République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Batna 2
Faculté de Technologie
Département d'Électronique

Mémoire

Présenté pour l'obtention du diplôme de
MAGISTER en Électronique

OPTION

Contrôle industriel

Par

BELLOUMI Rabie
(Ingénieur d'état en électronique)

Thème

Optimisation par Essaim de Particules
Application à un système complexe

Soutenu devant le jury composé de :

Dr. ARAR Djemai	Prof.	Université	de BATNA	Président
Dr. SLIMANE Noureddine	M.C.A.	Université	de BATNA	Rapporteur
Dr. ABDESSEMED Yacine	M.C.A.	Université	de BATNA	Examinateur
Dr. BOUKABOU Abdelkrim	Prof.	Université	de JIJEL	Examinateur

Remerciements

Je tiens à exprimer toute ma reconnaissance à mon Directeur de mémoire monsieur **SLIMANE Noureddine,** maître de conférences à l'Université de Batna. Je le remercie de m'avoir encadré, orienté, aidé et conseillé durant toute la durée de réalisation de ce travail.

Je voudrai exprimer mes sincères remerciements à monsieur **ARAR Djemai**, professeur à l'Université de Batna, d'avoir accepté la présidence du jury.

Je tiens à remercier également monsieur **ABDESSEMED Yassine**, Maître de Conférences à l'Université de Batna, d'avoir accepté d'examiner ce travail.

J'adresse mes vifs remerciements à monsieur **BOUKABOU Abdelkrim**, professeur à l'Université de Jijel, de m'avoir honoré en acceptant de faire partie du jury et d'examiner ce modeste travail.

Je ne saurai oublier mes collègues et amis, avec lesquels j'ai partagé de très bons moments, merci et bonne chance à tous.

Enfin, je remercie tous ceux qui ont contribué de près ou de loin à l'aboutissement de ce mémoire.

Sommaire

Chapitre III : La commande par mode glissant

Chapitre IV : La commande par mode glissant floue et floue adaptative

Introduction

« Le désir humain de perfection trouve son expression dans la théorie de l'optimisation. Elle étudie comment décrire et atteindre ce qui est meilleure, une fois que l'on connait comment mesurer et modifier ce qui est bon et ce qui est mauvais, la théorie de l'optimisation comprend l'étude quantitative des optimums et les méthodes pour les trouver » [1]

A partir de la citation ci-dessus on comprend que l'optimisation cherche à améliorer une performance en se rapprochant d'un point optimum.

Dans le cadre des mathématiques, l'optimisation cherche des réponses à un type général de problèmes, qui consistent à sélectionner le meilleur élément parmi plusieurs appartenant au même ensemble. En général l'optimisation peut se réaliser dans différents domaines, toujours avec le même objectif : améliorer le fonctionnement au moyen d'une gestion perfectionnée des ressources. L'optimisation peut avoir lieu à n'importe quelle étape quoi qu'il est conseillé de la mener à bien jusqu'à la fin du processus visé. .Aujourd'hui, l'optimisation dispose d'une vaste application dans différentes branches de la science, par exemple : contrôle, robotique, aérodynamisme, consommation d'énergie, ……etc.

En raison de la nécessité de traiter plusieurs variétés de problèmes d'optimisation, de nombreux changements et nouveautés en optimisation concernant les méthodes et les approches ont été proposés. Parmi ces approches, on trouve les algorithmes métaheuristiques qui sont des algorithmes standards, proposés pour la résolution des différents problèmes complexes et fortement non linéaires sans nécessiter de modification profonde de l'algorithme pour chaque problème à résoudre.

Les métaheuristiques présentent de nombreux avantages mais aussi un certain nombre de limitations. Nous résumons ci-dessous les différentes caractéristiques de ces méthodes :

- Application générale possible à une large classe de problèmes.

- Efficacité pour de nombreux problèmes.

- Possibilité de compromis entre qualité des solutions et temps de calcul.

- Optimum non garanti.

- Nécessité de réglage des paramètres.

- Difficulté de prévoir la performance.

Tout d'abord, rappelons qu'il s'agit des métaheuristiques dont l'objectif n'est pas le même que celui des méthodes exactes. En particulier, ces méthodes ne garantissent pas de trouver une solution exacte mais de trouver la solution optimale qui est la plus proche de la solution exacte.

L'optimisation par essaims particulaires (PSO: Particle Swarm Optimization) est l'une des métaheuristiques d'optimisation la plus utilisée, elle est proposée par Russel Eberhart (ingénieur en électricité) et James Kennedy (socio-psychologue) en 1995. Elle est basée sur le comportement des déplacements d'un essaim d'oiseaux ou de poissons pour la recherche de nourriture. Elle génère un essaim de particules dont chaque membre est une solution éventuelle du problème d'optimisation. Cet essaim vole dans l'espace de recherche (à n dimensions) et chaque membre de celui-ci est attiré par sa meilleure solution et celle de ses voisins. Chaque particule est donc dotée d'une mémoire contenant les données relatives à son vol (position, vitesse et sa meilleure solution au problème) ainsi que de la faculté de communiquer (ou socialiser) avec son entourage.

Cependant cet algorithme possède un inconvénient majeur qui est la convergence rapide qui va engendrer une optimisation local, pour cela plusieurs améliorations ont été faites sur l'algorithme de base pour augmenter ses performances.

L'objectif de ce travail est l'étude et l'application des algorithmes PSO dans le domaine du contrôle. On a essayé de commander un bras manipulateur à trois degrés de liberté en utilisant trois commandes basées sur le mode glissant, une commande classique, une commande par logique floue et une commande adaptative floue. Une fois les lois de commande développées, on a constaté que les performances de poursuite sont liées directement au choix des valeurs des paramètres ajustables utilisés. Pour ces raisons, on a utilisé trois algorithmes d'optimisation à base de PSO qui sont : le PSO classique, le PSO à convergence rapide (FCPSO :Fast convergence particle swarm optimisation en anglais) et le PSO hybride (GA-PSO) pour augmenter les performances de poursuite et pour déterminer le meilleur algorithme pour ce type de problèmes d'optimisation.

Outre l'introduction, ce mémoire est organisé en quatre chapitres répartis comme suit :

- Le premier chapitre est dédié à une présentation des différents modèles utilisés pour le mouvement des articulations d'un bras manipulateur.
- Le deuxième chapitre représente une description des algorithmes d'optimisation PSO
- Le troisième chapitre présente la description de la commande par mode glissant et l'application de cette commande sur un bras manipulateur à trois degrés de liberté. Des travaux de simulation concernant l'optimisation des paramètres de la commande avec les différents algorithmes cités précédemment ont été élaborés avec une présentation des différents résultats obtenus.
- Le quatrième chapitre présente la description de la commande par mode glissant floue et par mode glissant floue adaptatif avec une application de ces commandes sur notre bras manipulateur à trois degrés de liberté. Les résultats des travaux de simulation pour l'optimisation des paramètres ajustables des contrôleurs sont présentés.

On termine notre travail par une conclusion générale.

Chapitre I
Modélisation

I.1 Introduction :

Pour le développement d'une commande pour un manipulateur, il faut décrire les différentes relations mathématiques et donner un modèle mathématique précis, qui permet de définir les mouvements de ce dernier dans l'espace, pour connaitre la cinématique et la dynamique du manipulateur.

En général le modèle du manipulateur dépend de l'application envisagée : modèles géométriques, cinématiques ou dynamiques.

Dans le présent chapitre, on va présenter quelques définitions concernant ces modèles ainsi que les façons de leurs obtentions.

I.2 Généralités :

La structure mécanique du robot manipulateur contient deux parties distinctes comme montré dans la figure 1: une structure mécanique articulée (élément porteur) et un organe terminal. L'organe terminal est le dispositif d'interaction, fixé à l'extrémité mobile de l'élément porteur. Il est désigné par différentes appellations : effecteur, préhenseur, outil, organe terminal (OT) ou pince lorsque il s'agit d'une pince. L'élément porteur est un ensemble de corps, généralement rigides, liées par des articulations. Le rôle de l'élément porteur est de déplacer l'organe terminal d'une configuration à une autre, selon des requêtes spécifique de vitesse et d'accélération.

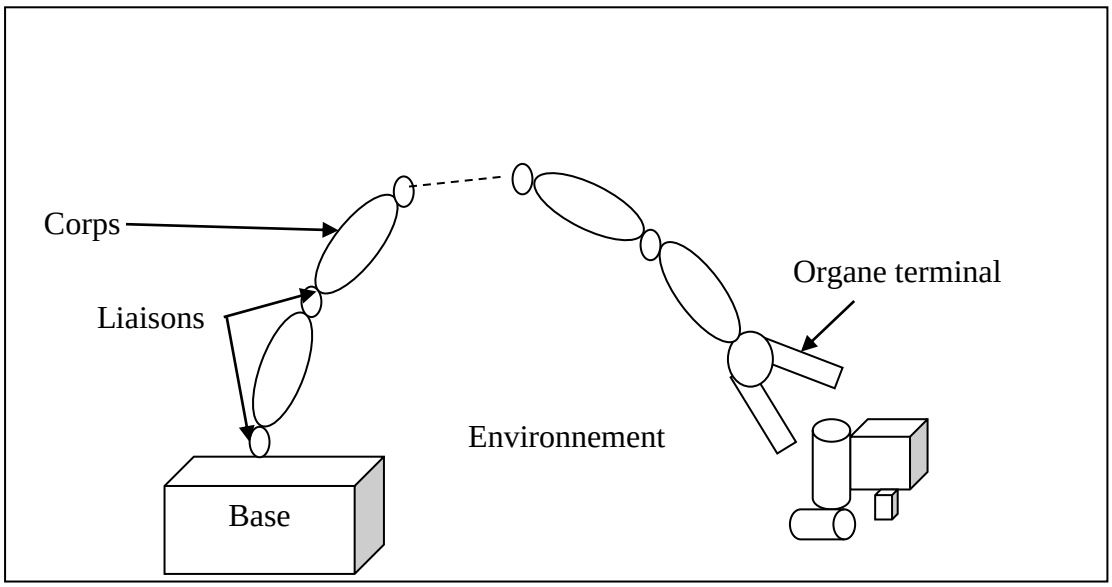


Figure I-1 : Robot à chaîne ouverte simple

❖ **Le degré de liberté**

Le degré de liberté représente le nombre de variables de positions (articulation) indépendantes, Le nombre d'articulation détermine le degré de liberté (ddl) du manipulateur. Pour pouvoir couvrir l'espace de travail, le manipulateur doit posséder au moins six ddl, trois pour le positionnement et trois pour l'orientation. Avec moins de six ddl, le manipulateur ne peut pas effectuer toutes les positions et orientations nécessaires dans son espace de travail. La difficulté de contrôler un manipulateur augmente rapidement avec le nombre

d'articulations. Un manipulateur ayant plus de six articulations est désigné sous le nom d'un manipulateur cinématiquement redondant [2].

❖ **Espace articulaire et opérationnel**

1- L'espace de configuration articulaire d'un robot manipulateur est la représentation de la situation de ses différentes articulations. On appelle coordonnées généralisées les variables ou coordonnées articulaires.

2- On appelle espace opérationnel un espace où est représenté l'emplacement de l'organe terminal. Les coordonnées permettant de définir l'emplacement de l'organe terminal sont appelées coordonnées opérationnelles [2].

I.3 Modélisation :

I.3.1 Modèle géométrique :

I.3.1.1 Modèle géométrique direct :

Le modèle géométrique direct (MGD) représente les différentes relations mathématiques qui permettent de déterminer les coordonnées opérationnelles du robot en fonction des coordonnées articulaires, c'est-à-dire d'exprimer la situation de l'organe terminal. Le MGD s'écrit sous la forme:

$$X = f(q) \tag{1.1}$$

q est le vecteur des variables articulaire tel que :

$$q = [q1\ q2\ \dots\dots\dots\ qn]T \tag{1.2}$$

❖ **Convention de Denavit-Hartenberg**

La géométrie des robots manipulateurs est décrite généralement en utilisant la convention de Denavit-Hartenberg pour la sélection des formes de référence. Dans cette convention, chaque transformation homogène T_i^{i-1}qui lie entre l'espace opérationnel et articulaire est représentée comme produit de quatre transformations de base [3] :

$$T_i^{i-1} = Trans_{z_{i-1},d_i} Rot_{z_{i-1},\theta_i} Trans_{x_{i-1},\alpha_i} R_{x_{i-1},\alpha_i} \tag{1.3}$$

$$T_i^{i-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos\theta_i & -sin\theta_i & 0 & 0 \\ sin\theta_i & cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\alpha_i & -sin\alpha_i & 0 \\ 0 & sin\alpha_i & cos\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.4}$$

$$T_i^{i-1} = \begin{bmatrix} cos\theta_i & -sin\theta_i cos\alpha_i & sin\theta_i sin\alpha_i & a_i cos\theta_i \\ sin\theta_i & cos\theta_i cos\alpha_i & -cos\theta_i sin\alpha_i & a_i\, sin\theta_i \\ 0 & sin\alpha_i & cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.5)$$

$$T_i^{i-1} = \begin{bmatrix} R_i^{i-1} & o_i^{i-1} \\ 0 & 1 \end{bmatrix} \quad (1.6)$$

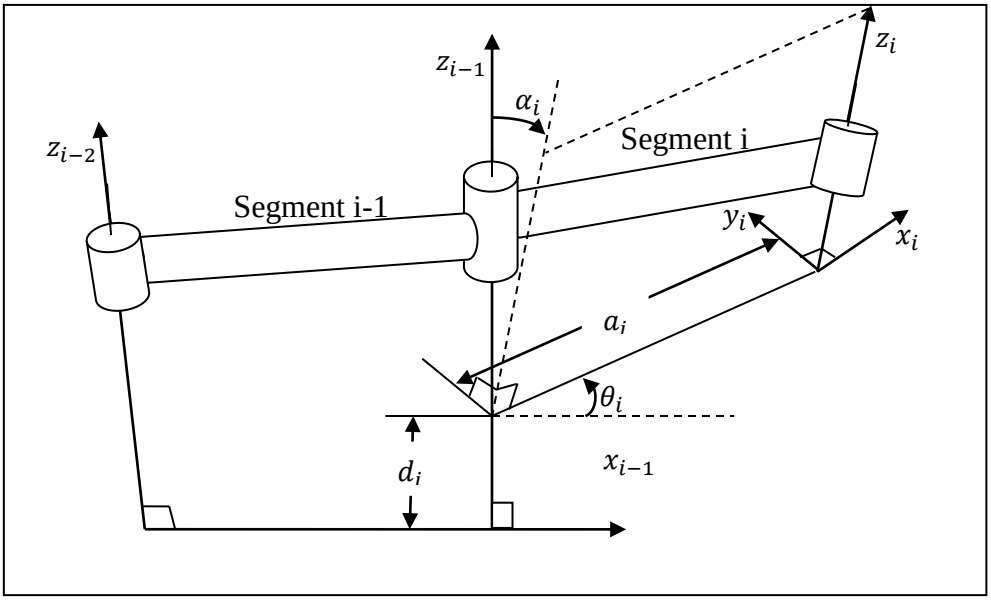


Figure 2 : Paramètres de Denavit-Hartenberg pour la liaison *i* et le segment *i*

Les paramètres de Denavit-Hartenberg sont alors :

- d_i : représente une translation le long de l'axe z_{i-1} entre les axes x_{i-1} et x_i.
- θ_i : représente l'angle de rotation autour de l'axe z_{i-1} entre les axesx_{i-1} et x_i.
- a_i : représente la distance minimale entre les axes z_{i-1} et z_i mesurer surx_i.
- α_i : représente l'angle de rotation autour de l'axe x_i entre les axes z_{i-1} et z_i.

La matrice R_i^{i-1} 3x3 formée des trois premières lignes et de trois premières colonnes représente l'orientation du repère R_{i-1} par rapport au repère R_i et le vecteur o_i^{i-1} constitué des trois premiers coefficients de la dernière colonne représente la position de l'origine O_{i-1}du repère R_{i-1} par rapport à l'origine $O_i$du repère R_i.

Le calcul du modèle géométrique direct consiste donc à exprimer la position du point O_{n+1} et l'orientation du repère R_n lié à l'organe terminal en fonction de sa configuration. Il faut pour cela multiplier les matrices de passage homogènes successives reliant le repère R_0 lié au bâti au repère R_n lié à l'organe terminal :

$$T_n^0(q) = T_1^0(q)\, T_2^1(q) \ldots\ldots T_n^{n-1}(q) \quad (1.7)$$

I.3.1.2 Modèle géométrique inverse

Le modèle géométrique direct permet le passage depuis les coordonnées articulaires du robot à la position de l'organe terminal de celui-ci. Le problème inverse est la détermination des coordonnées articulaires qui amènent l'organe terminal dans une position désirée en fonction de ses coordonnées opérationnelles. Alors à partir des coordonnées opérationnelles définies dans l'espace de la tâche, on utilise le modèle géométrique inverse pour définir les coordonnées articulaires.

Le modèle géométrique inverse est défini par la relation :

$$q = f^{-1}(X) \quad (1.8)$$

Après la détermination des coordonnées de la cible dans le repère de base, on calcule les valeurs des variables généralisées $q_1 = \theta_1, q_2 = \theta_2, q_3 = \theta_3$ qui amènent O_i au point visé. Il faut inverser les équations représentées par le système (1.1).

I.3.1.3 Problèmes du modèle géométrique inverse

- **Existence de solution**

L'existence de solution de l'équation $X = f(q)$ est conditionnée par le fait que l'effecteur évolue dans le domaine atteignable. Ce domaine est défini d'une part par des limitations dimensionnelles des éléments mécaniques formant le vecteur X et d'autre part d'éventuelles limitations structurelles.

a)limitation dimensionnelles

Ces limitations sont fonction :

- Des amplitudes finies des rotations et translations que peuvent réaliser les actionneurs.
- Des longueurs finies des segments constituant l'architecture mécanique du vecteur.

b) limitations structurelles

Ces limitations sont déterminées par des architectures particulières du vecteur X, ainsi que le nombre n d'actionneurs. Si ce dernier est inférieur ou égal à six le domaine atteignable est alors une variété de R^6 de dimension inférieure ou égale à n [4].

c) problèmes liés à l'inversion de la fonction f

Dans la majorité des cas, la fonction f est non linéaire et son inversion n'est possible que dans certaines régions du domaine atteignable.

Dans le cas où f est déterminée par une des méthodes citées précédemment, alors les problèmes suivants se posent :

- La solution analytique n'existe pas toujours et lorsqu'elle existe, son expression est souvent assez difficile à déterminer.
- Lorsque deux axes de rotation sont confondus ou lorsque deux axes de translation deviennent parallèles, l'organe terminal perd un degré de liberté.
- La prise en considération des butées articulaires et des obstacles est en général très difficile et nécessite l'élaboration d'algorithmes spéciaux.

I.3.2 Modèle cinématique :

I.3.2.1 Modèle cinématique directe :

Le modèle cinématique directe donne la relation entre les vitesses opérationnelles $\dot{X}$ et les vitesses généralisées q̇ du bras manipulateur :

$$\dot{X} = J(q)\dot{q} \tag{1.9}$$

En général, on prend la notation :

$$\dot{X} = \begin{bmatrix} \dot{X}_n & \dot{Y}_n & \dot{Z}_n & \omega_{Xn}^0 & \omega_{Yn}^0 & \omega_{Zn}^0 \end{bmatrix} \tag{1.10}$$

$$\dot{X} = \begin{bmatrix} v_n^0 \\ \omega_n^0 \end{bmatrix} = \begin{bmatrix} J_D \\ J_R \end{bmatrix} \dot{q} \tag{1.11}$$

Où :

-v_n^0 *et* ω_n^0 sont la vitesse linéaire et angulaire de $O_n$dans le mouvement de R_n par rapport à R_0

- $J(q)$ est la matrice Jacobienne de la fonction *f(q)* définie dans le repère *(X, Y, Z).*

-J_D et J_R sont la matrice Jacobienne de déplacement et de rotation.

Pour un manipulateur à *n* ddl on aura :

$$T_n^0 = \begin{bmatrix} R_n^0 & o_n^0 \\ 0 & 1 \end{bmatrix} \tag{1.12}$$

$$J_D = \begin{bmatrix} \frac{\partial o_{Xn}^0}{\partial q_1} & \frac{\partial o_{Xn}^0}{\partial q_2} \dots \frac{\partial o_{Xn}^0}{\partial q_n} \\ \frac{\partial o_{Yn}^0}{\partial q_1} & \frac{\partial o_{Yn}^0}{\partial q_2} \dots \frac{\partial o_{Yn}^0}{\partial q_n} \\ \frac{\partial o_{Zn}^0}{\partial q_1} & \frac{\partial o_{Zn}^0}{\partial q_2} \dots \frac{\partial o_{Zn}^0}{\partial q_n} \end{bmatrix} \tag{1.13}$$

$$J_R = \begin{bmatrix} \frac{\partial \omega_{Xn}^0}{\partial q_1} & \frac{\partial \omega_{Xn}^0}{\partial q_2} \dots \frac{\partial \omega_{Xn}^0}{\partial q_n} \\ \frac{\partial \omega_{Yn}^0}{\partial q_1} & \frac{\partial \omega_{Yn}^0}{\partial q_2} \dots \frac{\partial \omega_{Yn}^0}{\partial q_n} \\ \frac{\partial \omega_{Zn}^0}{\partial q_1} & \frac{\partial \omega_{Zn}^0}{\partial q_2} \dots \frac{\partial \omega_{Zn}^0}{\partial q_n} \end{bmatrix} \quad (1.14)$$

Tel que :

$$\omega_n^0 = \dot{R}_n^0 \, R_n^{0^T} \quad (1.15)$$

Le modèle cinématique peut être obtenu par la dérivation du modèle géométrique directe.
La matrice Jacobéenne peut être déduite par l'application de la loi de composition des vitesses [8].

I.3.2.2 Modèle cinématique inverse :

Le modèle cinématique inverse permet de déterminer la vitesse des variables articulaires en fonction de la vitesse des variables opérationnelles. Pour les manipulateurs non redondants (*n*=*m*), le modèle s'écrit [2] :

$$\dot{q} = J^{-1}(q)\dot{X} \quad (1.16)$$

I.3.3 Modèle dynamique

I.3.3.1 Modèle dynamique inverse

Le modèle dynamique inverse permet de déterminer la relation entre les couples articulaires exercés sur les actionneurs et les variables articulaires (positions, vitesses et accélérations). Le résultat est utile pour calculer la dynamique directe (modèle de robot) et certains contrôleurs (commande en avance, linéarisation par retour d'état etc.).
Pour établir le modèle dynamique inverse d'un robot manipulateur, on utilise le plus souvent les équations d'Euler-Lagrange pour le définir comme :

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = \tau_i \quad (1.17)$$

Avec :

- $L = K - V$ qui est appelé Lagrangien du système, K est l'énergie cinétique du système et V est l'énergie potentielle du système.

- q_i représente la ième coordonnée généralisée du système.
- τ_i est la force généralisée appliquée à l'ième élément du système. τ_i est un couple si l'articulation est rotoïde ou une force si l'articulation est prismatique.

L'énergie cinétique est donnée par :

$$K = \sum_{i=0}^{n} \frac{1}{2} m_i v_i^T v_i + \frac{1}{2} \omega_i^T I_i \omega_i \quad (1.18)$$

Avec :

I_i: Matrice d'inertie du corps *i*.

m_i: Masse du corps *i*.

v_i: vitesse linéaire du centre de gravité du corps *i*.

ω_i: vitesse angulaire du corps *i*.

Des équations (1.9-1.15) l'énergie cinétique du système peut s'écrire sous la forme :

$$K = \frac{1}{2}\, \dot{q}^T \sum_{i=1}^{n} [m_i J_{Di}^T(q) J_{vi}(q) + J_{\omega i}^T(q)\, R_{0,i} I_i R_{0,i}^T J_{Ri}(q) \dot{q} \quad (1.19)$$

La vitesse de rotation est exprimée dans le même repère que celui qui a servi à la détermination de la matrice d'inertie I_i.

L'équation (I.19) peut s'écrire sous forme :

$$K = \frac{1}{2} \dot{q}^T M(q) \dot{q} \quad (1.20)$$

L'énergie potentielle est donnée par :

$$V = g^T \sum_{i=1}^{n} O_{gi} m_i \quad (1.21)$$

Avec :

g : Vecteur de gravité

O_{gi}: Coordonnées de centre de gravité du corps i dans le repère de base.

On peut déduire les équations d'Euler-Lagrange en définissant le lagrangien comme suit :

$$L = K - V = \frac{1}{2} \sum_{i,j} d_{i,j}(q) \dot{q}_i \dot{q}_j - V(q) \quad (1.22)$$

On a :

$$\frac{\partial L}{\partial \dot{q}_k} = \sum_j d_{k,j}(q)\, \dot{q}_j \quad (1.23)$$

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{q}_k} = \sum_j d_{k,j}(q)\, \ddot{q}_j + \sum_j \frac{d}{dt} d_{k,j}(q)\, \dot{q}_j \quad (1.24)$$

$$\frac{d}{dt} \frac{\partial L}{\partial \dot{q}_k} = \sum_j d_{k,j}(q)\, \ddot{q}_j + \sum_{i,j} \frac{\partial d_{k,j}}{\partial q_i} \dot{q}_i\, \dot{q}_j \quad (1.25)$$

De même :

$$\frac{\partial L}{\partial q_k} = \frac{1}{2}\sum_{i,j}\frac{\partial d_{i,j}}{\partial q_k}\dot{q}_i\,\dot{q}_j - \frac{\partial V}{\partial q_k} \tag{1.26}$$

Ainsi, les équations d'Euler-Lagrange deviennent :

$$\sum_j d_{k,j}(q)\ddot{q}_j + \sum_{i,j}[\frac{\partial d_{k,j}}{\partial q_i} - \frac{1}{2}\frac{\partial d_{i,j}}{\partial q_k}]\dot{q}_i\dot{q}_j + \frac{\partial V}{\partial q_k} = \tau_k \tag{1.27}$$

En utilisant le fait que :

$$\sum_{i,j}\frac{\partial d_{k,j}}{\partial q_i}\dot{q}_i\dot{q}_j = \frac{1}{2}\sum_{i,j}[\frac{\partial d_{k,j}}{\partial q_i} + \frac{\partial d_{k,i}}{\partial q_j}]\dot{q}_i\dot{q}_j \tag{1.28}$$

On obtient :

$$\sum_{i,j}[\frac{\partial d_{k,j}}{\partial q_i} - \frac{1}{2}\frac{\partial d_{i,j}}{\partial q_k}]\dot{q}_i\dot{q}_j = \sum_{i,j}\frac{1}{2}\left[\frac{\partial d_{k,j}}{\partial q_i} + \frac{\partial d_{k,i}}{\partial q_j} - \frac{\partial d_{i,j}}{\partial q_k}\right]\dot{q}_i\dot{q}_j \tag{1.29}$$

On note :

$$c_{i,j,k} = \frac{1}{2}\left[\frac{\partial d_{k,j}}{\partial q_i} + \frac{\partial d_{k,i}}{\partial q_j} - \frac{\partial d_{i,j}}{\partial q_k}\right] \tag{1.30}$$

Et

$$\phi_k = \frac{\partial V}{\partial q_k} \tag{1.31}$$

Pour k fixe on a :

$$c_{i,j,k} = c_{j,i,k} \tag{1.32}$$

Finalement les équations d'Euler-Lagrange deviennent :

$$\sum_j d_{k,j}(q)\ddot{q}_j + \sum_{i,j} c_{i,j,k}(q)\,\dot{q}_i\dot{q}_j + \phi_k(q) = \tau_k \tag{1.33}$$

Ce qui peut s'écrire sous forme matricielle :

$$M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) = \tau \tag{1.34}$$

où :

$M(q)$: Matrice d'inertie du manipulateur.

$C(q,\dot{q})$: Vecteur de Coriolis et des couples centrifuges.

$G(q)$: Vecteur de gravité.

Dans $C(q,\dot{q})$ les termes impliquant un produit q_i^2 sont appelés centrifuges. Et ceux impliquant un produit $q_i q_j$ avec $i \neq j$ sont les termes de Coriolis.

I.3.3.2 Modèle dynamique direct

Le modèle dynamique direct détermine les variables articulaires (position, vitesse et accélération) en fonction du couple appliqué. Il est obtenu par inversion du modèle dynamique inverse, il s'écrit :

$$\ddot{q} = M(q)^{-1}[\tau - C(q,\dot{q})\dot{q} - G(q)] \quad (1.35)$$

I.3.3.3 Problèmes du modèle dynamique

Le développement de la technologie des ordinateurs et des logiciels de calcul peut corriger certains problèmes de modélisation tels que le risque d'erreur lors des calculs mais, il existe toujours d'autres problèmes. Le modèle d'un robot est conçu non seulement dans un but de simulation, mais surtout dans un but de commande. Dans ce dernier cas on se trouve face à trois types de problèmes :

- Le modèle doit être valide, c'est-à-dire suffisamment précis. Or, plus le comportement devient dynamique moins on va pouvoir négliger des effets considérés comme parasites.
- Le modèle dynamique étant essentiellement non linéaire, de nombreux coefficients, fonction de la configuration vont devoir être évalués en ligne ce qui pose un problème d'exécution en temps réel.

I.4 conclusion :

Dans ce chapitre on a présenté les différents types de modèles pour décrire les mouvements des articulations d'un robot manipulateur ainsi que les méthodes de l'obtention de ces modèles et on a vu que le choix du modèle dépend de l'application ou bien de l'objectif visé.

Dans ce travail on a choisi un modèle dynamique d'un bras manipulateur à 3ddl qui sera défini dans le chapitre trois, de plus la structure du manipulateur sera représenté dans l'annexe.

CHAPITRE II

Algorithmes d'optimisation

II.1 Introduction

L'optimisation par essaim de particules est une méthode métaheuristique, développée à partir de l'intelligence en groupe qui est basée sur le comportement du déplacement des oiseaux ou des poissons pour la recherche de la nourriture.

Lorsque les oiseaux sont à la recherche de nourriture, d'un endroit à l'autre, il y a toujours un oiseau qui peut sentir la nourriture et trouver l'endroit où elle peut être trouvée. Comme qu'il y'a un échange d'informations entres eux à tout moment, ils finiront par affluer vers l'endroit où la nourriture peut être trouvé. Une bonne information est égale à la solution la plus optimiste et la nourriture est égale à la solution la plus optimiste pendant tout le parcourt. L'algorithme proposé par Kennedy et Eberhart cherche à simuler ce comportement social basé sur l'analyse de l'environnement et du voisinage et constitue alors une méthode de recherche d'optimum par l'observation des tendances des individus voisins. Chaque individu cherche à optimiser ses chances en suivant une tendance qu'il modère par ses propres vécus. Le modèle qu'ils ont proposé à ensuite été étendu en un algorithme simple et efficace d'optimisation.

II.2 principe

L'optimisation par essaim de particules repose sur un ensemble d'individus originellement disposés de façon aléatoire et homogène, que nous appellerons dès lors des particules, qui se déplacent dans l'espace de recherche et constituent chacune une solution potentielle. Chaque particule dispose d'une mémoire concernant sa meilleure solution visitée ainsi que la capacité de communiquer avec les particules constituant son entourage. A partir de ces informations, la particule va suivre une tendance faite, d'une part, de sa volonté à retourner vers sa solution optimale, et d'autre part, de son mimétisme par rapport aux solutions trouvées dans son voisinage. A partir des optimums locaux et empiriques, l'ensemble des particules va normalement converger vers la solution optimale globale du problème traité [5] [10].

II.3 Formulation :

L'essaim de particules est constitué de n particules et la position de chaque particule représente une solution dans l'espace de recherche. Les particules changent d'état selon les trois principes suivants:

- Garder son inertie
- Changer d'état en fonction de sa position la plus optimiste
- Changer d'état selon la position la plus optimiste du groupe.

La position de chaque particule est affectée à la fois par la position la plus optimiste lors de son mouvement (expérience individuelle) et la position de la particule la plus optimiste dans

ses environs (expérience globale). La mise à jour de la position $x_i(t)$ et la vitesse $v_i(t)$ d'une particule p_i est représentée par les équations (2.1) et (2.2)

$$vi(t+1) = \omega vi(t) + c1r1\,[xpi(t) - xi(t)] + c2r2\,[g(t) - xi(t)] \quad (2.1)$$

$$xi(t+1) = xi(t) + vi(t+1) \quad (2.2)$$

où ω est l'inertie, c_1 et c_2 sont des coefficients constants fixés par l'utilisateur, r_1 et r_2 sont des nombres aléatoires dans la plage [0 1], tirés à chaque itération, $g(t)$ est la meilleure solution trouvée jusqu'à l'instant t et $x_{pi}(t)$ est la meilleure solution trouvée par la particule pi.
Soit $f(x)$ la fonction objectif à optimiser (fitness) et n le nombre de particules
Les étapes essentielles de l'optimisation par essaim de particules sont présentées par l'algorithme suivant :

1- Initialisation aléatoire de la population et des vitesses des particules.
2- Trouver la meilleure valeur de fitness f a t=0.
3- Traitement
   Répéter jusqu'à fin d'itération.
   Répéter pour chaque particule.
   Générer la nouvelle valeur de la vitesse en utilisant l'équation (2.1).
   Calculer la nouvelle position en utilisant l'équation (2.2).
   Evaluation de la valeur de fitness.
   Trouver la meilleure position pour chaque particule.
   Fin .
   Trouver la meilleure position globale
   Fin

En général le critère d'arrêt peut être un nombre d'itérations fixe ou bien en fonction de la fonction objective (fitness) ou bien lorsque les vitesses des particules tendent vers 0.

II.3 configuration des paramètres

II.3.1 Nombre de particules

Le nombre de particule utilisé pour la résolution du problème dépend essentiellement de deux facteurs, la taille de l'espace de recherche et le rapport entre les capacités de calcul de la machine et le temps maximum de recherche. Il n'y a pas de règle pour déterminer ce

paramètre, faire de nombreux essais permet de se doter de l'expérience nécessaire à l'appréhension de ce paramètre. En général le choix se fait aléatoirement.

II.3.2 Taille et topologie de voisinage

La topologie du voisinage constitue la structure du réseau social et définit avec qui chacune des particules va pouvoir communiquer. Il existe de nombreuses combinaisons dont les suivantes sont les plus utilisées (voir figure 1).

a) Topologie en étoile : chaque particule est reliée à toutes les autres, l'optimum du voisinage est l'optimum global.

b) Topologie en anneau: chaque particule est reliée à n particules (n = 3 en général), c'est la la plus utilisée.

c) Topologie en rayon : les particules ne communiquent qu'avec une seule particule centrale.

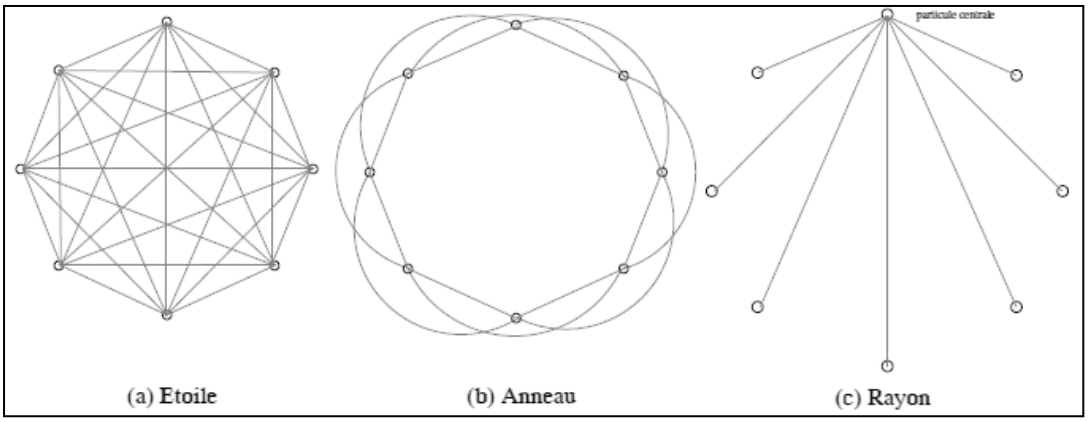


Figure II-1 : Topologies de voisinage

II.3.3 coefficients de confiance et coefficient d'inertie

Les coefficients c_1r_1 et c_2r_2 de l'équation (2.1) sont appelés coefficients de confiance, ils permettent de pondérer les tendances des particules à suivre leur instinct de conservation ou leur panurgisme. Les coefficients r_1, r_2 sont des variables aléatoires évaluées à chaque itération suivant une loi uniforme sur le domaine [0 1] et c_1, c_2 sont des constantes définies par la relation $c_1 + c_2 \leq 4$.

Le coefficient d'inertie appelé ω dans la formule vue auparavant permet de définir la capacité d'exploration de chaque particule en vue d'améliorer la convergence de la méthode.

Fixer ce paramètre revient à trouver un compromis entre une exploration globale (ω> 1) et une exploration locale (ω< 1). Il représente l'instinct aventureux de la particule.

II.4 Amélioration des algorithmes PSO

Malgré la simplicité et la facilité de l'utilisation de l'algorithme PSO il présente un risque de divergence ou de convergence rapide qui permet de stagner dans un optimum local, de ce fait plusieurs améliorations ont été apportées à l'algorithme de base.

II.4.1 Confinement des particules

Pour éviter que le déplacement des particules soit trop rapide, ce qui conduit à sortir de l'espace de recherche, nous pouvons introduire un nouveau paramètre V_{max}, qui permet de limiter la vitesse sur chaque dimension et ainsi de contrôler les particules.

Notons que cela ne restreint pas les valeurs de x_i à l'intervalle $[Vi_{min},Vi_{max}]$, mais limite seulement la distance maximale qu'une particule va parcourir au cours d'une itération. Cette méthode permet de contrôler la divergence de l'algorithme et de réaliser ainsi un compromis efficace entre intensification et diversification.

De plus, une stratégie de confinement des particules peut être introduite. Une telle stratégie permet de ramener une particule sortie de l'espace de recherche à l'intérieur de celui-ci. Dans ce cadre, plusieurs méthodes peuvent être employées :

- La particule est laissée à l'extérieur de l'espace de recherche, mais on n'évalue pas sa fonction objective. Ainsi, elle ne pourra pas attirer les autres particules en dehors de l'espace de recherche.
- La particule est arrêtée à la frontière et les composantes associées à sa vitesse sont annulées.
- La particule rebondit sur la frontière, elle est stoppée à la frontière mais les composantes correspondantes de la vitesse sont multipliées par un coefficient tiré aléatoirement dans l'intervalle [-1,0].

II.4.2 Coefficient de constriction

L'étude de la dynamique des particules au sein de l'essaim a conduit à la recherche de solutions pour éviter la divergence de l'algorithme, comme par exemple l'introduction du paramètre V_{max} que nous avons vu dans le paragraphe précédent et qui permet de limiter la

divergence des particules. De nombreuses études ont été menées sur la dynamique des particules concernant l'analyse des conditions de convergence de l'essaim.

La combinaison des paramètres ω, c_1 et c_2 permet de régler l'équilibre entre les phases de diversification et d'intensification du processus de recherche. L'utilisation d'un coefficient de constriction χ (ou facteur de constriction) permet de mieux contrôler la divergence de l'essaim et de s'affranchir de la définition de V_{max}. Cette variante de PSO, qui a été largement utilisée dans la littérature, est connue sous le nom de canonical PSO [6]. En utilisant ce coefficient, l'équation (2.1) devient :

$$vi(t+1) = \chi\,(vi(t) + \varphi 1 r1\,[xpi(t) - xi(t)] + \varphi 2 r2\,[g(t) - xi(t)]) \qquad (2.3)$$

Avec :

$$\chi = \frac{2}{\Phi - 2 + \sqrt{\Phi^2 - 4\Phi}} \qquad (2.4)$$

où : φ_{1+} $\varphi_{1=}$ Φ *et* $\Phi > 4$

II.4.3 Coefficient d'inertie

Le coefficient d'inertie ω, introduit par Y.Shi et Eberhart [7], contrôle l'influence de la direction de la particule sur le déplacement futur. Le but de l'introduction de ce paramètre est de réaliser un équilibre entre la recherche locale (exploitation) et la recherche globale (exploration). L'intensité de l'exploration de l'espace de recherche dépend de la valeur du poids d'inertie, une grande valeur de ω facilite l'exploration globale, alors qu'une petite valeur facilite l'exploration locale. Du fait de son influence sur les performances de l'algorithme PSO, le poids d'inertie a suscité un grand intérêt de la part de la communauté des chercheurs. Dans [7], les auteurs ont proposé un coefficient d'inertie dynamique qui varie au cours du temps. Il commence par une valeur proche de 0,9 et descend linéairement pour arriver à 0,4. Cette stratégie a beaucoup amélioré les performances du PSO dans plusieurs problèmes d'optimisation. Le coefficient d'inertie ω varie linéairement avec le temps selon la formule suivante :

$$\omega = \omega\,min + (\omega\,max - \omega\,min)\,\frac{iter}{iter\,max} \qquad (2.5)$$

où *iter* est l'itération courante et *itermax* est le nombre maximal d'itérations. Les coefficients ωmax et ωmin désignent respectivement les valeurs maximum et minimum du coefficient ω. Dans une autre variante Y.Shi et Eberhart ont proposé une valeur du coefficient d'inertie choisie au hasard selon une distribution uniforme telle que $\omega \in [0.5\ 1]$ [6].

II.4.4 Stratégie Fully Informed Particle Swarm (FIPS)

Cette approche présente une nouvelle manière d'utiliser la topologie g (meilleur global), appelée FIPS. Elle utilise une partie des informations de chaque voisine, au lieu de se baser seulement sur les informations de la meilleure voisine et la meilleure expérience propre à la particule. Par conséquent, l'utilisation d'une topologie entièrement connectée ne signifie pas que l'information utilisée soit seulement la meilleure solution trouvée par l'essaim. En effet, dans FIPS, elle est toujours utilisée, mais elle n'est pas la seule. Pour les algorithmes basés sur le principe de la topologie FIPS, l'information utilisée pour déplacer les particules est issue de toutes les autres particules.

Ainsi, toutes les voisines contribuent à l'ajustement de la vitesse d'une particule :

$$v_i^{t+1} = \chi \left[v_i^t + \sum_{n=1}^{N_i} \frac{U(0,\Phi)(P_{nbr(n)}^t - x_i^t)}{N_i} \right] \qquad (2.6)$$

où N_i est le nombre de voisins de la particule *i*, $P_{nbr(n)}^t$ est la nième particule voisine de la particule *i* et Φ est la constante d'accélération qui permet de contrôler la convergence des particules [6].

II.4.5 Algorithme TRIBES

TRIBES est un algorithme d'optimisation par essaim particulaire sans paramètres de contrôle, cet algorithme présente la particularité d'être totalement adaptatif, c'est-à-dire que tous les paramètres de contrôle sont calculés de manière autonome par l'algorithme. En effet, TRIBES est défini comme une boîte noire, pour laquelle l'utilisateur n'a plus aucun paramètre à régler. Il doit seulement définir le problème à résoudre (i.e. la fonction objective, l'espace de recherche et les contraintes), ainsi que le critère d'arrêt. Cependant, il est à signaler que TRIBES ne peut pas résoudre tous les problèmes. De plus, ses résultats sont probabilistes à cause de son caractère stochastique.

Le but de TRIBES, d'après son auteur, est d'être efficace dans la plupart des cas et de permettre à ses utilisateurs de gagner du temps, en évitant l'étape de réglage de la métaheuristique.

Dans TRIBES, l'essaim particulaire est divisé en plusieurs sous-essaims appelés « tribus ». Les tribus sont de tailles différentes qui évoluent au cours du temps. Le but est d'explorer simultanément plusieurs régions de l'espace de recherche, généralement des optima locaux, avant de prendre une décision globale. Dans le but de prendre une telle décision, les tribus échangent leurs résultats tout au long du traitement. Deux types de communications sont donc à définir, la communication intra-tribu et la communication inter-tribus. Chaque tribu est composée d'un nombre variable de particules. En effet, une tribu qui peine à améliorer ses résultats génère des particules plus exploratrices. Les particules générées par les différentes tribus forment une nouvelle tribu, qui reste en communication avec ses génitrices. Inversement, une tribu efficace tendra à supprimer celles de ses particules qui n'ont pas contribué à sa bonne performance.

TRIBES est un algorithme compétitif, qui permet de trouver rapidement des optima locaux (très utile pour l'optimisation dynamique). Cependant les particules ont tendance à rester dans ces optima locaux et ont du mal à en sortir [6].

Deux idées sont proposées qui permettent d'améliorer les performances de cet algorithme :

– La première idée consiste à utiliser un nouveau mode d'initialisation (initialisation régulière) pour assurer une couverture plus uniforme de l'espace de recherche par les particules. En pratique, les particules sont initialisées de manière à être les plus éloignées possible les unes des autres et les plus éloignées possible des frontières de l'espace de recherche.

– La deuxième idée consiste à utiliser une nouvelle stratégie de déplacement, basée sur une hybridation avec un algorithme à estimation de distribution, pour maintenir la diversité au sein de l'essaim, tout au long du traitement.

II.4.6 optimisation par essaim de particules à convergence rapides

L'optimisation par essaim de particules rapide (FCPSO : fast convergence particle swarm optimiation en anglais) est basée sur l'équilibre entre la diversité du lieu de la particule individuelle en introduisant un nouveau paramètre qui est la dimension particulaire moyenne (PMD) de toutes les particules qui peut améliorer les performances du PSO.

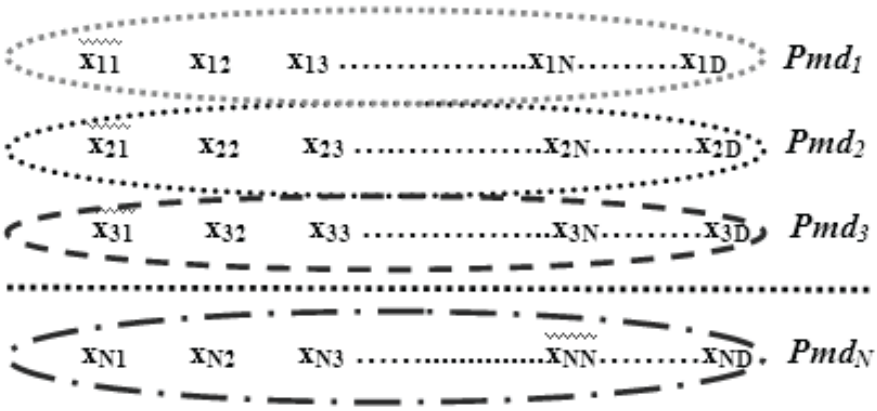


Figure II-2 : Calcul de la dimension particulaire moyenne pour N particules [8]

Après les mises à jour de l'essaim de particules, à partir de la génération de t à *t + 1* en dehors de la suite de la x_{pi} et g, la particule pourrait suivre qui choisir parmi l'essaim de particules. Le troisième paramètre *PMDi* de particules i^{eme} et l'équation de vitesse sont générés par les équations suivantes:

$$PMDi = (xi1 + xi2 + \dots\dots\dots\dots\dots\dots + xiD)/\mathrm{D} \quad (2.7)$$

$$vi(t+1) = \omega vi(t) + c1r1\,[xpi(t) - xi(t)] + c2r2\,[g(t) - xi(t)] + c3r3[PMDi(t) - xi(t)] \quad (2.8)$$

où c_3 est un coefficient constant fixé par l'utilisateur tel que : $c_1 + c_2 + c_3 \geq 4$.

r_3 est un nombre aléatoire dans la plage [0 1] tiré à chaque itération.

Après l'implantation du *PMDi* à la formule de vitesse, x_{pi}, g et *PMDi* fournissent des informations à la prochaine génération et augmentent ainsi la quantité d'informations. Par conséquent, il est possible de trouver la solution optimum rapidement. La position de g est utilisée pour améliorer le taux de convergence, mais réduit la diversité de la population qui mène à un optimum local. Dans le même temps le nouveau paramètre ajouté *PMDi* peut déplacer les particules à un meilleur emplacement et affaiblir l'attraction de g vers les optimums locaux.

II.5 hybridation de PSO avec algorithme génétique

L'hybridation des algorithmes d'optimisation est une tendance observée ces dernières années, elle a attiré l'attention de nombreux chercheurs afin d'améliorer les performances. L'objectif de l'hybridation consiste à combiner les caractéristiques de deux différentes méthodes pour tirer les meilleurs avantages. Dans cette partie on va étudier un algorithme

hybride GA-PSO proposée par Hanaa Hachimi [9] pour améliorer le temps de calcul et éviter le risque des optimums locaux.

II.5.1 algorithme génétique

Les algorithmes génétiques sont des techniques de recherche inspirées par l'évolution naturelle des êtres vivants et des mécanismes d'évolution: croisements, mutations, sélections, ….etc. Les algorithmes génétiques cherchent à optimiser une fonction objective (fitness) à partir de l'évaluation d'une population. Les meilleurs individus sont choisis pour constituer les parents appropriés donnant naissance à de meilleures descendances enfants, parmi lesquelles seront tirées les solutions acceptables du problème traité.

Un algorithme génétique se résume par les étapes suivantes :

1. **Codage et population initiale:** Un aspect important des algorithmes génétiques est la façon dont sont codées toutes les solutions. Les algorithmes génétiques établissent une analogie entre l'ensemble de solutions d'un problème et l'ensemble d'individus d'une population naturelle, en codant l'information sur chaque solution. En fonction du problème étudié, les codes utilisés peuvent être binaires ou réels.

   L'AG démarre avec une population initiale d'individus dans le codage retenu. Le choix des individus conditionne fortement la rapidité de l'algorithme.

   Si la position de l'optimum dans l'espace de recherche est totalement inconnue, il est intéressant que la population soit répartie sur tout l'espace de recherche. Si par contre des informations à priori sur le problème sont disponibles, il paraît évident de générer les individus dans un espace particulier afin d'accélérer la convergence. Disposant d'une population initiale souvent non homogène, la diversité de la population doit être entretenue aux cours des générations afin d'explorer le plus largement possible l'espace de recherche.
2. **L'évaluation de la population** : Cette étape consiste à évaluer chaque solution contenue dans la population, la mesure de performance des solutions s'appuie sur la valeur de la fonction objective. Cette étape permet de classer les solutions, afin de déterminer les solutions qui seront sélectionnées pour construire une nouvelle population de solutions.
3. **La sélection :** La sélection a pour objectif d'identifier les individus qui doivent se reproduire. Cet opérateur ne crée pas de nouveaux individus mais identifie les individus sur la base de leur fonction d'adaptation, les individus les mieux adaptés sont sélectionnés alors que les moins bien adaptés sont écartés. La probabilité de sélectionner un individu

donné est souvent traduite par le rapport entre la valeur de sa fonction d'adaptation et la somme de toutes les fonctions d'adaptation de la population. Il existe plusieurs techniques de sélection, nous en développons trois : la sélection par roulette biaisée (roulette wheel selection), la sélection par tournoi (tournement selection) et la sélection par rang (ranking selection).

4. **La création de nouveaux individus :** une fois la sélection faite, la création des nouveaux individus passe par les étapes suivantes :
   - **Le croisement :** processus où de nouveaux individus sont formés à partir des parents. Ces nouveaux individus, les rejetons, sont formés en effectuant un croisement entre deux parents. On choisit une position aléatoire k entre [1; l+1] ou l est la longueur de l'individu. Le croisement se fait en échangeant les bits de la position *k + 1 à l.*
     - **Exemple** : Soit $k = 4$ pour deux parents ($P1$ et $P2$) codés à cinq bits (donc $l = 5$). Les enfants sont $O1$ et $O2$.

     P1 = 0110|1 P2 = 1100|0

     O1 = 01100 O2 = 11001

     On voit bien l'échange qui s'est produit ici, le bit 5 (k + 1) a passé d'un individu à l'autre, pour ainsi former deux nouveaux individus (O1 and O2).

     Ce sont ces deux opérations, la sélection et la reproduction, qui sont à la base des algorithmes génétiques. Ceci peut paraître simple à première vue, puisque aucune opération mathématique complexe n'a été effectuée. Mais on peut comparer le processus précédent à l'innovation humaine, souvent les découvertes n'arrivent pas par chance. Elles sont le résultat d'un échange d'idées qui crée d'autres idées et finalement mènent à une solution désirée.
   - **La mutation :** processus aléatoire où un bit change de valeur. Ce processus joue un rôle secondaire dans l'algorithme génétique, mais il est quand même important. La mutation n'assure qu'aucun point dans l'espace de recherche à une probabilité nulle d'être atteint.

L'algorithme génétique détaillé est représenté ci-dessous :

1- Choisir un nombre de paramètre N pour représenter le problème, et déterminer l'intervalle de possibilité (*feasible range*) pour chaque paramètre :

$\{x_{1\min}, x_{1\max}\}, \{x_{2\min}, x_{2\max}\}, ..., \{x_{N\min}, x_{N\max}\}$,

Définir une variance pour chaque paramètre et la fonction à être optimisée.

2- Mettre une valeur initiale à chaque paramètre dans leur intervalle de possibilité respectif. Cet ensemble de valeurs constitue la population initiale de paramètres « parents ».

$x_1, x_2, ..., x_N$

3- Calculer la solution associée avec les paramètres « parents ».

$X = f(x_1, x_2, ..., x_N)$

4- Créer un descendant en additionnant une variable normalement distribuée *a* de moyenne zéro et de variance choisie (à l'étape 1) δ, pour chaque paramètre « parent ».

$x_i' = x_i + a(0, \delta), \quad i = 1, 2, ..., N$

Les mutations distribuées normalement de moyenne zéro reflètent le processus naturel de l'évolution où les changements plus petits arrivent plus fréquemment que les grands changements.

5- Calculer la solution associée avec les paramètres du « descendant ».

$X' = f(x_1', x_2', ..., x_N')$

6- Comparer la solution « descendant » avec la solution « parent ». Si la solution du « descendant » est meilleure que celle du « parent », remplacer la population de paramètres « parents » par la population de paramètres « descendant ». Sinon, garder la population de paramètres « parent ».

7- Retourne à l'étape 4 et répéter le processus jusqu'à ce qu'une solution satisfaisante soit obtenue, ou qu'un certain nombre de générations soit atteint.

II.5.2 algorithme GA-PSO

L'inconvénient de l'algorithme génétique est un peu lent quand la taille de la population devient grande. En effet, l'idée d'hybridation en insertion est venue pour minimiser le temps de calcul et aussi pourquoi ne pas améliorer les résultats de la fonction objective.

Cependant nous avons choisi d'hybrider les deux algorithmes heuristiques GA et PSO en insérant le PSO dans le GA. En d'autre terme à la place de faire la dernière étape de la mutation on la remplace par les mécanismes du PSO [9].

L'idée est de remplacer l'étape de mutation citée précédemment dans l'algorithme génétique par les mécanismes du PSO, la démarche de cet algorithme est présentée par les étapes suivantes :

- Initialisation et sélection.

-Evaluation.

-Croisement.

- Processus PSO : mise à jour des vitesses et des positions des particules
- Retour à l'étape d'évaluation jusqu'à la convergence du processus.

II.6 Conclusion

Dans ce chapitre on a présenté les différents algorithmes d'optimisation par essaim de particules qui ont rencontré un succès remarquable depuis leur création, grâce à leur simplicité et facilité d'implémentation des différents traitements sans que l'utilisateur ait à modifier la structure de base de l'algorithme.

On a remarqué que l'algorithme PSO présente un problème majeur, qui est le problème de la convergence rapide, ce dernier permet de trouver un optimum local et dans ce sens on a présenté plusieurs améliorations sur cette technique pour améliorer les performances et éviter ce cas de problèmes.

Dans les chapitres suivants on va utiliser trois algorithmes qui sont PSO, FCPSO et GA-PSO pour optimiser les paramètres des commandes utilisées pour le contrôle du bras manipulateur.

Chapitre III

La commande par mode glissant

III.1 Introduction

Les commandes à structures variables sont des techniques proposées pour la commande des systèmes non linéaires ou ayant des paramètres non constants et où les lois de commande classique ne peuvent être suffisantes, car elles ne remplissent pas les exigences de robustesse, précision et autres caractéristiques dynamiques des systèmes.

Ces techniques sont apparues suite aux travaux du mathématicien russe A. Fillipov sur la résolution des équations différentielles à second membre discontinu et les recherches de S. Emelyanov en 1967 et de V. Utkin en 1977, la commande par mode glissant étant l'une de ces techniques. L'importance de cette technique réside dans la réponse dynamique rapide, la stabilité, la simplicité de conception et implantation et la robustesse vis-à-vis de la variation des paramètres internes ou externes. Il existe trois types de configurations pour la synthèse de la loi de commande par mode glissant, la première basée sur le changement de structure par commutation d'une contre réaction des variables d'état, la deuxième change de structure au niveau de l'organe de commande et la troisième configuration change aussi de structure au niveau de l'organe de commande mais avec l'ajout d'une commande secondaire dite « commande équivalente ».
Dans ce travail on s'intéresse à la configuration de la commande par mode glissant avec commande équivalente. On va donc donner une description théorique de cette technique et développer une loi de commande pour un manipulateur à trois degrés de liberté pour ensuite optimiser les paramètres de cette commande et présenter les différents résultats de simulation obtenus.

III.2 Principe

La commande par mode glissant repose sur le principe d'amener en premier temps la dynamique du système à atteindre une surface prédéfinie, appelée surface de glissement, une fois arrivée à cette surface, elle glisse vers le point d'équilibre. La première phase est appelée mode de convergence et la deuxième c'est le mode de glissement [10] comme le montre la figure 1.

..

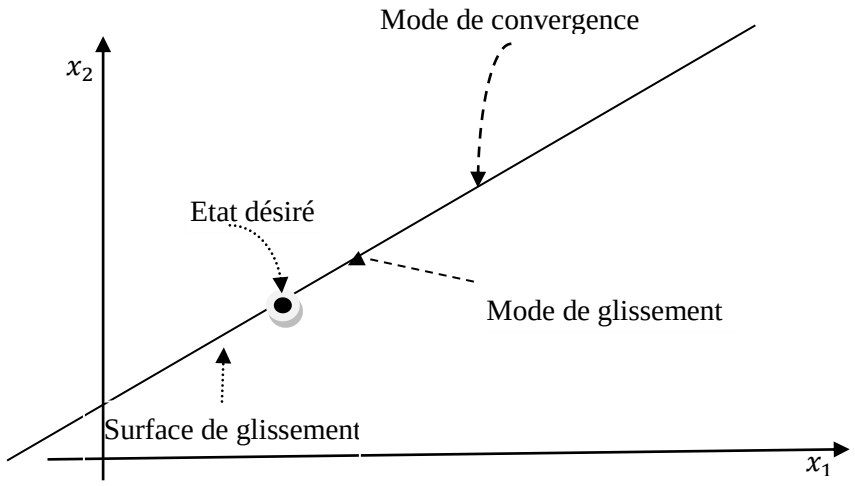


Figure III-1 : Convergence vers la surface de glissement

III.3 Synthèse de la loi de commande

La synthèse de la commande par modes glissants se fait en trois étapes :

- choix de la surface de glissement
- Etablissement de la condition de convergence
- synthèse de la loi de commande

III.3.1 Choix de la surface de glissement

On considère le système suivant :

$$\dot{x} = f(x) + g(x) \tag{3.1}$$

où *f* et *g* sont des fonctions non linéaires, *g* est supposée inversible.

u: l'entrée du système.

x: l'état du système.

Soit x_d la consigne désirée et *e* l'erreur de poursuite définie par :

$$e = x - xd \tag{3.2}$$

La formule générale de la surface de glissement est définie en fonction de l'ordre du système comme suit :

$$s = \left(\frac{d}{dt} + \lambda\right)^{n-1} e(t) \tag{3.3}$$

où : λ est une constante positive

n est le degré relatif, égal au nombre de fois qu'il faut dériver la sortie y pour faire apparaître la commande *u*.

III.3.2 condition d'existence et convergence

Une fois le choix de la fonction de glissement étant fait, l'étape prochaine consiste à déterminer une loi de commande qui puisse forcer le vecteur d'état *x* à converger vers la surface et y demeurer (s=O). Pour cela, il faut que la loi de commande soit conçue de telle manière à ce que la surface *S* soit attractive.

Pour déterminer la condition d'attractivité, considérons la fonction de Lyapounov suivante :

$$V(x) = \frac{1}{2} s^T(x) s(x) \tag{3.4}$$

Pour que le système soit stable il faut assurer une condition nécessaire et suffisante tel que :

$$\dot{V}(x) = s(x)\dot{s}(x) < 0 \tag{3.5}$$

L'équation (3.5) assure la convergence vers la surface de glissement

Le temps de convergence (raching time) dépend directement du choix de la surface du glissement *S*.

III.3.3 la loi de commande

La loi de commande par mode glissant (figure 2) est définie par la relation suivante :

$$u = u_{eq} + u_c \tag{3.6}$$

où u_d est une composante discontinue qui va forcer la trajectoire du système à atteindre la surface de glissement et de rester au voisinage de celle-ci malgré la présence de perturbations.

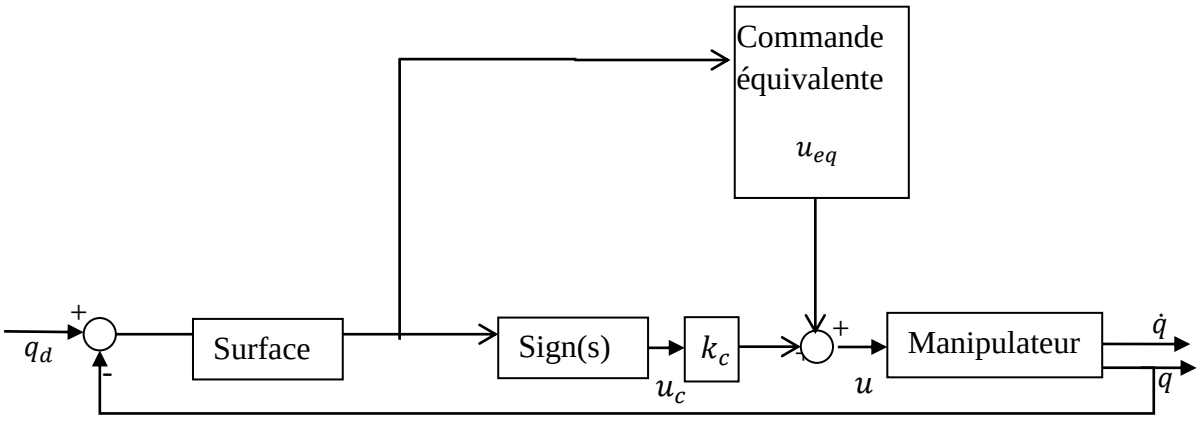


Figure III-2 : Principe de la commande par mode glissant avec commande équivalente

La composante u_c est définie par l'équation suivante :

$$u_c = -k\ sign(s) \tag{3.7}$$

où *k* est une constante positive

$$\text{sign(s)} = \begin{cases} +1 \ \text{si s} > 0 \\ 0 \ \text{si s} = 0 \\ -1 \text{si s} < 0 \end{cases} \tag{3.8}$$

La composante u_{eq} est une fonction continue qui permet de garder la dynamique du système sur la surface de glissement, elle est obtenue de façon unique par l'équation suivante :

$$\dot{s}(x) = 0 \tag{3.9}$$

L'exemple suivant [10] montre comment obtenir cette commande.

On considère le système suivant :

$$\ddot{x} = f + u \tag{3.10}$$

$$e = x - x_d \Longrightarrow s = \left(\frac{d}{dt} + \lambda\right)^1 e(t) = \dot{e} + \lambda e \tag{3.11}$$

$$\Rightarrow \dot{s} = \ddot{e} + \lambda\dot{e} = \ddot{x} - \ddot{x}_d + \lambda\dot{e} = f + u - \ddot{x}_d + \lambda\dot{e} \tag{3.12}$$

$$\dot{s} = 0 \ \Longrightarrow f + u_{eq} - \ddot{x}_d + \lambda\dot{e} = 0 \tag{3.13}$$

$$u_{eq} = \ddot{x}_d - f - \lambda\dot{e} \tag{3.14}$$

III.4 Application du mode glissant

Dans cette partie on va appliquer la commande par mode glissant pour contraindre un bras manipulateur à trois degrés de liberté à suivre une trajectoire désirée présentée par le vecteur $\theta(t) = [\theta_1(t)\ \theta_2(t)\ \theta_3(t)]^T$.

La dynamique du manipulateur est définie par l'équation (3.15)

$$M(\theta)\ddot{\theta} + C\left(\theta, \dot{\theta}\right)\dot{\theta} + G(\theta) = U + P \tag{3.15}$$

Tel que :

$\theta, \dot{\theta}$et $\ddot{\theta}$: Vecteur n×1 des variables articulaires et de leurs dérivées.

$M(\theta)$: Matrice n×n d'inertie du manipulateur.

$C\left(\theta, \dot{\theta}\right)$: Vecteur n×1de Coriolis et des couples centrifuges

$G(\theta)$: Vecteur n×1de gravité.

P : Vecteur n×1 de bruit externe.

U : Vecteur n×1 de couple et/ou force

$$\ddot{\theta} = M(\theta)^{-1}\,(U + P - C(\theta,\dot{\theta})\dot{\theta} - G(\theta)\,) \tag{3.16}$$

Représentation d'états :

Posant $x_1 = \theta$, $x_2 = \dot{x}_1$ et y= x_1

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = M(\theta)^{-1}\left(U + P - C(\theta,\dot{\theta})\dot{\theta} - G(\theta)\right) \\ y = x_1 \end{cases} \tag{3.17}$$

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = g_a\,U + f_a + b \\ y = x_1 \end{cases} \tag{3.18}$$

Tel que :

$$f_a = -M(\theta)^{-1}\left(C(\theta,\dot{\theta})\dot{\theta} + G(\theta)\right) \tag{3.19}$$

$$g_a = M(\theta)^{-1} \tag{3.20}$$

$$b = M(\theta)^{-1}P \tag{3.21}$$

Avec : $|b| < \alpha$, α représente la valeur maximale de la perturbation .

Pour un manipulateur à trois degrés de liberté on prend n= 03.

Pour la conception de la loi de commande la première étape est de choisir la surface de glissement, dans ce travail on a choisi un PID qui va définir les performances du système, ensuite, on procède à la conception de la loi qui va assurer la convergence de la réponse vers cette surface.

L'architecture du système est représentée sur la figure suivante :

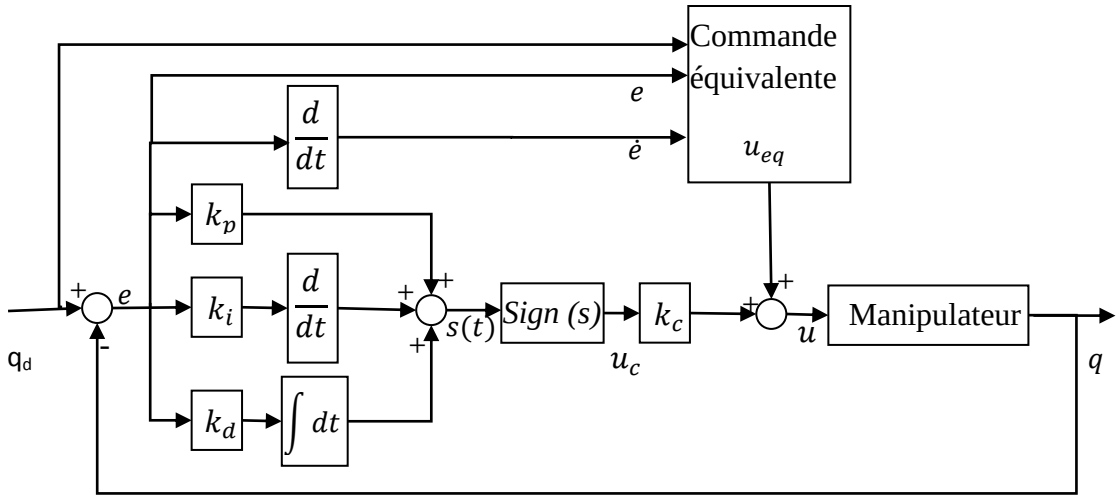


Figure III-3 : Schéma bloc de la commande par mode glissant avec une surface de glissement PID

Posons l'erreur de système $e = \theta - \theta_d$.

On étudie la stabilité du système sur S=0 l'expression de la surface est de la forme

$$S = \left(\frac{d}{dt} + \lambda\right)^{n-1} \left(\int_0^t e(t)dt\right) \tag{3.22}$$

D'autre part

$$S(t) = k_p e(t) + k_i \int e(t)dt + k_d \frac{d}{dt} e(t) \tag{3.23}$$

Où k_p,k_i et k_d ce sont des gains de valeur positive.

Et on continue, en cherchant la dérivé de la surface et étudier la stabilité

$$\dot{S}(t) = k_p \dot{e} + k_i e + k_d \ddot{e} \tag{3.24}$$

$$\dot{S}(t) = k_\mathrm{p} \dot{e} + k_i e + k_d \left[\ddot{\theta} - \ddot{\theta}_d\right] \tag{3.25}$$

$$\dot{S}(t) = S\left[k_p \dot{e} + k_i e + k_d \left[-\ddot{\theta}_d + g_a u + f_a + b\right]\right] \tag{3.26}$$

$$u_{eq=} \ (k_d g_a)^{-1} \left[\ddot{\theta}_d + f_a + b - k_p \dot{e} - k_i e\right] \tag{3.27}$$

Cette commande équivalente ne peut pas assurer les performances voulues avec la présence de perturbations et/ou d'incertitudes paramétriques, alors il faut ajouter un signal de commande qui va éliminer l'effet des perturbations, en assurant la stabilité et la robustesse du système. Pour cela, il faut élaborer une fonction de Lyapunov et faire une étude de stabilité.

On prend comme fonction de Lyapunov $V = \frac{1}{2} s^2$, avec la drivée

$$\dot{V} = s\dot{s} = s\left[k_p \dot{e} + k_i e + k_d \ddot{e}\right] \tag{3.28}$$

$$\dot{V} = s[k_p \dot{e} + k_i e + k_d \left[-\ddot{\theta}_d + g_a u + f_a + b\right]] \tag{3.29}$$

On choisit :

$$u = u_{eq} + u_c \tag{3.30}$$

Substituant u_{eq} par sa valeur (2.27) dans (2.29) on obtient :

$$\dot{V} = s[k_d g_a u_c + k_d b] \tag{3.31}$$

$$\dot{V} \leq |s|\, k_d\, |b| + s k_d g_a u_c \tag{3.32}$$

Avec $|b| < \alpha$, ce qui donne :

$$\dot{V} \leq |s| k_d \alpha + s k_d g_a u_c \tag{3.33}$$

Pour assurer que $\dot{V} \leq 0$, il faut que $k_c \geq g_a^{-1}\alpha$, ainsi le système va être attiré vers la surface s,

III.5 simulation

Tout au long de ce mémoire on va simuler les algorithmes et les commandes étudiés sur un modèle de bras manipulateur à 3 ddl en utilisant le logiciel de simulation Matlab.

L'architecture du système est représentée sur la figure 4

Le modèle dynamique utilisé est présenté dans l'annexe,

Les matrices k_p, k_i, k_det k_c sont choisies de façon automatique à l'aide des algorithmes d'optimisation PSO, CFPSO et GA-PSO étudiés précédemment.

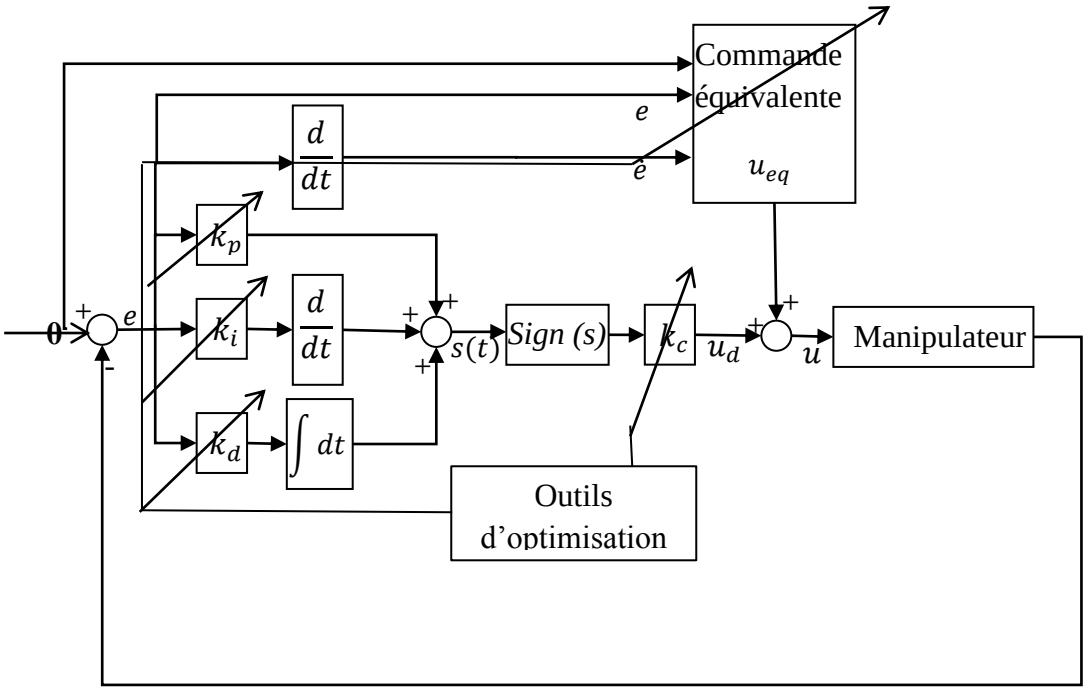
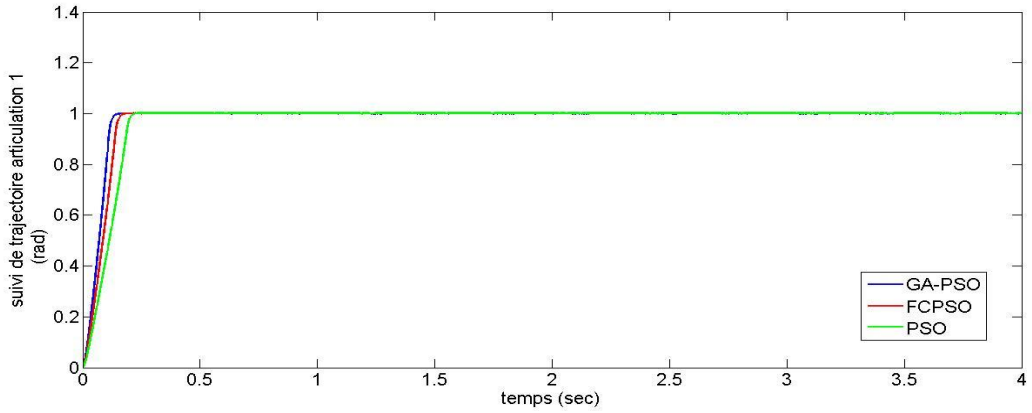


Figure III-4 : Architecture de la commande par mode glissant avec outils d'optimisation

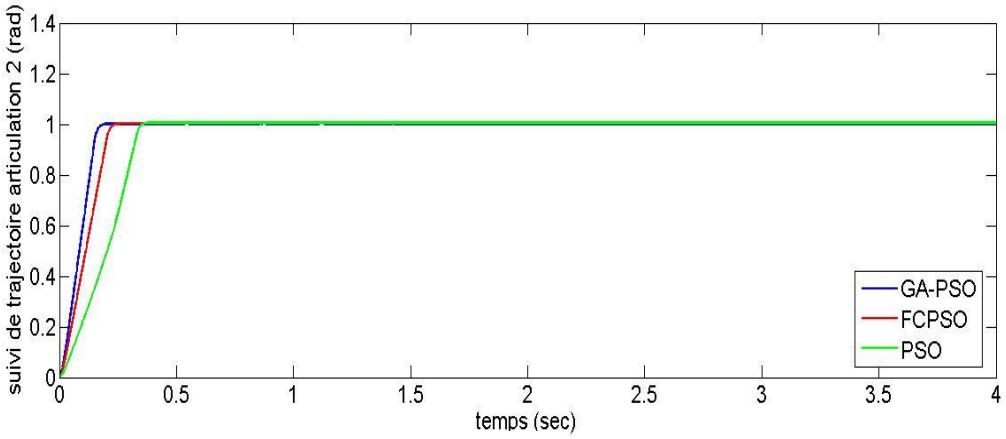
a) Régulation

La trajectoire désirée pour la régulation est définie par

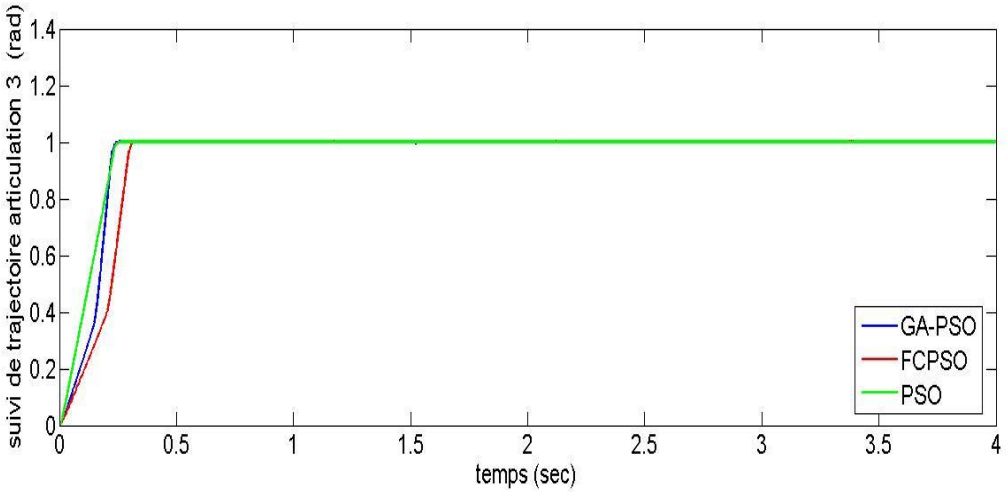
$$\theta_d = u(t) = \begin{cases} 1 \ rad & pour\ t \geq 0 \\ 0 & ailleurs \end{cases}$$



(a)

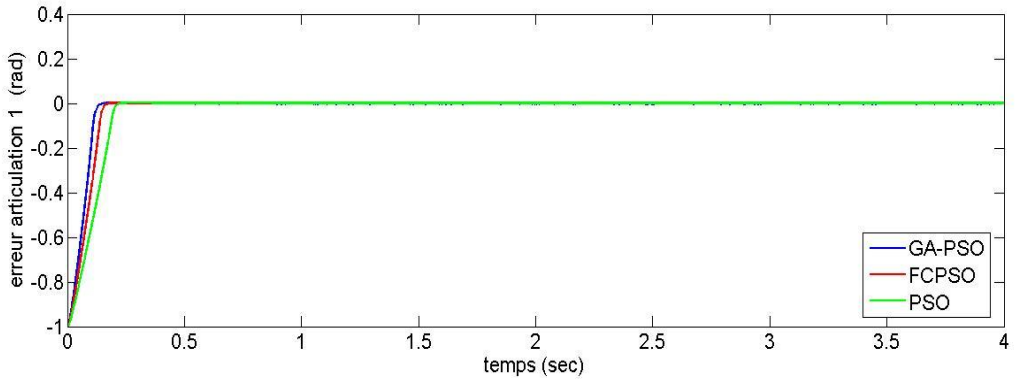


(b)

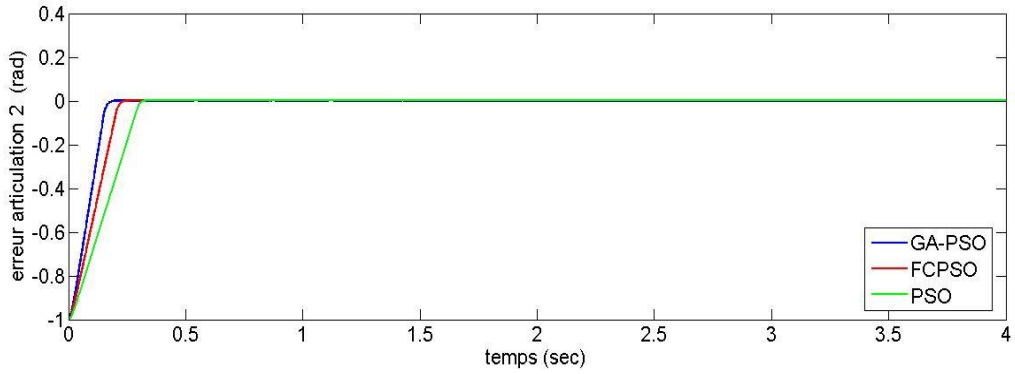


(c)

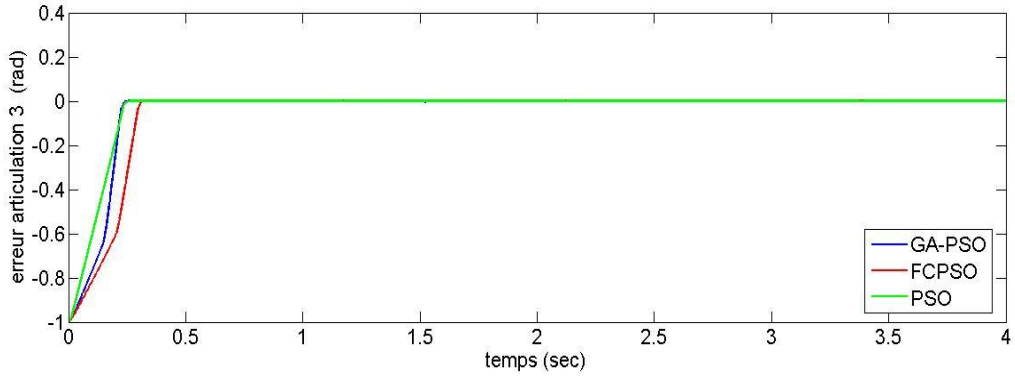
Figure III-5 : (a),(b),(c) : résultats de suivi de trajectoire pour les trois articulations



(a)

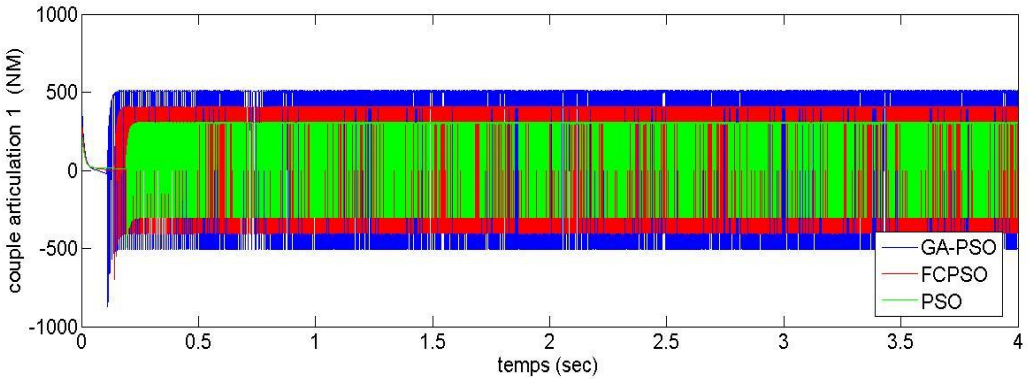


(b)

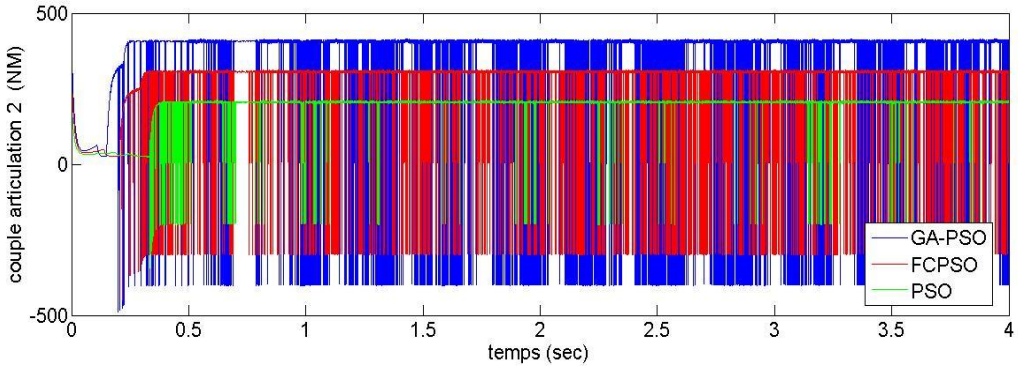


(c)

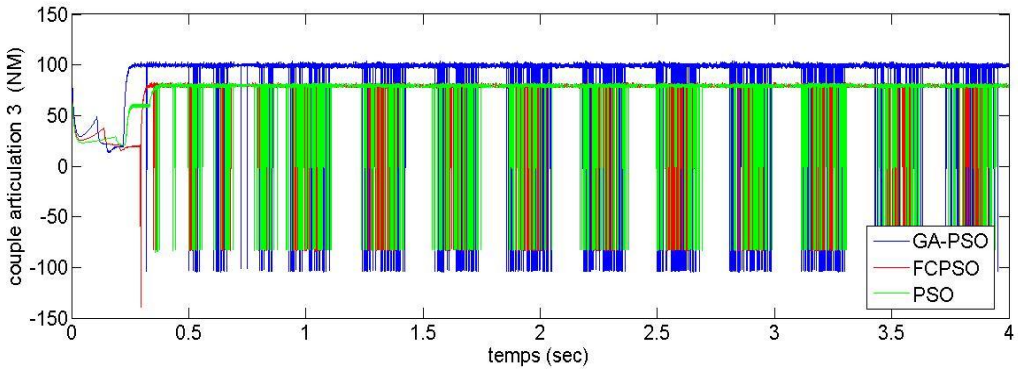
Figure III-6 : (a),(b),(c) : résultats d'erreur de suivi de trajectoire pour les trois articulations



(a)



(b)



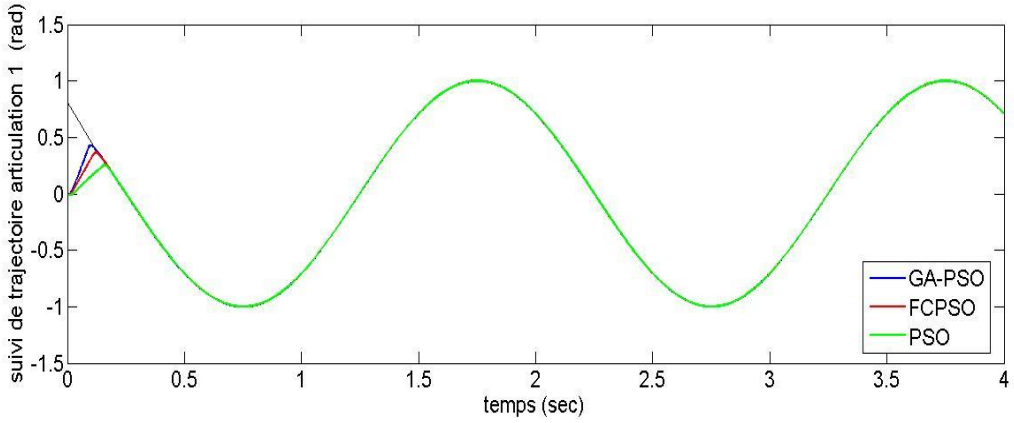
(c)

Figure III-7 (a),(b),(c) : Résultats des couples générer pour le suivi de trajectoire pour les trois articulations

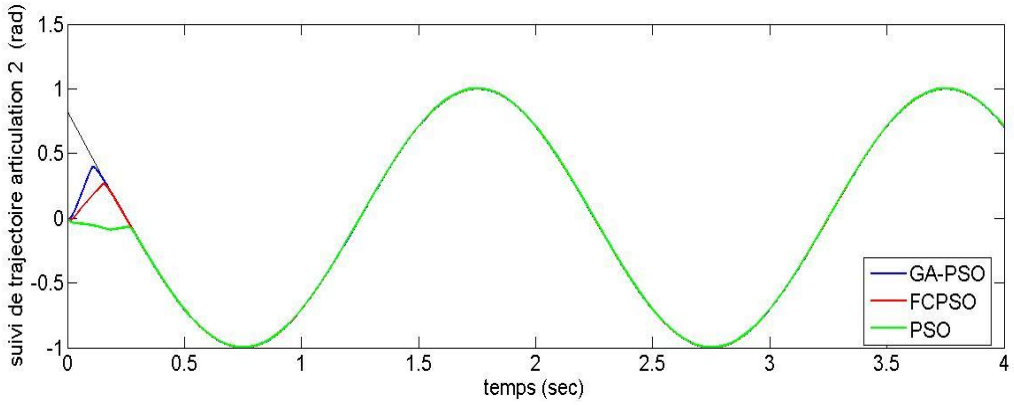
b) Trajectoire sinusoïdale :

La trajectoire désirée est de la forme suivante :

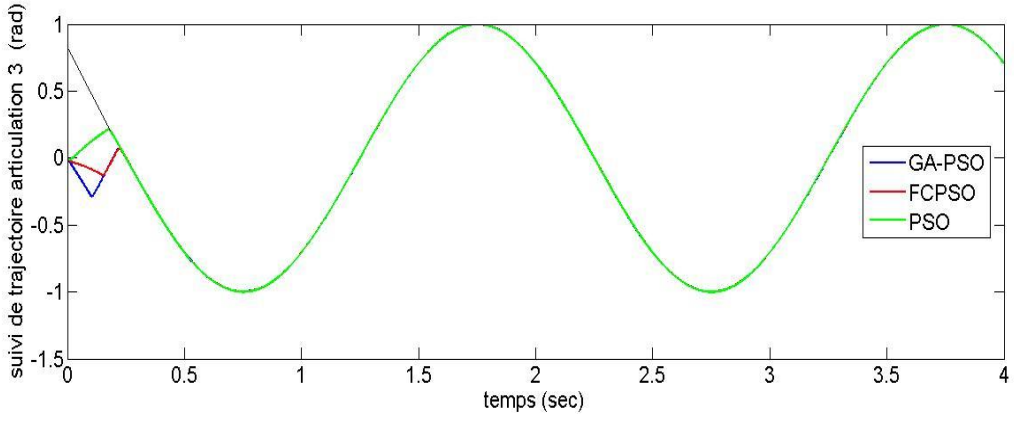
$$\theta_d = u(t) = \begin{cases} \sin\left(\pi t + \frac{3\pi}{4}\right) \text{ rad.} & \text{pour } t \geq 0 \\ 0 & \text{ailleurs} \end{cases}$$



(a)

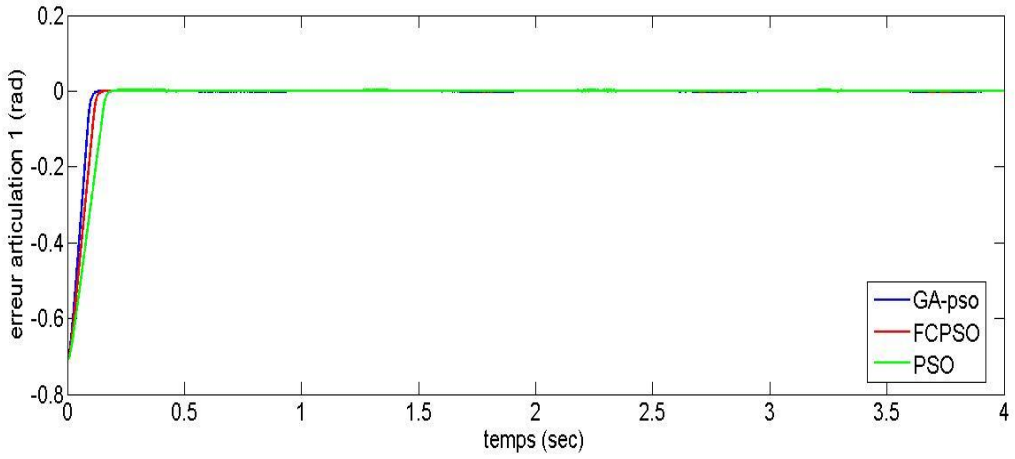


(b)

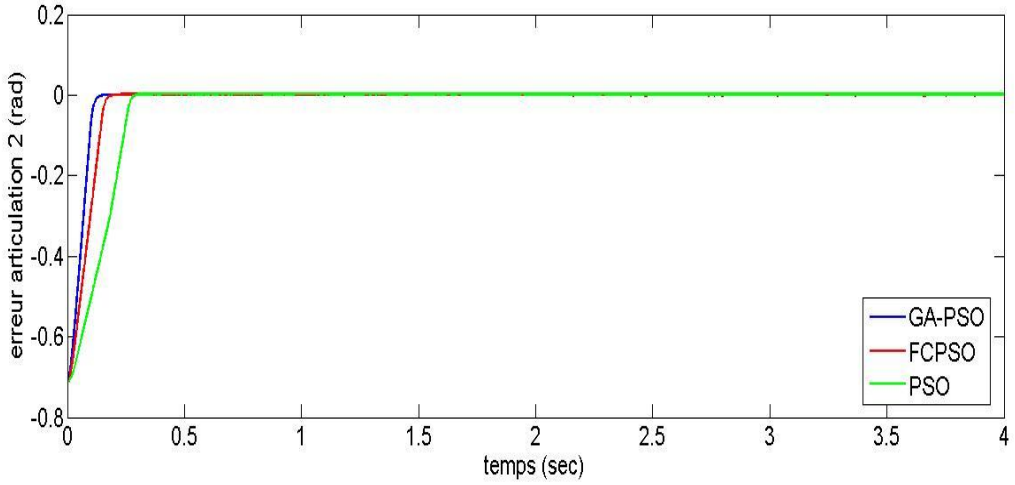


(c)

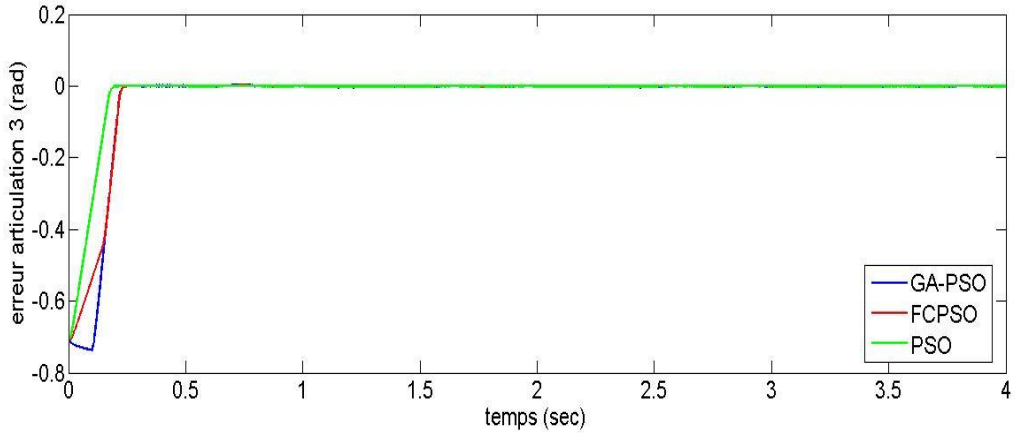
Figure III-8 (a),(b),(c) : résultats suivi de trajectoire sinusoïdale pour les trois articulations



(a)

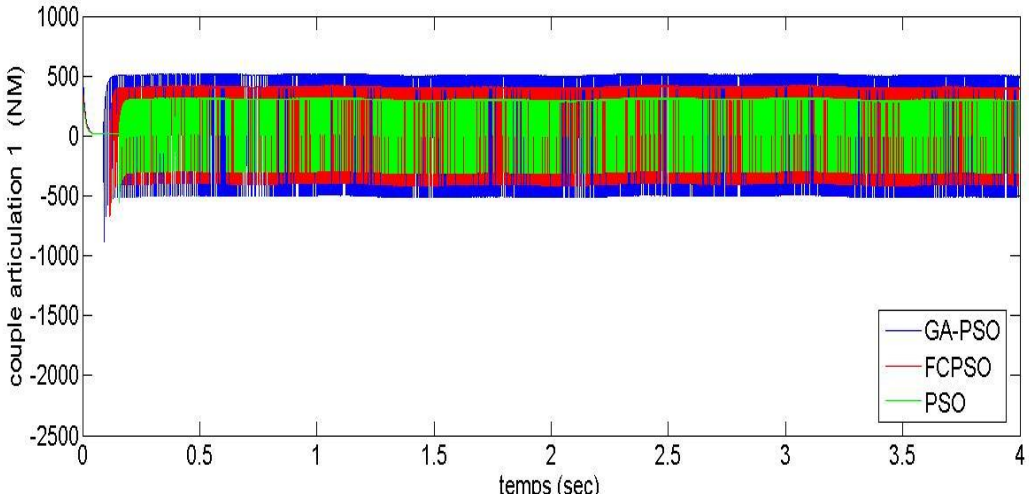


(b)

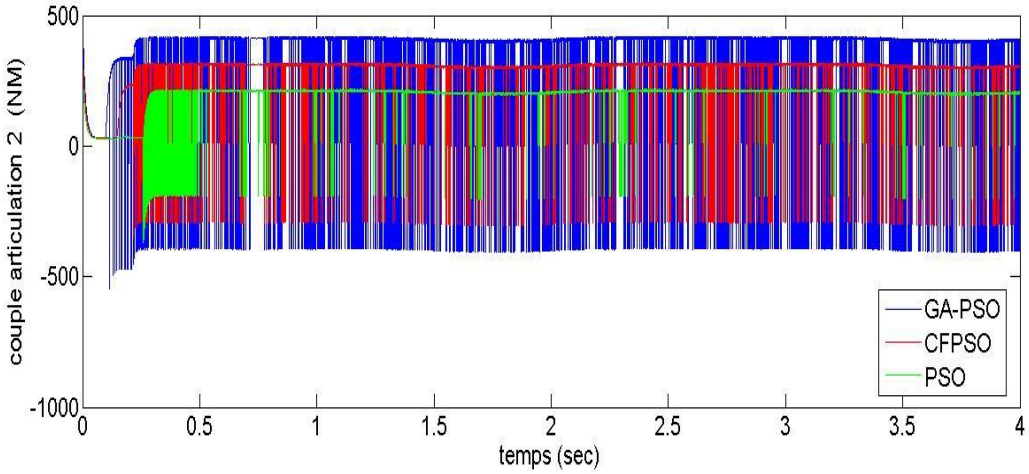


(c)

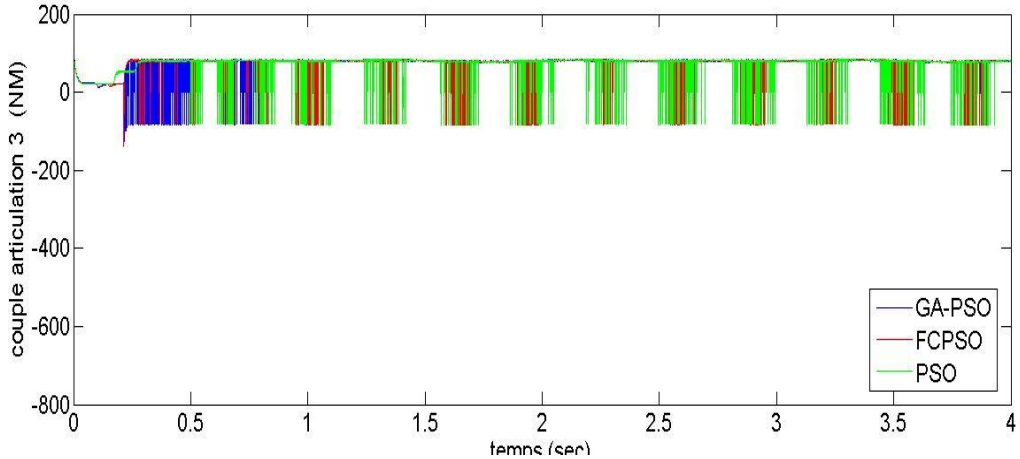
figure III-9 (a),(b),(c) : résultats d'erreur de suivi de trajectoire sinusoïdale pour les trois articulations



(a)



(b)



(c)

Figure III-10 (a),(b),(c) : couple générer pour le suivi de trajectoire sinusoïdale pour les trois articulations

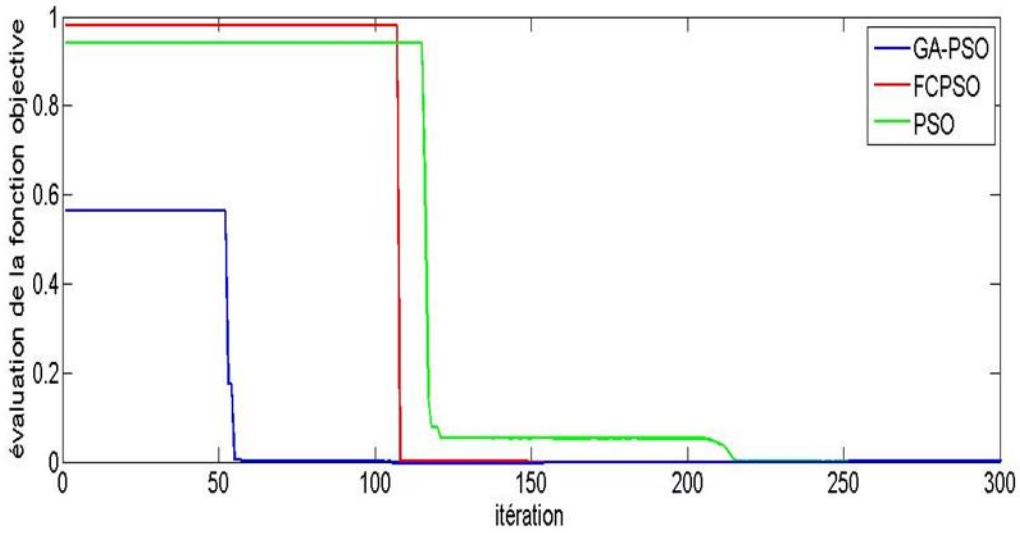


Figure III-11 : Evaluation de la fonction objective

Les résultats obtenus montrent bien l'efficacité de la commande par mode glissant. Il est clair que le signal de commande est un signal alternatif avec une certaine fréquence, ce phénomène est appelé réticence.

Les figures 3-5 à 3-10, montrent bien l'efficacité des algorithmes d'optimisation appliqués, qui donnent de bons résultats.

La figure 3-11 montre que l'algorithme GA_PSO, est meilleur que les deux autres algorithmes, il donne le meilleur résultat avec un minimum de temps de calcul.

III-6 conclusion

Nous avons fait dans ce chapitre la description théorique da la commande par mode glissant puis l'appliquer pour commander un bras manipulateur à trois degrés de liberté. Ensuite on a utilisé trois algorithmes d'optimisation qui sont le PSO, le CFPSO et le GA-PSO pour optimiser ces paramètres dans le but de trouver l'outils d'optimisation le plus performant pour ce type de système complexe.

Les résultats de simulation obtenus montre que l'algorithme GA-PSO donne les meilleurs résultats dans un temps de calcul relativement court par rapport aux autres algorithmes utilisés.

Chapitre IV

La commande floue et floue adaptative par mode glissant

IV.1 Introduction

La commande par mode glissant a montré son efficacité et sa simplicité d'application à travers des études théoriques rapportées. L'avantage que procure une telle commande et qui la rend aussi importante est sa robustesse vis-à-vis des perturbations et des incertitudes du modèle, cependant ces performances sont obtenues au prix de certains inconvénients : un phénomène de réticence qui peut causer l'instabilité et qui a un effet néfaste sur les actionneurs.

Il existe plusieurs approches proposées pour la diminution, voir l'élimination de la réticence. La majorité de ces méthodes se base sur le changement de la nature discontinue qui définit le mode glissant, parmi ces méthodes on trouve la technique floue qui repose sur le remplacement de la fonction discrète de l'algorithme classique par un contrôleur flou qui modélise son comportement. Le gain de sortie de la fonction discrète peut être aussi remplacé par un autre contrôleur flou pour l'ajustement automatiquement de sa valeur, pour obtenir un contrôleur adaptatif.

IV.2 la commande par mode glissant floue

IV.2.1 généralité sur la logique floue type-1

La logique floue est une extension de la logique classique basée sur la théorie des ensembles flous proposée par Lotfi ZEDAH en 1965. L'intérêt de cette extension réside dans sa capacité à traiter l'imprécision et l'incertitude. Cette logique a été introduite dans le but d'approcher le raisonnement humain à l'aide d'une représentation adéquate des connaissances. Ainsi, le succès de la commande floue trouve en grande partie son origine dans sa capacité à traduire une stratégie de contrôle d'un opérateur qualifié en un ensemble de règles linguistiques « si … alors » facilement interprétables.

La structure de base d'un système flou se divise en trois parties principales comme le montre la figure 1.

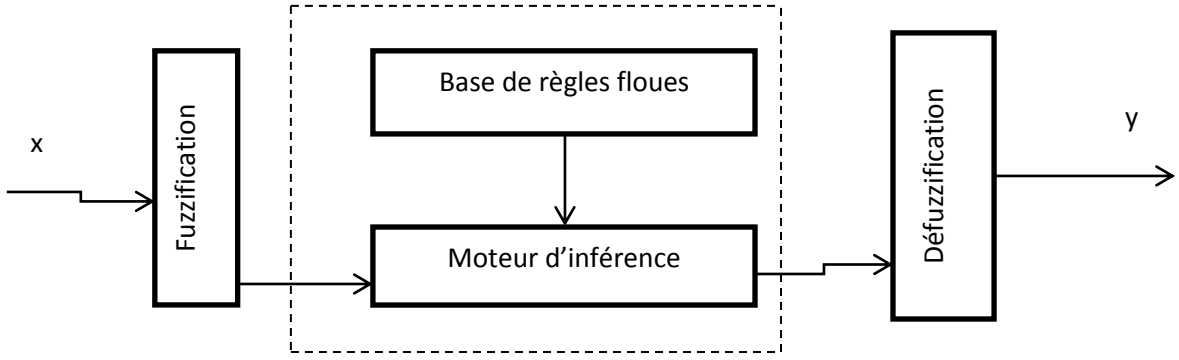


Figure IV-1 : Structure de base d'un contrôleur flou

- **Fuzzification**

La fuzzification consiste à transformer les variables réelles d'entrée et de sortie qui ont une nature numérique en variables linguistiques pour pouvoir effectuer une logique d'inférence. Ainsi on associe, à chacune de ces variables, des ensembles caractérisant les termes linguistiques choisis pour les présenter. Ces termes seront utilisés pour écrire les règles d'inférence.

Les fonctions d'appartenances qui définissent ces variables ne sont pas d'une forme précise car les études comparatives ont montré qu'avec différentes formes, les résultats sont pratiquement similaires en boucle fermée [11]. La forme la plus fréquemment utilisée en commande floue est la forme triangulaire qu'on va utiliser. En général, on introduit pour une variable linguistique un nombre impair d'ensembles flous (trois, cinq ou sept,…..) qui se répartissent autour de zéro et le choix de ce nombre dépend de la précision souhaitée. Les fonctions d'appartenance peuvent être symétriques, non symétriques et équidistantes ou non équidistantes.

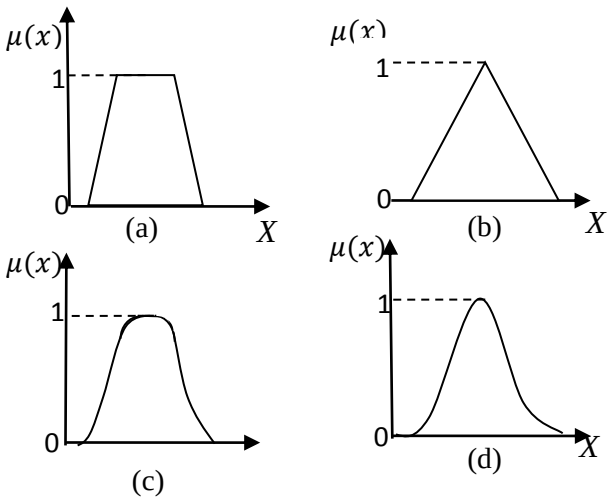


Figure IV-2 : Différentes formes de fonctions d'appartenance

(a) Trapézoïdale, (b) Triangulaire, (c) Gaussienne, (d) Gaussienne généralisée

- **La base de règles**

La base de règles floues est la collection de règles qui permet de définir la liaison entre les variables d'entrée et de sortie. La description de la commande se fait par l'intermédiaire de ces règles qui ont la forme suivante :

$$Si\ x_1\ est\ A_i\ ET\ x_2\ est\ A_2\ \ Alors\ \ y\ est\ B \tag{4.1}$$

Les variables x_1 et x_2 représentent les grandeurs physiques caractéristiques du système. Les variables A_1 et A_2 sont des termes linguistiques. La valeur de B permet d'avoir deux possibilités, si c'est une valeur linguistique, le contrôleur est dit de type Mamdani et si c'est une valeur numérique ou une équation mathématique, alors le contrôleur est dit de type Takagi-Sugeno [12]. Le ET de conjonction est réalisé en effectuant le minimum entre les degrés de vérité des propositions floues.

- **Méthode d'inférence floue**

C'est l'opération qui permet de calculer l'ensemble flou associé à la sortie qui se fait par les opérations d'inférence floue et l'agrégation des règles.

L'inférence repose sur l'utilisation d'un opérateur d'implication floue pour chaque règle à analyser. Cet opérateur quantifie la force de liaison entre la prémisse et la conclusion de la règle. Soit la règle suivante : Si *x* est *A* alors *y* est *B*, l'inférence peut être exprimée mathématiquement par l'expression suivante :

$$\mu'_B(y) = I\big(\mu_A(x_0), \mu_B(y)\big) \tag{4.2}$$

où I désigne l'opérateur d'inférence, comme elle peut être exprimée par une description linguistique, par matrice d'inférence ou par tableau d'inférence. Deux approches d'inférence sont couramment utilisées :

- Implication de Mamdani :$\mu'_B(y) = min\big(\mu_A(x_0), \mu_B(y)\big)$
- Implication de Larsen : $\mu'_B(y) = \mu_A(x_0)\mu_B(y)$

Pour extraire une conclusion à partir de l'ensemble des règles établies, il faut procéder à une agrégation de ces règles par un opérateur disjonctif, ce qui revient à lier les règles par un opérateur OU. Généralement l'opérateur est utilisé pour agréger un ensemble de *n* règles :

$$\mu_B(y) = \underset{i=1\ldots n}{max}\mu_{B_i}(y) \tag{4.3}$$

- **Défuzzification**

Pour avoir une sortie convenable à l'utilisation, issue du traitement des règles d'inférence qui fournit une valeur floue, l'étape de défuzzification est une démarche nécessaire pour transformer l'ensemble flou, résultant de l'agrégation des règles, en une grandeur précise à appliquer au processus. Cette opération peut être réalisée par différentes stratégies qu'on trouve dans la littérature, telle que la moyenne des maxima, le centre des aires et le centre des maxima. On utilise souvent la méthode de défuzzification par le centre de gravité en commande floue du fait qu'elle fournit intuitivement la valeur la plus représentative de l'ensemble flou issu de l'agrégation des règles. Cette approche consiste à calculer le centre de gravité de la surface formée par la fonction d'appartenance résultante.

La commande *u* est obtenue par une simple moyenne pondérée selon les niveaux d'activation w^lde chacune des règles.

$$u = \frac{\sum_{l=1}^{K} w^l f^l(x_1, x_2, .., x_n)}{\sum_{l=1}^{K} w^l} \tag{4.4}$$

Avec $w^l = T\left(\mu_{A_1^l}(x_1), \mu_{A_2^l}(x_2), \dots, \mu_{A_n^l}(x_n)\right)$ où T est une t-norme choisie très souvent égale à l'opérateur produit.

IV.3 Conception du contrôleur (FSMC)

En partant de la particularité de modélisation des notions vagues et incertaines du langage naturel de la logique floue, on va projeter la stratégie de la commande par mode glissant, en extrayant les relations qui lient les ensembles flous, pour obtenir une approximation du comportement de la partie discrète « la fonction signe » de l'algorithme du mode glissant classique. Ainsi un mécanisme d'inférence floue est utilisé et la méthodologie du contrôleur par mode glissant flou (Fuzzy Sliding Mode Control, FSMC) [13] est définie.

L'ensemble (manipulateur et contrôleur) du système complet est représenté su la figure 4. La commande se divise en deux parties, une commande équivalente identique à celle du mode glissant classique et une autre partie (4.5) représentant la loi de commande qui assure le mode de convergence.

$$u_c = k_f u_f \tag{4.5}$$

Où K_f est le facteur de normalisation de la variable de sortie de dimension 3×3 et u_f est le vecteur de sortie du FSMC de dimension 3×1.

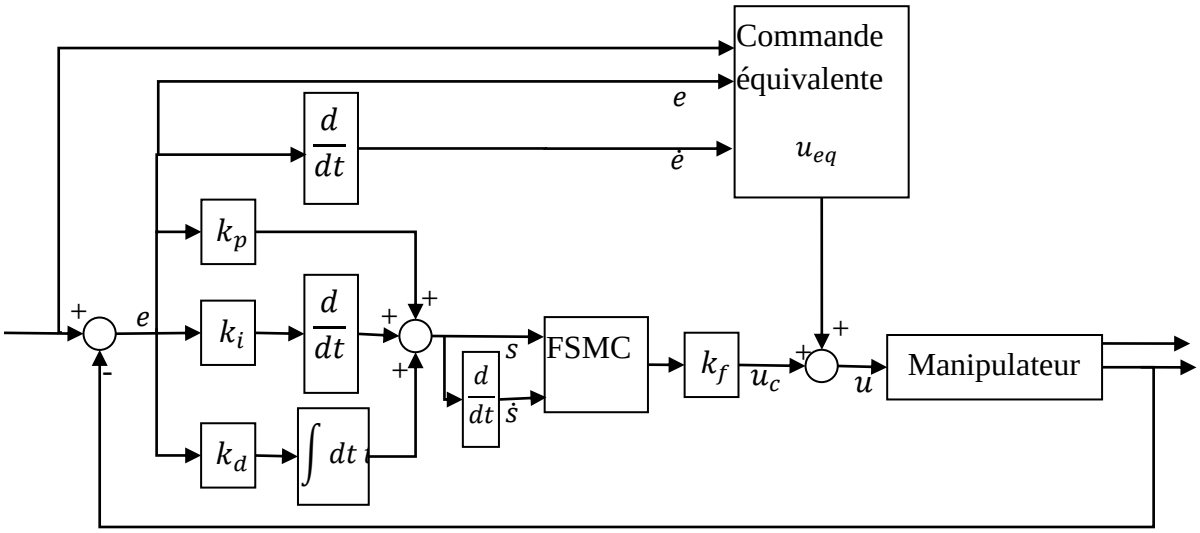


Figure IV-3 : Schéma bloc de la commande par mode glissant flou (FSMC) avec une surface PID.

Notre contrôleur utilise la même surface que celle du chapitre précédent, alors l'expression globale de notre commande devient :

$$u = u_{eq} + u_c = u_{eq} + k_f u_f \tag{4.6}$$

où la partie floue a la forme :

$$u_f = FSMC(s, \dot{s}) = [u_{f1}, u_{f2}, \dots\dots, u_{fn}] \quad (4.7)$$

$$u_f = [FSMC(s_1, \dot{s}_1), FSMC(s_2, \dot{s}_2), \dots\dots\dots, FSMC(s_n, \dot{s}_n)] \quad (4.8)$$

La variable S_i représente la surface correspondante à une variable généralisée q_i, ce qui veut dire un nombre n de surfaces pour n variables généralisées, contrôlées par n contrôleurs flous. Pour un bras manipulateur à 3 ddl, on aura :

$$u_f = [FSMC(s_1, \dot{s}_1), FSMC(s_2, \dot{s}_2), FSMC(s_3, \dot{s}_3)] \quad (4.9)$$

La conception de la partie floue du contrôleur suit les étapes suivantes :

IV.3.1Fuzzification

La conception du contrôleur proposé nécessite deux entrées et une sortie qui sont S, $\dot{S}$ et le signal de commande u_f avec le choix d'utiliser les mêmes variables linguistiques pour les trois ensembles flous $\{(s, \mu(s)), (\dot{s}, \mu(\dot{s})), (u_f, \mu(u_f))\}$.

Les variables linguistiques utilisées sont :

NG: négatif grand, NM : négatif moyen, NP: négatif petit, Z : zéros, PP : positif petit

PM : positif moyen, PG : positif grand.

La plage d'entrée et de sortie du contrôleur a été normalisée S, $\dot{S}$ et u_f respectivement, comme présenté sur les figures 4 et 5.

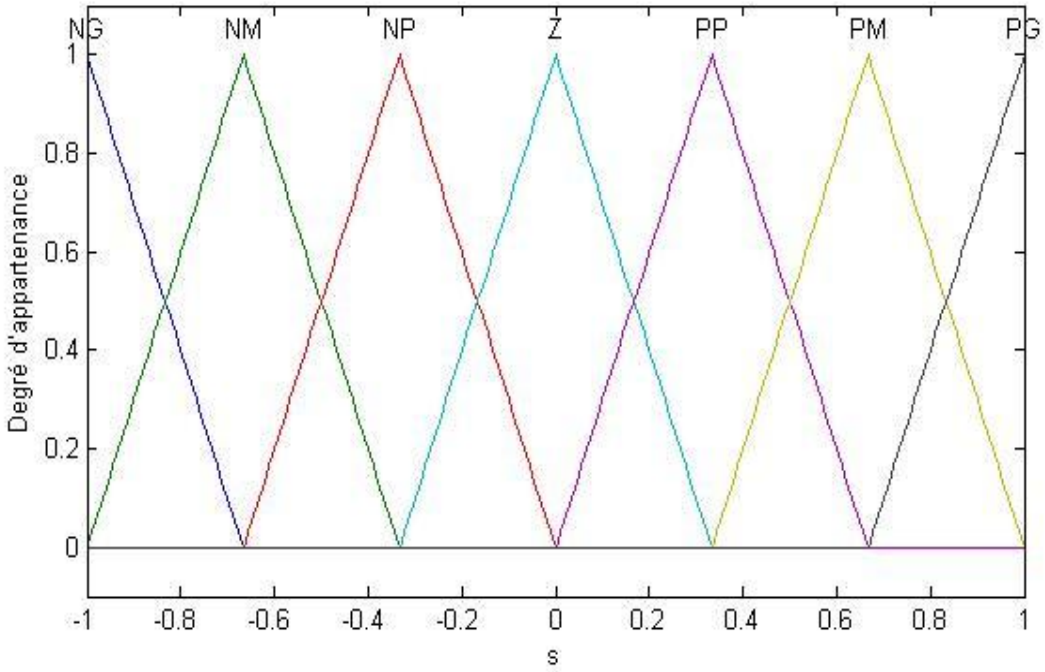


Figure IV- 4 : Fonctions d'appartenances des variables linguistiques de S et $\dot{S}$.

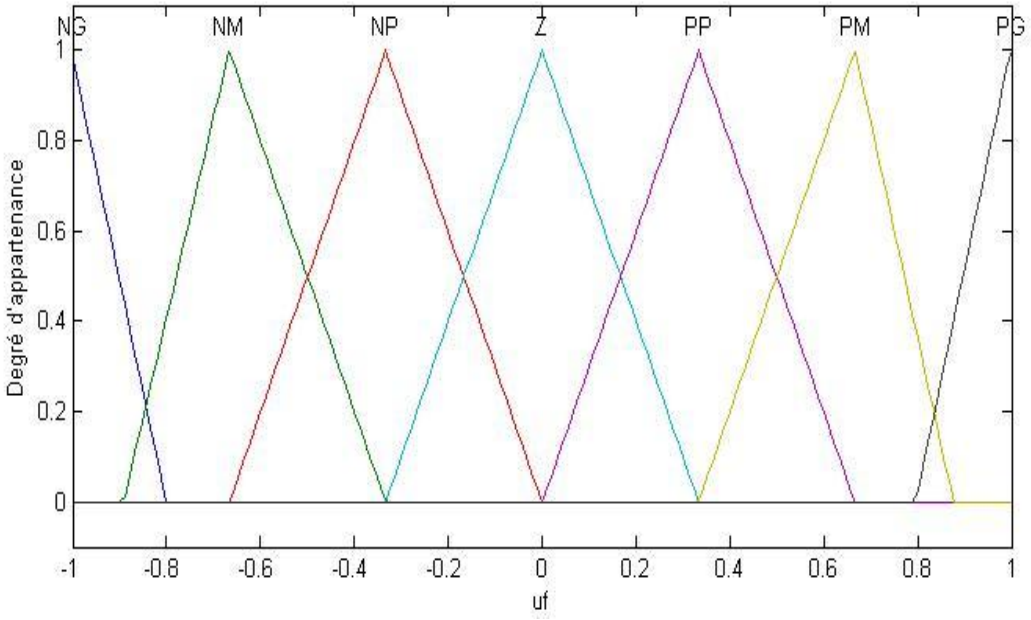


Figure IV-5: Fonctions d'appartenances des variables linguistiques de u_f.

IV.3.2 La base de règles

L'ensemble des relations qui lient les variables linguistiques d'entrée *S et* $\dot{S}$ aux variables linguistiques de sortie u_f présentent les relations qui définissent le contrôleur flou, elles sont de la forme:

$$R^{(l)} :: si\ S(t)\ est\ A_1^l\ ET\ \dot{S}(t)\ est\ A_2^l\ alors\ u_f\ est\ B^l \qquad (4.10)$$

Tel que A_1^l, A_2^l et B^l représentent les fonctions d'appartenances des ensembles flous.

Le tableau I est conçu pour définir la stratégie de commande, les bases des règles sont extraites de telle manière que la stabilité du système soit satisfaite [13, 14], en respectant la condition clé suivante :

On s'assure toujours que le produit $S\dot{S}$ Soit negatif. Si la multiplication de *S* et $\dot{S}$ est positive, la sortie u_f va être réglée de telle manière que son signe soit l'opposé de celui de S, ce qui donne : $su_f \leq -|s|$.

Les règles sont rangées dans le tableau suivant, chaque entrée du contrôleur à sept ensembles flous de sortie, ce qui nous donne 49 règles floues.

u_f		s_i						
		PG	PM	PP	Z	NP	NM	NG
$\dot{s}_i$	PG	NG	NG	NG	Z	Z	Z	Z
	PM	NG	NG	NG	Z	Z	Z	PP
	PP	NG	NG	NM	Z	Z	PP	PM
	Z	NG	NM	NP	Z	PP	PM	PG
	NP	NM	NP	Z	Z	PM	PG	PG
	NM	NP	Z	Z	Z	PG	PG	PG
	NG	Z	Z	Z	Z	PG	PG	PG

Tableau IV-I : Ensemble de règles floues

IV.3.3 Inférence et défuzzification

Le mécanisme d'inférence utilisé est de Mamdani et la fonction minimum d'intersection a été employée pour l'implication floue. Pour la défuzzification, on prend la méthode du centre de gravité pour calculer les valeurs du vecteur de sortie u_f.

On obtient la surface de commande suivante :

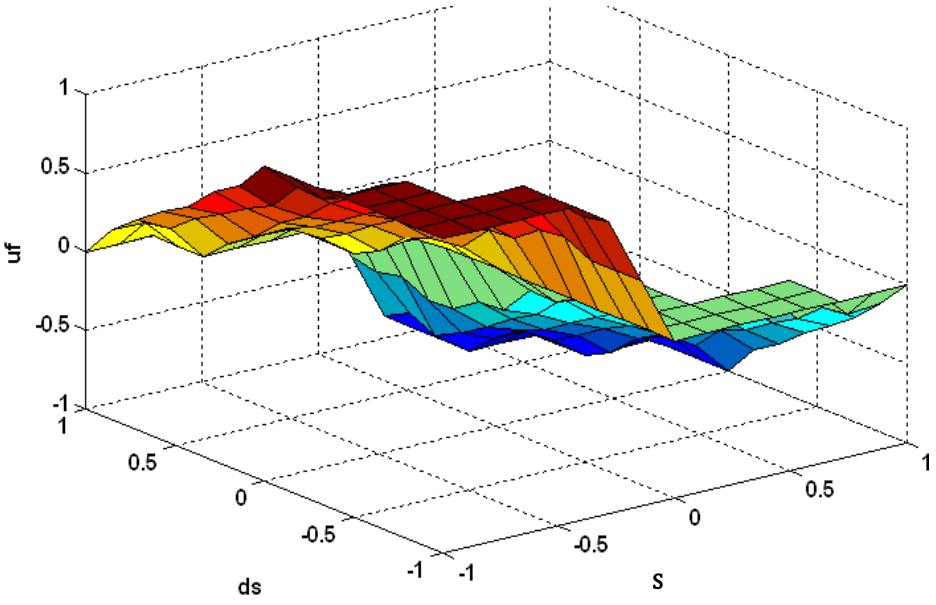


Figure IV-6 : Surface de commande de u_f

IV.3.4 Analyse de stabilité

On considère le modèle présenté par l'équation (3.15) et on choisit la surface de glissement *s* tel que :

$$s(t) = k_p e(t) + k_i \int e(t)dt + k_d \frac{d}{dt} e(t) \tag{4.11}$$

$$\dot{s}(t) = k_p \dot{e} + k_i e + k_d \dot{e} \tag{4.12}$$

$$\dot{s}(t) = k_p \dot{e} + k_i e + k_d \left[\dot{\theta}_d - \dot{\theta}\right] \tag{4.13}$$

$$k_p \dot{e} + k_i e + k_d \left[-\ddot{\theta}_d + g_a u + f_a + b\right] \tag{4.14}$$

On choisit comme fonction de Lyapunov, la fonction $V = \frac{1}{2}s^2$, d'où sa dérivée :

$$\dot{V} = s\dot{s} = s\left[k_p \dot{e} + k_i e + k_d \dot{e}\right] \tag{4.15}$$

$$\dot{V} = s[k_p \dot{e} + k_i e + k_d[\ddot{q}_d - g_a u - f_a - b(t)] \quad / u = u_{eq} + u_c \tag{4.16}$$

$$\dot{V} = s\left[-k_d g_a u_f - k_d b\right] / u_{eq} = (k_d g_a)^{-1}\left[k_p \dot{e} + k_i e - k_d f_a + k_d \dot{\theta}_d\right] \tag{4.17}$$

$$\dot{V} = s\left[-k_d g_a k_f FSMC(s, \dot{s}) - k_d b(t)\right] \tag{4.18}$$

$$\dot{V} \leq -k_d g_a k_f s FSMC(s, \dot{s}) + k_d |b(t)||s| \leq -k_d g_a k_f |s| + k_d \alpha |s| \tag{4.19}$$

Avec α représentant l'extrémité des bornes d'incertitudes et de perturbations :$|b(t)| < \alpha$. Si on choisit $K_f < g_\alpha^{-1}\ \alpha$, alors la commande va assurer la convergence de l'état vers la surface de glissement, ainsi le système bouclé est asymptotiquement stable et l'erreur va converger vers zéro.

La commande du système est

$$u = u_{eq} + u_c = (k_d g_a)^{-1}\left[k_p \dot{e} + k_i e - k_d f_a + k_d \ddot{\theta}_d\right] + k_f FSMC(s, \dot{s}) \tag{4.20}$$

IV.4 Commande par mode glissant flou adaptatif

La commande par mode glissant flou (FSMC) suppose d'avoir une certaine connaissance sur les bornes des incertitudes et des perturbations qui peuvent survenir, d'où le choix d'une valeur fixe du gain K_f, ce qui est difficile à entreprendre. Pour résoudre ce problème et

minimiser les contraintes sur la connaissance des bornes, un superviseur flou [13] a été ajouté au contrôleur précédent pour ajuster le gain K_f et on obtient l'architecture suivante :

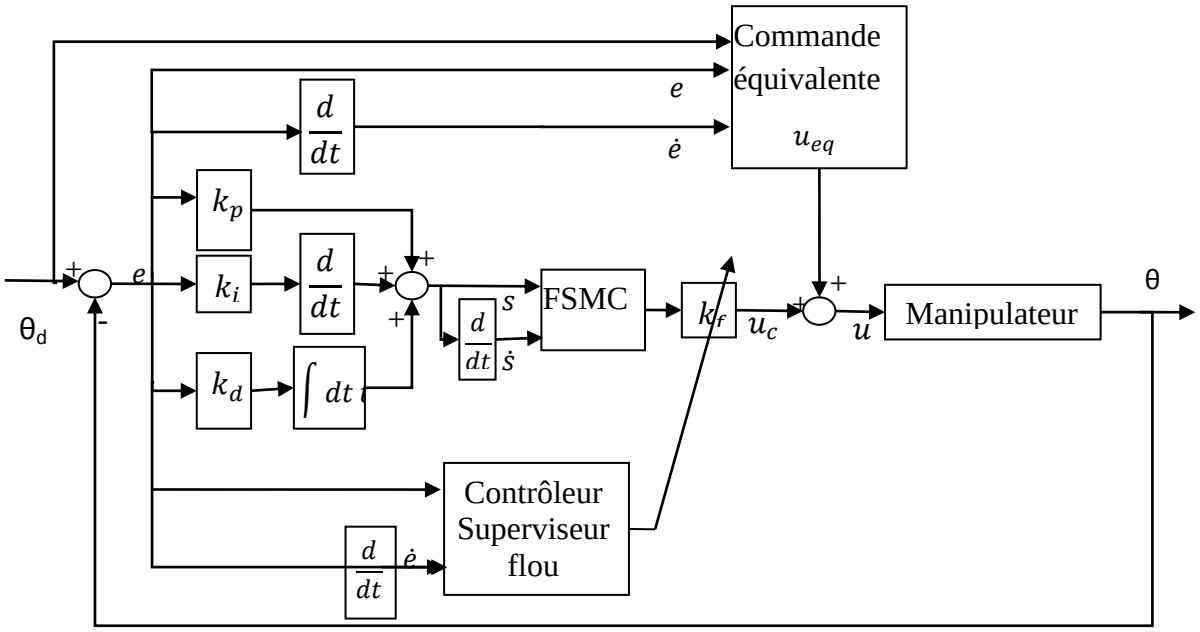


Figure IV-7 : Schéma bloc de la commande par mode glissant flou adaptatif (AFSMC) avec une surface PID.

IV.4.1 Conception du contrôleur (AFSMC)

Le contrôleur par mode glissant flou adaptatif (Adaptive Fuzzy Sliding Mode Control) conserve le même contrôleur flou avec le remplacement du gain fixe par un système de surveillance floue [13], afin d'ajuster de manière adaptative le gain de la commande de convergence K_f dans le but d'améliorer le rendement du contrôleur. La conception du superviseur se déroule comme ci-dessous:

IV.4.1.1 Fuzzification

On fuzzifie les variables d'entrée et de sortie du contrôleur e et $\dot{e}$ et le signal de commande K_f. Les variables linguistiques proposées sont exprimées comme suit:

1) - pour les entrées :

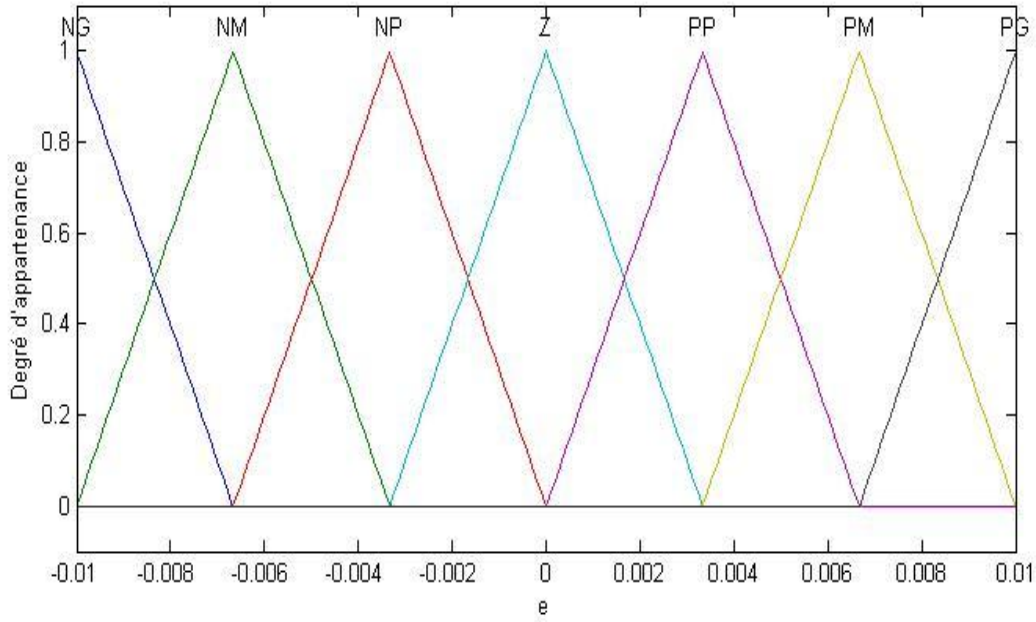
NG: négatif grand, NM : négatif moyen, NP: négatif petit, Z : zéros, PP : positif petit

PM : positif moyen, PG : positif grand.

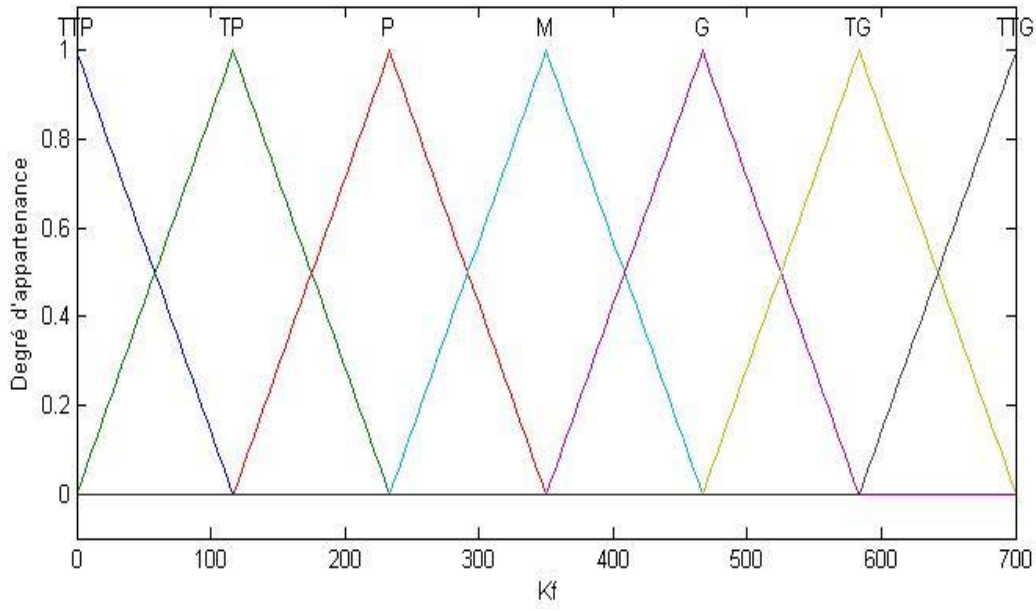
2) - pour la sortie :

TTP : très très petit, TP : très petit, P : petit, M : moyen, G: grand, TG : très grand

TTG : très très grand.



(a)



(b)

Figure IV-8 : Fonctions d'appartenances des variables linguistiques *e*, *ė* et u_f
(a) : Fonctions d'appartenances de *e et ė*, (b) : Fonctions d'appartenances de u_f

IV.4.1.2 La base de règles

Les règles de contrôle du système superviseur flou sont conçues à base du vecteur de l'erreur *e* et de sa dérivée *ė*, *et* $K_f = [k_{f1}\, k_{f2}\, k_{f3}]^T$ avec l'expression des règles comme suite :

$$R^{(i)} :: SI\ e(t)\ est\ E_1^i\ ET\ \dot{e}(t) est\ E_2^i\ alors\ k_f\ est\ G^i \qquad (4.21)$$

Tel que E_1^i, E_2^i et G^i représentent les ensembles d'appartenance de *e(t), ė(t) et* K_f respectivement. Les règles linguistiques sont présentées dans le tableau II

k_f		e						
		NG	NM	NP	Z	PP	PM	PG
$\dot{e}$	NG	M	P	TP	TTP	TP	P	M
	NM	G	M	P	VP	P	M	G
	NP	TG	G	M	P	M	G	TG
	Z	TTG	VG	G	M	G	TG	TTG
	PP	TG	G	M	P	M	G	TG
	PM	G	M	P	VP	P	M	G
	PG	M	P	VP	TTP	TP	P	M

Tableau IV-II : Ensemble des règles floues

IV.4.1.3 Inférence et défuzzification

Le mécanisme d'inférence de Mamdani est utilisé avec l'emploie de la fonction minimum pour l'intersection et la méthode du centre de gravité pour la défuzzification de la sortie.

Le signal de sortie global du contrôleur est comme suit :

$$u = u_{eq} + u_c = u_{eq} + k_f u_f = (k_d g_a)^{-1}\left[k_p \dot{e} + k_i e - k_d f_a + k_d \dot{\theta}_d\right] + k_f u_f \quad (4.21)$$

Tel que :

$$u_f = FSMC(s, \dot{s}) \; et \; k_f = f(e, \dot{e}) \quad (4.22)$$

Remarque :

En ce qui concerne la stabilité du système, elle est assurée, comme il a été prouvé dans le paragraphe (IV-3-4), tant que la condition $k_f > g_a^{-1}\alpha$ est vérifiée, ce qui nous donne des valeurs de bornes d'incertitude et de perturbation plus flexibles.

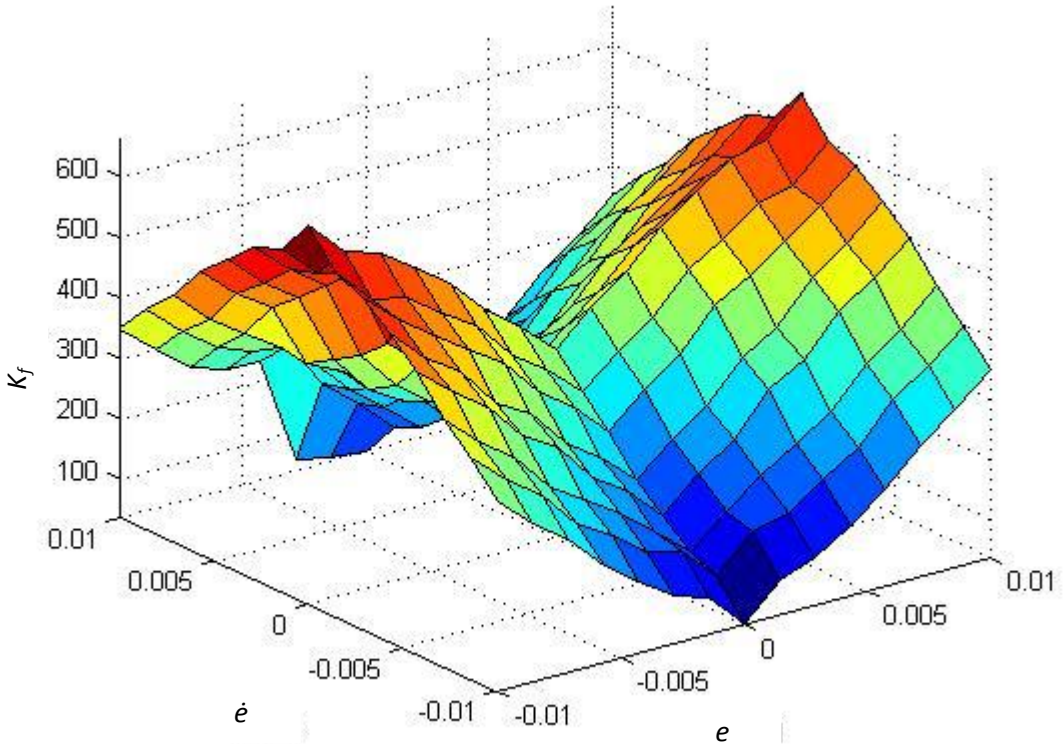


Figure IV-9 : Surface de la commande K_f

IV.5 Simulation et résultats

IV.5.1 Commande par mode glissant flou

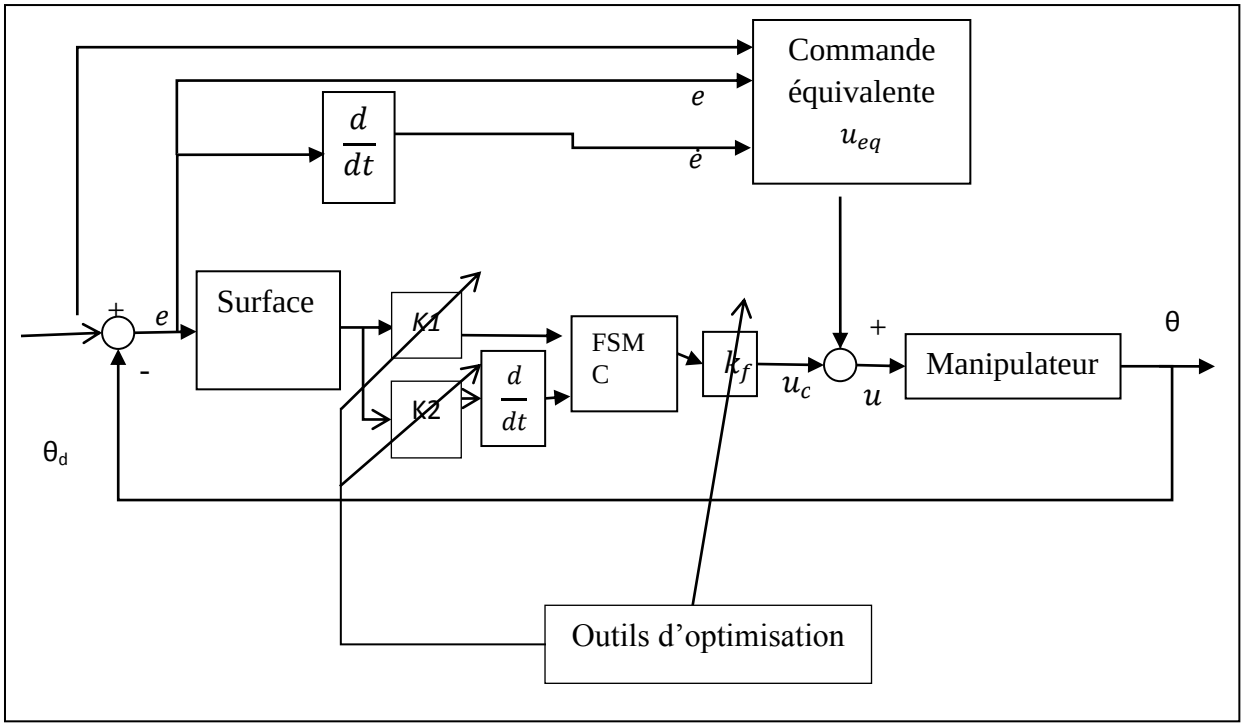


Figure IV-10 : Architecture de la commande par mode glissant flou avec outils d'optimisation

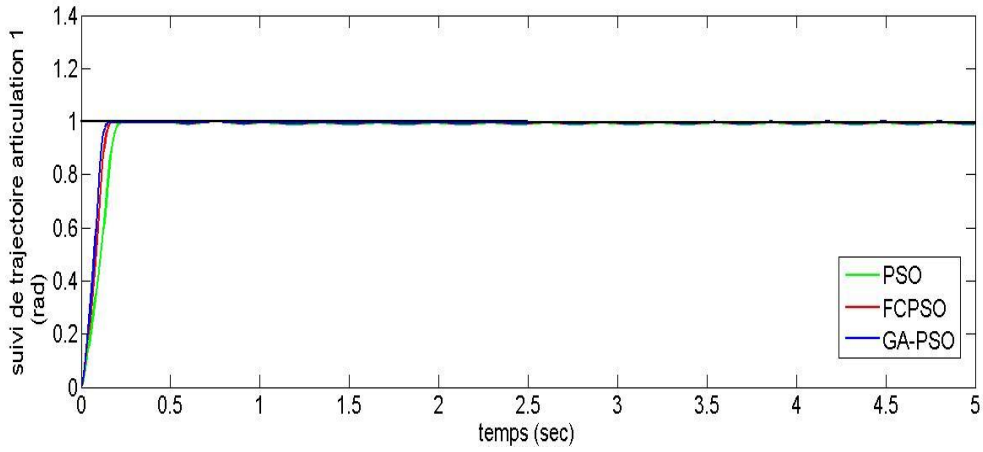
Durant les travaux de simulation deux types de trajectoires ont été pris en considération, une trajectoire en forme d'échelon et une seconde en forme de sinusoïde. On s'est intéressé au suivi de trajectoires, aux erreurs de suivi et aux couples générés par les actionnaires des trois articulations en utilisant les trois algorithmes d'optimisation qui sont le PSO, le CFPSO et l'algorithme hybride le GA-PSO. L'évaluation de la fonction objective est aussi illustrée.

1) Régulation

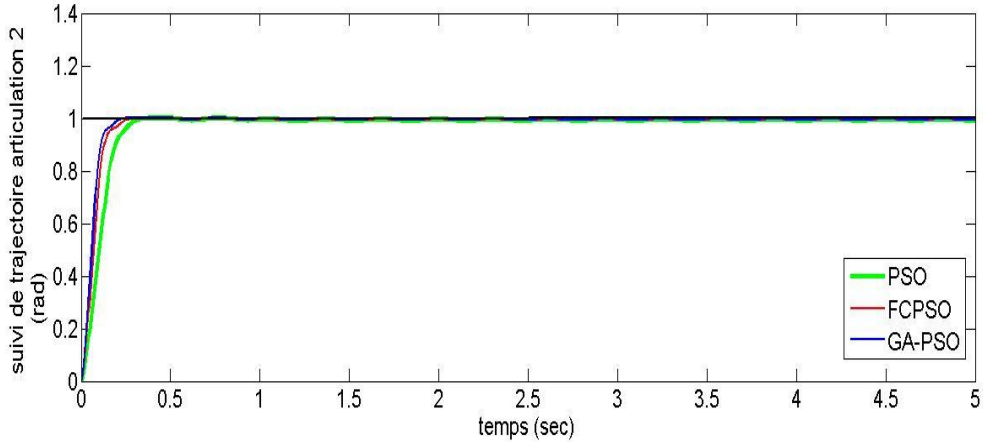
La trajectoire désirée pour la régulation des trois articulations est:

$$\theta = u(t) = \begin{cases} 1 \text{ rad pour } t \geq 0 \\ 0 \quad \text{ailleurs} \end{cases}.$$

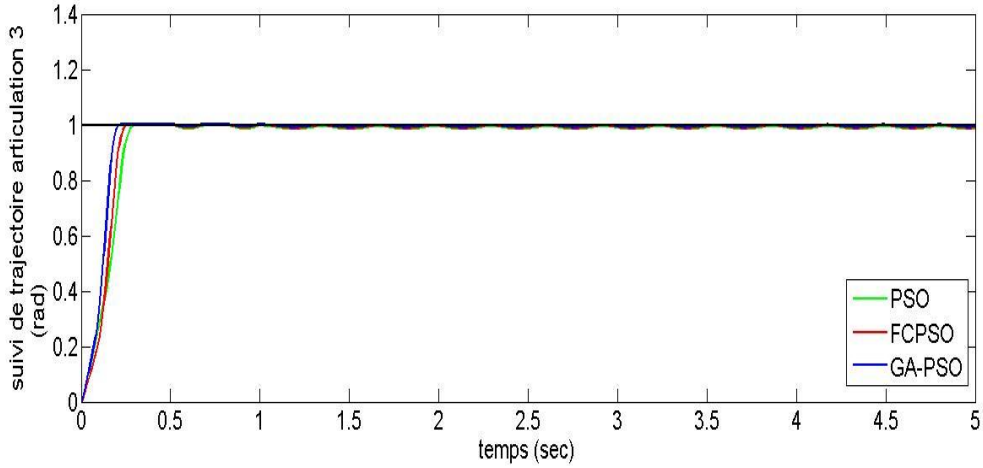
Les matrices K_1, K_2 et K_f sont choisies de façon automatique à l'aide des algorithmes d'optimisation utilisés qui sont le PSO, le CFPSO et le GA-PSO.



(a)

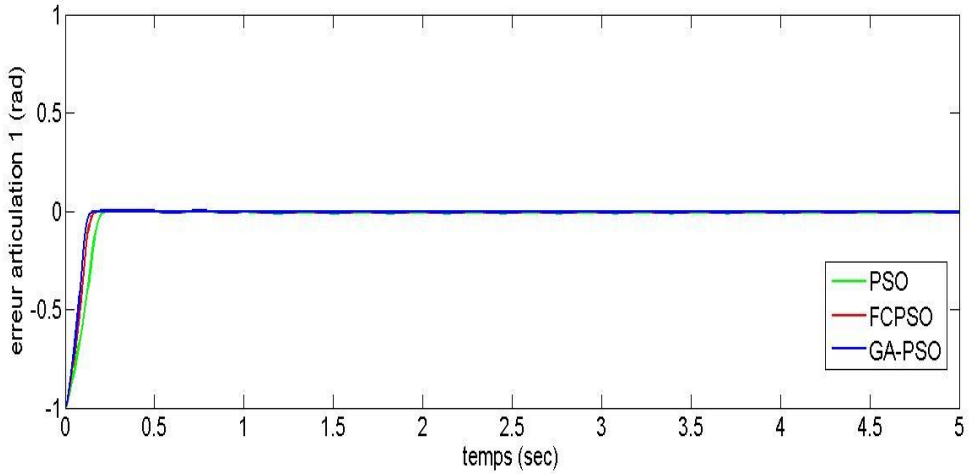


(b)

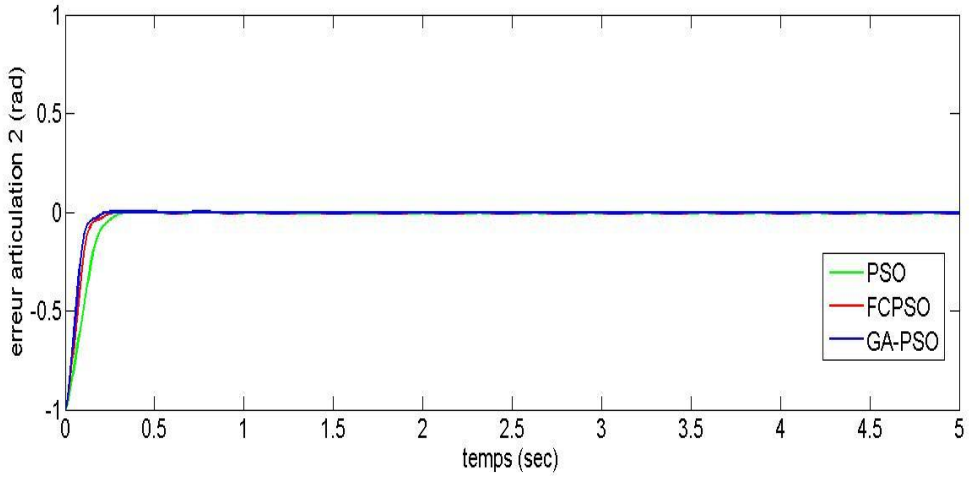


(c)

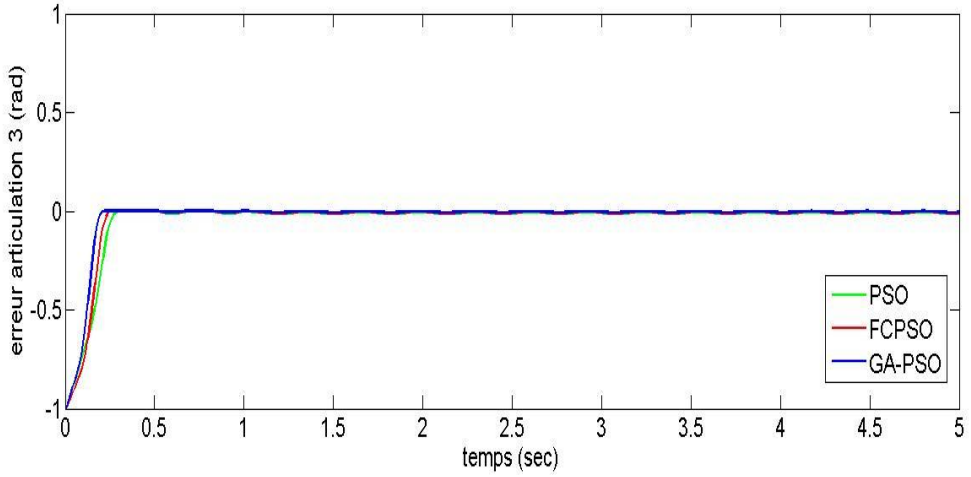
Figure IV-11 : Suivi de trajectoires pour les trois articulations
(a) : Articulation 1, (b) : Articulation 2 et (c) : Articulation 3.



(a)

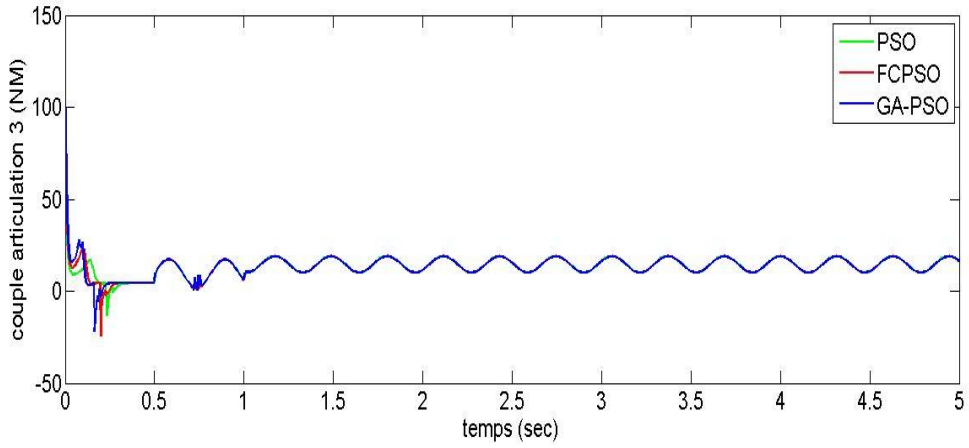


(b)

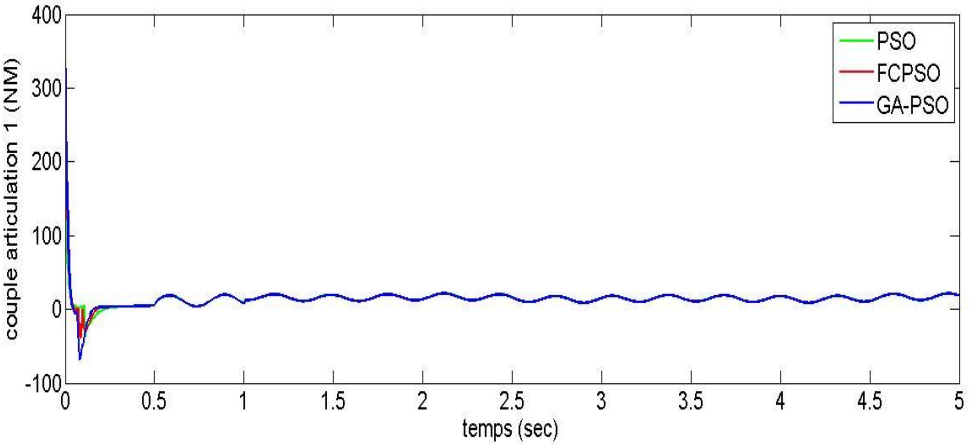


(c)

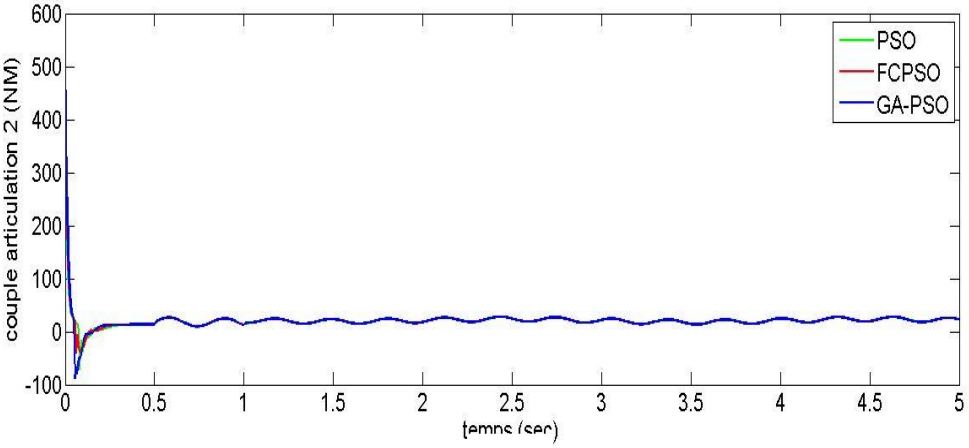
Figure IV-12 : Erreurs de suivi pour les trois articulations
(a) : Articulation 1, (b) : Articulation 2 et (c) : Articulation 3.



(a)



(b)



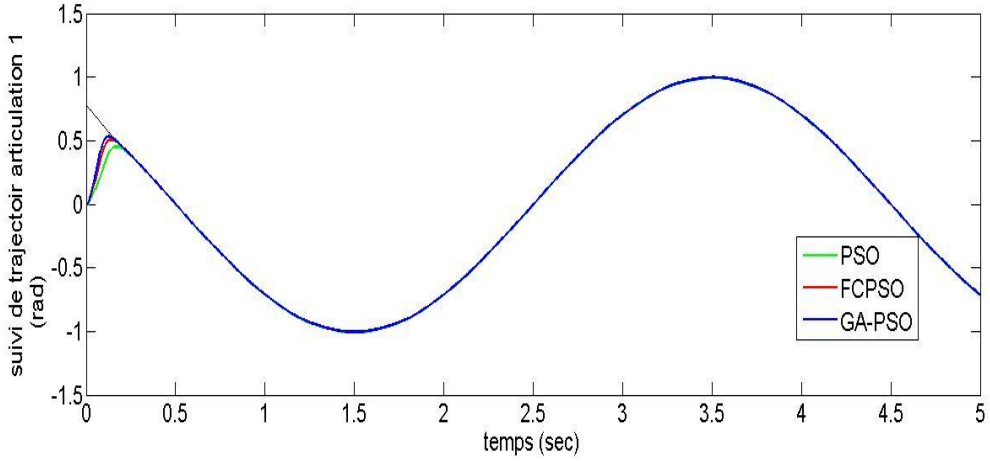
(c)

Figure IV-13 : Couples générés pour le suivi de trajectoires.
(a) : Articulation 1, (b) : Articulation 2 et (c) : Articulation 3.

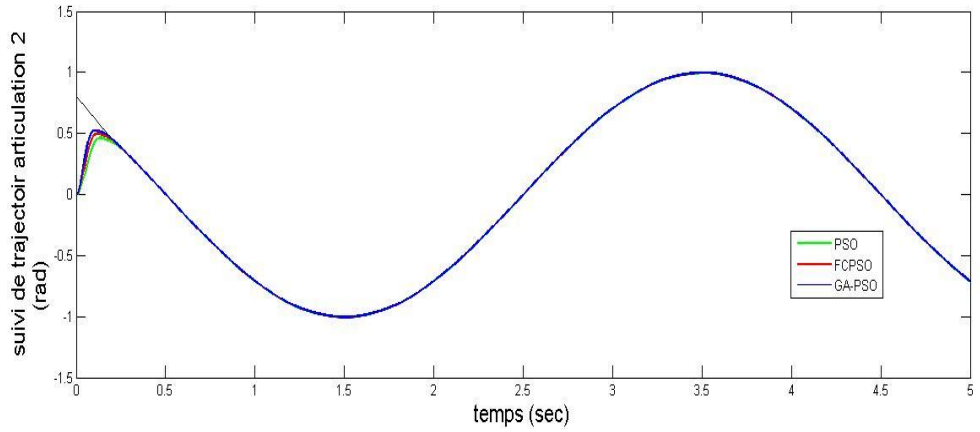
2) Trajectoire sinusoïdale

La trajectoire désirée est de la forme suivante

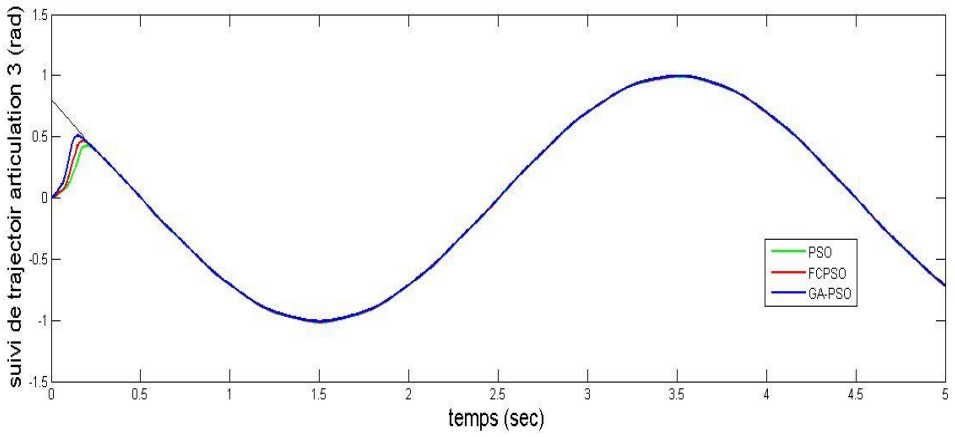
$$\theta_d = u(t) = \begin{cases} \sin\left(\pi t + \frac{3\pi}{4}\right) \ rad.\ rad\ \ pour\ t \geq 0 \\ 0 \qquad\qquad ailleurs \end{cases}$$



(a)

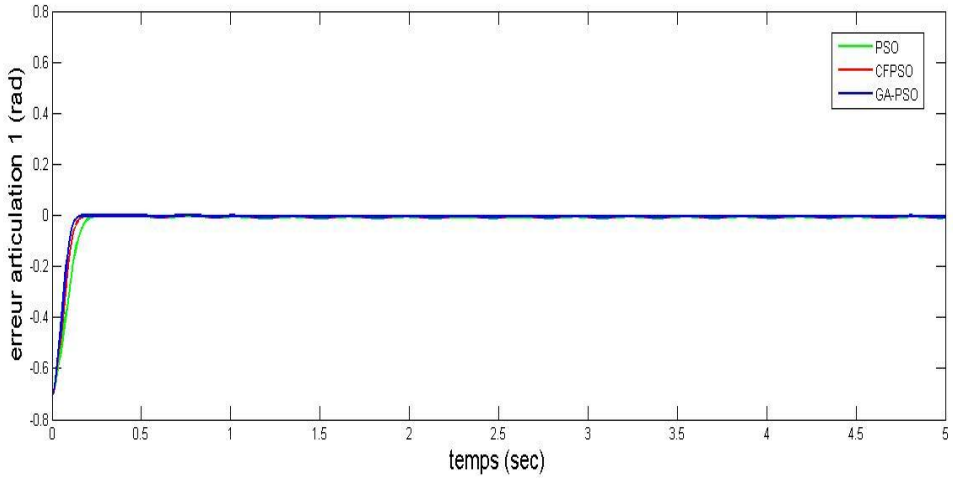


(b)

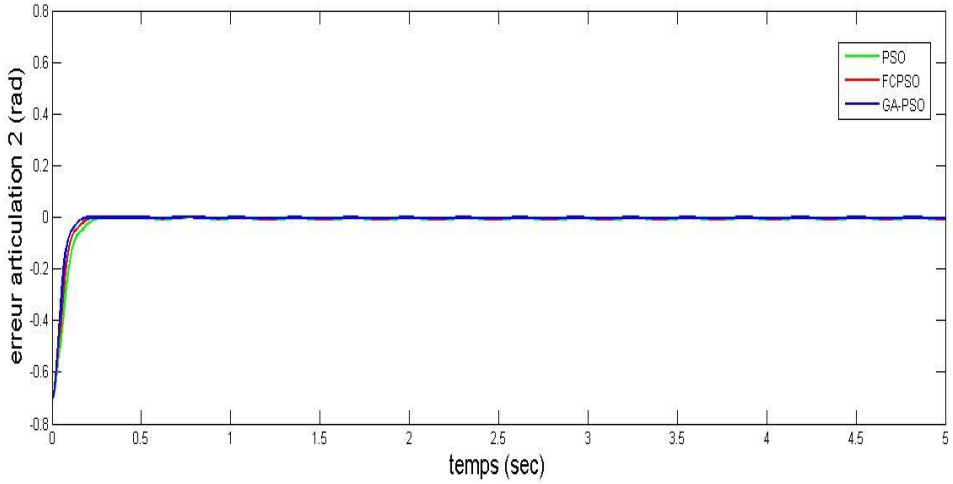


(c)

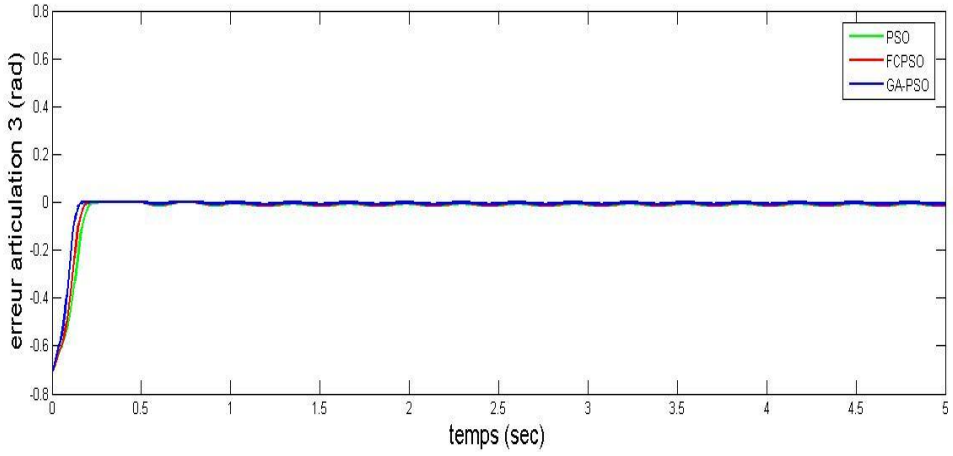
Figure IV-14 : Suivi d'une trajectoire sinusoïdale pour les trois articulations.



(a)

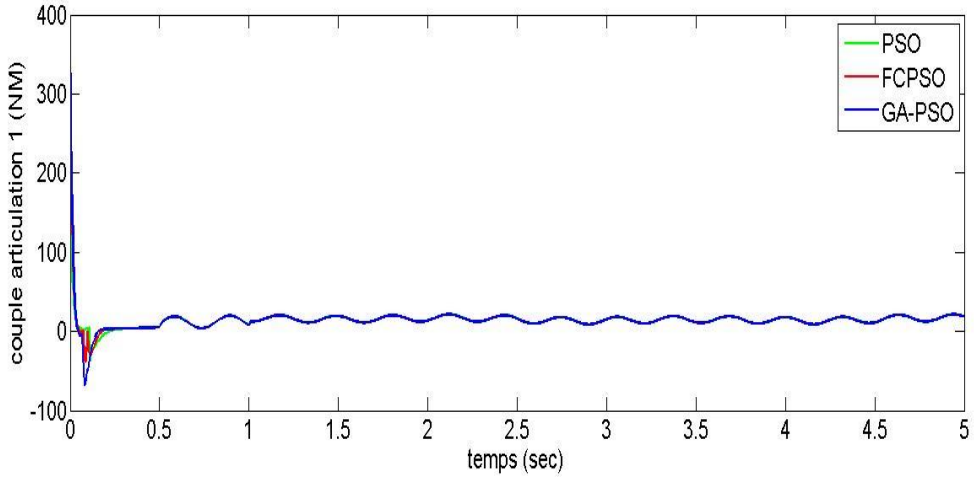


(b)

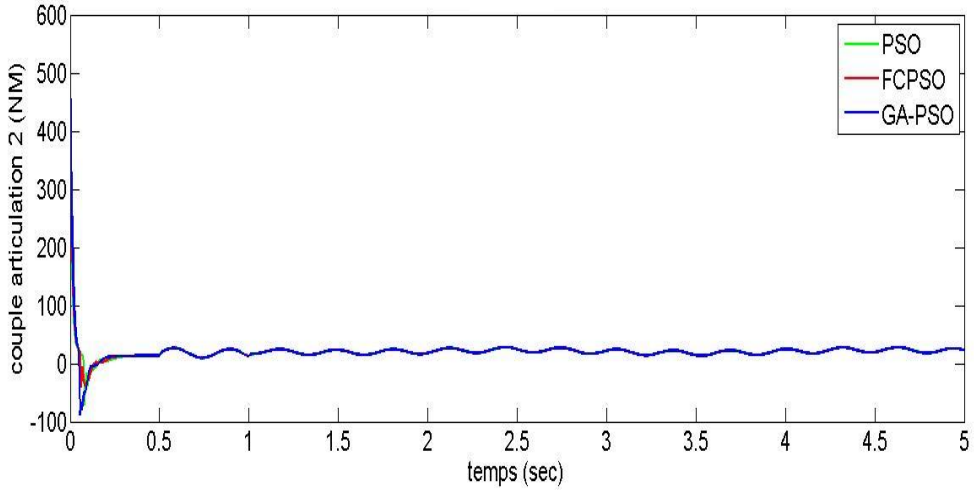


(c)

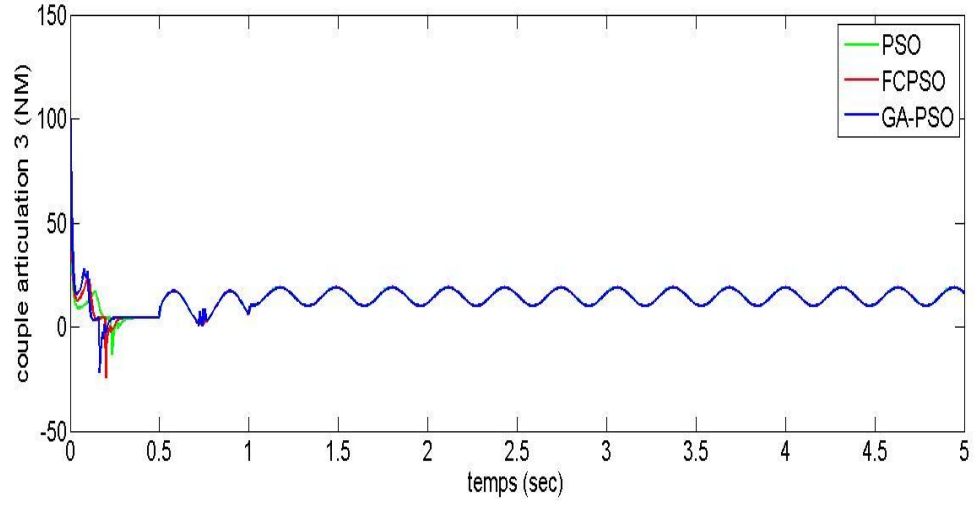
Figure IV-15 : Erreurs de suivi d'une trajectoire sinusoïdale pour les trois articulations.



(a)



(b)



(c)

Figure IV-16 : Couples générés pour le suivi d'une trajectoire sinusoïdale pour les trois articulations.

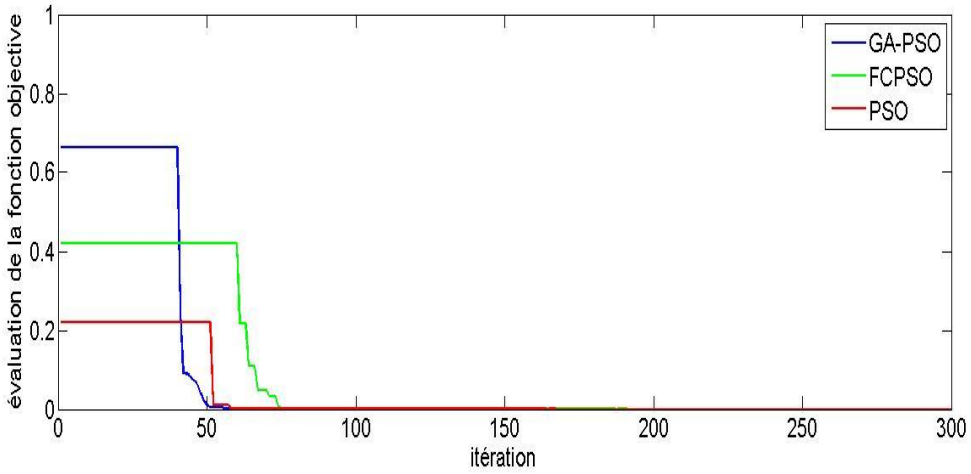


Figure IV-17 : Evaluation de la fonction objective

IV.5.2 Commande par mode glissant floue adaptative

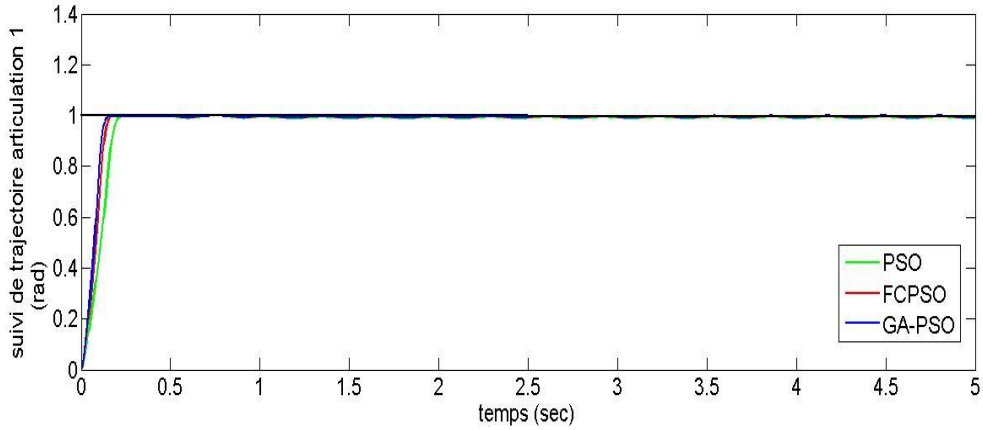
Les mêmes travaux de simulation, que dans le cas précédent, sont entrepris en considérant cette fois-ci la méthode de commande adaptative floue par mode de glissement. Un superviseur flou est ajouté au contrôleur précédent pour ajuster le gain K_f . Deux types de trajectoires sont considérés en utilisant les mêmes algorithmes. .

1) Régulation

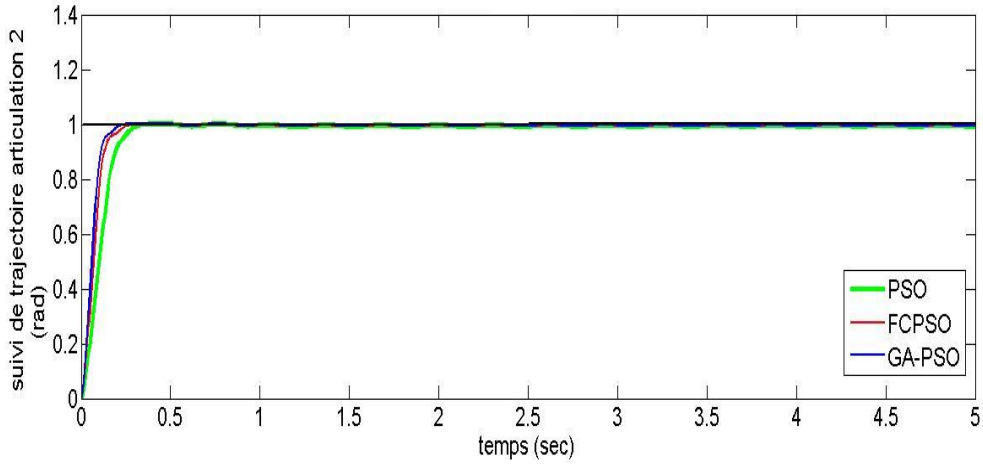
La trajectoire désirée pour la régulation des trois articulations est:

$$\theta = u(t) = \begin{cases} 1 \text{ rad pour } t \geq 0 \\ 0 \quad \text{ailleurs} \end{cases}.$$

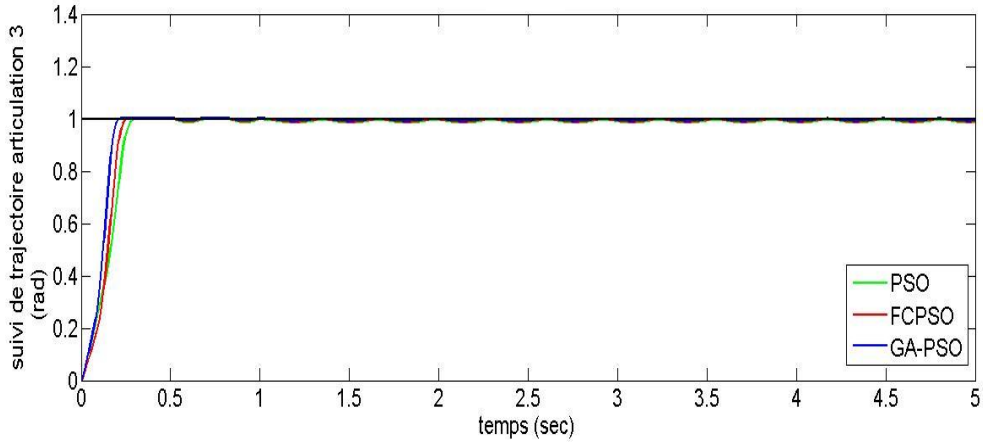
Les matrices K_1, K_2 sont choisies de façon automatique à l'aide des trois algorithmes d'optimisation PSO, CFPSO et GA-PSO.



(a)

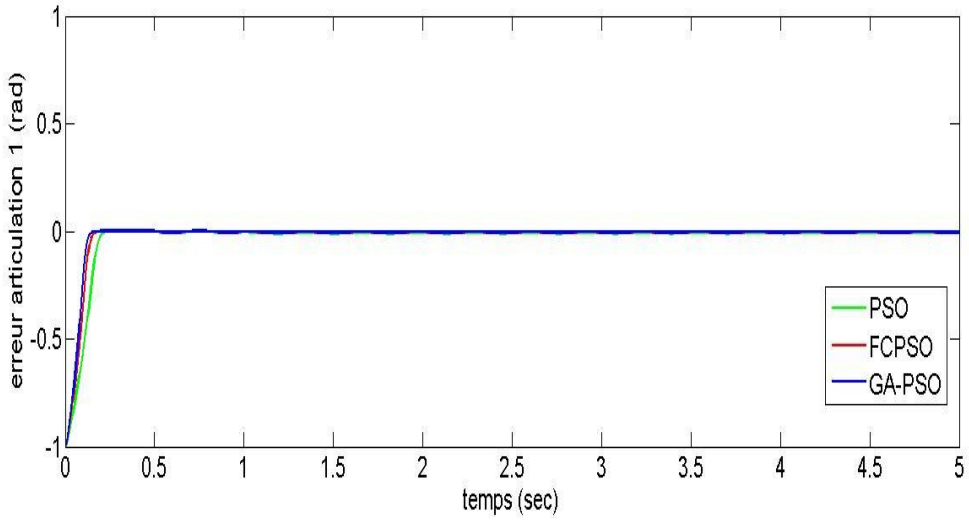


(b)

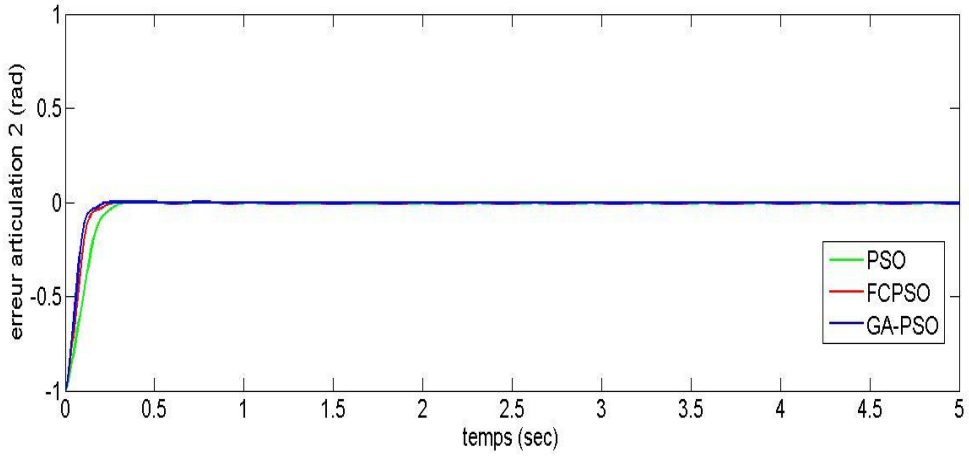


(c)

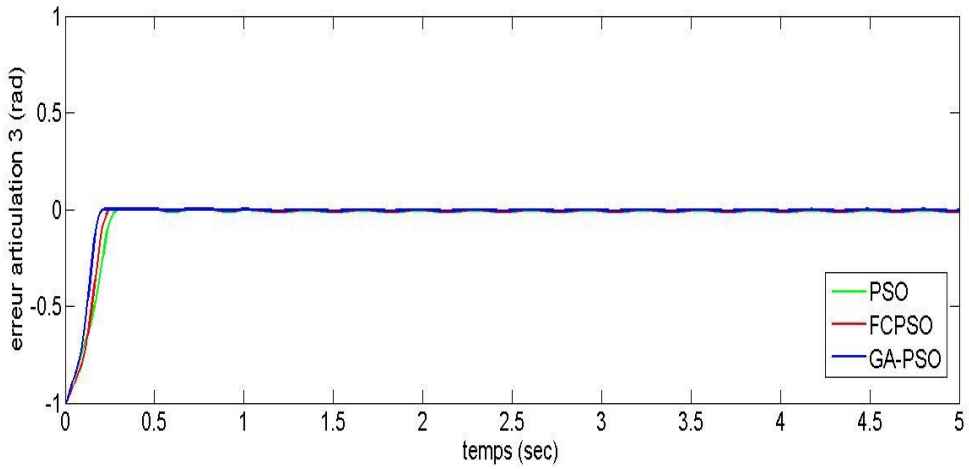
Figure IV-18 : Suivi de trajectoires pour les trois articulations.



(a)

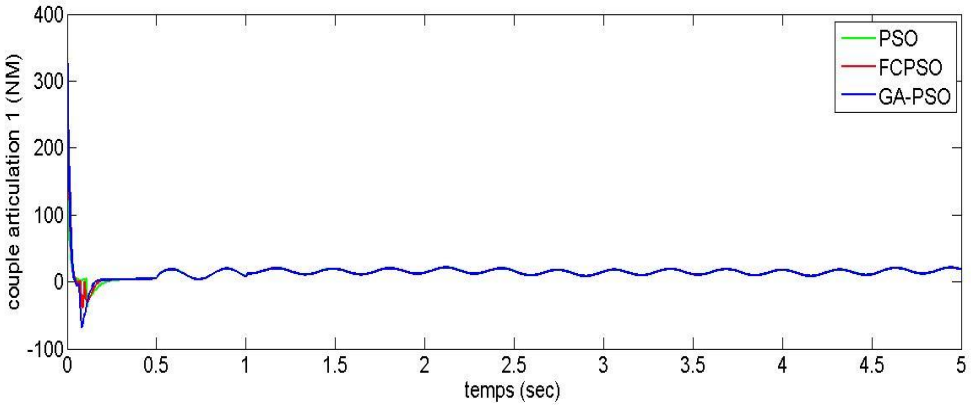


(b)

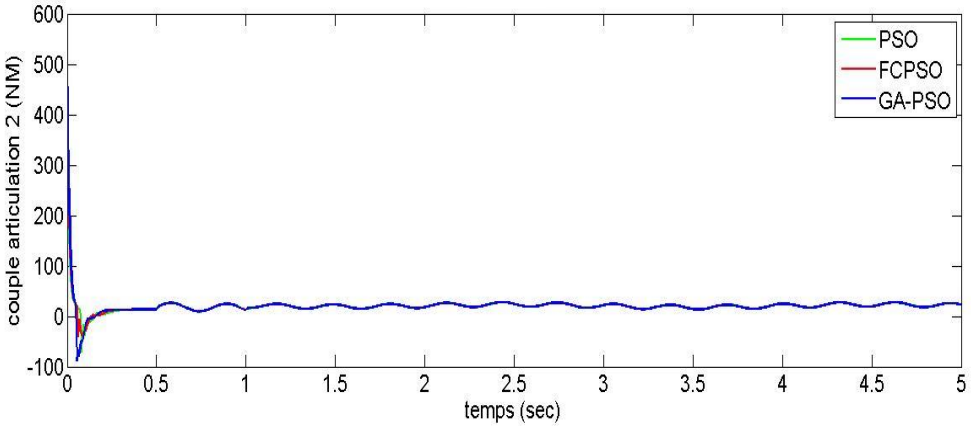


(c)

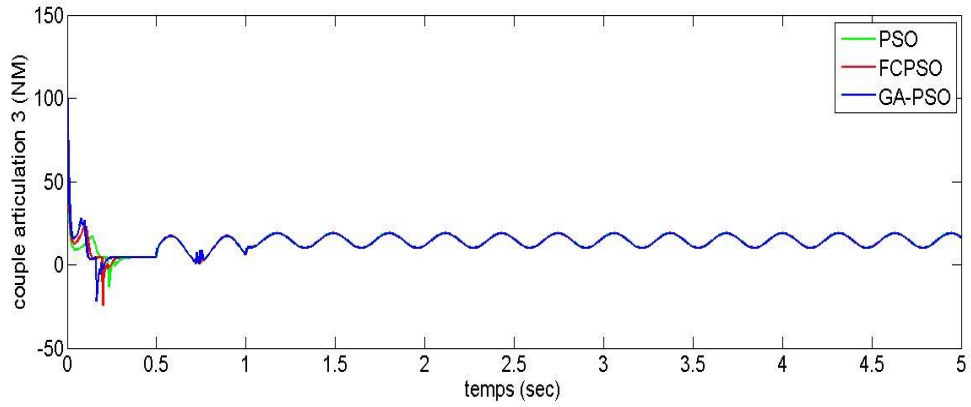
Figure IV-19 : Erreurs de suivi pour les trois articulations.



(a)



(b)



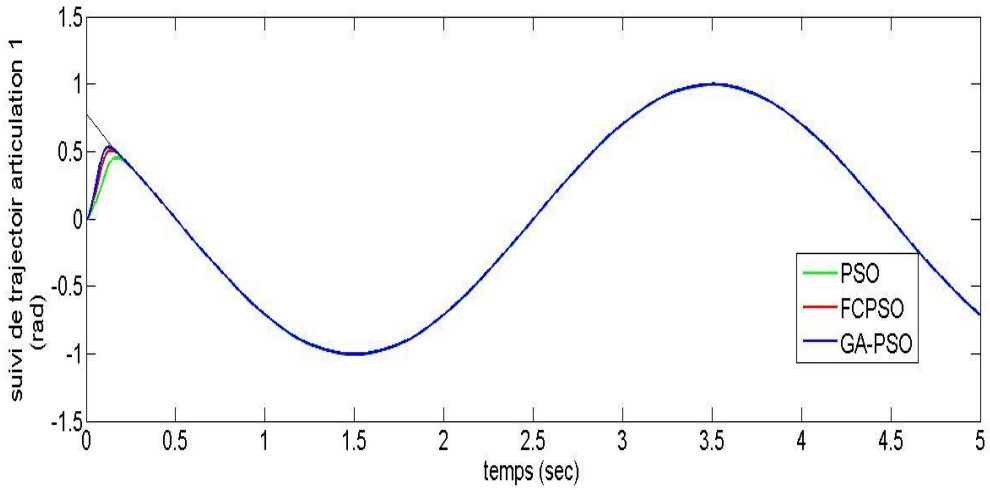
(b)

Figure IV-20 : Couples générés pour le suivi de trajectoires pour les trois articulations.

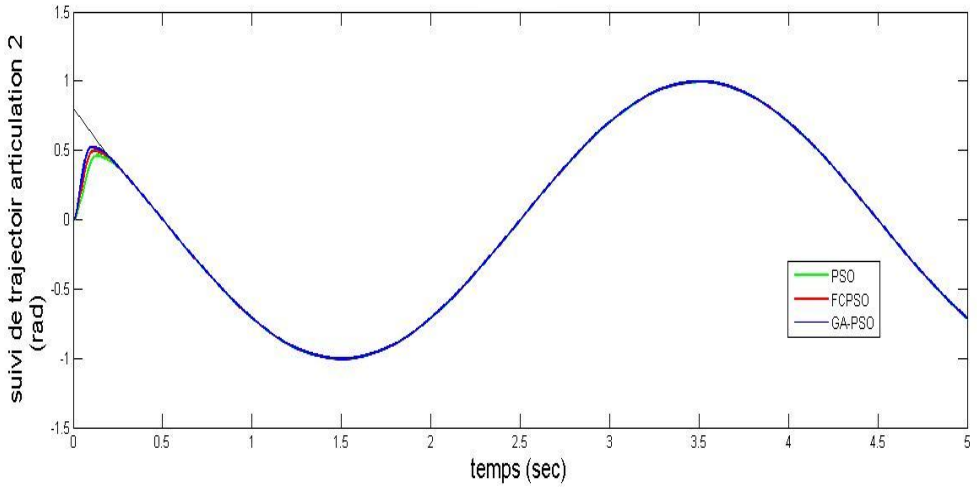
2) Trajectoire sinusoïdale

La trajectoire désirée est de la forme suivante :

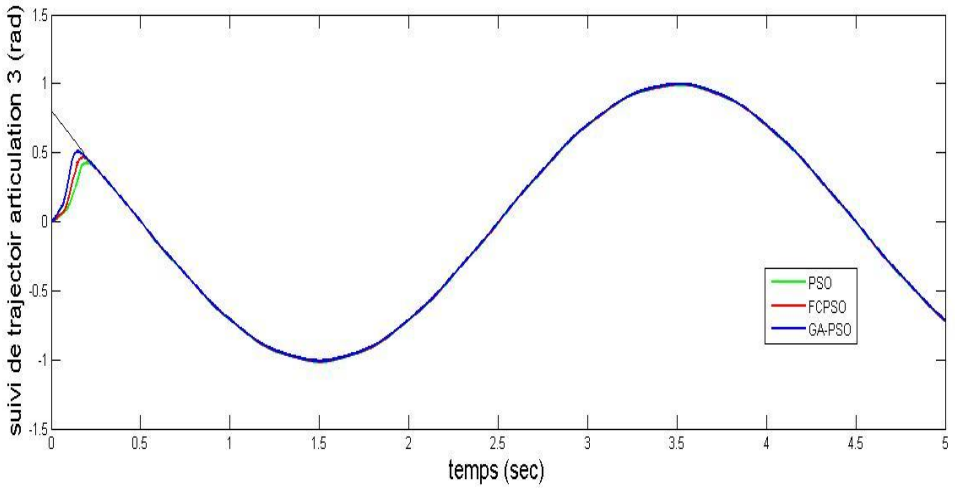
$$\theta_d = u(t) = \begin{cases} \sin\left(\pi t + \frac{3\pi}{4}\right) \ rad.\ rad\ \ pour\ t \geq 0 \\ 0 \qquad\qquad ailleurs \end{cases}$$



(a)

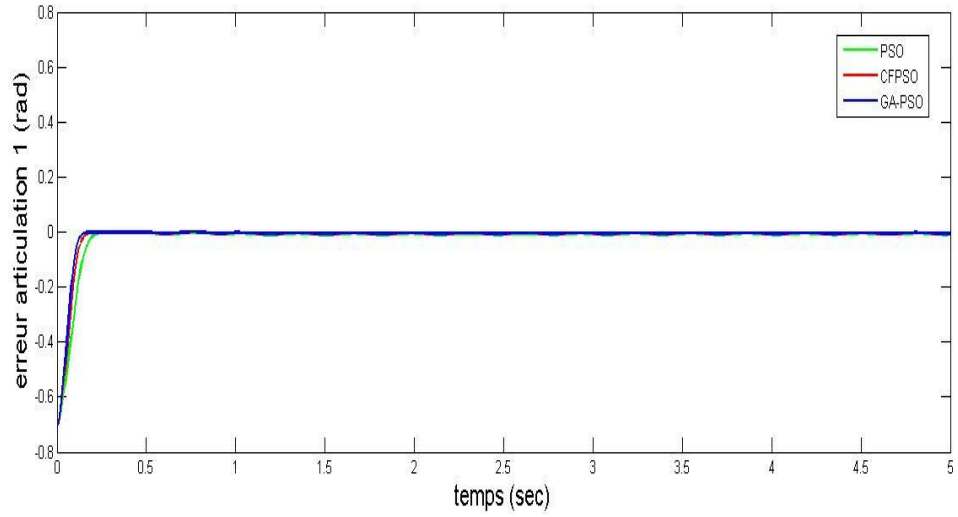


(b)

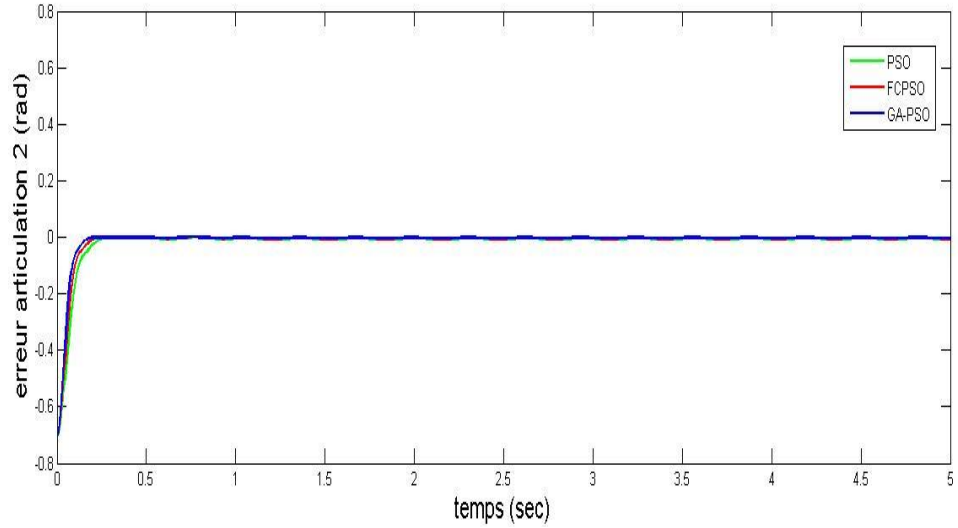


(c)

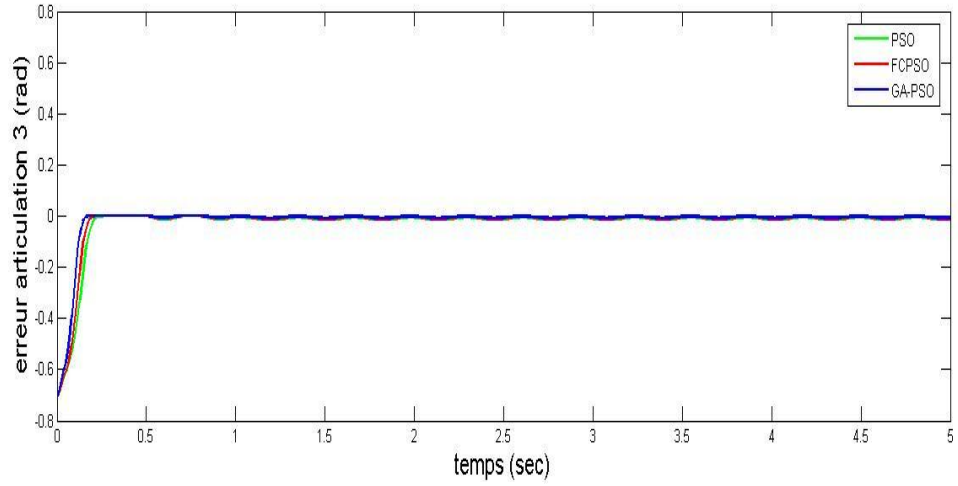
Figure IV-21 : Suivi d'une trajectoire sinusoïdale pour les trois articulations.



(a)

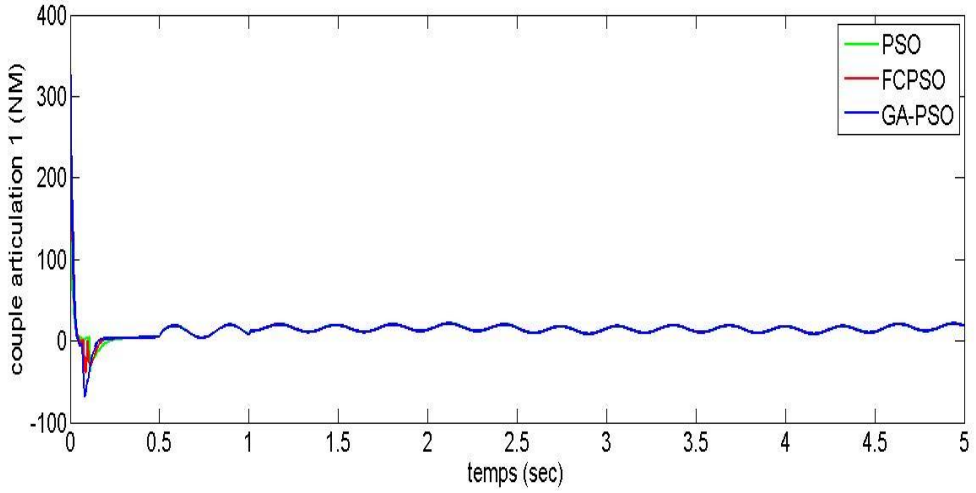


(b)

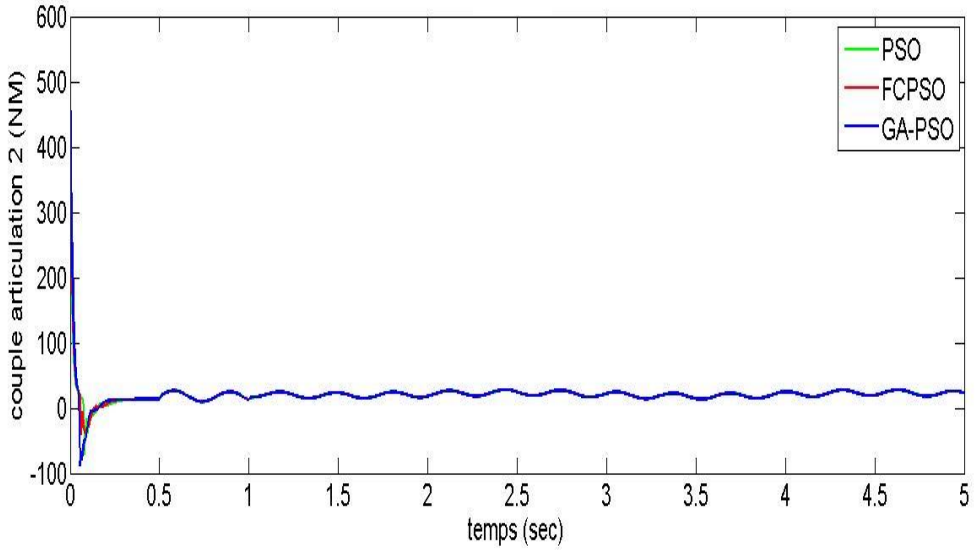


(c)

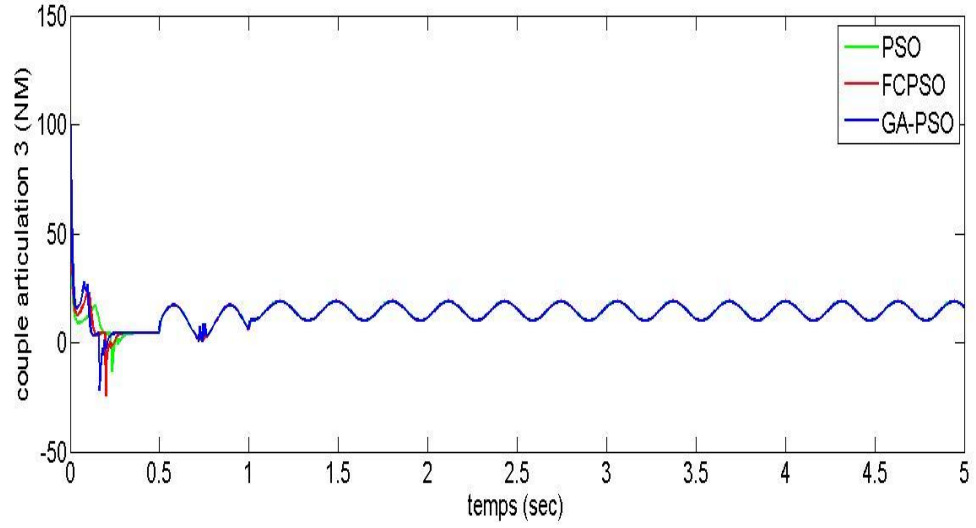
Figure IV-22 : Erreurs de suivi d'une trajectoire sinusoïdale pour les trois articulations.



(a)



(b)



(c)

Figure IV-23 : Couples générés pour le suivi de la trajectoire pour les trois articulations.

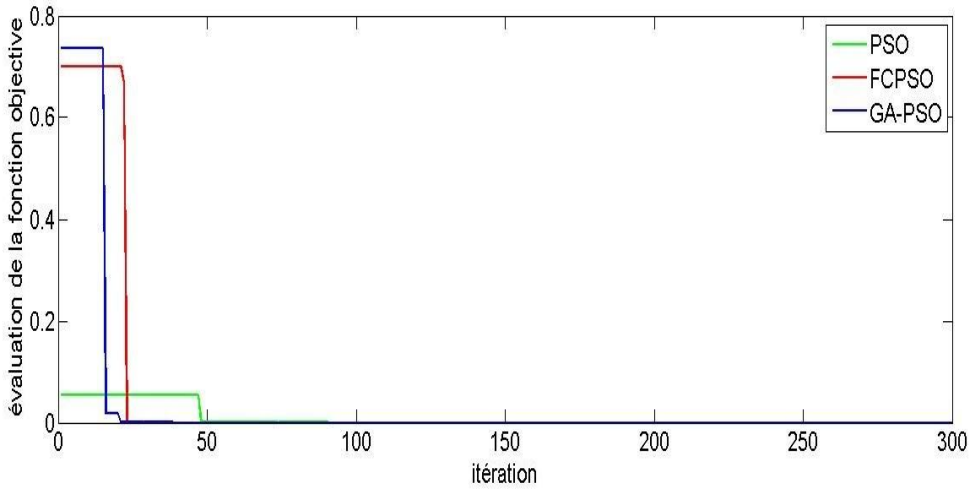


Figure IV-23 : Evaluation de la fonction objective

Les travaux de simulation réalisés montrent que la précision des résultats trouvés par le GA-PSO est meilleure que celle du FCPSO et du PSO et le temps de calcul est moindre. On remarque en plus que l'utilisation du contrôleur flou réduit le temps de convergence et élimine le phénomène de réticence.

IV.6 Conclusion

Dans ce chapitre et dans le but de tester les performances et la robustesse des algorithmes d'optimisation étudiés, on a fait une description théorique de la commande par mode glissant floue et floue adaptative, et appliqué ces commandes sur un bras manipulateur à trois degrés de liberté, ensuite on a optimisé les paramètres de ces commandes en utilisant les trois algorithmes étudiés.

Les résultats de simulation montrent que le GA-PSO donne un résultat plus précis dans un temps de calcul réduit par rapport au FCPSO et PSO.

Conclusion Générale

Les essaims de particules ont pour particularité d'être l'un des algorithmes métaheuristiques le plus simple en terme de complexité d'équations. Ainsi seule la mémorisation du meilleur globale g et du meilleur individuel x_{pi} est nécessaires pour le calcul de l'itération suivante en utilisant uniquement deux équations. Cette particularité est intéressante dans le cadre d'une implémentation dans un système à faibles ressources informatiques ou encore soumis à des contraintes de type temps réel.

L'objectif de notre travail est la détermination des paramètres optimaux de la commande à appliquer à un bras manipulateur à trois degrés de liberté pour le suivi de trajectoires. Trois types de commandes ont été appliqués à notre système.

La première partie de notre travail est portée sur la modélisation des bras manipulateurs où on a vu les différents types de modèles utilisés pour la description des mouvements articulaires, ensuite on a présenté une description de quelques types d'algorithmes d'optimisation par essaim de particules pour optimiser les paramètres de notre système.

Dans la deuxième partie on a fait une étude théorique de la commande par mode glissant, ensuite on a développé une loi de commande par mode glissant pour contrôler notre manipulateur et on a utilisé les trois algorithmes d'optimisation considérés pour le choix des paramètres de la commande (surface et gain) pour que l'erreur de suivi de trajectoire et les couples générer soient minimaux.

Dans la troisième partie on a étudié une commande floue et une commande floue adaptative, ensuite on a développé les lois de commande en utilisant les résultats obtenus dans la deuxième partie de notre travail pour ensuite utiliser les trois algorithmes considérés pour le choix des gains des contrôleurs flous pour minimiser les couples générés et l'erreur de suivi de trajectoires.

Plusieurs travaux de simulation ont été réalisés et les résultats obtenus montrent que les améliorations faites sur l'algorithme PSO de base sont efficaces, elles améliorent la précision des résultats et diminuent le temps de calcul. L'introduction de la logique floue sur la loi de commande par mode glissant augmente aussi les performances de la commande classique vis-à-vis de la minimisation du phénomène de réticence et l'ajout du superviseur flou donne une certaine flexibilité par rapport aux perturbations externes. Les travaux entrepris montrent que la commande adaptative permet une convergence plus rapide quant au suivi de trajectoires.

Les améliorations faites sur l'algorithme PSO de base sont très variées mais les résultats obtenus montrent que la méthode hybride GA-PSO est plus efficace et efficiente pour l'optimisation des systèmes car elle combine la vitesse de convergence du PSO et l'obtention efficace de l'optimum globale des algorithmes génétiques (GA).

Notre démarche consiste alors en une étude et exploitation des algorithmes PSO qui peuvent être d'une efficacité face à n'importe quel problème notamment au problème de choix des paramètres des commandes.

A cet effet, leur application a montré leur intérêt dans ce type de domaine et ouvre de nombreuses perspectives de recherche pour l'avenir telles que :

- Application des algorithmes étudiés sur d'autres types de commande des systèmes non linéaires.
- Utilisation de ces algorithmes avec d'autres méthodes d'intelligence artificielle telles que les réseaux de neurones, la logique floue type 2,…….
- Amélioration des performances des algorithmes PSO par hybridation avec d'autres algorithmes d'optimisation tels que le Ant Colony et le Métaheuristique.

Références

[1] Wild Beightler, Phillips. "Foundations of optimization. 2nd ed".Englewood Cliffs, 1979.

[2] Mark W. Spong, Seth Hutchinson, and M. Vidyasagar, "Robot Dynamics and Control" 2eme edition

[3] Reza N. Jazar, "Theory of Applied Robotics", 2eme edition.

[4] M. Hacene "contribution à l'étude de la programmation et de la commande d'un robot manipulateur " thèse du doctorat.

[5] Abdelmalek Gacem " Utilisation des méthodes d'optimisations métaheuristiques pour la résolution du problème de répartition optimale de la puissance dans les réseaux électriques", thèse magister Centre Universitaire d'El-oued.

[6] Abbas El Dor. "Perfectionnement des algorithmes d'optimisation par essaim particulaire : applications en segmentation d'images et en électronique". thèse de doctorat en informatique Université Paris-Est, 2012.

[7] Y. Shi and E. R. C. "Empirical study of particle swarm optimization". Vol. 3

[8] Amaresh Sahua, Sushanta Kumar Panigrahib, Sabyasachi Pattnaikc. "Fast Convergence Particle Swarm Optimization for Functions Optimization" science direct Procedia Technology 4 (2012)

[9] Hanaa Hachimi "Hybridations d'algorithmes métaheuristiques en optimisation globale et leurs applications", thèse de doctorat en optimisation des structures, Institut national des sciences appliquées de Rouen 2013.

[10] Jean-Jacques E. Slotine, Weiping Li, "Applied Nonlinear Control"

[11] Abdelghani El Ougli, "Intégration des techniques floues à la synthèse de contrôleurs adaptatifs", thèse de doctorat en Automatique, Signaux et Systèmes, Fès, Maroc, 2009.

[12] Kai Michels, Frank Klawonn, Rudolf Kruse, Andreas Nürnberger, "Fuzzy Control".

[13] Ahmed F. Amer, Elsayed A. Sallam, Wael M. Elawady, "Adaptive fuzzy sliding mode control using supervisory fuzzy control for 3 DOF planar robot manipulators", Applied Soft Computing 11 (2011).

[14] M. Roopaei, M. Zolghadri Jahromi, "Chattering-free fuzzy sliding mode control in MIMO uncertain systems", Nonlinear Analysis 71 (2009).

[15] R.C. Eberhart, J. Kennedy. "A new optimizer using particle swarm theory". Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, 1995.

[16] R. Eberhart, J. Kennedy, and Y. Shi. "Swarm intelligence. Evolutionary Computation" Morgan Kaufmann, 2001.

[17] Yuzheng Guo, Peng-Yung Woo, "Adaptive Fuzzy Sliding Mode Control for Robotic Manipulators", Proceedings of the 42nd IEEE, Conference on Decision and Control, 2003.

[18] Philippe Mullhaupt, "Introduction à l'Analyse et à la Commande des Systèmes Non Linéaires".

[19] D.H. Kim, A. Abraham, and K. Hirota "Hybrid Genetic: Particle Swarm Optimization Algorithm" springer Volume 75 of the series Studies in Computational Intelligence

[20] M. Clerc, J. Kennedy. "The particle swarm: explosion, stability, and convergence in multidimensional complex space". IEEE Trans, on evolutionary computation 2002.

[21] R. Eberhart and Y. Shi. "Comparing inertia weights and constriction factors in particle swarm optimization". In: Proceedings of the 6th IEEE Congress on Evolutionary Computation, IEEE Press 2000.

[22] Seung-Bok Choi and Jung-Sik Kim, "A Fuzzy Sliding Mode Controller for Robust Tracking of Robotic Manipulators", Mechatronics Vol. 7, No. 2, pp. 199-216, 1997.

[23] S. Aloui, O. Pages, A. El Hajjaji, A. Chaari, Y. Koubaa, "Improved fuzzy sliding mode control for a class of MIMO nonlinear uncertain and perturbed systems", Applied Soft Computing 11 (2011).

[24] Neil Munro, Frank L.Lewis, "Robot Manipulator Control: Theory and Practice", 2eme

[25] Nurkan Yagiz, Yuksel Hacioglu, "Robust control of a spatial robot using fuzzy sliding mode", Mathematical and Computer Modelling 49 (2009).

[26] Samuel, G.G.; Asir Rajan, C.C. "Hybrid Particle Swarm Optimization — Genetic algorithm and ParticleSwarm Optimization — Evolutionary programming for long-term generation maintenance scheduling Renewable Energy and Sustainable Energy (ICRESE) ", 2013 International Conference, 2013

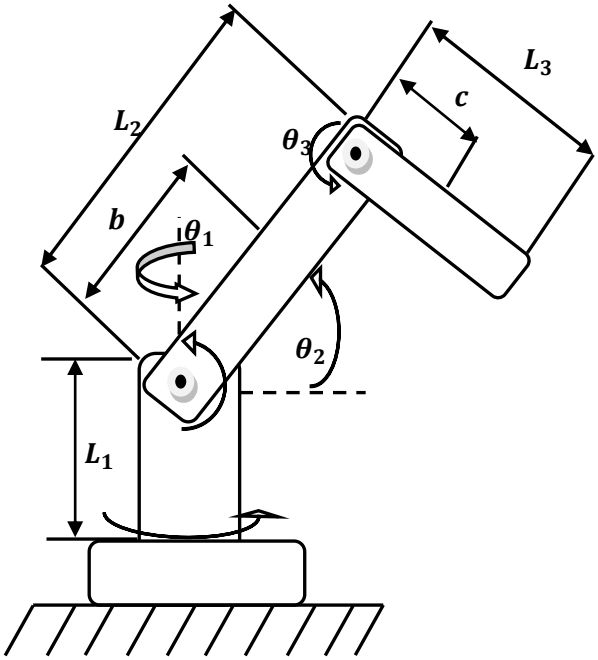
[27] W. Li, X.G. Chang, F.M. Wahl, Jay Farrell, "Tracking control of a manipulator under uncertainty by FUZZY PID controller ", Fuzzy Sets and Systems 122 (2001).

[28] Thomas R. Kurfess, "Robotics and Automation Handbook", 2005.

[29] N.Azoui, "Commande non linéaire d'un bras manipulateur", Thèse de Magister, département d'électronique université de Batna 2009,

ANNEXE A

Modèle dynamique d'un bras manipulateur 03 DDL



$$M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + G(\theta) = U + P$$

$M(1,1) = m_2 b^2 \cos^2\theta_2 + m_3(L_2\cos\theta_2 + c.\cos(\theta_2 + \theta_3))^2 + I_1$

$M(1,2) = 0$

$M(1,3) = 0$

$M(2,1) = 0$

$M(2,2) = [m_2 b^2 + m_3(L_2^2 + c^2 + 2cL_2\cos\theta_3) + I_2]$

$M(2,2) = m_3(c^2 + cL_2\cos\theta_3)$

$M(3,1) = 0$

M(3,2) = $m_3(c^2 + cL_2 \cos\theta_3)$

M(3,3) = $[m_3c^2 + I_3]$

C(1,1)= $\left[b_{t1} - m_2b^2 \sin(2\theta_2)\dot{\theta}_2\right] +$
$2m_3\left[-L_2\dot{\theta}_2 \sin\theta_2 - c.\sin(\theta_2 + \theta_2)\left(\dot{\theta}_2+\dot{\theta}_3\right)\right][L_2 \cos\theta_2 + c.\cos(\theta_2 + \theta_3)]$

C(1,2) = 0

C(1,3) = 0

C(2,1) = $\left[\frac{1}{2}\sin(2\theta_2)(m_2b^2 + m_3L_2^2) + m_3cL_2 \sin(2\theta_2 + \theta_3) + \frac{1}{2}m_3c^2 \sin\left(2(\theta_2 + \theta_3)\right)\right]$

C(2,2) = $b_{t2} - m_3cL_2 \sin\theta_3$

C(2,2) = $-m_3cL_2 \sin\theta_3$

C(3,1) = $m_3\left[\frac{1}{2}L_2^2 \sin(2\theta_2) + cL_2 \sin(2\theta_2 + \theta_3) + \frac{1}{2}c^2 \sin\left(2(\theta_2 + \theta_3)\right)\right]$

C(3,2) = $m_3cL_2 \sin\theta_3$

C(3,3) = b_{t3}

$G(1)$ = 0

$G(2) = cos\,\theta_2\,(m_2gb + m_3gL_2) + m_3gc.cos(\theta_2 + \theta_3)$

G(3) = $m_3gc.cos(\theta_2 + \theta_3)$

P(t) = $20 + 20\sin\left(20(t-1)\right) + 30\sin\left(10(t-0.5)\right) + 20u(t-0.5) + 20u(t-1)$

Avec :

m_1=3; m_2=2; m_3=1

L_1=0.5; L_2=0.5; L_3=0.3

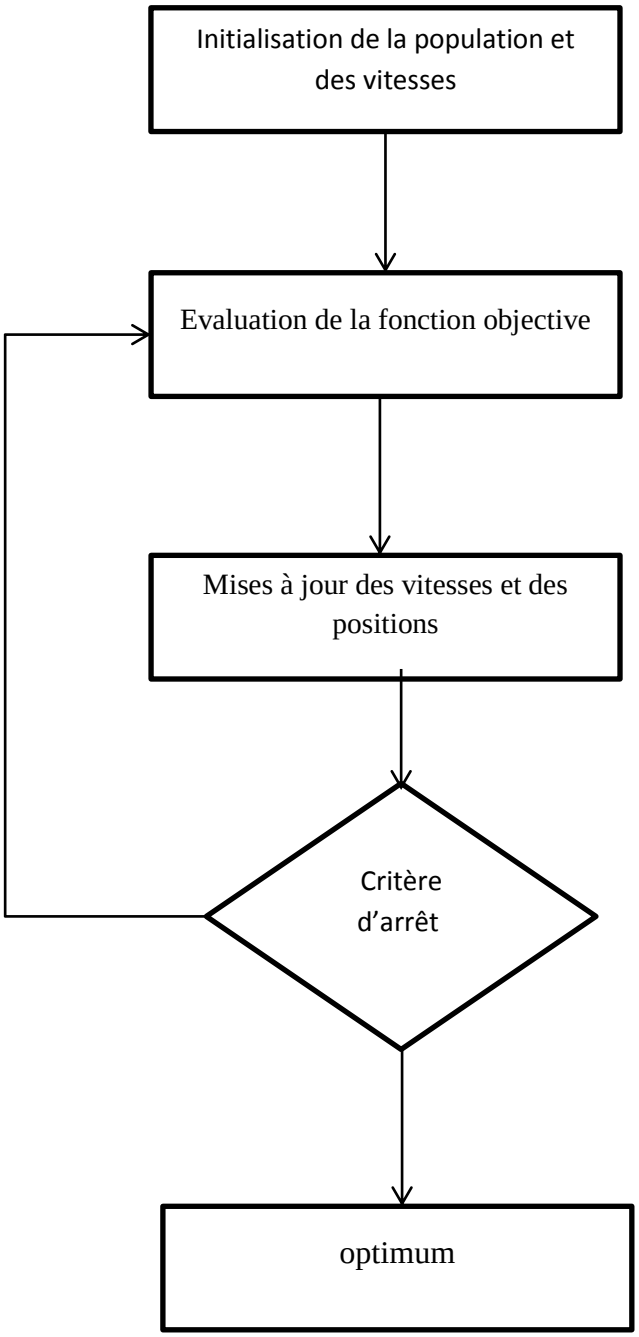
I_1=0.052; I_2=0.052; I_3=0.052;

b=0.25; c=0.15

b_{t1}=0.4; b_{t2}=b_{t1}; b_{t3}=b_{t2};

ANNEXE B

Organigramme PSO



Organigramme GA-PSO