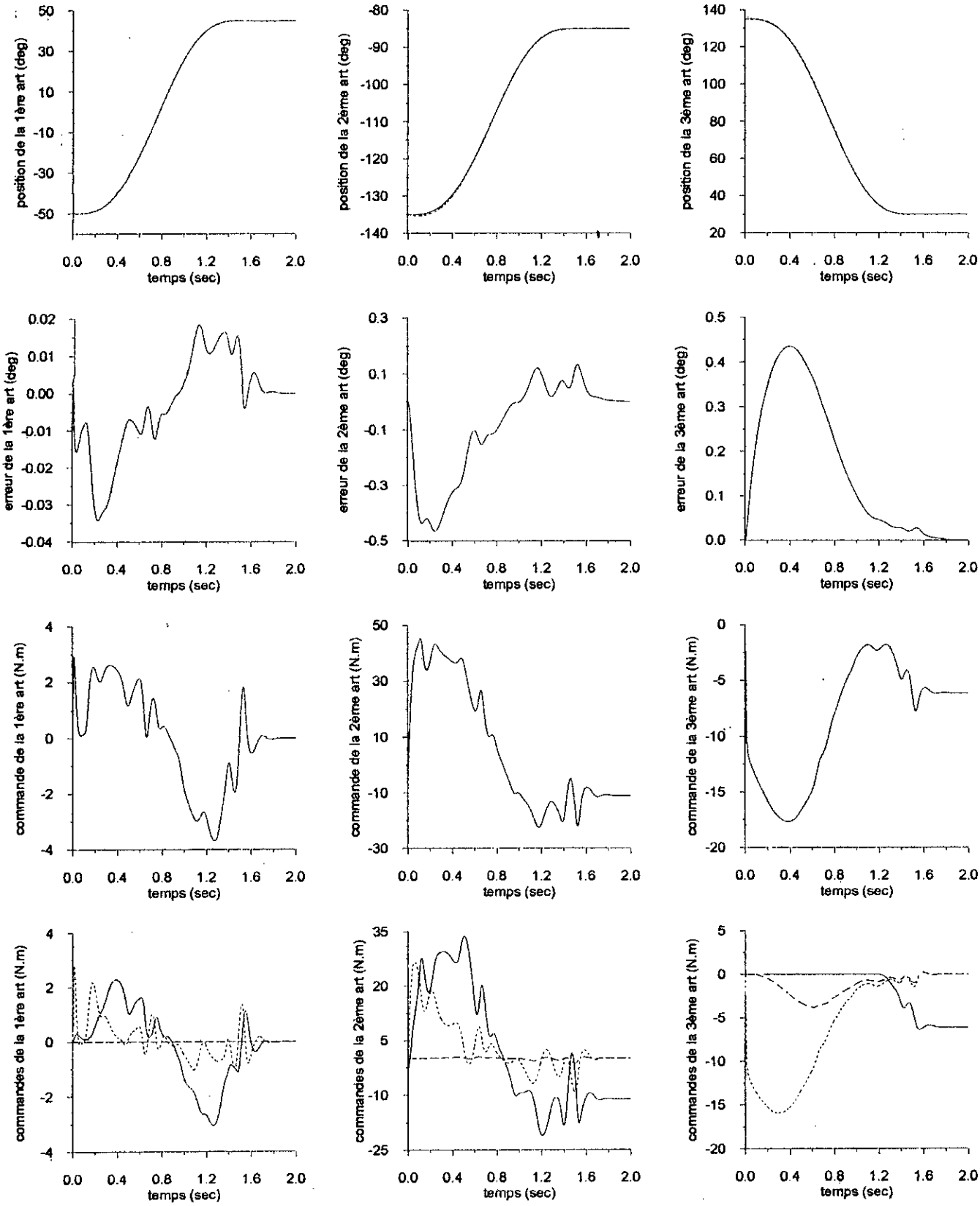

Figure III .4d . ***Poursuite à vide du robot puma 560 (trajectoire de LEAHVY) - réseau RBF -***

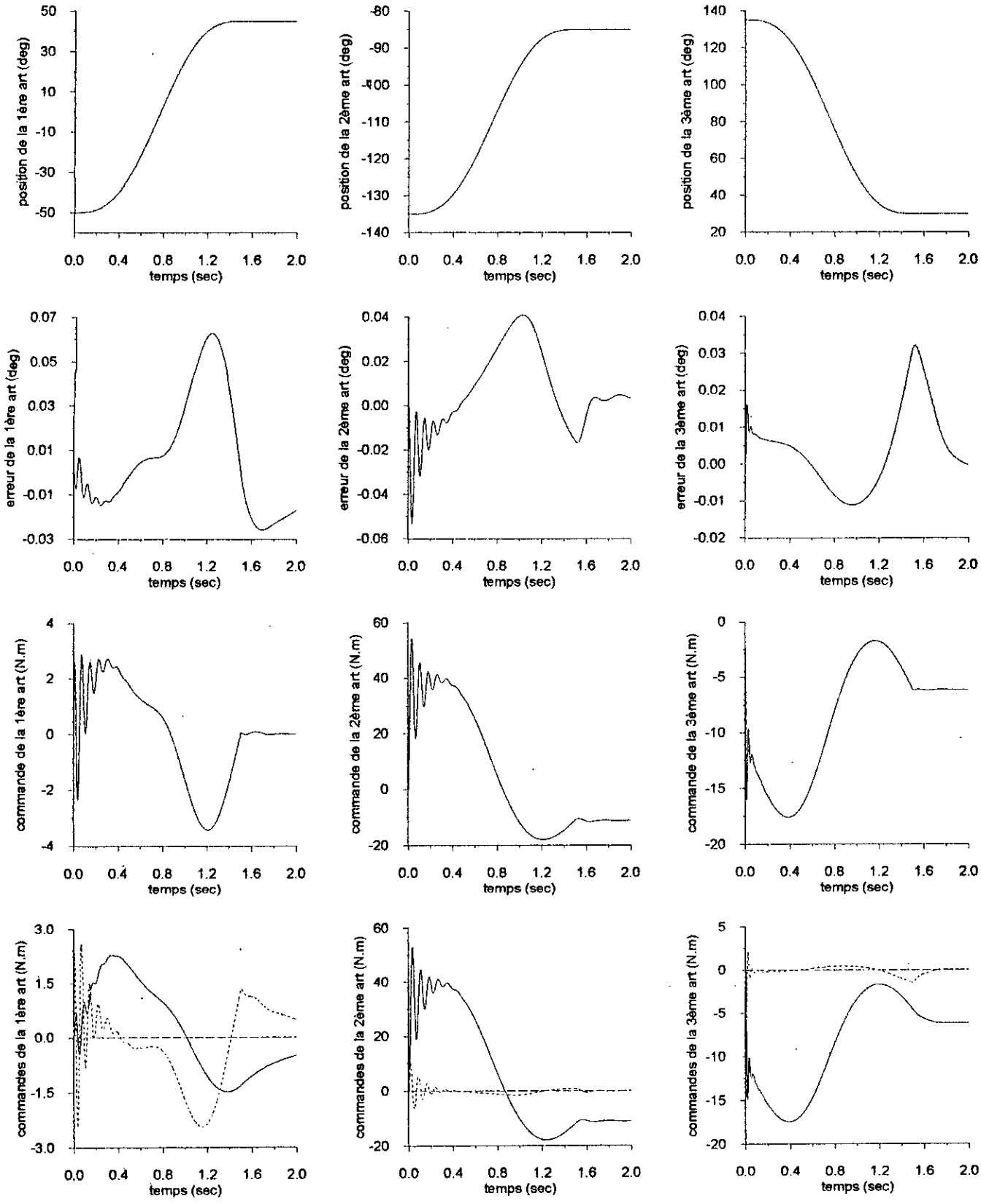

Figure III .4e. ***Poursuite à vide du robot puma 560 (trajectoire de LEAHVY) - réseau dynamique -***

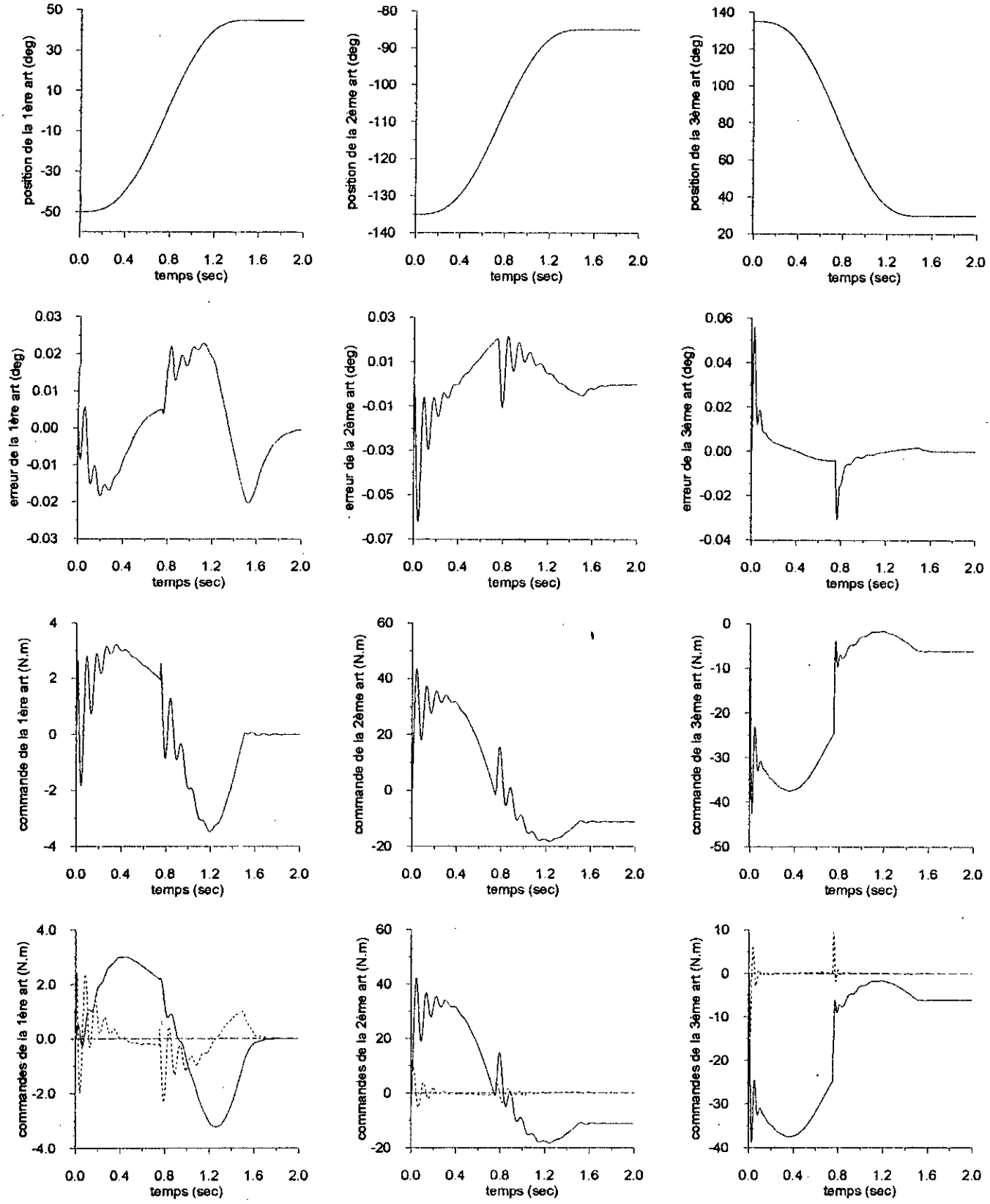

Figure III .5a . ***Poursuite du robot puma 560 avec lâcher de la charge(4kg) après 0.75 sec (trajectoire de LEAHVY) - réseau statique -***

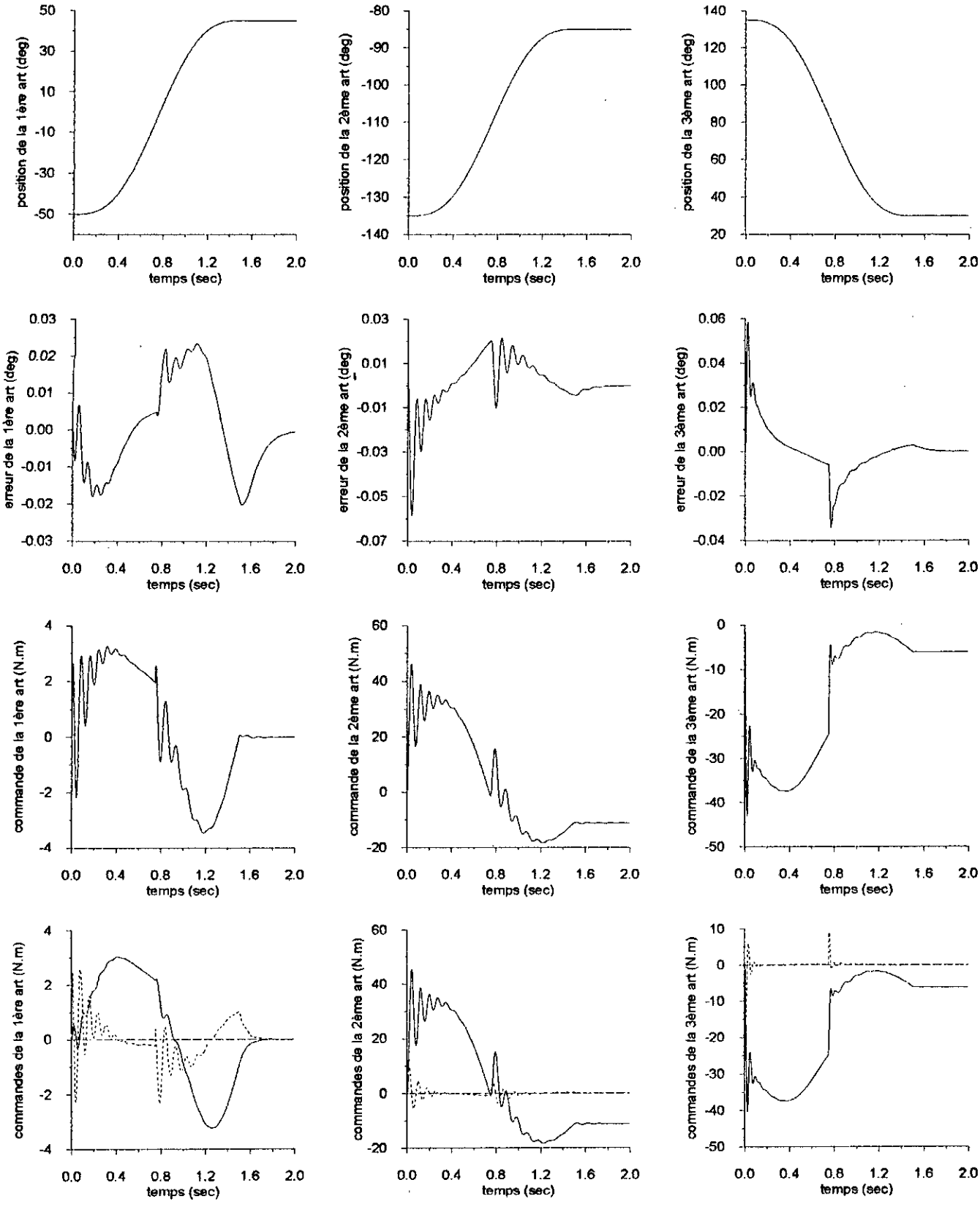

Figure III .5b. ***Poursuite du robot puma 560 avec lâcher de la charge(4kg) après 0.75 sec (trajectoire de LEAHVY) - réseau récurrent 1 -***

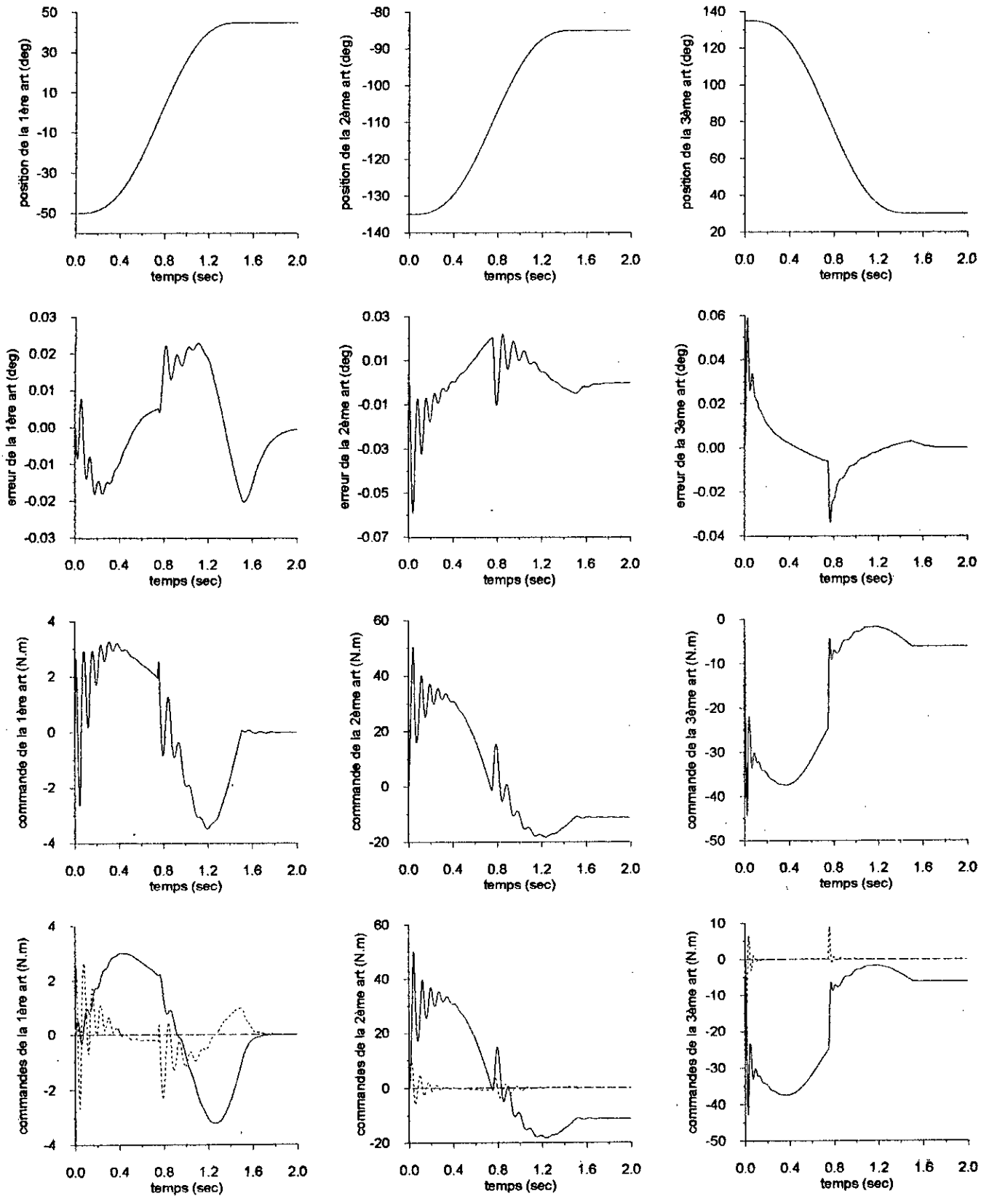

Figure III .5c. ***Poursuite du robot puma 560 avec lâcher de la charge(4kg) après 0.75 sec(trajectoire de LEAHVY) - réseau récurrent 2 -***

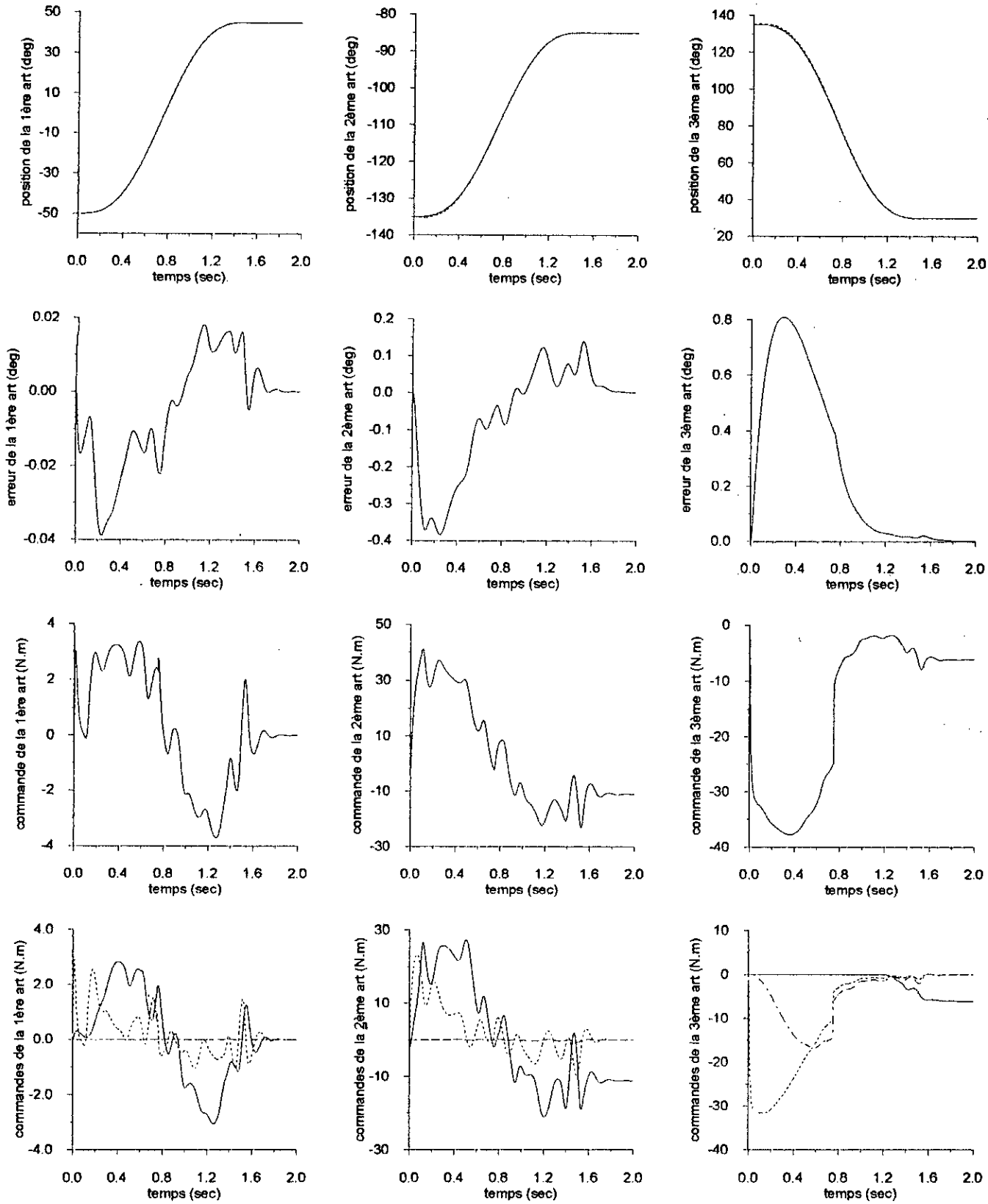

Figure III .5d . ***Poursuite du robot puma 560 avec lâcher de la charge(4kg) après 0.75 sec (trajectoire de LEAHVY) - réseau RBF-***

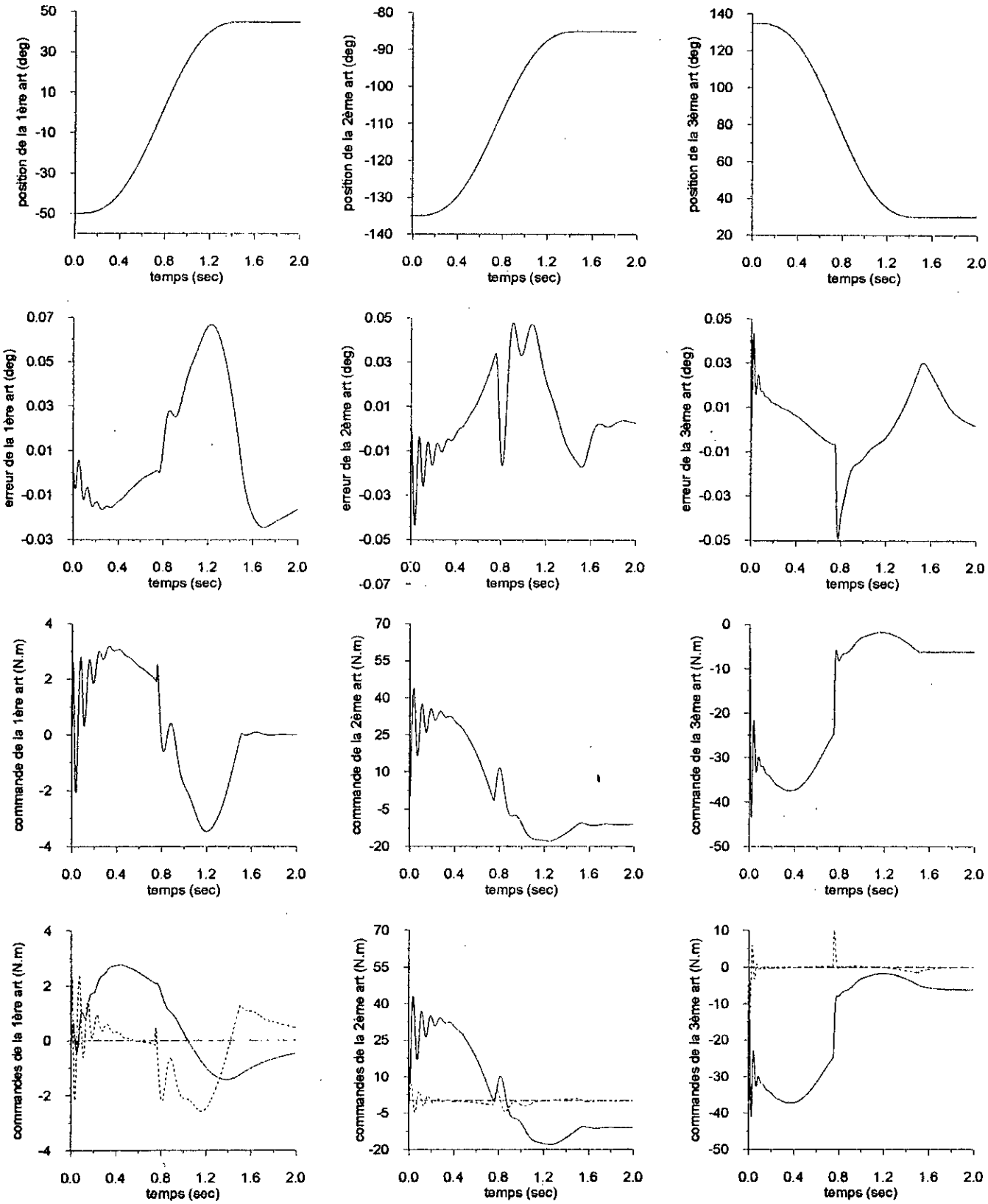

Figure III .5e. ***Poursuite du robot puma 560 avec lâcher de la charge(4kg) après 0.75 sec (trajectoire de LEAHVY) - réseau dynamique-***

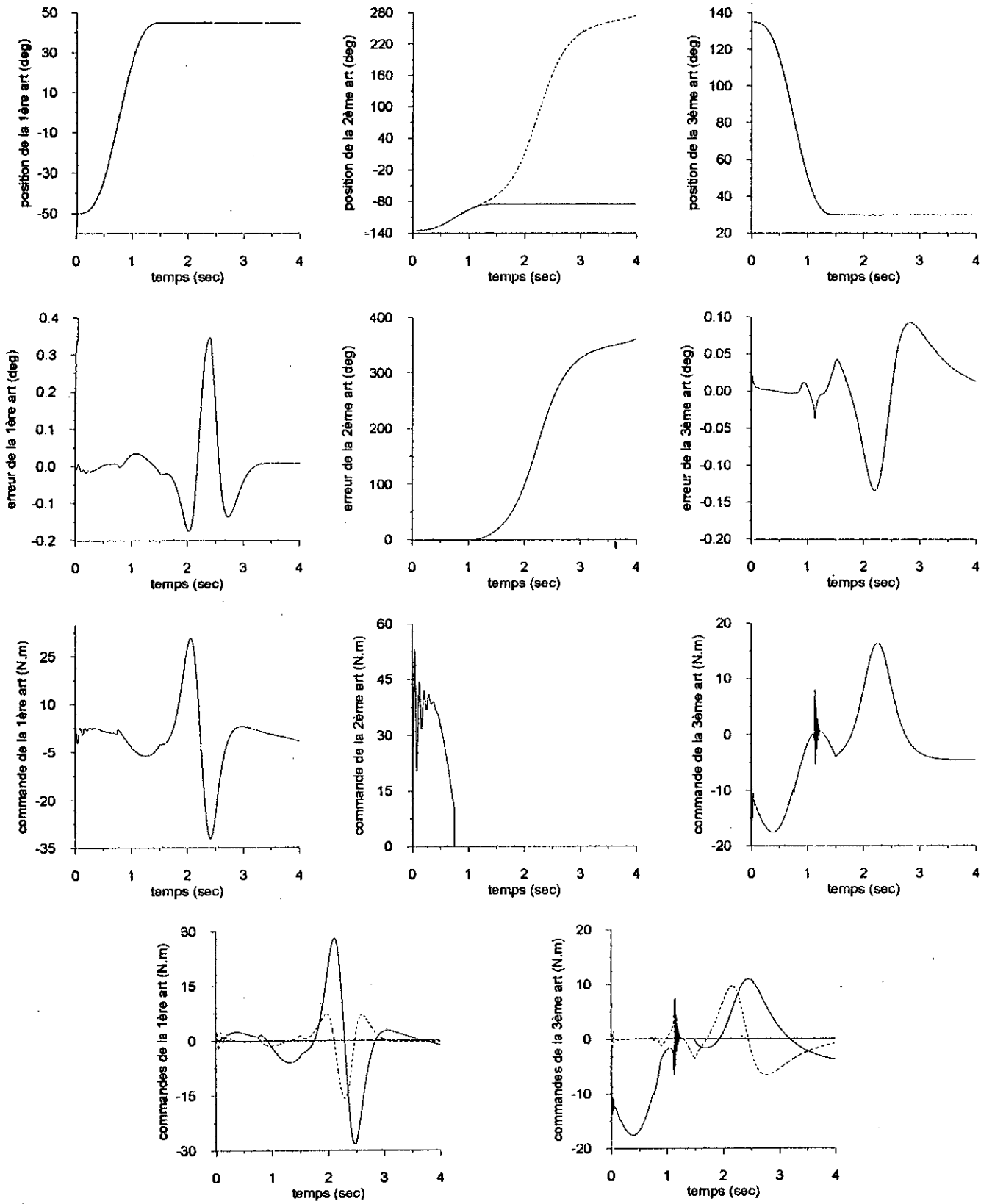

Figure III .6a. ***poursuite du robot puma 560 avec rupture de la 2ème commande à t = 0.75 sec (trajectoire de LEAHVY) - réseau statique -***

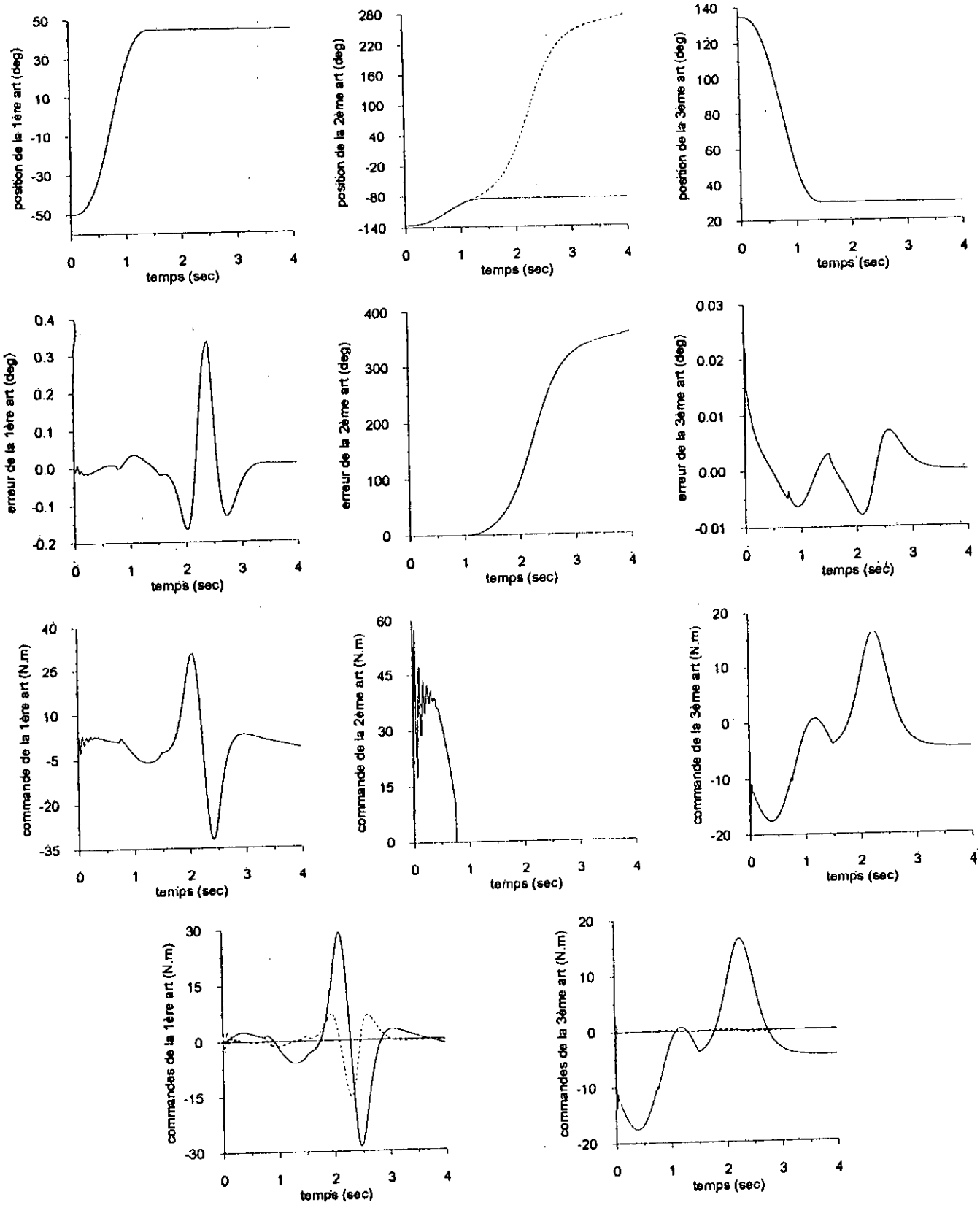

Figure III .6b. ***poursuite du robot puma 560 avec rupture de la 2ème commande à t = 0.75 sec (trajectoire de LEAHVY) - réseau récurrent 1 -***

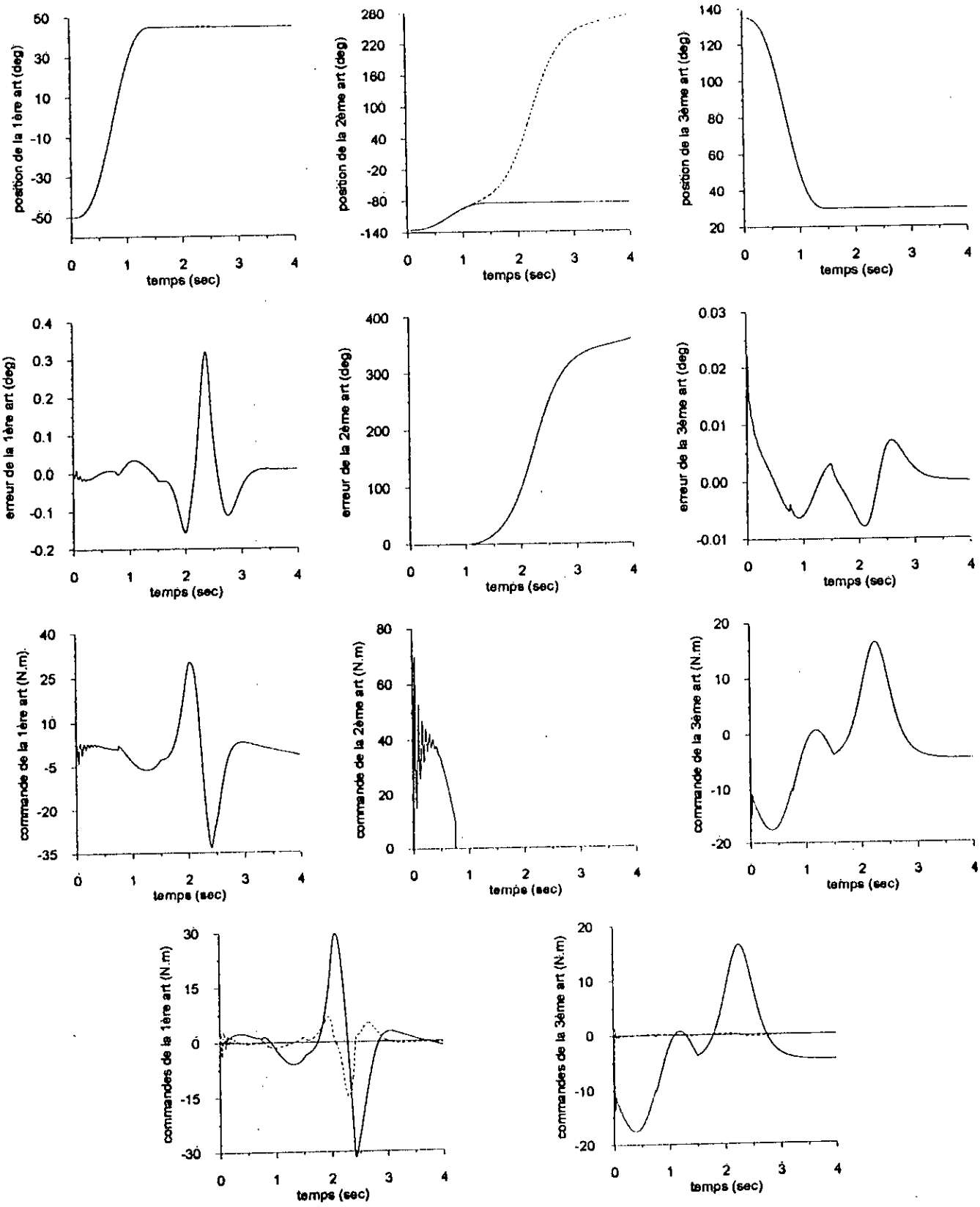

Figure III .6c. ***poursuite du robot puma 560 avec rupture de la 2ème commande à t = 0.75 sec (trajectoire de LEAHVY) - réseau récurrent 2 -***

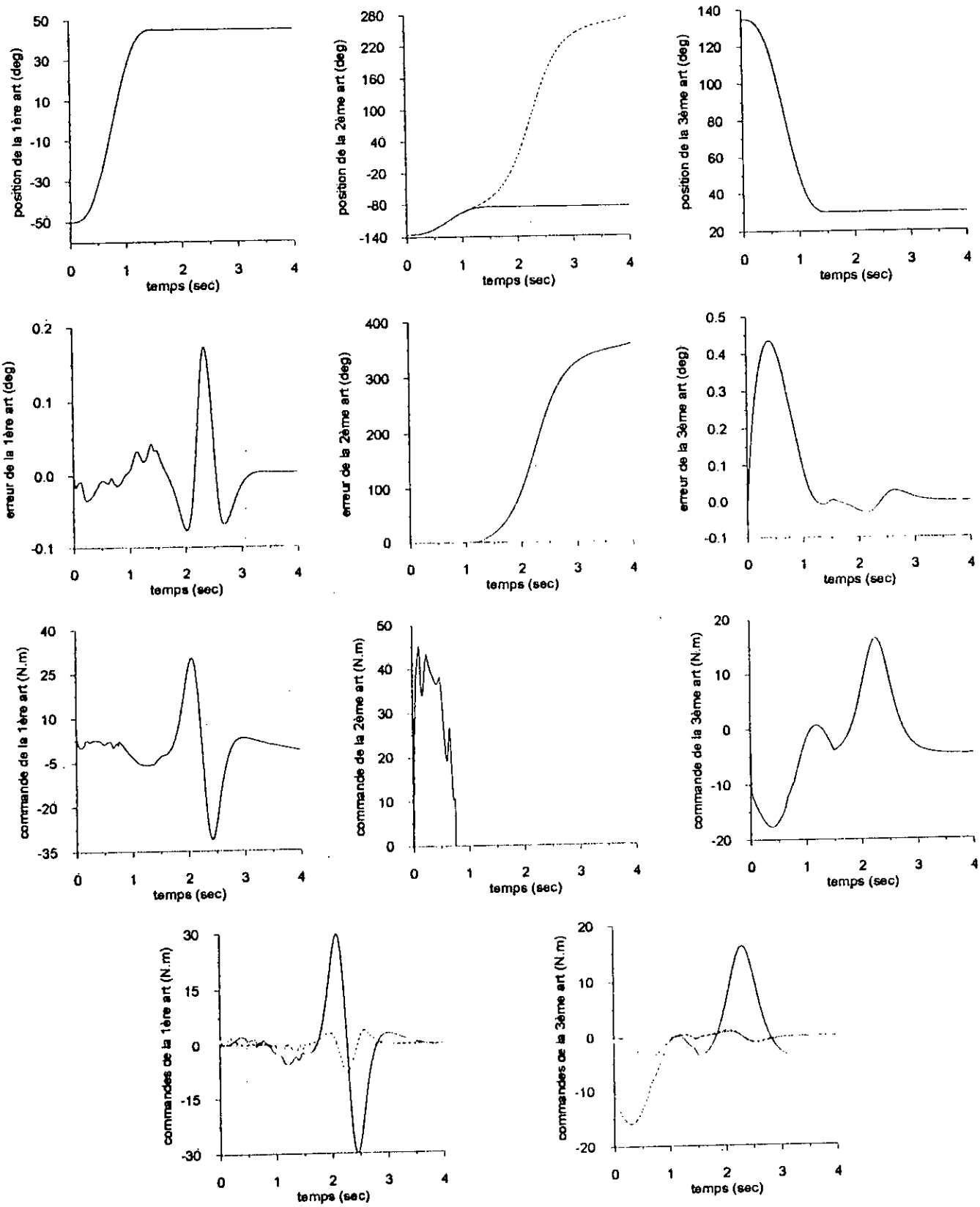

Figure III .6d. ***poursuite du robot puma 560 avec rupture de la 2èmecommande à t = 0.75 sec (trajectoire de LEAHVY) – réseau RBF -***

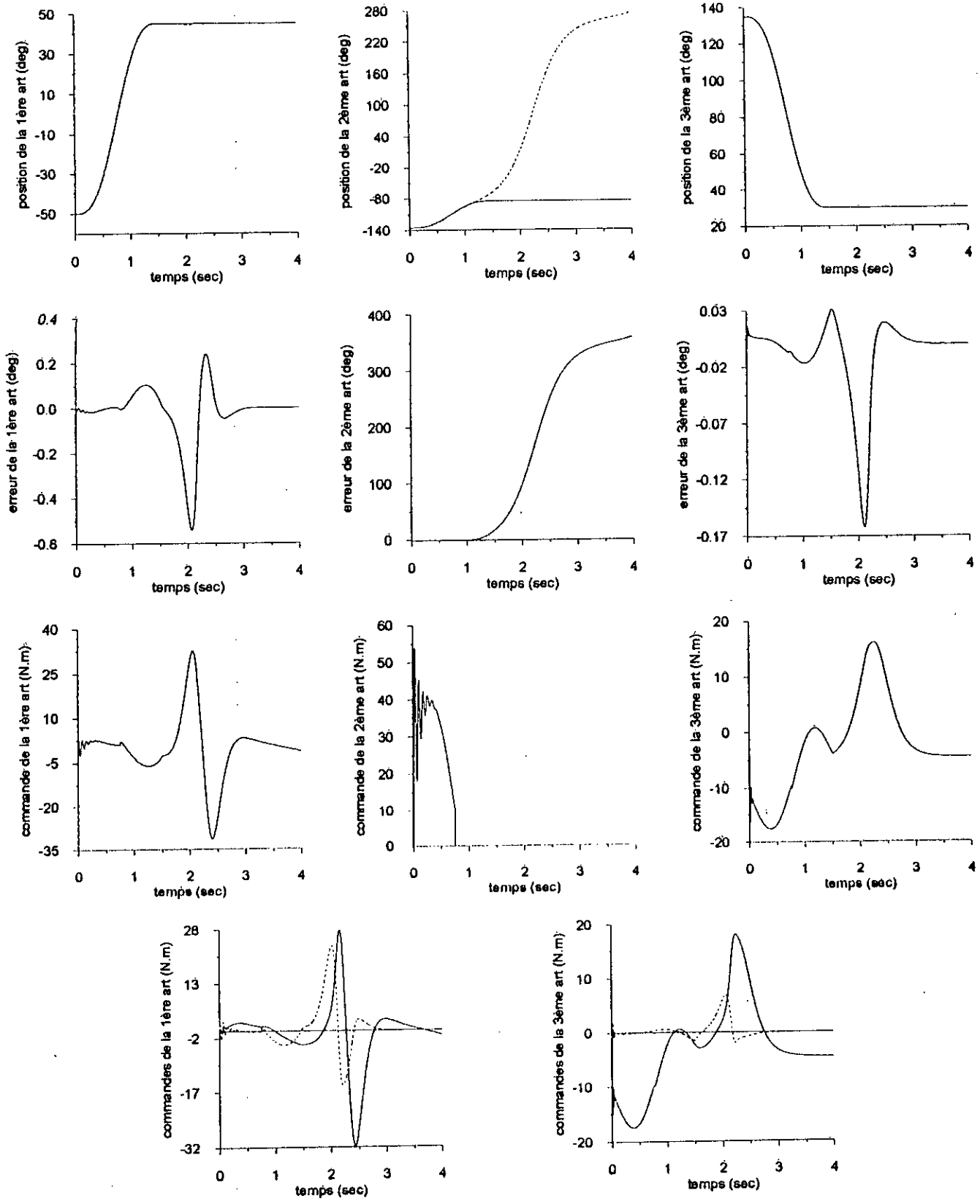

Figure III .6e. ***poursuite du robot puma 560 avec rupture de la 2èmecommande à t = 0.75 sec (trajectoire de LEAHVY) – réseau dynamique -***

b . Le bras de robot Scara :

Nous avons choisi pour le bras de robot Scara à deux degrés de liberté une trajectoire cycloïdale (voir chapitre I) telle que les articulations se déplacent respectivement de la position {0°,0°} à la position {90° , 90°}en un temps de mouvement de *3 sec.*

Les paramètres de la commande sont choisis comme suit :

$\gamma_i = 5$ $\delta_i(0) = 30$ $k_{vi} = 50$ $\lambda_i = 10$ $i = 1,2$

Ils seront les mêmes pour tout les réseaux.

- ***Réseaux statiques :***

Les paramètres des réseaux sont :

$F1 = 1$ $G1 = 0.5$
$F2 = 0.1$ $G2 = 0.05$

Chaque réseau est à trois couches, la couche cachée contient dix neurones et les fonctions d'activation sigmoïdales sont données par l'équation (III.76).
Les poids initiaux ont tous été pris nuls.

Le vecteur d'entrée du $i^{ème}$ réseau est: $x_i = (e_i, \dot{e}_i, q_{di}, \dot{q}_{di}, \ddot{q}_{di})$

- ***Réseaux récurrents 1:***

Les paramètres du réseau sont :

$F1 = 0.5$ $G1 = 0.05$ $G1' = 0.005$
$F2 = 0.05$ $G2 = 0.01$ $G2' = 0.005$

Chaque réseau est à trois couches, la couche cachée contient dix neurones, les fonctions d'activation sigmoïdales sont données par l'équation (III.76) et les poids initiaux sont tous nuls.

Le vecteur d'entrée du $i^{ème}$ réseau est: $x_i = (e_i, \dot{e}_i, q_{di}, \dot{q}_{di}, \ddot{q}_{di})$

- ***Réseaux récurrents 2 :***

Les paramètres des réseaux sont :

$F1 = 0.2$ $G1 = 0.005$
$F2 = 0.02$ $G2 = 0.005$

Chaque réseau est à trois couches, la couche cachée contient dix neurones, les fonctions d'activation sigmoïdales sont données par l'équation (III.76) et Les poids initiaux des réseaux ont tous été pris nuls.

Le vecteur d'entrée du $i^{ème}$ réseau est: $x_i = (e_i, \dot{e}_i, q_{di}, \dot{q}_{di}, \ddot{q}_{di}, z(y_i))$

- ***Réseaux à bases radiales (RBF) :***

Les paramètres des réseaux sont :

$FF1 = 1$ $FF2 = 0.5$

Les poids initiaux des réseaux sont :

$W_1(i, j)=0.05$ $W_2(i, j)=0.05$

Les réseaux sont à trois couches. on a utilisé pour chaque réseau cinq centres fixes, et une variance constante v^2 égale à $\frac{1}{5\pi}$. On subdivise les espaces normalisés de chacune des 5 entrées en 5 sous espaces réguliers. Ce qui donne un nombre de 5^5=3125 neurones au niveau de la couche cachée de chacun des trois réseaux.

Le vecteur d'entrée du $i^{ème}$ réseau est: $x_i = (e_i, \dot{e}_i, q_{di}, \dot{q}_{di}, \ddot{q}_{di})$

- ***Réseaux dynamiques :***

Les paramètres des réseaux sont :

$\mu_{h1} = 0.05$ $\mu_{v1} = 0.005$ $\mu_{w1} = 5\times10^{-6}$

$\mu_{h2} = 0.005$ $\mu_{v2} = 0.0005$ $\mu_{w2} = 5\times10^{-6}$

Les réseaux sont à trois couches et les fonctions d'activation sont des sigmoïdes données par l'équation (III.76) et les poids initiaux sont tous nuls.

Le vecteur d'entrée du $i^{ème}$ réseau est: $x_i = (e_i, \dot{e}_i, q_{di}, \dot{q}_{di}, \ddot{q}_{di})$

- **Essai à vide :**

- ***Réseaux statiques :*** (figure III.7a)

Les erreurs de poursuite sont faibles, la commande est peu oscillatoire au début et devient très vite lisse.

- ***Réseaux récurrents 1 «with self feedback » :*** (figure III.7b)

Les erreurs de poursuite sont légèrement plus grandes que pour le cas de l'utilisation de réseaux statiques et la commande est peu oscillatoire au début puis devient très vite lisse.

- ***Réseaux récurrents 2 «feedforward NN» :*** (figure III.7c)

Les performances de poursuite restent sensiblement les mêmes que lors de l'utilisation de réseaux récurrents 1 (with self feedback).

- ***Réseaux RBF :*** (figure III.7d)

Les erreurs de poursuite sont plus grandes que lors de l'utilisation des réseaux précédants mais restent acceptables, ceci est probablement dû au nombre réduit de centres (cinq), les commandes présentent des oscillations de grandes fréquences.
Cependant, on peut obtenir de plus faibles erreurs de poursuite en augmentant les gains des proportionnels dérivées. Les résultats de simulation avec $k_{vi} = 200 \quad (i=1,2)$ sont présentés à la figure (III.7d'), la poursuite est largement meilleure mais les commandes ne sont pas améliorées.

- ***Réseaux dynamiques :*** (figure III.7e)

Les erreurs de poursuite sont légèrement augmentées par rapport aux réseaux statiques et récurrents, mais restent de valeurs relativement faibles ; les commandes sont douces.
Cependant, on peut diminuer l'ordre des erreurs de poursuite en augmentant les gains des proportionnels dérivées. Les résultats de simulation avec $k_{vi} = 200 \quad (i=1,2)$ sont présentés à la figure (III.7e'), les erreurs de poursuite sont réduites et les commandes restent douces.

- **Essai avec lâcher de charge :**

Pour ce test, l'effecteur contient une charge de 4kg. On ajoute une perturbation au robot qui consiste à ce que la masse tombe accidentellement après 1.5 sec.
Les résultats de la poursuite de trajectoire pour les différents réseaux de neurones utilisés sont présentés aux figures III.8a, iii.8b, iii.8c, III.8d et III.8e.
Les commandes agissent au moment du lâcher de la masse pour contrer la perturbation, ce qui prouve le caractère adaptatif et robuste de la commande.
Toutes les remarques faites lors de l'essai à vide restent vérifiées pour ce test.

- **Essai avec perturbation externe :**

Ce test consiste à considérer que le bras SCARA reçoit un choc à 1.5 sec. Ce choc est traduit par l'introduction d'une force de perturbation externe $T_{di} = -T_i \quad (i = 1,2)$ à 1.5 sec et de durée d'application *0.01 sec.*
Les résultats de simulation (figures III.9a, III.9b, III.9c, III.9d, III.9e) montrent que les commandes agissent pour contrer cette perturbation, ce qui donne lieu à un nouveau régime transitoire oscillatoire en commandes durant lequel les erreurs sont légèrement augmentées, mais elles reviennent très vite à leurs formes initiales.
Les mêmes remarques que celles données lors de l'essai à vide restent valables pour ce test.

- Un autre problème est à soulever. Il s'agit du cas où les positions initiales des différentes articulations du bras de robot diffèrent de celles des trajectoires désirées. Ce problème a fait l'objet d'un dernier test, dans lequel le bras SCARA a pour positions initiales {-30°, 30°} tandis que les trajectoires désirées commencent à la position {0°, 0°}. On a utilisé pour ce test un réseau RBF avec essai à vide du robot SCARA.

Les résultats de simulation (figure III.10) sont satisfaisants ; les commandes sont grandes au début pour assurer que les positions articulaires suivent les trajectoires désirées, et dès que c'est fait elles reviennent à leurs formes lors de l'essai à vide (figure III.7d).

- **Etude comparative :**

Dans ce tableau sont présentées les erreurs maximales (en degrés) de chaque articulation du robot SCARA pour les tests effectués précédemment, ainsi qu'une qualification des commandes correspondantes.

N° art	Test	Réseau statique	Réseau récurrent 1	Réseau récurrent 2	Réseau gaussien (RBF)	Réseau dynamique
1	à vide	0.017	0.032	0.037	0.476	0.098
	Décharge	0.017	0.036	0.034	0.569	0.100
	Perturbation à 1.5 s	0.047	0.069	0.069	0.476	0.107
2	à vide	0.036	0.058	0.077	0.425	0.166
	Décharge	0.036	0.075	0.087	0.323	0.220
	Perturbation à 1.5 s	0.080	0.146	0.146	0.425	0.272
Commandes		Douces	Douces	Douces	Oscillatoires	Très douces

Tableau III .2. ***étude comparative des différents réseaux pour la commande adaptative robuste du bras SCARA .***

Le choix du réseau de neurones approprié pour la commande adaptative du bras de robot SCARA reste conditionné par le cahier des charges.

Il est tout de même à noter que le réseau dynamique représente le meilleur compromis : faibles erreurs et commandes douces. Dans les applications très précises, on préférera alors l'utilisation des réseaux statique qui donne lieu aux erreurs de poursuite les plus faibles tout en assurant des commandes assez douces.

On voit bien que les réseaux agissent différemment pour les deux modèles de bras manipulateurs, ce qui est normal, car l'apport du réseau dans la commande dépend de la dynamique du bras manipulateur, de son inertie, des autres paramètres de commande

Il est donc impossible de généraliser les résultats obtenus pour n'importe quel modèle de bras de robot à commander.

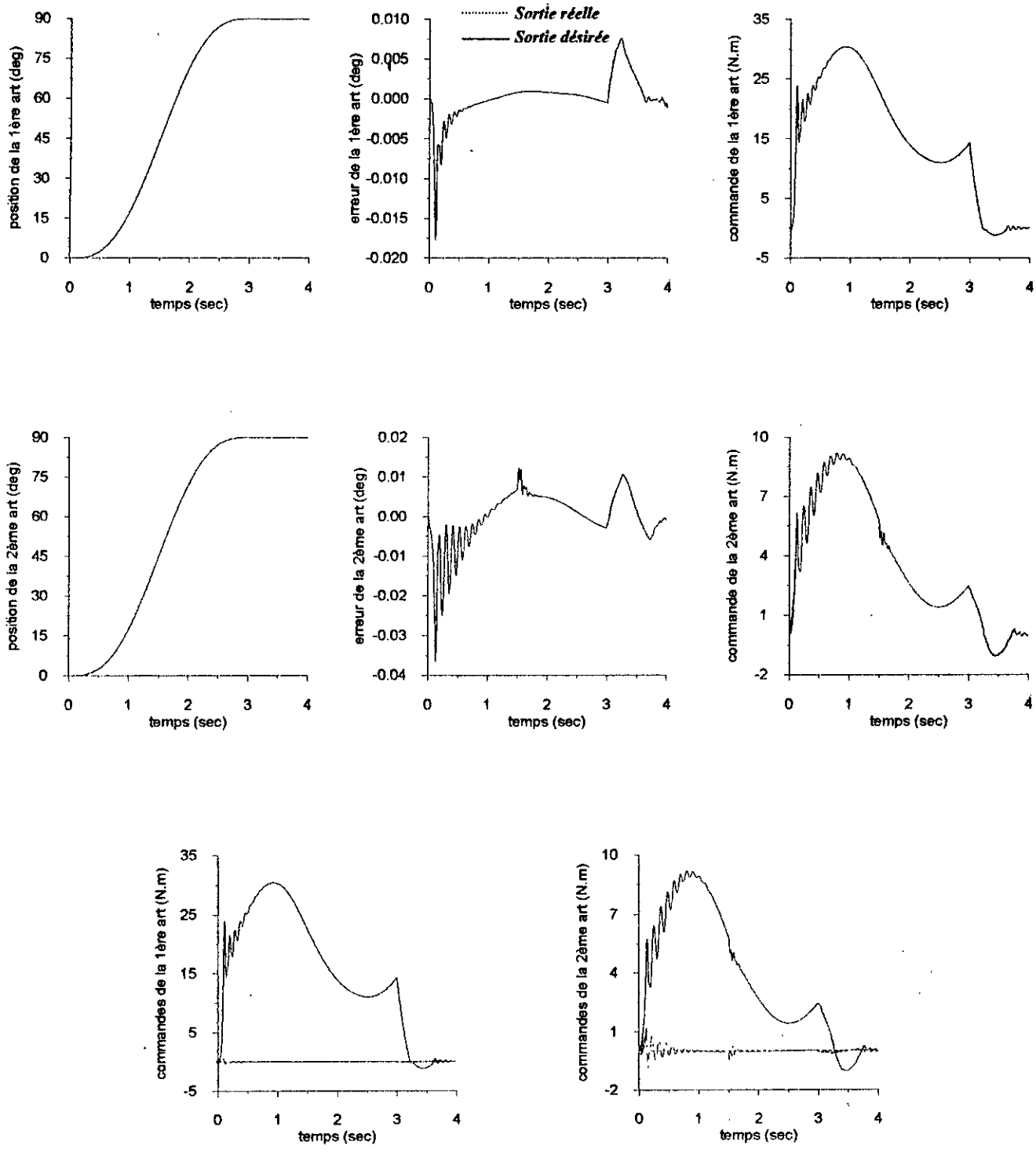

Figure III .7a . ***Poursuite à vide du robot SCARA***
- réseau statique-

——— ***sortie du réseau***
····· ***terme robuste***
-·-·-·- ***sortie du PD***

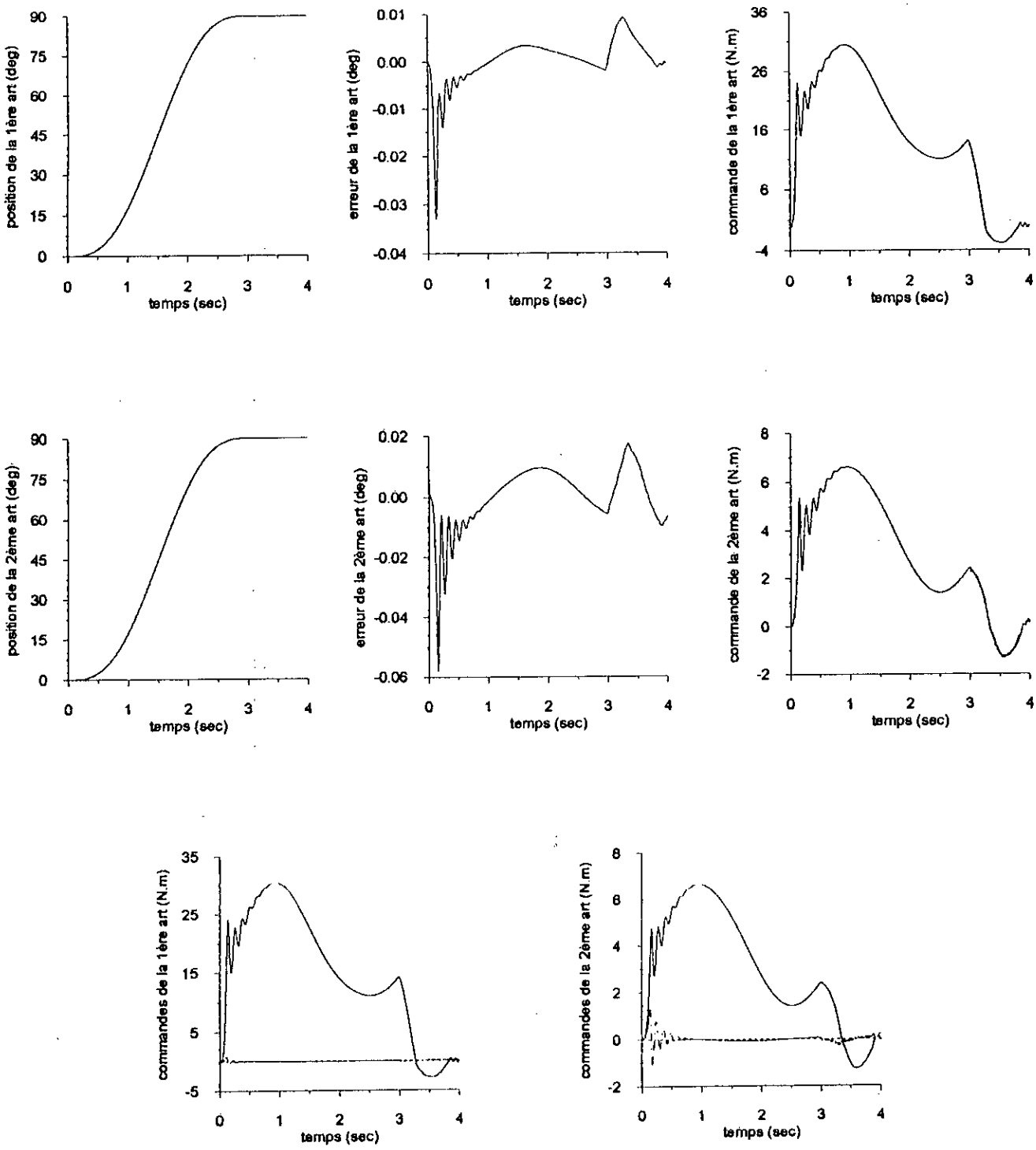

Figure III .7b . ***Poursuite à vide du robot SCARA***
- réseau récurrent 1-

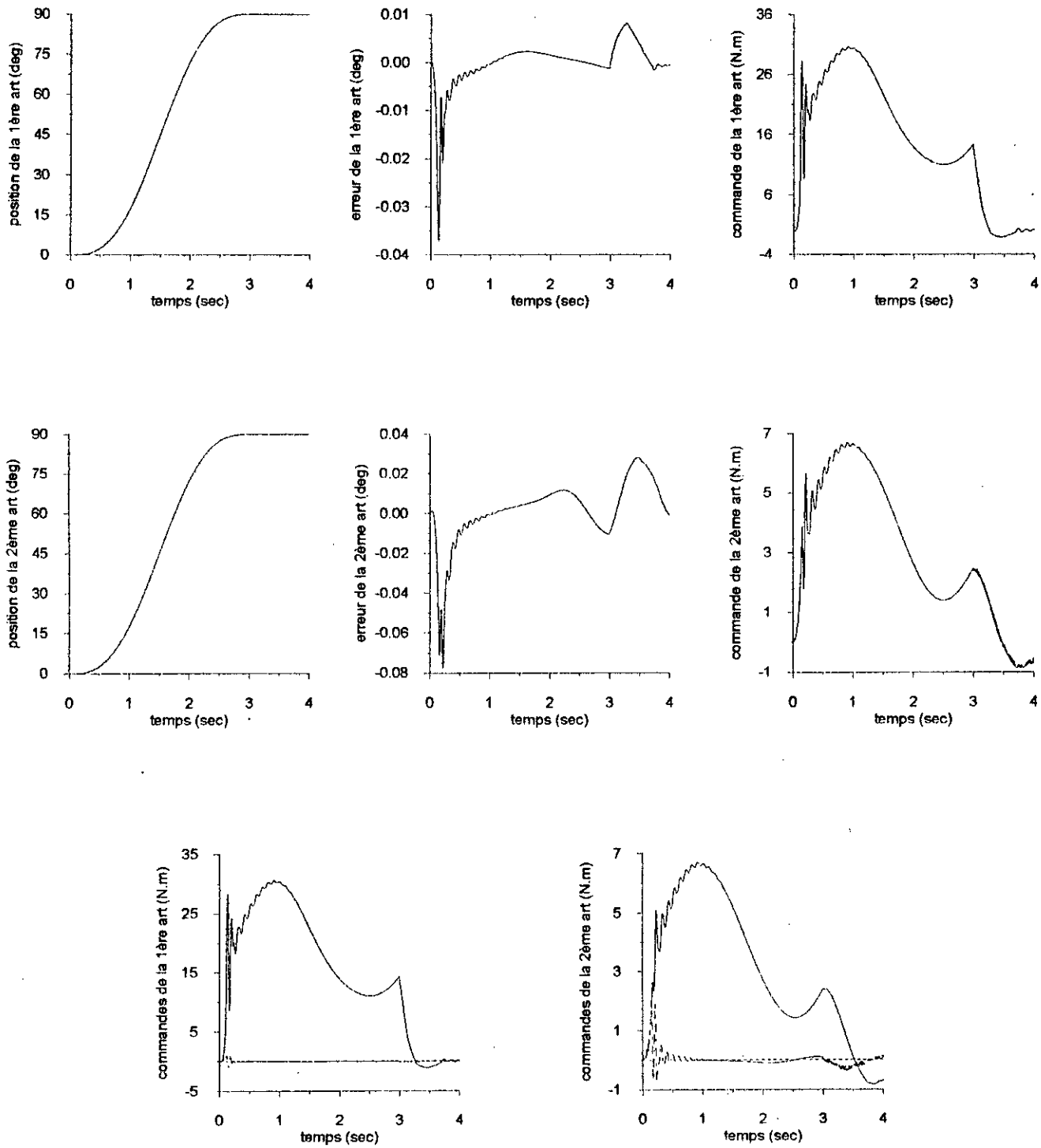

Figure III .7c . ***Poursuite à vide du robot SCARA***
- réseau récurrent 2 -

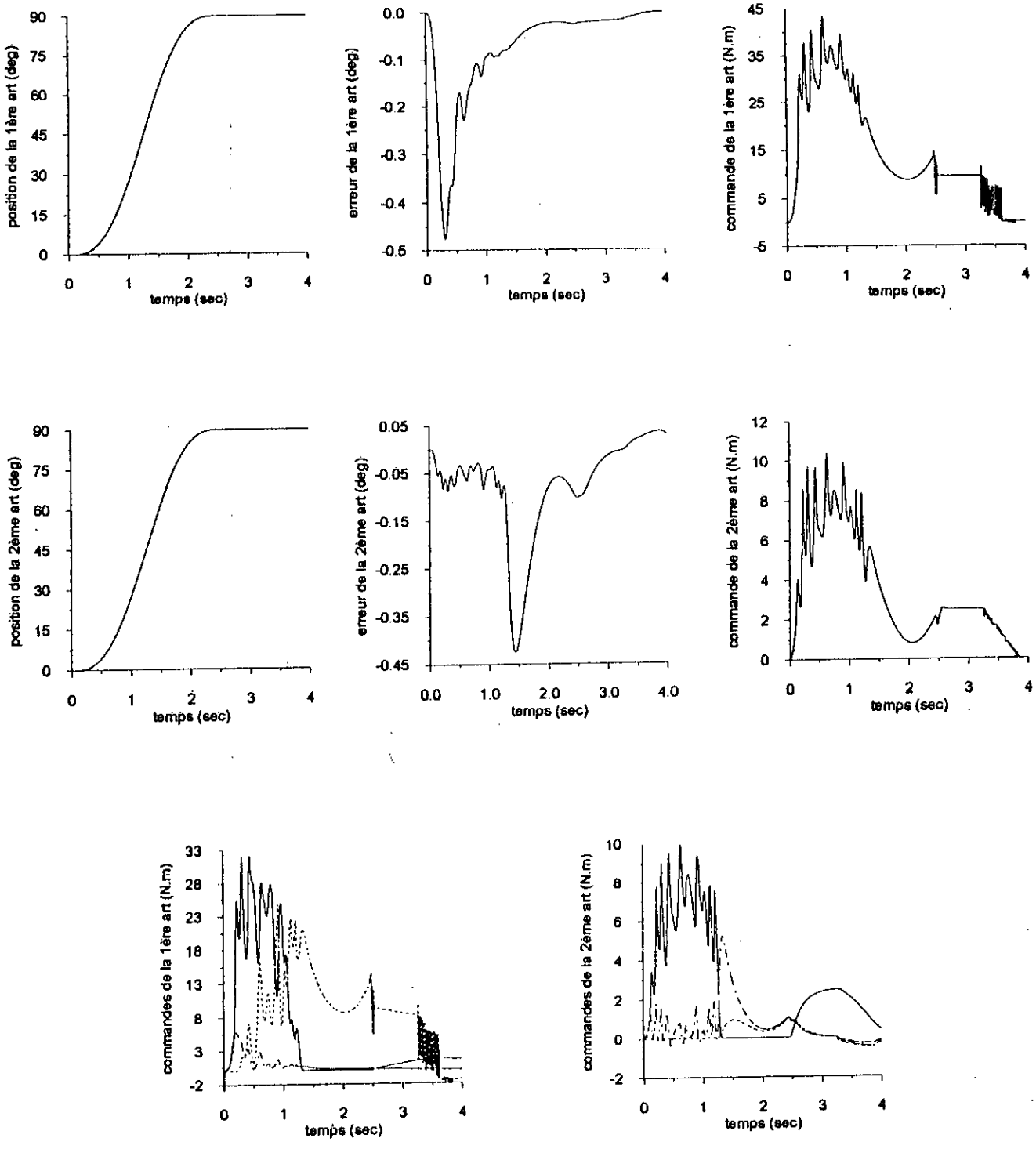

Figure III .7d . ***Poursuite à vide du robot SCARA***
- réseau RBF -

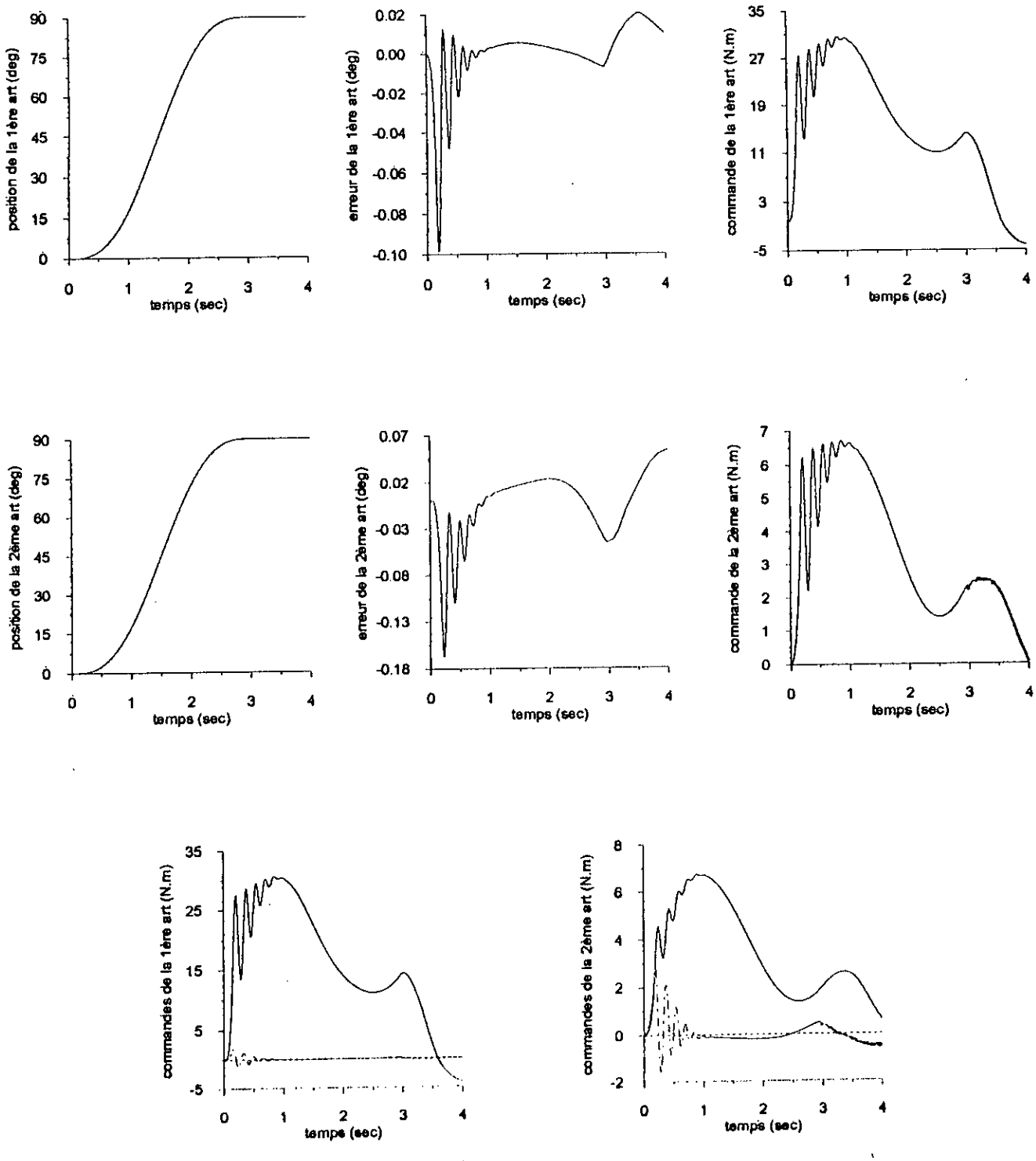

Figure III .7e . ***Poursuite à vide du robot SCARA***
- réseau dynamique -

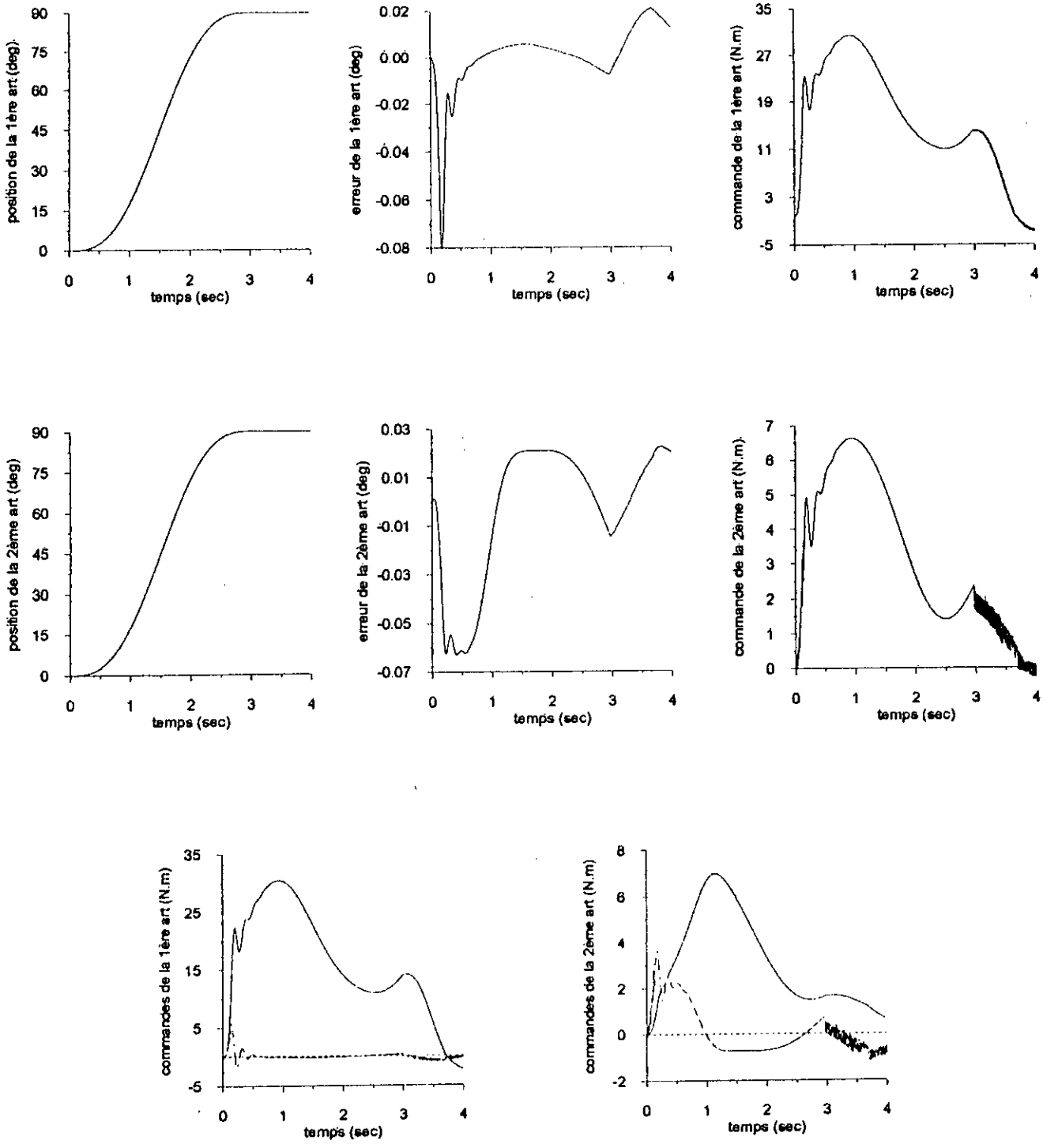

Figure III .7e'. ***Poursuite du robot SCARA à vide (k_v =200)***
- réseau dynamique -

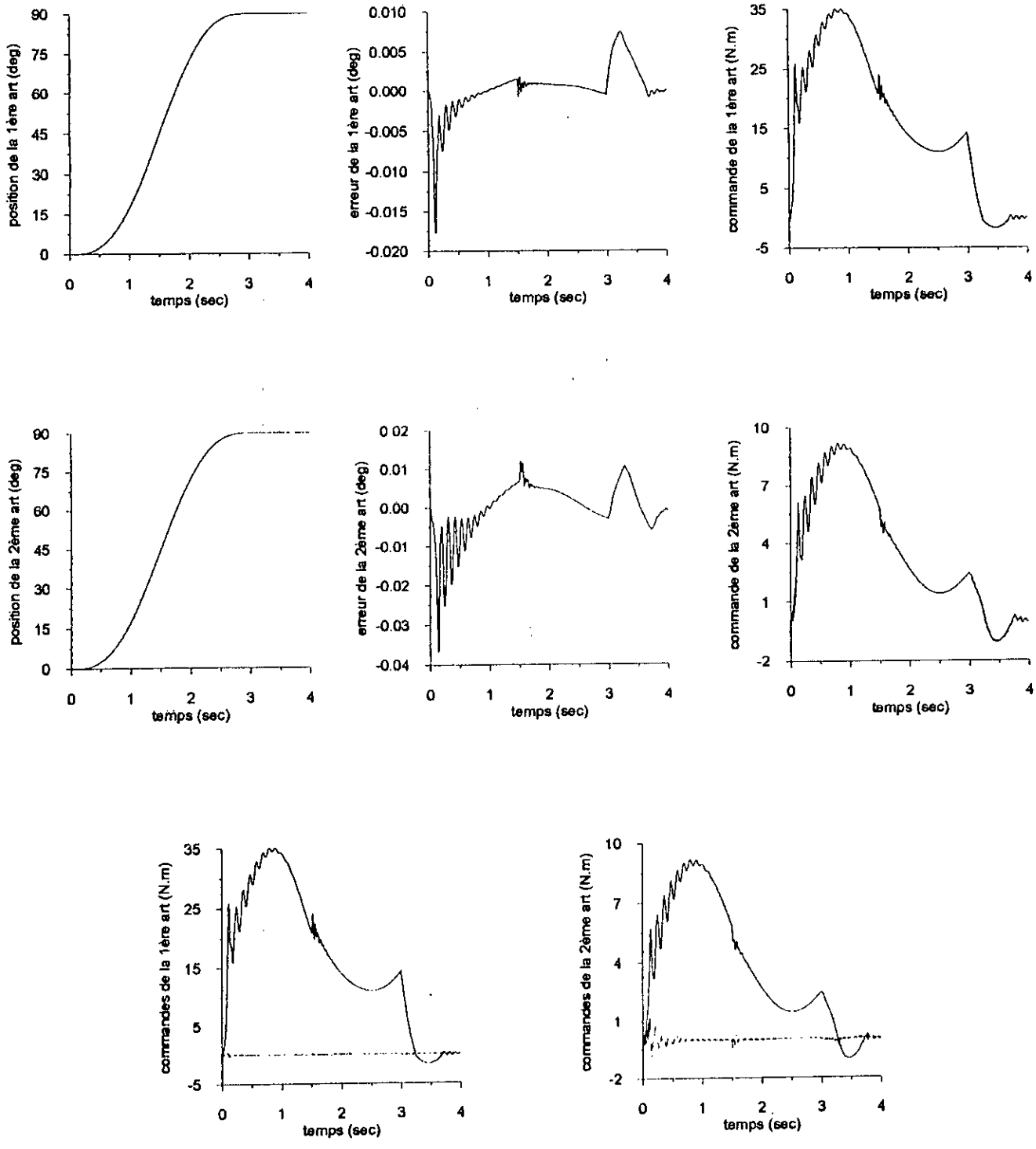

Figure III .8a . ***Poursuite du robot SCARA avec lâcher de la charge (4kg) après 1.5 sec - réseau statique -***

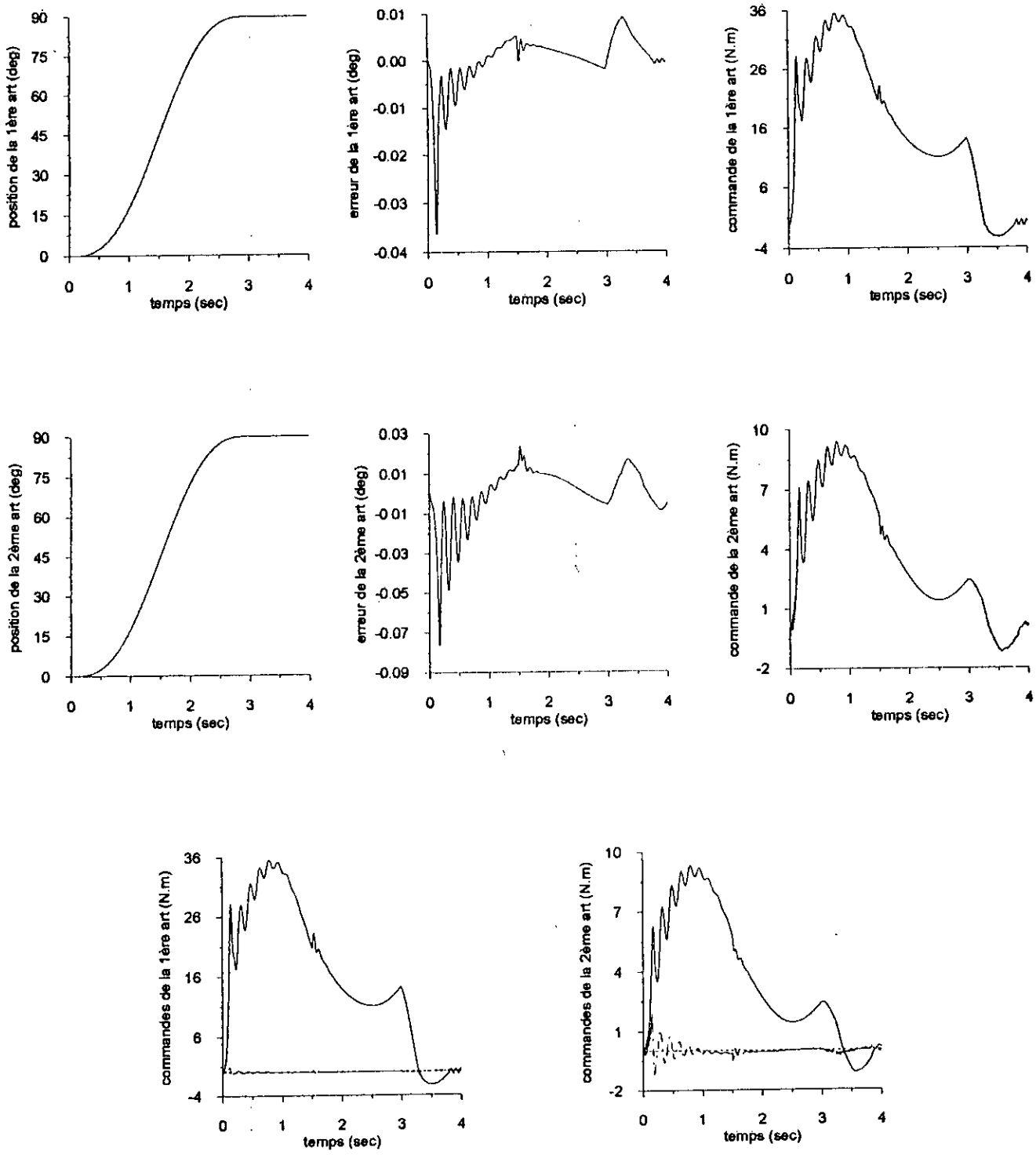

Figure III .8b . ***Poursuite du robot SCARA avec lâcher de la charge (4kg) après 1.5 sec - réseau récurrent 1 -***

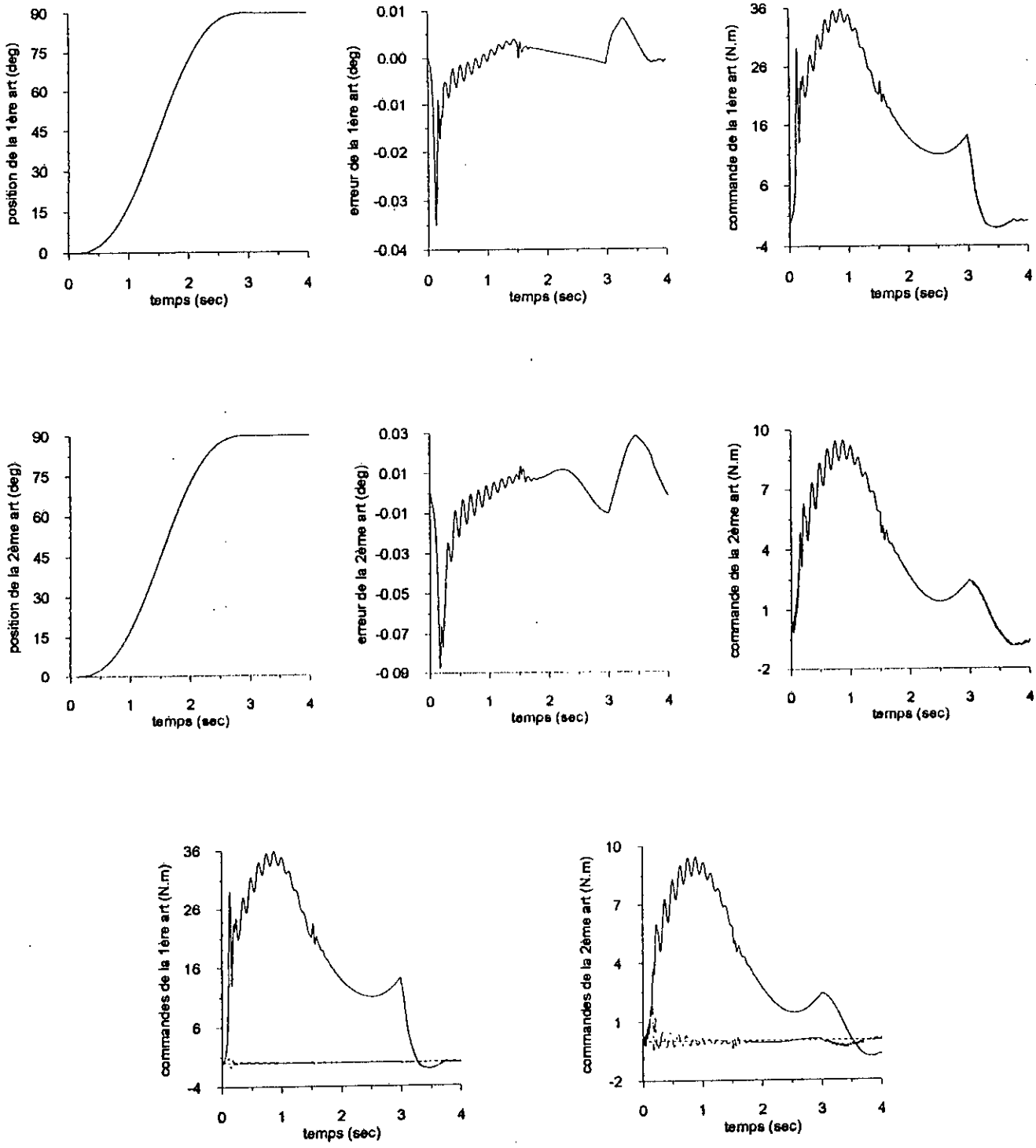

Figure III .8c . ***Poursuite du robot SCARA avec lâcher de la charge (4kg) après 1.5 sec - réseau récurrent 2 -***

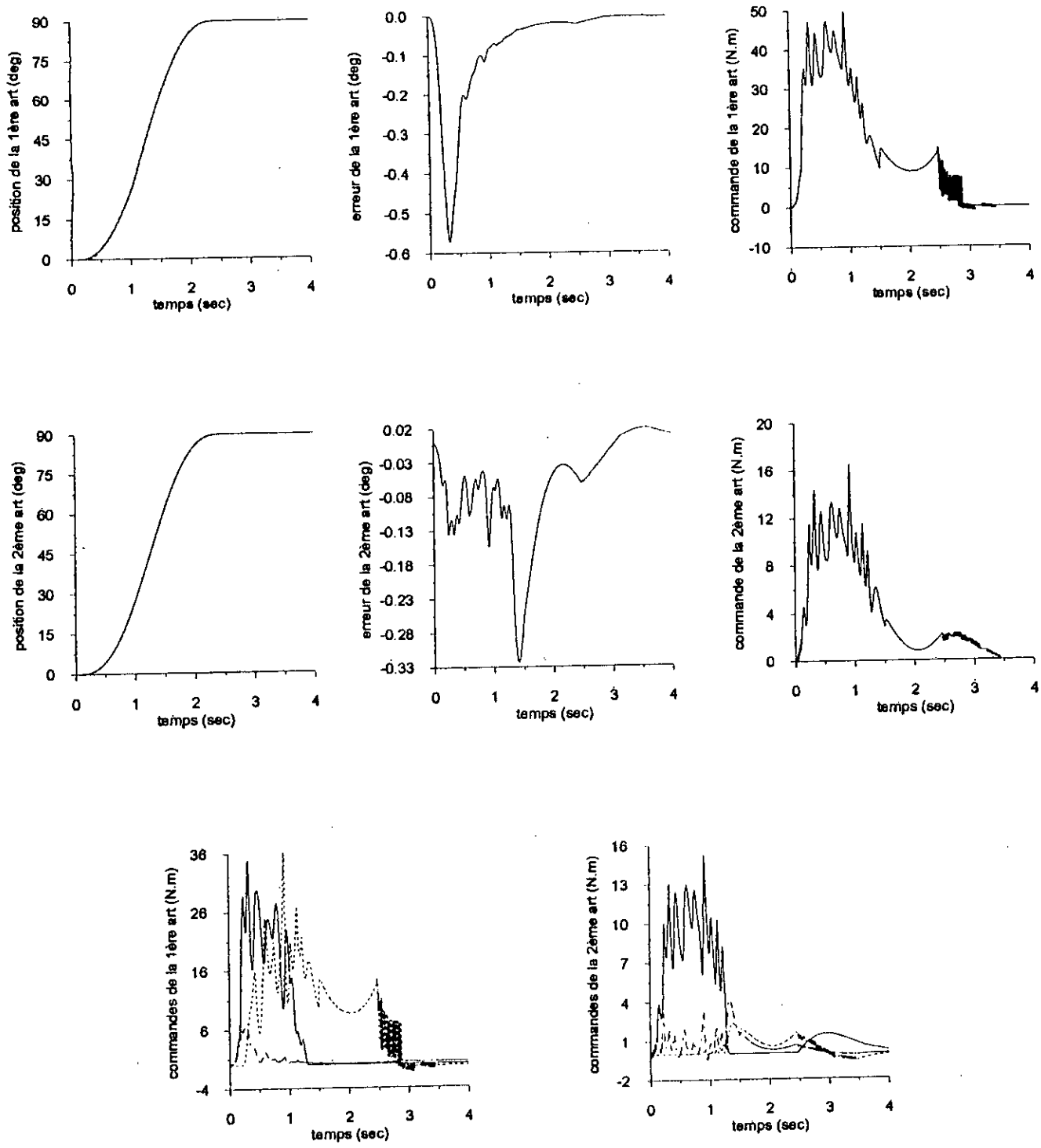

Figure III .8d . ***Poursuite du robot SCARA avec lâcher de la charge (4kg) après 1.5 sec – réseau RBF -***

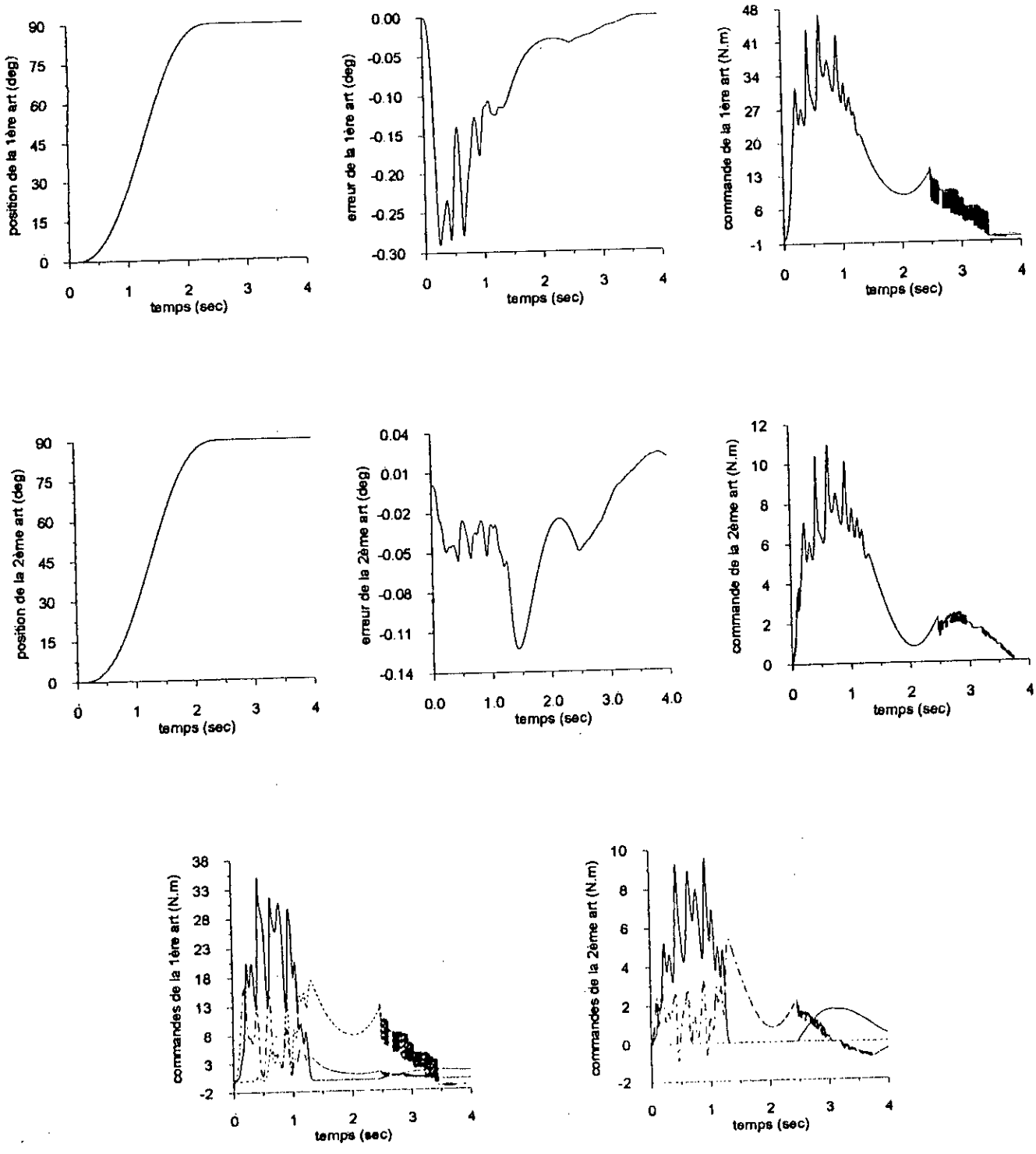

Figure III .8d' . ***Poursuite du robot SCARA avec lâcher de la charge (4kg) après 1.5 sec (k_v =200) - réseau RBF -***

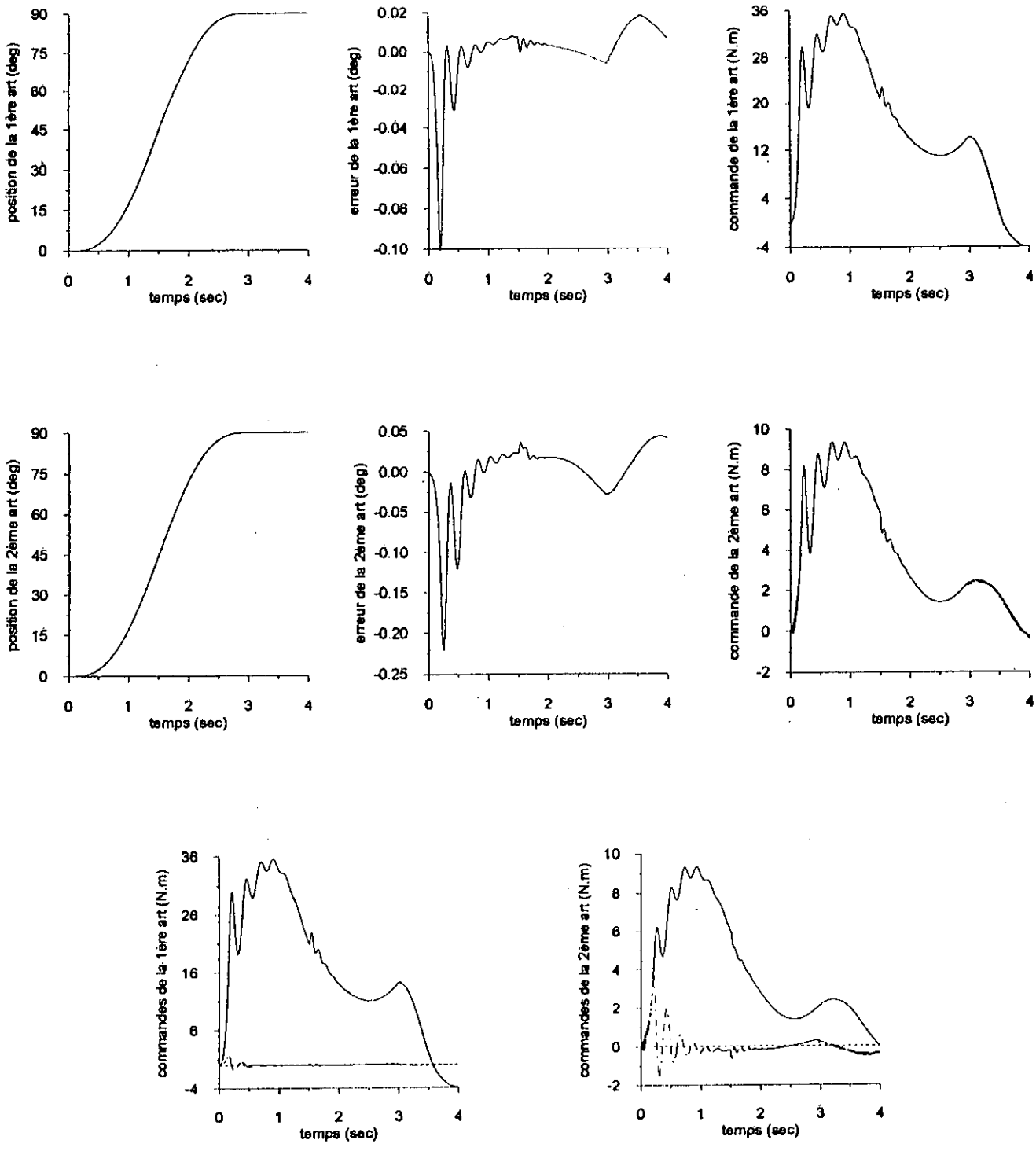

Figure III .8e . ***Poursuite du robot SCARA avec lâcher de la charge (4kg) après 1.5 sec – réseau dynamique -***

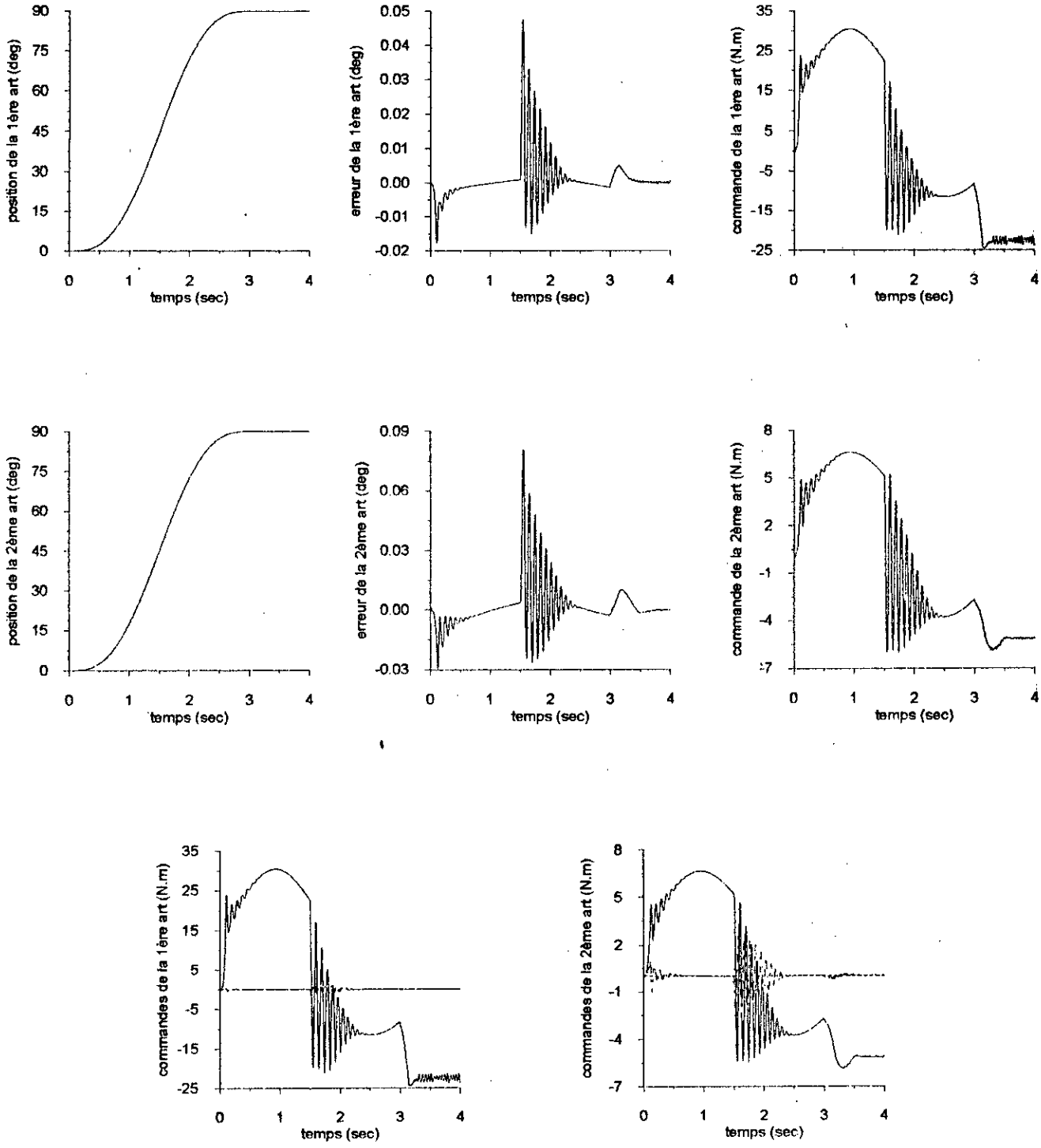

Figure III .9a . ***Poursuite du robot SCARA avec perturbation externe à 1.5 sec - réseau statique -***

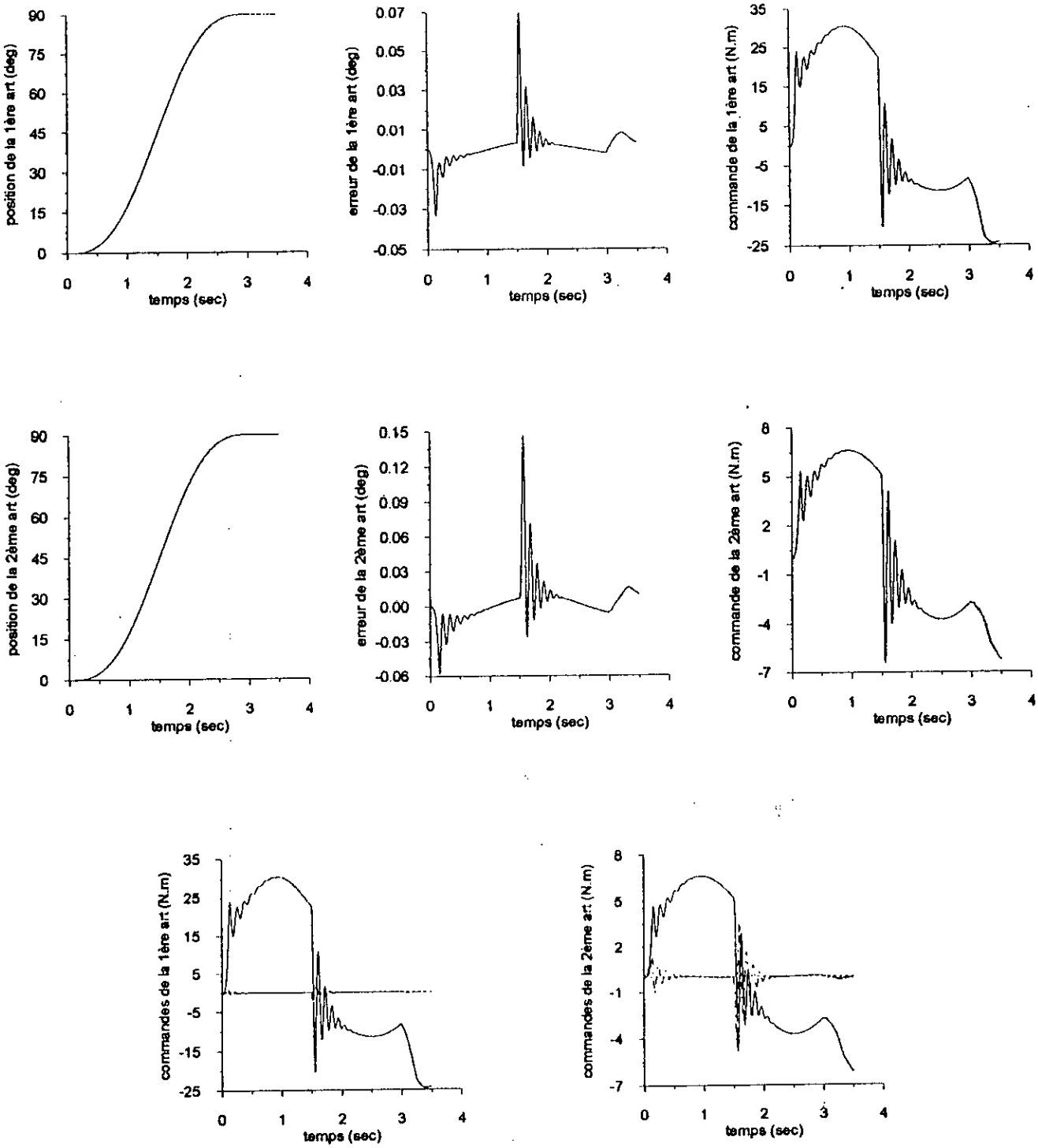

Figure III .9b . ***Poursuite du robot SCARA avec perturbation externe à 1.5 sec - réseau récurrent 1 -***

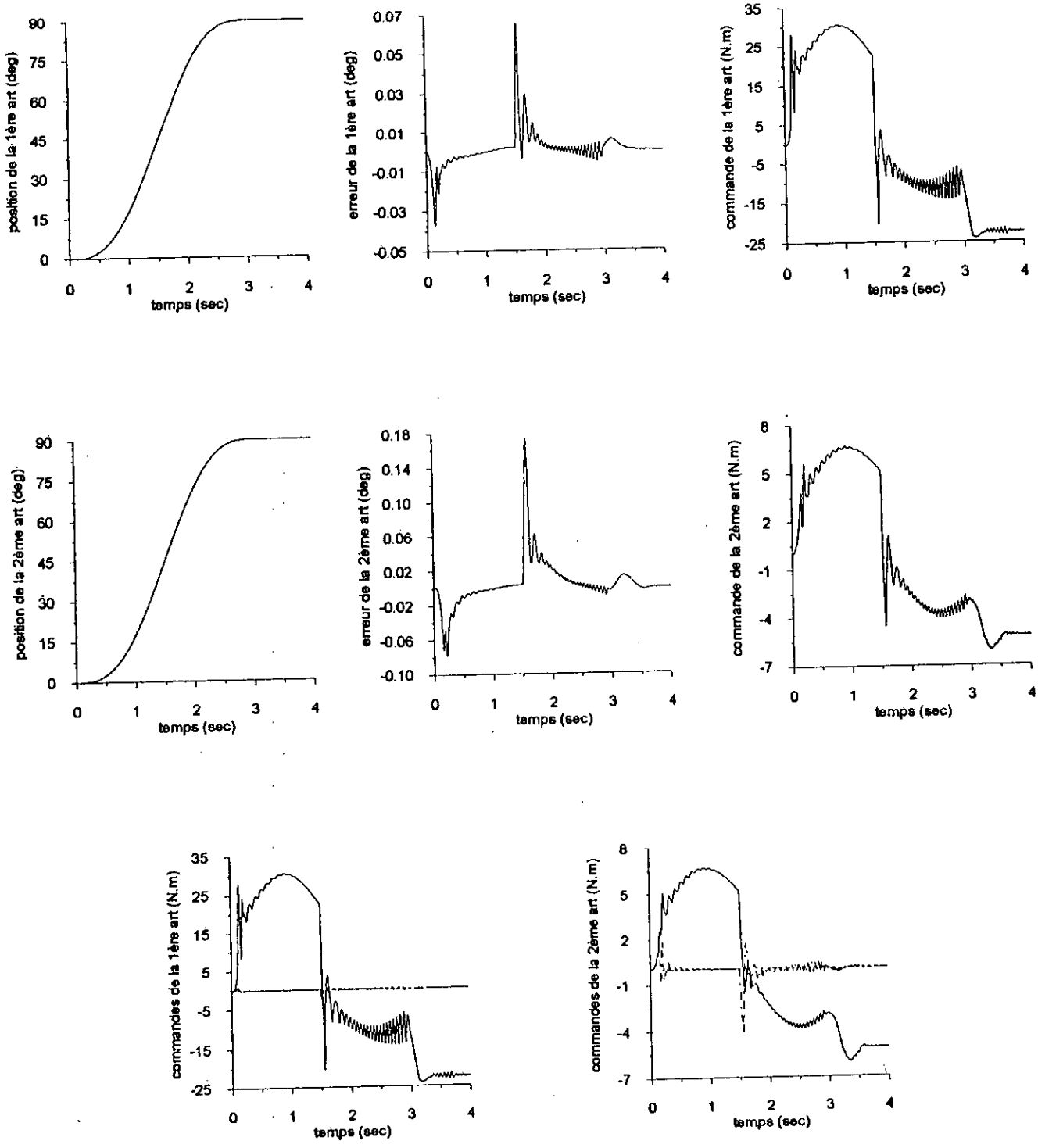

Figure III .9c . ***Poursuite du robot SCARA avec perturbation externe à 1.5 sec - réseau récurrent 2 -***

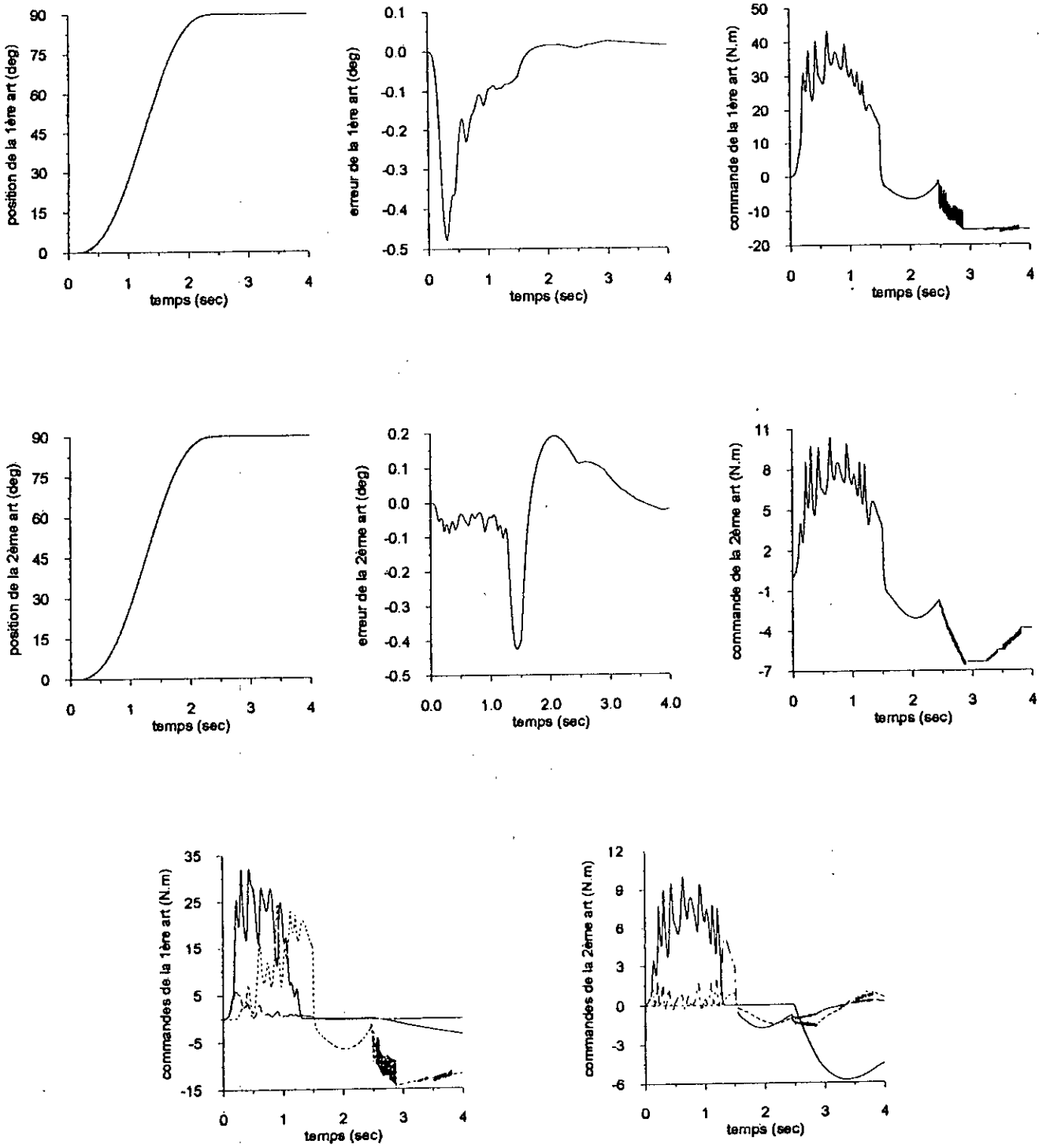

Figure III .9d . ***Poursuite du robot SCARA avec perturbation externe à 1.5 sec - réseau RBF -***

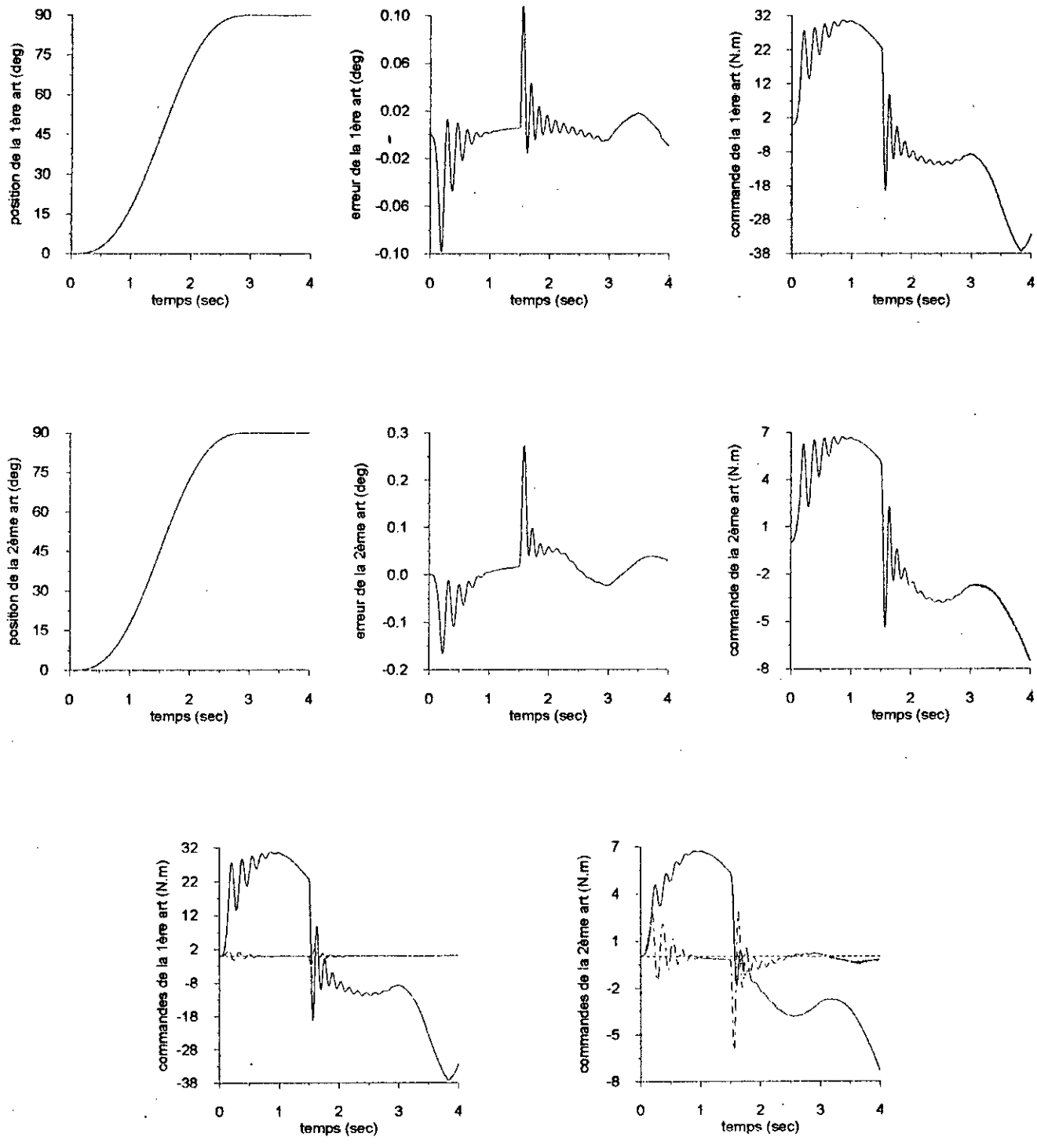

Figure III .9e . ***Poursuite du robot SCARA avec perturbation externe à 1.5 sec - réseau dynamique -***

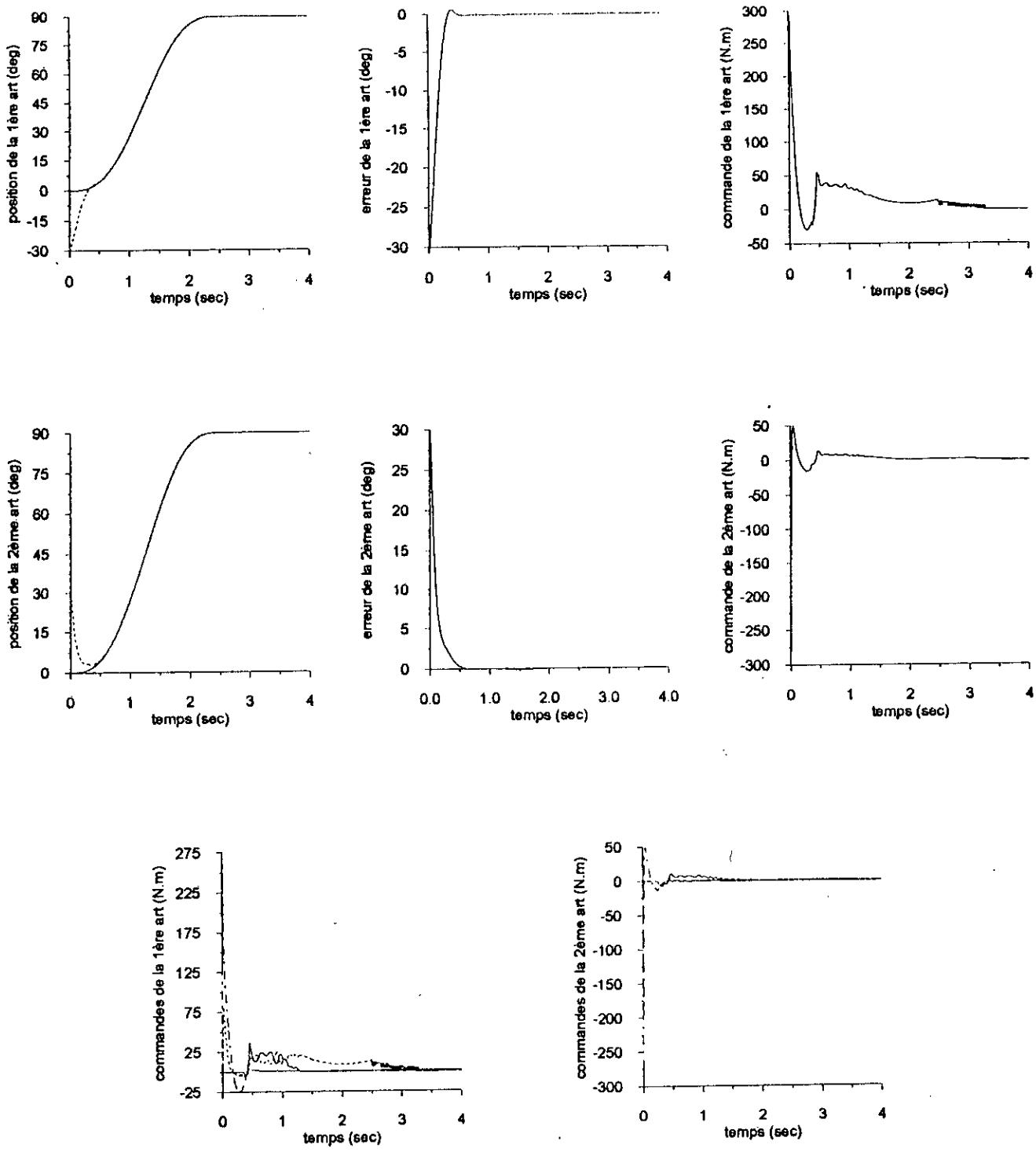

Figure III .10. ***Poursuite à vide du robot SCARA***
- réseau RBF -

III .6. Conclusion :

Dans ce chapitre, une approche décentralisée d'une commande adaptative robuste de bras de robots a été présentée et appliquée à deux modèles différents de bras manipulateurs fortement non linéaires et mal connus (PUMA 560 et SCARA).
Des simulations ont été effectuées en utilisant cinq types différents de réseaux de neurones (les réseaux statiques, les réseaux récurrents «with self feedback », les réseaux récurrents «feedforward NN », les réseaux de fonctions à base radiale et les réseaux dynamiques) en vue d'une comparaison de leur contribution dans la poursuite de trajectoire de chacun des bras de robots considérés.
Les résultats obtenus sont très satisfaisants et il est important à noter que pour les réseaux que nous avons utilisés, le choix de l'un d'entre eux pour la commande des robots considérés reste essentiellement conditionné par le cahier des charges (performances souhaitées) et les caractéristiques des différents moteurs utilisés pour générer les couples de commande de chaque articulation des bras manipulateurs étudiés.

Conclusion générale et perspectives

Conclusion générale et perspectives

Dans ce travail, on s'est intéressé à la commande adaptative décentralisée ainsi qu'à la contribution des différentes structures de réseaux de neurones dans la commande des bras manipulateurs.
La loi de commande contient trois termes élémentaires :

- Un terme linéaire assuré par un régulateur proportionnel dérivée ;
- Un terme non linéaire assuré par un réseau de neurones qui doit approximer une certaine fonction non linéaire inconnue ;
- Un terme robuste qui agit face aux perturbations externes et aux erreurs de modélisation.

Une extension au cas décentralisé de la commande adaptative par compensation désirée introduite par LEWIS a été établie. Cela consiste à l'introduction d'un contrôleur local au niveau de chaque articulation du bras de robot, pour générer la commande appropriée, en fonction seulement des données disponibles localement. La preuve de la stabilité générale du robot avec cette nouvelle structure de commande a été élaborée.

Par la suite, plusieurs structures de réseaux de neurones ont été introduites dans la commande et appliquées à des robots 'SCARA' et 'PUMA 560' en gardant les paramètres du terme robuste et du régulateur proportionnel dérivée, pour chaque articulation, constants ; ceci en vue d'effectuer une étude comparative de l'apport de chacun des réseaux dans la commande des bras de robots considérés.

Les résultats de simulations numériques à base des modèles des bras de robots établis ont montré l'efficacité de la stratégie de commande adoptée, qui assure de bonnes poursuites et qui est très robuste aux variations massiques subites, allant jusqu'à 20% de la masse des robots considérés, ainsi qu'aux perturbations extérieures.
On a constaté aussi que les différentes structures de réseaux utilisées dans la commande agissent différemment dans la conduite des deux robots. Donc, on ne peut pas généraliser les résultats de l'étude comparative effectuée pour la commande d'un bras de robot quelconque.
Le choix d'un réseau approprié pour la conduite des bras de robots dépend essentiellement du robot à commander, des spécifications du cahier des charges ainsi que des caractéristiques des moteurs utilisés pour générer les couples de commande.

Le choix très varié des différentes structures de réseaux utilisés dans ce mémoire est très significatif ; et on a vu que quel que soit le cahier des charges, il y a toujours un réseau qui remplit ses spécifications. On peut alors dire de manière générale qu'un choix de réseau de neurones approprié assurant les performances désirées est toujours possible, et ceci quelque soit le bras de robot utilisé.
La contribution très satisfaisante des réseaux de neurones dans la commande de bras de robots est désormais prouvée ; ces derniers étant des systèmes fortement non linéaires et mal connus, ceci nous conduit à donner encore plus d'importance pour les réseaux de neurones dans la conduite des systèmes non linéaires.

Comme perspectives, il serait intéressant d'élargir le domaine d'exploitation des réseaux de neurones artificiels, en les utilisant dans l'identification comme des observateurs non linéaires dans la commande des systèmes complexes.
Il serait tout aussi intéressant d'essayer de développer une stratégie de commande où seul le réseau de neurones assurerait la conduite du bras manipulateur ou tout autre système non linéaire à commander.

Annexe

La théorie de l'échantillonnage et de la reconstruction

A .1. La transformée de Fourier :

Considérons le signal continu *x(t)* à énergie finie.

$$\int_{-\infty}^{\infty} |x(t)|^2 dt < \infty \qquad \text{(A.1)}$$

La représentation de Fourier dans le domaine fréquentiel de ce signal est :

$$x(j\Omega) = \int_{-\infty}^{\infty} x(t) exp(-j\Omega t) dt \qquad \text{(A.2)}$$

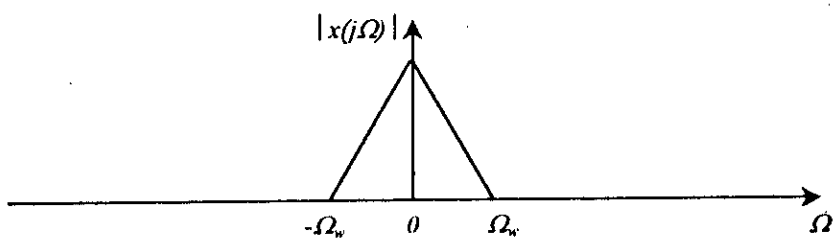

Figure A.1. ***spectre de fréquences du signal x(t).***

La transformée inverse est donnée par :

$$x(t) = \frac{1}{2} \int_{-\infty}^{\infty} x(j\Omega) exp(j\Omega t) d\Omega \qquad \text{(A.3)}$$

si la fréquence maximale du signal $x(t)$ est w Hz, on définit la pulsation $\Omega_w = 2\pi w$ rad/sec tel que :

$$x(j\Omega) = 0 \quad si \quad |\Omega| \geq \Omega_w = 2\pi w \tag{A.4}$$

A .2. L'échantillonnage : [Jayant 84] , [Astrom 90]

Définissons l'extension périodique du signal $x(j\Omega)$ (fig.2) :

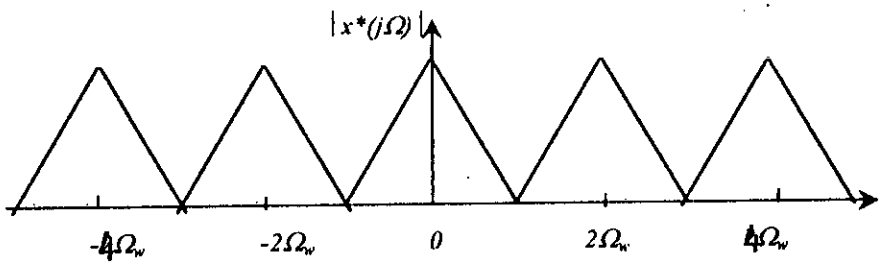

Figure A.2. ***extension périodique du spectre de fréquences du signal x(t).***

$$x^*(j\Omega) = \sum_{k=-\infty}^{\infty} x[\, j(\Omega - 2k\Omega_w)] \tag{A.5}$$

Il est clair qu'on peut reconstruire le signal $x(j\Omega)$ à partir du signal $x^*(j\Omega)$ (fig.2) comme suit :

$$x(j\Omega) = x^*(j\Omega) rect\left(\frac{\Omega}{2\Omega_w}\right) = \begin{cases} x^*(j\Omega) & si \quad |\Omega| \leq \Omega_w \\ 0 & ailleurs \end{cases} \tag{A.6}$$

La decomposition en série de Fourier du signal périodique $x^*(j\Omega)$ est de la forme :

$$x^*(j\Omega) = \sum_{k=-\infty}^{\infty} c_k \exp(-jk\pi\, \Omega/\Omega_w) \tag{A.7a}$$

avec : c_k est le vecteur des coefficients de Fouier telle que :

$$c_k = \frac{1}{2\Omega_w} \int_{-\Omega_w}^{\Omega_w} x(j\Omega) exp(jk\pi\Omega/\Omega_w) \qquad \text{(A.7b)}$$

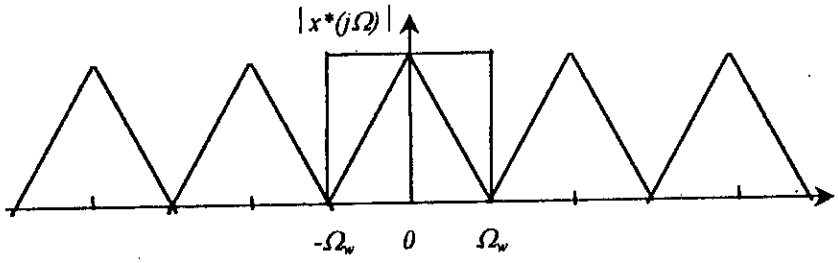

Figure A.3. ***régénération de*** $|x(j\Omega)|$ ***à partir de*** $|x^*(j\Omega)|$

Des équations (A.3) et (A.7.b), on obtient :

$$\begin{cases} c_k = \dfrac{\pi}{\Omega_w} x\left(\dfrac{k\pi}{\Omega}\right) = T\,x(kT); \\ T = \dfrac{\pi}{\Omega_w} = \dfrac{1}{2w} \end{cases} \qquad \text{(A.8)}$$

T représente la période d'échantillonnage du signal $x(t)$ appelée l'intervalle de Nyquist.

A.3. La reconstruction : [Jayant 84] , [Astrom 90] , [Slotine 92]

Disposant des échantillons $x(kT)$ du signal $x(t)$, on calcule les coefficients de Fourier c_k, et on peut reconstruire exactement le signal continu $x(t)$ en reprenant la transformée de Fourier inverse.

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

$$x(j\Omega) = T\ rect\left(\frac{\Omega}{2\Omega_w}\right) \sum_{k=-\infty}^{\infty} x(kT) exp(-jk\Omega T) \qquad \text{(A.9)}$$

La reconstruction du signal continu $x(t)$ sera donnée par les équations suivantes :

$$x(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty}\left[T\ rect\left(\frac{\Omega}{2\Omega_w}\right)\sum_{k=-\infty}^{\infty}x(kT)exp(-jk\Omega T)\right]exp(j\Omega t)d\Omega \qquad (A.10)$$

$$x(t) = \frac{1}{2\pi}\int_{-\Omega_w}^{\Omega_w}\left[T\sum_{k=-\infty}^{\infty}x(kT)exp(-jk\Omega T)\right]exp(j\Omega t)d\Omega \qquad (A.11)$$

$$x(t) = T\sum_{k=-\infty}^{\infty}x(kT)\left[\frac{1}{2\pi}\int_{-\Omega_w}^{\Omega_w}exp\, j\Omega t(t-kT)d\Omega\right] \qquad (A.12)$$

sachant que :

$$\frac{1}{2\pi}\int_{-\Omega_w}^{\Omega_w}exp\, j\Omega t(t-kT)d\Omega = \frac{sin(u\Omega_w)}{\pi u} \qquad (A.13)$$

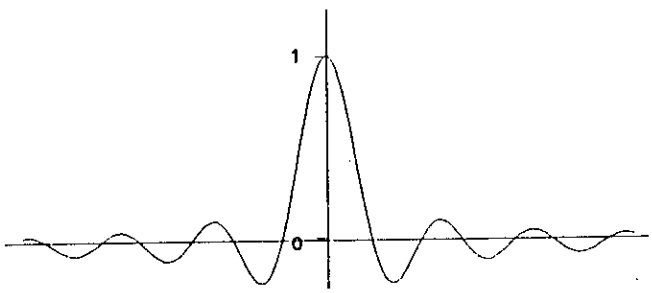

Figure A .4. ***la fonction sinc***

on obtient :

$$x(t) = \sum_{k=-\infty}^{\infty}x(kT)\ sinc\left[\frac{\pi}{T}(t-kT)\right]; \qquad sinc(x) = \frac{sin(x)}{x} \qquad (A.14)$$

En augmentant la fréquence d'échantillonnage, la représentation de Fourier du signal échantillonné sera sous la forme présentée sur la figure (5).

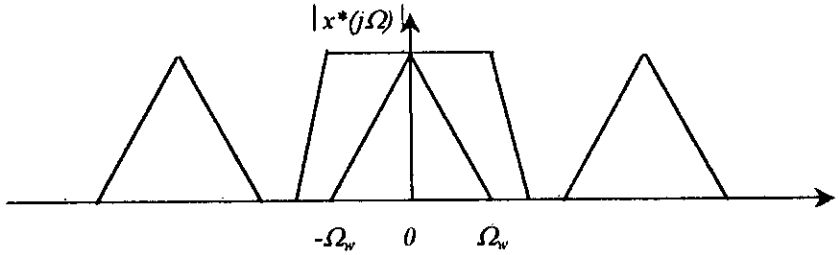

Figure A.5. ***régénération de*** $|x(j\Omega)|$ ***à partir de*** $|x^*(j\Omega)|$ ***pour T<1/2w.***

Ainsi, dans la théorie de Shanon, pour éviter les chevauchements dans la représentation de Fourier et pouvoir reconstruire le signal continu en utilisant pratiquement des filtres, la fréquence d'échantillonnage doit être superieure à deux fois la fréquence maximale du signal réel.

On remarque bien que la reconstruction du signal continu *x(t)* est la somme d'une infinité de fonctions *sinc* pondérées par les échantillons *x(kT)*. Cela constitue l'idée de base de la reconstruction de fonctions par les réseaux RBF (A.5).

A .4. Le filtrage et l'interpolation : [Jayant 84] , [Astrom 90] , [Slotine 92]

Le signal échantillonné est obtenu comme suit :

$$x^*(t) = s_p(t)x(t) = T\sum_{k=-\infty}^{\infty} x(kT)\,\delta(t-kT) \qquad \text{(A.15)}$$

où $s_p(t)$ est le train d'impulsions de Dirac.

La reconstruction idéale de *x(t)* est possible par un passage du signal *x*(t)* par un filtre passe bas idéal (fig.A.6) de fonction de transfert :

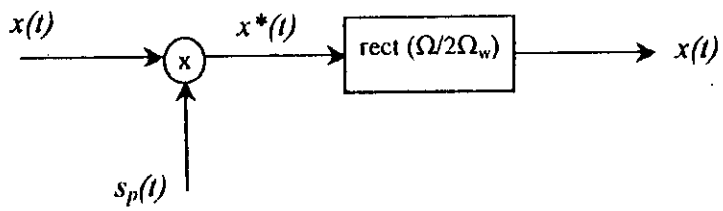

Figure A.6. ***échantillonnage et reconstruction par filtrage idéal.***

$$h(t) = \frac{1}{T} sinc(\Omega_w t) \tag{A.16}$$

La réponse fréquentielle du filtre est donnée par :

$$H(j\Omega) = rect\left(\frac{\Omega}{2\Omega_w}\right) = rect\left(\frac{\Omega T}{2\pi}\right) \tag{A.17}$$

Dans le domaine temporel, la sortie du filtre sera le produit de convolution entre le train d'échantillons $x^*(t)$ et la fonction du filtre $h(t)$.

$$x(t) = x^*(t) * h(t) = \int_{-\infty}^{\infty} x^*(t) h(t-\tau) d\tau$$

$$x(t) = \sum_{k=-\infty}^{\infty} x(kT)\, sinc\left[\frac{\pi}{T}(t-kT)\right] \tag{A.18}$$

A .5. Reconstruction par réseaux gaussiens : [Slotine 92]

Pratiquement, on ne dispose pas de filtre idéal. Une fonction de transfert gaussiènne du filtre est facilement réalisable, donc on opère sur un spectre de fréquence du signal, défini comme suit :

$$x(j\Omega) = \begin{cases} \frac{1}{5} exp(\Omega^2)(1-|\Omega|) & si \quad |\Omega| \le \Omega_w \\ 0 & ailleurs \end{cases} \tag{A.19}$$

Cette approximation présente une erreur négligeable par rapport à l'apport du filtre gaussien dans la reconstruction ; ainsi, dans [slotine 92], on approxime le signal continu $x(t)$ comme suit :

$$x(t) = \sum_{\zeta \in I} c(\zeta) g(t-\zeta) \tag{A.20}$$

où $c(\zeta)$ est le vecteur des pondérations en fonction des points autours desquels on veut approximer le signal $x(t)$, obtenus dans les réseaux gaussiens par des techniques d'apprentissage présentées au chapitre II. La fonction g est l'ensemble des gaussiènnes.

Il est clair que la reconstruction ne se fait pas seulement dans le domaine temporaire, ainsi, toute fonction continue $f(x)$ peut être reconstruite dans son domaine, en estimant les paramètres de pondération.

Bibliographie

[Armstrong 86] B. Armstrong , O. Khatib & J. Burdick : ''The explicit dynamic model and inertial parameters of the PUMA 560 arm''. Proc international conference on Robotics and automation, 1986.

[Aström 90] K.J.Aström : ''Computer controlled systems''. Edition Prentice Hall ,1990.

[Bali 95] N. Bali : ''Etude des performances de la commande prédictive généralisée appliquée aux robot manipulateurs PUMA et SCARA''. Thèse de magister, ENP, 1995.

[Benallegue 91] A . Benallegue : ''Contribution à la commande dynamique adaptative des robots manipulateurs rapides''.Thèse de doctorat de l'université PARIS VI, novembre 1991.

[Bouhali 96] O. Bouhali :''Commande adaptative à structure décentralisée par RNA : application à un bras de robot manipulateur '', PFE -ENP, juin 1996.

[Bouhali 97] O. Bouhali, D.Boukhetala, F. Boudjema, M. S.Boucherit : ''Commande adaptative décentralisée par réseaux de neurones gaussiens appliquée à un pendule''. Journal of the IEEA, vol 1, N° 1, Jan 1997.

[Freeman 92] J. A. Freeman : ''Neural networks : Algorithms, applications and programming techniques''. Edition Addison Wesley ,1992.

[Fu 87] K. S. Fu, A. C. Gonzalez and C. S. G. Lee : ''Robotics : control, sensing, vision and intelligence''. Edition Mc Graw Hill , 1987.

[Jayant 84] N. S. Jayant : ''Digital coding of wave forms''. Edition Prentice Hall , 1984.

[Karakasoglu 93] A . Karakasoglu , S. Sudharsanan :''Identification and decentralize adaptive control using dynamical neural networks with application to robotic manipulators''. IEEE Trans. on neural networks, Vol. 4, N° 6, Nov 1993.

[Kosco 92] A . Kosco : ''Neural network and fuzzy systems''. Edition Printice Hall 1992.

[Lallemand 94] Lallemand : '' Robotique : aspets fondamentaux ''. Edition Masson, 1994.

[Lewis 93] F. L .Lewis , A .Yesildirek & K . Liu : ''neural net robot controller with guaranteed stability ''. IEEE 1993.

[Lewis 95] F. L . Lewis , C .M .Kwan , D. .M. Dawson : '' robust adaptive control of robots using neural network : global tracking stability''. conference on decision & control , New Orleans , IEEE 1995.

[Madani 97] T. Madani , S. Amrati : ''Commande décentralisée à structures variables''. PFE – ENP, juin 1997.

[Nedjari 96] M . S Nedjari , H . Boukari : ''commande adaptative décentralisée. Application en robotique''. PFE - ENP, juin 1996.

[Paul 83] R.P. Paul : ''Robots manipulators : mathematics, programming and control''. MIT press 1983.

[Renders 95] Renders : ''Algorithmes génétiques et réseaux de neurones''. Edition Hermès, Avril 1995.

[Sadegh 90] N . Sadegh, R . Horowitz : ''Stability and robustness analysis of a class of adaptive controllers for robotic manipulators''. Int. J. Robot. Res. , Vol 9, 1990.

[Seraji 89] H . Seraji :''decentralized adaptive control of manipulators : theory, simulation and experimentation''. IEEE trans. on robotics and automation, Vol. 5, N° 2, Avril 1989.

[Slotine 92] J-J. E. Slotine and R. M. Sanner : ''Gaussien Networks for Direct Adaptive Control ''. IEEE Trans. on neural networks, Vol. 3, N° 6, Nov 1992.

[Yang 96] X . Yang : ''Neural Network Model and System Used for Nonliear Control''. IFAC ,1996.

ملخص : في هذا العمل ، قمنا بتطوير طريقة للتحكم اللامركزي التلاؤمي بواسطة الشبكات العصبونية الاصطناعية على أساس الاتزان الكلي. تم التحصل على بنية التحكم انطلاقا من طريقة التحكم المركزي التلاؤمي المدرج من طرف LEWIS ، طبقت بعد ذلك هذه التقنية على ذراعين آليين من نوع SCARA و PUMA 560 باستعمال عدة أنواع من الشبكات العصبونية (شبكات ساكنة ، تراجعية ، ذوات القواعد القطرية و ديناميكية).

أخيراً ، وبواسطة المحاكاة العددية قارننا مساهمة الشبكات العصبونية في التحكم الآلي للأذرع المشار إليها سابقاً.

الكلمات المفتاحية : التحكم التلاؤمي ، اللامركزية ، الشبكات العصبونية الإصطناعية ، الأيدي الآلية.

Abstract : From a centralized robust adaptive control of robots (desired compensation law with neural networks) introduced by ''LEWIS'', an extension to a decentralized structure based on the global stability analysis is developed at first. Then , a comparative study of the contribution of different structures of neural networks (static, recurrent, dynamic neural networks and neural networks with radial basis functions) used for the control of robot 'SCARA' and 'PUMA 560' is presented in base of simulation results.

Key words : adaptive control, decentralization, artificial neural networks, robot manipulators.

Résumé: A partir de la stratégie de la commande adaptative centralisée par compensation désirée par réseaux de neurones appliquée aux robots manipulateurs introduite par ''LEWIS'', une extension au cas décentralisé basée sur la stabilité globale est développée en premier lieu. En second lieu, une étude comparative de la contribution de différentes structures des réseaux de neurones (réseaux statiques, récurrents, de fonctions à bases radiales et dynamiques), utilisées pour la commande des bras 'SCARA' et 'PUMA 560' est effectuée sur la base de résultats de simulations.

Mots clés: la commande adaptative, la désentralisation, les réseaux de neurones artificiels, les bras de robots.