REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITE L'ARBI BEN M'HIDI D'OUM-EL-BOUAGHI
INSTITUT DES SCIENCES TECHNOLOGIQUE DE AIN BEIDA
DEPARTEMENT DE GENIE MECANIQUE

MEMOIRE

Présenté en vue d'obtenir Ie diplôme de Magister

Spécialité : Robotique et Systèmes Intelligents

Présenté par

ZERROUKI NADJIB

Ingénieur en Electronique

OPTIMISATION MULTIOBJECTIF PAR LES ALGORITHMES GENETIQUES, APPLICATION A L'IDENTIFICATION ET LA COMMANDE

Soutenue Ie 29 avril 2009 devant Ie jury composé de:

President:	Zaatri A. El Ouahab	Prof	Université de Constantine
Rapporteur :	Goléa Nourreddine	Prof	Université d'Oum El Bouaghi
Examinateurs:	Djouambi A. El Baki	M.C	Université d'Oum El Bouaghi
	Mahfoudi Chawki	M.C	Université d'Oum El Bouaghi

Résumé

L'Optimisation traite le problème de la recherche des solutions sur un ensemble de choix possibles afin d'optimiser certains critères. S'il y un seul critère, nous avons un problème d'optimisation mono objectif, un type d'études approfondies au cours des 50 dernières années. S'il existe plusieurs critères et qui doivent être traités simultanément, nous avons un problème d'optimisation multiobjectif. Les problèmes multiobjectifs se posent dans la conception, la modélisation et la planification d'un grand nombre des systèmes complexes dans les domaines de la production industrielle, les transports urbains, la gestion des forêts, la gestion des réservoirs, l'agencement et l'aménagement des plans des nouvelles villes, et la distribution de l'énergie, par exemple. Presque chaque problème de décision réel important implique des objectifs multiples et contradictoires qui doivent être abordés tout en respectant les diverses contraintes, ce qui conduit à la complexité du problème. Les algorithmes génétiques ont reçu une attention considérable comme approche originale aux problèmes d'optimisation multiobjectifs, ayant pour résultat plusieurs recherches et applications connues sous le nom des algorithmes génétiques multiobjectifs. Dans ce travail, nous proposons d'aborder ce problème dans le domaine du contrôle à l'aide d'algorithmes génétiques. Les algorithmes génétiques sont un outil d'optimisation heuristique qui permet de résoudre des problèmes d'optimisation, sans évaluer des gradients et sans tomber dans les problèmes des minimums locaux.

Mots clés : Optimisation Multiobjectif, Algorithmes Génétiques, Contrôle.

Abstract

Optimization deals with the problem of seeking solutions over a set of possible choices to optimize certain criteria. If there is only one criterion to consider, it becomes a single-objective optimization problem, a type studied extensively for the past 50 years. If there is more than one criterion and they must be treated simultaneously, we have a multiple-objective optimization problem. Multiple-objective problems arise in the design, modeling and planning of many complex real systems in the areas of industrial production, urban transportation, forest management, reservoir management, layout and landscaping of new cities, and energy distribution, for example. Almost every important real-world decision problem involves multiple and conflicting objectives that need to be tackled while respecting various constraints, leading to overwhelming problem complexity. Genetic algorithms have received considerable attention as a novel approach to multi-objective optimization problems, resulting in a fresh body of research and applications known as genetic multi-objective optimizations. In this work, we propose to approach this problem in the area of control by using genetic algorithms. Genetic algorithms are a heuristic tool for optimization that can solve optimization problems, without evaluating gradients and without falling into the problems of local minimum.

Key- words: Multi-objective Optimization, Genetic Algorithms, Control.

Remerciements

Je remercie en premier lieu mon Dieu tout puissant clément et miséricordieux de m'avoir soigné et aidé.

Je voudrais exprimer toute ma reconnaissance à mon directeur de mémoire, le Professeur Noureddine Goléa pour son appui scientifique, mais aussi moral; la bonne humeur dont il a toujours fait preuve est un élément de motivation non négligeable. Je suis ravi d'avoir travaillé en sa compagnie car outre sa totale disponibilité, son suivi permanent et ses critiques fondées qui m'ont permis d'avancer dans mon mémoire, il a su me donner la confiance nécessaire à l'aboutissement de ce travail.

Je remercie les membres de jury qui ont accepté de juger ce travail et d'y apporter leur caution :

Monsieur Dr. A. Zaatri, Professeur à l'université de Constantine, qui me fait le grand honneur d'accepter cordialement la présidence du jury.

Monsieur Dr. A. Djouambi, maître de conférences à l'université d'Oum El Bouaghi, pour l'honneur qu'il me fait en acceptant de participer à ce jury, d'être examinateur de ce mémoire.

Monsieur, Dr. C. Mahfoudi, maître de conférence à l'université d'Oum El Bouaghi, pour l'honneur qu'il me fait en acceptant également de participer à ce jury, d'être examinateur de ce mémoire.

J'adresse mes vifs remerciements à tous les enseignants qui, par leur enseignement, leur encouragement et leur aide, ont contribués à ma formation durant toutes mes études à l'université d'Oum El Bouaghi.

Je remercie mes parents pour leur soutient et sacrifice, pas seulement durant la période de recherche, mais durant toutes mes années d'études.

LISTE DES FIGURES

LISTE DES TABLEAUX

Introduction générale

Introduction générale

Depuis une trentaine d'années, le domaine d'optimisation multiobjectif connaît une évolution importante. Cette évolution s'est traduite par le développement d'un grand nombre de méthodes. La multitude des méthodes d'optimisation multiobjectif est perçue comme une richesse incontestable de ce domaine. D'ailleurs, certains la justifient par la diversité des problèmes ainsi que par l'existence de différentes approches de résolution possibles et légitimes de ces problèmes.

Un problème d'optimisation multiobjectif diffère d'un problème mono objectif parce qu'il contient plusieurs objectifs qui exigent l'optimisation simultanément. En optimisant un problème mono objectif, la meilleure solution trouvée est le but. Mais pour des problèmes multiobjectifs, avec plusieurs objectifs (probablement contradictoires), il n'y a habituellement aucune solution optimale. Par conséquent, le décideur est exigé pour choisir une solution à partir d'un ensemble fini en faisant des compromis. Une solution appropriée devrait fournir une performance acceptable pour plus de tous les objectifs.

Dans la plupart des problèmes d'optimisation il y a toujours des limitations imposées par les caractéristiques particulières de l'environnement ou des ressources disponibles (par exemple, limitations physiques, limitations de temps, etc.). Ces limitations doivent être satisfaites afin de considérer une certaine solution acceptable. Toutes ces limitations en général s'appellent les contraintes, et elles décrivent les dépendances entre les variables de décision et des constantes (ou des paramètres) impliqués dans le problème.

L'optimisation sous différentes contraintes, souvent contradictoires ou concurrentes, est un problème d'actualité dans multiples domaines de recherche. La solution de ce problème repose sur des développements mathématiques très complexes et nécessite l'évaluation des gradients des fonctions coût, chose qui n'est pas toujours facile. Dans ce travail, on propose d'approcher ce problème, dans le domaine de contrôle, par l'utilisation des algorithmes génétiques. Les algorithmes génétiques sont un outil d'optimisation heuristique qui permet de résoudre des problèmes d'optimisation, sans évaluer des gradients et sans tomber dans les problèmes des minimums locaux. Elles sont adaptées à la recherche de surface de Pareto puisqu'ils travaillent sur une "population" de solutions candidates, à la différence de la plupart des méthodes traditionnelles, qui manipulent en général un unique point de l'espace de recherche. Cette particularité des algorithmes génétiques rend possible l'obtention d'un ensemble de

compromis de Pareto approché en un seul essai de l'algorithme, et sans obliger l'utilisateur a définir des paramètres subjectifs supplémentaires tel que les poids relatifs des critères d'optimisation.

Résoudre un problème comprenant plusieurs critères, appelé communément problème multiobjectif, nécessite l'étude de trois points particuliers :

- La définition de l'ensemble des solutions réalisables,
- L'expression de l'objectif à optimiser,
- Le choix de la méthode d'optimisation à utiliser.

Les deux premiers points relèvent de la modélisation du problème, le troisième de sa résolution. Afin de définir l'ensemble des solutions réalisables, il est nécessaire d'exprimer l'ensemble des contraintes du problème. Ceci ne peut être fait qu'avec une bonne connaissance du problème sous étude et de son domaine d'application.

Le choix de l'objectif à optimiser requiert également une bonne connaissance du problème. La définition de la fonction objective mérite toute l'attention de l'analyste car rien ne sert à développer des bonnes méthodes d'optimisation si l'objectif à optimiser n'est pas bien défini, il peut être très difficile de trouver un objectif unique d'optimisation. Nous pouvons donc être amené à proposer une optimisation multiobjectif.

Enfin le choix de la méthode de résolution à mettre en œuvre dépendra souvent de la complexité du problème.

L'objectif principal de ce mémoire est l'application des algorithmes génétiques à des problèmes multiobjectifs d'identification et de commande des systèmes. Les exemples d'applications prises sont des 'Benchmarks' très populaires dans le domaine de l'automatique, à savoir le pendule inversé et le robot à deux degrés de libertés.

L'utilisation des algorithmes génétiques pour résoudre des problèmes de cette nature a été motivée principalement en raison de la nature basée sur la population des algorithmes génétiques qui permet la génération de plusieurs éléments de l'ensemble optimal de Pareto dans une seule exécution. En plus, la complexité des problèmes d'optimisation multiobjectifs (par exemple, les espaces de recherche très grands, incertitude, bruit, discontinuité des courbes, etc.) peut empêcher l'utilisation des techniques traditionnelle.

Les algorithmes génétiques ont reçu une attention considérable en ce qui concerne leur potentiel comme une nouvelle technique d'optimisation. Il existe trois avantages majeurs lors de leurs applications aux problèmes optimisation:

1. Adaptabilité : les AGs n'ont pas beaucoup de condition mathématique concernant les problèmes d'optimisation. En raison de la nature évolutionnaire, ils permettront de rechercher des

solutions sans considération de la spécificité du fonctionnement interne du problème. Ils peuvent manipuler n'importe quel genre de fonctions objectives et n'importe quel genre de contraintes, c.-à-d, linéaire ou non linéaire, définies sur des espaces de recherche discrets, continus ou mixtes.

2. Robustesse : L'utilisation des opérateurs d'évolution rend les AGs très efficaces en exécutant une recherche globale (dans la probabilité), alors que la plupart d'heuristique conventionnelle exécute habituellement une recherche locale. On l'a prouvé par beaucoup d'études que les AGs sont plus efficaces et plus robustes en localisant la solution optimale.

3. Flexibilité : Les AGs nous fournissent une grande flexibilité d'hybrider avec des différents domaines pour faire une mise en œuvre efficace pour un problème spécifique.

Ce mémoire est constitué de deux parties. La première commence par une introduction sur les problèmes d'optimisation. Au premier chapitre, nous présentons tout d'abord un ensemble de définitions liées aux problèmes d'optimisation multiobjectifs. Puis nous exposons les approches analytiques d'optimisation multiobjectif. Nous introduisons des concepts de base du domaine de l'optimisation multi-objectif tels que la dominance, la surface de compromis. Nous décrivons aussi les principales approches non Pareto de résolution pour ces problèmes. Ensuite, le deuxième chapitre présente quelques approches Pareto, appartenant à la classe des métaheuristiques, dédiées aux problèmes d'optimisation multiobjectifs. On parle tout d'abord de l'histoire des algorithmes génétiques qui passent des approches non Pareto qui ramanent un problème multiobjectif à un problème mono objectif tel que Vector Evaluated Genetic Algorithm « VEGA » et Hajela et Lin Genetic Algorithm « HLGA » aux approches Pareto qui traitent un véritable problème multiobjectif tel que Multi Objective Genetic Algorithm « MOGA », Nondominated Sorting Genetic Algorithm « NSGA », Niched Pareto Genetic Algorithm « NPGA », et Nondominated Sorting Genetic Algorithm II « NSGA-II ». Il contient également le vocabulaire particulier utilisé dans ce domaine, qui donne la clef pour traduire dans le langage de l'optimisation certains termes et expressions employés par la suite. En outre, Les algorithmes génétiques sont des techniques de recherche et d'optimisation inspirées par le principe de sélection naturelle. Ainsi, ils sont des techniques d'optimisation globale naturellement bien adaptées pour des problèmes sans contrainte et puisque la plupart des problèmes réels impliquent des contraintes, une quantité considérable de recherche a été récemment déclenchés pour résoudre ce problème en utilisant des techniques de gestion des contraintes. Les techniques utilisées pour cet effet sont basées sur l'utilisation des fonctions de pénalité. Dans cette approche, la quantité de violation de contrainte est employée pour pénaliser les individus infaisables de sorte que des individus faisables soient favorisés par le processus de sélection.

La deuxième partie est entièrement consacrée aux applications des algorithmes génétiques pour la résolution des problèmes d'optimisation multiobjectif. Nous avons appliqué ces algorithmes à deux problèmes différents choisis pour leur complexité croissante mais aussi parce qu'ils permettent de vérifier la qualité des résultats obtenus. Il s'agit, successivement, du problème d'identification des paramètres inertiels et de frottements d'un robot, d'une application au problème du contrôle d'un pendule inverse, et du problème de commande d'un robot. Au troisième chapitre, nous proposons d'appliquer les algorithmes génétiques au problème d'identification des paramètres d'inertie et ceux de frottements d'un robot à deux degrés de libertés, ces paramètres sont estimés à partir des mesures effectuées pendant une phase d'excitation. Le modèle dynamique est utilisé pour calculer une trajectoire de compensation pour la commande du robot. Les méthodes d'identification et de compensation sont mises en œuvre sur un robot manipulateur à deux degrés de liberté. D'ailleurs, nous comparons notre algorithme avec l'algorithme NSGA.II et nous terminons par la validation des résultats obtenus par ces deux algorithmes et leurs interprétations.

Nous aborderons au quatrième chapitre le problème de la commande. Nous choisissons, dans un premier temps, la commande du pendule inversé comme problème test. Le contrôle de ce système est très difficile, le système étant non linéaire et instable en boucle ouverte. Nous présenterons des résultats de simulation concernant le pendule inversé sur un chariot, le tout est commandé par la seule vitesse du chariot, l'angle que fait le pendule avec la verticale et la position du chariot sont mesurés; le but est d'établir une loi de commande qui permet de stabiliser le pendule de sorte que le chariot ne sort pas de la piste. Ensuite, nous abordons le problème de la commande d'un robot à deux degrés de liberté. Pour l'amélioration de l'exactitude du robot en terme de poursuite et de régulation, on utilise une loi de commande basée sur le modèle dynamique du manipulateur et d'un régulateur PID, cette méthode qui s'appelle ‘‘la commande par couple calculé’’ où l'inclusion du modèle dynamique dans la boucle de rétroaction permet à la fois la linéarisation ainsi que le découplage. Une analyse de cette loi de commande est effectuée de plusieurs façons : premièrement sans prise en compte des frottements avec et sans contraintes sur les couples, puis par la prise en compte des frottements avec et sans contraintes sur les couples. Les résultats de simulation des cas étudiés sont montrés à la fin du chapitre.

Ce mémoire se termine par une conclusion générale qui fait une synthèse des principaux apports du travail présenté.

Chapitre I

Les approches analytiques d'optimisation multiobjectif

I.1. Introduction

La résolution d'un problème d'optimisation consiste à explorer un espace de recherche afin de maximiser (ou minimiser) une fonction donnée. Les complexités (en taille ou en structure) relatives de l'espace de recherche et de la fonction à optimiser conduisent à utiliser des méthodes de résolutions radicalement différentes. En première approximation, on peut dire qu'une méthode déterministe est adaptée à un espace de recherche petit et complexe et qu'un espace de recherche grand nécessite plutôt une méthode de recherche stochastique (recuit simulé, algorithme génétique,...).

I.2. Vocabulaire et définitions

Problème d'optimisation : est défini par un espace d'état, une ou plusieurs fonctions objectives et un ensemble de contraintes.

Espace d'état : est défini par l'ensemble des domaines de définition des variables du problème.

Dans la plupart des problèmes, cet espace est fini car la méthode de résolution utilisée a besoin de travailler dans un espace restreint (Exemples : la méthode Monte-Carlo, les algorithmes génétiques). Cette limitation n'est pas problématique car lorsqu'un problème est posé, l'utilisateur précise un domaine des valeurs envisageable à chacune des variables. De plus, pour des raisons opératoires et de temps de calcul, il est préférable de travailler sur des domaines finis.

Les variables du problème : peuvent être de nature diverse (réelle, entière, booléenne, etc.) et exprimer des données qualitatives ou quantitatives.

Une fonction objective : représente le but à atteindre pour l'utilisateur (minimisation de coût, de durée, d'erreur, etc.). Elle définit un espace de solutions potentielles au problème.

L'ensemble de contraintes: définit des conditions sur l'espace d'état que les variables doivent satisfaire. Ces contraintes sont souvent des contraintes d'inégalité ou d'égalité et permettent en général de limiter l'espace de recherche.

Une méthode d'optimisation : recherche le point ou un ensemble de points de l'espace des états possibles qui satisfait au mieux un ou plusieurs critères. Le résultat est appelé valeur optimale ou optimum.

I.3. Les méthodes déterministes

Considérons le problème d'optimisation formulé sous la forme :

$$\min_{x \in R^n} f(x) \qquad (I.1)$$

où $f : R^n \rightarrow R$ est une fonction non linéaire.

Définition 1

x^* est un minimum local de f sur le domaine Ω si $\exists\ \varepsilon \geq 0$ tel que :

$$f(x) \geq f(x^*)\ \ \forall\ \left|x - x^*\right| < \varepsilon \qquad (I.2)$$

x^* est un minimum global de f sur le domaine Ω si $\forall\ x \in \Omega$:

$$f(x) \geq f(x^*) \qquad (I.3)$$

Définition 2

On appelle direction admissible en x, une direction d, le long de laquelle on peut se déplacer, tout en restant dans Ω c'est à dire: $\exists \bar{\alpha} > 0$ donc pour tout $\alpha \in \left[0 \quad \bar{\alpha}\right]$ tel que :

$$x + \alpha d \in \Omega \qquad (I.4)$$

Si on désigne par $D(x)$ l'ensemble des directions admissibles en x, alors, s'il existe un $\gamma < \alpha$ tel que pour tout $t \in \left[0 \quad \gamma\right]$ et $d \in D(x)$, on a :

$$f(x + td) \leq f(x) \qquad (I.5)$$

Alors la direction admissible d est dite direction de descente.

I.3.1. Les conditions d'optimalité

I.3.1.1. Condition du premier ordre

La condition nécessaire du premier ordre, pour qu'un point x^* soit un optimum est que, si f est continûment différentiable sur Ω, alors pour toute direction admissible d en x^* on a :

$$\nabla f\left(x^*\right) \cdot d \ \geq 0 \qquad (I.6)$$

En particulier, pour un minimum on a :

$$\nabla f\left(x^*\right) = 0 \qquad (I.7)$$

I.3.1.2. Condition du second ordre

Si f est deux fois dérivable sur Ω et si x^* est un optimum local de f sur Ω alors:

1°/ Pour toute direction d admissible en x^* , on a :

$$\nabla f\left(x^*\right)^T .d \geq 0 \qquad (I.8)$$

2°/ Si

$$\nabla f(x^*)^T .d = 0 \qquad (I.9)$$

Alors :

$$d^T \nabla^2 f\left(x^*\right) \; . \; d \geq 0 \qquad (I.10)$$

où :

$$\nabla^2 f(x) = \begin{bmatrix} \dfrac{\partial^2 f}{\partial x^2} & \dfrac{\partial^2 f}{\partial x_1 x_2} & & \dfrac{\partial^2 f}{\partial x_1 x_n} \\ \vdots & & & \\ \vdots & & & \\ \dfrac{\partial^2 f}{\partial x_n \partial x_1} & & & \dfrac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

Représente la matrice des dérivées secondes ou Hessien.

C'est-à-dire $\nabla^2 f\left(x^*\right)$ est semi définie positive sur l'ensemble des direction admissible.

Du plus si :

$$\nabla f(x^*) = 0 \qquad (I.11)$$

Et : $\nabla^2 f(x^*)$ est défini positive.

x^* est un optimum global.

I.3.2. La méthode du gradient

La méthode la plus classique pour minimiser une fonction non linéaire de plusieurs variables est la méthode de la plus grande pente ou méthode du gradient à pas optimal. Cette méthode est basée sur la constatation élémentaire suivante:

Soit un point x^0 tel que :

$$\nabla f\left(x^0\right) \neq 0 \qquad (I.12)$$

L'approximation au premier ordre, de f au voisinage de x^0, peut être exprimée par la fonction g suivant :

$$g(x) = f\left(x^0\right) \; + \; \nabla f\left(x^0\right)^T \left(x - x^0\right) \qquad (I.13)$$

L'algorithme du gradient à pas optimal peut être résumé aux étapes suivantes :

i) partant d'un point x^0, on commence le processus itératif en calculant :

$$x' = x^0 - \alpha^0 \nabla f\left(x^0\right) \qquad (I.14)$$

avec :

$$\alpha^0 = \arg \min_{\alpha>0} f\left(x^0 - \alpha \nabla f\left(x^0\right)\right) \qquad (I.15)$$

α^0 désigné le pas optimal correspondant.

Le point x^1 obtenu sera le meilleur point possible, dans la direction $-\nabla f\left(x^0\right)$. C'est pour cela que l'on parle de plus "grande pente" ou "gradient à pas optimal".

ii) A l'étape k, connaissant x^k on calcule x^{k+1} par:

$$x^{k+1} = x^k - \alpha^k \nabla f\left(x^k\right) \qquad (I.16)$$

avec

$$\alpha^k = \arg \min_{\alpha>0} f\left(x^k - \alpha \nabla f\left(x^k\right)\right) \qquad (I.17)$$

iii) Les itérations sont répétées jusqu'à ce que l'algorithme vérifie l'un des tests de convergence. Les tests les plus couramment utilisés sont les suivants :

$$\left\|x^{k+1} - x^k\right\| < \varepsilon_1 \qquad (I.18)$$

ou:

$$\left\|\nabla f(x^k)\right\| < \varepsilon_2 \qquad (I.19)$$

ε_1 ε_2 désignent les tolérances de calculs.

I.3.3. La méthode de Newton

La méthode de Newton repose sur le principe de minimisation successive des approximations au second ordre de la fonction f.

La méthode de Newton est la généralisation multidimensionnelle de la méthode Newton-Raphson, appliquée à la recherche des racines de:

$$b(x) = \nabla f(x) \qquad (I.20)$$

Elle fonctionne très bien avec les problèmes pour lesquels le calcul du Hessien est simple et que son inverse existe. Dans les autres cas, elle est inapplicable ou inefficace. La méthode de Newton consiste à faire les itérations suivantes :

$$x^{k+1} = x^k - \alpha.\left(x^k\right)\beta\left(x^k\right)\nabla f\left(x^k\right) \qquad (I.21)$$

$\alpha(x^k)$ et $\beta(x^k)$ représentent respectivement le pas et la direction laquelle on minimise.

Le principal avantage de la méthode de Newton est quelle converge pour tout point choisi dans le voisinage d'un minimum local, avec une vitesse de convergence quadratique. Toutefois, son utilisation reste restreint à cause de :

- La condition sur la matrice des dérivées secondes de la fonction f qui doit être définie positive.
- Le calcul souvent cher en temps, et peu évident du Hessien.

Pour remédier à ces inconvénients, certaines méthodes ont été proposées dans la littérature. Elles allient les avantages de la méthode de Newton et ceux de la méthode de décente. Parmi ces méthodes on citera:

a) La méthode de Levenberg-Marquardt, elle consiste en l'augmentation du Hessien lorsque ce dernier n'est pas défini positif.

b) La recherche linéaire le long de la direction de Newton.

I.3.4.Les Méthodes quasi-Newton

On désigne par méthodes quasi-Newton les méthodes qui utilisent la même forme itérative que l'algorithme de Newton, mais sans calculer explicitement le Hessien de f, ni son inverse.

C'est à dire, au lieu de procéder à l'itération :

$$x^{k+1} = x^k - \left[\nabla^2 f(x^k)\right]^{-1}.\nabla f(x^k) \qquad (I.22)$$

on calcule x^{k+1} par :

$$x^{k+1} = x^k - \alpha^k S^k \nabla f(x^k) \qquad (I.23)$$

où S^k est une approximation symétrique définie positive du Hessien ou de son inverse. Et α^k un paramètre positif fourni par une recherche linéaire le long de la direction $d^k = -S^k \nabla f(x^k)$.

Il est évident que plus S^k sera proche de $\left[\nabla^2 f(x^k)\right]^{-1}$, plus l'algorithme convergera rapidement.

La matrice S^k est remise à jour à chaque itération grâce à une formule simple, additive :

$$S^{k+1} = S^k + C^k \qquad (I.24)$$

C^k est une matrice de correction, qui corrige au mieux la nouvelle information fourni par x^{k+1} et $\nabla f(x^{k+1})$.

En particulier, C^k est choisie de telle manière à ce que S^{k+1} satisfasse l'équation suivante :

$$S^{k+1}\left[\nabla f(x^{k+1}) - \nabla f(x^k)\right] = (x^{k+1} - x^k) \qquad (I.25)$$

Parmi les méthodes utilisant ce principe, on peut citer :

La méthode DFP (Davidon-Fletcher-Powell) et la méthode BFGS (Broyden-Fletcher-Goldfarb-Shannon).

I.3.5. Méthode de Davidon Fletcher Powell

Cette méthode est très performante surtout lorsqu'elle est appliquée à une fonction quadratique. Elle construit non seulement l'inverse du Hessien, mais de plus, elle engendre des directions conjuguées. La mis à jour de S^{k+1} en fonction de S^k se fait en ajoutant deux matrices de rang un.

Algorithmes de DFP :

i) On choisit une matrice S^0 , symétrique définie positive et un point x^0 .

ii) A l'étape k , la recherche linéaire se fait dans la direction

$$d^k = -S^k \nabla f(x^k) \qquad (I.26)$$

et produit α^k, d'où

$$x^{k+1} = x^k + \alpha^k d^k \ , \ \delta^k = x^{k+1} - x^k = \alpha^k d^k \qquad (I.27)$$

et :

$$\gamma^k = \nabla f(x^{k+1}) - \nabla f(x^k) \qquad (I.28)$$

On remet à jour la matrice S^k en utilisant l'expression :

$$S^{k+1} = S^k + \frac{\delta^k \left(\delta^k\right)^T}{\left(\delta^k\right)^T \gamma^k} - \frac{S^k \gamma^k \left(\gamma^k\right)^T S^k}{\left(\gamma^k\right)^T S^k \gamma^k} \qquad (I.29)$$

I.3.6. La méthode de Broyden Fletcher Goldfard Shannon

Cette méthode utilise le même principe que la méthode précédente, la différence que la matrice S^{k+1} approxime le Hessien et non son inverse.

La formule de remis à jour est la suivante [1]:

$$S^{k+1} = S^k + \left(1 + \frac{\left(\gamma^k\right)^T S^k \gamma^k}{\left(\delta^k\right)^T \gamma^k}\right) \frac{\delta^k . \left(\delta^k\right)^T}{\left(\delta^k\right)^T \gamma^k} - \frac{\delta^k \left(\gamma^k\right)^T S^k + S^k \gamma^k \left(\delta^k\right)^T}{\left(\delta^k\right)^T \gamma^k} \qquad (I.30)$$

Cette formulation est moins sensible aux erreurs dans les recherches linéaires, c'est pourquoi la méthode BFGS est aujourd'hui considérée comme l'une des plus performants.

I.3.7.La méthode de Nelder-Mead ou la méthode du simplexe

L'intérêt principal de la méthode du simplexe par rapport aux précédentes est qu'elle ne nécessite pas de calcul de gradient. Elle est uniquement basée sur l'évaluation de la fonction. Cela la rend utilisable pour des fonctions bruitées.

Cette méthode locale effectue une recherche multidirectionnelle dans l'espace d'état [2]. Soit une fonction f à minimiser, on appelle simplexe de R^n tout ensemble $(x_0,\ x_1,\ \ldots, x_n)$ de points de R tel que $f(x_0)\ \geq\ f(x_i)\ \forall\ \ i \in [1..n]$. x_0 est donc le meilleur élément.

Après la construction d'un simplexe initial, l'algorithme va modifier celui-ci en appliquant des calculs de réflexions, expansions et contractions jusqu'à la validation d'un critère d'arrêt.

I.3.8.L'algorithme de séparation - évaluation : branch and bound

Le principe de cette méthode est de découper (branch) l'espace initial de recherche en domaines de plus en plus restreints afin d'isoler l'optimum global (principe de séparation). L'algorithme de recherche forme ainsi un arbre dont chaque nœud représente une partie de l'espace. Ensuite chaque nœud est évalué de façon à déterminer sa borne (bound) inférieure en fonction d'un critère d'évaluation. Si cette borne n'est pas meilleure que la solution courante alors la recherche sur ce nœud est stoppée, sinon la séparation continue.

L'efficacité de cette technique dépend essentiellement du choix des critères de séparation et d'évaluation. Un bon choix permettra à l'algorithme de réduire l'arbre de recherche en évitant la construction de certaines branches. Dans le pire des cas, un mauvais choix peut amener l'algorithme à explorer tout l'espace de recherche. L'utilisation de cette méthode reste limitée à des espaces de petites dimensions.

Cette méthode est essentiellement utilisée sur des problèmes discrets. Couplée avec une méthode d'analyse d'intervalle [3] elle peut être également utilisée dans le cadre de problèmes continus avec l'avantage de ne pas exiger la continuité de la fonction et d'éviter que la propagation d'erreurs numériques empêche la validation d'une solution.

I.4. Les méthodes stochastiques

Les méthodes stochastiques sont caractérisées par :

- un processus de création aléatoire ou pseudo-aléatoire des points dans l'espace d'état,
- une heuristique qui permet de guider la convergence de l'algorithme.

Ces méthodes sont utilisées dans des problèmes où on ne connaît pas d'algorithme de résolution en temps polynomial et pour lesquels on espère trouver une solution approchée de l'optimum global.

Dans la pratique, les méthodes stochastiques qui connaissent le plus de succès sont : la méthode Monte Carlo, le recuit simulé, les algorithmes évolutionnaires et la méthode tabou. Leurs atouts principaux sont les suivants :

- facilité d'implantation,
- flexibilité par rapport aux contraintes des problèmes,
- qualité élevée des solutions.

D'un point de vue théorique, il existe des théorèmes de convergence pour les algorithmes génétiques [4], [5] et pour le recuit simulé [6] qui justifient l'usage de ces méthodes. En général, il est établi que l'on a une probabilité très élevée de trouver une solution optimale, si un temps de calcul très important est alloué.

I.4.1.La méthode Monte Carlo

Les méthodes de type Monte Carlo recherchent l'optimum d'une fonction en générant une suite aléatoire de nombres en fonction d'une loi uniforme [7].

I.4.2.Les algorithmes évolutionnaires

Les algorithmes évolutionnaires sont inspirés des concepts issus du Lamarckisme, Darwinisme et du mutationnisme. Une étude détaillée sur ces algorithmes est présentée en chapitre II.

I.4.3.Le recuit simulé

La méthode du recuit simulé est basée sur une analogie avec le processus physique de recuit des matériaux cristallins. Ce dernier consiste à amener un solide à basse température après l'avoir élevé à forte température. Lorsque le solide est à une forte température, chaque particule possède une très grande énergie et peut effectuer de grands déplacements aléatoires dans la matière. Au fur à mesure que la température est abaissée, chaque particule perd de l'énergie et sa capacité de déplacement se réduit. Les différents états transitoires de refroidissement permettent d'obtenir des matériaux très homogènes et de bonne qualité.

Pour appliquer ce comportement à une méthode d'optimisation, les déplacements aléatoires de chacun des points vont être liés à une probabilité dépendante d'une variable T représentant la température du matériau.

I.4.4.La méthode Tabou

La recherche Tabou [8] est une méthode de recherche locale. A chaque itération, l'algorithme examine le voisinage V d'un point x et retient le meilleur voisin x' tel que $\forall x'' \in V(x)$, $f(x')$ est meilleur que $f(x'')$ et $x' \notin$ à la liste tabou, sauf si $f(x')$ est meilleur que $f(x)$: c'est le phénomène d'aspiration. On note que x' est retenu même si $f(x')$ n'est pas meilleur que $f(x)$.

L'élément fondamental de la méthode tabou est l'utilisation d'une mémoire dynamique dite liste tabou qui permet d'enregistrer les informations pertinentes des étapes de recherche précédentes.

Cette liste empêche le blocage de la recherche sur un optimum local en interdisant à plus ou moins court terme de revenir sur des solutions précédemment visitées de l'espace d'état, solutions dites taboues. La durée de cette interdiction dépend d'un paramètre k appelé teneur tabou. Ce dernier est difficile à déterminer car il est très dépendant de la nature du problème.

Si la valeur est faible la méthode risque de se bloquer sur un optimum local alors qu'une valeur élevée diminuera la capacité de la méthode à exploiter le voisinage de la solution courante. La méthode Tabou est souvent utilisée avec des techniques dites d'intensification et de diversification. L'intensification est fondée sur l'enregistrement des propriétés des situations a priori de bonne qualité afin de les exploiter ultérieurement. La diversification cherche à diriger l'algorithme vers des régions de l'espace de recherche non encore explorées.

I.5. Compromis entre exploration et exploitation

La capacité d'exploitation est l'aptitude d'une méthode à utiliser des résultats déjà obtenus pour faire converger l'algorithme.

La capacité d'exploration est l'aptitude d'une méthode à explorer avec efficacité l'espace d'état. Cette aptitude a tendance à ralentir la convergence.

Lors de la conception d'une méthode d'optimisation, le chercheur doit choisir entre maintien de la diversité et convergence alors que le décideur doit choisir entre efficacité et efficience d'une méthode.

Une méthode efficace fournira un résultat optimal ou proche de l'optimum. Dans ce cas, le temps de calcul n'a aucune importance. Pour assurer un résultat optimal, ces méthodes sont obligées d'effectuer une recherche exploratoire très importante de façon à ne laisser aucune partie de l'espace des états non visitée. Ces méthodes optent pour un maintien de la diversité dans le temps. Or, si l'on fait l'hypothèse d'une population de taille fixe, ce maintien entraîne un ralentissement de la convergence.

Une méthode efficiente est une méthode capable de donner un résultat satisfaisant en un temps de calcul relativement court. Dans ce genre de méthode, le décideur privilégie le temps de calcul à la précision du résultat. La conception de ces méthodes est basée sur une capacité d'exploitation importante des résultats précédents. Ainsi, le processus de convergence est accéléré au risque de voir la méthode trompée et dirigée dans une zone de l'espace non optimale. Cette réutilisation systématique des caractères des générations passées entraîne une perte rapide de diversité.

L'adaptation des individus n'a pas le temps de devenir optimale.

Les méthodes Monte Carlo ont une bonne capacité d'exploration mais une mauvaise capacité d'exploitation. Dans les méthodes de gradient ou multistart, l'exploitation des résultats précédents est efficace mais leur capacité d'exploration est limitée. Les méthodes de recuit simulé et les algorithmes génétiques offrent pour certains types de problèmes un bon compromis entre exploration et exploitation qui dépend du choix judicieux des paramètres de réglages.

Après cette introduction synthétique sur les problèmes d'optimisation, nous allons nous intéresser plus particulièrement aux problèmes d'optimisation multiobjectifs.

I.6. Optimisation avec contraintes

Divers problèmes industriels ne peuvent être résolus en appliquant aveuglement un outil mathématique pour optimiser le modèle. En pratique, beaucoup de contraintes limitent la région de recherche. Elles peuvent correspondre aux limites de validité du modèle ou représenter les limitations

physiques de la technologie. Par conséquent, une analyse numérique doit affronter ce problème et développer des outils plus élaborés.

I.6.1.Types des contraintes

Les contraintes sont des relations qui limitent le domaine de variations des variables. Elles peuvent être linéaires ou non linéaires. Elles sont de deux types:

- Les contraintes d'égalité sont des équations reliant certaines variables entre elles : les équations du modèles sont des contraintes d'égalité.
- Les contraintes d'inégalité définissent des limites aux valeurs de certaines variables ou de certaines fonctions de variables. Elles apparaissent à cause des caractéristiques des équipements (tension où couple maximum de fonctionnement d'un moteur), des normes légales (émissions) ou des spécifications commerciales (pureté d'un produit).

Une contrainte d'inégalité $g_j(x) \leq 0$ est active en x si $g_j(x) = 0$. Elle est inactive si $g_j(x) < 0$

I.6.2. Les algorithmes d'optimalisation avec contraintes

Etant donné la complexité et la grande variété des problèmes non linéaires, trouver le meilleur algorithme d'optimisation est hautement improbable. Des expériences sont nécessaires pour évaluer la valeur des codes disponibles. Plusieurs approches ont été adoptées dans les codes les plus efficaces actuellement et sont présentées brièvement dans les sections suivantes.

I.6.2.1. Méthode du gradient réduit généralisé

Les algorithmes GRG ("generalized reduced gradient") utilise les p contraintes d'égalité pour résoudre p des variables x_b (appartenant à l'ensemble de base) en fonction des n - p variables indépendantes x_{nb} (n'appartenant pas à la base).

Les contraintes peuvent être écrites comme suit et leur jacobien peut être partitionné :

$$h(x_b, x_{nb}) = 0 \qquad (I.31)$$

$$\frac{\partial h}{\partial x} = \left(\frac{\partial h}{\partial x_b}, \frac{\partial h}{\partial x_{nb}} \right) = (B_b, B_{nb}) \qquad (I.32)$$

L'algorithme doit partir d'un point réalisable ("feasible") et l'ensemble de base devrait être tel que B_b soit non singulier dans le but de permettre le calcul des variables de base en fonction de l'ensemble n'appartenant pas à la base.

La fonction objective peut ainsi être exprimée en fonction de x_{nb} :

$$F(x_{nb}) = f(x_b(x_{nb}), x_{nb}) \qquad (I.33)$$

et le problème original est transformé en un problème réduit, plus simple, non contraint :

minimiser $F(x_{nb})$ soumis aux bornes sur x_{nb} .

La fonction transformée est appelée fonction objective réduite ("reduced objective") et son gradient, le gradient réduit ("reduced gradient"). Il est calculé à chaque itération comme suit

$$\frac{\partial F}{\partial x_{nb}} = \frac{\partial f}{\partial x_{nb}} - z^T B_{nb} \qquad (I.34)$$

où le vecteur z est obtenu en résolvant:

$$B_b z^T = \frac{\partial f}{\partial x_b} \qquad (I.35)$$

Les algorithmes GRG procèdent en résolvant une séquence de problèmes réduits. Les variables n'appartenant pas à la base et qui ne sont pas à leur limite inférieure ou supérieure sont appelées "superbasiques". A chaque itération, le gradient réduit avec les variables n'appartenant pas à la base fixées à une de leurs limites permet de vérifier si certaines de ces variables doivent être relaxées et rejoindre l'ensemble des variables "superbasiques".

Les directions de recherche sont généralement obtenues en appliquant la méthode du gradient conjugué ou une méthode à métrique variable, basées sur le gradient des variables "superbasiques" Une minimisation unidimensionnelle est accomplie dans la direction de recherche, et à chaque pas, les contraintes de liaisons sont utilisées pour recalculer les variables de base. Des itérations par la méthode de Newton sont employées dans la plupart des cas pour résoudre l'ensemble des équations. L'expérience montre que cette stratégie peut être implémentée dans des algorithmes qui sont à la fois fiables et efficaces. Puisque la méthode essaye toujours de vérifier les contraintes, tous les résultats intermédiaires sont réalisables et peuvent fournir des informations utiles; cela même si l'optimisation échoue, puisque l'on attend un progrès par rapport un point de départ de la valeur de la fonction objective à chaque itération importante.

I.6.2.2. Programmation linéaire séquentielle

Ces méthodes sont basées sur la linéarisation de la fonction objective non linéaire et des fonctions de contrainte autour du point courant de la base :

$$f(x) \approx f(x^0) + [\nabla f(x^0)]^T \Delta x \qquad (I.36)$$

$$h_j(x) \approx h_j(x^0) + [\nabla h_j(x^0)]^T \Delta x \qquad (I.37)$$

Les limites inférieure et supérieure sont imposées sur Δx pour être certain que les erreurs d'approximation sont limitées. Le problème de programmation linéaire résultant est alors résolu pour des variables Δx et le nouveau point $x^0 + \Delta x$ est testé. Si une amélioration est observée, il est accepté et devient le point de base pour la prochaine itération. Sinon, les limites sur Δx sont rendues plus étroites et le problème de programmation linéaire est résolu à nouveau. Les points intermédiaires ne doivent pas être réalisables : les méthodes SLP ("successive linear programming") peuvent prendre un chemin irréalisable vers l'optimum même si le point initial de base est réalisable. L'avantage des méthodes SLP est la disponibilité de codes efficaces pour les problèmes de programmation linéaire de grande échelle dans les bibliothèques informatiques.

I.6.2.3. Programmation quadratique séquentielle

L'idée sous-jacente est semblable à celle des méthodes SLP, mais les méthodes SQP ("successive quadratic programming") résolvent une série de programmes quadratiques (c'est-à-dire, une fonction objective quadratique avec des contraintes linéaires). La fonction objective quadratique est :

$$f(x) = [\nabla f(x^0)]^T \Delta x + (\Delta x)^T Q \Delta x \qquad (I.38)$$

où Q est l'approximation de la matrice hessienne de la fonction de Lagrange:

$$L = f - \lambda^T g - \mu^T h \qquad (I.39)$$

où λ et μ sont les multiplicateurs de Lagrange .

Une manière de résoudre le problème est d'appliquer la méthode de Newton à l'ensemble des équations exprimant le critère d'existence d'un optimum (conditions de Kuhn-Tucker). La solution de programmation quadratique n'est pas généralement adoptée pour former un nouveau point, mais bien une direction de recherche pour laquelle est accompli une minimisation de la fonction de Lagrange dans la direction Δx .

L'approximation de la matrice hessienne Q est généralement modifiée à chaque itération par l'intermédiaire d'une formule de mise à jour qui préserve sa structure (matrice symétrique, définie positive). Une autre technique pour maintenir une matrice définie positive Q est d'utiliser un lagrangien augmenté comme fonction objective où les termes de pénalité impliquant les carrés des écarts aux contraintes sont ajoutés à l'objectif.

Quand la matrice Q est définie positive, le problème quadratique avec les contraintes linéaires est convexe et l'optimum global peut être démontré. Toutefois, des problèmes peuvent apparaître quand les contraintes (linéarisées) sont telles qu'aucune solution réalisable n'existe pour un sous problème quadratique. D'autres limitations surgissent de la difficulté de manipuler les matrices Q de grande taille. Le développement des codes efficaces de programmation quadratique de grande échelle et des formules efficaces de mise à jour qui conservent les propriétés creuses de l'estimation de la matrice hessienne font aussi partie du développement actuel.

I.7. Problèmes d'optimisation mono objectifs

La résolution d'un problème d'optimisation consiste à trouver une solution qui minimise ou maximise un critère particulier. Dans la plupart des cas, l'optimum découvert n'est pas unique. Ainsi il existe un ensemble de solutions minimisant ou maximisant le critère considéré. Nous pouvons décrire formellement un problème d'optimisation comme suit :

Soit $f : IR^n \rightarrow IR^m$

Et soit X un ensemble fermé de IR^n alors l'ensemble des minimums est :

$\hat{X} = \{x \in X \,|\, \forall x^* \in X, f(x^*) \leq f(x)\}$ et le minimum du problème est : $f(\hat{X})$. $(I.40)$

L'ensemble X représente l'ensemble des solutions potentielles du problème. Cet ensemble est déterminé à l'aide des contraintes (souvent analytiques) données dans l'énoncé du problème. Le formalisme précédent ne faisant pas explicitement intervenir les contraintes du problème, un problème d'optimisation mono objectif est plus souvent donné sous la forme suivante :

Minimiser $f(x)$ (fonction à optimiser). $(I.41)$

Tel que $g_j(x) \leq 0 \quad j = 1,..,K$ (j contrainte à satisfaire).

Avec $x \in R^n, g(x) \in R^j$

Ces deux écritures sont équivalentes, la deuxième étant le plus souvent rencontrée dans les domaines de l'Intelligence Artificielle et de l'Optimisation. Il est clair que pour minimiser (respectivement. maximiser) une fonction d'un problème, il faut déjà être capable de déterminer l'ensemble des solutions potentielles. On distingue ainsi deux notions : la notion d'espace de recherche représentant l'ensemble des valeurs pouvant être prises par les variables, et la notion d'espace réalisable représentant l'ensemble des valeurs des variables satisfaisant les contraintes.

I.8.Problèmes d'optimisation multiobjectifs

La plupart des problèmes d'optimisation réels sont décrits à l'aide de plusieurs objectifs ou critères souvent contradictoires devant être optimisés simultanément. Alors que, pour les problèmes n'incluant qu'un seul objectif, l'optimum cherché est clairement défini, celui-ci reste à formaliser pour les problèmes d'optimisation multiobjectifs. En effet, pour un problème à deux objectifs contradictoires, la solution optimale cherchée est un ensemble de points correspondant aux meilleurs compromis possibles pour résoudre notre problème.

Les problèmes d'optimisation multiobjectifs sont une généralisation à n fonctions objectifs des problèmes d'optimisation classiques. Ils sont définis formellement comme suit :

Considérons un problème de minimisation :

Soit $f : IR^n \rightarrow IR^m$

Et soit X un ensemble fermé de IR^n alors l'ensemble des minimums est :

$$\hat{X} = \left\{ x \in X \middle| \forall x^* \in X, (\exists i \in [1,...,m] f_i(x) < f_i(x^*)) \vee (\forall i \in [1,...,m] f_i(x) = fi(x^*)) \right\} \qquad (I.42)$$

et le minimum du problème est : $f(\hat{X})$.

D'après cette définition, il est clair que l'optimum n'est plus une simple valeur comme pour les problèmes à un seul objectif, mais un ensemble de points, appelé l'ensemble des meilleurs compromis ou le front Pareto. Dans la suite, nous adopterons la formulation suivante, équivalente à la précédente, mais plus souvent utilisée dans les travaux actuels :

$$\begin{cases} \min \quad f_m(x) & m = 1,...,M; \\ g_j \geq 0, & j = 1,...,K; \\ h_j = 0, & j = K+1,...,L \\ x_i^l \leq x_i \leq x_i^u & i = 1,...,n \end{cases} \qquad (I.43)$$

I.8.1. Conditions d'optimalité

Dans cette section nous citons les conditions nécessaire et suffisante d'optimalité au sens de Pareto formulées pour le problème $(\mathrm{I}.43)$. Supposons que toutes les fonctions objectives $f_m, m = 1,...,M$, ainsi que les fonctions de contraintes g_j, $j = 1,..,K$, et h_j, $j = K+1,...,L$, sont continûment différentiables.

I.8.1.1. Condition nécessaire de Fritz-John

Pour que x soit une solution Pareto-optimale du problème $(\mathrm{I}.43)$, il est nécessaire qu'il existe des vecteurs $\lambda \geq 0$ et $\mu \geq 0$ ($\lambda \in R^M, u \in R^K$ et $\lambda, \mu \neq 0$) tels que les conditions suivantes soient vérifiées:

1.

$$\sum_{m=1}^{M} \lambda_m \nabla f_m(x^*) - \sum_{j=1}^{K} \mu_j \nabla g_j(x^*) = 0 \qquad (\mathrm{I}.44)$$

2.

$$\mu_j g_j(x^*) = 0 \text{ pour tout } j = 1,..,K \qquad (I.45)$$

La preuve de ce théorème peut être trouvée, par exemple, dans [9]. En effet, il n'est pas difficile de noter la similarité entre ce théorème et la condition nécessaire de l'optimalité de Kuhn-Tucker pour l'optimisation mono objectif.

I.8.1.2. Condition suffisante de Karush-Kuhn-Tucker

Supposons que les fonctions $f_m, m = 1,...,M$, sont convexes, les fonctions g_j, $j = 1,..,K$, sont non-convexes et les fonctions h_j, $j = K+1,...,L$,sont linéaires (autrement dit, le problème multiobjectif $(\mathrm{I}.43)$ est supposé être convexe. Alors, si la condition nécessaire de Fritz-John est vérifiée, la solution x est une solution Pareto optimale du problème $(I.43)$.

I.8.2. Relations d'ordre et de dominance

Comme la solution optimale est une multitude de points de IR^m, il est vital pour identifier ces meilleurs compromis de définir une relation d'ordre entre ces éléments. Dans le cas des problèmes d'optimisation multiobjectifs, ces relations d'ordre sont appelées relations de dominance. Plusieurs relations de dominance ont été déjà présentées : la a-dominance [10], la dominance au sens de Geoffrion [11], la cône-dominance [12], mais la plus célèbre et la plus utilisée est la dominance au sens de Pareto. C'est cette relation de dominance que nous allons définir de manière à décrire clairement et formellement cette notion, les relations =, $\leq$ et < usuelles sont étendues aux vecteurs.

Définition 1 **:** Soient u et v, deux vecteurs de même dimension,

$$u = v \ ssi \ \forall i \in \{1,2,...,m\}, u_i = v_i \qquad (I.46)$$

$$u \leq v \ ssi \ \forall i \in \{1,2,...,m\}, u_i \leq v_i \qquad (I.47)$$

$$u < v \ ssi \ u \leq v \wedge u \neq v \qquad (I.48)$$

Les relations $\geq$ et > sont définies de manière analogue.

Les relations définies précédemment ne couvrent pas tous les cas possibles. En effet, il est impossible de classer les points a = (1; 2) et b = (2; 1) à l'aide d'une de ces relations.

Contrairement aux problèmes à un seul objectif où les relations usuelles $<, \leq, \ldots$ suffisent pour comparer les points, elles sont insuffisantes pour comparer des points issus des problèmes multiobjectifs. Nous définissons donc maintenant la relation de dominance au sens de Pareto permettant de prendre en compte tous les cas de figures rencontrés lors de la comparaison de deux points (ici des vecteurs).

Définition 2: La dominance au sens de Pareto : Considérons un problème de minimisation. Soient u et v deux vecteurs de décision,

$$u < v \ \ (u \text{ domine } v) \ ssi \ f(u) < f(v) \qquad (I.49)$$

$$u \leq v \ (u \text{ domine faiblement } v) \ \ ssi \ f(u) \leq f(v) \qquad (I.50)$$

$$u \approx v \, (u \text{ est incomparable (ou non-dominé)avec } v) \ ssi \ f(u) \nleq f(v) \wedge f(v) \nleq f(u) \qquad (I.51)$$

Pour un problème de maximisation, ces relations sont définies de manière symétrique.

Définition 3 : Une solution $x \in X_f$ est dite non dominée par rapport à un ensemble $x_a \subseteq x_f$

Si et seulement si : $\exists \ \not\in_a \in X_a, x_a < x$.

Définition 4 : L'optimalité globale au sens de Pareto : Un vecteur de décision x $\in X$ est dit Pareto globalement optimal si et seulement si : $\exists \, y \in X, y \not< x$. Dans ce cas, $f(x) \in F$ est appelé : solution efficace.

Définition 5 : L'optimalité locale au sens de Pareto : Un vecteur de décision $x \in X$ est dit Pareto optimal localement si et seulement si, pour un $\delta > 0$ fixé :

$\exists \, y \in x, f(y) \in B(f(x), \delta) et \ \ y < x, où > B(f(x), \delta)$ représente une boule de centre $f(x)$ et de rayon δ .

I.8.3. Front Pareto et surface de compromis

Nous rappelons que la solution que nous cherchons à un problème d'optimisation multiobjectif n'est pas un point unique mais un ensemble de points que nous avons appelé l'ensemble des meilleurs compromis.

Nous avons défini jusqu'ici les notions de dominance (au sens de Pareto) et d'optimalité. Il nous reste maintenant à définir la solution d'un problème multiobjectif utilisant ces concepts. Cet ensemble des meilleurs compromis, appelé aussi surface de compromis ou front Pareto, est compose des points qui ne sont dominés par aucun autre. Nous définissons d'abord formellement l'ensemble des solutions non dominées :

Définition 1 : Ensemble des solutions non dominées : Soit F l'image dans l'espace des objectifs de l'ensemble réalisable X . L'ensemble des solutions non dominées de X , est défini par l'ensemble $ND(X)$: $ND(X) = \left\{x \in X \mid x \ \ est \ \ non \ do\min é \ par \ rapport \ à \ X\right\}$

Nous pouvons définir le front Pareto de F de manière analogue :

Définition 2 : Front Pareto : Soit F l'image dans l'espace des objectifs de l'ensemble réalisable X . Le front Pareto $ND(F)$ de F est défini comme suit :

$$ND(F) = \left\{y \in F \mid il \ \ n'existe \ \ pas \ z \in F, z < y\right\}$$

Le front Pareto est aussi appelé l'ensemble des solutions efficaces ou la surface de compromis.

Définition 3 : Point Idéal : Les coordonnées du point idéal (p_i) correspondent aux meilleurs valeurs de chaque objectif des points du front Pareto. Les coordonnées de ce point correspondent aussi aux valeurs obtenues en optimisant chaque fonction objective séparément.

$$pi_i = \min\left\{y_i \mid y \in ND(F)\right\}$$

Définition 4 : Point Nadir : Les coordonnées du point Nadir (p_n) correspondent aux pires valeurs de chaque objectif des points du front Pareto.

$$pn_i = \max\left\{y_i \mid y \in ND(F)\right\}$$

Définition 5 : Convexité : Un ensemble A est convexe, si et seulement si l'équivalence suivante est vérifiée :

$$x \in A \wedge y \in A \Leftrightarrow Segment(x, y) \subset A$$

La convexité est le premier indicateur de la difficulté du problème. En effet, certaines méthodes sont incapables de résoudre des problèmes non convexes de manière optimale, mais il existe d'autres indicateurs tout aussi importants, notamment la continuité, la multi modalité, la nature des variables de décision (entières ou réelles).

I.9. Approches de résolution

Un grand nombre d'approches existent pour résoudre les problèmes multiobjectifs. Certaines utilisent des connaissances du problème pour fixer des préférences sur les critères et ainsi contourner l'aspect multicritère du problème. D'autres mettent tous les critères au même niveau d'importance, mais là aussi il existe plusieurs façons de réaliser une telle opération. Plusieurs ouvrages ou articles de synthèse ont été rédiges, des états de l'art plus complets peuvent être consultés notamment dans [11], [12], [13], [14], [15] et [16].

Parmi toutes ces approches, il faut distinguer deux catégories : les approches non Pareto et les approches Pareto. Les approches non Pareto ne traitent pas le problème comme un véritable problème multiobjectif. Elles cherchent à ramener le problème initial à un ou plusieurs problèmes mono objectifs. Par opposition aux méthodes non Pareto, les approches Pareto ne transforment pas les objectifs du problème, ceux-ci sont traités sans aucune distinction pendant la résolution.

Nous allons dans cette section, décrire les principales approches non Pareto et commenter leurs avantages et leurs inconvénients.

I.9.1. Les méthodes d'agrégation des objectifs

La première approche discutée ici est aussi la plus évidente. Elle consiste à transformer un problème multiobjectif en un problème à un seul objectif en agrégeant les différents critères sous la forme d'une somme pondérée :

$$\min \sum_{i=1}^{k} w_i f_i(x) \text{ avec } w_i \geq 0 \qquad (I.52)$$

Les w_i, appelés poids, peuvent être normalises sans perte de généralité : $\sum_{i=1}^{k} w_i = 1$. Il est clair que la résolution d'un problème pour un vecteur de poids w_i fixé ne permet de calculer que quelques solutions Pareto optimales. Pour obtenir un ensemble contenant un grand nombre de solutions Pareto optimales, il faut résoudre plusieurs fois le problème en changeant à chaque fois les valeurs de w_i.

Cette approche à l'avantage évident de pouvoir réutiliser tous les algorithmes classiques dédie aux problèmes d'optimisation à un seul objectif. C'est souvent la première approche adoptée lorsqu'un chercheur se retrouve devant un nouveau problème multiobjectif.

Cependant cette approche a aussi deux inconvénients. Le premier est dû au fait que pour avoir un ensemble de points bien répartis sur le front Pareto, les différents poids w_i doivent être choisis

judicieusement. Il est donc nécessaire d'avoir une bonne connaissance du problème. Le deuxième inconvénient provient du fait que cette méthode ne permet pas, de calculer intégralement la surface de compromis lorsque celle-ci n'est pas convexe. En effet, pour un vecteur de poids $w_i = (w_1, w_2)$ fixé, la valeur optimale atteignable pour la fonction objective créée est p. Les deux points Pareto optimaux trouvés sont A et C. En faisant varier le vecteur des poids w_i, il est possible de trouver d'autres points Pareto optimaux. Seulement, tous ces points se trouveront sur les parties convexes de la surface de compromis. Il n'existe pas de valeur possible pour w_i permettant de trouver par exemple le point B. En effet, cette approche ne permet pas d'approcher la totalité du front Pareto lorsque celui-ci est non convexe.

I.9.2. Goal programming

Cette méthode est également appelée target vector optimisation [17], [18]. L'utilisateur fixe un but T_i à atteindre pour chaque objectif f_i [19]. Ces valeurs sont ensuite ajoutées au problème comme des contraintes supplémentaires. La nouvelle fonction objective est modifiée de façon à minimiser la somme des écarts entre les résultats et les buts à atteindre

$$\sum_{i=1}^{k} \left| f_i(x) - T_i \right| \text{ avec } x \in F \qquad (I.53)$$

où T_i représente la valeur à atteindre pour le $i^{ème}$ objectif.

Différentes variantes et applications de cette technique ont été proposées [18] et [20].

Nous pouvons reprendre l'inconvénient fait pour la méthode d'agrégation des objectifs. La méthode est facile à mettre en œuvre mais la définition des poids et des objectifs à atteindre est une question délicate qui détermine l'efficacité de la méthode.

Cette méthode à l'avantage de fournir un résultat même si un mauvais choix initial conduit l'utilisateur à donner un ou plusieurs buts T_i non réalisables.

I.9.3. L'approche Min-Max

Cette approche consiste à transformer le problème multiobjectif en un problème à un seul objectif ou l'on cherche à minimiser l'écart relatif par rapport à un point de référence appelé but, fixé par la méthode ou le décideur. Il existe plusieurs manières de caractériser la distance entre un point de référence (le but) et un autre, notamment à l'aide des normes.

Une norme est définie de la manière suivante :

$$Lr(f_i(x)) = \left[\sum_{i=1}^{m} \left|B_i - f_i(x)\right|^r\right]^{\frac{1}{r}} \qquad (I.54)$$

Les principales normes utilisées sont : $L_1 = \sum_{i=1}^{m} \left|B_i - f_i(x)\right|$ la distance classique, et la norme $L_\infty = \max_{i\in(1,..,m)}(B_i - f_i(x))$ C'est cette dernière qui est utilisée dans l'approche min-max appelée aussi approche de Tchebychev [14] :

$$\min \max_{i\in(1,..,m)}(B_i - f_i(x)) \quad i = 1,...,M \qquad (I.55)$$

tel que $g_j(x) \leq 0 \;\; j = 1,...,K$

Dans cette approche, le point de référence joue un rôle fondamental. S'il est mal choisi, la recherche peut s'avérer être très laborieuse.

Les méthodes de résolution implémentant cette approche utilisent souvent le point idéal comme point de référence. Ce point idéal évolue donc en fonction de la recherche. En effet, plus la surface de compromis courante trouvée par la méthode se rapproche du front Pareto optimal, plus le point idéal se rapprochera du point idéal du problème.

I.9.4. Le but à atteindre

Cette approche, comme celle de min-max, utilise un point de référence pour guider la recherche. Mais elle introduit aussi une direction de recherche, si bien que le processus de résolution devra suivre cette direction. A la différence de l'approche min-max, qui utilise des normes pour formaliser la distance au point de référence, l'approche du but à atteindre utilise des contraintes, à l'instar de l'approche $\varepsilon-$contrainte, pour déterminer la position du point de référence (aussi appelé le but). L'écart par rapport à ce but est contrôlé grâce à la variable λ introduite à cet effet :

$$\min \lambda \text{ tel que } f_m(x) - \omega_m.\lambda \leq B_m \;\; m = 1,...,M \qquad (I.56)$$

$$\text{et } g_j(x) \leq 0 \;\; j = 1,...,K$$

Ainsi en minimisant λ et en vérifiant toutes les contraintes, la recherche va s'orienter vers le but B et s'arrêter sur le point A faisant partie de la surface de compromis. Cette approche permet, comme l'approche par $\varepsilon-$contrainte et l'approche min-max, de trouver les parties non convexes des fronts Pareto.

Cependant, cette approche, comme les précédentes, doit être itérée plusieurs fois dans le but d'obtenir un ensemble de points Pareto optimaux. Les paramètres $\vec{w}$ et B doivent être bien choisis par

l'utilisateur. Bien que ces paramètres permettent une grande flexibilité de la recherche (orientation et but), s'ils sont mal choisis, ils peuvent, dans certains cas extrêmes, donner des résultats non cohérents.

Il est à noter que cette approche est très similaire à celle de la « goal programming».

I.9.5. L'approche par ε-contrainte

Une autre façon de transformer un problème d'optimisation multiobjectif en un problème simple objectif est de convertir $(m-1)$ des (m) objectifs du problème en contraintes et d'optimiser séparément l'objectif restant. Le problème peut être reformulé de la manière suivante :

$$\min f_i(x) \quad \text{avec } f_j(x) \leq \varepsilon_j, \forall j \neq i \qquad (I.57)$$

L'approche par ε-contrainte doit aussi être appliquée plusieurs fois en faisant varier le vecteur ε_j pour trouver un ensemble de points Pareto optimaux. Cette approche a l'avantage par rapport à la précédente de ne pas être trompée par les problèmes non convexes.

L'inconvénient de cette approche réside dans le fait qu'il faille lancer un grand nombre de fois le processus de résolution. De plus, pour obtenir des points intéressants et bien répartis sur la surface de compromis, le vecteur ε_j doit être choisi judicieusement. Il est clair qu'une bonne connaissance du problème a priori est requise.

I.10.Conclusion

Nous avons présenté dans ce chapitre les principaux concepts de l'optimisation multiobjectif. Les différentes approches de résolution présentées dans ce chapitre sont des approches non Pareto. Elles transforment un problème d'optimisation multiobjectif en un ou plusieurs problèmes à un seul objectif. Que ce soit sous la forme d'une somme pondérée, ou sous la forme d'une distance à un but. Cette transformation permet d'utiliser facilement les méthodes d'optimisation issues de l'optimisation à un objectif, mais ces méthodes ont aussi des inconvénients. Certaines ne peuvent traiter complètement des problèmes non convexes et sont donc très sensibles à la forme du front Pareto (somme pondérée,...), bien que pouvant traiter les problèmes non convexes, restent quand même sensibles à la forme du front Pareto (min max, but à atteindre, . . .).

Un autre inconvénient important est qu'il faille relancer plusieurs fois les algorithmes de résolution avec des valeurs différentes pour certains paramètres (vecteur de poids par exemple) pour obtenir plusieurs points distincts de la surface de compromis. Ces méthodes nécessitent aussi souvent une bonne connaissance du problème a priori, notamment pour fixer les vecteurs de poids ou les points de référence.

Récemment les métaheuristiques, notamment les algorithmes évolutionnaires, ont permis la réalisation des méthodes de résolution dites Pareto très performantes. Elles permettent de déterminer en une seule exécution une approximation de l'intégralité du front Pareto, et ceci même si les problèmes sont non convexes. Le chapitre suivant est dédie à cette classe de méthodes relativement récentes.

Chapitre II

Les algorithmes génétiques multiobjectifs

II.1. Introduction

Les Algorithmes Evolutionnaires (AE) élaborés depuis les années soixante, se classent en trois catégories de base qui sont : les algorithmes génétiques [21], [22], les stratégies d'évolution [23] et la programmation évolutive [24] Dans ce chapitre on s'intéresse à la première classe de ces métaheuristiques. Les Algorithmes Génétiques (AG) sont probablement les algorithmes les plus connus et les plus utilisés dans le calcul évolutionnaire. Ils ont été introduits par Hollande en 1975 [21] cérémonieusement comme un modèle de méthode adaptative. Ils s'appuient sur un codage de l'information et d'un ensemble d'opérateurs génétiques : la sélection, la mutation, le croisement. Un algorithme génétique est typiquement composé de trois éléments fondamentaux :

- une population constituée de plusieurs individus représentant des solutions potentielles (configurations) du problème donné.
- un mécanisme d'évaluation des individus permettant de mesurer l'adaptation de l'individu à son environnement.
- un mécanisme d'évolution de la population permettant, grâce à des opérateurs prédéfini, d'éliminer certains individus et d'en créer de nouveaux.

L'avantage crucial de ces méthodes devant les algorithmes d'optimisation traditionnels consiste en ce que les AE sont capable de connaître les valeurs de la fonction objective non seulement sans faire appel à la dérivée de cette fonction mais même sans exiger son expression analytique.

II.2. Vocabulaire

Cette section est très importante, car elle permet d'établir le rapport entre l'optimisation et tout ce qui est écrit dans les chapitres restants de ce mémoire.

- Individu : un élément de l'espace de recherche.
- Performance : (notée F comme fitness) la mesure de la qualité des individus basée sur l'objectif de l'optimisation et permettant de comparer les individus entre eux afin d'en déterminer les plus et les moins aptes.
- Evaluation d'un individu : le calcul de sa performance.
- Population : un ensemble fini (de taille N) d'individus.
- Evolution : un processus itératif de recherche d'un (ou plusieurs) individu optimal.

- Génération : correspond à l'itération, c'est-à-dire, repère le moment de l'évolution. Mais, parfois, ce terme signifie la population en une certaine itération.
- Croisement : l'opérateur de reproduction appliqué avec la probabilité P_c et correspondant à un brassage d'information entre les individus de la population. Il consiste à échanger des parties composantes (gènes) entre deux individus.
- Mutation : l'opérateur de modification d'un ou plusieurs gènes appliqué avec la probabilité P_m dans le but d'introduire une nouvelle variabilité dans la population.
- Sélection : processus du choix des individus pour la reproduction basé sur leur performance.
- Remplacement : processus de formation d'une nouvelle population à partir des ensembles de parents et d'enfants effectué le plus souvent sur la base de leur performance.

II.3. Historique

Les AG ont été développés dans les années soixante par Holland [25]. L'idée de son système était d'étudier, dans le cadre de la psychologie et la biologie, le processus d'adaptation des populations à la base des données sensorielles introduites au système grâce à des détecteurs binaires [21][25] .

Les AG ont été appliqués à l'optimisation paramétrique pour la première fois par DeJong en 1975 [26], qui a posé les fondements de cette technique d'application. Cependant, le manque de puissance des ordinateurs à l'époque ne permettait pas leur application sur des problèmes réels de grande taille. Ce n'est que pendant les années quatre vingt dix, précisément, avec l'apparition de l'ouvrage de référence écrit par Goldberg [22], que les AG se sont faits connaître dans la communauté scientifique.

II.4. Les opérateurs

II.4.1. Génération aléatoire de la population initiale

Le choix de la population initiale conditionne fortement la rapidité de l'algorithme. Si la position de l'optimum dans l'espace de recherche est totalement inconnue, il est naturel de générer aléatoirement des individus en faisant des tirages uniformes dans chacun des domaines associés aux composantes de l'espace recherche en veillant à ce que les individus produits respectent les contraintes. Si par contre, des informations à priori sur le problème sont disponibles, il parait bien naturel de générer les individus dans un sous domaine particulier afin d'accélérer la convergence.

Dans l'hypothèse où la gestion des contraintes ne peut se faire directement, les contraintes sont généralement incluses dans le critère à optimiser sous forme de pénalités. Il est clair qu'il vaut mieux, lorsque c'est possible ne générer que des éléments de population respectant les contraintes.

II.4.2. Opérateur de codage

Le codage des variable est une étape importante dans un problème d'optimisation par les algorithmes génétiques .A chaque paramètre du dispositif, on doit faire correspondre un gène, sachant qu'un ensemble de gènes représente un chromosome, chaque dispositif est présenté par un individu doté d'un génotype constitué d'un ou de plusieurs chromosomes. La population sera un ensemble de N individus, qui évoluera d'une génération à une autre.

Pour un codage binaire, un gène est représenté par un nombre dont la longueur est exprimée en bits. Différents codes peuvent être utilisés pour le codage : Gray, Binaire, pur.

Un des avantages du codage binaire, est la facilité avec laquelle on peut représenter différent objectifs : les réels, les entiers, les valeurs booléennes, les chaînes de caractères pour passer d'une représentation à une autre, il suffit d'utiliser des fonctions de codage ou de décodage .

Pour mieux expliquer cette procédure, on considère l'espace de recherche fini.

$$x_{i\min} \; < \; x_i \; < \; x_{i\max} \;\; i \in [1, n] \qquad (II.1)$$

Pour coder des variables réelles en binaire et sur l bits, l'espace de recherche est subdivisé en $(2^l\text{-}1)$ valeurs discrètes.

A chaque variable réelle x_i, on associe un entier y_i tel que :

$$y_i = \sum_{i=0}^{l-1} b_i 2^i \qquad (II.2)$$

où chaque b_i est codé sur 1 bit et l représente la longueur du chromosome. La formule de codage est représentée par :

$$y_i = \frac{x_i - x_{i\min}}{x_{i\max} - x_{i\min}} . y_{\max} \qquad (II.3)$$

II.4.3. Opérateur de Croisement

Le croisement a pour but d'enrichir la diversité de la population en manipulant la structure des chromosomes. Classiquement, les croisements sont envisagés avec deux parents et génèrent deux enfants.

Initialement, le croisement associé au codage par chaînes de bits est le croisement à découpage de chromosomes (slicing crossover). Pour effectuer ce type de croisement sur des chromosomes constitués de M gènes, on tire aléatoirement une position dans chacun des parents. On échange ensuite les deux sous chaînes terminales de chacun des deux chromosomes (croisement en un point), ce qui produit deux enfants E_1 et E_2.

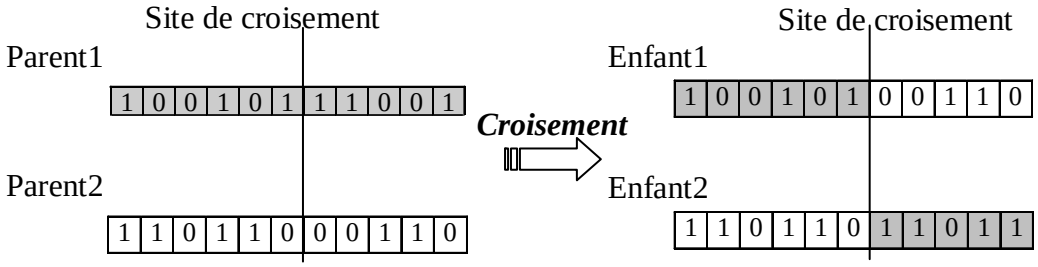

Figure : *II*.1 Croisement en un point

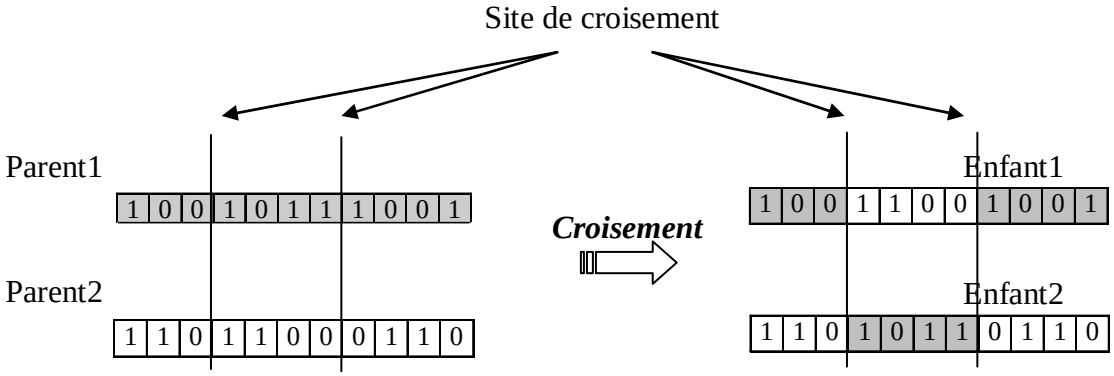

Figure : *II*.2 Croisement en deux points

On peut étendre ce principe en découpant le chromosome non pas en 2 sous chaînes mais en 3 (croisement en deux points).

II.4.4. Opérateur de mutation

La mutation est l'inversion d'un bit dans un chromosome, cela revient à modifier aléatoirement la valeur d'un paramètre du dispositif. Les mutations jouent le rôle de bruit et empêchent l'évolution de cette branche, Elles permettent d'assurer une propriété de recherche aussi bien globale d'espace. Cette propriété indique que l'algorithme génétique sera susceptible d'atteindre tous les points de l'espace de recherche, sans pour autant les parcourir tous dans le processus de résolution. Les propriétés de convergence des algorithmes génétiques sont donc fortement dépendantes de cet opérateur sur le plan théorique.

II.4.5. Opérateur de sélection

La sélection permet d'identifier statistiquement les meilleurs individus d'une population et d'éliminer les mauvais. On trouve dans la littérature un nombre important de principes de sélection plus ou moins adaptés aux problèmes qu'ils traitent. Les plus utilisés sont :

a) La sélection proportionnelle

Cette méthode consiste à dupliquer chaque individu de la population proportionnellement à son adaptation dans son milieu.

$$P_s(x_i) = \frac{F(x_i)}{\sum_{i}^{N} F(x_j)} \qquad (II.4)$$

En utilisant cette probabilité de reproduction, on peut créer une roue de loterie. Où chaque individu de la population occupe une section, proportionnellement à son adaptation, et qui indique aléatoirement quel individu peut se reproduire.

b) La sélection par tournoi

Dans ce type de sélection, K individus de la population sont choisis aléatoirement; celui dont la performance est la plus élevée, est retenu pour participer à la reproduction, l'opération est répétée autant de fois qu'il y a d'individus à sélectionner.

La probabilité pour qu'un individu de rang sélectionner après compétition, est donnée par la relation **:**

$$P_s(x_i) = \begin{cases} \frac{C_{N-1}^{K-1}}{C_N^K} & si : 1 \le i \le N-K-1 \\ \\ 0 & si : N-K-2 \le i \le N \end{cases} \qquad (II.5)$$

Où $C_N^K = \frac{N!}{K!(N-K)!}$ est une combinaison.

Dans le cas particulier, où deux individus sont choisis au hasard, le plus adapté l'emporte avec une probabilité :

$$P_s(x_i) = \frac{2}{N}\frac{N-i}{N-1} \qquad (II.6)$$

Si certains individus gagnent plusieurs fois, ils peuvent participer aux tournois. Ceci favorisera la pérennité de leurs gènes.

Dans la plupart des problèmes la probabilité de la reproduction est une fonction directe de la position de l'individu, ce qui a comme conséquence la sélection d'un « supper individu ». La population tend alors vers son génome, et les différences de niveau d'adaptation entre les individus tendent, à s'estomper. Les meilleurs individus ont alors, sensiblement, la même probabilité de sélection que les autres éléments, l'algorithme génétique ne progresse plus.

Pour éviter ces problèmes, il est possible d'effectuer différents types de transformations sur les valeurs d'adaptation par exemple (fenêtrage, exponentiel, Transformation linéaire).

II.5. Modèle élitiste

Le modèle élitiste consiste à recopier automatiquement le meilleur individu de chaque génération dans la suivante. En effet, le caractère aléatoire de la sélection fait que rien ne garantit que la meilleure solution soit conservée. L'élitisme permet donc de s'assurer que la meilleure solution de chaque génération ne peut que s'améliore avec le temps. Le modèle élitiste n'est pas une méthode de sélection en soi, mais plutôt une variante qu'on ajoute aux méthodes déjà citées.

II.6. Améliorations classiques

Les processus de sélection présentés sont très sensibles aux écarts de fitness et dans certains cas, un très bon individu risque d'être reproduit trop souvent et peut même provoquer l'élimination complète de ses congénères; on obtient alors une population homogène contenant un seul type d'individu.

Pour éviter ce comportement, il existe d'autres modes de sélection (ranking) ainsi que des principes (scaling, sharing) qui empêchent les individus forts d'éliminer complètement les plus faibles. On peut également modifier le processus de sélection en introduisant des tournois entre parents et enfants.

Enfin, on peut également introduire des recherches multiobjectifs, en utilisant la notion de dominance lors de la sélection.

Parmi les transformations les plus connues on citera :

II.6.1. La mise à l'échelle « Scaling »

Le scaling ou mise à l'échelle, modifie les fitness afin de réduire ou d'amplifier artificiellement les écarts entre les individus. Le processus de sélection n'opère plus sur la fitness réelle mais sur son image après scaling. Parmi les fonctions de scaling, on peut envisager le scaling linéaire et le scaling exponentiel [27].

II.6.2. Le partage « Sharing »

Le partage explicite de la performance a été introduit par Goldberg et Richardson en 1987 [28]. L'idée consiste à dégrader la valeur de la performance de l'individu en fonction de la densité de la population dans son voisinage. Le but principal est de pénaliser les individus qui sont trop proches les uns des autres. Ainsi, la performance partagée (shared fitness) F' d'un individu est donnée par:

$$F'(x_i) = \frac{F(x_i)}{nc_i} \qquad (II.7)$$

et

$$nc_i = \sum_{j=1}^{N} Sh(d(x_i, x_j) \qquad (II.8)$$

où Sh désigne la fonction de partage (sharing function) qui dépend de la distance *d* entre deux solutions. Elle retourne '1' si les deux solutions sont identiques, '0' si la distance entre elles dépasse un certain seuil (σ_{share}) et une valeur intermédiaire pour des degrés de dissimilarité intermédiaires. La fonction de partage la plus utilisée est:

$$Sh\ (d)\ = \begin{cases} 1 - (\frac{d}{\sigma_{share}})^{\alpha}, si\ \ d_{ij} \leq \sigma_{share} \\ 0, \qquad \sin on \end{cases} \qquad (II.9)$$

où α est une constante utilisée pour contrôler l'allure de la fonction de partage. Le calcul de la distance entre deux individus peut se faire dans l'espace génotypique (exemple : distance de Hamming) ou phénotypique (exemple : distance Euclidienne), selon le problème traité.

Cependant, des études expérimentales faites par Deb et Goldberg en 1989 [29] sur les mêmes tests ont montré que le partage dans l'espace phénotypique donne des résultats légèrement meilleurs.

La procédure de partage classique a une complexité élevée d'ordre $(O(N^2))$, vu qu'elle nécessite une comparaison deux à deux de tous les individus de la population. Afin de réduire cette complexité, Yin et Germay, en 1993, ont proposé de classifier la population en niches avant la procédure de partage [30]. Cette dernière est appliquée ensuite sur chaque niche, ce qui réduit la complexité à $O(N\ \log(N))$.

Cette méthode a été ensuite rendue adaptative par Alliot [31], qui propose d'actualiser la distance minimale de nichage à chaque génération, de manière à maintenir un bon niveau de performance des individus appartenant aux différentes classes.

II.7. Gestion des contraintes

Considérons le problème d'optimisation multiobjectif, formulé sous la forme :

$$\begin{cases} \min \quad f_m(x) \quad m = 1,...,M; \\ g_j \geq 0, \qquad j = 1,...,K; \\ h_j = 0, \qquad j = K+1,...,L \\ x_i^l \leq x_i \leq x_i^u \qquad i = 1,...,n \end{cases} \qquad (II.10)$$

Dans cette section, nous nous plaçons dans le contexte de la résolution du problème $(\text{II}.10)$ avec $M = 1$ et $L > 0$. Dans le cadre des AGs, de nombreuses approches ont été développées pour traiter ce type de problèmes. Chacune de ces approches appartient à l'une des catégories suivantes: méthodes basées sur la pénalisation, algorithmes de recherche des solutions faisables, techniques préservant la faisabilité des solutions et méthodes hybrides. On présente la première méthode parmi ces quatre groupes d'une façon plus détaillée, car les techniques de pénalisation sont évoquées plus tard lors de la description de la prise en compte des contraintes par les AG multiobjectifs.

II.7.1. Méthodes basées sur le concept de pénalité

Cette méthode est basé sur une idée proche de celle du partage dans le mesure où, là aussi, on se donne une formule selon laquelle la performance de certains individus sera corrigée. Dans le contexte de la prise en compte des contraintes, il s'agit de dégrader la performance des individus infaisables en fonction de leur proximité de la région faisable de l'espace de recherche.

Pour chaque élément de l'espace de recherche, sa proximité de la région faisable peut être mesurée à travers le degré de violation de chaque contrainte:

$$\delta_j(x) = \max \begin{cases} (g_j(x), 0), & j = 1,...,K \\ \left|h_j(x)\right|, & j = K+1,...,L \end{cases} \qquad (II.11)$$

En utilisant cette mesure d'infaisabilité de l'individu x par rapport à chaque contrainte, nous pouvons introduire la fonction de pénalité sous forme générale suivante:

$$F_{penalty}(x) = c(t)\sum_{j=1}^{L} P(\delta_j(x)) \qquad (II.12)$$

où P est un polynôme qui est dans la pratique constitué d'un seul terme, une puissance de $\delta_j(x)$ avec un coefficient. Notons que pour x vérifiant toutes les contraintes, $F_{penaty}(x) = 0$.

Le problème d'optimisation initial $(\text{II}.10)$ avec $M = 1$ est alors reformulé comme un problème d'optimisation sans contraintes:

$$\begin{cases} \min F(x) = f(x) + F_{penalty}(x) \\ x_i^l \le x_i \le x_i^u \end{cases} \qquad i = 1,...,n \qquad (II.13)$$

Les techniques qui appartiennent à cette catégorie diffèrent dans leur façon de définir la fonction de pénalité. Nous citons ici quelques exemples:

PÉNALITÉS STATIQUES [32]:

$$F_{penalty}(x) = \sum_{j=1}^{L} R_{ij}\delta_j^2(x) \qquad (II.14)$$

où R_{ij} sont les coefficients de pénalité définis pour chaque contrainte j. L'indice i représente un intervalle d'une famille d'intervalles de violation définie a priori par l'utilisateur.

PÉNALITÉS DYNAMIQUES [33]:

La pénalité augmentée en cours de l'évolution:

$$F_{penalty}(x) = (C \times t)^{\alpha} \sum_{j=1}^{L} \delta_j^{\beta}(x), \qquad (II.15)$$

où C, α et β sont les constantes à fixer par l'utilisateur.

II.8. Les Algorithmes Génétiques Multiobjectifs

II.8.1. Vector Evaluated Genetic Algorithm (VEGA)

En 1985, Schaffer a proposé le premier Algorithme Evolutionnaire Multiobjectif « AEMO » pour trouver les solutions non dominées d'un problème multiobjectif [34]. Il l'a appelé VEGA - Vector Evaluated Genetic Algorithm - car son algorithme utilise directement le vecteur des valeurs des objectifs pour l'évaluation des individus. Le schéma de l'évaluation est très simple: à chaque génération la population est aléatoirement divisée en autant de sous populations (de tailles égales) qu'il y a d'objectifs, ensuite chaque sous population est évaluée selon l'une des fonctions objectives. L'opérateur de sélection est ensuite restreint à chaque sous population. Cela est particulièrement important dans le cas, où les fonctions objectives sont d'ordres de magnitude très différents.

Pseudo code de l'algorithme « VEGA »

1. Soit le compteur des objectifs $m = 1$ et soit $q = \dfrac{N}{M}$ (M étant le nombre d'objectifs).

N est la taille de la population.

2. Pour toute solution de $j = 1 + (m - 1) * q$ à $j = m * q$, la valeur de performance soit : $F(x^{(j)}) = f_m(x^{(j)})$.
3. Effectuer la sélection par "roulette" pour toutes les q solutions et créer la population des parents P_m.
4. Si $m = M$, passer à l'étape 5. Sinon, incrémenter m de 1 et retourner à l'étape 2.
5. Rassembler toutes les sous populations des parents $P = U_{m=1,..,M} P_m$. Appliquer les opérateurs de croisement et de mutation à l'ensemble P pour créer une nouvelle population.

Il a été observé que cet algorithme a la tendance de trouver les solutions de bonne qualité pour des objectifs individuels. Afin de rendre possible la recherche de bons compromis intermédiaires, l'auteur a proposé de croiser toutes les paires d'individus de la population dans l'idée que le croisement entre deux solutions bonnes chacune pour un des objectifs donnerait un bon compromis entre ces deux objectifs.

II.8.2. Hajela et Lin Genetic Algorithm (HLGA)

En 1993, Hajela et Lin ont proposé la méthode HLGA [35], dont la sélection est basée sur la performance des individus calculée comme la somme pondérée des objectifs. A la différence de la méthode de l'agrégation pondérée traditionnelle, ici, un vecteur des poids différent est associé à chaque individu.

Le point très important de cette approche est la préservation de la diversité entre les vecteurs des poids. Cela peut être fait de deux façons suivantes. Soit le vecteur des poids fait partie de l'individu et la technique de nichage est appliquée, soit des sous populations habilement choisies sont évaluées en utilisant des vecteurs des poids différents prédéfinis.

Approche basée sur le partage

Dans le cadre de cette approche, deux options sont possibles. La première c'est qu'un vecteur de nombres réels de taille M fait partie de chaque individu. Afin de s'assurer que la somme de ses composantes soit égale à 1, chacune d'elles est normalisée par la somme de toutes les composantes:

$$w_i = \frac{w_i}{\sum_{j=1}^{M} w_j} \qquad (II.16)$$

la deuxième option est plus appropriée au cas où seulement peu de solutions de Pareto sont désirées. L'ensemble de vecteurs des poids est choisi à priori et une variable entière x_w qui désigne le numéro

identifiant chaque vecteur s'ajoute à chaque individu. Le nombre de vecteurs des poids K correspond au nombre de solutions qu'on veut trouver.

Une solution est évaluée selon l'équation :

$$F(x^{(i)}) = \sum_{j=1}^{M} w_j^{x_w^{(i)}} \frac{f_j(x^{(i)}) - f_j^{\min}}{f_j^{\max} - f_j^{\min}} \qquad (II.17)$$

Ici, $f_j^{\min}$ et $f_j^{\max}$ sont les valeurs extrêmes de l'objectif j trouvées parmi les solutions de la population courante.

Afin de préserver la diversité des solutions, la technique de partage est utilisée. Le partage ici concerne seulement la partie de l'individu correspondante au vecteur des poids, c'est-à-dire x_w. Cela signifie que seulement les valeurs $x_w^{(i)}$ et $x_w^{(j)}$ sont prises en compte lors du calcul de la distance d_{ij} entre les solutions i et j :

$$d_{ij} = | x_w^{(i)} - x_w^{(j)} | \qquad (II.18)$$

La fonction de partage $Sh(d_{ij})$ est calculée avec le paramètre σ_{share}. Rappelons-nous que la variable x_w est entière et prend des valeurs avec un pas égale à 1. Alors, σ_{share} = 1 signifierait que seulement les solutions avec la même valeur de x_w sont partagées. σ_{share} = 2 permettrait la contribution de deux vecteurs des poids voisins mais cela risque de réduire la diversité nécessaire pour trouver toutes les K solutions Pareto optimales différentes, chacune correspond à une combinaison des poids.

Le "niche count" nc_i de chaque solution i est calculé comme la somme des valeurs de la fonction de partage de toutes les solutions dans la population. Enfin, la performance calculée et divisée par nc_i ce qui donne la valeur de performance corrigée.

Calcul de la performance dans HLGA basé sur le partage

1. Pour toute fonction objectif j, trouver $f_j^{\max}$ et $f_j^{\min}$.
2. Pour toute solution $j = 1,2,...,N$ calculer les distances $d_{ik}, k = 1,2,...,N$. Ensuite calculer la fonction de partage comme suite:

$$Sh\ (d_{ik}) = \begin{cases} 1 - \dfrac{d_{ik}}{\sigma_{share}}, si\ \ d_{ik} \le \sigma_{share} \\ 0, \qquad \sin on \end{cases} \qquad (II.19)$$

Enfin, calculer le "niche count" $nc_i = \sum_{k=1}^{N} Sh(d_{ik})$.

3. Pour toute solution $j = 1,2,\ldots,N$, calculer la valeur de performance F_i et ensuite la corriger par le partage.

Quand les valeurs de F_i' sont calculées pour tous les individus de la population, la sélection proportionnelle est effectuée comme dans un AG ordinaire. Les opérateurs de croisement et de mutation sont appliqués aux individus, la part x_w inclue.

Cette version de HLGA ne demande pas de changements considérables par rapport à un AE mono objectif ordinaire. De plus, sa complexité est relativement réduite.

Les approches multiobjectifs basées sur l'assemblage de critères en une somme pondérée partagent les mêmes défauts: ils sont incapables de trouver les solutions situées sur les régions concaves de la surface de Pareto. De plus, même un ensemble de vecteurs des poids uniformément distribués n'assurent pas forcément l'uniformité de la distribution des solutions correspondantes. Cela rend le choix approprié de ces vecteurs difficile.

Approche basée sur les sous populations

Cette méthode est similaire à VEGA. D'abord, un ensemble de K différents vecteurs des poids $w^{(k)},\ k = 1,\ldots,K,$ est choisi. Ensuite, chacun de ces vecteurs est utilisé pour calculer la performance pondérée normalisée pour tous les N membres de la population. Les $\frac{N}{M}$ meilleurs d'entre eux sont groupés en une sous population (qui est associée au vecteur $w^{(k)}$) pour le processus génétique postérieur. C'est-à-dire, la sélection, le croisement et la mutation sont restreints aux sous populations. La taille de sous population $\frac{N}{M}$ pour chaque vecteur est maintenue, ce qui assure qu'à la fin de ces opérations la taille de la population entière est toujours N. A la génération suivante encore, chaque vecteur est utilisé pour trouver la sous population correspondante des meilleurs.

Il y a donc K sous populations, chacune correspondant à un vecteur de poids. Notons qu'un membre de la population peut être associé à plus d'un vecteur. En particulier, les solutions non extrêmes seront incluses dans plus d'une sous population.

Dans cet algorithme, de multiples solutions sont trouvées grâce à la préservation explicite de K différents vecteurs des poids. Il n'est alors pas nécessaire d'introduire un opérateur de nichage supplémentaire.

Calcul de la Performance dans HLGA basé sur les sous populations

1. Soit le compteur des solutions $k = 1$.
2. Calculer la performance F_j de chaque solution j utilisant le vecteur $w^{(k)}$. Choisir $\frac{N}{K}$ meilleures solutions à partir des valeurs F_j. Copier ces solutions dans sous population P_k.
3. Effectuer la sélection, appliquer le croisement et la mutation aux membres de P_k pour créer une nouvelle population de taille $\frac{N}{K}$.
4. Si $k < K$, incrémenter k de 1 et retourner à l'étape 2.
5. Sinon, combiner toutes les sous populations pour créer une population $P = U_{k=1}^{K} P_k$. Si $| P | < N$, ajouter des solutions aléatoires pour compléter la population.

Notons qu'à l'étape 2, N solutions sont évaluées et cette étape est répétée K fois, ce qui coûte KN évaluations. Cependant, cette approche est meilleure du point de vue de la complexité par rapport à l'approche basée sur le partage grâce à l'absence du mécanisme de nichage avec ses calculs des distances.

II.8.3. Multi Objective Genetic Algorithm (MOGA)

Fonseca et Fleming [36] ont été les premiers à proposer l'algorithme qui utilise la notion de dominance directement pour évaluer la performance des individus.

Dans leur algorithme, pour chaque solution i, le nombre n_i de solutions la dominant est calculé, et le rang $r_i = (1 + n_i)$ lui est associé.

Le rang de chaque solution non dominée est donc égal à 1 et le rang maximal ne peut pas être plus grand que la taille de la population N. La performance F_i attribuée à chaque individu est basée sur son rang. La fonction qui transforme le rang en une valeur de performance est telle que tous les individus du même rang ont la même performance et cette performance est à maximiser.

Bien que l'idée de convertir les valeurs de la "performance vectorielle" en une métrique utilisant le rang de dominance représente un avancement considérable dans l'optimisation multiobjectif, la préservation de la diversité des solutions ne peut pas être assurée à l'aide de cette simple procédure.

Fonseca et Fleming ont donc introduit le nichage entre les solutions du même rang. D'abord, les distances normalisées entre chaque paire de solutions du même rang sont calculées dans l'espace de critères:

$$d_{ij} = \sqrt{\sum_{k=1}^{M} (\frac{f_k^{(i)} - f_k^j}{f_k^{\max} - f_k^{\min}})^2}. \qquad (II.20)$$

où $f_k^{\min}$ et $f_k^{\max}$ sont les valeurs minimale et maximale de la fonction objective k atteintes sur la population courante. La fonction de partage $Sh(\mathrm{d_{ij}})$ est calculée avec $\alpha = 1$ et le "niche count" donné par la formule suivante:

$$nc_i = \sum_{j=1}^{\mu(r_i)} sh(d_{ij}) \qquad (II.21)$$

où $\mu(r_i)$ est le nombre de solutions de rang r_i. Enfin, la valeur de performance F_i attribuée à la base du rang est corrigée comme nous l'avons vu dans l'algorithme HLGA:

$$F_i' = \frac{F_i}{nc_i} \qquad (II.22)$$

Les valeurs de la performance "partagée" F' ne sont plus les mêmes pour les individus du même rang, les solutions situées dans des régions plus éparses ayant une meilleure performance. Cela se traduit par la pression de sélection plus haute pour telles solutions.

Mise à jour de σ_{share}

Dans le contexte de l'optimisation multiobjectif, le paramètre de partage σ_{share} doit être choisi de façon à assurer la distribution uniforme de la population de taille N sur la surface de Pareto. Fonseca et Fleming ont suggéré une procédure simple pour calculer σ_{share} à chaque génération.

Tout d'abord, les valeurs extrêmes $f_j^{\min}$ et $f_j^{\max}$ de chaque objectif j sont calculées à partir des solutions de la population courante. Les images de toutes ces solutions dans l'espace de critères se trouvent dans l'hyper volume :

$$V = \prod_{j=1}^{M} (f_j^{\max} - f_j^{\min}) \qquad (II.23)$$

Par définition, une surface de Pareto ne peut pas croiser une droite parallèle à une des axes en plus d'un point. Supposons que l'on a tracé une telle surface passant par toutes les solutions non

dominées de la population courante. L'aire de cette surface est inférieure à la somme de ses projections le long de chacun des axes.

Enfin, vu que l'aire maximale de chaque projection n'est pas supérieure à l'aire de la face correspondante de l'hyper parallélogramme défini par les points $(f_1^{\min} - f_M^{\min})$ et $(f_1^{\max} - f_M^{\max})$ (voir la figure II.3), l'aire de notre surface est inférieure à la somme de ces faces, c'est-à-dire à la semi aire de l'hyper parallélogramme:

$$A = \sum_{i=1}^{M} \prod_{j=1, j\neq i}^{M} (f_j^{\max} - f_j^{\min}) \qquad (II.24)$$

Supposons que les objectifs sont normalisés, ce qui permet de choisir le même σ_{share} pour tous les objectifs. Alors, le nombre de points destinés à représenter la surface A sans interférer entre eux, peut être calculé comme le nombre d'hyper cubes de volume σ_{share}. D'autre part, nous avons à notre disposition N individus de la population et notre but initial consiste à choisir σ_{share} de façon qu'ils soient uniformément distribués le long la surface de Pareto.

Autrement dit, étant donné un nombre de points N, il est possible d'estimer σ_{share} en résolvant l'équation suivante:

$$N = \frac{\prod_{j=1}^{M}(f_j^{\max} - f_j^{\min} + \sigma_{share}) - \prod (f_j^{\max} - f_j^{\min})}{\sigma_{share}^{M}} \qquad (II.25)$$

pour $\sigma_{share} > 0$, ce qui revient à trouver les racines positives d'un polynôme du degré $M-1$

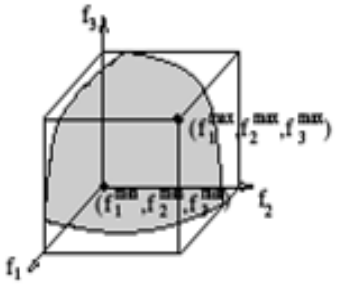

Figure : II.3 Surface de Pareto.

Calcul de la performance dans MOGA

1. Choisir une valeur de σ_{share}.

   Initialiser $\mu(r_i) = 0$ pour tous les rangs possibles $j = 1,2,...,N$. Soit $j = 1$ le compteur des solutions.

2. Pour la solution i, calculer n_i, le nombre de solutions la dominant, puis son rang $r_i = (1 + n_i)$. Incrémenter le compteur de solutions de rang r : $r_i = \mu(r_i) + 1$.

3. Si $i < N$, $i = i + 1$ et retourner à l'étape 2. Sinon, passer à l'étape 4.

4. Identifier le rang maximal r^* c'est-à-dire r_i maximal tel que $\mu(r_i) > 0$. Pour toutes les solutions $i = 1,2,..., N$, calculer leur valeurs de performance comme suite:

$$F_i = N - \sum_{k=1}^{r_i - 1} \mu(k) - 0.5(\mu(r_i) - 1). \qquad (II.26)$$

Soit $r = 1$ le compteur des rangs.

5. Pour toute solution i de rang r, calculer le "niche count" nc_i avec les autres solution du même rang et, ensuite, la performance "partagée" F_i'. Afin de préserver la même performance moyenne, mettre F_i' à l'échelle de façon suivante:

$$F_i' = \frac{F_i \mu(r)}{\sum_{k=1}^{\mu(r)} F'_k} F_i' \qquad (II.27)$$

6. Si $r < r^*$, $r = r + 1$ et retourner à l'étape 5. Sinon, la procédure est terminée.

II.8.4. Nondominated Sorting Genetic Algorithm (NSGA)

L'idée de Goldberg d'utiliser le concept de ranking par dominance dans les AG [22] a été plus directement mise en œuvre par Srinvas et Deb en 1994 dans leur méthode NSGA – Nondominated Sorting Genetic Algorithm [37]. Encore une fois, le double objectif convergence diversité est atteint, d'une part, par l'utilisation d'un schéma du calcul de la performance qui préfère les solutions non dominées et, d'autre part, par l'application de la technique du partage entre les solutions du même front non dominé.

La première étape de NSGA consiste à trier la population P selon le principe de dominance. Cette procédure divise la population en un nombre de classes distinctes P_j de façon suivante:

Tous le individus non dominés de P appartiennent à l'ensemble P_1 ensuite, tous les éléments non dominés de $P \setminus P_1$ sont placés dans l'ensemble P_2 etc. La procédure est terminée quand toute la population est triée:

$$P = U_{j=1}^{r} P_j \qquad (II.28)$$

Notons que entre deux solutions de la même classe, aucune ne peut être considérée meilleure de l'autre compte tenu de tous les objectifs du problème. Le nombre total de classes, noté r dans l'équation ci-dessus, dépend de la population P .

Quand toute la population est triée, l'ensemble P_1 contient toutes les solutions non dominées de P . Ces solutions sont les meilleures au sens de leur proximité de la surface de Pareto du problème. La valeur de performance la plus grande sera donc attribuée à ces individus; la performance va progressivement diminuer en passant d'un front à l'autre. Toute solution i de l'ensemble P_1 reçoit la valeur de performance $F_i = N$.

Afin de préserver la diversité entre les solutions de même qualité (c'est-à-dire, celles qui appartiennent au même ensemble non dominé), la valeur de performance F_i sera dégradée en fonction de nombre de solutions "voisines" de la solution i.

Dans la version originale de NSGA, la procédure du partage est basée sur les distances calculées dans l'espace de décision. C'est-à-dire, la distance entre deux solutions i et j du même front P_l est donnée par l'équation :

$$d_{ij} = \sqrt{\sum_{k=1}^{P_l} \left(\frac{x_k^{(i)} - x_k^j}{x_k^{\max} - x_k^{\min}}\right)^2} \qquad (II.29)$$

Ces distances sont utilisées pour calculer la fonction du partage $Sh(d_{ij})$ avec $\alpha = 2$. Le "niche count" est calculé comme d'habitude et, enfin, la performance initiale F_i est dégradée de façon à augmenter la pression de sélection dans les régions plus éparses :

$$F_i' = \frac{F_i}{nc_i} \qquad (II.30)$$

Calcul de la performance dans NSGA :

1. Choisir le paramètre σ_{share} et un nombre petit positif ε et initialiser $F_{\min} = N + \varepsilon$. Soit le compteur de fronts $j = 1$.

2. Trier la population P selon non dominance:

$$(P_1, P_2, ..., P_r) = Sort(P, \prec)$$

3. Pour tout $q \in P_j$

- Lui associer la performance $F_j^{(q)} = F_{\min} - \varepsilon$.
- Calculer nc_q comme la somme des fonctions de partage des individus de P_j seulement.
- Calculer la performance "partagée" $F_j'^{(q)} = \frac{F_j^{(q)}}{nc_q}$

4. $F_{\min} = \min\ (\ F_j'^{(q)} : q \in P_j)$ et $j = j+1$.

5. Si $j \leq r$, retourner à l'étape 3. Sinon, le processus est terminé.

Cette procédure peut être insérée dans un AG mono objectif. Ainsi, les auteurs de NSGA ont initialement employé la sélection par roulette.

II.8.5. Niched Pareto Genetic Algorithm (NPGA)

En 1994, Horn et al [38] ont proposé leur méthode, elle aussi basé sur le concept de dominance. NPGA – Niched Pareto Genetic Algorithm - diffère des algorithmes décrits ci-dessus au stade de sélection. Dans NPGA, la sélection par tournoi est utilisée au lieu de la sélection proportionnelle. Ce choix est basé sur l'étude théorique [39] consacrée aux opérateurs de sélection utilisés dans l'optimisation évolutionnaire mono objectif.

Un nichage mis à jour dynamiquement [40] est utilisé dans NPGA. Deux solutions i et j sont choisies de la population P de façon aléatoire. Ensuite, toujours aléatoirement, une sous population T de taille $t_{dom} << N$ est choisie. Chacune des solutions i et j est comparée avec toutes les solutions de l'ensemble T au sens de la dominance. Les scénarios suivants sont considérés:

- Si une des solution i ou j domine toutes les solutions de T tant que l'autre est dominée par au moins une solution, alors celle première est choisie.
- Si les deux solutions i et j sont soit dominées par au moins une solution de T, soit dominent tout l'ensemble T, alors elles sont mises toutes les deux dans la population des enfants Q (partiellement remplie) et leur "niche count" dans Q est calculé. La solution, qui a le "niche count" plus petit gagne le tournoi.

Au début, quand la population des enfants est vide, il peut se produire une légère déviation due à la procédure décrite ci-dessus. Quand le second scénario est suivi pour les deux premières solutions, une des deux solutions est choisie aléatoirement en tant que parent potentiel. Ensuite deux parents sont

croisés et mutés pour créer deux enfants. Dès la troisième reprise de ce scénario, la procédure originale est adoptée.

Notons qu'à la différence des méthodes présentées jusqu'à maintenant, NPGA ne suggère pas de calcul de performance scalaire pour chaque individu. La pression de sélection vers les solutions Pareto optimales ainsi que la préservation de la diversité des solutions sont assurées par le principe du tournoi.

Pseudo code de l'algorithme «NPGA »

1. Réordonner aléatoirement la population P. Soit $i = 1$ et $Q = \phi$.
2. Effectuer la sélection par tournoi comme décrit ci-dessus,

   Trouver le premier parent P_1 =NPGA-tournament$(i, i+1, Q)$.
3. $i = i + 2$, et trouver le second parent P_2 =NPGA-tournament$(i, i+1, Q)$.
4. Croiser P_1 et P_2 en créant les enfants E_1 et E_2 et appliquer l'opérateur de mutation à chacun d'eux.
5. Mettre à jour la population des enfants $Q = Q \cup \{E_1, E_2\}$.
6. $i = i + 1$. Si $i < N$, retourner à l'étape 2. Sinon, si $|Q| = \frac{N}{2}$, réordonner aléatoirement P, mettre i égal à 1 et retourner à l'étape 2. Sinon, le processus est terminé.

II.8.6. Nondominated Sorting Genetic Algorithm-II (NSGA-II)

En 2000, Deb et al [41] ont proposé un AGMO élitiste qu'ils ont appelé NSGA-II pour indiquer les origines de la nouvelle approche. En fait, ce n'est que le principe de base du calcul de la performance que NSGA-II a hérité de son prédécesseur NSGA.

Dans NSGA-II, la population des enfants Q_t est d'abord créée à partir de la population des parents P. Ensuite, elles sont réunies en ensemble $R = PUQ_t$, qui est trié selon le principe de dominance. La population suivante est remplie par les solutions des sous-ensembles non dominés de R l'un après l'autre en commençant, évidement, par le premier front.

Vu que la taille de R_t est $2.N$, tous ces sous-ensembles ne peuvent pas rentrer dans la nouvelle population dont la taille doit être égale à N. Pour choisir les solutions qui vont survivre du front dont seulement une partie peut être placée dans la population suivante, une mesure de la densité des solutions dans l'espace de critères est utilisée.

Il est clair qu'au début de l'évolution ces solutions peuvent être choisies aléatoirement sans influencer le processus de l'optimisation. Mais à partir d'un certain moment t, le premier front de R commence à contenir plus de N individus. Dans ce cas la procédure de préservation de la diversité est appliquée à tous les individus non dominés, ce qui à une grande importance pour la bonne représentation finale de l'ensemble de Pareto.

La population initiale de NSGA-II, P_0, est remplie aléatoirement, et ensuite triée selon le principe de dominance. A l'étape de sélection des parents, NSGA-II utilise un tournoi qui est basé tout d'abord sur le rang non dominé, et utilise ensuite une mesure de densité dite crowding distance pour comparer deux solutions du même rang (cette mesure sera détaillée plus loin). La population Q_0 de taille N est obtenue par l'application des opérateurs de croisement et de mutation aux parents sélectionnés lors du tournoi.

Pseudo code de l'algorithme «NSGA-II »

Après initialisation aléatoire de la population initiale P_0, une itération de NSGA-II se déroule comme suit:

1. Créer Q_t à partir de P en utilisant le tournoi et en appliquant des opérateurs de variation génétique au individus gagnants.
2. Réunir les populations des parents et des enfants $R_t = P_t U Q_t$. Trier l'ensemble résultant en sous-ensembles F_i.
3. Soit une nouvelle population $P_{t+1} = \phi$. Soit le compteur des sous-ensembles non dominés $i = 1$.
4. Tant que $| P_{t+1} | + | F_i | < N$, $P_{t+1} = P_{i+1} U F_i$ et $i = i + 1$.
5. Ordonner l'ensemble F_i selon les "distances de surpeuplement" (crowding distances; la procédure de leur calcul est présentée ci-dessous) et inclure N- $| P_{t+1} |$ solutions ayant les valeurs de distance les plus grandes dans la population P_{t+1} .

Il est important de noter que le tri de R_t en sous-ensembles non dominés fait en stade 1 et le remplissage de la population P_{t+1} peut être effectué simultanément. Chaque fois qu'un nouveau front est trouvé, on vérifie s'il peut rentrer dans P_{t+1} entièrement. Si ce n'est pas le cas, le processus du ranking s'arrête.

Calcul des distances de surpeuplement

Pour estimer la densité des solutions voisines d'une solution j dans un ensemble non dominé F, la quantité d_i, appelée ici "la distance de surpeuplement", est calculée de façon suivante.

Calcul des distances de surpeuplement dans un ensemble non dominé F

1. Soit $l = |F|$. Soit d'abord $d_i = 0$ pour toute solution i de F. Soit le compteur d'objectifs $m = l$.
2. Pour l'objectif m, réordonner l'ensemble F de façon que les valeurs de f_m sur ses éléments diminuent. Soit $I^m = sort_{[fm,>]}(F)$ le vecteur des indices, c'est-à-dire I^m dénote l'indice de la solution i dans la liste ordonnée selon l'objectif m.
3. Pour chaque solution i tant que : $2 \le I_i^m \le l-1$, mettre à jour la valeur de d_i comme suite :

$$d_i = d_i + \frac{f_m^{I_i^m+1} - f_m^{I_i^m-1}}{f_m^{\max} - f_m^{\min}}. \qquad (II.31)$$

Et associer les valeurs de distance très grandes aux solutions sur les extrémités de F, c'est-à-dire si $I_i^m = 1$ ou $I_i^m = l$, $d_i = \inf$.

4. Si $m = M$, la procédure est terminée, sinon incrémenter le compteur d'objectifs $m = m + 1$ et retourner à l'étape 2.

La quantité d_i correspond au semi périmètre du cuboïde dont les vertexes sont les voisins les plus proches de i.

Le calcul des distances de surpeuplement exige d'ordonner chacun des fronts considérés M fois, chaque ordonnancement ayant le coût d'ordre $O(N.\log(N))$. Le coût de l'étape 3 est d'ordre $O(N)$. Le coût du calcul des distances de surpeuplement est alors de l'ordre $O(M.N.\log(N))$.

II.9. Optimisation Multiobjectif sous contraintes

Les méthodes décrites dans les sections précédentes de ce chapitre sont destinées à résoudre le problème d'optimisation (II.10) avec $K = L = 0$, c'est-à-dire sans contraintes. Cependant, dans la plupart des applications réelles, l'optimisation est soumise à un certain nombre de contraintes qui sont souvent difficiles à satisfaire. Cette section présente quelques MOGA utilisables dans ce cas.

II.9.1. Pénalisation et MOGA

La prise en compte des contraintes par la majorité des AG mono objectif se base sur l'utilisation d'une fonction de pénalisation. Cette stratégie s'adapte facilement aux MOGA.

En effet, il suffit d'appliquer un des MOGA au problème de l'optimisation multiobjectif suivant :

$$\begin{cases} \min F_m(x), & m = 1,...,M; \\ x_i^l \le x_i \le x_i^u, & i = 1,...,N. \end{cases} \qquad (II.32)$$

Avec

$$F_m(x) = f_m(x) + F_{penalty}^{(m)}(x). \qquad (II.33)$$

Ici, la fonction de pénalité $F_{penalty}$ est définie sous forme générale par l'égalité :

$$F_{penalty}^{(m)}(x) = c_m(t)\sum_{j=1}^{L} P(\delta_j(x)) \qquad (II.34)$$

et $\delta_j(x)$ dénote la mesure du degré de violation de la $j^{ème}$ contrainte. A la différence de l'optimisation mono objectif, il est important ici de mettre à l'échelle les valeurs de δ_j pour tous les $j = 1,...,L$ avant de calculer $F_{penalty}^{(m)}$. Les coefficients $c_m(t)$ sont différents pour différents objectifs afin de rendre $F_{penalty}^{(m)}$ du même ordre de magnitude que f_m.

II.9.2. Tournois indirects

Jiménez et al [42] ont proposé une approche pour l'optimisation multiobjectif sous contraintes qui utilise le tournoi binaire basé sur les notions de faisabilité et de dominance. De plus, une mesure de la densité des solutions sert du critère secondaire pour choisir un gagnant.

Pour deux individus participant au tournoi x_1 et x_2, il y a trois possibilités: soit les deux sont faisables, soit les deux sont infaisables, soit, enfin, un des deux respecte toutes les contraintes et l'autre non. Dans ce dernier cas, l'individu faisable est déclaré gagnant contre l'infaisable. Par contre, quand x_1 et x_2 sont équivalents au sens de faisabilité, un des deux scénarios suivants (similaires de ceux employés dans NPGA) est utilisé:

a) Tous les deux sont faisables

Un certain nombre d'individus faisables est choisi aléatoirement pour jouer le rôle d'ensemble de comparaison. Chacun des x_1 et x_2 est comparé à cet ensemble au sens de dominance. Si un des deux

domine tout l'ensemble et l'autre non, le premier est déclaré gagnant. Si tous les deux sont équivalents de ce point de vue, la technique de partage est utilisée pour déclarer comme gagnant l'individu situé dans une région de moindre densité.

b) Tous les deux sont infaisables

Dans ce cas, l'ensemble de comparaison est choisi parmi les individus infaisables x_1 et x_2 sont comparés cette fois seulement au meilleur représentant de cet ensemble. Le critère de comparaison approprié à la situation peut être, par exemple, le degré de l'infaisabilité des individus en question. Pour décider entre les individus équivalents, le partage rentre en jeu comme dans le scénario précédent.

Notons que dans les deux scénarios, la fonction de partage pour chacun des individus comparés est calculée avec $\alpha = 2$ et elle nécessite le calcul des distances entre ces individus et tous les autres membres de la population.

II.9.3. Mesures d'infaisabilité

Les méthodes que nous allons présenter dans cette section diffèrent par des détails (plus ou moins significatifs) mais se basent sur le même système de préférences, qui permet d'ordonner la population en mettant en tête les meilleurs et finissant par les pires individus. Cet ordonnancement est effectué en utilisant le ranking par dominance qui tient compte non seulement des objectifs mais aussi (et, en fait, tout d'abord) des degrés de violation des contraintes.

Soit $C(x)$ une mesure (scalaire ou vectorielle) de l'infaisabilité d'un individu x . La priorité accordée aux individus respectant toutes les contraintes faites parties du "système de valeurs" adopté par les approches décrites ci-dessous. Ceci dit, la règle de base est que tout individu x tel que $C(x) = 0$ est meilleur que tout individu y pour lequel $C(y) > 0$ (s'il s'agit d'une mesure C vectorielle, il suffit que la dernière inégalité soit vérifiée au moins pour une composante du vecteur *C(y)*).

La partie faisable de la population est classée par le principe de dominance comme dans NSGA ou MOGA, et une technique de préservation de la diversité est appliquée.

Il reste à choisir le critère qui indiquerait le pire des pires, c'est-à-dire, à définir la mesure C .

Rangs d'infaisabilité

Dans la méthode de Fonseca et Fleming, $C(x) = (\delta_1(x), ..., \delta_r(x))$ où $\delta_j(x)$ est le degré de violation de la $j^{ème}$ contrainte défini par l'équation $(\text{II}.10)$. L'opérateur de comparaison $C(x) < C(y)$ est défini par la relation de dominance entre ces vecteurs en vue de minimisation de

toutes leurs composantes. L'égalité $C(x) = C(y)$ correspond au fait qu'aucun des deux vecteurs ne domine l'autre.

Dans ce cas, tous les individus infaisables peuvent être ordonnés selon les principes de comparaison suivants:

- Si $C(x) < C(y)$, x est meilleur que y ;
- Si $C(x) = C(y)$, alors x et y sont comparés au sens de dominance dans l'espace des objectifs du problème, f_m pour $m = 1,\ldots,M$
- Si $C(x) = C(y)$ et x et y sont équivalents aussi au sens de dominance dans l'espace des objectifs, alors la technique de partage entre les individus du même front non dominé dans l'espace des objectifs est appliquée.

II.10.Conclusion

Dans ce chapitre, nous avons vu les fondements théoriques des méthodes récentes d'optimisation multiobjectif ainsi que les modifications qui peuvent être apportées à un algorithme génétique multiobjectif afin de l'adapter à la gestion des contraintes. L'intérêt de ces modifications est de rendre possible l'obtention d'un tel ensemble en un seul essai de l'algorithme, et sans devoir définir des paramètres supplémentaires comme les poids relatifs des objectifs, les vecteurs ε et des autres vus dans le chapitre I.

L'intérêt pour ces méthodes récentes s'explique notamment par leur capacité de trouver une bonne approximation de l'ensemble des compromis de Pareto en un seul essai de l'algorithme, à la différence des approches traditionnelles pour l'optimisation multiobjectif, qui ne trouvent qu'une solution compromise.

Chapitre III

Application des algorithmes génétiques multiobjectifs à l'identification

III.1. Introduction

L'identification des systèmes physiques complexes, qui sont principalement non linéaires, implique faire face aux problèmes comme le bruit, l'identifiabilité de paramètres, les signaux d'entrée appropriés, et les paramètres couplés. Les méthodes principales d'identification font face à ces situations en utilisant des fonctions de transfert ou la représentation d'état directe, comme les méthodes de maximum de vraisemblance, de sous-espace ou de fréquence [44]. L'approche de maximum de vraisemblance maximise la probabilité des événements observés, d'après une fonction de densité de probabilité. La méthode de sous-espace emploie la base de la représentation d'état, et l'estimation fréquentielle essaye de s'adapter aux données directement en fréquence.

Mais ce n'est pas la seule voie de représenter un système : une autre manière de faire de l'identification est l'approche du modèle inverse développée en [45], utilisée principalement en robotique, où le robot est modélisé comme une chaîne articulée suivant un schéma bien prouvé.

On présente dans ce chapitre les caractéristiques de la méthode du modèle inverse appliqué à l'identification des paramètres d'un robot, en utilisant les algorithmes génétiques. C'est un exemple représentatif d'un système complexe parce que la dynamique du robot est très riche et n'est pas facilement modélisée.

III.2. Les méthodes générales d'identification

Identifier un processus (système), c'est chercher un modèle mathématique, appartenant à une classe de modèles connue, et qui, soumis à des signaux tests en entrée, donne une réponse dynamique ou statique en sortie, la plus proche possible du système réel [46].

La notion de modèle mathématique d'un système, d'un processus ou d'un phénomène, est un concept fondamental. Il existe une multitude de modèles, chacun étant destiné à une application particulière. Nous pouvons les décliner en deux grandes catégories :

- Les modèles de connaissance (basés sur les lois de la physique, de la chimie…), donnent une description complète des systèmes et sont utilisés pour la simulation et la conception des procédés. Ce sont souvent des modèles complexes.
- Les modèles dynamiques de commande, qui donnent la relation entre les variations des entrées d'un système et les variations de la sortie, sont utilisés en automatique.

Les modèles dynamiques sont de deux sortes :

- Modèles non paramétriques (réponse fréquentielle, réponse à un échelon)
- Modèles paramétriques (fonction de transfert, équations différentielles)

L'identification est une approche expérimentale pour la détermination du modèle dynamique d'un système. Cette approche peut être décomposée en quatre étapes (Figure III.1):

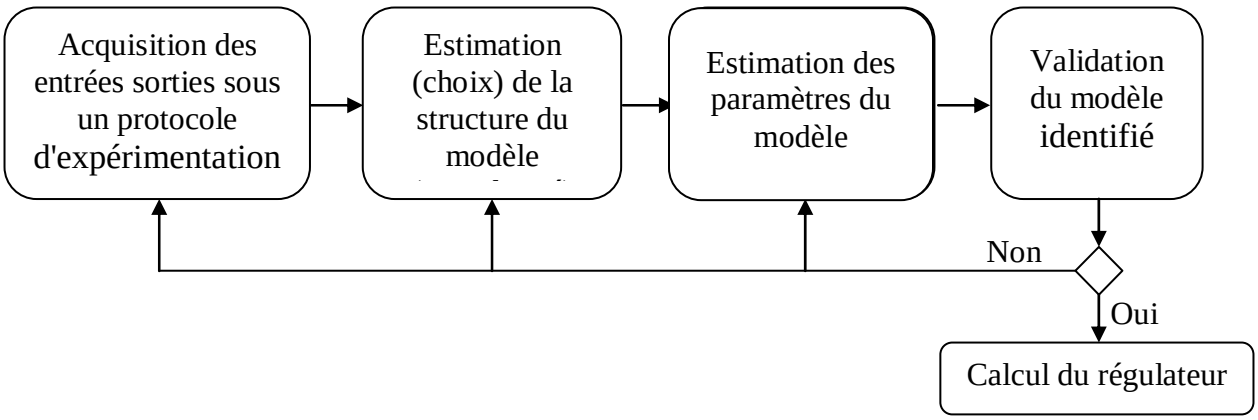

Figure (III.1): procédure d'identification d'un modèle

III.3.Vue générale sur la modélisation et l'identification des robots

III.3.1. Modélisation cinématique et dynamique des robots

La littérature de robotique offre de diverses théories sur la modélisation cinématique et dynamique [47]-[48], [49], [50]. Un certain nombre de méthodes de modélisation est disponible, répondant à de diverses exigences. Quand il s'agit d'une cinématique de robot, le modèle est calculé entre l'espace opérationnel (en général un espace de six dimension des coordonnées de l'organe terminale) et l'espace articulaire (dimension égale au nombre de degrés de liberté du robot). Le modèle cinématique d'un robot décrit les vitesses des coordonnées opérationnelles en fonction des vitesses articulaires alors que le modèle cinématique inverse permet de calculer, à partir d'une configuration donnée, les vitesses articulaires qui assurent au repère terminale une vitesse opérationnelle imposée. Ces modèles peuvent être représenté sous forme récursive ou modèle en boucle fermée. La représentation en boucle fermée facilite la manipulation des modèles et permet une analyse mathématique directe. Comme une grande précision de calcul peut être atteinte plus rapide avec les modèles en boucle fermée, ils sont préférables pour la commande en temps réel. Les modèles dynamiques relient couples/forces appliqués aux actionneurs et les positions, vitesses et accélérations articulaires. Une variété de méthodes est disponible pour leur calcul, et comme les modèles cinématiques, elles peuvent être représentées dans diverses formes. Les modèles algébriques récursifs exigent moins de calcul et admettent une représentation très compacte. Ceux-ci sont les arguments

traditionnels qui démontre pourquoi les méthode algébriques récursives sont préférable pour les applications en temps réel [48]. Les calculateurs numériques modernes rapides rendent tels arguments moins appropriés. D'autre part, une représentation algébrique en boucle fermée d'un modèle dynamique permet une analyse explicite de chaque effet, tel que l'effet de l'inertie, de centrifuge et de Coriolis, de gravité et de frottement, et d'une évaluation directe de leur influence sur le comportement dynamique. Une analyse de la structure du modèle mène à une commande directe, ainsi qu'elle facilite la compensation pour chaque effet particulier. Par exemple, un modèle en boucle fermée comprend un terme de gravité peut être employé pour la compensation des charges gravitationnelles. L'apparition des calculateurs numériques modernes de grande puissance de calcul et des capacités de stockage en mémoire plus élevé permet la mise en œuvre en ligne des modèles dynamiques en boucle fermée employée pour la commande. Malheureusement, le calcul des modèles en boule fermée sous une forme compacte, en particulier les modèles cinématique et dynamique inverses n'est pas habituellement facile, même si un logiciel pour le calcul symbolique est employé. Le calcul exige une série d'opérations accompagnée d'une simplification des résultats intermédiaires. Par conséquent, il est important de mentionner n'importe quelle contribution dans ce domaine. Quand un modèle est tiré, il est utile d'établir son exactitude en le comparant directement à une certaine représentation récursive de la même cinématique ou dynamique avec les logiciels disponibles. Les logiciels spécialisés pour les problèmes robotiques sont présentés dans [51] et [52].

III.3.2. Estimation des paramètres du modèle

Quand un modèle est disponible, la prochaine étape est d'obtenir ses paramètres, de sorte que le modèle correspond au système réel. Les paramètres cinématiques, tels que les longueurs de corps, la position et l'orientation de chaque corps sont habituellement connus avec plus de précision que ceux d'inertie (masses des corps, position de leurs centres de la masse, et le moment d'inertie de chaque corps).

Les paramètres cinématiques sont directement fournis par le fabricant ou peuvent être obtenus par des mesures directes sur le robot. L'ajustement de ces paramètres peut se faire par un étalonnage cinématique [49]. Puisque les paramètres d'inerties sont rarement fournis par le fabricant, ils sont plutôt reconstruits par les utilisateurs du robot. La modélisation et l'estimation de frottement sont également les questions importantes, car le frottement peut poser des problèmes de commande, par exemple, erreurs statiques, qui peut conduire à des cycles limites.

Il existe une vaste littérature sur l'estimation des paramètres d'inertie et de frottement [49], [50], [53]-[54]. Nous présentons ici un aperçu de certains résultats. Quand on estime des paramètres

inertiels, pour chaque corps de robot, on devrait déterminer la masse, trois coordonnées cartésiennes du centre de la masse et six éléments du tenseur d'inertie [48], [49]. Dans le cas général, pour un robot avec n degré de liberté, ils existent $10 * n$ inconnus. Dans la pratique, au lieu d'estimer chaque paramètre, il est plus courant d'identifier les combinaisons algébriques qui constituent ce qu'on appelle les paramètres de base (base parameters set 'BPS'). Les éléments de cet ensemble peuvent être uniquement identifiés dans la pratique. Il existe une représentation du modèle dynamique linéaire par rapport aux éléments de (BPS) [47], [49], [50], [53]-[54], [55]. La matrice de régression dépend de la trajectoire de robot, c-à-d, les positions, vitesses, et accélérations, prémultiplie des BPS sous la forme de vecteur pour ajuster les entrées de commande. L'énergie total est également linéaire en BPS [50], [53], [56].La propriété de linéarité facilite l'estimation des BPS.

III.3.2.1. Estimation des paramètres d'inertie

Une fois une stratégie pour la modélisation est effectuée, on peut procéder à l'estimation des BPS. À cet effet, une expérimentation d'identification doit être conçue. En pratique, le robot effectue des mouvements spécifiques, en utilisant une approche décentralisée de commande. Cela signifie l'utilisation des contrôleurs (PID) découplés pour produire des entrées de commande. Les données requises pour l'estimation des paramètres sont les entrées de commande et le mouvement (trajectoire désirée), c-à-d, les postions, vitesses et, si nécessaire, des accélérations. Ces données sont obtenues par des mesures directes et/ou sont reconstruit en utilisant des filtres ou des observateurs.

La fiabilité de l'estimation dépend de la trajectoire exécutée par le robot lors de l'identification expérimentale. La trajectoire d'excitation devrait être soigneusement conçue, et ceci est fait en optimisant une certaine propriété des matrices [50], [53], [55], [56], [57], [58].Cette matrice est formée en empilant verticalement les matrices de régression qui correspondent à différents points de la trajectoire d'excitation. Le conditionnement de la matrice de régression, sa plus grande valeur singulière, et la détermination des coefficients de pondération entre la matrice de régression et son transposé sont certains choix pour les critères de performance à être optimisé. Quel que soit le critère de performance adopté, le problème d'optimisation résultant est très probablement non convexe, ce qui signifie que la trajectoire d'excitation optimale ne peut pas être obtenus facilement. Différents conditions initiales peuvent conduire à différente trajectoire d'excitation, tout correspondant aux minimum locaux du critère d'optimisation. Ici, nous voudrions examiner ce que les chercheurs considèrent comme "un bon trajectoire d'excitation", et qu'elle exigence, à notre point de vue, devrait satisfaire.

Le calcul de la trajectoire d'excitation en utilisant le calcul des variations est proposé dans [53].

Le problème d'optimisation était de minimiser le nombre de condition de la matrice d'excitation persistante, créée en utilisant la matrice de régression. L'objectif était d'augmenter l'immunité d'estimation en présence de mesure du bruit. L'algorithme d'optimisation génère une trajectoire d'excitation, point par point. Une telle représentation de trajectoire désignée sous le nom " générale", qui signifie que l'espace de la trajectoire à rechercher n'est pas restreint. Ceci, généralement peut contribuer à un processus inefficace d'optimisation. Le même problème est inhérent à la méthode de génération de la trajectoire d'excitation présentée dans [58]. Une autre fois, la trajectoire est calculée point par point, circonscrire l'espace de recherche uniquement en imposant des contraintes sur les gammes admissibles des positions, vitesses et accélérations de robot.

En outre, les deux approches ne spécifient pas explicitement la fréquence contenue dans les trajectoires calculées. C'est un inconvénient supplémentaire, car le choix des fréquences contenues dans une trajectoire est un principe de précaution important pour éviter des composants de fréquence qui peuvent exciter les flexibilités de robot. La paramétrisation intentionnelle d'une trajectoire d'excitation est une manière d'imposer le contenu désiré de fréquence, et de réduire l'espace de recherche afin d'accélérer le processus d'optimisation. Par exemple, dans [55] et [59], les trajectoires d'excitation sont postulées en tant que série de Fourier finie avec une fréquence fondamental prédéfini et avec des coefficients à déterminer en optimisant un certain critère d'exécution. La trajectoire en résultant est périodique, ce qui permet la répétition continue de l'expérimentation d'identification et de l'établissement d'une moyenne des données mesurées pour améliorer le rapport signal/bruit. Les mérites additionnels sont que, par conception, on peut imposer les fréquences contenues dans la trajectoire et leurs résolutions.

Les trajectoires d'excitation sont réalisées sur les systèmes robotiques utilisant les contrôleurs PID en boucle fermée. Les entrées de commande résultantes et les coordonnées de mouvement sont mesurées et enregistrées pour l'estimation des éléments de BPS. Plusieurs algorithmes d'estimation, des méthodes des moindres carrés aux celles de maximum de vraisemblance, sont étudiés jusqu'à maintenant.

Le succès des algorithmes sophistiqués dépend fortement d'une connaissance précise des propriétés statistiques du signal de bruit et d'autres perturbations qui affectent le signal utile. Quelques algorithmes ont été expérimentalement comparés dans [54]. L'algorithme d'estimation le plus impliquée basé sur la méthode de maximum de vraisemblance a montré effectivement les meilleures performances, mais les améliorations atteintes au cours de la plus simple algorithme ne semblent pas tellement étonnantes. Une alternative pour des techniques avancées d'estimation est d'améliorer la

qualité des signaux mesurés sur le système robotique, en réduisant l'effet des perturbations [60] - [61]. Une étude comparative de plusieurs techniques numériques pour l'estimation des vitesses à partir des mesures de position est donnée dans [60]. L'efficacité de ces techniques pour l'estimation de l'accélération n'a pas été discutée, de sorte que leur utilité dans l'obtention des données de qualité suffisante pour l'estimation de BPS ne peut être immédiatement prévu. D'autre part, on peut utiliser un filtre de Kalman [62], tel qu'il est formulé dans [61] -[63], pour reconstruire de façon optimale toutes les coordonnées de mouvement comprenant l'accélération, en présence du bruit de mesure et de l'incertitude dans la dynamique du système. Si les coordonnées reconstruites de mouvement sont employées, alors même l'estimation des moindres carrés la plus simple des éléments de BPS donne des résultats satisfaisants.

III.3.2.2. Modélisation et Estimation des paramètres de frottement

Le frottement est une interaction complexe entre les surfaces de contact des objets agissants les uns sur les autre, ce qui peut être représentée par un certain nombre de modèles [64] - [65]. Ces modèles sont utilisés pour l'analyse théorique des phénomènes de frottement et pour la commande. Dans ce chapitre, on met l'accent sur la dernière, c-à-d., compensation de frottement de robot on basant sur son modèle. Deux régimes de frottement sont observés: avant glissant « presliding » et glissant «sliding » [64], [66]. L'avant glissant se produit si les objets en interaction ne sont pas en mouvements les uns par rapport aux autres. Alors que le régime glissant surgit après que le mouvement relatif soit établi. Le niveau du frottement avant le régime glissant s'appelle le frottement statique. Il est déterminé en équilibrant les forces statiques agissant sur les objets en interaction. Un modèle de frottement sophistiqué, par exemple, le modèle de LuGre [53], [67], [68], [69], caractérise le régime avant glissant par sa rigidité et l'amortissement de l'interaction dynamique entre les surfaces de contact. Au début du régime glissant, la force de frottement décroît dans un premier temps lorsque la vitesse de déplacement augmente jusqu'à atteindre un minimum pour ensuite croître en fonction de la vitesse, ceci est connue comme effet de Stribeck.

Le modèle de frottement de base, qui couvre juste les effets de coulomb et visqueux, admet une paramétrisation linéaire, qui permet l'estimation simultané de ses paramètres avec les éléments de BPS [50], [55], [56], [57].Contrairement au modèle de base, des modèles de frottement plus compliqué ne sont pas linéaire dans certains paramètres qui devraient être déterminés séparément aux paramètres de BPS. La fonction de trois sigmoïde considéré dans [70] est un exemple d'un tel modèle de frottement. Si un concepteur de commande adopte un modèle qui n'est pas linéaire dans tous les paramètres, alors on a les options suivantes :

- Identification des paramètres de frottement séparément des BPS et utiliser des estimations pour la compensation de frottement [64], [65], [66], [69], [70], [71] ;
- Estimation des paramètres de frottement en ligne simultanément en employant un modèle de compensation de frottement [67], [72], [73] ;
- Renonciation des modèles de frottement structuré et adoption d'une stratégie de compensation de certains effets de frottement non modelé comprenant la détection directe du frottement utilisant les sondes de force [73]. La première option est employée souvent dans la commande de robot.

III.3.2.3. Validation expérimental du modèle

Après l'estimation est terminée, une validation expérimentale du modèle doit être faite. Une stratégie de validation rigoureuse de deux étapes a été établie dans [49]. L'objectif de la première étape consiste à vérifier l'exactitude du modèle qui produit les sorties quand il est associé avec des contrôleurs PID. Si cette exactitude est satisfaisante, on procède à la deuxième étape de validation : la vérification de l'utilité du modèle pour la commande.

Dans un certain nombre de publications sur la modélisation et l'identification des robots, juste la première étape de la stratégie de validation est prise en compte. Par exemple, dans [59], la validation est effectuée le long d'une trajectoire choisie aléatoirement dans l'espace de travail de robot. La trajectoire doit satisfaire les contraintes sur les positions, vitesses et accélérations. Il est prévu que l'accord qualitatif des couples rendus par le modèle et les couples mesurés sous la commande PID, le long de cette trajectoire aléatoire, soit garantie suffisamment l'exactitude du modèle.

Une validation qualitative semblable peut être trouvée dans [54] et [55]. En [55], la validation est effectuée avec une trajectoire lisse définie dans l'espace articulaire. Le choix d'une sinusoïde comme trajectoire de validation, comme dans [54], est probablement la validation la plus simple qu'on pourrait envisager, car elle teste le modèle pour une seule composante harmonique. Les résultats satisfaisants pour un seul harmonique n'impliquent pas nécessairement la même chose pour d'autres composants de fréquence.

Dans [49], la validation expérimentale a été effectuée à l'aide d'une trajectoire polynomiale de cinquième ordre, définie dans l'espace articulaire. La trajectoire a été choisie pour assurer des mouvements articulaires rapides, nécessitant la pleine autorité de toutes les commandes. Tout d'abord, le modèle dynamique considéré a été validé qualitativement, c'est-à-dire, en comparant ses couples de sorties avec des couples mesurés sous la commande (PD). Ensuite, le modèle a été mis en place dans

deux algorithmes de commande 'feedforward' et de couple calculé pour générer les entrées de commande en plus des régulateurs (PD). Le dernier test a vérifié que la commande fondée sur le modèle dynamique présente des petites erreurs de poursuite, si on les compare avec les erreurs obtenues avec une simple commande PD.

III.3.3. Dynamique non couverte par le modèle

Dans la pratique, le modèle, en particulier des corps rigide, peut couvrir la vraie dynamique seulement dans les base fréquences , comme la vraie dynamique est entouré par les effets des flexibilités en hautes fréquences qui sont causés par l'élasticité dans les articulations et par la distribution des liaisons flexibles [74]. Si l'on exige une haute performance, l'influence des flexibilités ne peuvent être ignorés. A cet effet, des études théorique et expérimentale présentées dans [75], [76] montrent que la connaissance de la dynamique non couverts par un modèle dynamique est essentiel pour améliorer la performance et la robustesse.

Dans le présent chapitre, nous considérons juste l'objectif conventionnel d'identification de la dynamique des corps rigides et celle des paramètres de frottements.

III.4. Identification des paramètres inertiels d'un robot

III.4.1.Méthodes d'estimation des paramètres inertiels

Plusieurs techniques peuvent être envisagées pour le calcul des paramètres inertiels d'un robot:

i) les mesures:on procède à des essais expérimentaux sur chacun des corps du robot pris isolément [77] :

- les masses des corps sont obtenues par simple pesée;
- les positions des centres de masse sont repérées par les points d'équilibre des corps;
- les moments d'inertie sont obtenus par la méthode du pendule.

Les produits d'inertie ne peuvent cependant pas être déterminés expérimentalement. En outre, la difficulté majeure provient de la nécessité de démonter le robot pour effectuer les mesures. La méthode n'est envisageable que dans une phase précédant son montage.

ii) le calcul: à partir des considérations géométriques sur les corps constitutifs du robot et de l'hypothèse d'une répartition uniforme des masses, on peut obtenir une évaluation des paramètres inertiels du mécanisme. Le résultat est immédiat lorsque les différents corps sont représentés dans la base de données géométrique d'un système de CAO, ce dernier possédant généralement des fonctionnalités de calcul des paramètres inertiels.

iii) l'identification: les robots industriels étant difficilement démontables, leurs charges pouvant varier ou être inconnues, la meilleure solution est encore de recourir à des techniques d'identification.

III.4.2. Principe de l'identification des paramètres dynamiques

L'identification des paramètres dynamiques des robots manipulateurs a fait l'objet de nombreuses recherches [77]. Les méthodes proposées présentent de nombreux points communs, notamment:

- l'utilisation d'un modèle de connaissance linéaire vis-à-vis des paramètres dynamiques inconnus (modèle dynamique, énergétique, modèle de puissance ou modélisation des efforts exercés sur la base) ;
- la construction d'un système linéaire surdéterminé par échantillonnage du modèle à différents instants au cours d'un mouvement du robot;
- l'estimation des paramètres par des techniques de régression linéaire (moindres carrés ordinaires ou autres variantes).

III.4.2.1. Méthode de résolution

Le système à résoudre est donné par la relation générale suivant :

$$Y(\Gamma, \dot{q}) = W(q, \dot{q}, \ddot{q})\chi + \rho \qquad (III.1)$$

où W est la matrice d'observation de dimension $(r \times c)$ avec $r >> c$, c est le nombre de paramètres et ρ est le vecteur des résidus ou vecteur des erreurs.

La théorie de l'estimation offre une gamme assez large de méthodes, que ce soit dans le cas déterministe ou dans le cas stochastique.

On définit la solution χ de l'équation $(III.1)$ au sens des moindres carrés par:

$$\hat{\chi} = Arg.\min_{\chi} \|\rho\|_2 \qquad (III.2)$$

Un problème important dans l'identification des paramètres dynamiques vient du fait que la matrice d'observation W n'est pas déterministe mais aléatoire. Par ailleurs, W et ρ sont des réalisations de variables aléatoires corrélées, ce qui peut biaiser l'estimateur des moindres carrés [77]. Une difficulté supplémentaire provient de la non linéarité de W par rapport à q et $\dot{q}$ qui rend difficile le calcul du biais et de la variance de l'erreur d'estimation, sauf si l'on admet certaines hypothèses d'indépendance sur les bruits [53]. C'est pourquoi, il est important de valider les résultats obtenus par

des expérimentations appropriées.

Pratiquement, on calcule une estimation de l'écart type sur les valeurs identifiées en considérant que W est déterministe et que ρ est un vecteur aléatoire centré, de composantes indépendantes, d'écart type σ_ρ et de matrice de variance covariance C_ρ telle que:

$$C_\rho = E(\rho\rho^T) = \sigma_\rho{}^2 I_r \qquad (III.3)$$

E désignant l'espérance mathématique et I_r la matrice unité d'ordre r .

En supposant que le vecteur des erreurs est centré, de composantes indépendantes et d'égales dispersions, l'écart type σ_ρ peut être calculé par l'estimateur non biaisé suivant:

$$\sigma_\rho = \frac{\|Y - W\hat{\chi}\|^2}{(r-c)} \qquad (III.4)$$

La matrice de variance covariance de l'erreur d'estimation a alors pour expression [77]:

$$C_{\hat{\chi}} = E[(\chi - \chi)(\chi - \chi)^T] = \sigma\rho(W^T W)^T \qquad (III.5)$$

On en déduit l'écart type sur le $j^{ème}$ paramètre

$$[\sigma_{\hat{\chi}j}]^2 = C_{\hat{\chi}jj} \qquad (III.6)$$

Cette interprétation a été utilisée dans [77] mais doit être considérée avec prudence car, dans notre cas, les hypothèses ne sont pas satisfaites puisque W n'est pas déterministe.

L'écart type relatif est estimé par:

$$\sigma_{\hat{\chi}jr}\% = 100\frac{\sigma_{\hat{\chi}j}}{\hat{\chi}_j} \qquad (III.7)$$

L'écart type relatif peut être utilisé comme critère pour déterminer la qualité de l'estimation de la valeur obtenue pour chaque paramètre identifié.

III.4.2.2. Paramètres identifiables

L'identification des paramètres inertiels standard à partir du système $(III.1)$ ne peut pas fournir une solution unique car la matrice d'observation W n'est jamais de rang plein (certaines colonnes de$W$ sont, en effet, des combinaisons linéaires d'autres colonnes et ce, quelles que soient les données $q, \dot{q}$ et $\ddot{q}$. On doit donc déterminer un jeu de paramètres identifiables encore appelés paramètres de base ou paramètres minimaux.

La détermination des paramètres identifiables est un préliminaire nécessaire à l'identification des paramètres dynamiques. Dans ce cas, les formules de regroupement ne sont pas nécessaires, seule la connaissance des paramètres à éliminer ou à regrouper est indispensable.

III.4.2.3. Prise en compte des frottements

En retenant le modèle simplifié habituel des frottement secs et visqueux pour les vitesses non nulles, on peut écrire l'expression du frottement sur l'articulation j :

$$\Gamma_{fj} = F_{Sj}\, Sign(\dot{q}_j) + F_{Vj}\dot{q}_j \qquad (III.8)$$

Où F_{si} et F_{vi} désignent respectivement les paramètres de frottement sec et visqueux de l'articulation i.

Deux approches sont possibles pour identifier ces paramètres :

i) identification globale des paramètres inertiels et des paramètres de frottement: les modèles exposés dans ce chapitre tiendront compte systématiquement de l'équation $(III.8)$.

ii) identification séparée des paramètres de frottement. Compte tenu des difficultés de l'identification liées au nombre assez important de paramètres dynamiques, il semble plus judicieux d'identifier séparément les paramètres de frottement. Pour ce faire, on peut procéder à des essais spécifiques tels que des mouvements axe par axe a vitesse constante [77]. Ils sont ensuite considérés comme connus dans le modèle d'identification. L'inconvénient de cette procédure est le risque de cumul d'erreurs lors de l'identification des paramètres inertiels.

III.4.2.4. Choix des mouvements d'identification

Le choix de mouvements appropriés est déterminant pour l'identification des paramètres dynamiques. En général, un mouvement quelconque ne permet pas d'exciter tous les paramètres. Or, ce problème est directement lié au conditionnement du système linéaire $(III.1)$. Il faut donc prouver un mouvement excitant au cours duquel la matrice d'observation sera bien conditionnée, c'est-à-dire au cours duquel tous les paramètres seront sensibilisés.

On notera que les éléments de la matrice d'observation sont perturbés par les erreurs de modélisation ou d'estimation des variables géométriques qui y interviennent ainsi que par les bruits de mesures. On montre qu'une solution robuste vis-à-vis des perturbations est obtenue lorsque le conditionnement de la matrice W est "bon" et lorsque sa dernière valeur singulière n'est pas trop petite [77].

Pour obtenir un bon conditionnement, deux stratégies peuvent être mises en œuvre :

- calcul d'un mouvement qui aboutit à un conditionnement proche de l'unité (mouvement

excitant);

- identification des paramètres par petits groupes avec des mouvements simples ne sollicitant que certaines articulations, les autres étant bloquées (mouvements séquentiels).

III.4.2.5. Evaluation des variables articulaires

La matrice d'observation W dépend de q et $\dot{q}$ mais aussi de $\ddot{q}$ si l'on utilise le modèle dynamique. En général les robots industriels sont équipés de capteurs de position dont les données sont précises et acceptables. En revanche, les mesures de $\dot{q}$ et de $\ddot{q}$ posent des problèmes. Il en est de même de la détermination de ces variables à partir des informations de position. En effet, la mesure de position par des codeurs introduit un bruit de quantification. La vitesse estimée par dérivation numérique de q est bruitée. De même, la double dérivation pour calculer $\ddot{q}$ aboutit à des données inexploitables car la différentiation amplifie les bruits hautes fréquences.

Une solution est de filtrer le signal de position par un filtre passe-bas avant de le dériver [77]. Cette stratégie a été utilisée avec succès pour identifier les paramètres inertiels du robot Acma SR400 par Restrepo [77] : les dérivations numériques sont faites par différence centrale pour éviter le déphasage : le filtre passe-bas pour la position est du type filtre aller-retour de Butterworth d'ordre 4 avec une fréquence d'échantillonnage de 1 kHz et une fréquence de coupure de 80 Hz. Les paramètres du filtre peuvent être calculés par la fonction "butter" de Matlab.

Une autre solution consiste à éviter le calcul des accélérations articulaires en utilisant des modèles d'identification ne contenant pas l'accélération comme le modèle dynamique filtré proposé par plusieurs auteurs [77] ou le modèle énergétique, ou encore grâce à des filtres stochastiques tel que le filtre de Kalman étendu.

III.4.2.6. Evaluation des couples articulaires

Quelle que soit la méthode employée, l'estimation des paramètres dynamiques revient à minimiser une fonction de l'écart entre la prédiction du modèle (dynamique ou énergétique) et une mesure, fonction des couples et/ou forces articulaires. On voit donc toute l'importance de la précision de cette mesure.

L'usage de couple mètres ou de capteurs d'efforts est peu répandu, en tout cas en milieu industriel, et souvent seules les consignes de courant à l'entrée des amplificateurs de puissance (les variateurs) qui alimentent les moteurs sont connues. La relation entre ces valeurs et les couples doit alors être estimée. En général, compte tenu de la bande passante élevée de la boucle de courant, cette

relation s'apparente à un simple gain constant. Pour l'articulation j, cette relation s'écrit:

$$\Gamma_{mj} = G_{Tj} u_j \qquad (III.9)$$

avec

$$G_{Tj} = N_j \; K_{aj} \, K_{Tj}$$

où G_{Tj} est la constante de la chaîne d'actionnement j, u_j est la consigne envoyée à la chaîne d'actionnement, N_j est le rapport de réduction. K_{aj} est le gain de l'amplificateur et K_{Tj} est la constante de couple du moteur.

Dans le cas d'un asservissement numérique en couple, la consigne u_j est égale à la consigne de commande engendrée par le calculateur. Différentes techniques issues du génie électrique permettent d'identifier chacun des termes entrant dans le calcul du gain global G_{Tj} [77].

D'autres méthodes d'identification des paramètres dynamiques qui ne nécessitent pas explicitement d'évaluer les couples articulaires ont été proposées [77]. Elles consistent à inclure dans le modèle d'identification les gains des chaînes d'actionnement.

III.5. Méthodes d'identification applicable aux robots

III.5.1 Moindres carrés d'erreur de sortie avec modèle direct (Méthode du modèle)

Cette méthode est aussi connue sous l'appellation historique de méthode du modèle [78], [79]. Elle repose sur la minimisation de l'erreur de sortie. Ce critère se distingue de celui fondé sur une erreur d'équation par sa signification physique et sa sensibilité plus importante aux erreurs de structure du modèle et aux valeurs des paramètres (Figure $(III.2)$).

L'estimé $\hat{\chi}_s$ de χ_S obtenu par minimisation d'un critère $J(\varepsilon(\chi_s))$ est défini par :

$$\hat{\chi}_s = \underset{\chi_s}{Arg.\min} J(\varepsilon) \qquad (III.10)$$

Où:

$J(\varepsilon) = \|\varepsilon\|^2 = \varepsilon^T \varepsilon$

$\varepsilon = [\varepsilon_1 ... \varepsilon_{Ne}]^T$ est l'échantillonnage de $\varepsilon(t)$ tel que $\varepsilon_k = \varepsilon(t_k), k = 1,...,N_e$

N_e est le nombre d'échantillons

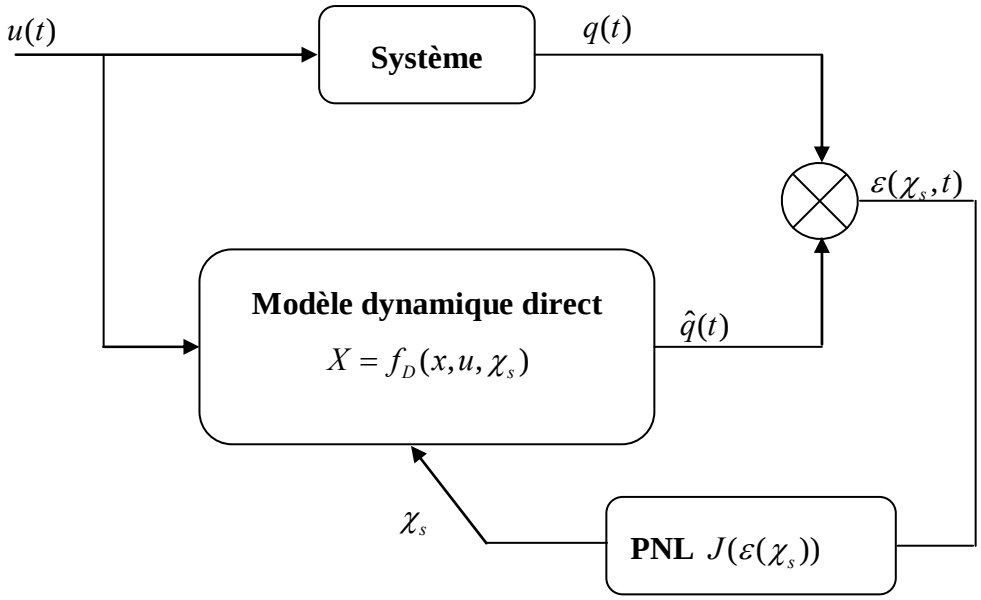

Figure (*III*.2) : Méthode d'erreur de sortie

L'inconvénient de cette méthode apparaître dans sa mise en œuvre plus délicate due à la non linéarité de l'erreur de sortie par rapport aux paramètres, ce qui conduit à un problème d'optimisation non linéaire, souvent difficile à résoudre, et la nécessite de nombreuses intégrations du modèle direct sur un horizon de temps assez long qui est très exigeant en temps de calcul.

III.5.2 Moindres carrés d'erreur d'entrée avec modèle inverse

Cette technique d'identification repose le plus souvent sur un modèle linéaire en fonction des paramètres et l'estimation des paramètres correspond à une minimisation de l'erreur sur l'entrée [80], [81]. Par rapport au méthode de minimisation de l'erreur de sortie, elles ont l'avantage d'être plus simples à mettre en œuvre et d'éviter d'avoir recours à la minimisation d'un critère généralement non convexe, avec tous les problèmes de minima locaux et d'initialisation bien connus. Par contre, elles nécessitent la reconstruction du régresseur, qui contient les vitesses et les accélérations, à partir de mesures bruitées. Ces erreurs sont susceptibles de fausser l'estimation et des filtres doivent être inclus.

L'estimation $\hat{\chi}_s$ de χ_S est obtenue par minimisation d'un critère $J(\rho)$ tel que:

$$\hat{\chi}_s = \underset{\chi_s}{Arg.\min} J(\rho) \qquad (III.11)$$

Dans le cas des moindres carrés ordinaires $J(\rho)$ est défini par :

$$J(\rho) = \|\rho\|2 = \rho^T \rho \qquad (III.12)$$

avec $\rho = [\rho_1 ... \rho_{Ne}]$ et $\rho_k = \rho(t_k)$

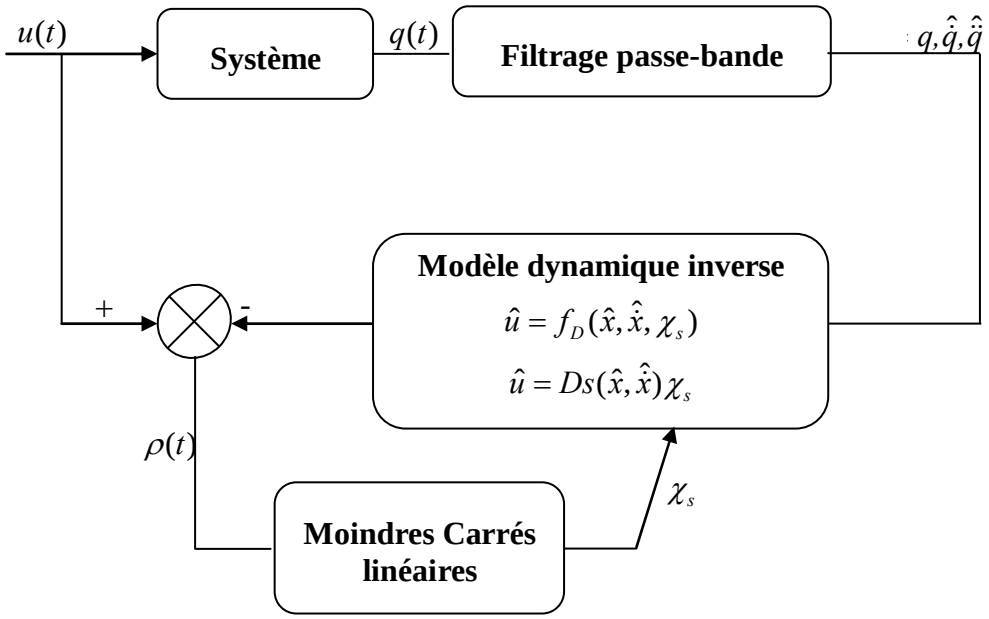

Figure (*III*.3) : Méthode d'erreur d'entrée

Avantages:

- équation de prédiction $\hat{u}$ est donné par le modèle dynamique inverse obtenu naturellement à partir des équations de la physique, forme algébrique par rapport à l'état et sa dérivée.
- Il est plus facile et plus immédiat à calculer que le modèle d'état direct, en utilisant des logiciels de calcul formel spécialisés [82].
- pas d'intégration d'équation différentielle.
- Pas de problème des conditions initiales sur l'état et les paramètres.
- $\hat{\chi}_s$ est obtenu par les techniques de régression linéaire multivariable.

Inconvénients :

- estimation de l'état et de sa dérivée, nécessaires au calcul du régresseur Ds .

III.5.3. Mise en œuvre des moindres carrés

III.5.3.1. Identifiabilité

Pour les robots, on calcule un jeu de paramètres identifiables χ , appelés paramètres minimaux ou encore paramètres de base [83], caractérisant complètement le modèle dynamique. L'utilisation de ces paramètres dans le calcul du modèle dynamique diminue sa complexité. En outre, cette étape est un préliminaire indispensable pour l'évaluation par identification des paramètres inertiels, les paramètres de base constituant les seuls paramètre identifiables. Ils sont obtenus à partir des paramètres standard

en éliminant ceux qui n'ont pas d'effet sur le modèle dynamique et en regroupant certains autres entre eux.

Le modèle dynamique simplifié et minimal en nombre de paramètres est donné par:

$$\Gamma = D(q,\dot{q},\ddot{q})\chi \qquad (III.13)$$

$D(q,\dot{q},\ddot{q})$: régresseur minimal associé au vecteur χ des paramètres minimaux.

III.5.3.2. Problème du risque de biais

L'échantillonnage du modèle minimal $(III.13)$ conduit à un système linéaire surdéterminé de plein rang structurel donné par :

$$Y_m = W_m(\hat{q},\hat{\dot{q}},\hat{\ddot{q}})\chi + \rho_m \qquad (III.14)$$

$W_m(\hat{q},\hat{\dot{q}},\hat{\ddot{q}})$: matrice $((n \times N_e) \times p)$ d'observation, de plein rang structurel.

La solution des moindres carrés ordinaires de l'équation $(III.13)$ obtenu par minimisation d'un critère $J(\rho)$ est défini par:

$$\hat{\chi} = \underset{\chi}{Arg.\min} \left\|\rho_m\right\|^2 = Wm + Ym \qquad (III.15)$$

Puisque les mesures expérimentales ou les estimations de $\Gamma(t_i), q(t_i), \dot{q}(t_i), \ddot{q}(t_i)$ sont bruitées, alors les matrices Wm et Ym sont perturbées, ce qui risque de biais possible de Moindres Carres si ces matrices aléatoires ne sont pas indépendantes.

Les coefficients du régresseur $D(q,\dot{q},\ddot{q})$ sont des fonctions non linéaires de $(q,\dot{q},\ddot{q})$ et il est pratiquement impossible de calculer l'effet théorique de ces perturbations sur le biais et la variance de l'estimation. Afin de minimiser leurs effets, une stratégie est visée. Elle s'appuie sur les deux points suivants:

- filtrage des données pour calculer un système filtré équivalent au système de l'équation $(III.13)$, mais peu perturbé.
- identification en boucle fermée de position pour suivre un mouvement riche en information dit excitant, qui excite les paramètres.

III.5.4. Validation du modèle

La validation est un aspect très important pour vérifier l'applicabilité générale d'un procédé d'identification développé. Cependant, il n'est pas fréquemment discuté dans la littérature. Bien que la

validation n'ait pas d'influence sur l'amélioration des résultats d'identification, c'est une étape indispensable du procédé d'identification.

Une validation insuffisante mènera à reconsidérer certaines étapes précédentes du procédé d'identification. Cette section propose quelques méthodes pour valider l'exactitude d'une identification expérimentale. Deux méthodes principales existent pour:

1. évaluer l'exactitude du couple prévu.

2. vérifier l'exactitude des paramètres estimés.

La première méthode de validation est importante pour la commande fondée sur le modèle dynamique, tandis que la seconde est appliquée pour l'identification de la charge. La première méthode de validation simule le système utilisant les valeurs de paramètre estimées. En particulier, les équations du modèle dynamique inverse sont évaluées et les couples requis pour le mouvement désiré sont calculés. Une bonne validation exige que la trajectoire de validation doive être différente de la trajectoire d'excitation.

La prise en compte des couples est une méthode appropriée de validation lorsque le modèle identifié sera utilisé pour la commande. Les critères suivants peuvent être employés pour évaluer la qualité des couples prévus :

A). La comparaison des couples prévus et mesurés.

Le traçage des deux couples à la fois prévus et mesurés sur la même figure nous permet de faire évaluer leurs similarités facilement.

B). Erreur de prédiction.

Une alternative emploie la différence entre le couple prévu et mesuré, c-à-d l'erreur de prédiction ou le résidu de prédiction.

$$\varepsilon = \Gamma - D(q, \dot{q}, \ddot{q})\chi \qquad (III.16)$$

L'erreur de prédiction devrait être petite devant le couple.

Ces techniques graphiques sont très faciles à utiliser parce qu'elles donnent une impression directe du résultat. Un utilisateur expérimenté pourrait être capable de trouver une indication de ce qui est probablement la cause des erreurs au cours de la phase d'identification. L'inconvénient de ces techniques est leur subjectivité.

C). Valeur efficace de l'erreur de prédiction.

Pour améliorer l'objectivité de la validation, une certaine mesure de qualité est nécessaire pour évaluer le degré de précision d'une expérience particulière d'identification. La valeur utilisée à cet effet est la valeur efficace de l'erreur de prédiction. Elle est calculée de la manière suivante:

$$\Delta\Gamma_{RMS} = \sqrt{\frac{1}{K}\sum_{k=1}^{K}\varepsilon_2^k} = \sqrt{\frac{1}{K}(\Gamma - D(q,\dot{q},\ddot{q})\chi)^T(\Gamma - D(q,\dot{q},\ddot{q})\chi)} \qquad (III.17)$$

où K est le nombre d'échantillon dans la trajectoire.

L'erreur de prédiction pour chaque couple donne une idée de l'incertitude sur la prédiction de couple.

III.6. Formulation du problème

III.6.1. *Modèle dynamique du robot*

Le robot manipulateur utilisé pour le problème d'identification est donné par la figure suivante :

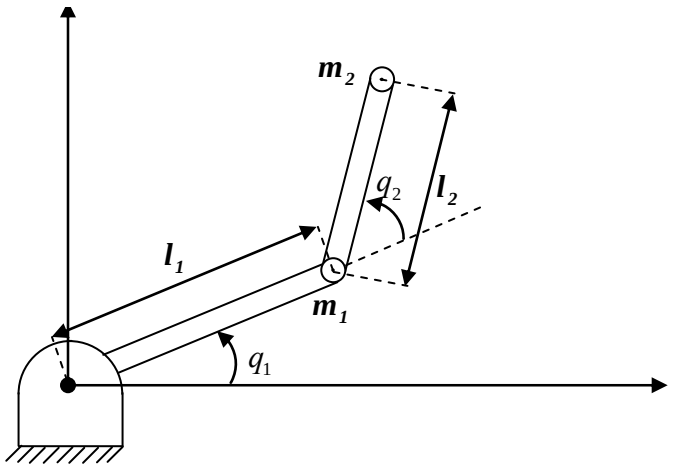

Figure (III.4) : Robot manipulateur à deux degrés de liberté

Son modèle dynamique peut être construit par l'application des formules de Lagrange, il est donné par l'équation matricielle suivante :

$$\Gamma = M(q).\ddot{q} + C(q,\dot{q}).\dot{q} + G(q) + H(\dot{q}) \qquad (III.18)$$

Avec :

$q = [q_1 \quad q_2]^T$ vecteur des variables articulaires généralisées.

$\Gamma = [\Gamma_1 \quad \Gamma_2]^T$ vecteur des couples généralisés.

La matrice d'inertie

$$M(q) = \begin{bmatrix} a_1 + a_2\cos(q2) & a_3 + \frac{a_2}{2}.\cos(q_2) \\ a_3 + \frac{a_2}{2}.\cos(q_2) & a_3 \end{bmatrix} \qquad (III.19)$$

Vecteur de force de Coriolis et de centrifuge

$$N(q,\dot{q}) = C((q,\dot{q})\dot{q}) = \begin{bmatrix} -(a_2\sin(q_2))(\dot{q}_1.\dot{q}_2 + \frac{\dot{q}_2^2}{2}) \\ (a_2\sin(q_2))\frac{\dot{q}_1^2}{2} \end{bmatrix} \qquad (III.20)$$

Vecteur des forces de gravité

$$G(q) = \begin{bmatrix} a_4\cos(q_1) + a_5\cos(q_1+q_2) \\ a_5\cos(q_1+q_2) \end{bmatrix} \qquad (III.21)$$

Vecteur exprimant les frottements.

$$H(\dot{q}) = \begin{bmatrix} V_1.\dot{q}_1 + V_2.\mathrm{sgn}(\dot{q}_1) \\ V_3.\dot{q}_2 + V_4.\mathrm{sgn}(\dot{q}2) \end{bmatrix} \qquad (III.22)$$

$a_1 = l_1^2(m_1+m_2) + l_2^2 m_2$, $a_2 = 2l_1 l_2 m_2$, $a_3 = l_2^2 m_2$, $a_4 = (m_1+m_2)l_1 g$, $a_5 = m_2 l_2 g$ sont les paramètres d'inertie.

V_1 et V_3 sont les paramètres de frottement sec, V_2 et V_4 sont les paramètres de frottement visqueux.

III.6.2. ***Paramètres de base identifiables***

La méthode utilisée est basée sur un modèle linéaire en fonction des paramètres et l'estimation de ces paramètres correspond à une minimisation de l'erreur sur l'entrée.

Rappelons que le modèle dynamique simplifié et minimal en nombre de paramètres est donné par l'équation $(III.13)$:

$\Gamma_e = D(q,\dot{q},\ddot{q})\chi$

$D(q,\dot{q},\ddot{q})$: régresseur minimal associé au vecteur χ des paramètres minimaux.

Où le vecteur des paramètres à identifier est donné par :

$$\chi = [a_1 \quad a_2 \quad a_3 \quad a_4 \quad a_5 \quad V_1 \quad V_2 \quad V_3 \quad V_4] \qquad (III.23)$$

Le problème d'optimisation multiobjectif est défini de tel sorte à minimiser l'écart entre le couple calculé et le couple estimé, il est donné par :

$$\rho = \Gamma - D(q,\dot{q},\ddot{q})\chi \qquad (III.24)$$

avec

$\rho = \begin{bmatrix} \rho_1 \\ \rho_2 \end{bmatrix}$: l'écart entre le couple estimé et calculé.

$\Gamma = \begin{bmatrix} \Gamma_1 \\ \Gamma_2 \end{bmatrix}$: le couple calculé, $\Gamma_e = D(q,\dot{q},\ddot{q})\chi = \begin{bmatrix} \Gamma_{e1} \\ \Gamma_{e2} \end{bmatrix}$: le couple estimé.

Le problème d'optimisation multiobjectif est donc défini par :

$$\begin{cases} \min J(\rho_1) = \rho_1^T \rho_1 \\ \min J(\rho_2) = \rho_2^T \rho_2 \end{cases} \qquad (III.25)$$

III.6.3. *Collections des données*

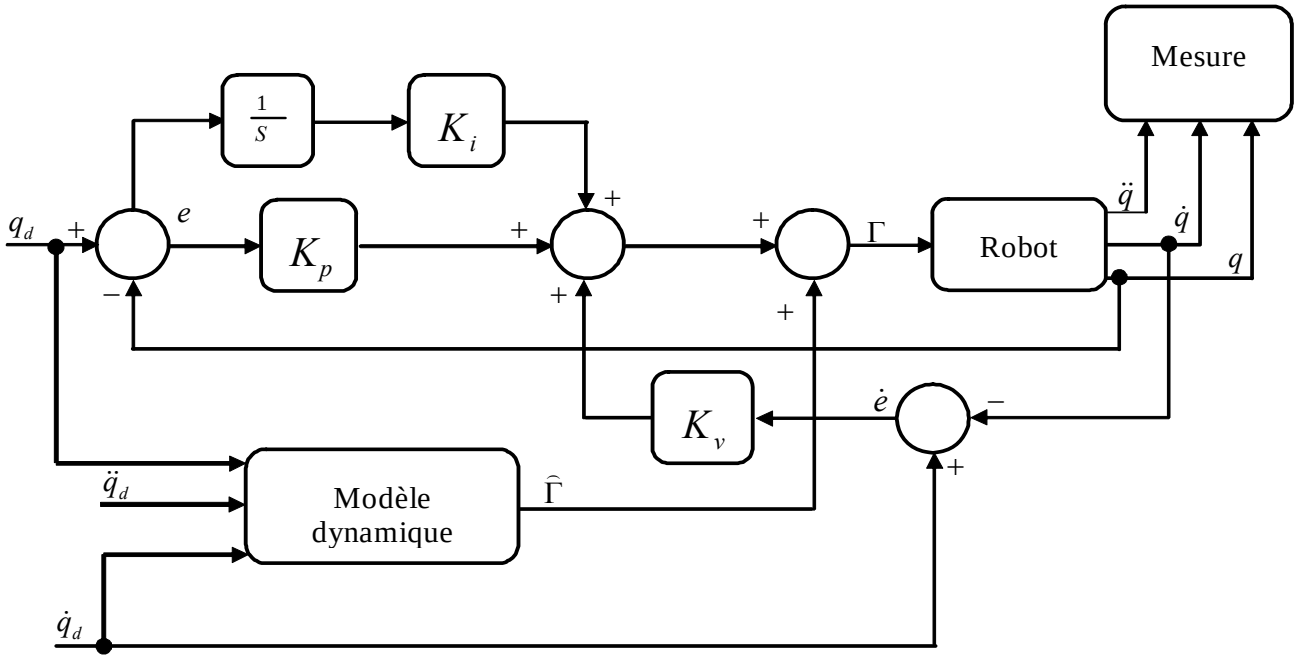

Figure *(III .5)* : Schéma bloc du robot dans la boucle de commande

Les données nécessaires à l'identification (Figure *(III .5)*), qui sont les positions, les vitesses et les accélérations, sont collectées alors que le robot sous contrôle suive une trajectoire de référence qui vérifie la condition d'excitation persistante (du point de vue des systèmes linéaires).

Les couples appliqués au robot sont générés à partir de la loi de commande utilisée en boucle

fermée. Il s'agit des couples estimés par modèle dynamique associé avec un régulateur PID (commande par couple calculé) où les paramètres du régulateur sont optimisés par l'algorithme génétique. Le modèle dynamique direct (robot simulé) nous fournit les accélérations articulaires $\ddot{q}$, alors que les vitesses $\dot{q}$ et les positions articulaires q sont calculées par l'intégration du modèle dynamique direct par la méthode numérique d'Euler.

III.6.4. ***Trajectoires d'excitation***

Les trajectoires ont été calculées de façon à assurer un bon conditionnement de la matrice d'observation W. L'identification est réalisée en boucle fermée, c'est-à-dire que les données nécessaires à l'identification sont prises alors que le robot, commandé par la méthode du couple calculé, suive les trajectoires excitantes (Figure *(III.5)*).

La procédure utilisée pour choisir des trajectoires qui excitent au mieux les paramètres, et assure un bon conditionnement de la matrice d'observation est basée sur deux critères :

- L'utilisation d'une trajectoire riche en fréquence qui contient au minimum un nombre de fréquence supérieur ou égale à la moitié du nombre des paramètres à identifier. Pour cet effet on a introduit une trajectoire qui contient 5 fréquences (on a 9 paramètres à identifier). Nous avons considéré des trajectoires de référence sinusoïde assurant une continuité en position, vitesse et accélération donnée par :

$$q_i^{exc}(t) = (\sum_{j=1}^{K} \frac{1}{j2\pi\Delta f}[a_{i,j}\sin(j2\pi\Delta ft\ ) + b_{i,j}\cos(j2\pi\Delta ft)]\ \ i = 1,2 \qquad (III.26)$$

  Où K défini le nombre de fréquence contenu dans la trajectoire

- Construire une trajectoire pour identifier spécifiquement certains paramètres. Par exemple, pour identifier le couple de frottement, on peut utiliser des paliers de vitesse croissante. Pour identifier les termes de gravité, on peut utiliser des allers retours à faible vitesse [84] et pour les inerties on peut choisir des trajectoires sinusoïdales.

On peut aussi assurer le bon conditionnement de la matrice d'observation par la résolution d'un problème d'optimisation non linéaire dont le critère à minimiser est $cond(W)$ avec :

$cond(W) = \sqrt{\frac{\sigma_{\max}}{\sigma_{\min}}}$ où $\sigma_{\max}$ et $\sigma_{\min}$ sont respectivement les valeurs singulières maximales et minimales de la matrice W.

III.6.5.*Résolution par les algorithmes génétiques*

La procédure d'implémentation des algorithmes génétiques multiobjectifs, pour le problème d'identification, est présentée par l'organigramme de la figure *(III .6)* (Algorithme NSGA-II) et par la figure*(III .7)* pour notre propre algorithme (S_MOGA). Il commence par l'acquisition des données nécessaires (couples, positions, vitesses, et accélérations) et des intervalles de recherche des paramètres considérés par le problème qui sont les paramètres inertiels :

- a_1 définit sur l'intervalle $[5,10]$;
- a_2 définit sur l'intervalle $[0.1,5]$;
- a_3 définit sur l'intervalle $[0.1,5]$;
- a_4 définit sur l'intervalle $[110,120]$;
- a_5 définit sur l'intervalle $[40,50]$;

Et les paramètres de frottement :

- V_1, V_2, V_3 et V_4 définies sur l'intervalle $[0.1,2]$ chacune.

III.6.5.1. Implémentation de l'algorithme NSGA-II

L'algorithme NSGA-II utilisé dans ce stade est démarré par une génération aléatoire de la population initiale de taille $N = 100$ sur tout le domaine de définition des paramètres du problème qui sont représentés par des valeurs réelles (codage réel). Ensuite la population est triée selon le principe de dominance (classement en fronts).

La génération de la population enfant est basée sur l'application des opérateurs génétiques (sélection, croisement, et mutation), l'algorithme utilise la sélection par tournoi, il adopte un croisement de type 'SBX' (*Simulated Binary Crossover*) avec une probabilité de 0.9 et une mutation polynomiale avec une probabilité de 0,1. Après que la population enfant est crée, elle est groupée avec la population parent et reproduit une population de taille $2N$ qui est classée à son rôle en fronts.

La création de la nouvelle population se base, en premier lieu, sur le remplissage par ordre croissant des fronts jusqu'à ce que la taille de la population soit égale à N. A ce point là, si la population dépasse la taille exigée le dernier front est inclus dans la population en se basant sur les individus avec la grande distance (Crowding distance)

Les meilleurs individus de chaque génération sont recopiés comme archive et employés dans les générations suivantes avec un taux égale à $0{,}1$ (modèle élitiste).

Cette procédure a été répété autant de fois jusqu'à atteindre le nombre maximale de générations demandé qui est fixé par $\max gen = 100$.

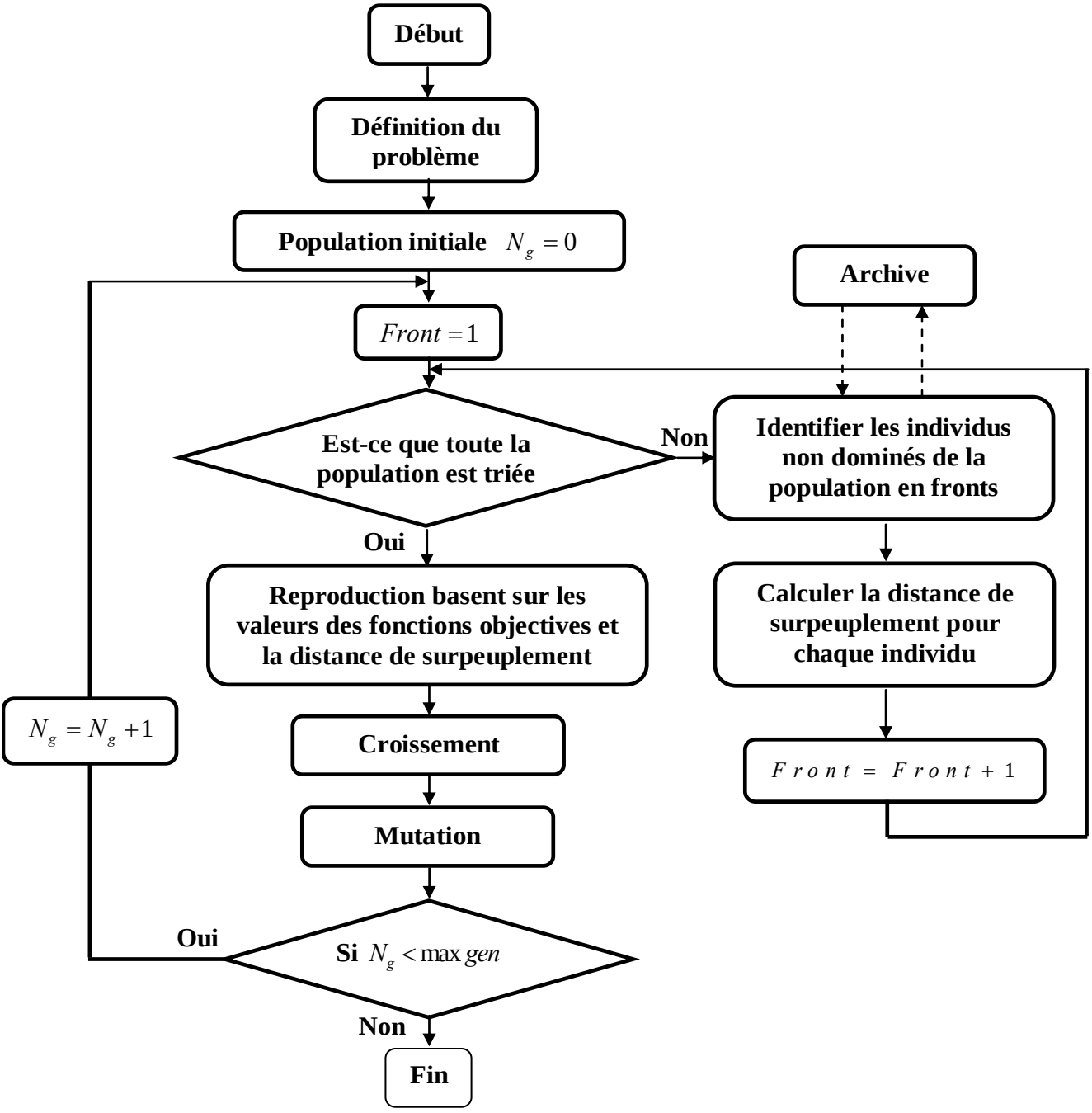

Figure *(III .6)* : Organigramme de l'algorithme NSGA-II

***III.6.5.2**. Implémentation de l'algorithme proposé (S_MOGA)*

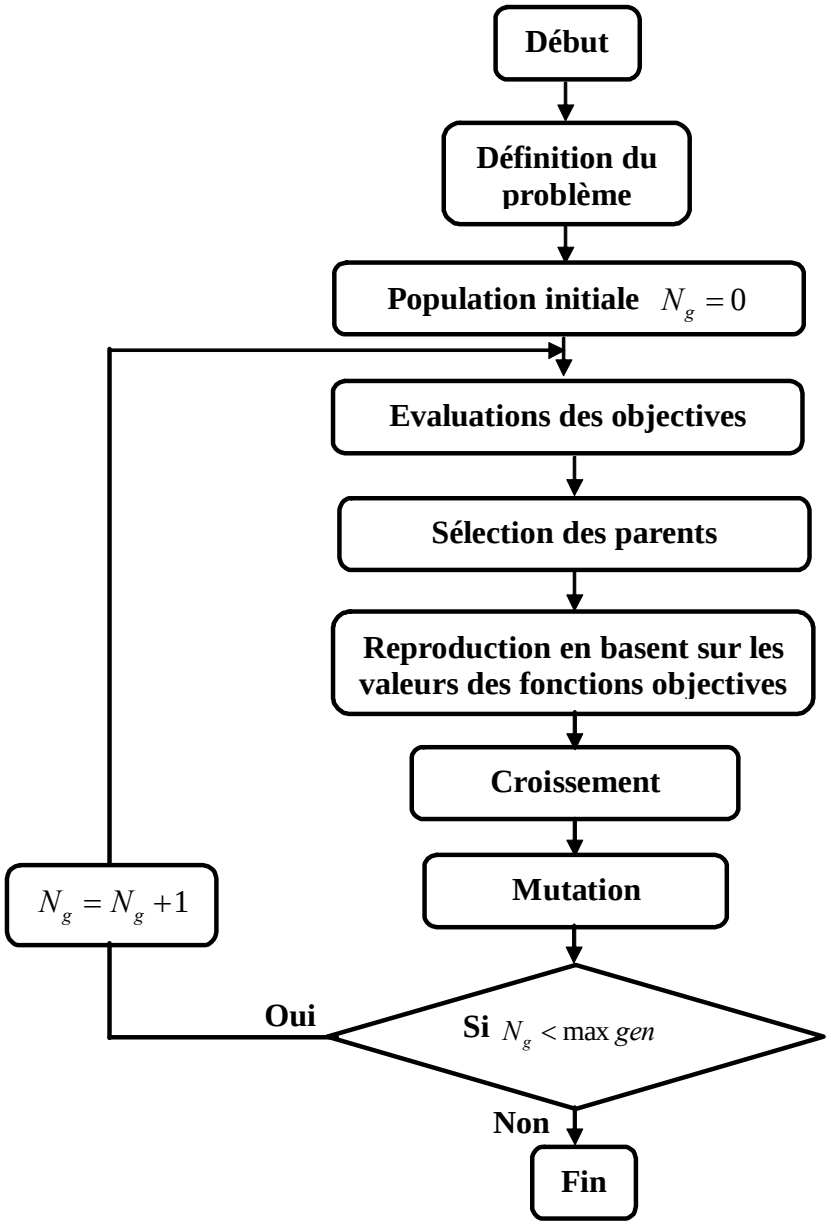

Figure *(III .7)* : Organigramme de l'algorithme S_MOGA

La mise en œuvre de l'algorithme proposé est basée sur une génération aléatoire de la population initiale de taille $N = 100$ sur tout l'espace de recherche, où les individus sont représentés par des chaînes de bits identifiantes les paramètres du problème (codage binaire).

Pour l'évolution des populations et la génération des nouvelles solutions nous avons adopté une sélection proportionnelle basée sur les valeurs des fonctions objectives (sélection par roulette) qui favorise la sélection et la reproduction des individus les plus adaptés (population parent). Cette dernière est soumise à l'opérateur de croisement en un site avec une probabilité de 0.7 et une mutation sélective

avec une probabilité de 0.02 pour créer la population enfant. Ensuite, une compétition a lieu entre parents et enfants pour déterminer les éléments de la génération suivante.

La population initiale donne ainsi naissance à des générations successives, mutées et hybridées à partir de leurs parents. Ce processus est réitéré jusqu'à ce que un critère d'arrêt soit satisfait. Pour notre algorithme le critère d'arrêt est un nombre maximal de générations qui est fixé par $\max gen = 100$.

III.7. Résultats de simulation

III.7.1. ***Résultats d'identification par moindres carrés d'erreur d'entrée***

Le problème d'identification a été résolu en premier temps en utilisant l'algorithme d'optimisation multiobjectif "Non dominated Sorting Genetic Algorithm II" (NSGAII) présenté ci-dessus (Figure *(III .6)*), ensuite en utilisant notre algorithme S_MOGA présenté par la figure *(III .7)* . Une fois l'ensemble des paramètres est identifié, il sera possible de choisir le meilleur à l'aide d'une étape de validation sur une base dédiée. Les résultats d'exécution de ces deux algorithmes sont montrés dans le tableau ci-après :

Tableau *III*.1 : ***Paramètres identifiés par les deux algorithmes NSGA-II et S_MOGA***

Paramètres	Valeur à priori	Valeur estimée avec l'algorithme NSGA II	Valeur estimée avec l'algorithme S_MOGA
a_1	7,2093	7.3004	7.2023
a_2	4,2401	4.2268	4.2463
a_3	2,1200	2.1358	2.1085
a_4	115,5681	115.6300	115.4504
a_5	48,1428	48.0177	48.1232
V1	1	0.9096	0.9997
V2	0,5	0.5970	0.5021
V3	1	0.9701	0.9419
V4	0,5	0.3984	0.4450

L'évolution de la valeur efficace de l'erreur de prédiction au cours de chaque génération est présentée par la figure suivante :

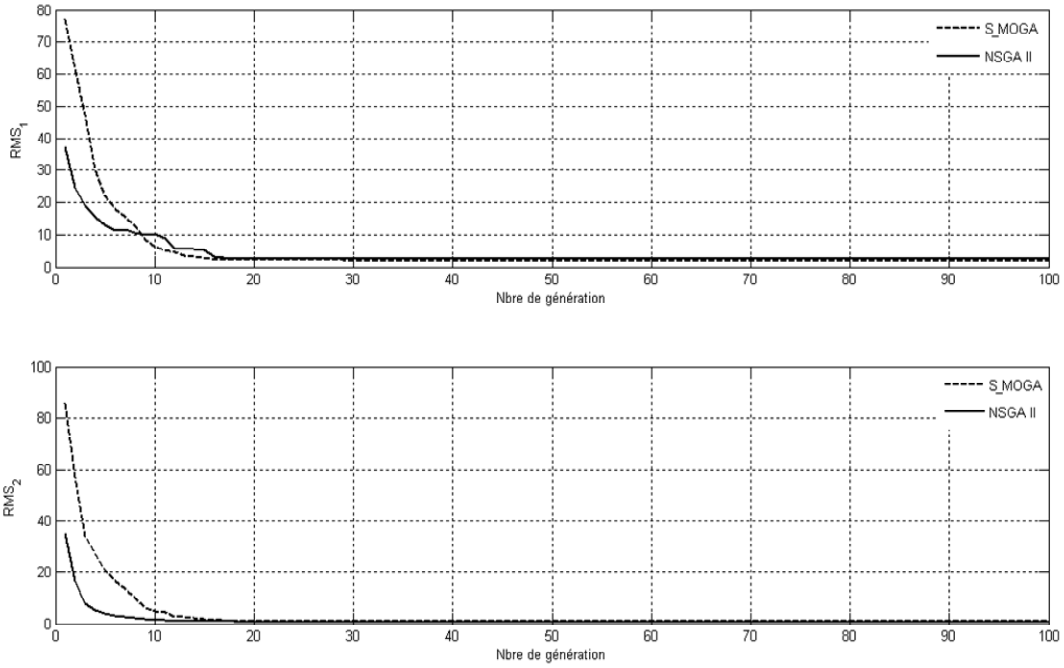

Figure (*III* .8) : Evolution de la valeur efficace de l'erreur de prédiction

Ensuite, l'évolution des paramètres d'inertie est présentée par la figure suivante :

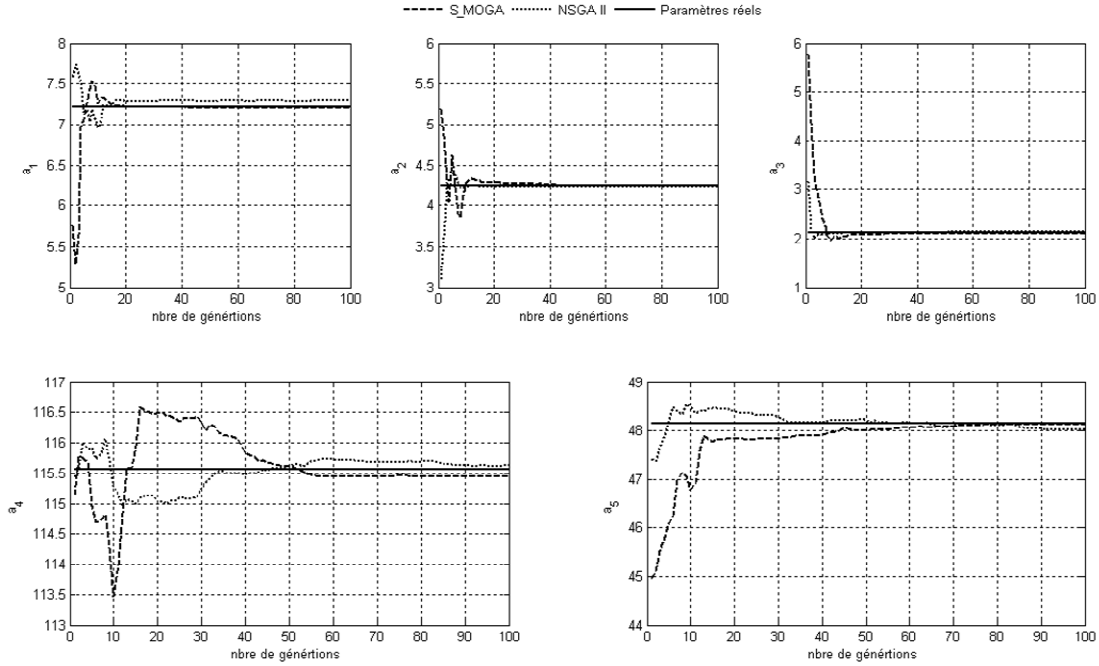

Figure (*III* .9) : Evolution des paramètres d'inertie

En fin, l'évolution des paramètres de frottements est présentée par la figure suivante :

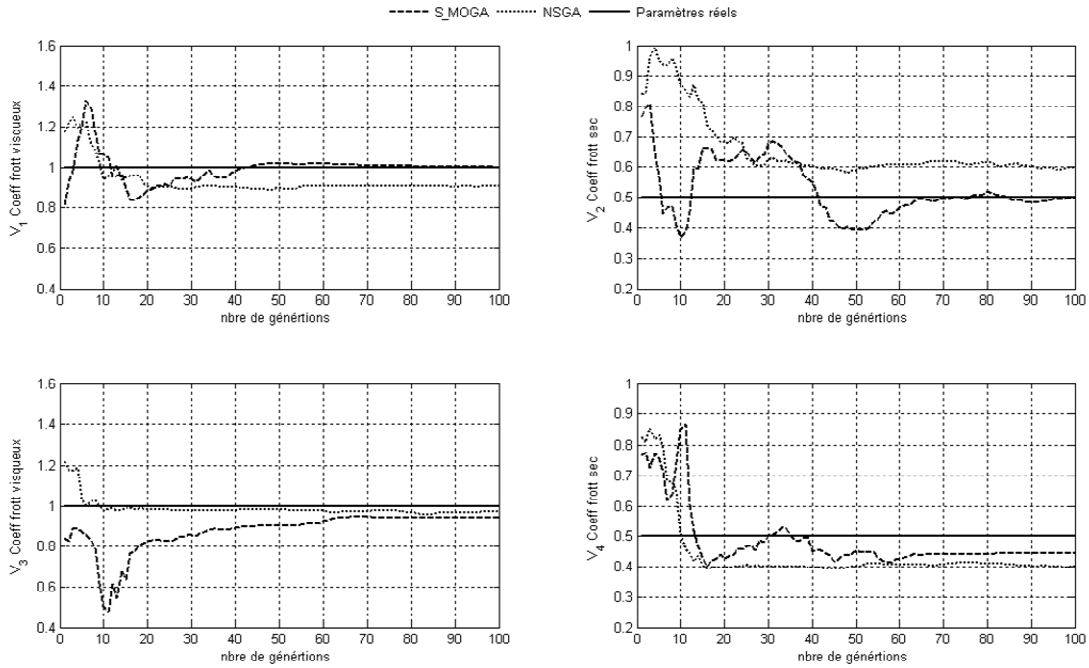

Figure (*III*.10) : Evolution des paramètres de frottements

Interprétation des résultats

Les résultats obtenus par les deux algorithmes durant la phase d'identification montrent une légère différence entre les valeurs finales, notant que notre algorithme est légèrement meilleur, mais l'algorithme NSGA II montre leur convergence très rapide vers la solution Pareto-optimale, ce qui est confirmé par la convergence des paramètres d'inertie et ceux de frottements.

Notre algorithme s'avère meilleur que NSGA-II car ce dernier utilise un mécanisme d'évolution qui rassemble à partir d'une certaine génération toute la population dans le premier front non-dominé. A ce stade, des solutions Pareto-optimales situées dans une région très "peuplée", peuvent être éliminées en laissant la place à des solutions non-dominées dans la population courante mais qui ne sont pas Pareto-optimales.

III.7.2. Résultats Validation :

Apres cette phase d'identification, il est indispensable de procéder à une étape de validation. La méthode proposée pour cet effet permet d'évaluer la différence entre le couple calculé par modèle dynamique avec paramètres réels et le couple estimé par modèle dynamique avec paramètres obtenus par les algorithmes génétiques (Figure *(III*.11*)*).

Cette méthode est très importante pour la commande fondée sur le modèle dynamique employé

par la suite. Elle permet de vérifier la conformité des paramètres identifiés par rapport aux paramètres réels.

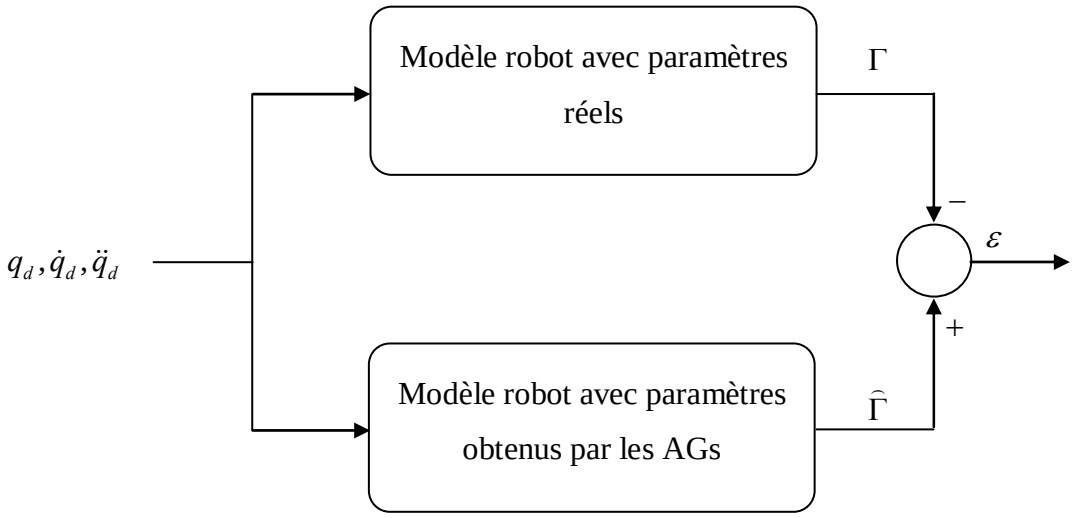

Figure *(III .*11*)* : Schéma bloc de validation.

La validation du modèle est effectuée avec des trajectoires différentes de celles utilisées pour la phase d'identification en mettant en œuvre le modèle dynamique comme dans l'algorithme de commande par couple calculé.

Les résultats de validation sont présentés par les figures (*III* .12) et (*III* .13) .

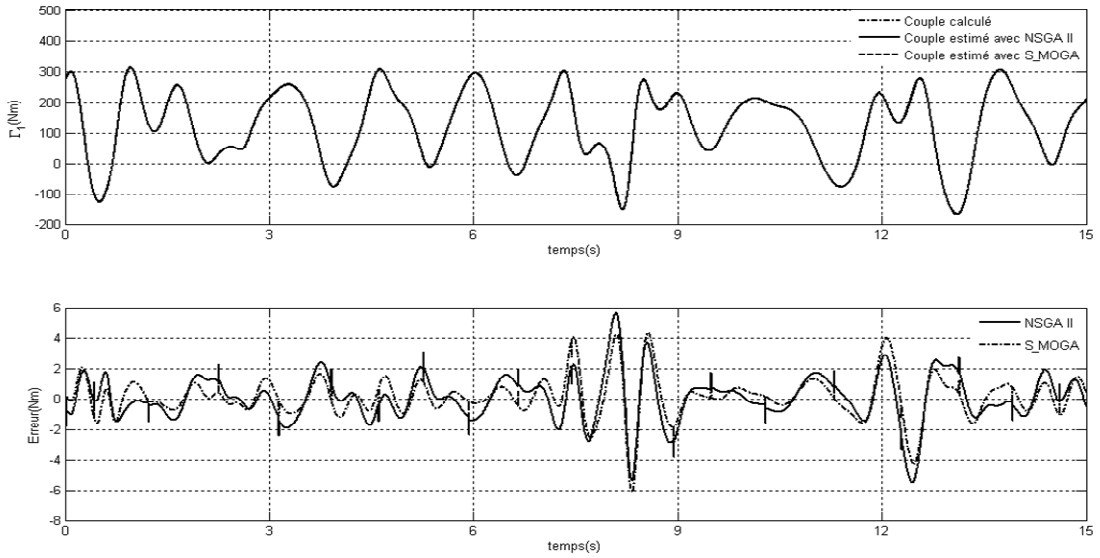

Figure (*III* .12) : Erreur de prédiction sur le 1^{er} couple

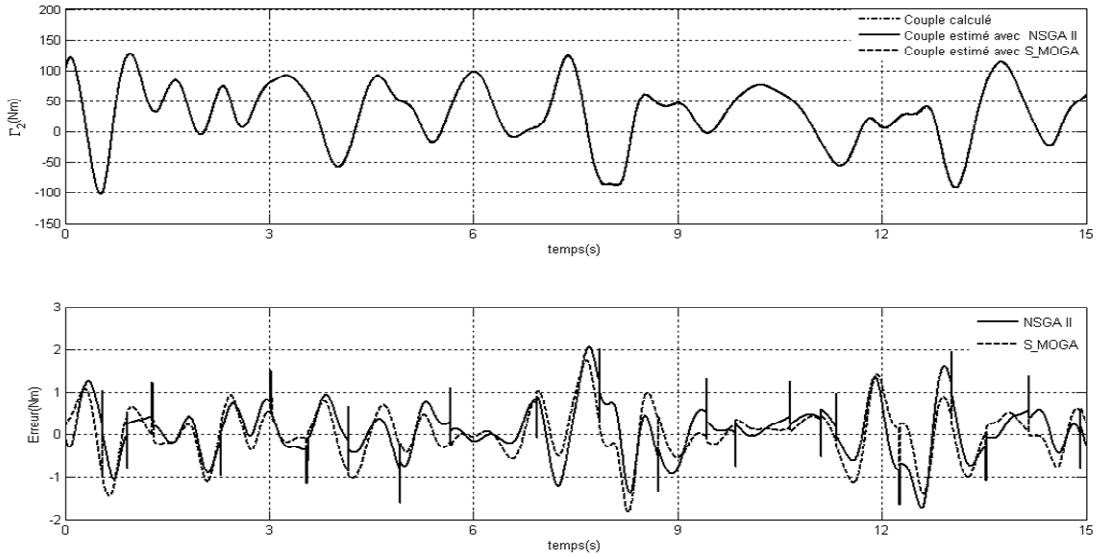

Figure (*III*.13) : Erreur de prédiction sur le $2^{ème}$ couple

Pour mesurer la qualité des résultats et évaluer le degré de précision des algorithmes utilisés, nous avons appliqué les métriques citées dans la section *III*.5.4.

La valeur efficace des erreurs de prédiction du modèle, en utilisant la trajectoire d'excitation et de validation respectivement, est présentée dans le tableau suivant :

Tableau *III*.2 : ***Valeur efficace de l'erreur de prédiction sur les deux articulations***

Erreur prédiction *RMS*	Couple 1		Couple 2	
	NSGA II	S_MOGA	NSGA II	S_MOGA
trajectoire excitation	2.0557	2.0019	0.7558	0.7551
trajectoire validation	1.5163	1.3420	0.6140	0.5836

Interprétation des résultats

Les résultats de simulation montrent que les couples estimés et les couples calculés qui sont tracés sur les mêmes figures sont similaires, ce qui est confirmé par l'erreur de prédiction qui est très petit devant ces couples.

Les valeurs efficaces de l'erreur de prédiction calculée pour les deux trajectoires d'excitation et de validation montrent que notre algorithme S_MOGA est plus précis que l'algorithme NSGA-II.

III.8. Conclusion

Le problème d'identification étudié dans ce chapitre est un problème d'optimisation multiobjectif. La qualité des solutions qui est trouvée lors de l'optimisation apparaît donc plus importante que la rapidité de leur obtention surtout en utilisant l'algorithme NSGA II. D'autre part, la validation du modèle obtenue reste une étape indispensable pour vérifier leur exactitude. Le choix des paramètres de l'algorithme d'optimisation qui définissent le temps de calcul a été pris en compte.

La performance de la distance de surpeuplement (voir chapitre II) a été testée dans le cadre de l'optimisation des paramètres dynamiques et de frottements du robot, formulé ici comme un problème bi-objectif. La comparaison de résultats obtenus avec et sans utilisation de cet opérateur de diversification témoigne de l'accélération de la convergence lors de l'application des deux algorithmes à l'étape de l'optimisation des objectifs du problème.

Chapitre IV

Application des algorithmes génétiques multiobjectifs à la commande

IV.1. Introduction

La synthèse d'une loi de commande passe par l'utilisation des modèles qui ne sont qu'une représentation imparfaite de la réalité : hormis le fait que les lois de la physique ne fournissent qu'une représentation globale des phénomènes, donc valable à une certaine échelle, il y a toujours des incertitudes de modélisation, de sorte qu'on ne peut pas décrire exactement par un modèle mathématique le comportement d'un processus physique.

Dès lors qu'on travaille sur des modèles, dont la validité est limitée, il faut se préoccuper de la robustesse de la loi de commande, c'est-à-dire être capable de garantir non seulement la stabilité mais aussi certaines performances vis-à-vis d'incertitudes de modèle. Ce dernier point demande de compléter le travail de modélisation par une description précise des incertitudes de modèle, et de les inclure dans un formalisme général permettant de les prendre en compte et d'en déduire certaines conclusions.

La synthèse d'une loi de commande s'articule ainsi autour de deux étapes fondamentales, qui, en pratique, sont répétées alternativement jusqu'à ce que le concepteur juge les résultats satisfaisants :

- Calcul du régulateur : dans cette étape, peuvent être pris en compte certains objectifs de performances et certains objectifs de robustesse seulement.
- Analyse des propriétés du système commandé, du point de vue des performances que de la robustesse de celles-ci.

L'évaluation des performances d'un système ou la synthèse d'un correcteur peut se faire suivant différents critères mathématiques. On peut par exemple chercher à :

- obtenir un régime transitoire au moins aussi rapide que la fonction e^{-at} (avec $a > 0$).
- placer les pôles du système asservi dans une région du plan complexe.
- limiter la commande à une valeur donnée, pour un ensemble de conditions initiales spécifie.

Moyennant certaines hypothèses de ces objectifs peuvent se formuler comme des contraintes, et donc déboucher sur des procédures de synthèse de correcteur.

IV.2. Application des algorithmes génétiques multiobjectifs au problème de commande

Afin de valider nos approches et de montrer leur efficacité, deux procédés ont été retenus. Ces derniers présentent des caractéristiques intéressantes. Nous ne manquerons pas de donner les raisons pour lesquelles nous les avons choisis et nous expliquerons pourquoi la commande multiobjectif apparaît appropriée à ce type de procédés. Tout d'abord, le premier procédé est le pendule inversé, ce

procédé est un système non linéaire instable en boucle ouverte et il est couramment utilisé dans la littérature. Il représente en quelque sorte un système de référence en automatique. Le deuxième procédé est un robot manipulateur à deux degrés de liberté. D'une part, il présente des caractéristiques non linéaires introduites par le couple résistant de la charge, d'autre part, la variation de certains paramètres tels que la masse de la charge est envisageable. Comme nous le verrons par la suite, la commande multiobjectif peut parfaitement s'appliquer à ce procédé.

Des résultats de simulation sur le pendule inversé et le robot manipulateur seront exposés dans ce chapitre.

IV.2.1. Procédés utilisés

IV.2.1.1.Pendule inversé

Le problème du pendule inverse a été très largement utilisé au cours des dernières années pour tester les algorithmes appliqués au contrôle des systèmes non linéaires ([85], [86], [87], [88], [89], [90]), fait figure d'étalon «Benchmark» dans le domaine du contrôle [91].

Le pendule inverse est constitué d'un chariot mobile se déplaçant sur une piste linéaire et portant un balancier pouvant pivoter autour d'un axe fixé sur le chariot (figure $(IV.1)$). Le système est simplement commandé en appliquant au chariot une force longitudinale u, le but étant de maintenir le balancier en équilibre sans que le chariot ne sorte de la piste.

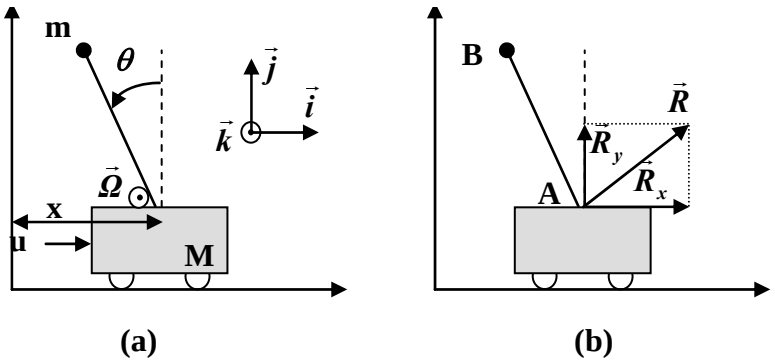

Figure $(IV.1)$:Pendule inversé

IV.2.1.2.Modélisation du pendule inversé

On considère le système, appelé pendule inversé, formé d'un pendule posé en équilibre instable sur un chariot roulant, comme représenté sur la figure $(IV.1)$. La quantité u est la force exercée sur le chariot de masse M, x indique la position du chariot, θ est l'angle entre le pendule et la verticale et $\vec{R}$ est la force exercée par le chariot sur le pendule. A l'extrémité B du pendule est fixé une masse m.

On négligera la masse de la tige du pendule. Enfin, A est le point d'articulation entre la tige et le chariot et $\vec{\Omega} = \dot{\theta}.\vec{k}$ est le vecteur de rotation associé à la tige.

Pour obtenir les équations d'état de ce système [92], nous allons appliquer le principe fondamental de la dynamique sur le chariot et le pendule qui nous donne :

$(u - R_x)\vec{i} = M\ddot{x}\vec{i}$ (Chariot en translation) $(IV.1)$

$R_x\vec{i} + R_y\vec{j}$ - $mg\vec{j} = m\dot{v}_B$ (Pendule en translation) $(IV.2)$

$R_x \cos\theta + R_y \sin\theta = 0$. (Pendule en rotation) $(IV.3)$

où v_B est le vecteur vitesse du point B. Pour la troisième équation, le moment d'inertie du pendule a été posé nul. Puisque

$\overrightarrow{OB} = (x - l.\sin\theta\vec{i}) + l\cos\theta\,\vec{j}$ $(IV.4)$

nous avons

$v_B = (\dot{x} - l.\dot{\theta}.\cos\theta)\vec{i}$ - $l.\dot{\theta}.\sin\theta\,\vec{j}$ $(IV.5)$

donc, l'accélération du point B est donnée par

$\dot{v}_B = (\ddot{x} - l.\ddot{\theta}.\cos\theta + l.\dot{\theta}^2.\sin\theta)\vec{i}$ - $(l.\ddot{\theta}.\sin\theta + l.\dot{\theta}^2.\cos\theta)\vec{j}$ $(IV.6)$

après décomposition scalaire des équations de la dynamique $(IV.1), (IV.2)$ et $(IV.3)$, on obtient :

$M\ddot{x} = u - R_x$ $(IV.7)$

$R_x = m(\ddot{x} - l\ddot{\theta}\cos\theta + l\dot{\theta}^2\sin\theta)$ $(IV.8)$

$R_y - mg = -m(l\ddot{\theta}\sin\theta + l\dot{\theta}^2\cos\theta)$ $(IV.9)$

$R_x\cos\theta + R_y\sin\theta = 0$ $(IV.10)$

Ces quatre équations décrivent respectivement le chariot en translation, le pendule en translation suivant $\vec{i}$, le pendule en translation suivant $\vec{j}$ et le pendule en rotation. On vérifie bien que le nombre de degré de liberté (x et θ) additionné au nombre de composantes des forces intérieures (ici R_x et R_y) est égal au nombre d'équations. Sous forme matricielle, ces équations s'écrivent :

$$\begin{bmatrix} M & 0 & 1 & 0 \\ -m & m.l.\cos\theta & 1 & 0 \\ 0 & m.l.\sin\theta & 0 & 1 \\ 0 & 0 & \cos\theta & \sin\theta \end{bmatrix} \cdot \begin{bmatrix} \ddot{x} \\ \ddot{\theta} \\ R_x \\ R_y \end{bmatrix} = \begin{bmatrix} u \\ m.l.\dot{\theta}^2 \sin\theta \\ m.g - m.l.\dot{\theta}^2 \cos\theta \\ 0 \end{bmatrix} \qquad (IV.11)$$

Donc

$$\begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} M & 0 & 1 & 0 \\ -m & m.l.\cos\theta & 1 & 0 \\ 0 & m.l.\sin\theta & 0 & 1 \\ 0 & 0 & \cos\theta & \sin\theta \end{bmatrix}^{-1} \cdot \begin{bmatrix} u \\ m.l.\dot{\theta}^2 \sin\theta \\ m.g - m.l.\dot{\theta}^2 \cos\theta \\ 0 \end{bmatrix} \qquad (IV.12)$$

Ce calcul se fait aisément grâce à Maple, on obtient :

$$\begin{bmatrix} \ddot{x} \\ \ddot{\theta} \end{bmatrix} = \begin{bmatrix} \dfrac{-m.\sin\theta(l.\dot{\theta}^2 - g.\cos\theta)}{M + m.\sin^2\theta} \\ \dfrac{\sin\theta((M+m).g - m.l.\dot{\theta}^2.\cos\theta)}{l.(M + m.\sin^2\theta)} \end{bmatrix} + \begin{bmatrix} \dfrac{l}{l.(M + m.\sin^2\theta)} \\ \dfrac{\cos\theta}{l.(M + m.\sin^2\theta)} \end{bmatrix} .u \qquad (IV.13)$$

Les équations d'état s'écrivent donc :

$$\frac{d}{dt}\begin{bmatrix} x \\ \dot{x} \\ \theta \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dfrac{-m.\sin\theta(l.\dot{\theta}^2 - g.\cos\theta)}{M + m.\sin^2\theta} \\ \dot{\theta} \\ \dfrac{\sin\theta((M+m).g - m.l.\dot{\theta}^2.\cos\theta)}{l.(M + m.\sin^2\theta)} \end{bmatrix} + \begin{bmatrix} 0 \\ \dfrac{l}{l.(M + m.\sin^2\theta)} \\ 0 \\ \dfrac{\cos\theta}{l.(M + m.\sin^2\theta)} \end{bmatrix} .u \qquad (IV.14)$$

IV.2.1.3.Stabilisation du pendule inverse

On considère le système modélisé à la section précédente. Rappelons que l'entrée u est la force exercée sur le chariot de masse M , x est la position du chariot et θ est l'angle entre le pendule et la verticale.

En prenant pour vecteur d'état $X = (x, \dot{x}, \theta, \dot{\theta})$ et en supposant que la position du chariot x et l'angle entre le pendule et la verticale θ sont mesurées, les équations d'état sont données par :

$$\begin{bmatrix}\dot{x}_1\\ \dot{x}_2\\ \dot{x}_3\\ \dot{x}_4\end{bmatrix}=\begin{bmatrix}x_2\\ \dfrac{m.\sin x_3(g.\cos x_3 - l.x_4{}^2)+u}{(M+m.\sin^2 x_3)}\\ x_4\\ \dfrac{\sin x_3((M+m).g - m.l.x_4{}^2\cos x_3)+\cos x_3.u}{l.(M+m.\sin^2 x_3)}\end{bmatrix} \qquad (IV.15)$$

$y1 = x1$ Position du chariot (mètres).

$y2 = x3$ Angle du pendule (radians).

IV.2.1.4.Le régulateur d'état

Le régulateur d'état est un régulateur fournissant un signal de commande $u(t)$ proportionnel à chacune des quatre variables d'état décrivant le système [93] :

$$u(t) = -K.X = -(K_x \quad K_{\dot{x}} \quad K_\theta \quad K_{\dot{\theta}})\begin{pmatrix}x(t)\\ \dot{x}(t)\\ \theta(t)\\ \dot{\theta}(t)\end{pmatrix} \qquad (IV.16)$$

Le vecteur ligne K contient les gains associés à chaque variable d'état. D'un point de vue classique, on constate que ce régulateur d'état est en fait un double régulateur proportionnel dérivé utilisé pour régler l'inclinaison du pendule avec $\theta(t)$ et $\dot{\theta}(t)$ et régler la position du chariot à l'aide de $x(t)$ et $\dot{x}(t)$.

IV.2.1.5.Critères d'optimisation pour le régulateur

La synthèse du régulateur se fait en minimisant les écarts quadratiques de l'angle et de la position longitudinale ainsi que l'énergie mise en jeu pour déplacer l'ensemble chariot balancier [93].

Ceci nous conduit à définir des coefficients de pondération pour chacune des positions et vitesses et pour le signal de commande. Les fonctions à minimiser sont alors les suivantes :

$$f_1(Q_1,R_1) = \int_0^\infty (X^T(t).Q_1.X(t) + u^t(t).R_1.u(t)).dt \qquad (IV.17)$$

avec

$$Q_1 = \begin{bmatrix} Q_x & 0 \\ 0 & Q_{\dot{x}} \end{bmatrix}, X = \begin{bmatrix} x(t) \\ \dot{x}(t) \end{bmatrix}$$

$$f_2(Q_{2,}R_2) = \int_0^\infty (\vartheta^T(t).Q_2.\vartheta(t) + u^t(t).R_2 u(t)).dt \qquad (IV.18)$$

avec

$$Q_2 = \begin{bmatrix} Q_\theta & 0 \\ 0 & Q_{\dot{\theta}} \end{bmatrix}, \vartheta = \begin{bmatrix} \theta(t) \\ \dot{\theta}(t) \end{bmatrix}$$

Le problème d'optimisation multiobjectif est défini par :

$$\begin{cases} \min f_1(Q1, R1) \\ \min f_1(Q1, R1) \\ u_{\min} \le u \le u_{\max} \end{cases} \qquad (IV.19)$$

Ou d'une façon agréger :

$$J(Q,R) = \int_0^\infty (\chi^t(t).Q.\chi(t)) + u^t(t).R.u(t).dt \qquad (IV.20)$$

avec

$$Q = \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix}, R = \begin{bmatrix} R_1 & 0 \\ 0 & R_2 \end{bmatrix}$$

Pour fixer les coefficients de pondération, on a adopté les critères suivants :

- l'amplitude du débattement angulaire n'est pas très importante.
- le déplacement du chariot sur le rail doit être limité.
- on ne se préoccupe pas de limiter les vitesses atteintes.
- le signal de commande $u(t)$ ne doit pas être trop grand.

On emploie notre algorithme génétique multiobjectif pour le calcul des paramètres du contrôleur liés aux quatre variables d'état considérons le problème d'optimisation multiobjectif défini ci-dessus sous une contrainte sur le signal de commande.

La gestion du contrainte sur le signale de commande ce fait en introduisant une fonction de pénalité aux fonctions objectives à minimiser, il s'agit de dégrader la performance des individus infaisables en fonction de leur proximité de la région faisable de l'espace de recherche.

Pour chaque élément de l'espace de recherche, sa proximité de la région faisable peut être mesurée à travers le degré de violation de chaque contrainte:

$$\delta(x) = \max(u(x), 0). \qquad (IV.21)$$

Avec $g(x) = \mathrm{abs}(u\text{-}u_{\max})$

En utilisant cette mesure d'infaisabilité de l'individu x par rapport à chaque contrainte, nous pouvons introduire la fonction de pénalité sous la forme suivante:

$$F_{penalty}(x) = c(t)\sum_{j=1}^{1} P(\delta_j(x)) \qquad (IV.22)$$

Cette fonction est rajoutée à chaque une des fonctions objectives à minimiser pour dégrader la valeur de la performance des individus qui ne respectent pas la contrainte.

IV.2.1.6.Schéma fonctionnel

Le schéma fonctionnel correspondant à la description que nous venons de voir est donné à la figure

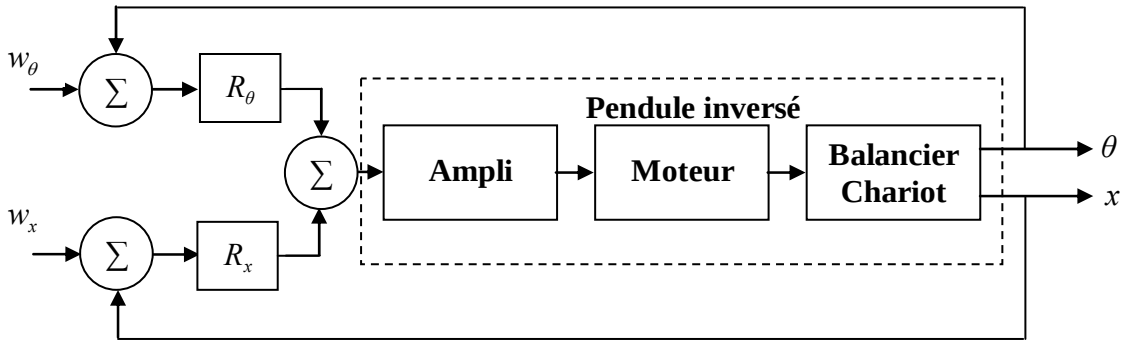

Figure $(IV.2)$: schéma fonctionnel du pendule inversé

Où w_θ , w_x sont les consignes de positions angulaire et longitudinale.

IV.2.1.7. Résolution par algorithme génétique

Afin de maintenir une diversité entre les algorithmes utilisés pour les problèmes abordés, la structure d'implémentation de l'algorithme génétique multiobjectif proposé pour le problème d'identification a été modifiée et introduit pour le problème de commande. Cette modification consiste

à trier la population initiale, générée aléatoirement en fronts, en se basant sur le concept de dominance, la sélection par tournoi est utilisée au lieu de la sélection proportionnelle.

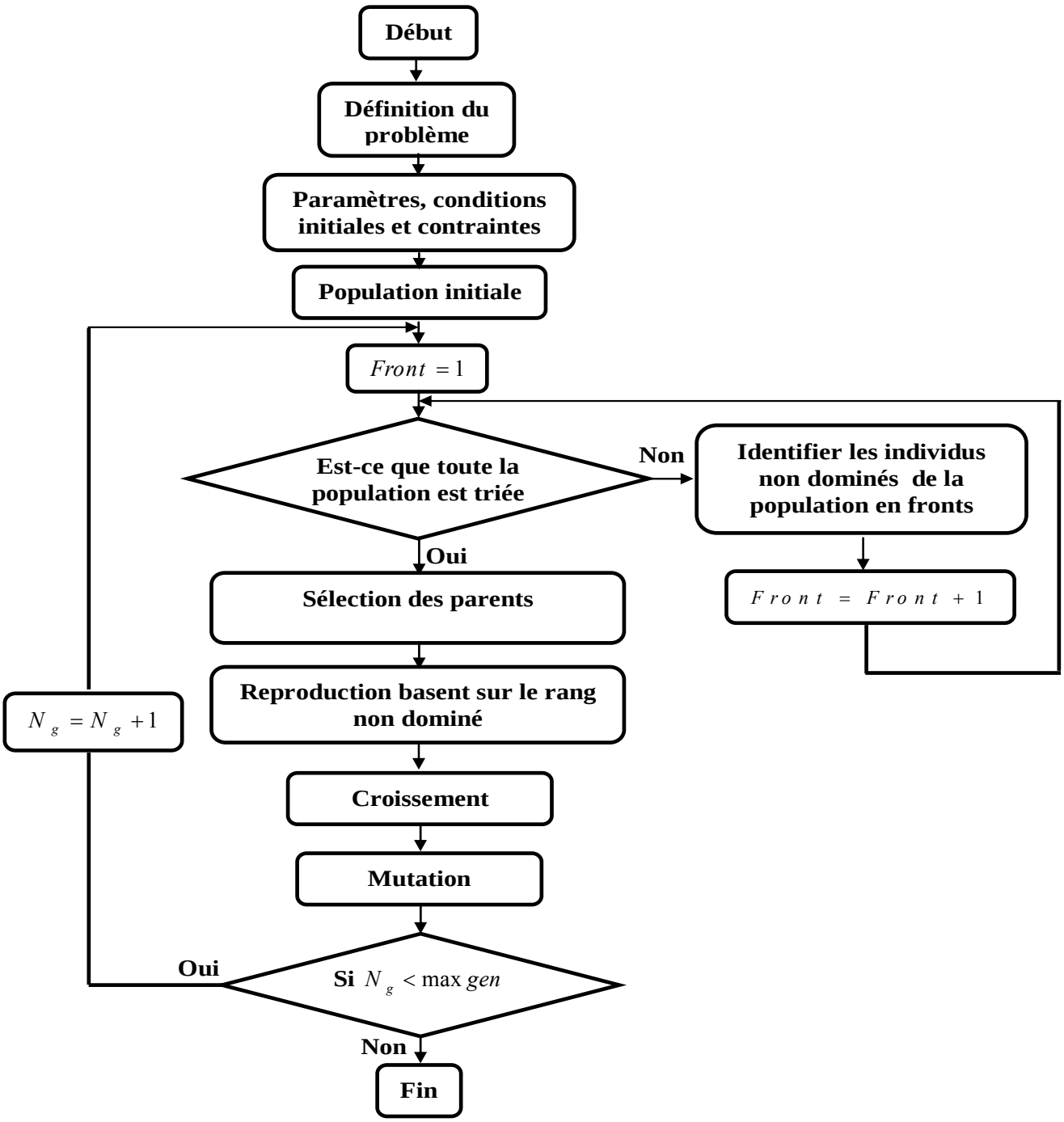

Figure $(IV.3)$: Organigramme de l'algorithme génétique multiobjectif utilisé pour la commande

La procédure à suivre est décrite par l'organigramme de la Figure $(IV.3)$. La première étape consiste à introduire toutes les données nécessaires à l'exécution (paramètres, conditions initiales et

contraintes), et la génération aléatoire de la population initiale de taille $N = 100$ sur toute l'espace de recherche des paramètres relatifs au problème qui sont :

- $[K_{p1\min}, K_{1v\min}, K_{p2\min}, K_{v2\min}] = [1,1,-40,-30]$, $[K_{p1\max}, K_{1v\max}, K_{p2\max}, K_{v2\max}] = [80,50,40,30]$ pour le pendule inversé.

- $[K_{pj\min}, K_{vj\min}, K_{ij\min}] = [0.01,0.01,0.01]$, $[K_{pj\max}, K_{vj\max}, K_{ij2\max}] = [300,200,150]$ avec $j = 1,2$ pour le robot à deux degrés . Ensuite un traitement répétitif s'opère de la manière suivante :

- Ordonnancement de la population en fronts selon les valeurs des fonctions objectives.
- Sélection des parents par tournoi;
- Création de la population enfant par l'application des opérateurs de reproduction, a savoir le croisement (le croisement en deux sites a été employé avec un taux de 0.7) et la mutation (mutation sélective avec un taux de 0.02);
- Génération de la nouvelle population par la réunion des deux populations parents et enfants de taille $\frac{N}{2}$ chacune ;

Cette procédure est répétée jusqu'à atteindre le nombre maximal de générations fixé par $\max gen = 50$.

IV.2.1.8.Résultats de Simulation du pendule inversé

Un pendule inversé correspondant à la description présentée plus haut a été simulé en se servant du logiciel MATLAB. La méthode numérique d'Euler a été utilisée avec un pas de 10^{-2} seconde lors de cette simulation. Les essais effectués ont conduit aux résultats présentés dans les figure $(IV.3)$ et $(IV.4)$. Les paramètres du régulateur obtenus par l'algorithme génétique sont donnés par le tableau suivant :

Tableau $IV.1$: ***Valeurs des paramètres du régulateur***

Variables d'état	K_p	K_v	K_p	K_v	K_p	K_v
	Sans contrainte		Avec contrainte $\lvert U_{max} \rvert \le 9$		Avec contrainte $\lvert U_{max} \rvert \le 8$	
x	**-3,52**	**-10,09**	**-4.0274**	**-20.2053**	**-3.9492**	**-20.7331**
θ	**72,49**	**23,70**	**51.5816**	**25.9071**	**45.4037**	**25.6197**

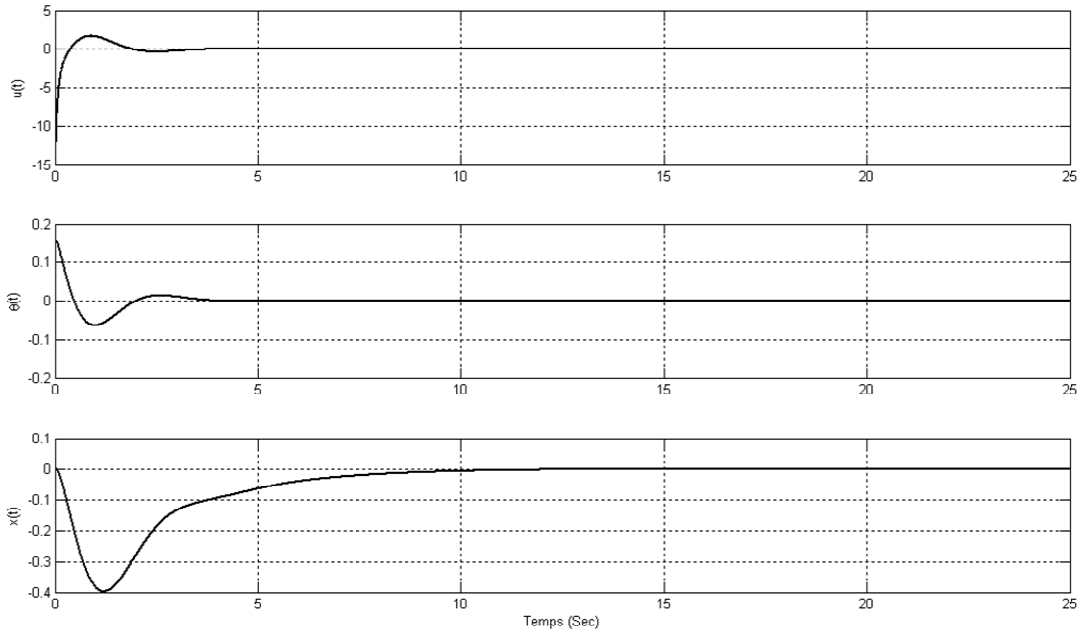

Figure (*IV* .4) : ***Résultats de simulation du pendule inversé sans contrainte sur la commande***

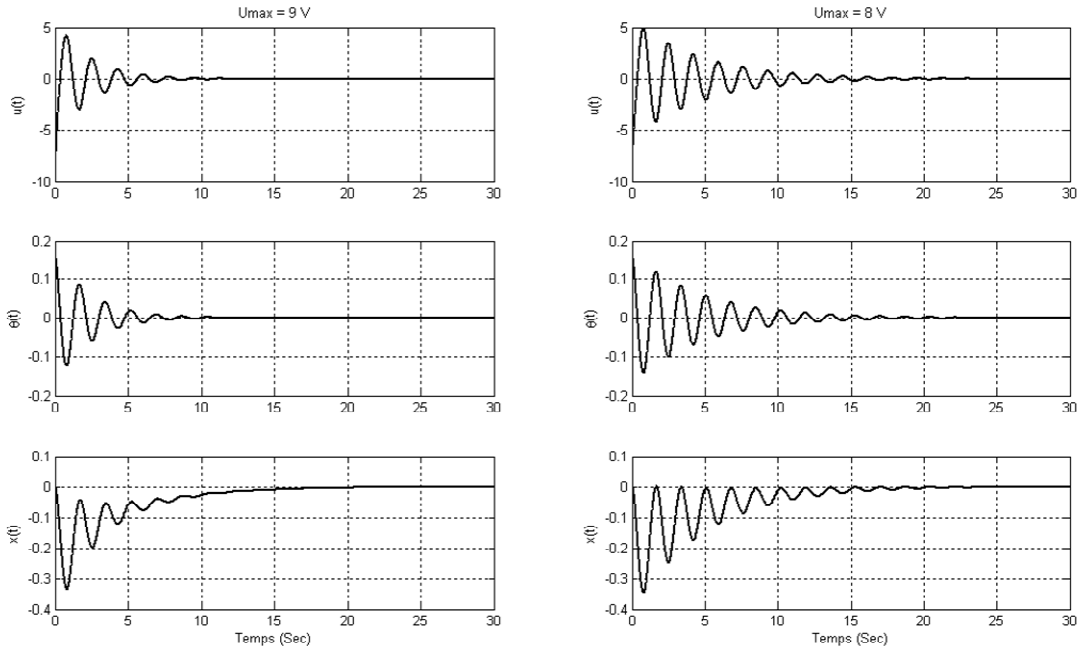

Figure (*III* .5) : ***Résultats de simulation du pendule inversé avec contrainte sur la commande***

Interprétation des résultats

Les résultats montrent que le système (pendule inversé) atteint sa stabilité en un temps minimale en appliquant un signal de commande sans contrainte, par ailleurs si on limite le signal de

commande on voit que le système prend un temps en plus pour atteindre sa stabilité. Ces résultats s'expliquent par le fait qu'avec la prise en compte de la contrainte, les paramètres du contrôleur obtenu sont modifiés de façon à satisfaire la contrainte.

Dans le tableau on observe que la prise en compte de la contrainte entraine la diminution du gain K_p liée au variable d'état θ et l'augmentation du gain K_v liée au variable d'état x en valeur absolue. Cette augmentation a pour but d'accélérer la réponse et d'améliorer la stabilité en permettant un amortissement dû à l'apparition d'une variation dans la commande

IV.2.2.1. Robot manipulateur à deux degrés de liberté

Bien que la commande de la plupart des robots industriels actuels soit encore conçue à partir de l'automatique linéaire, des méthodes plus avancées, tenant compte du caractère non linéaire des structures articulées doivent être envisagées pour les applications qui exigent de grandes performances dynamiques (vitesse, précision).

Pour cette raison, nous avons choisi un robot manipulateur à deux degrés de liberté pour l'application de la commande par découplage non linéaire ou (commande dynamique ou encore "couple calculé").

La mise en œuvre de cette méthode exige le calcul du modèle dynamique en ligne et la connaissance des valeurs numériques des paramètres inertiels et de frottements, ce qui ne constitue plus maintenant une limite rédhibitoire. Le problème de calcule en ligne est en effet résolu pratiquement grâce aux méthode de modélisations que nous aurons présenté ci-dessous et aux évolutions technologique en microinformatique.

IV.2.2.2.Méthodes de modélisation des robots

Le modèle dynamique est la relation entre les couples (et/ou forces) appliqués aux actionneurs et les positions, vitesses et accélérations articulaires [77]. On représente le modèle dynamique par une relation de la forme :

$$\Gamma = f(q, \dot{q}, \ddot{q}, f_e) \qquad (IV.23)$$

avec:

- Γ : vecteur des couples/forces des actionneurs. selon que l'articulation est rotoïde ou prismatique. Dans la suite, on écrira tout simplement couples;
- q : vecteur des positions articulaires;

- $\dot{q}$: vecteur des vitesses articulaires;
- $\ddot{q}$: vecteur des accélérations articulaires.
- f_e : vecteur représentant l'effort extérieur (forces et moments) qu'exerce le robot sur l'environnement.

On convient d'appeler modèle dynamique inverse, ou tout simplement modèle dynamique, la relation de la forme $(IV.23)$.

Le modèle dynamique direct est celui qui exprime les accélérations articulaires en fonction des positions, vitesses et couples des articulations. Il est alors représenté par la relation :

$$\ddot{q} = g(q, \dot{q}, \Gamma, f_e) \qquad (IV.24)$$

Parmi les applications du modèle dynamique, on peut citer:

- la simulation, qui utilise le modèle dynamique direct.
- le dimensionnement des actionneurs.
- l'identification des paramètres inertiels et des paramètres de frottement du robot.
- la commande, qui utilise le modèle dynamique inverse.

Les deux méthodes les plus populaires employées pour obtenir le modèle dynamique des robots sont la méthode de Newton-Euler et la méthode de Lagrange. Les équations obtenues en utilisant la formulation lagrangienne sont plus appropriées à l'application de la théorie de commande moderne que les équations récursives obtenues en utilisant la méthode de Newton-Euler. En formulation lagrangienne, le modèle dynamique est obtenu en utilisant des énergies cinétiques et potentielles liées aux corps rigides en mouvement. La dérivation est systématique et conceptuellement simple. Cette méthode donne des équations dynamiques en boucle fermée qui expriment explicitement les variables articulaires en termes de couples. Pour arriver à ces équations, on commence par un ensemble des variables articulaires généralisées $q_i \quad i = 1,..,n$; celui permet d'établir le modèle dynamique et de trouver l'énergie cinétique C et potentielle P du système [77].

IV.2.2.2.1. Formalisme de Lagrange

Le but de cette section est d'étudier la forme générale du modèle dynamique, de mettre en évidence les différents termes qui y interviennent et de déduire les propriétés caractéristiques de ces termes. La méthode présentée n'est pas celle qui donne le modèle le plus performant du point de vue

du nombre d'opérations, mais c'est la méthode la plus simple compte tenu de ces objectifs. Dans un premier temps, nous considérerons un robot idéal sans frottement sans, élasticité et ne subissant ou n'exerçant aucun effort extérieur. Le facteur de frottement sera pris en compte par la suite.

Le formalisme de Lagrange décrit les équations du mouvement en termes de travail et d'énergie du système, ce qui se traduit, lorsque l'effort extérieur sur l'organe terminal est supposé nul, par l'équation suivante :

$$\Gamma_i = \frac{d}{dt}.\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} \qquad i = 1,..,n \qquad (IV.25)$$

avec

- Γ_i : le vecteur des couples.
- L : lagrangien du système égal à $C - P$.
- C : énergie cinétique totale du système.
- P : énergie potentielle totale du système.

L'énergie cinétique du système est une fonction quadratique des vitesses articulaires :

$$E = \frac{1}{2}\dot{q}^T M \dot{q} \qquad (IV.26)$$

où M est la matrice de l'énergie cinétique, d'élément générique M_{ij}, appelée aussi matrice d'inertie du robot, qui est symétrique et définie positive. Ses éléments sont fonction des variables articulaires q.

L'énergie potentielle étant fonction des variables articulaires q, le couple Γ peut se mettre, à partir des équations $(IV.25)$ et $(IV.26)$, sous la forme:

$$\Gamma = M(q).\ddot{q} + C(q,\dot{q}).\dot{q} + G(q) \qquad (IV.27)$$

avec

- $C(q,\dot{q})\dot{q}$: vecteur de dimension $(n,1)$ représentant les forces de Coriolis et des forces centrifuges, tel que :

$$C\dot{q} = \dot{M}\dot{q} - \frac{\partial E}{\partial q} \qquad (IV.28)$$

- $G = [G_1 ... G_n]^T$: vecteur des forces de gravité, tel que

$$G_i = \frac{\partial U}{\partial q_i} \qquad (IV.29)$$

Les éléments de M, C et G sont fonction des paramètres géométriques et inertiels du mécanisme. Les équations dynamiques d'un système mécanique articulé forment donc un système de n équations différentielles du second ordre, couplées et non linéaires.

IV.2.2.2.2. Formalisme de Newton-Euler

La méthode de Luh, Walker et Paul [94], considérée comme une avancée important vers la possibilité de calculer le modèle dynamique des robots en ligne, utilise ces équations et est fondée sur une double récurrence. La récurrence avant de la base du robot vers l'effecteur, calcule successivement les vitesses et accélérations des corps, puis leur torseur dynamique. Une récurrence arrière, de l'effecteur vers la base, permet le calcul des couples des actionneurs en exprimant pour chaque corps le bilan des efforts, cette méthode permet d'obtenir directement le modèle dynamique inverse.

Dans cette partie, nous présentons un algorithme de Newton-Euler fondé sur la double récurrence de la méthode de Luh et al. [94].

Pour utiliser pratiquement l'algorithme de Newton-Euler exposé ci-dessus- il faut projeter dans un même repère les vecteurs et tenseurs qui apparaissent dans une même équation. Nous reprenons ici le choix de Luh et al. [94] qui consiste à projeter les grandeurs relatives à un corps dans le repère qui lui est lié. Les équations de Newton-Euler s'écrivent :

a) **<u>Récurrence</u> <u>avant</u>** elle permet de calculer la résultante des forces ${}^{i}F_i$ et le moment des efforts extérieur ${}^{i}M_i$ exercés sur le corps C_i. Les équations de Récurrence avant s'écrivent :

$${}^{i+1}\omega_{i+1} = {}^{i+1}R_i \, {}^{i}\omega_i + \dot{q}_{i+1} \, {}^{i+1}\hat{Z}_{i+1} \qquad (IV.30)$$

$${}^{i+1}\dot{\omega}_{i+1} = {}^{i+1}R_i \, {}^{i}\dot{\omega}_i + {}^{i+1}R_i \, {}^{i}\omega_i \times \dot{q}_{i+1} \, {}^{i+1}\hat{Z}_{i+1} + \ddot{q}_{i+1} \, {}^{i+1}\hat{Z}_{i+1} \qquad (IV.31)$$

$${}^{i+1}\dot{v}_{i+1} = {}^{i+1}R_i({}^{i}\dot{\omega}_i \times {}^{i}P_{i+1} + {}^{i}\omega_i \times ({}^{i}\omega_i \times {}^{i}P_{i+1}) + {}^{i}\dot{v}_i) \qquad (IV.32)$$

$${}^{i+1}\dot{v}_{i+1} = {}^{i+1}\omega_{i+1} \times {}^{i+1}P_{ci+1} + {}^{i}\dot{v}_i) + {}^{i+1}\omega_{i+1} \times ({}^{i+1}\omega_{i+1} \times {}^{i+1}P_{ci+1}) + {}^{i+1}\dot{v}_{i+1} \qquad (IV.33)$$

$${}^{i+1}F_{i+1} = m_{i+1} \, {}^{i+1}\dot{v}_{c_{i+1}} \qquad (IV.34)$$

$$^{i+1}F_{i+1} = {}^{i+1}Ic_{i+1}\,{}^{i+1}\dot{\omega}_{i+1} + {}^{i+1}\omega_{i+1} \times {}^{i+1}Ic_{i+1}\,{}^{i+1}\omega_{i+1} \qquad (IV.35)$$

$$^{i+1}M_{i+1} = {}^{i+1}Ic_{i+1}\,{}^{i+1}\dot{\omega}_{i+1} + {}^{i+1}\omega_{i+1} \times {}^{i+1}Ic_{i+1}\,{}^{i+1}\omega_{i+1} \qquad (IV.36)$$

b) **Récurrence arrière** Les équations composant la récurrence arrière sont obtenues à partir du bilan des efforts sur chaque corps, écrit à l'origine. On obtient :

$$^{i}f_i = {}^{i}R_{i+1}\,{}^{i+1}f_{i+1} + {}^{i}F_i \qquad (IV.37)$$

$$^{i}m_i = {}^{i}M_i\,{}^{i}R_{i+1}\,{}^{i+1}m_{i+1} + {}^{i}P_{c_i} \times {}^{i}F_i + {}^{i+1}P_{i+1} \times {}^{i}R_{i+1}\,{}^{i}f_{i+1} \qquad (IV.38)$$

$$\Gamma_i = {}^{i}m_i^{T}\,{}^{i}\hat{Z}_i \qquad (IV.39)$$

IV.2.2.3. Prise en compte des frottements

De nombreuses études ont été réalisées afin de mieux analyser les frottements au niveau des articulations, des réducteurs et des transmissions. Les frottements non compensés provoquent en effet des erreurs statiques, des retards et des cycles limites. Différents modèles de frottement ont été proposés dans la littérature. Citons par exemple les travaux de [95], [53], [64].

Le modèle du type frottement sec (ou de Coulomb) fait l'hypothèse d'un couple constant de frottement en opposition au mouvement. Au démarrage (vitesse nulle), un couple supérieur au couple de frottement sec doit être développé pour amorcer le mouvement. Si ce modèle de frottements secs fait l'unanimité, il n'en est pas de même pour représenter la dépendance des autres types de frottements avec la vitesse. Des tests expérimentaux [95] ont prouvé que le couple de frottement au démarrage décroît exponentiellement à faible vitesse, puis croît ensuite proportionnellement à la vitesse. Le modèle correspondant est donné par :

$$\Gamma_{fi} = F_{si}sign(\dot{q}_i) + F_{vi}\dot{q}_i + F_{ei}e^{-|\dot{q}_i|B_i}sign(\dot{q}_i) \qquad (IV.40)$$

Dans cette expression, le couple des frottements de l'articulation i est noté Γ_{fi}; F_{si} et F_{vi} désignent respectivement les paramètres de frottement sec et visqueux de l'articulation i ; le couple statique de démarrage est égal à $(F_{si} + F_{ei})sign(\dot{q}_i)$.

Dans bon nombre d'applications, l'expression du couple de frottement est donné par :

$$\Gamma_{fi} = F_{si}sign(\dot{q}_i) + F_{vi}\dot{q}_i \qquad (IV.41)$$

On peut donc tenir compte des forces et des couples de frottements en ajoutant au deuxième membre de l'expression $(IV.23)$ le vecteur Γ_f tel que:

$$\Gamma_f = diag(\dot{q})F_S + diag[sign(\dot{q})]F_V \qquad (IV.42)$$

avec:

$Fs = [F_{s1}...F_{sn}]^T$

et $F_V = [F_{V1}...F_{Vn}]^T$

IV.2.2.4.Modèle dynamique direct

Pour simuler le comportement du robot et de sa boucle de commande, on utilise le modèle dynamique direct. Même si les contraintes de temps de calcul sont moins critiques que pour la commande, il est cependant intéressant de disposer d'un modèle performant. Nous présentons ici l'une de ces méthodes qui procède par inversion de la matrice d'inertie.

IV.2.2.4.1.Calcul par inversion de la matrice d'inertie du robot

A partir de l'équation $(IV.27)$ et en supposant l'existence d'un effort extérieur sur l'organe terminal, le modèle dynamique direct s'écrit :

$$\ddot{q} = M(q)^{-1}(\Gamma - \psi(q,\dot{q})) \qquad (IV.43)$$

avec :

$$\psi(q,\dot{q}) = C(q,q)q + G(q) + J^T fe \qquad (IV.44)$$

Cette forme se prête bien à l'intégration numérique. Pour réaliser une telle simulation, il faut donc calculer le vecteur $\psi(q,\dot{q})$ et la matrice $M(q)$. Le calcul du vecteur $\psi(q,\dot{q})$ peut être effectué par l'algorithme de Newton-Euler, en remarquant qu'il est égal à Γ si $\ddot{q} = 0$.

Le calcul de la matrice $M(q)$ par une méthode lagrangienne peut être moins coûteux en temps [90], On peut cependant utiliser l'algorithme de Newton-Euler pour calculer ses éléments en remarquant, à partir de la relation $(IV.27)$, que la $i^{ème}$ colonne est égale à Γ si:

$$\dot{q} = 0, g = 0, \ddot{q} = u_i, f_e = 0, m_e = 0 \qquad (IV.45)$$

où u_i étant le vecteur *(nxl)* dont tous les éléments sont nuls sauf la $i^{ème}$ composante qui est égale à 1.

Pour améliorer les performances d'un tel algorithme, on peut utiliser la technique symbolique itérative en écrivant un sous-programme pour le calcul de chaque colonne de la matrice $M(q)$ et en tenant compte de sa symétrie.

A partir de la relation $(IV.43)$, on déduit une forme possible des équations d'état du modèle dynamique comme étant :

$$\begin{bmatrix}\dot{q}\\ \ddot{q}\end{bmatrix}=\begin{bmatrix}\dot{q}\\ M(q)^{-1}(\Gamma-\Psi(q,\dot{q})\end{bmatrix} \qquad (IV.46)$$

et

$$y=q \text{ ou } y=X \qquad (IV.47)$$

Dans cette formulation, le vecteur des variables d'état est $[q^T\ \dot{q}^T]^T$ tandis que y désigne la sortie soit dans l'espace articulaire soit dans l'espace opérationnel.

IV.3. Modélisation d'un Robot manipulateur à deux degrés de liberté

IV.2.3.1.Application de la méthode de Lagrange

On considère un robot planaire avec deux degrés de liberté, montrés dans la figure $(IV.6)$.Pour la simplicité, on suppose que les masses m_1 et m_2 des deux corps sont représentées par les masses de point à la fin des corps. Les longueurs des corps sont l_1 et l_2, respectivement. Le vecteur des variables articulaires généralisées est q_1 et q_2. On sait que l'énergie cinétique d'une masse m se déplaçant à une vitesse linéaire v est donnée par $C=\frac{1}{2}.m.v^2$ et l'énergie potentielle liée à une masse m située à un hauteur h dans un champ de gravité est donnée par $P=m.g.h$, où g est la constante de la gravité [96].

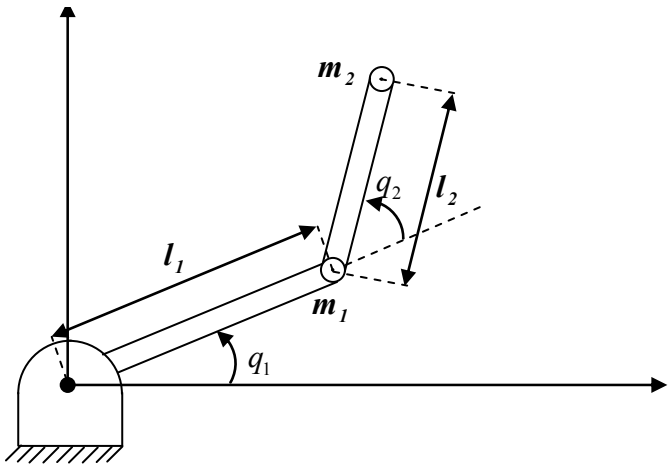

Figure *(IV .6)* : Robot manipulateur à deux degrés de liberté

Donc l'énergie cinétique C_1 pour la masse m_1 est donné par :

$$x_1 = l_1.\cos(q_1). \tag{IV.48}$$

$$y_1 = l_1.\sin(q_1). \tag{IV.49}$$

$$v_1^{\,2} = \dot{x}_1^{\,2} + \dot{y}_1^{\,2} \tag{IV.50}$$

$$C_1 = \frac{1}{2}.m_1.l_1.\dot{q}_1^{\,2} \tag{IV.51}$$

De même, l'énergie cinétique C_2 pour la masse m_2 est donnée par

$$C_2 = \frac{1}{2}.m_2.l_2.v_2^{\,2} \tag{IV.52}$$

$$v_2^{\,2} = \dot{x}_2^{\,2} + \dot{y}_2^{\,2} \tag{IV.53}$$

De la figure $(IV.5)$ on a :

$$x_2 = l_1.\cos(q_1) + l_2.\cos(q_1 + q_2) \tag{IV.54}$$

$$y_2 = l_1.\sin(q_1) + l_2.\sin(q_1 + q_2) \tag{IV.55}$$

$$v_2^{\,2} = l_1^{\,2}\dot{q}_1^{\,2} + l_2^{\,2}(\dot{q}_1 + \dot{q}_2)2 + 2.l_1.l_2.\cos(q_2)(\dot{q}_1^{\,2} + \dot{q}_1\dot{q}_2) \tag{IV.56}$$

Les énergies potentielles pour les masses m_1 et m_2 sont données par :

$$P_1 = m_1.g.l_1.\sin(q_1) \tag{IV.57}$$

$$P_2 = m_2.g.(l_1\sin(q_1) + l_2.\sin(q_1 + q_2)) \tag{IV.58}$$

La prochaine étape est de former le lagrangien :

$$L = \sum_{i=1}^{2} C_i - P_i \tag{IV.59}$$

Le modèle dynamique du robot est obtenu en utilisant l'équation $(IV.25)$:

$$\Gamma_1 = \frac{d}{dt}(\frac{\partial L}{\partial \dot{q}_1}) - \frac{\partial L}{\partial q_1} \tag{IV.60}$$

$$\Gamma_2 = \frac{d}{dt}(\frac{\partial L}{\partial \dot{q}_2}) - \frac{\partial L}{\partial q_2} \tag{IV.61}$$

où Γ_1 et Γ_2 sont les couples généralisées.

On développe les équations $(IV.60)$ $(IV.61)$, on peut se mettre le modèle dynamique de ce robot, à partir de l'équation $(IV.27)$ et en tenant compte le couple de frottement présenter par l'équation $(IV.40)$, sous la forme matricielle suivante **:**

$$\Gamma = M(q).\ddot{q} + C(q,\dot{q}).\dot{q} + G(q) + H(\dot{q}) \qquad (IV.62)$$

$\Gamma(n,1)$: vecteur des couples généralisés

$q(n,1)$: vecteur des variables articulaires généralisées.

$\dot{q}(n,1)$: vecteur des vitesses angulaires.

$\ddot{q}(n,1)$: vecteur des accélérations.

$M(q)$: matrice d'inertie de dimension (n,n) symétrique et définie positive.

$C(q,\dot{q})\dot{q}$: vecteur de dimension $(n,1)$ spécifiant l'effet de centrifuge et de Coriolis.

$G(q)$: vecteur de dimension $(n,1)$ exprimant l'effet gravitationnel.

$H(\dot{q})$: vecteur de dimension $(n,1)$ exprimant les frottements.

n: Degré de liberté (nombre d'articulations).

avec :

$$M(q) = \begin{bmatrix} l_2^{\,2}.m_2 + 2.l_1 l_2.m_2.\cos(q2) + l_1^{\,2}(m_1+m_2) & l_2^{\,2}.m_2 + l_1 l_2 m_2.\cos(q_2) \\ l_2^{\,2}.m_2 + l_1 l_2 m_2.\cos(q_2) & l_2^{\,2}.m_2 \end{bmatrix}$$

$$N(q,\dot{q}) = C((q,\dot{q}).\dot{q}) = \begin{bmatrix} -m_2.l_1.l_2.\sin(q_2).\dot{q}_2^{\,2} - 2.m_2.l_1 l_2.\sin(q_2).(\dot{q}_1.\dot{q}_2) \\ m_2.l_1.l_2.\sin(q_2).\dot{q}_1^{\,2} \end{bmatrix}$$

$$G(q) = \begin{bmatrix} m_2.l_2.g.\cos(q_1+q_2) + (m_1+m_2).l_1.g.\cos(q_1) \\ m_2.l_2.g.\cos(q_1+q_2) \end{bmatrix}$$

En tenant compte le couple de frottement exposé en équation $(IV.40)$, on peut écrire:

$$H(\dot{q}) = \begin{bmatrix} V_1.\dot{q}_1 + V_2.\mathrm{sgn}(\dot{q}_1) \\ V_3.\dot{q}_2 + V_4.\mathrm{sgn}(\dot{q}2) \end{bmatrix}$$

Noter que la matrice d'inertie $M(q)$ est une fonction de la position q seulement. En général, la matrice d'inertie est symétrique et définie positif et ainsi inversible. Les éléments diagonaux de cette matrice représentent les inerties pertinentes aux articulations respectives, alors que les éléments hors diagonale représentent les inerties de couplage. Par exemple, le terme $m_2 l_2^2$ représente l'inertie du corps 2, et le terme $l_2^2.m_2 + l_1 l_2 m_2.\cos(q_2)$ représente le couplage des inerties entre le corps 1 et 2, c.-à-d., l'effet de l'accélération du corps 1 sur le corps 2.

La matrice $N(q,\dot{q})$ contient toutes les termes liées aux forces de centrifuge et de Coriolis. Les termes qui dépendent du carré de la vitesse des corps sont les forces centrifuges. Alors que les termes qui contiennent le produit des vitesses des corps sont des forces de Coriolis. Dans notre cas d'étude, le terme $-m_2.l_1.l_2.\sin(q_2).\dot{q}_2^{\,2}$ représente la force de centrifuge agissant au corps 1 due à la vitesse du corps 2. De même, le terme $m_2.l_1.l_2.\sin(q_2).\dot{q}_1^{\,2}$ représente la force de centrifuge agissant au corps 2 due à la vitesse du corps 1. Le terme $-2.m_2.l_1 l_2.\sin(q_2).(\dot{q}_1.\dot{q}_2)$ représente la force de Coriolis agissant au corps 2 due aux vitesses des deux corps 1 et 2.

Le vecteur $G(q)$ comporte tous les termes contenant la constante de la gravité g. Noter que ces termes dépendent seulement de la position du bras dans le champ de la gravité. Si le bras fonctionne dans l'environnement pesanteur libre, alors ces termes deviennent zéro.

Le vecteur $H(\dot{q})$ comporte les forces de frottement agissant sur le robot, où V_1, V_3 sont les coefficients des frottements visqueux et V_2, V_4 sont les coefficients des frottements secs.

IV.2.3.2.Application de la méthode de Newton-Euler

Ici nous calculons le modèle dynamique en boucle fermée pour le robot manipulateur montré dans la figure $(IV.5)$.Pour la simplicité, nous supposons que la distribution de masse est extrêmement simple : Toute les masses existe comme un point à l'extrémité de chaque corps et par conséquence le tenseur d'inertie écrit au centre de la masse pour chaque corps est nul [97].

Il n'y a aucune force agissant sur le terminal, et ainsi nous avons

$$f_3 = 0\,, \qquad (IV.63)$$

$$m_3 = 0\,. \qquad (IV.64)$$

La base du robot ne tourne pas, et par conséquent nous avons

$$\omega_0 = 0\,, \qquad (IV.65)$$

$$\dot{\omega}_0 = 0\,. \qquad (IV.66)$$

Pour inclure la force de gravité nous emploierons

$$^0\dot{v}_0 = g\hat{Y}_0\,. \qquad (IV.67)$$

Les matrices de transformations entre repère sont donnent par:

$$^iR_{i+1} = \begin{bmatrix} c_{i+1} & -s_{i+1} & 0 \\ s_{i+1} & c_{i+1} & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (IV.68)$$

$$^{i+1}R_i = \begin{bmatrix} c_{i+1} & s_{i+1} & 0 \\ -s_{i+1} & c_{i+1} & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad (IV.69)$$

Nous appliquons maintenant les équations de $(IV.30)$ jusqu'a $(IV.39)$ on obtient:

La récurrence avant pour première articulation donne:

$$^1\omega_1 = \dot{q}_1\,{}^1\hat{Z}_1 = \begin{bmatrix} 0 \\ 0 \\ \dot{q}_1 \end{bmatrix} \qquad (IV.70)$$

$$^1\dot{\omega}_1 = \ddot{q}_1\,{}^1\hat{Z}_1 = \begin{bmatrix} 0 \\ 0 \\ \ddot{q}_1 \end{bmatrix} \qquad (IV.71)$$

$$^1\dot{v}_1 = \begin{bmatrix} c_1 & s_1 & 0 \\ -s_1 & c_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 0 \\ g \\ 0 \end{bmatrix} = \begin{bmatrix} gs_1 \\ gc_1 \\ 0 \end{bmatrix} \qquad (IV.72)$$

$$^1\dot{v}c_1 = \begin{bmatrix} 0 \\ l_1\ddot{q}_1 \\ 0 \end{bmatrix} + \begin{bmatrix} -l_1\dot{q}_1^{\,2} \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} gs_1 \\ gc_1 \\ 0 \end{bmatrix} = \begin{bmatrix} -l_1\dot{q}_1^{\,2} + gs_1 \\ l_1\ddot{q}_1 + gc_1 \\ 0 \end{bmatrix} \qquad (IV.73)$$

$$ {}^{1}F_{1} = \begin{bmatrix} -m_1 l_1 \dot{q}_1^{\,2} + m_1 g s_1 \\ m_1 l_1 \ddot{q}_1 + m_1 g c_1 \\ 0 \end{bmatrix} \qquad (IV.74) $$

$$ {}^{1}M_{1} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \qquad (IV.75) $$

La récurrence avant pour la deuxième articulation donne

$$ {}^{2}\omega_{2} = \begin{bmatrix} 0 \\ 0 \\ \dot{q}_1 + \dot{q}_2 \end{bmatrix} \qquad (IV.76) $$

$$ {}^{2}\dot{\omega}_{2} = \begin{bmatrix} 0 \\ 0 \\ \ddot{q}_1 + \ddot{q}_2 \end{bmatrix} \qquad (IV.77) $$

$$ {}^{2}\dot{v}_{2} = \begin{bmatrix} c_2 & s_2 & 0 \\ -s_2 & c_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -l_1 \dot{q}_1^{\,2} + g s_1 \\ l_1 \ddot{q}_1 + g c_1 \\ 0 \end{bmatrix} = \begin{bmatrix} l_1 \ddot{q}_1 s_2 - l_1 \dot{q}_1^{\,2} c_2 + g s_{12} \\ l_1 \ddot{q}_1 c_2 + l_1 \dot{q}_1^{\,2} g c_{12} \\ 0 \end{bmatrix} \qquad (IV.78) $$

$$ {}^{2}\dot{v}c_{2} = \begin{bmatrix} 0 \\ l_2(\ddot{q}_1 + \ddot{q}_2) \\ 0 \end{bmatrix} \begin{bmatrix} -l_1(\dot{q}_1 + \dot{q}_2)^2 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} l_1 \ddot{q}_1 s_2 - l_1 \dot{q}_1^{\,2} c_2 + g s_{12} \\ l_1 \ddot{q}_1 c_2 + l_1 \dot{q}_1^{\,2} g c_{12} \\ 0 \end{bmatrix} \qquad (IV.79) $$

$$ {}^{2}F_{2} = \begin{bmatrix} m_2 l_1 \ddot{q}_1 s_2 - m_2 l_1 \dot{q}_1^{\,2} c_2 + m_2 g s_{12} - m_2 l_2 (\dot{q}_1 + \dot{q}_2)^2 \\ m_2 l_1 \ddot{q}_1 c_2 + m_2 l_1 \dot{q}_1^{\,2} s_2 + m_2 g c_{12} + m_2 l_2 (\ddot{q}_1 + \ddot{q}_2) \\ 0 \end{bmatrix} \qquad (IV.80) $$

$$ {}^{2}M_{2} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \qquad (IV.81) $$

La récurrence arrière pour la deuxième articulation donne

$$^{2}f_{2} = {}^{2}F_{2} \tag{IV.82}$$

$$^{2}m_{2} = \begin{bmatrix} 0 \\ 0 \\ m_2 l_1 l_2 c_2 \ddot{q}_1 + m_2 l_1 l_2 s_2 \dot{q}_1^{\,2} + m_2 l_2 g c_{12} + m_2 l_2^{\,2}(\ddot{q}_1 + \ddot{q}_2) \end{bmatrix} \tag{IV.83}$$

La récurrence arrière pour la première articulation donne

$$^{1}f_{1} = \begin{bmatrix} c_2 & -s_2 & 0 \\ s_2 & c_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} m_2 l_1 \ddot{q}_1 s_2 - m_2 l_1 \dot{q}_1^{\,2} c_2 + m_2 g s_{12} - m_2 l_2 (\dot{q}_1 + \dot{q}_2)^2 \\ m_2 l_1 \ddot{q}_1 c_2 + m_2 l_1 \dot{q}_1^{\,2} s_2 + m_2 g c_{12} + m_2 l_2 (\ddot{q}_1 + \ddot{q}_2) \\ 0 \end{bmatrix} + \begin{bmatrix} -m_1 l_1 \dot{q}_1^{\,2} + m_1 g s_1 \\ m_1 l_1 \ddot{q}_1 + m_1 g c_1 \\ 0 \end{bmatrix} \tag{IV.84}$$

$$\begin{aligned} ^{1}m_{1} = & \begin{bmatrix} 0 \\ 0 \\ m_2 l_1 l_2 c_2 \ddot{q}_1 + m_2 l_1 l_2 s_2 \dot{q}_1^{\,2} + m_2 l_2 g c_{12} + m_2 l_2^{\,2}(\ddot{q}_1 + \ddot{q}_2) \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ m_1 l_1^{\,2} \ddot{q}_1 + m_1 l_1 g c_1 \end{bmatrix} \\ & + \begin{bmatrix} 0 \\ 0 \\ m_2 l_1^{\,2} \ddot{q}_1 - m_2 l_1 l_2 s_2 (\dot{q}_1 + \dot{q}_2)^2 + m_2 l_1 g s_2 s_{12} + m_2 l_1 l_2 c_2 ((\ddot{q}_1 + \ddot{q}_2) + m_2 l_1 g c_2 c_{12} \end{bmatrix} \end{aligned} \tag{IV.85}$$

Extrayant les composants $\hat{Z}$ de $^{i}m_{i}$ nous trouvons les couples appliqué à chaque articulation qui donné par:

$$\begin{aligned} \Gamma_1 = & m_2 l_2^{\,2}(\ddot{q}_1 + \ddot{q}_2) + m_2 l_1 l_2 c_2 (2\ddot{q}_1 + \ddot{q}_2) + (m_1 + m_2) l_1^{\,2} \ddot{q}_1 - m_2 l_1 l_2 s_2 \dot{q}_2^{\,2} - 2 m_2 l_1 l_2 s_2 \dot{q}_1 \dot{q}_2 \\ & + m_2 l_2 g c_{12} + (m_1 + m_2) l_1 g c_1 \end{aligned} \tag{IV.86}$$

$$\Gamma_2 = m_2 l_1 l_2 c_2 \ddot{q}_1 + m_2 l_1 l_2 s_2 \dot{q}_1^{\,2} + m_2 l_2 g c_{12} + m_2 l_2^{\,2}(\ddot{q}_1 + \ddot{q}_2) \tag{IV.87}$$

IV.2.3.3. Formulation des équations dans l'espace d'état

Quand les équations de Newton-Euler sont évaluées symboliquement pour n'importe quel manipulateur, elles donnent un modèle dynamique qui peut être écrite sous la forme matricielle de l'équation $(IV.62)$.

Pour le manipulateur de la section précédente, la matrice d'inertie $M(q)$ se compose de tous les termes qui dépendent de la position q seulement. Par conséquent nous avons :

$$M(q)=\begin{bmatrix} l_2^{\,2}m_2+2l_1l_2m_2\cos(q2)+l_1^{\,2}(m_1+m_2) & l_2^{\,2}m_2+l_1l_2m_2\cos(q_2) \\ l_2^{\,2}m_2+l_1l_2m_2\cos(q_2) & l_2^{\,2}m_2 \end{bmatrix} \qquad (IV.88)$$

La matrice $N(q,\dot{q})$ est obtient en regroupant toutes les termes qui dépendent du carré de la vitesse (forces centrifuges) et les termes qui contiennent le produit des vitesses (forces de Coriolis), on obtient ainsi:

$$N(q,\dot{q})=C((q,\dot{q})\dot{q})=\begin{bmatrix} -m_2l_1l_2\sin(q_2)\dot{q}_2^{\,2}-2m_2l_1l_2\sin(q_2)(\dot{q}_1.\dot{q}_2) \\ m_2l_1l_2\sin(q_2)\dot{q}_1^{\,2} \end{bmatrix} \qquad (IV.89)$$

Le vecteur $G(q)$ dépend de tous les termes contenant la constante de la gravité g , ce qui donne :

$$G(q)=\begin{bmatrix} m_2l_2g\cos(q_1+q_2)+(m_1+m_2)l_1g\cos(q_1) \\ m_2l_2g\cos(q_1+q_2) \end{bmatrix} \qquad (IV.90)$$

On tenant compte de l'équation $(IV.40)$, en peut approcher le vecteur des forces de frottement $H(\dot{q})$ qui agissant sur le robot par :

$$H(\dot{q})=\begin{bmatrix} V_1.\dot{q}_1+V_2.\mathrm{sgn}(\dot{q}_1) \\ V_3.\dot{q}_2+V_4.\mathrm{sgn}(\dot{q}2) \end{bmatrix} \qquad (IV.91)$$

où V_1,V_3 sont les coefficients des frottements visqueux et V_2,V_4 sont les coefficients des frottements secs.

IV.4.Commande non-linéaire des robots

Puisque le champ de la théorie de commande non-linéaire est grand, nous devons limiter notre attention à une méthode qui paraître bien convenu aux robots manipulateurs. En conséquence, le but principal du chapitre sera une méthode particulière appelée la méthode de calcul du couple, on présentera également une méthode d'analyse de stabilité des systèmes non-linéaires connus sous le nom de méthode de Lyapunov.

Pour commencer notre discussion des techniques non-linéaires pour commander les manipulateurs, nous revenons encore au robot manipulateur modélisé ci-dessus.

IV.4.1. Loi de commande

Plusieurs algorithmes de commande qui incorporent la dynamique ont été développés. Beaucoup de ces derniers sont des variations de la méthode de calcul du couple qui est semblable à la méthode de linéarisation par feedback employée pour la commande des systèmes non-linéaires [98].

Dans la méthode de calcul du couple cité ci-dessus, les couples requis pour l'entrée sont calculés comme suit :

$$\Gamma = \hat{M}(q)(\ddot{q}_d + K_p(q_d - q) + K_i\int(q_d - q) + K_v(\dot{q}d - \dot{q})) + \hat{N}(q,\dot{q}) + \hat{G}(q) + \hat{H}(\dot{q}) \qquad (IV.92)$$

où le K_p, K_i, et K_v sont des matrices diagonales avec les éléments diagonaux représentant les gains proportionnelles, intégrales, et dérivés respectivement. Si le couple est choisi comme entrée dans l'équation $(IV.61)$, et supposant que le modèle est précis, c-à-d, $\hat{M}(q) = M(q), \hat{N}(q,\dot{q}) = N(q,\dot{q})$, $\hat{G}(q) = G(q)$ et $\hat{H}(\dot{q}) = H(\dot{q})$, nous obtenons:

$$M(q)((\ddot{q}_d - \ddot{q}) + K_p(q_d - q) + K_i\int(q_d - q) + K_v(qd - q)) = 0 \qquad (IV.93)$$

Puisque la matrice d'inertie $M(q)$ est non singulière, nous obtenons:

$$\ddot{E} + K_p.E + K_i\int E + K_v\dot{E} = 0 \qquad (IV.94)$$

Ce qui représente un ensemble d'équations découplées où l'erreur $E = q_d - q$. Si nous choisissons les valeurs du K_p, K_i et K_v tels que l'équation caractéristique $(IV.94)$ avoir des racines a partie réelle négative alors l'erreur E approche de zéro asymptotiquement.

L'efficacité de cet algorithme dépend fortement de deux facteurs :

- l'exactitude du modèle.
- la capacité pour calculer les matrices de coefficient des équations du mouvement en temps réel.

Si le modèle n'est pas une représentation exacte du système, l'équation $(IV.94)$ devient :

$$\ddot{E} + K_p.E + K_i\int E + K_v\dot{E} = R(\ddot{q},\dot{q},q) \qquad (IV.95)$$

où $R(\ddot{q},\dot{q},q)$ est la disparité entre le modèle et la dynamique réelle du robot. Ceci est donné par :

$$R(\ddot{q},\dot{q},q) = \hat{M}^{-1}(q).((M(q) - \hat{M}(q))\ddot{q} + (N(q,\dot{q}) - \hat{N}(q,\dot{q})) + G(q) - \hat{G}(q) + H(\dot{q}) - \hat{H}(\dot{q})) \quad (IV.96)$$

Voir que si le modèle est une représentation exacte, l'équation $(IV.95)$ se ramène à l'équation $(IV.94)$ et la convergence de q à q_d peut être garantie.

Même si le modèle est précis, la capacité pour calculer la dynamique en temps réel est un problème toujours. On l'estime que le manipulateur a besoin de 2000 additions et 1500 multiplications pour calculer tous les couples. Une manière de surmonter ce problème est d'employer une loi de commande où le modèle est en dehors de la boucle de feedback [97], dans ce cas-ci, les couples désirés sont calculés à priori en utilisant le modèle donné dans l'équation $(IV.62)$ comme suite :

$$\hat{\Gamma} = \hat{M}(q_d)\ddot{q}_d + N(q_d, \dot{q}_d) + G(q_d) + H(\dot{q}_d) \qquad (IV.97)$$

Alors de la figure $(IV.5)$, nous obtenons :

$$M(q)\ddot{q} + N(q, \dot{q}) + G(q) + H(\dot{q}) = \hat{\Gamma} + K_p(q_d - q) + K_i \int (q_d - q) + K_v(\dot{q}_d - \dot{q}) \qquad (IV.98)$$

Si la disparité entre le modèle et la dynamique réelle du robot est petite, alors nous obtenons :

$$M(q)(\ddot{q}_d - \ddot{q}) + K_p(q_d - q) + K_i \int (q_d - q) + K_v(\dot{q}_d - \dot{q}) = 0 \qquad (IV.99)$$

Puisque la matrice d'inertie est non singulière, nous pouvons récrire l'équation ci-dessus comme suite :

$$\ddot{E} + M^{-1}K_pE + M^{-1}K_i \int E + M^{-1}K_v\dot{E} = 0 \qquad (IV.100)$$

Où $E = q_d - q$ qui est approché de zéro asymptotiquement en choisissant les gains K_p, K_i et K_v d'une manière appropriée.

Cette méthode a un avantage par rapport à la méthode précédente parce que le modèle n'a pas besoin d'être évalués en temps réel. Cependant, elle ne fournit pas le découplage complet parce que la matrice d'inertie n'est pas diagonale. En outre, puisque les gains sont continûment modifiés par la matrice d'inertie, la réponse est une fonction de la configuration et de la charge. Une manière d'éviter ce problème est de continûment modifier les gains K_p, K_i et K_v. Ceci évidemment suggère une approche de commande adaptative.

IV.4.2.Analyse de stabilité

Pour les systèmes non-linéaires, l'analyse de stabilité est beaucoup plus difficile, dans cette section nous présentons une méthode d'analyse de stabilité à la laquelle est applicable les systèmes linéaires et non-linéaires.

Une telle loi de commande utilise de manière essentielle la structure Lagrangienne des systèmes mécaniques pour faire décroître une fonction de Lyapunov. Elle repose sur la propriété suivante du modèle dynamique [99][100][101] :

Pour tout vecteur $X \in R^n$, on admet que :

$$X^T(\dot{M}(q,\dot{q}) - 2C(q,\dot{q}))X = 0 \qquad (IV.101)$$

Posons :

$$\dot{q}_r = \dot{q}_d + \Lambda E(t) \qquad (IV.102)$$

Où Λ est une matrice de gains définie positive.

et :

$$S(t) = \dot{q}_r - \dot{q} = \dot{E}(t) + \Lambda E(t) \qquad (IV.103)$$

Considérons la loi de commande suivante [100] :

$$\Gamma = M(q)\ddot{q}_r + C(q,\dot{q})\dot{q}_r + G(q) + K_v S + K_p E \qquad (IV.104)$$

où K_p et K_v sont des matrices constantes, symétriques et positives de dimension (n,n), en combinant l'équation $(III.104)$ avec $(III.62)$ on aura l'équation de l'erreur en boucle fermée:

$$M(q)\dot{S} + C(q,\dot{q})S = -K_v S - K_p E \qquad (IV.105)$$

L'équation $(IV.105)$ définit un système passif ayant pour entrée $u = -K_v S$ et pour sortie $y = S$, le vecteur d'état étant $X = \begin{pmatrix} E \\ S \end{pmatrix}$. En effet, considérons la fonction d'énergie définie positive suivante :

$$V(X,t) = \frac{1}{2} S^T M(q) S + \frac{1}{2} E^T K_p E \qquad (IV.106)$$

En dérivant *V(t)* par rapport au temps et en utilisant la propriété $(IV.101)$, on obtient donc :

$$\dot{V}(t) = -S^T K_v S - E^T (A^T K_p) E \qquad (IV.107)$$

Il est évident que : $-S^T K_v S \geq \dot{V}(t)$ .ou encore : $y^T u \geq \dot{V}(t)$ d'où :

$$\int_{t0}^{t} y^T(\sigma) u(\sigma) d\sigma \geq V(X(t),t) - V(X(t_0),t_0) \qquad (IV.108)$$

L'équation $(IV.105)$ montre que le système bouclé avec cette loi de commande est stable au sens de Lyapunov, cependant les performances sont limitées par la précision du modèle.

IV.4.2.Formulation du problème

Grace au modèle identifié précédemment différents lois de commande on été employée :

- la commande PID avec compensation de la gravité (figure *(IV .7)*):

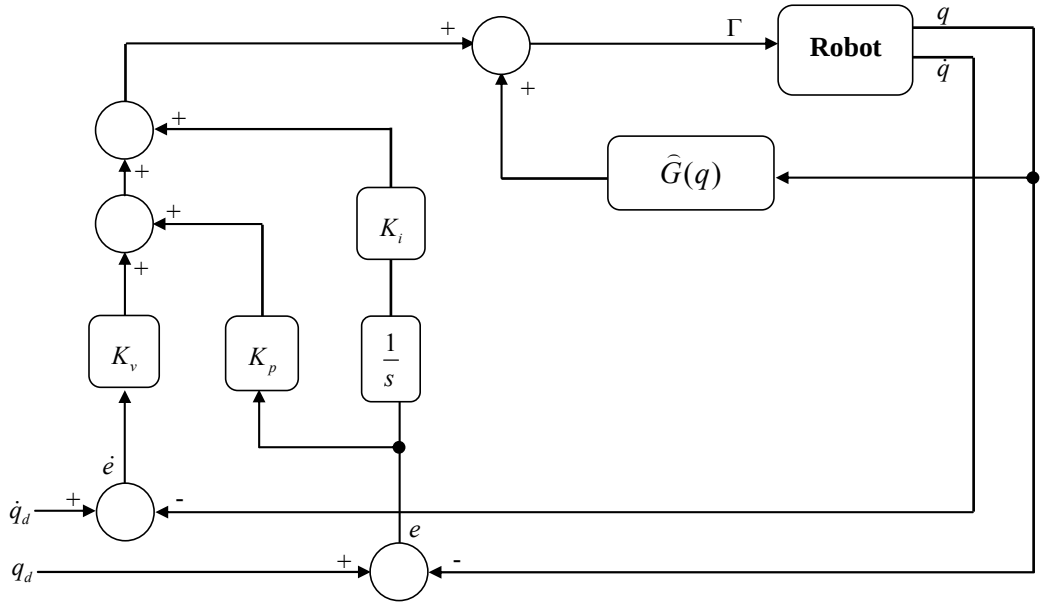

Figure *(IV .7)* : Commande PID avec compensation de la gravité

- la commande par couple calculé sans compensation des forces de centrifuges et de Coriolis (figure *(IV .8)*):

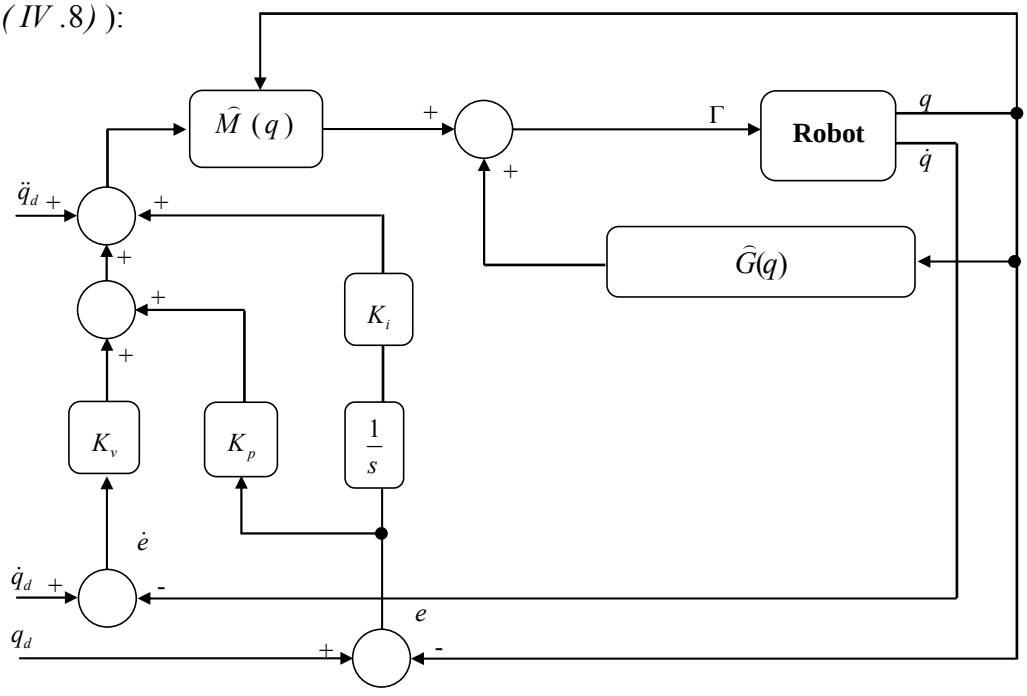

Figure *(IV .8)* : Commande par couple calculé sans compensation des forces de centrifuges et de Coriolis

- la commande par couple calculé (figure *(IV .9)*):

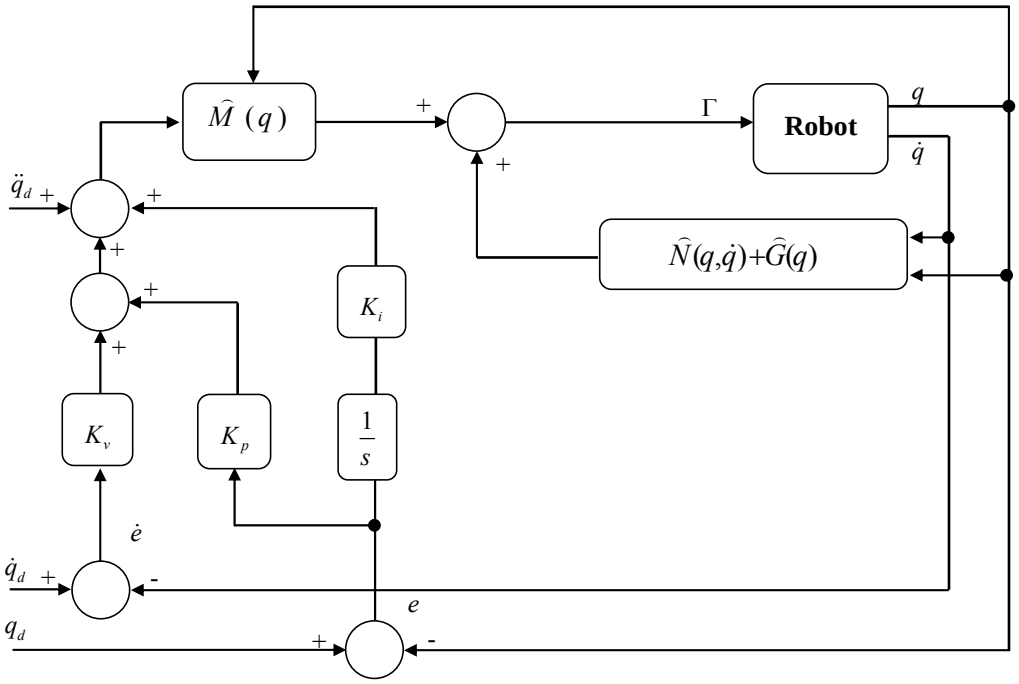

Figure *(IV .9)* : Commande par couple calculé

- la commande par couple calculé en mode glissant (figure *(IV .10)*):

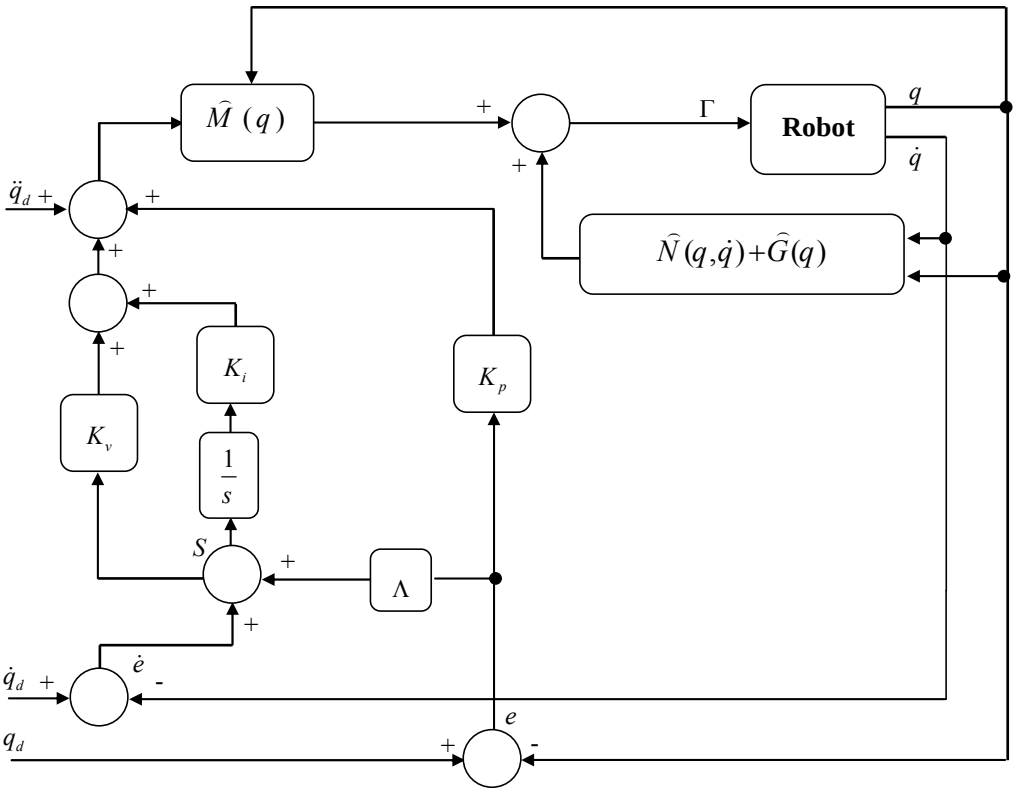

Figure *(IV .10)* : Commande par couple calculé en mode glissant

Pour les trois premières lois de commande, on défini le problème d'optimisation multiobjectif comme suite :

$$\begin{cases} \min J(\varepsilon_1) = \varepsilon_1^T \varepsilon_1 \\ \min J(\varepsilon_2) = \varepsilon_2^T \varepsilon_2 \end{cases} \quad (IV.109)$$

Avec les contraintes suivantes :

$$\begin{cases} \Gamma_{1\min} \le \Gamma_1 \le \Gamma_{1\max} \\ \Gamma_{2\min} \le \Gamma_2 \le \Gamma_{2\max} \end{cases} \quad (IV.110)$$

Où

$J(\varepsilon_1)$ est l'erreur de position sur la première articulation.

$J(\varepsilon_2)$ est l'erreur de position sur la deuxième articulation.

Pour la quatrième loi de commande le problème sera défini comme suite :

$$\begin{cases} \min J(S_1) = S_1^T S_1 \\ \min J(S_2) = S_2^T S_2 \end{cases} \quad (IV.111)$$

Avec les contraintes suivantes :

$$\begin{cases} \Gamma_{1\min} \le \Gamma_1 \le \Gamma_{1\max} \\ \Gamma_{2\min} \le \Gamma_2 \le \Gamma_{2\max} \end{cases} \quad (IV.112)$$

Où

$$S(t) = \dot{q}_r - \dot{q} = \dot{E}(t) + \Lambda E(t) \quad (IV.113)$$

et

$\Lambda = \begin{bmatrix} \Lambda_1 & 0 \\ 0 & \Lambda_2 \end{bmatrix}$ est une matrice de gains définie positive.

et $S = \begin{bmatrix} S_1 \\ S_2 \end{bmatrix}, E = \begin{bmatrix} E_1 \\ E_2 \end{bmatrix}, \dot{E} = \begin{bmatrix} \dot{E}_1 \\ \dot{E}_2 \end{bmatrix},$

$E = q_d - q \;\; ; \; \dot{E} = \dot{q}_d - \dot{q}$

IV.5. Résultats de simulation du Robot

Afin d'évaluer l'apport de la modification introduite dans la loi de commande qui a été exposée, nous présentons les résultats de simulation de cette dernière appliquée à un robot manipulateur à deux degrés de liberté .Cette simulation a été réalisée en se servant du logiciel MATLAB. La méthode numérique d'Euler a été utilisée avec un pas de 4.10^{-3} seconde lors de cette simulation pour l'intégration de l'équation $(IV.43)$, alors que notre algorithme génétique est employé pour l'optimisation des paramètres de régulateur tenant en compte le problème d'optimisation multiobjectif formulé par les équations $(IV.109),(IV.110)$ et $(IV.111),(IV.112)$.

Les erreurs de position des deux articulations ont été collectées au cours d'un mouvement du bras en poursuite d'une trajectoire (régulation dans le cas d'une trajectoire constant) assurant une continuité en position, vitesse et accélération donnée par :

- pour la poursuite:

$$\theta_{1d}(t) = 2\cos(\frac{4.\pi.t}{5}) + \sin(\frac{2.\pi.t}{5}) \quad rad$$

$$\theta_{2d}(t) = 1 - 2\cos(\frac{4.\pi.t}{5}) + \sin(\frac{2.\pi.t}{5}) \quad rad$$

- pour la régulation

$$\theta_{1d}(t) = \frac{\pi}{2} \quad rad \ ,$$

$$\theta_{1d}(t) = \frac{\pi}{6} \quad rad \, .$$

Les résultats de simulation des différentes lois de commande décrites précédemment sont illustrés par les figures ci-dessous, ainsi que les paramètres du régulateur optimisés par l'algorithme génétique sont présents.

Nous avons aussi testé la robustesse de la dernière loi de commande par l'introduction d'une perturbation externe ajoutée au système sous la forme d'une charge.

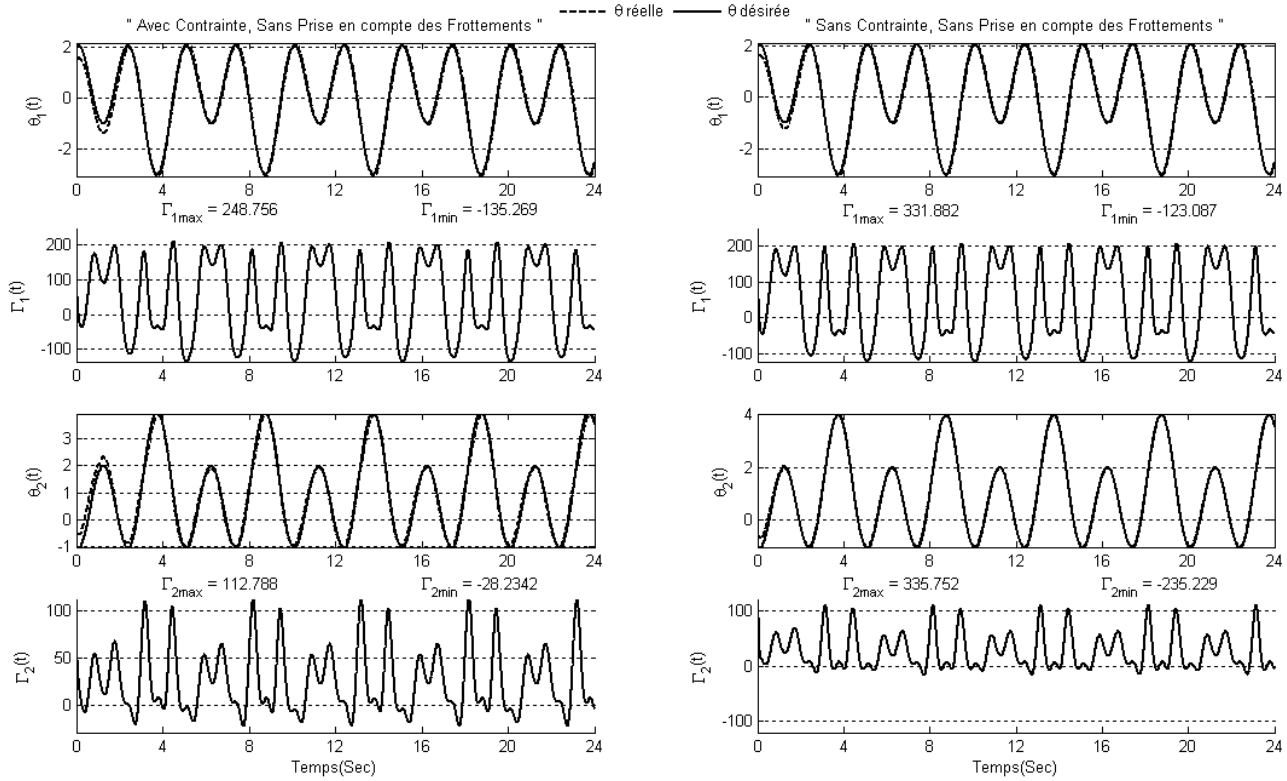

Figure $(IV.11)$: ***Résultat de poursuite de la commande PID avec compensation de la gravité sans prise en compte des frottements***

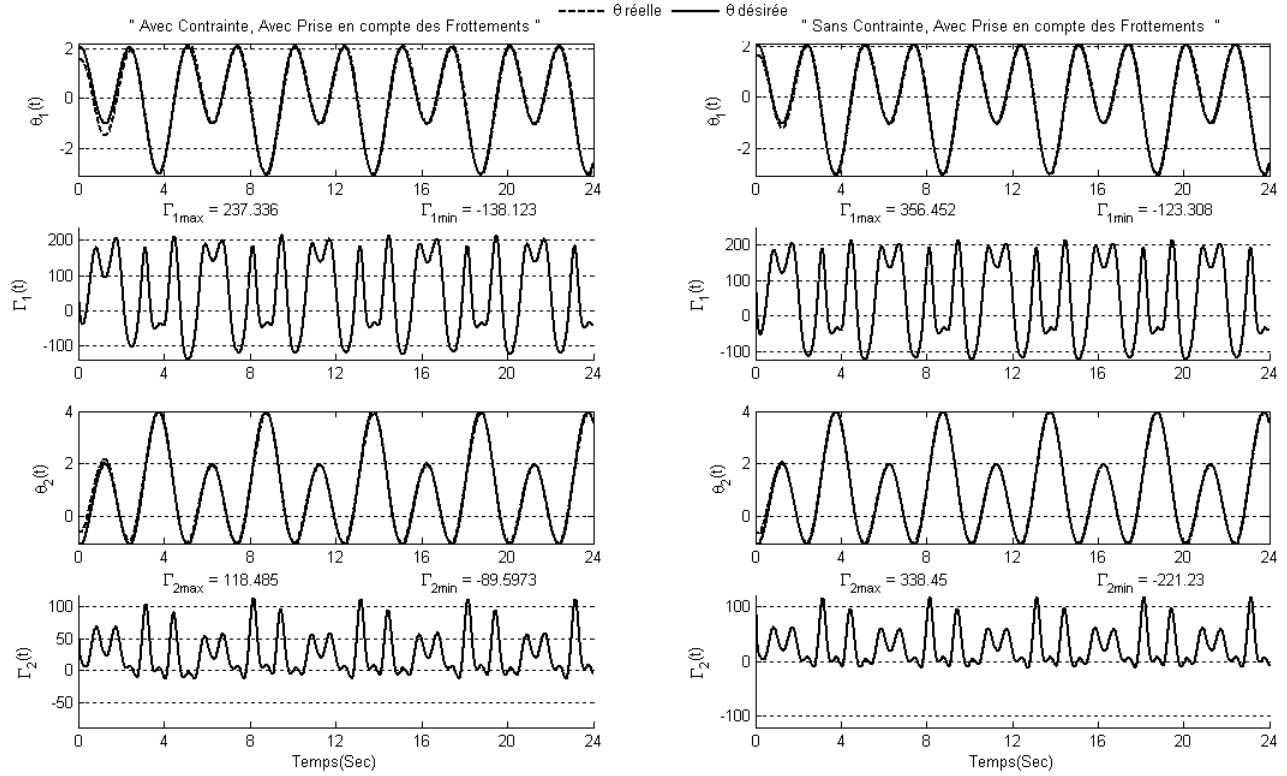

Figure $(IV.12)$: ***Résultat de poursuite de la commande PID avec compensation de la gravité avec prise en compte des frottements***

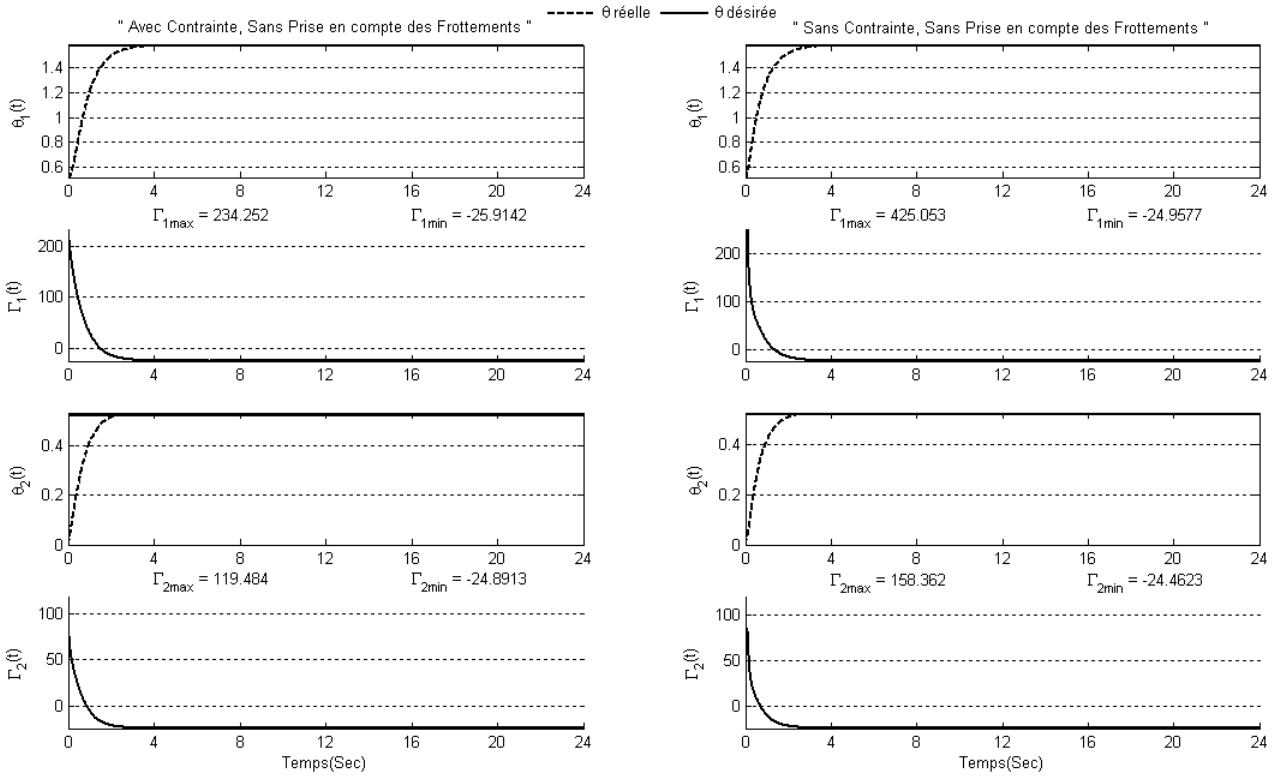

Figure $(IV.13)$: ***Résultat de régulation de la commande PID avec compensation de la gravité sans prise en compte des frottements***

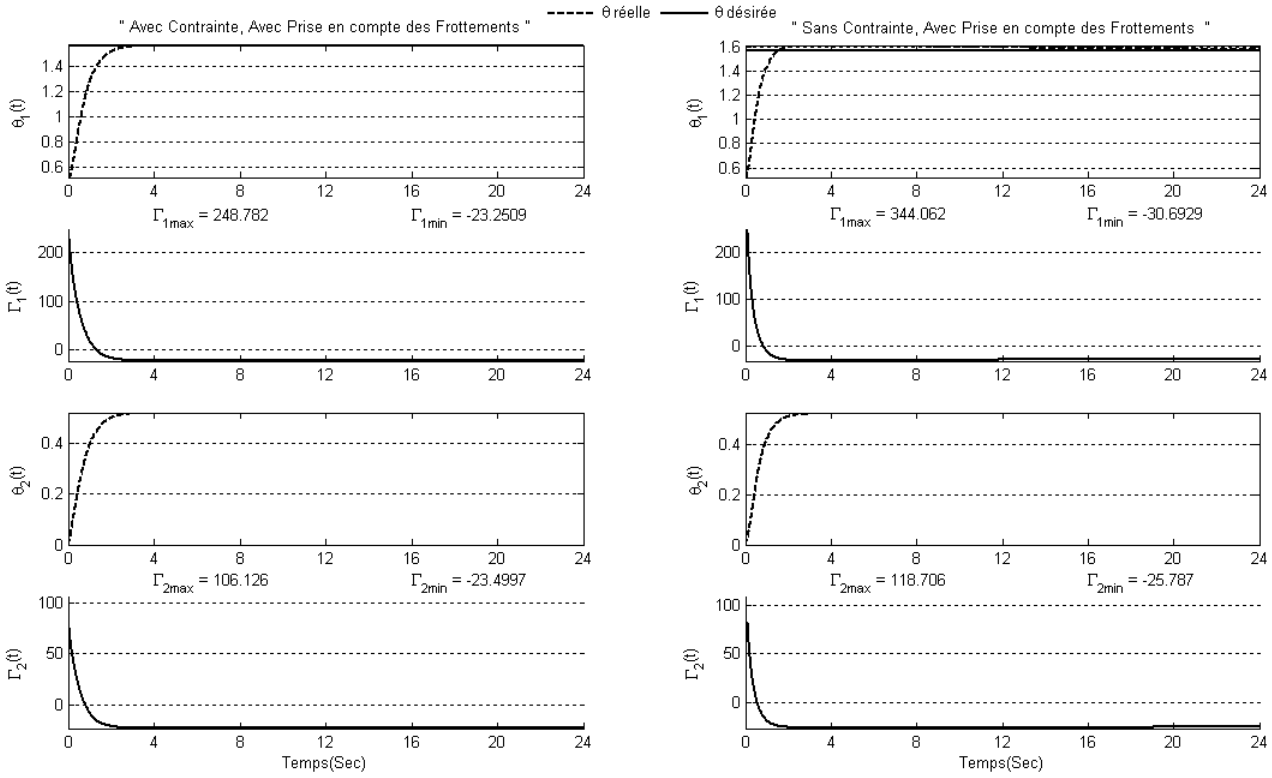

Figure $(IV.14)$: ***Résultat de régulation de la commande PID avec compensation de la gravité avec prise en compte des frottements***

Tableau $(IV.2)$: **Paramètres du régulateur avec les valeurs extrême des couples de la commande PID avec compensation de la gravité**

Mode \ Gains		K_p	K_v	K_i	$\Gamma_{\min}$	$\Gamma_{\max}$	K_p	K_v	K_i	$\Gamma_{\min}$	$\Gamma_{\max}$
		Avec contrainte					**Sans contrainte**				
Poursuite		**Sans frottements**									
	q_1	**173,32**	**162,47**	**95,46**	**-135,27**	**248,76**	**141,94**	**180,84**	**0,30**	**-123,09**	**331,88**
	q_2	**44,88**	**49,47**	**38,42**	**-28,23**	**112,79**	**234,90**	**123,17**	**14,09**	**-235,23**	**335,75**
		Avec frottements									
	q_1	**140,18**	**177,52**	**112,76**	**-138,12**	**237,34**	**196,19**	**181,04**	**3,82**	**-123,31**	**356,45**
	q_2	**59,25**	**76,06**	**12,33**	**-89,60**	**118,48**	**198,54**	**124,73**	**11,30**	**-221,23**	**338,45**
Régulation		**Sans frottements**									
	q_1	**72,15**	**66,09**	**0,45**	**-25,91**	**234,25**	**257,19**	**190,23**	**1,18**	**-24,96**	**425,05**
	q_2	**149,27**	**97,95**	**4,70**	**-24,89**	**119,48**	**153,08**	**99,91**	**1,77**	**-24,46**	**158,36**
		Avec frottements									
	q_1	**93,85**	**72,73**	**0,16**	**-23,25**	**248,78**	**186,22**	**104,79**	**14,38**	**-30,69**	**344,06**
	q_2	**123,76**	**86,03**	**0,16**	**-23,50**	**106,13**	**128,45**	**82,31**	**0,45**	**-25,79**	**118,71**

Interprétation:

Les figures $(IV.11)$, $(IV.12)$ $(IV.13)$ et $(IV.14)$ montrent les résultats de la commande PID avec compensation de la gravité. D'un côté, avec la prise en compte des contraintes, on peut voir des temps de réponse assez longs en régime transitoire et des erreurs importantes en terme de poursuite. Ce comportement est dû à la non considération des non linéarités du robot, définies par l'inertie du robot et les forces de Centrifuges et de Coriolis, dans la loi de commande. Par contre, pour la régulation, on obtient des temps de réponse courts et des erreurs nulles en régime permanent. D'un autre côté, en raison de l'absence des contraintes, les couples sont amplifiés. Le comportement du système reste le même malgré cette augmentation à cause de la mise en place des saturations qui peuvent entraîner des erreurs statiques en régime permanent en régulation. Vu l'importance des effets de l'inertie et des forces de Centrifuges et de Coriolis, l'effet des frottements n'apparaît pas. Alors, le régulateur réagit de façon à compenser les deux premiers effets.

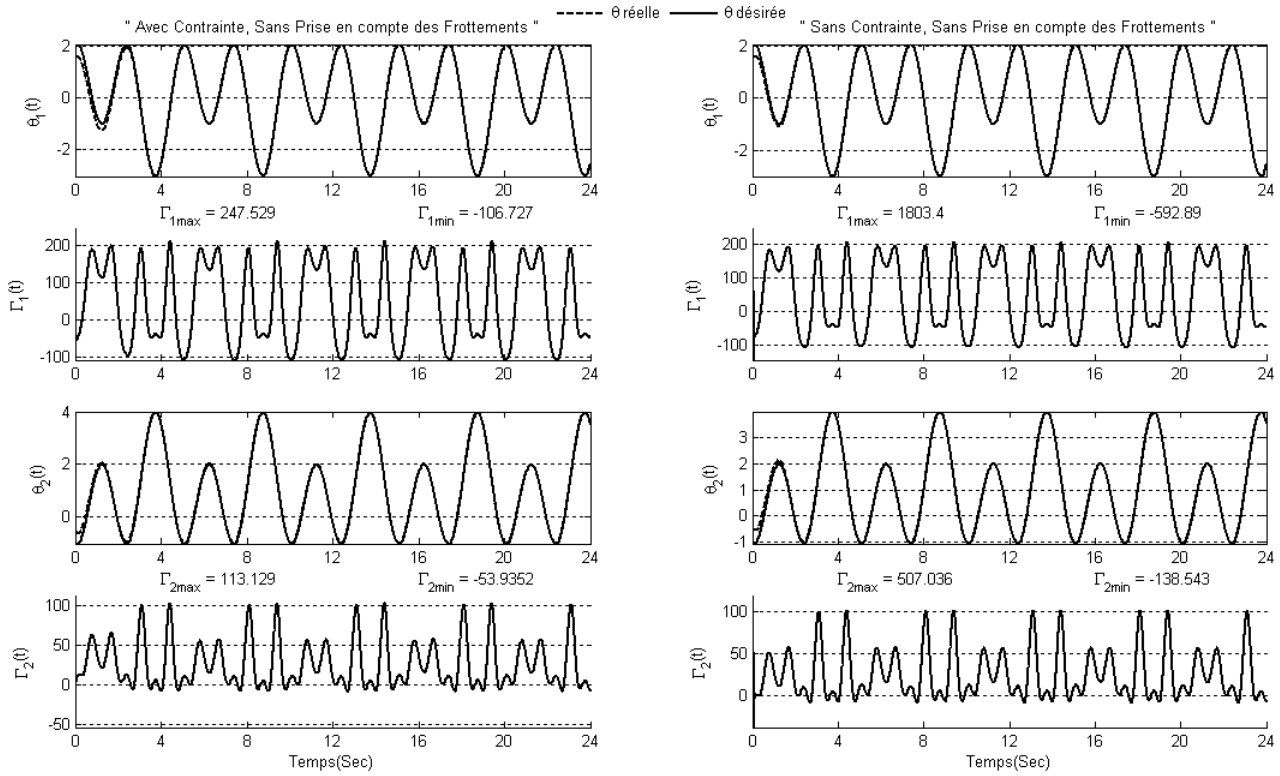

Figure (IV .15) : ***Résultat de poursuite de la commande par couple calculé sans compensation des forces de Centrifuges et de Coriolis sans prise en compte des frottements***

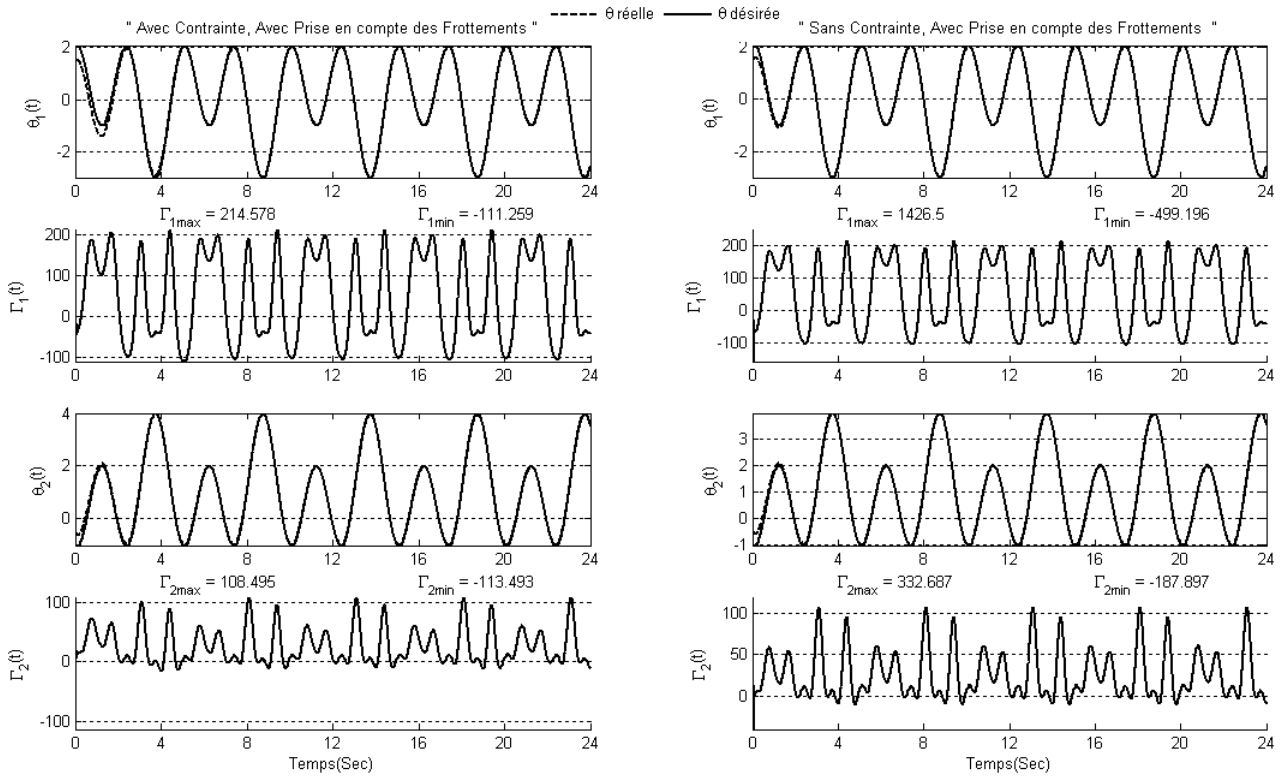

Figure (IV .16) : ***Résultat de poursuite de la commande par couple calculé sans compensation des forces de Centrifuges et de Coriolis sans prise en compte des frottements***

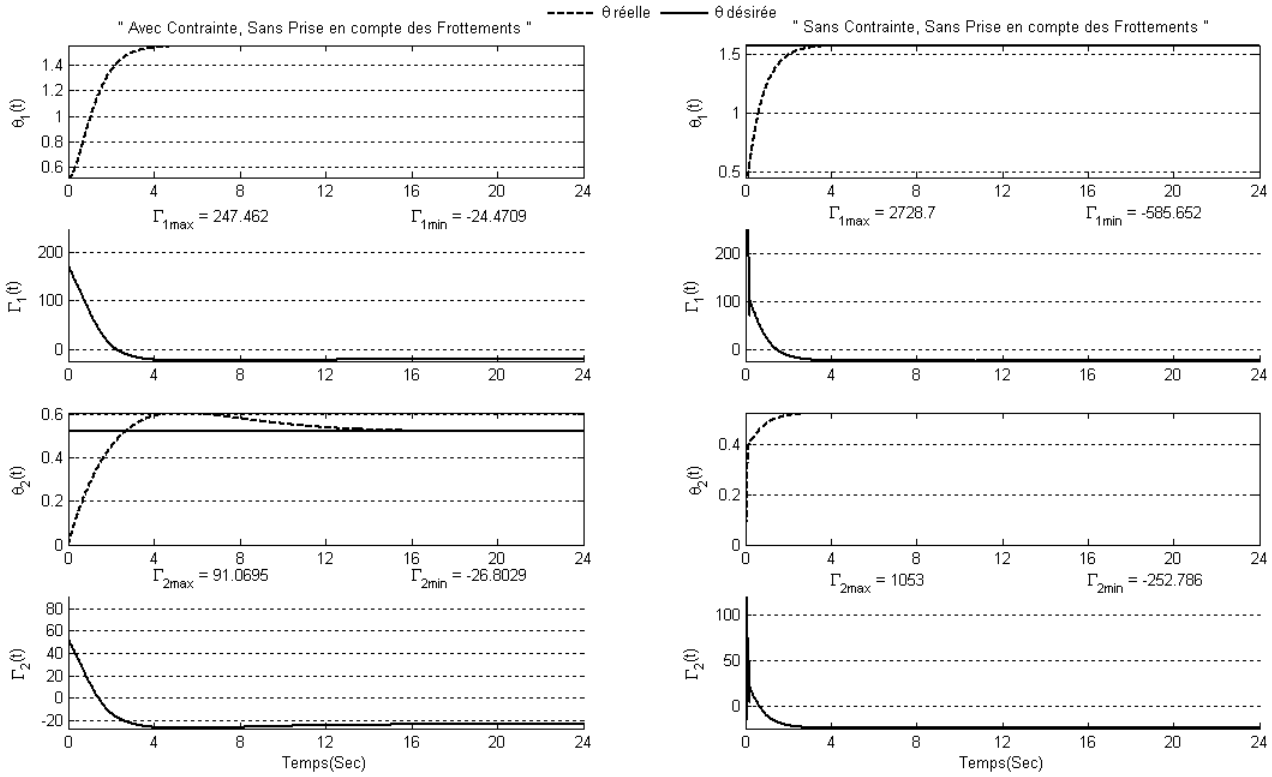

Figure $(IV.17)$: ***Résultat de régulation de la commande par couple calculé sans compensation des forces de Centrifuges et de Coriolis sans prise en compte des frottements***

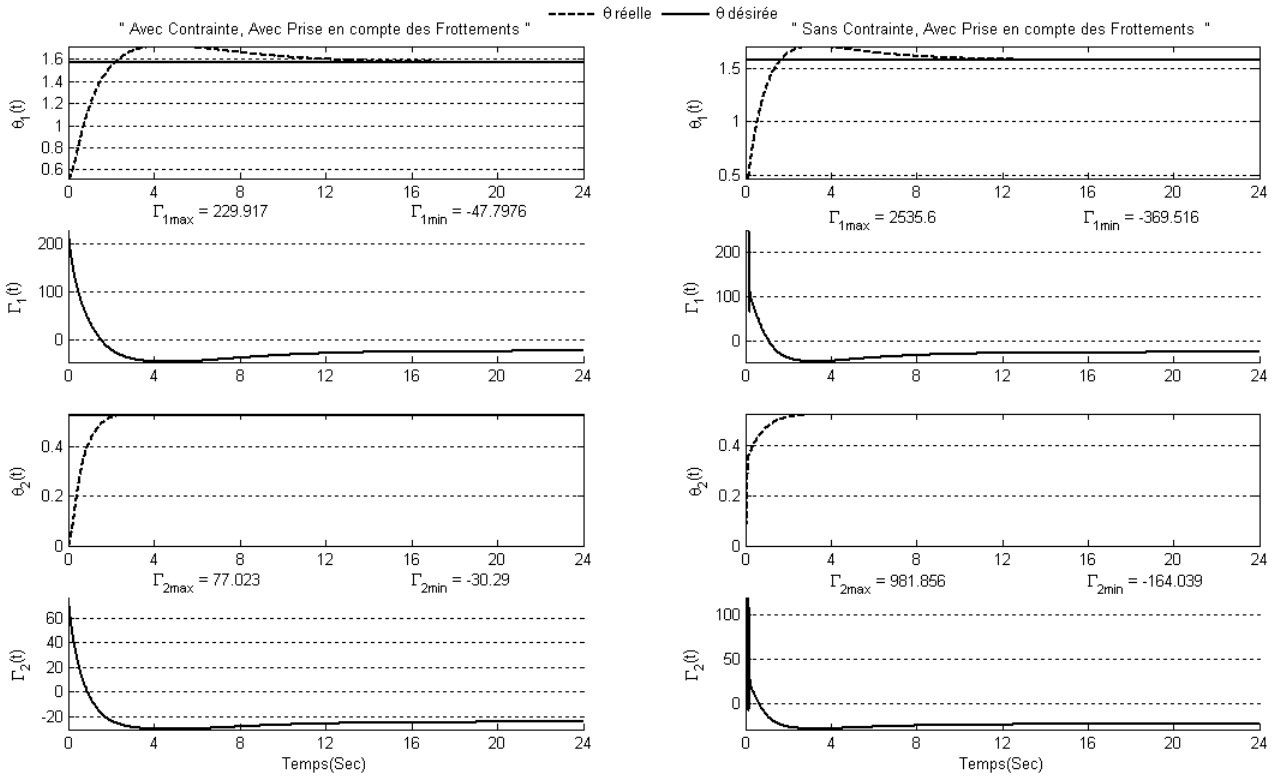

Figure $(IV.18)$: ***Résultat de régulation de la commande par couple calculé sans compensation des forces de Centrifuges et de Coriolis avec prise en compte des frottements***

Tableau (*IV* .3) : **Paramètres du régulateur avec les valeurs extrême des couples de la commande par couple calculé sans compensation des forces de Centrifuges et de Coriolis**

Gains / Mode		K_p	K_v	K_i	Γ_{min}	Γ_{max}	K_p	K_v	K_i	Γ_{min}	Γ_{max}
		Avec contrainte					**Sans contrainte**				
Poursuite		**Sans frottements**									
	q_1	**147,81**	**69,02**	**83,73**	**-106,73**	**247,53**	**197,07**	**136,66**	**2,80**	**-592,89**	**1803,4**
	q_2	**203,52**	**99,32**	**29,19**	**-53,94**	**113,13**	**180,94**	**149,95**	**16,28**	**-138,54**	**507,04**
		Avec frottements									
	q_1	**112,32**	**103,82**	**112,76**	**-111,26**	**214,58**	**185,63**	**132,75**	**3,24**	**-499,20**	**1426,5**
	q_2	**216,13**	**127,67**	**39,45**	**-113,49**	**108,50**	**256,01**	**173,02**	**17,75**	**-187,90**	**332,69**
Régulation		**Sans frottements**									
	q_1	**2,36**	**3,14**	**0,01**	**-24,47**	**247,46**	**173,32**	**125,13**	**0,16**	**-585,65**	**2728,7**
	q_2	**35,79**	**51,62**	**7,78**	**-26,80**	**91,07**	**211,15**	**150,15**	**0,01**	**-252,79**	**1053,0**
		Avec frottements									
	q_1	**5,29**	**6,27**	**1,04**	**-47,80**	**229,92**	**157,78**	**111,05**	**37,69**	**-369,52**	**2535,6**
	q_2	**9,98**	**3,92**	**0,60**	**-30,29**	**77,02**	**212,32**	**150,15**	**1,04**	**-164,04**	**981,86**

Interprétation:

Les résultats de la commande par couple calculé, sans compensation des forces de Centrifuges et de Coriolis, sont présentés par les figures (*IV* .15) , (*IV* .16) , (*IV* .17) et (*IV* .18) . Lors de la prise en compte des contraintes, dans le cas de poursuite, on peut voir une amélioration significative en termes de rapidité et de précision (des temps de réponse plus courts en régime transitoire et des faibles erreurs en régime permanent) par rapport à la commande précédente. Ce comportement est dû à la prise en compte de la non linéarité de l'inertie du robot dans la loi de commande. Dans le cas de régulation, on peut voir des dépassements des postions désirées qui augmentent le temps de réponse en régime transitoire et qui ne sont pas pratiques dans un environnement avec contraintes (obstacles, murs,…). Les erreurs en régime permanent restent nulles. Ce comportement est dû à la prise en compte de l'inertie du robot dans la loi de commande qui engent des couples supplémentaire et réduit la réaction du régulateur à cause des contraintes imposées ce qui est confirmé par des valeurs des paramètres du régulateur inférieurs à celles de la première loi de commande. Par contre, sans prise en en compte des contraintes, et pour les deux cas de poursuite et de régulation, la boucle de rétroaction produit des valeurs élevées des couples qui sont limités par la mise en place des saturations. Par conséquent, le comportement du système ne change pas. Les frottements n'ont pas d'effet devant les forces de Centrifuges et de Coriolis. Dans ce cas, le régulateur réagit de façon à compenser les effets de ces dernières.

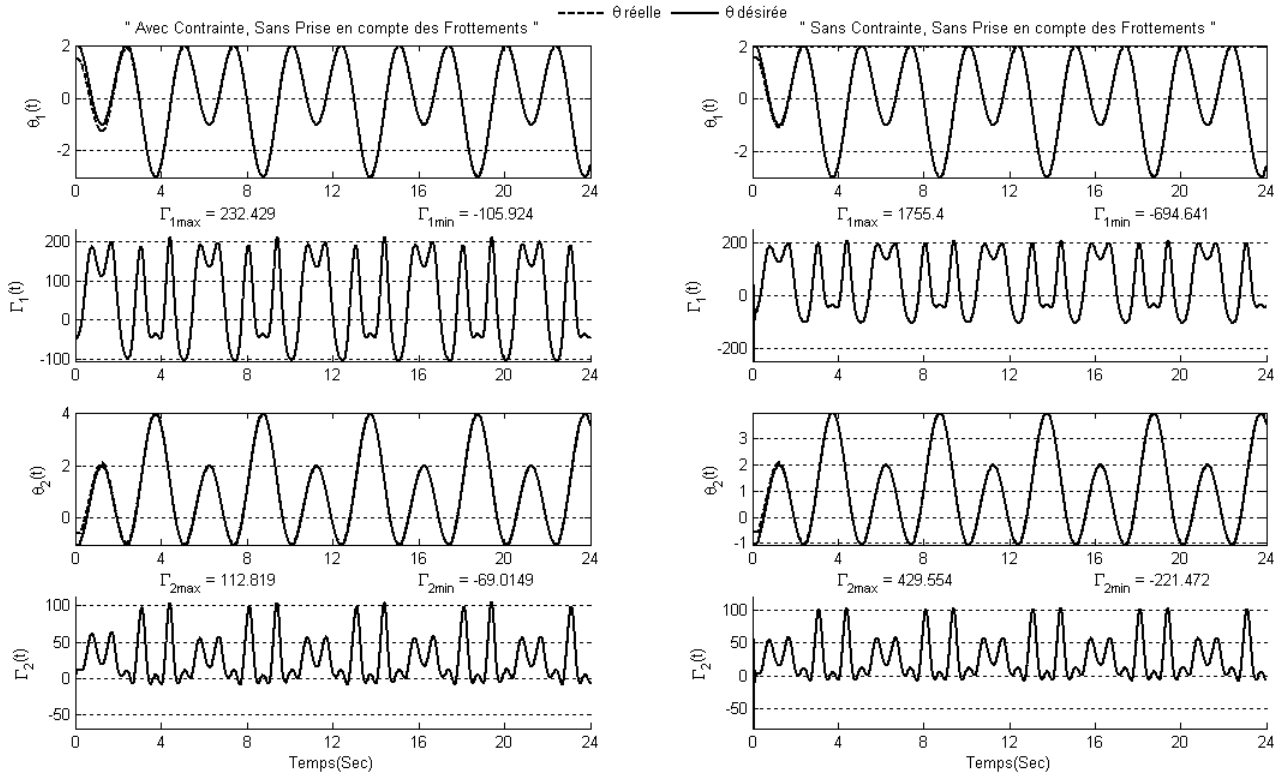

Figure (*IV* .19) : ***Résultat de poursuite de la commande par couple calculés sans prise en compte des frottements***

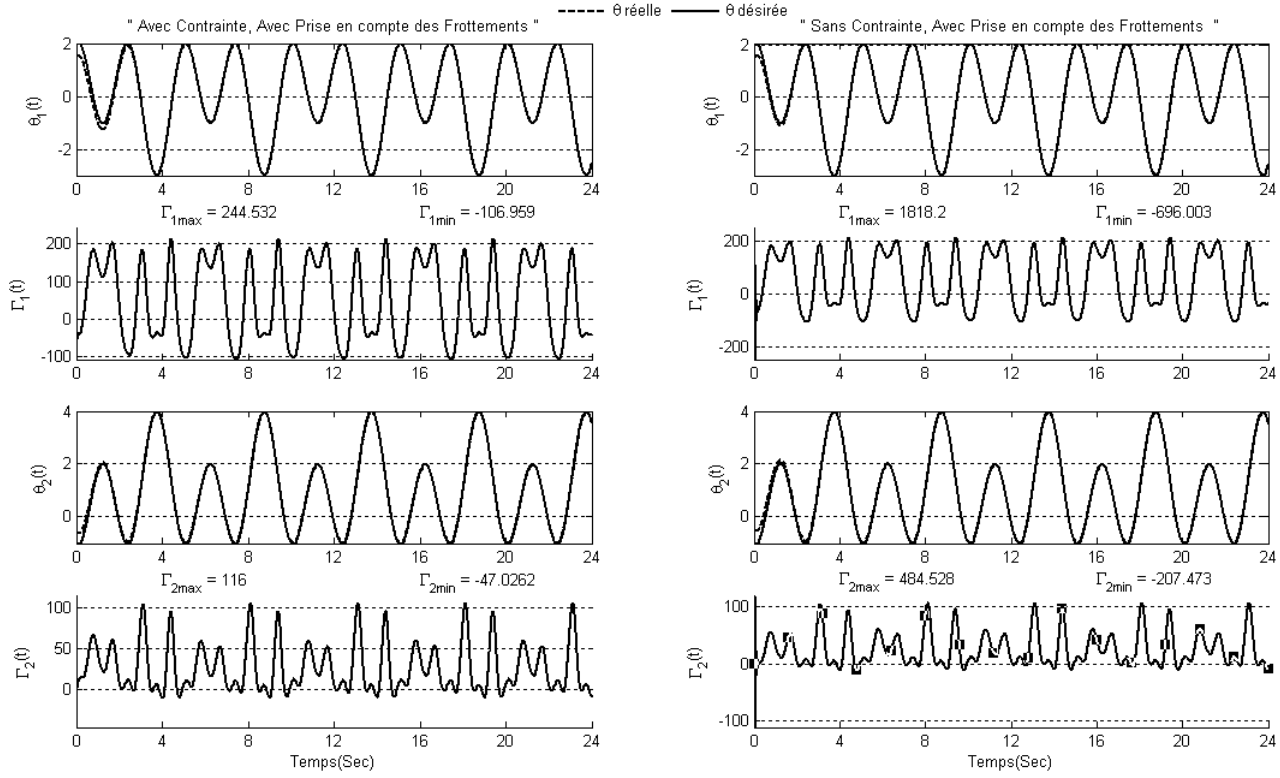

Figure (*IV* .20) : ***Résultat de poursuite de la commande par couple calculés avec prise en compte des frottements***

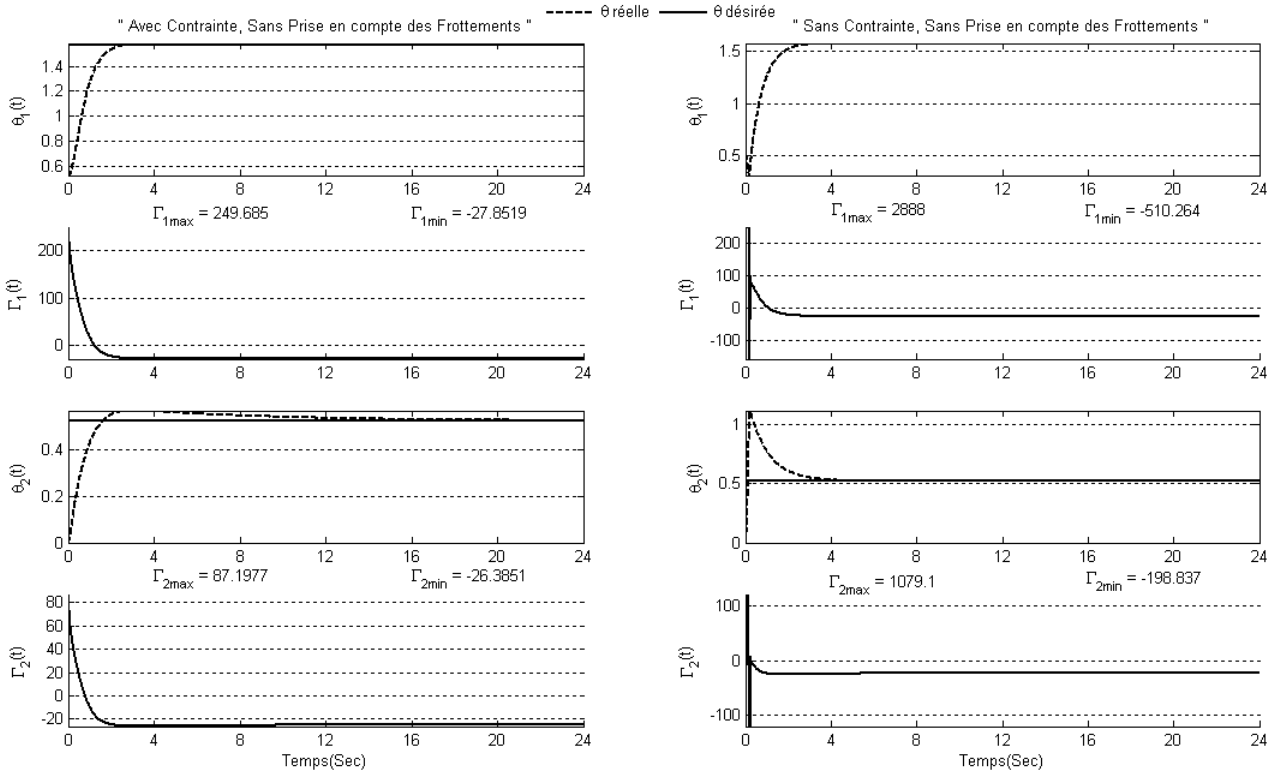

Figure (IV .21) : ***Résultat de régulation de la commande par couple calculé sans prise en compte des frottements***

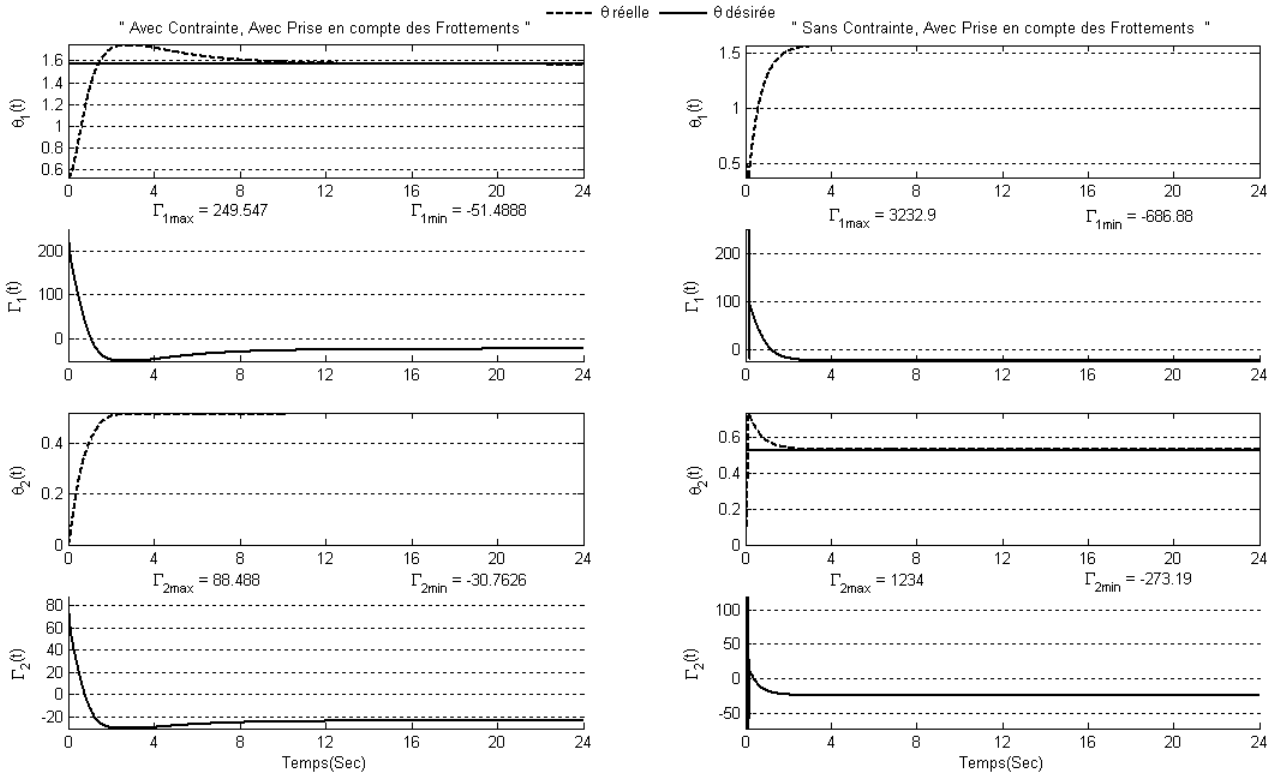

Figure (IV .22) : ***Résultat de régulation de la commande par couple calculé avec prise en compte des frottements***

Tableau $(IV.4)$: **Paramètres du régulateur avec les valeurs extrême des couples de la commande par couple calculé**

Gains / Mode		K_p	K_v	K_i	Γ_{min}	Γ_{max}	K_p	K_v	K_i	Γ_{min}	Γ_{max}
		Avec contrainte					**Sans contrainte**				
Poursuite		**Sans frottements**									
	q_1	**149,86**	**66,28**	**93,85**	**-105,92**	**232,43**	**225,22**	**153,28**	**8,37**	**-694,64**	**1755,4**
	q_2	**236,37**	**99,91**	**66,43**	**-69,01**	**112,82**	**295,60**	**188,27**	**28,16**	**-221,47**	**429,55**
		Avec frottements									
	q_1	**181,53**	**83,68**	**113,79**	**-106,96**	**244,53**	**206,45**	**150,15**	**6,31**	**-696,00**	**1818,2**
	q_2	**223,76**	**112,22**	**39,60**	**-47,03**	**116,00**	**279,18**	**186,32**	**75,08**	**-207,47**	**484,53**
Régulation		**Sans frottements**									
	q_1	**5,58**	**4,31**	**0,16**	**-27,85**	**249,68**	**208,21**	**122,00**	**0,30**	**-510,26**	**2888,0**
	q_2	**19,36**	**14,09**	**2,50**	**-26,39**	**87,20**	**85,34**	**78,40**	**1,18**	**-198,84**	**1079,1**
		Avec frottements									
	q_1	**4,70**	**3,53**	**1.18**	**-51.49**	**249,55**	**219,06**	**128,25**	**0,16**	**-686,88**	**3232,9**
	q_2	**24,06**	**15,65**	**0,16**	**-30.76**	**88,49**	**185,34**	**107,73**	**0,01**	**-273,19**	**1234,0**

Interprétation:

On visualise sur les figures $(IV.19)$, $(IV.20)$, $(IV.21)$ et $(IV.22)$ les résultats de la commande par couple calculé. Pour la poursuite et avec la présence des contraintes sur les couples, on observe une légère amélioration en terme de rapidité et de précision par rapport à la commande précédente, mais le temps de réponse reste moins court. Ce comportement est dû à l'utilisation d'une loi de commande qui réagit en fonction d'une erreur assez grande au démarrage et qui entraîne des petits temps de réponses. En revanche, pour la régulation, on peut voir des dépassements des postions désirées qui augmentent le temps de réponse en régime transitoire et des erreurs nulles en régime permanent. Ce comportement est dû à la prise en compte de l'inertie du robot et des forces de Coriolis et de Centrifuges dans la loi de commande. En l'absence des contraintes, l'effet de frottement conduit à des couples supérieurs comparés au cas où ces effets sont négligés. Ces couples sont limités par l'utilisation des saturations et peuvent entraîner des temps de réponse non réalisable dans le cas de régulation.

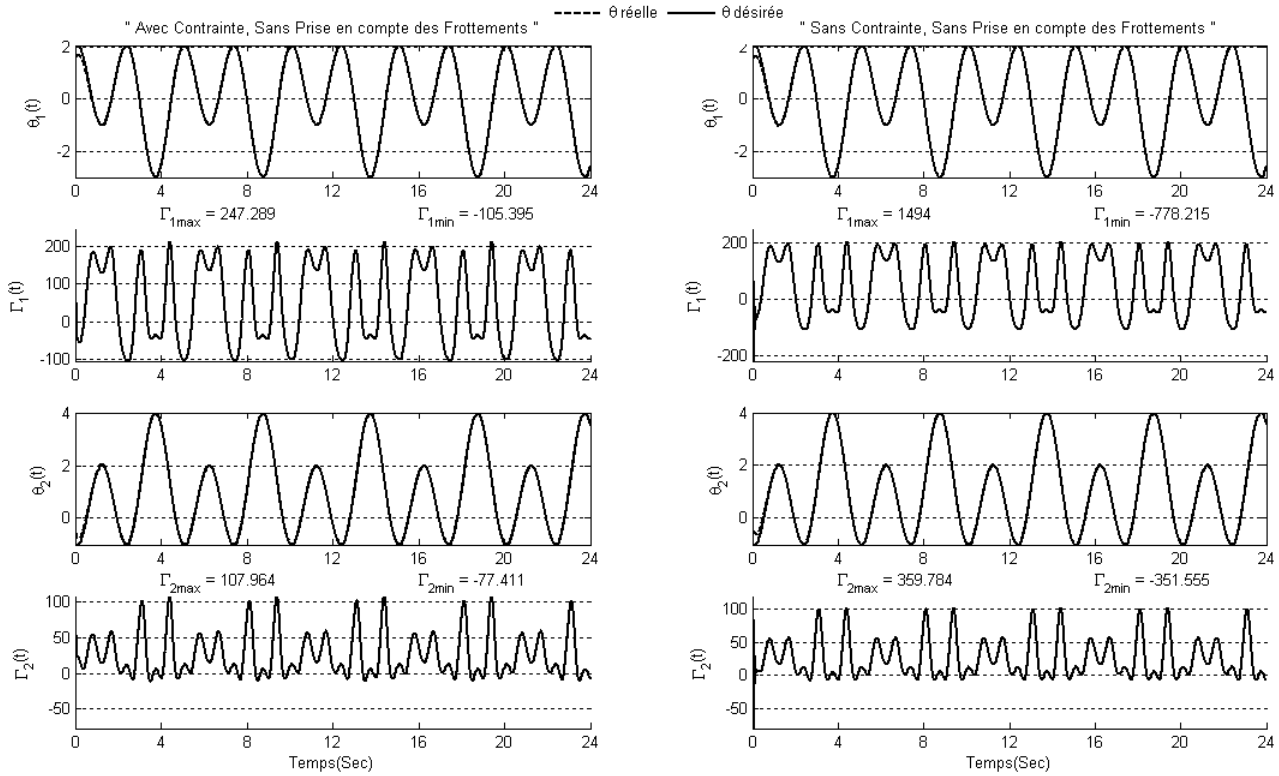

Figure (IV.23) : ***Résultat de poursuite de la commande par couple calculé sans prise en compte des frottements (mode glissant)***

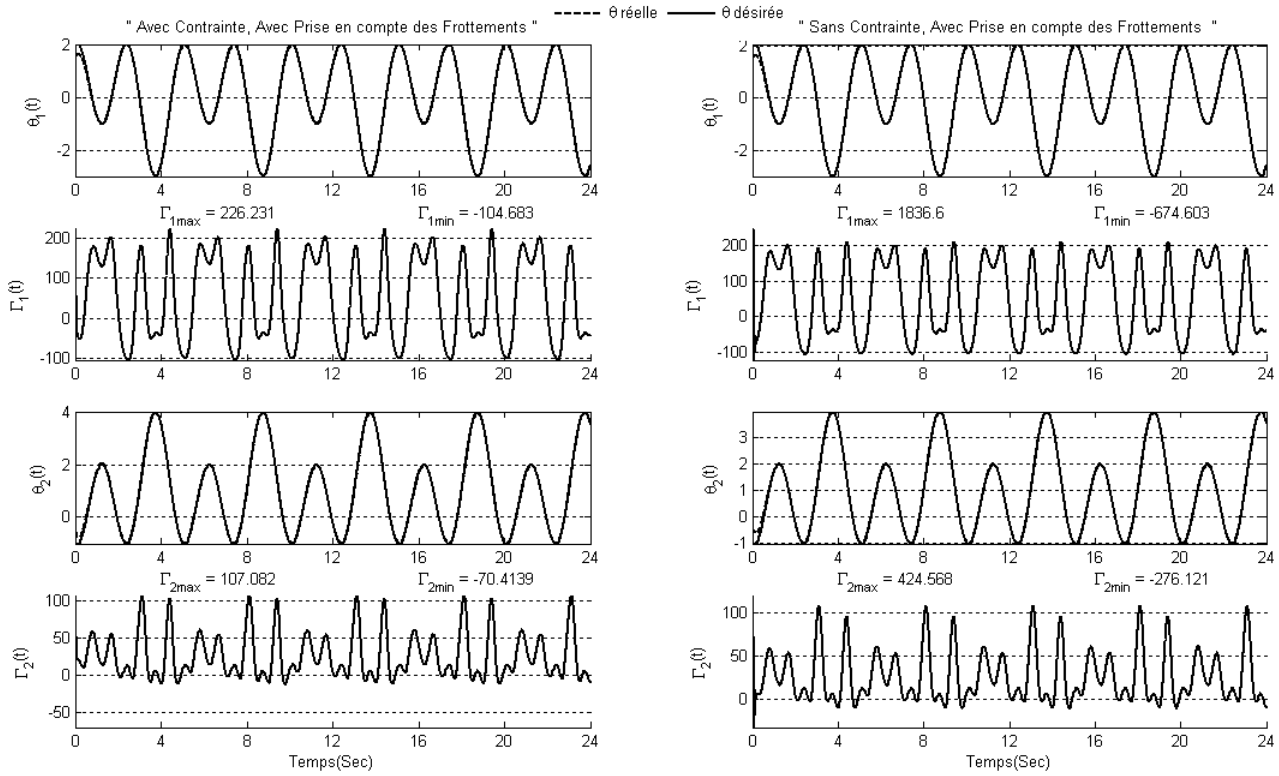

Figure (IV .24) : ***Résultat de poursuite de la commande par couple calculé avec prise en compte des frottements (mode glissant)***

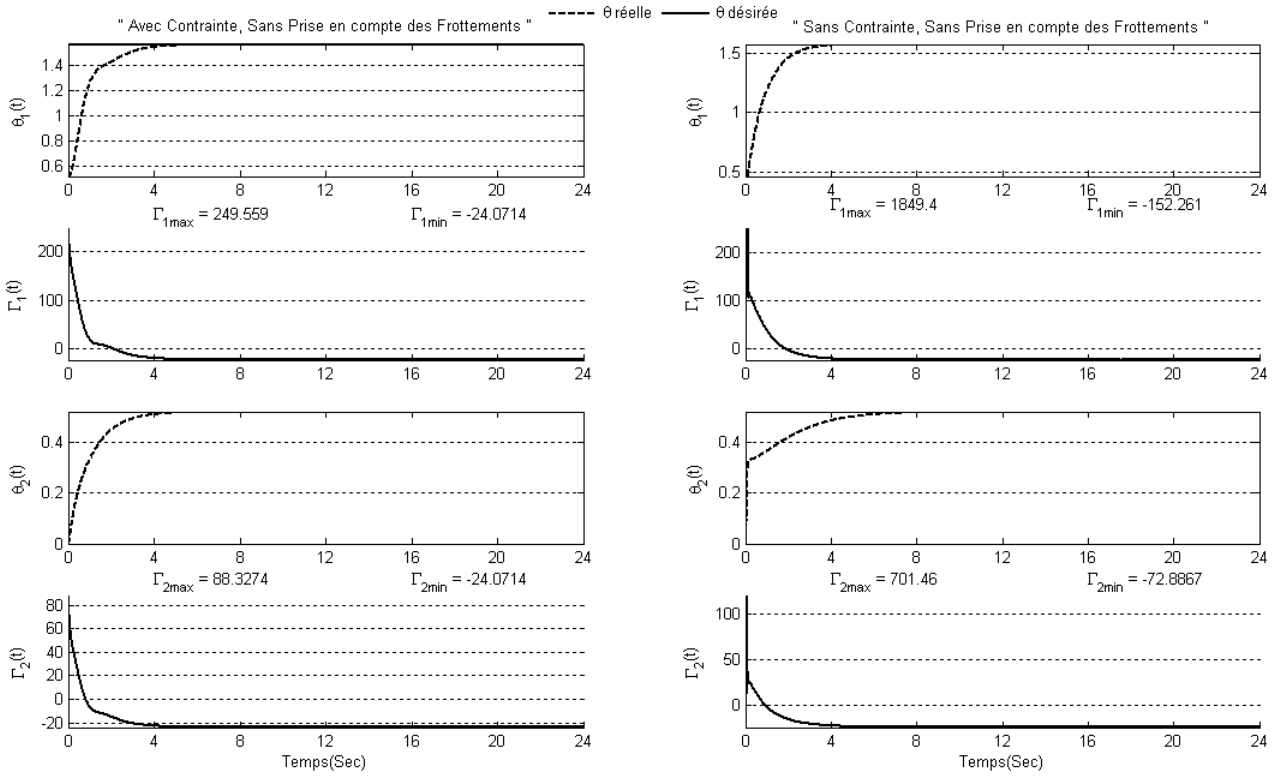

Figure (*IV* .25) : ***Résultat de régulation de la commande par couple calculé sans prise en compte des frottements (mode glissant)***

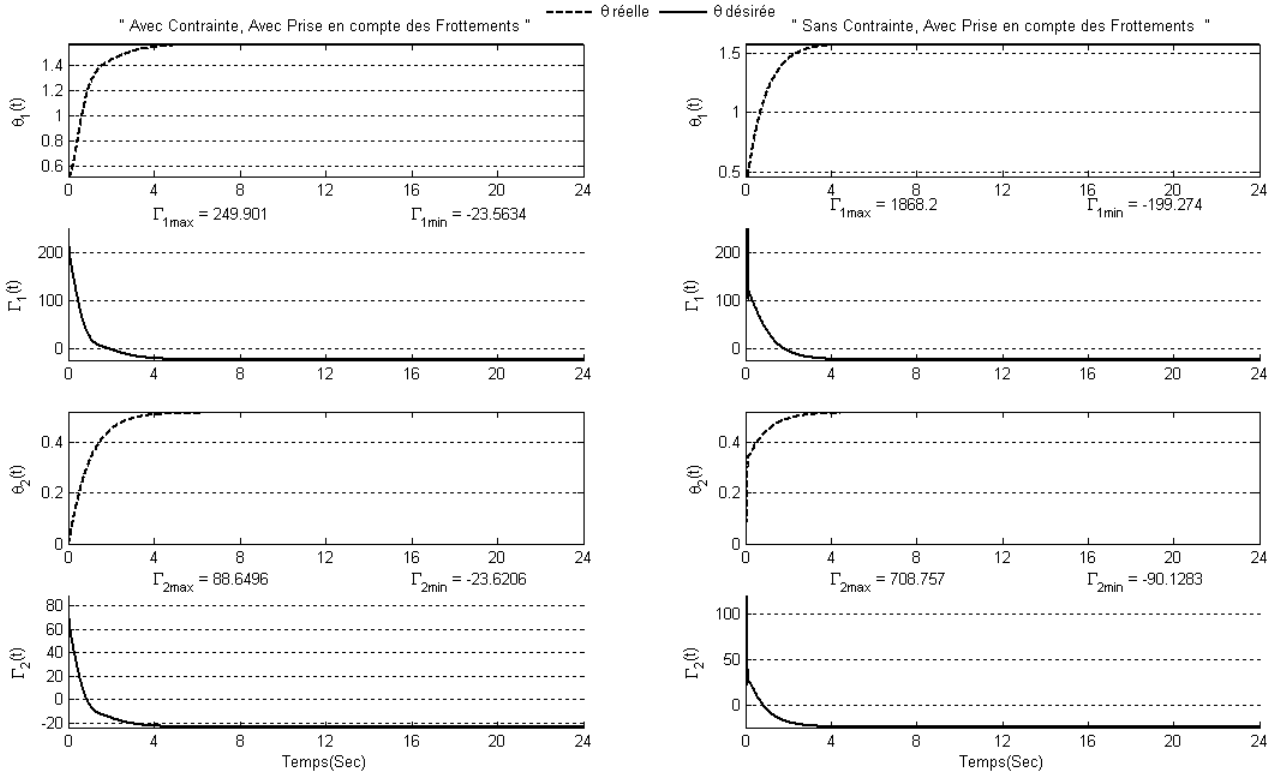

Figure (*IV* .26) : ***Résultat de régulation de la commande par couple calculé avec prise en compte des frottements (mode glissant)***

Tableau $(IV.5)$: **Paramètres du régulateur avec les valeurs extrême des couples de la commande par couple calculé en mode glissant**

Gains / Mode		K_p	K_v	K_i	Γ_{min}	Γ_{max}	K_p	K_v	K_i	Γ_{min}	Γ_{max}
		Avec contrainte					**Sans contrainte**				
Poursuite		**Sans frottements**									
	q_1	**0,89**	**49,08**	**126,39**	**-105,39**	**247,29**	**0,01**	**139,98**	**149,56**	**-778,21**	**1494,0**
	q_2	**197,95**	**65,50**	**149,85**	**-77,41**	**107,96**	**112,62**	**186,71**	**149,41**	**-351,55**	**359,78**
		Avec frottements									
	q_1	**4,12**	**46,34**	**73,91**	**-104,68**	**226,23**	**0,01**	**144,48**	**105,14**	**-674,60**	**1836,6**
	q_2	**188,27**	**62,37**	**149,71**	**-70,41**	**107,08**	**3,53**	**183,97**	**150,00**	**-276,12**	**424,57**
Régulation		**Sans frottements**									
	q_1	**0.30**	**4.51**	**9.54**	**-24.07**	**249.56**	**10,27**	**107,14**	**59,39**	**-152,26**	**1849,4**
	q_2	**2.94**	**20.54**	**124.05**	**-24.07**	**88.33**	**0,01**	**113,98**	**72,15**	**-72,89**	**701,46**
		Avec frottements									
	q_1	**0,01**	**4,70**	**8,95**	**-23,56**	**249,90**	**6,17**	**112,22**	**58,07**	**-199,27**	**1868,2**
	q_2	**0,30**	**23,86**	**4,70**	**-23,62**	**88,65**	**0,01**	**115,74**	**2,36**	**-90,13**	**708,76**

Interprétation:

L'analyse des résultats de la commande par couple calculé en mode glissant (présentés par les figures $(IV.23)$, $(IV.24)$, $(IV.25)$ et $(IV.26)$ permet de juger l'utilisation de la surface de glissement. En exerçant des contraintes sur les couples, on peut voir, dans les deux cas de poursuite et de régulation, une amélioration très intéressante en terme de rapidité et de précision par rapport aux trois cas précédents (temps de réponse très court en régime transitoire et des erreurs nulles en régime permanent). Ce comportement est dû à l'utilisation d'une commande qui réagit en fonction d'une erreur filtrée et à la prise en compte des différentes non linéarités du robot,, tel que l'inertie du robot et les forces de Centrifuges et de Coriolis. Lorsqu'on ne tient pas des contraintes, on obtient des valeurs élevées des couples qui sont limités avec la mise en place des saturations. Le comportement du système reste le même, mais les valeurs des couples sont supérieures dans le cas de la prise en compte des frottements par rapport au cas où ces effets sont négligés.

Cette loi de commande donne des meilleurs résultats en terme de poursuite et de régulation. A cet effet nous avons choisi cette loi de commande pour étudier sa robustesse dans le cas d'une charge ajoutée sur la deuxième articulation.

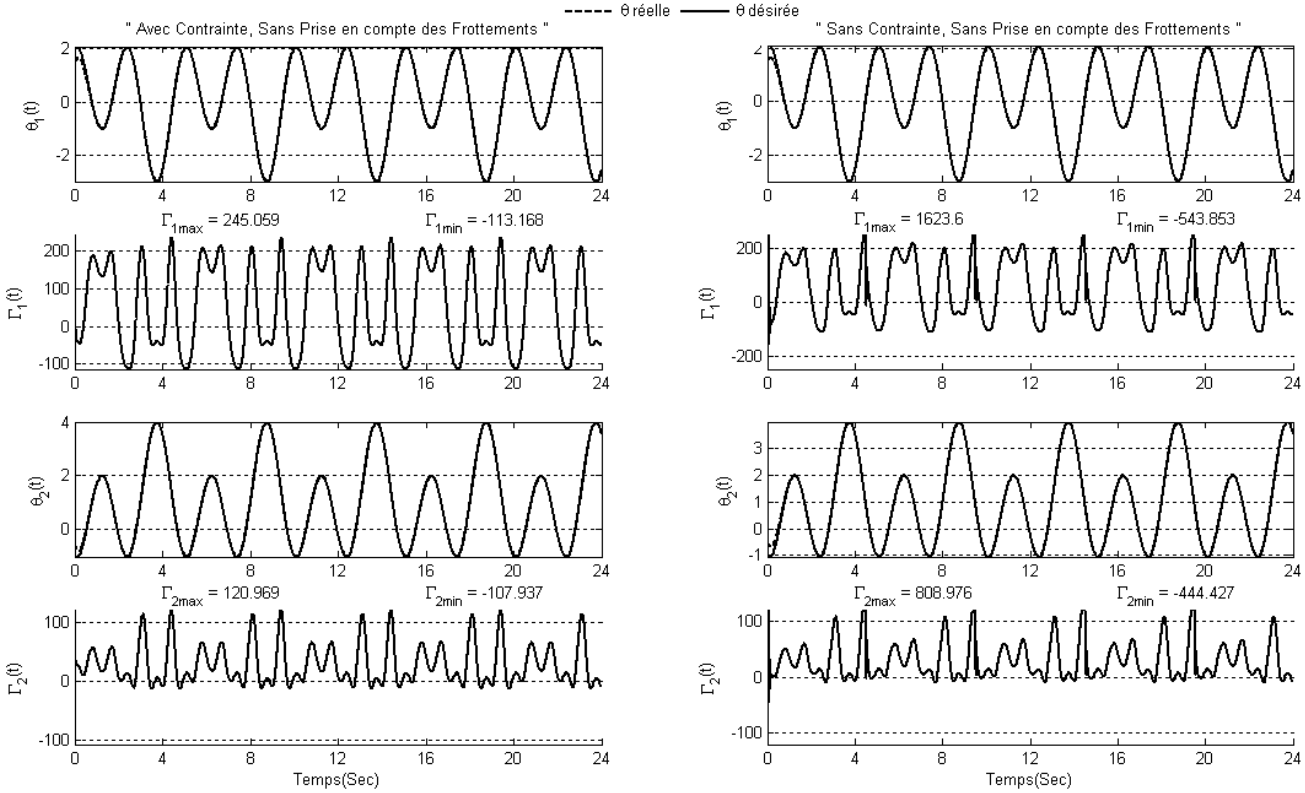

Figure (IV .27) : ***Résultat de poursuite de la commande par couple calculé sans prise en compte des frottements (mode glissant) avec charge***

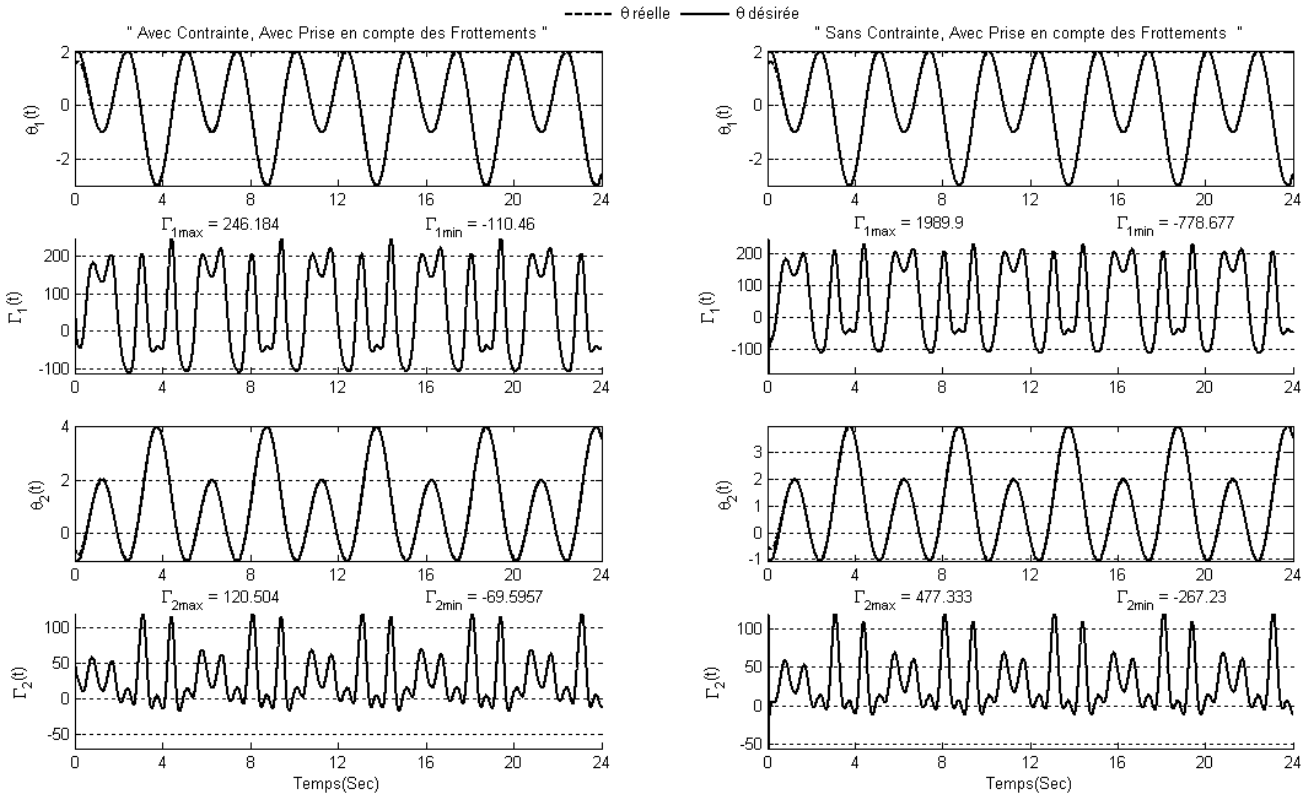

Figure (IV .28) : ***Résultat de poursuite de la commande par couple calculé avec prise en compte des frottements (mode glissant) avec charge***

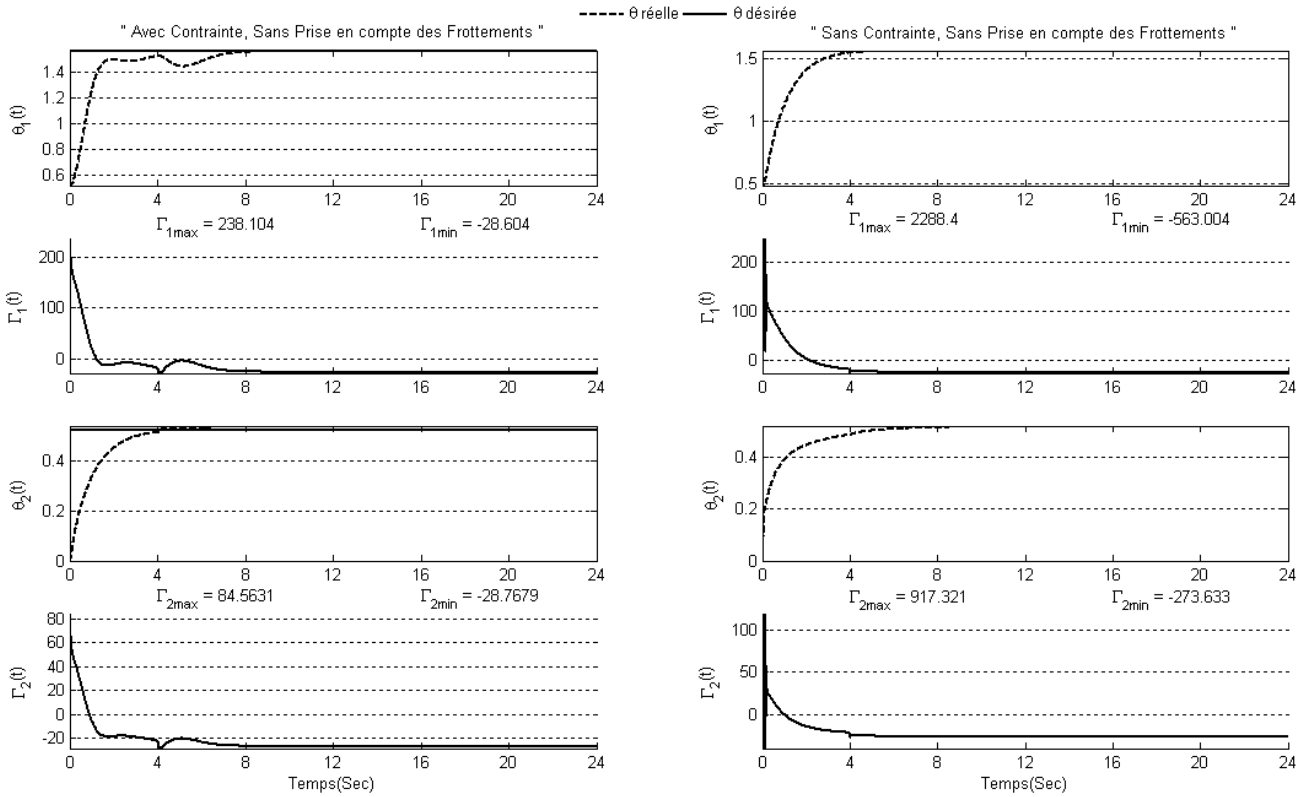

Figure (IV .29) : ***Résultat de régulation de la commande par couple calculé sans prise en compte des frottements (mode glissant) avec charge***

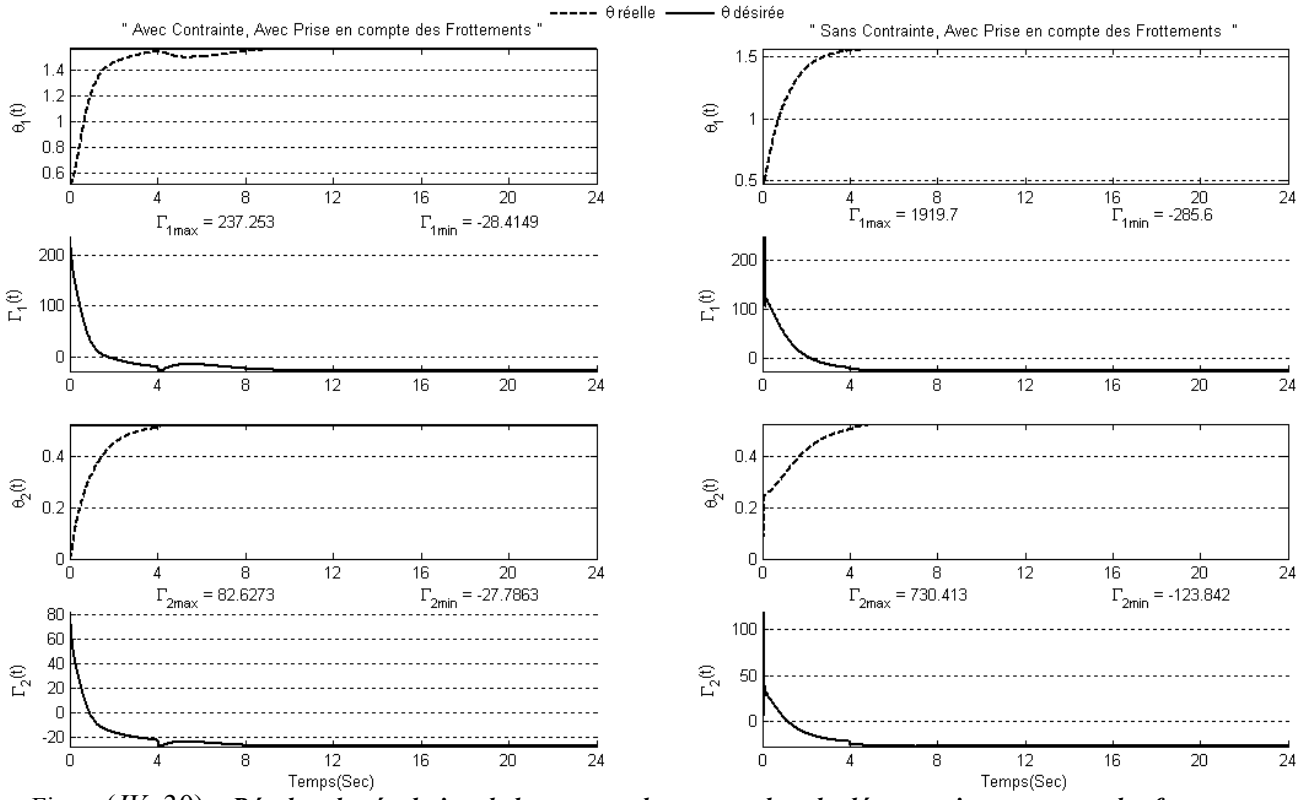

Figure (IV .30) : ***Résultat de régulation de la commande par couple calculé avec prise en compte des frottements (mode glissant) avec charge***

Tableau $(IV.6)$: **Paramètres du régulateur avec les valeurs extrême des couples de la commande par couple calculé en mode glissant avec charge**

Mode \ Gains		K_p	K_v	K_i	$\Gamma_{\min}$	$\Gamma_{\max}$	K_p	K_v	K_i	$\Gamma_{\min}$	$\Gamma_{\max}$
		Avec contrainte					**Sans contrainte**				
Poursuite		**Sans frottements**									
	q_1	**0,01**	**62,37**	**0,01**	**-113,17**	**245,06**	**2,36**	**49,67**	**100,30**	**-543,85**	**1623,6**
	q_2	**281,53**	**78,01**	**52,35**	**-107,94**	**120,97**	**1,77**	**188,07**	**149,85**	**-444,43**	**808,98**
		Avec frottements									
	q_1	**1,77**	**54,55**	**7,49**	**-110,46**	**246,18**	**0,30**	**151,32**	**124,49**	**-778,68**	**1989,9**
	q_2	**253,08**	**61,79**	**147,36**	**-69,60**	**120,50**	**0,01**	**186,51**	**150,00**	**-267,23**	**477,33**
Régulation		**Sans frottements**									
	q_1	**0,60**	**2,94**	**3,53**	**-28,60**	**238,10**	**0,01**	**119,07**	**2,36**	**-563,00**	**2288,4**
	q_2	**0,30**	**24,84**	**150,00**	**-28,77**	**84,56**	**131,09**	**179,47**	**144,72**	**-273,63**	**917,32**
		Avec frottements									
	q_1	**0,01**	**4,51**	**6,75**	**-28,41**	**237,25**	**0,01**	**120,63**	**2,36**	**-285,60**	**1919,7**
	q_2	**0,89**	**18,58**	**149,71**	**-27,79**	**82,63**	**0,01**	**125,71**	**149,71**	**-123,84**	**730,41**

Interprétation:

L'étude de la robustesse de la commande par couple calculé en mode glissant sous l'influence de la charge sur la deuxième articulation conduit au résultats présentés par les figures $(IV.27)$, $(IV.28)$, $(IV.29)$ et $(IV.30)$. En imposant des contraintes sur les couples, dans le cas de poursuite, on peut voir des temps de réponse très courts en régime transitoire et des erreurs nulles en régime permanent, avec une augmentation des valeurs des couples dès l'addition de la charge. Cette augmentation a pour effet d'éliminer des éventuelles erreurs en régime permanent. En outre, pour la régulation, on peut voir des temps de réponse courts en régime transitoire et des erreurs nulles en régime permanent avec une petite déviation dès l'addition de la charge et une augmentation des couples à cette instant pour maintenir une erreur statique nulle en régime permanent. Par contre, sans prise en compte des contraintes, on obtient des valeurs des couples supérieures lorsqu'on considère les frottements par rapport au cas où on les néglige. Ces couples élevés qui servent à compenser l'effet de la charge sont limités par la mise an place des saturations et peuvent ralentir le système ou même entraîner la destruction des moteurs utilisés.

IV.6. Conclusion

Dans ce chapitre, nous avons appliqué les algorithmes génétiques pour résoudre deux problèmes multiobjectifs sous contraintes. Le premier problème est celui de la commande d'un pendule inversé qui peut être considéré comme un problème teste, le deuxième est celui de la commande d'un robot manipulateur basé sur son modèle obtenu par la méthode de Newton-Euler ou la méthode de Lagrange.

Dans le but d'améliorer les performances du robot manipulateur utilisé, nous avons présenté une loi de commande très connue en robotique et nécessitant la connaissance précise du modèle dynamique du robot, malheureusement, les performances obtenues avec cette loi de commande sont limitées par la précision du modèle.

Nous avons constaté, en simulation, qu'une légère modification dans la loi de commande, tenant compte des incertitudes du modèle peut améliorer significativement les performances obtenues en terme de poursuite et de régulation et ce même dans le cas où les perturbations sont prises en compte.

Conclusion générale

Conclusion générale

Nous avons présenté dans cette étude des problèmes d'optimisation multiobjectifs qui sont souvent rencontrés dans le domaine de l'identification et commande. Nous avons vu que cette problématique est abordée par la communauté scientifique suivant deux approches. La première approche tente de ramener un problème multiobjectif à un problème simple objectif au risque d'enlever toute signification au problème. La deuxième approche adopte un point de vue plus global, prenant en compte l'ensemble des critères et en utilisant la notion de dominance au sens de Pareto. Nous avons également constaté que ce domaine scientifique utilise abondamment les algorithmes génétiques pour apporter une réponse aux problèmes d'optimisation. Les premières recherches ont simplement modifié l'heuristique de sélection des algorithmes génétiques. Par la suite, une heuristique de partage a été introduite pour répartir les solutions sur l'ensemble Pareto optimal. Récemment, une population externe, appelée archive, sert à stocker les meilleurs individus. Cette archive exige la mise en place d'une stratégie de mise à jour et de réutilisation de son contenu fondée sur des heuristiques de remplacement. Les derniers travaux proposent l'utilisation des méthodes inspirées des stratégies d'évolution. Cette richesse d'idées pour augmenter l'efficience de ces méthodes a pourtant un prix : l'augmentation du nombre de paramètres de réglage et la complexification de l'écriture des algorithmes.

Le choix de l'algorithme génétique multiobjectif est le résultat d'un travail de revue des méthodes d'optimisation, dont leurs avantages et inconvénients ont été analysés par rapport aux caractéristiques particulières des problèmes rencontrés en robotique (non linéarités et discontinuités possibles des fonctions objectives, difficulté à obtenir leurs dérivées, nécessité fréquente de faire appel à des méthodes numériques, etc.). Il est nécessaire de prendre en compte le caractère multiobjectif dans l'approche des problèmes où il existe de multiples aspects à améliorer. Notre but était de trouver des compromis, par exemple entre performance maximale et coût minimal de temps. Percevoir les relations entre les objectifs nous aide à mieux comprendre le problème, et ainsi à obtenir la solution optimale.

Dans ce mémoire, nous avons abordé les problèmes d'optimisation multiobjectifs sous plusieurs aspects. D'abord, nous avons examiné l'importance du problème d'identification du modèle dynamique en boucle fermée. L'identification en boucle fermée devrait être considérée dans les situations suivantes:

1. Processus instables en boucle ouverte.
2. Un contrôleur existe déjà.

3. L'amélioration de l'exécution du système en boucle fermée.

Dans le but de mieux évaluer la robustesse des lois de commande, il apparaît indispensable de disposer d'un modèle du robot manipulateur permettant de reproduire fidèlement son comportement pour des configurations diverses. Dans ce mémoire, nous avons présenté une approche permettant d'identifier un tel modèle.

Pour notre application sur le problème d'identification d'un robot à deux degrés de liberté, deux modèles ont été présentés, modèle direct qui se base sur la minimisation des moindres carrés d'erreur de sortie et un modèle inverse qui se base sur la minimisation des moindres carrés d'erreur d'entrée. L'analyse de l'identifiabilité et de la sensibilité ont été facilement faites parce que le modèle est linéaire par rapport aux paramètres. L'estimation du vecteur des paramètres a été faite avec minimisation des moindres carrés d'erreur d'entrées.

L'objectif de l'identification en boucle fermée est d'obtenir, pour un contrôleur donné, un modèle dynamique permettant la meilleure description du comportement du système en boucle fermée. L'identification en boucle fermée, à condition que des algorithmes appropriés soient employés et avec un choix précis du signale d'excitation, permet d'obtenir un meilleur modèle pour la conception du contrôleur.

La méthode proposée, qui s'appuie sur une formulation linéaire en fonction des paramètres, a été évaluée en simulation. Les positions articulaires, les vitesses, les accélérations et les couples ont été calculés tandis que le robot suive une trajectoire excitante contenant d'une part des mouvements lents (pour l'estimation des paramètres de frottements) et d'autre part des dynamiques rapides (pour l'estimation des paramètres inertiels). Ces trajectoires ont été pré-calculées de façon à assurer un bon conditionnement de la matrice d'observation. L'identification est réalisée en boucle fermée c'est-à-dire que les mesures nécessaires à l'identification sont prises alors que le robot, asservi par un régulateur PID, suive les trajectoires excitantes.

Une étude comparative a été menée sur le problème d'identification, en considérant les deux critères importants pour un algorithme multiobjectif : se rapprocher le plus possible du front de Pareto et obtenir une bonne dispersion des solutions sur ce front. Les résultats de cette étude montrent que notre algorithme a donné des résultats équivalents ou meilleurs, pour un coût de calcul moindre à ceux de l'algorithme NSGA-II mais la diversité des solutions est meilleure pour ce dernier.

Les méthodes d'identification des erreurs d'entrées sont bien adaptées pour l'identification du modèle dynamique en boucle fermée. Ces méthodes exigent la connaissance du contrôleur, et comme les méthodes d'identification en boucle ouverte, le modèle identifié en boucle fermée devrait être validé. Plusieurs critères pour la validation du modèle dans la boucle fermée ont été retenus.

Les algorithmes génétiques utilisés dans ce stade prouvent leurs convergences vers la solution Pareto-optimale, et la validation du modèle obtenu permet de mesurer sa fiabilité.

La deuxième étude concerne l'application des algorithmes génétiques au problème de commande. Nous avons choisi en premier lieu le pendule inversé comme plate-forme pour illustrer ces algorithmes. Le pendule inversé est un système instable et sous-actionné (c'est à dire, il a moins d'entrées que les degrés de liberté). Il présente des défis considérables de conception de commande et donc approprié pour examiner la performance de la technique d'optimisation utilisée. En outre, cette phase nous permet d'examiner notre algorithme et de mettre en évidence son efficacité pour l'obtention des meilleurs contrôleurs tenant compte des contraintes sur les entrées de commande.

Ensuite, nous avons étudié la commande par couple calculé pour le contrôle d'un robot manipulateur à deux degrés de liberté. Après construction du modèle dynamique avec la méthode de Lagrange ou celle de Newton Euler, l'étape suivante était de chercher à introduire une loi de commande basée sur ce modèle pour calculer les couples prévus d'un mouvement articulaire désiré.

En réalité, nous n'avons jamais une connaissance exacte du modèle du robot en raison de nombreux problèmes liés à la formulation du modèle. Parmi eux, il y'a deux incertitudes, dans les applications robotiques, qui sont les masses inconnues des corps dû aux perturbations de la charge utile et aux coefficients inconnus des frottements. Une façon de traiter ces types d'incertitudes paramétriques était d'employer la commande du couple calculé avec une certaine évaluation des paramètres inconnus au lieu des paramètres effectifs.

Cette stratégie de commande est motivée par la raison qu'on pourrait s'attendre à de meilleures performances si toutes les non linéarités sont compensées et les paramètres du contrôleur sont bien ajustés surtout quand on utilise la surface de glissement.

Afin de réduire les erreurs causées par l'inadéquation du modèle avec le système réel, plusieurs adaptations dans la loi de commande ont fait l'objet de notre étude. La loi de commande introduite dans chaque cas est basée sur la compensation des effets de la gravité, des forces de centrifuge et Coriolis, avec la modification du critère à optimiser soit en utilisant des erreurs quadratiques ou des erreurs filtrés (surface de glissement). Les résultats de simulation obtenus en ce mémoire montrent une amélioration significative de l'exactitude de poursuite et de régulation à chaque fois qu'on introduit des termes de compensations et ça avec la prise en compte des contraintes sur les couples calculés.

La raison d'employer la commande par couple calculé est d'étudier l'influence de chaque effet (la force de la gravité, les forces de centrifuges et de Coriolis, et les frottements) sur les performances du système.

Les algorithmes génétiques se sont montrés un outil extrêmement efficace, son principal avantage étant l'aide qu'il apporte à la compréhension du problème qu'on souhaite résoudre, ce qui était mis en évidence dans ce mémoire. L'inconvénient de cette méthode est le coût de calcul nécessaire pour la découverte de solutions efficaces, qui peut être supérieur à celui des outils traditionnels (quand ces outils sont capables de résoudre le problème). Ce coût supérieur est surtout dû à la plus grande généralité de ces algorithmes.

[1] J.C. Cuilioli, Introduction a l'optimisation. Ellipse 1994.

[2] J. A. Nelder and R. Mead, A Simplex Method for Function Minimization, Comput. J., vol. 7, p. 308-313, 1965.

[3] F. Messine, Méthodes d'Optimisation Globale basées sur l'Analyse d'Intervalle pour la Résolution de Problèmes avec Contraintes, Thèse de doctorat de l'Université Paul Sabatier Toulouse III, 1997.

[4] D. E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley, Reading, Massachusetts, 1989.

[5] R. Cerf, Une Théorie Asymptotique des Algorithmes Génétiques, Thèse de doctorat de l'Université Montpellier II, 1994.

[6] B. Hajek, Cooling Schedules for Optimal Annealing, Mathematics of Operations Research, vol. 13, n° 2, p. 311-329, 1988.

[7] G. S. Fishman, Monte-Carlo: Concepts, Algorithms and Applications, Springer-Verlarg, 1997.

[8] F. Glover and M. Laguna, Tabu Search, Kluwer Academic Publishers, 1997.

[9] N.D. Cunha and E. Polak. Constrained minimization under vector-evaluated criteria in finite dimensional spaces. Journal of Mathematical Analysis and Applications, 19(1):103 124, 1967.

[10] I. Othmani. Optimisation multicritère : Fondements et Concepts. PhD thesis, Université de Grenoble, 1998.

[11] M. Ehrgott. Multicriteria optimization. In Lecture Notes in Economics and Mathematical Systems, volume 491. Springer, 2000.

[12] Y. Collette et P. Siarry. Optimisation multi-objectif. Eyrolles, 2002.

[13] E.L. Ulungu and J. Teghem. Multi-objective combinatorial optimization: a survey. Journal of Multi-Criteria Decision Analysis, 3:83{104, 1994.

[14] K. Miettinen. Nonlinear multiobjective optimization. Kluwer Academic Publishers, 1999.

[15] M. Ehrgott and X. Gandibleux. A survey and annotated bibliography of multiobjective combinatorial optimization. OR Spektrum, 22:425{460, 2000.

[16] K. Deb. Multi-objective optimization using evolutionary algorithms. John Wiley, 2001.

[17] Carlos. A. C. Coello, An Empirical Study of Evolutionary Techniques for Multiobjective Optimization in Engineering Design, Ph. D. thesis, Department of Computer Science, Tulane University, New Orleans, 1996.

[18] D. A. Van Veldhuizen, Multiobjective, Evolutionary Algorithms : Classification, Analyses and New Innovation, Air Force Institute of Technology, United States, 1999.

[19] A. Charnes and W. Cooper, Management Models and Industrial Applications of Linear Programming, vol. 1, John Wiley, New-York, 1961.

[20] J. P. Ignizio, The Determination of a Subset of Efficient Solutions via Goal Programming, Computing and Operations Research 3, p.9-16, 1981.

[21] J.H. Holland. Adaptation in natural and artificial systems. PhD thesis, University of Michigan Press, 1975.

[22] D. E. Goldberg. Genetic algorithms in search, optimization and machine learning. Addison Wesley, 1989.

[23] H-P. Schwefel. Numerical optimization of computer models. Wiley, Chichester, 1981.

[24] D. Fogel. Evolutionary Computation: Toward a New Philosophy of Machine Intelligence (second edition). IEEE Press, 2000.

[25] J. H. Holland. Outline for a logical theory of adaptive systems. Journal of the association of computing machinery, 3, 1962.

[26] K. A. DeJong. The Analysis of the Behavior of a Glass of Genetic Adaptive Systems. PhD thesis, University of Michigan, Ann Harbor, 1975. Dissertation Abstract International, 36(10), 5140B. (University Microfilms No 76-9381).

[27] Z. Michalewicz. Genetic algorithms and Data Structures for Evolution Programs. Springer-verlag, 1992.

[28] D. E. Goldberg and J. Richardson. Genetic algorithms with sharing for multi-modal function optimization. In J. J. Grefenstette, editor, Proceedings of the 2 International Conference on Genetic Algorithms, pages 41—49. Lawrenice Erlbaum Associates, 1987.

[29] K. Deb and D. E. Goldberg. All investigation of niche and species formation in genetic function optimization. In J. D. Schaffer, editor, Proceedings of the 3 International Conference on Genetic Algorithms, pages 42 50. Morgan Kaufman, 1989.

[30] X. Yin and N. Germay. A fast genetic algorithm with sharing scheme using cluster analysis methods in multimodal function optimization. In Artificial Neural Nets and Genetic Algorithrris, pages 450 457, Wien, 1993. Springer Verlag.

[31] J. M Alliot. Techniques d'optimisation stochastique appliquées aux problèmes du trafic aérien. PhD thesis, ENA, 1996.

[32] A. Homaifar, S. H. Y. Lai, and X. Qi. Constrained optimization via genetic algorithms. Simulation, 62(4):242 254, 1994.

[33] J.A. Joines and C.R. Houck. On the use of non- stationary penalty functions to solve nonlinear constrained optimization problems with GAs. In Z. Michalewicz, J. D. Schaffer, H.-P. Schwefel,

D. B. Fogel, and H. Kitano, editors, Proceedings of the First IEEE International Conference on Evolutionary Computation, pages 579 584. IEEE Press, 1994.

[34] J. D. Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In J. J. Grefenstette, editor, Proceedings of the 1st International Conference on Genetic Algorithms, pages 93 100. Laurence Erlbaum Associates, 1985.

[35] P. Hajela alld C. Y. Lin. Genetic search strategies in multicriterion optimal design. Structural Optimization, 4:99-107, 1992.

[36] C. M. Fonseca and Peter J. Fleming. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In Genetic Algorithms: Proceedings of the Fifth International Conference, pages 416 423. Morgan Kaufmann, 1993.

[37] N. Srinvas and K. Deb. Multi-objective function optimization using non-dominated sorting genetic algorithms. Evolutionary Computation, 2(3):221 248, 1994.

[38] J. Horn, N. Nafploitis, and D. Goldberg. A nichad pareto genetic algorithm for multiobjective optimization. In Z. Michalewicz, J. D. Schaffer, H.-P. Schwefel, D. B. Fogel, and H. Kitano, editors, Proceedings of the First IEEE International Conference on Evolutionary Computation, pages 82 88. IEEE Press, 1994.

[39] D. Goldberg and K. Deb. A comparison of selection schemes used in genetic algorithms. Tri G. J. E. Rawlins, editor, Foundations of Genetic Algorithms, pages 69-93, San Mateo, 1991. Morgan Kaufmann.

[40] C. K. Oei, D. Goldberg, and S. J. Chang. Tournament selection, niching, and the preservation of diversity. Technical report, University of Illinois, Urbana-Champaign, 1991.

[41] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II. In M. Schoenauer and al., editors, Proceedings of the 6th Conference on Parallel Problem Solving from Nature, pages 849 858. Spriilger Verlag, 2000.

[42] F. Jimenez, J. L. Verdegey, and A. F. Gomez-Skarmeta. Evolutionary techniques for constrained multi-objective optimization problems. In GECCO99editors, editor, Proceedings of the Workshop on multi-Criterion Optimization Using Evolutionary Methods held at GEGCO-1999. Morgan Kaufmann, 1999.

[43] C. M. Fonseca and Peter J. Fleming. Multi-objective optimization and multiple constraint handling with evolutionary algorithms-Part I: A unified formulation. IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans, 28(1) :26- 37, 1998.

[44] L. Ljung. System Identification, theory for the user. Prentice Hall, 1999.

[45] M. Gautier and P. Poignet. Identification non-linéaire continue en boucle fermée des paramètres physiques des systèmes mecatroniques par modèle inverse et moindres carrés d'erreur d'entrée. Journées Identification et Modélisation Expérimentale, Vandoeuvre-lès-Nancy, 2001.

[46] M. Villain, Automatique 2 – Systèmes asservis linéaires. Ellipses, 1996.

[47] L. Sciavicco and B. Siciliano, Modeling and Control of Robot Manipulators. New York: McGraw-Hill, 1996.

[48] K. S. Fu, R. C. Gonzales, and C. S. G. Lee, Robotics: Control, Sensing, Vision, and Intelligence. New York: McGraw-Hill, 1987.

[49] C. H. An, C. G. Atkeson, and J. M. Hollerbach, Model-Based Control of a Robot Manipulator. Cambridge, MA: MIT Press, 1988.

[50] K. Kozlowski, Modeling and Identification in Robotics. New York: Springer-Verlag, 1998.

[51] J. Nethery and M.W. Spong, "Robotica: A mathematica package for robot analysis," IEEE Robot, Automat. Mag., vol. 1, pp. 13-20, Mar. 1994.

[52] P. I. Corke, "A robotics toolbox for MATLAB," IEEE Robot Automat. Mag., vol. 3, pp. 24-32, Mar. 1996.

[53] B. Armstrong, "On finding exciting trajectories for identification experiments involving systems with nonlinear dynamics," int. J. Robot. Res., vol. 8, no. 6, pp. 28-48, 1989.

[54] M. M. Olsen and H. G. Petersen, "A new method for estimating parameters of a dynamic robot model," IEEE Trans. Robot. Automat., vol. 17, pp. 95-100, Feb. 2001.

[55] G. Calafiore, M. Indri, and B. Bona, "Robot dynamic calibration: Optimal excitation trajectories and experimental parameter estimation," J. Robot Syst, vol. 18, no. 2, pp. 55-68, 2001.

[56] K. R. Kozlowaki and P. Dutkiewicz, "Experimental identification of robot and load dynamics," in Proc. 13th Triennial IFAC World Congr., San Francisco, CA, 1996, pp. 397-402.

[57] M. Gautier and W. Khalil, "On the identification of the inertial parameters of robots," in Proc. IEEE Conf. Dec. Control, Austin, TX, 1988, pp. 2264–2269.

[58] M. Gautier andW. Khalil, "Exciting trajectories for the identification of base inertial parameters of robots," Int. J. Robot. Res., vol. 11, no. 4, pp. 362–375, 1992.

[59] J. Swevers, C. Ganseman, D. B. Tukel, J. de Schutter, and H. van Brussel, "Optimal robot excitation and identification," IEEE Trans. Robot. Automat, vol. 13, pp. 730–740, Oct. 1997.

[60] R. H. Brown, S. C. Schneider, and M. G. Mulligan, "Analysis of algorithms for velocity estimation from discrete position versus time data," IEEE Trans. Ind. Electron., vol. 39, pp. 11–19, Feb. 1992.

[61] H. Berghuis and H. Nijmeijer, "Robust control of robots via linear estimated state feedback," IEEE Trans. Automat. Contr., vol. 39, pp. 2159–2162, Oct. 1994.

[62] A. Gelb, Applied Optimal Estimation. Cambridge, MA: MIT Press, 1996.

[63] P. R. Bélanger, “Estimation of angular velocity and acceleration from shaft encoder measurements,” in Proc. IEEE Int. Conf. Robotics Automation, Nice, France, 1992, pp. 585–592.

[64] B. Armstrong-Hélouvry, P. Dupont, and C. C. de Wit, “A survey of models, analysis tools and compensation methods for the control of machines with friction,” Automatica, vol. 30, no. 7, pp. 1083—1138, 1994.

[65] S. J. Kim and I. J. Ha, “A frequency-domain approach to identification of mechanical systems with friction,” IEEE Trans. Automat. Contr., vol. 46, pp. 888–893, Nov. 2001.

[66] H. Olsson, K. J. Åström, C. Canudas de Wit, M. Gäfvert, and P. Lischinsky, “Friction models and friction compensation,” Eur. J. Control, vol. 4, no. 3, pp. 176–195, 1998.

[67] C. Canudas de Wit and P. Lischinsky, “Adaptive friction compensation with partially known dynamic friction model,” Int. J. Adapt. Contol Sig. Process., vol. 11, no. 1, pp. 65–80, 1997.

[68] J. Swevers, F. Al-Bender, C. G. Ganseman, and T. Prajogo, “An integrated friction model structure with improved presliding behavior for accurate friction compensation,” IEEE Trans. Automat. Contr., vol. 45, pp. 675–686, Apr. 2000.

[69] D. Kostié, R. Hensen, B. de Jager, and M. Steinbuch, “Modeling and Identification of an RRR-robot,” in Proc. IEEE Conf Decision Control, Orlando, FL, 2001, pp. 1144-1149.

[70] R. H. A. Hensen, G. Z. Angelis, M. J. G. van de Molengraft, A. G. de Jager, and J. J. Kok, “Grey-box modeling of friction: An experimental case study,” Eur. J. Control, vol. 6, no. 3, pp. 258–267, 2000.

[71] R. H. A. Hensen, M. J. G. van de Molengraft, and M. Steinbuch, “Frequency domain identification of dynamic friction model parameters,” IEEE Trans. Contr. Syst Technol., vol. 10, pp. 191-196, Mar. 2001.

[72] B. de Jager, “Improving the tracking performance of mechanical systems by adaptive extended friction compensation,” Control Eng. Pract., vol. 1, no. 6, pp. 1009–1018, 1993.

[73] L. R. Ray. A. Ramasubramanian. and J. Townsend. “Adaptive friction compensation using extended Kalman-Bucy filter friction compensation,” Control Eng. Pract, vol. 9, no.2, pp. 169-179,2001.

[74] A. De Luca, “Feedforward/feedback laws for the control of flexible robots,” in Proc. IEEE int. Conf Robotics Automation, San Francisco, CA, 2000, pp. 233-240.

[75] D. Kostié, B. de Jager, and M. Steinbuch, “Experimentally supported control design for a direct drive robot,” in Proc. IEEE int. Conf Control Applications, Glasgow, U.K., 2002, pp. 186-191.

[76] B.H.M. Bukkems, D. Kostié, A.G. de Jager, and M. Steinbuch, “Frequency domain iterative learning control for direct drive robots,” in Eur. Control Conf, Cambridge, U.K., 2003.

[77] W. Khalil and E. Dombre. Modélisation, identification et commande des robots(2ème Edition Revue et Augmentée). Hermes, 1999

[78] J. Richalet, A. Rault, et R. Pouliquen. Identification des processus par la méthode du modèle. Gordon & Breach, Londres, 1971.

[79] J.C. Trigeassou. Recherche de modèles expérimentaux assistée par ordinateur. Lavoisier, Paris, tec et doc edition, 1988.

[80] W. Khalil and E. Dombre. Modelisation, identification et commande des robots. Hermes, 1999.

[81] M. Gautier and P. Poignet. Identification non linéaire continue en boucle fermée des paramètres physiques de systèmes mécatroniques par modèle inverse et moindre carrés d'erreur d'entrée. In Journées Identification et Modélisation Expérimentale, Nancy-France, 2001.

[82] W. Khalil, D Creusot. SYMORO+: a system for the symbolic modeling of robots, Robotica, Vol. 15, 1997, p. 153-161.

[83] H. Mayeda, K Yoshida, and K Osuka, Base parameters of manipulator dynamic models. IEEE Journal of Robotics and Automation,1990, 6(3):312–321.

[84] M.T. Pham, M. Gautier et Ph. Poignet, "Identification of joint stiffness with bandpass filtering", In: Proc. IEEE Int. Conf. on Robotics and Automation, May 2001, pp. 2867-2872.

[85] BARTO, A.G., SUTTON, R.S., ANDERSON, C.W. - «Neuronlike Adaptive Elements that can Solve Difficult Learning Control Problems » - IEEE Transactions on Systems, Man, and Cybernetics - 1983, Vol. SMC-13, N°. 5 - p. 834-846

[86] C.W. ANDERSON, Learning to Control an Inverted Pendulum Using Neural Networks, IEEE Control System Magazine - avril 1989 - p. 31-37

[87] M. SAERENS, A. SOQUET, Neural Controller Based on Back-propagation Algorithm, IEE Proceedings - F - IEE Eds, 1991 - Vol. 138 - N°. 1 - p. 55-62

[88] C.S. LIN, H. KIM, "CMAC-Based Adaptive Critic Self-Learning Control" - IEEE Transactions on Neural Networks - 1991, Vol. 2, N°. 5 - p. 530-533

[89] S.B. THURN, A. LINDEN, K. MÖLLER, «Adaptive Look-ahead Planning » -Actes des journées NeuroNîmes, Nîmes, 12-16 Novembre 1990 - Nanterre : EC2 Eds, 1990 - p. 405-417.

[90] T. LANGLOIS, Algorithmes d'Apprentissage par Renforcement pour la Commande Adaptative - Thèse de troisième cycle - Université de Technologie de Compiègne, 1992 -p 125.

[91] S. GEVA, J. SITTE, «A Cartpole Experiment Benchmark for Trainable Controllers » - IEEE Control Systems - 1993, Vol. 13, N°. 5 - p. 40-51.

[92] Luc Jaulin, «Commande par espace d'état» , Octobre 2007, p 22-23.

[93] http://php.iai.heig-vd.ch/~mee//seminaires/NAPinv.pdf

[94] J.Y. Luh, M.W. Walker, and R.P. Paul, “On-Line Computational Scheme for Mechanical Manipulators,” Transactions of the ASME Journal of Dynamic Systems, Measurement, and Control, 1980.

[95] B. Armstrong, “Dynamics for robot control: friction modeling and ensuring excitation during parameter identification” Ph.D Thesis, Dept. of Electrical Engineering. Stanford University, May 1988.

[96] Lasky, T.A., Hsia, T.C., Tummala, R.L., Odrey, N.G. “Robotics” *The Electrical Engineering Handbook* Ed. Richard C. Dorf Boca Raton: CRC Press LLC, 2000

[97] M. W. Spong and M. Vidyasagar, Robot Dynamics and Control, New York: Wiley, 1989..

[98] J. J. Craig, Introduction to Robotics, Reading, Mass.: Addison-Wesley, 1989.

[99] R. Ortega, M.W. Spong,“Adaptive motion control of rigid robots: A tutorial,” Automatica, vol. 25, 1989, p. 877-888.

[100] N. Sadegh, R. Horowitz,“ Stability and robustness analysis of a class of adaptive controllers for robotic manipulators,” The International Journal of Robotics Research, vol. 9, 1990, p. 74-94.

[101] H. Berghuis “Model-based robot control: from theory to practice,” CIP-DATA KONINKLIJKE BIBLIOTHEEK, The Netherlands, juni 1993.