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

Université de Batna 2 – Mostefa Ben Boulaïd
Faculté de Technologie
Département d'Electronique

Thèse

Présentée pour l'obtention du titre de :
Docteur en Sciences en Electronique
Option :
Contrôle

Sous le Thème :

Contribution à la planification des trajectoires par l'utilisation des métaheuristiques pour la commande des robots manipulateurs

Présentée par :

Nadjib ZERROUKI

Devant le jury composé de :

Dr. Lamir SAIDI	Prof.	Université de Batna 2	Président
Dr. Noureddine GOLEA	Prof.	Université d'Oum El Bouaghi	Rapporteur
Dr. Nabil BENOUDJIT	Prof.	Université de Batna 2	Co-Rapporteur
Dr. Chawki MAHFOUDI	Prof.	Université d'Oum El Bouaghi	Examinateur
Dr. Abdelbaki DJOUAMBI	Prof.	Université d'Oum El Bouaghi	Examinateur

Remerciements

Le travail présenté dans cette thèse a été réalisé au sein du Laboratoire de Recherche LGEA (**L**aboratoire de **G**énie **E**lectrique et **A**utomatique) de l'université de LARBI BEN M'HIDI -OUM EL BOUAGHI-.

Tout d'abord, je tiens à exprimer ma sincère gratitude et révérence à mes rapporteurs de thèse Monsieur Noureddine GOLEA, Professeur à l'université d'Oum El Bouaghi, et Monsieur Nabil BENOUDJIT, Professeur à l'université Batna 2, pour leurs soutiens continus, pour leurs patiences, leurs motivations. Leurs conseils m'ont aidé tout au long de la recherche et de la finalisation de cette thèse.

Je tiens à remercier Monsieur Lamir SAIDI, Professeur à l'université Batna 2, pour m'avoir fait l'honneur de présider le jury de ma thèse.

Je remercier aussi Messieurs Chawki MAHFOUDI et Abdelbaki DJOUAMBI, Professeurs à l'université d'Oum El Bouaghi, pour avoir accepté d'être membres de jury de ma thèse.

Je tiens également à remercier tous les membres du LGEA (**L**aboratoire de **G**énie **E**lectrique et **A**utomatique) à l'université de Larbi Ben M'hidi –Oum El Bouaghi et surtout Messieurs Kamel BARRA et Djamel RAHEM**,** Professeurs à l'université d'Oum El Bouaghi, pour leurs encouragements et leurs motivations.

Enfin, je remercier toute personne ayant contribué de près ou de loin à la finalisation de cette thèse.

Résumé

La synthèse de la loi de commande des robots manipulateurs, qui peuvent réaliser des taches simple comme le chargement, le polissage et le positionnement, ou complexes comme la robotique de service et la robotique médicale, se décompose en deux niveaux. Le premier niveau correspond à la planification des trajectoires dont l'objectif est de générer les consignes pour les boucles de commande de chaque articulation selon les contraintes imposées par la tâche à réaliser et par l'environnement de travail. Le deuxième niveau correspond à la loi de commande pour l'asservissement des robots manipulateurs. La planification des trajectoires consiste à calculer les positions articulaires du robot en fonction du temps de manière que l'organe terminal se déplace d'une situation initiale à une situation finale, en assurant que les trajectoires des articulations commencent et se terminent au même moment respectant un certain nombre de contraintes. Les contraintes dépendent généralement des caractéristiques du robot, de l'environnement et du type de tâche à réaliser. En l'occurrence, les contraintes relatives au robot résultent de ces caractéristiques cinématiques et dynamiques. Les contraintes émanant de l'environnement concernent plus particulièrement l'évitement des obstacles. Ce problème de génération des trajectoires peut être transformé en un problème d'optimisation avec contraintes. Cette étude vise à résoudre ce problème dans le domaine de la commande des robots manipulateurs par l'utilisation des métaheuristiques afin d'obtenir des bonnes performances. Les critères à satisfaire pendant la résolution de ce problème concernent la longueur de la trajectoire parcourue par le robot, le temps nécessaire à son exécution, et l'évitement des singularités et d'obstacles.

Mots-clés : planification des trajectoires**,** méta-heuristiques, robot manipulateurs.

Abstract

The synthesis of control law of manipulators, which can perform simple tasks like loading, polishing, positioning, or complex ones like service robotics and medical robotics, can be divided into two levels. The first level corresponds to the planning of the trajectories whose objective is to generate the signals for the control loops of each articulation according to the constraints imposed by the task to be performed and by the working environment. The second level concerns the manipulators' control law. Trajectory planning consists of calculating the robot's joint positions as a function of time so that the end effector moves from an initial situation to a final situation, ensuring that the joint trajectories begin and end at the same time respecting a certain number of constraints.

These constraints usually depend on the robot's characteristics, the environment and the type of task to be performed. In this case, the constraints relating to the robot derive from its kinematic and dynamic characteristics. The constraints emanating from the environment concern more particularly obstacle avoidance. The trajectory planning problem can be transformed to an optimization problem with constraints. In order to obtain better performances, this study aims at treating this problem in the field of robots' control, using metaheuristic approaches. The criteria to be met when dealing with this problem cover the trajectory's length, the time required for its execution, and singularities and obstacle avoidance.

Keywords: trajectory planning, metaheuristic approaches, robot manipulators.

ملخص

يمكن تقسيم توليف قانون مراقبة الروبوتات، التي تستطيع القيام بمهام بسيطة مثل التحميل ، التلميع ، واعادة التموضع ، أو مهام معقدة ، مثل روبوتات الخدمة الآلية و الروبوتات الطبية ، إلى مستويين. المستوى الأولى يتعلق بتخطيط المسار الذي يهدف إلى توليد تعليمات لحلقات التحكم لكل مفصل وفقا للقيود المفروضة من قبل المهمة التي يتعين القيام بها وبيئة العمل. المستوى الثاني يتعلق بقانون المراقبة الخاص بالروبوت. تخطيط المسار يتضمن حساب المواضع لكل مفصل بدلالة الزمن بحيث تتحرك نهاية الروبوت من الموضع الأولي إلى الموضع النهائي ، مما يضمن أن مسارات كل المفاصل تبدأ وتنتهي في نفس الوقت وذلك مع احترام القيود المفروضة. هذه القيود تتعلق عموما بخصائص الروبوت ، البيئة، ونوع المهمة التي يتعين القيام بها. في هذه الحالة ، القيود المرتبطة بالروبوت ناتجة أساسا من خصائصه الحركية والديناميكية . فيما يخص القيود المنبثقة من البيئة فتتعلق بشكل خاص بتجنب العقبات. في هذا الشأن، يمكن تحويل مشكلة تخطيط المسارات إلى مشكلة التحسين مع قيود مفروضة. هذه الدراسة تهدف إلى معالجة هذه المشكلة في مجال مراقبة الروبوتات، من خلال استخدام عدة طرق مثل الخوارزميات الجنية، محاكات الأسراب و التخمير المحاكي، من أجل الحصول على أداء جيد. المعايير التي يتعين احترامها أثناء حل هذه المشكلة تشمل طول المسار الذي يسلكه الروبوت، الوقت اللازم لتنفيذه وتجنب العقبات و الحالات التي لا يمكن تطبيقها.

الكلمات المفتاحية: تخطيط المسار، الخوارزميات الجنية، محاكات الأسراب، التخمير المحاكي

Table des matières

LISTE DES ABREVIATIONS ET DES SYMBOLES

Abréviations

CMU : Carnegie-Mellon University ;

NURBS : Non-Uniforme Rational B-Spline

Obs : Obstacle ;

Rot : Rotation ;

Trans : Translation ;

QPSO : Quantum Particle Swarm Optimization ;

Symboles

a_j : vecteur unitaire selon l'axe z_j ;

$B_j^p(u)$:fonction B-spline de base de degré p ;

ΔE : différence d'energie;

D_l : distance linéaire de la source à la cible.

D_t : distance totale de la source à la cible.

F_j : résultante de forces extérieures sur le corps C_j ;

f_e : résultante du torseur dynamique appliqué sur le corps C_j ;

f_{ej} : résultante du torseur dynamique appliqué par le corps C_j sur l'environnement extérieur;

F_{sj} : paramètres de frottement sec ;

F_{vj} : paramètres de frottement visqueux ;

g : accélération de la pesanteur ;

Ia_j : moment d'inertie de l'actionneur j ;

K_B : constant de Boltzmann ;

M_j : masse du corps C_j ;

MS_j : premier moment du corps C_j autour de l'origine de son repère R_j ;

$\mathbf{M}_j$: moment résultant des efforts extérieurs appliqués sur le corps C_j autour de O_j ;

m_j : moment du torseur dynamique appliqué sur le corps C_j ;

m_{ej} : moment du torseur dynamique appliqué par le corps C_j sur l'environnement extérieur;

η_{ij} : visibilité du site *j* pour le site *i*

τ_{ij} : l'intensité de la trace de phéromone sur le chemin *ij* ;

q : vecteur des positions articulaires ;

q_s : configuration initiale ;

q_f : configuration finale ;

$\dot{q}$: vecteur des vitesses articulaires ;

$\ddot{q}$: vecteur des accélérations articulaires ;

V_j : vitesse de translation de l'orgine O_j ;

$\dot{V}_j$: accélération de translation l'origine O_j ;

ω_j : vitesse de rotation du corps C_j ;

$\dot{\omega}_j$: accélération de rotation du corps C_j

ξ : paramètre d'ajustement des forces d'attraction;

Table des figures

Liste des tableaux

Introduction Générale

Introduction générale

1. Problématique

Tous les robots, mobiles ou manipulateurs, sont conçu pour fonctionner dans des environnements inconnus ou dans des environnements qui changent continuellement. Par conséquent, pour accomplir leurs tâches, simples ou complexes, elles nécessitent une étape très importante. Cette étape consiste à générer une trajectoire qui leurs permettent de naviguer sans entrer en collision avec les obstacles. En outre, la trajectoire devrait commencer à partir d'une situation initiale et atteindre une situation finale. L'objectif principal de la planification des trajectoires pour les robots mobiles est de les rendre capables de réagir à toute nouvelle situation, ainsi d'augmenter leurs autonomies, alors que pour les robots manipulateurs, la planification est nécessaire pour augmenter leur vitesse et leur précision et par conséquent pour satisfaire l'exigence de productivité et de sécurité.

En général, les approches utilisées pour résoudre le problème de planification des trajectoires peuvent être classées en deux classes: les approches conventionnelles et les approches métaheuristiques. D'une part, la classe des approches conventionnelles couvre les algorithmes bug [Lumelsky 86, 87], le champ potentiel [Khatib 86], les roadmaps [Canny 88] et la décomposition cellulaire [Choset 86]. Généralement, ces méthodes dépendent des modèles mathématiques complexes. En plus, elles présentent des inconvénients communs tels que la limitation à l'espace bidimensionnel, l'incapacité de s'échapper les minima locaux et l'incomplétude. Malgré le temps d'exécution élevé, elles produisent des trajectoires longues et rugueuses, résultant d'une compilation des formes linéaires. Souvent, ces trajectoires ne peuvent être exécutées, par ces robots, sans violer les contraintes dynamiques et cinématiques. D'autre part, la classe des approches métaheuristiques regroupe des méthodes artificielles telles que les algorithmes évolutionnaires comme les algorithmes génétiques [Holland 75], la programmation génétique [Koza 92], la programmation évolutive [Fogel 66] et les stratégies évolutives [Rechenberg 65], les colonies de fourmis [Colorni 92] et les essaims particulaires [Kennedy 95, Shi 98]. Ces approches émergent pour surmonter les défauts de la classe des approches conventionnelles. Elles sont généralement des approches à base population ce qui leur permet de faire une exploration intensive de l'espace de recherche. De plus, cette classe des approches peut traiter le problème des minima locaux et de la classe élevée de l'espace de recherche et elle n'a pas besoin de gradient, des dérivées élevées et d'estimation

de la solution initiale. En plus, elle génère des trajectoires lisses qui conviennent mieux aux robots.

2. Contributions

L'objectif de ce travail est de développer différentes approches pour la planification des trajectoires basées sur la description du chemin en utilisant un type amélioré des courbes spline appelé NURBS. Le contrôle local de ces courbes permet de planifier des trajectoires lisses et simplifie la façon de les faire passer à travers un ensemble de points imposés. Cela permet également d'améliorer la précision des trajectoires planifiées sous différentes contraintes telles que le temps d'exécution et la consommation d'énergie.

Ce travail vise à planifier des trajectoires hors ligne dans un environnement libre. Ces trajectoires doivent obéir à différentes exigences telles que le temps d'exécution minimal et la consommation d'énergie minimale. De plus, les trajectoires générées doivent suivre un ensemble de points intermédiaires sans conduire à aucune configuration singulière pour le robot.

Un autre objectif est d'appliquer les approches développées au problème de la génération de trajectoire dans des environnements avec obstacles. En plus des exigences de temps minimal, d'énergie minimale et d'évitement de singularités, les trajectoires générées doivent éviter les obstacles en assurant le passage par les points intermédiaires. Pour cette raison, les approches proposées sont basées sur la déformation locale de la trajectoire pour résoudre le problème d'évitement d'obstacles.

3. Plan de la thèse

Cette introduction est composée de quatre chapitres. Dans le premier chapitre, nous présentons la méthode de description géométrique des robots. Nous introduisons aussi les modèles nécessaires pour définir les configurations des robots et leurs vitesses en plus du modèle pour calculer les couples requises pour leurs mouvements. Après, nous montrons la différence entre chemin et trajectoires et nous présentons les différents types de trajectoires et les méthodes pour éviter les collisions. Dans le deuxième chapitre nous décrivons les approches conventionnelles pour la planification des trajectoires en basant sur les classes suivantes : les algorithmes bug, Les roadmaps, les diagrammes de Voronoi, le graphe de visibilité, le champ potentiels et la décomposition cellulaire. Dans le troisième chapitre, nous introduisons deux différents types des métaheuristiques. Le premier type inclue les approches à base population qui sont les algorithmes génétiques, les essaimes particulaires et les colonies de fourmis. Le deuxième type couvre les approches, à base individu, qui

sont la recuit simulé et la recherche tabou. Dans le quatrième chapitre, nous présentons nos approches qui sont basé principalement sur la description des chemins à l'aide des courbes NURBS. Ensuite nous montrons les résultats et les discussions de l'application de ces approches pour la planification des trajectoires dans trois situations différentes. Ces situations concernent la planification des trajectoires dans un environnement libre, avec un seul obstacle et avec deux obstacles respectivement. En conclusion nous rappelons nos contributions et nous donnons nos perspectives.

4. Publication

ZERROUKI Nadjib, GOLÉA Noureddine, BENOUDJIT Nabil. Particle Swarm Optimization of Non Uniform Rational B-Splines for Robot Manipulators Path Planning. *Periodica Polytechnica Electrical Engineering and Computer Science*, v. 61, n. 4, p. 337-349, 2017.

ZERROUKI Nadjib, GOLÉA Noureddine, BENOUDJIT Nabil. Genetic Algorithm Based High Performance Control for Rigid Robot Manipulators. *Journal of Computer Science and Control Systems*, v. 4, n. 2, p.73-84, 2011.

Chapitre 1

Robotique et Planification des Trajectoires

1.1. Introduction

Dans ce chapitre, nous abordons les différents modèles de description des robots, comme le modèle géométrique, pour la description de la position du robot, le modèle cinématique, pour décrire son vitesse, et le modèle dynamique, pour le calcul des couples requises pour son mouvement. Puis, nous présentons les deux types de planification de trajectoires. Ces deux types sont reliés à l'espace où le chemin est décrit, soit dans l'espace articulaire ou dans l'espace cartesian. Ensuite, nous décrivons les critères les plus utilisés pour l'optimisation des trajectoires. Finalement, nous présentons l'aspect le plus important dans le domaine de planification des trajectoires qui est l'évitement des collisions.

1.2. Description géométrique des robots

La méthode la plus connue pour décrire la géométrie des robots est celle proposée par [Denavit 55]. Elle est basée sur l'utilisation de quatre paramètres pour définir chaque articulation par rapport à la précédente. De plus, deux paramètres sont utilisés dans le cas où plus de deux articulations sont présentées. Bien que cette notation rend la méthode plus puissante lorsqu'elle ne traite que deux articulations, elle mène à certaines ambiguïtés une fois prolongée à plus de deux.

Afin de rendre cette méthode générale aux robots à chaîne ouverte et à chaîne fermée, et d'éviter toute ambiguïté quand il y a un plus grand nombre d'articulations, [Khalil 86] suggère quelques modifications à appliquer à la version initiale. Les quatre paramètres et les paramètres supplémentaires sont également conservés dans la nouvelle version, mais les indices de souscription d'axe sont modifiés. La procédure entière est décrite comme suit :

La structure est composée de n+1 corps connectés les uns aux autres. A partir de la base fixe, corps (0), jusqu'au terminal, corps (n), chaque articulation (i) relie le corps (i-1) au corps (i). Pour décrire un repère relié au corps (i) par rapport au repère relié au corps (i-1), il est nécessaire de définir les paramètres suivants :

Chaque corps (i) a un repère fixe R_i.

Chaque articulation (i) a un axe Z_i.

X_i est définie comme le perpendiculaire commun de l'axe Z_i et l'axe Z_{i+1}.

Tous les paramètres décrivant le passage d'un repère R_{i-1} à un autre repère R_i sont donnés dans la figure1.1.

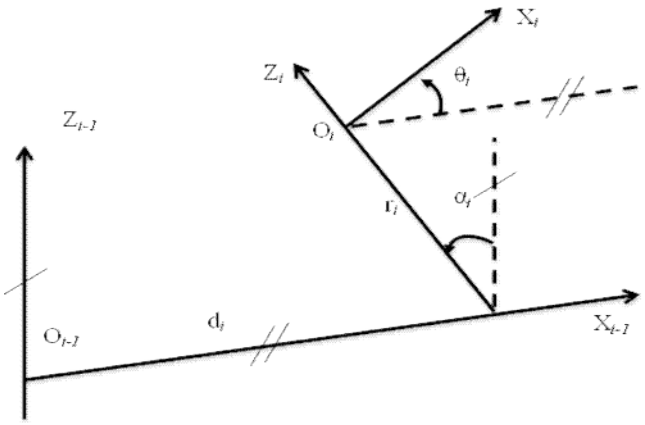


Figure1.1. paramètres géométrique

α_i: angle entre Z_{i-1} et Z_i autour de X_{i-1}.

d_i: distance entre Z_i et Z_{i-1} le long de X_{i-1}.

θ_i: angle entre X_{i-1} et X_i autour de Z_i.

r_i: distance entre X_i et X_{i-1} le long de Z_i

Les paramètres supplémentaires sont q_i et σ_i. Ils sont utilisés pour indiquer si l'articulation est rotoïde ou prismatique selon l'équation suivante:

$$q_i = \theta_i(1-\sigma_i) + r_i\sigma_i \tag{1.1}$$

L'articulation est rotoïde quand $\sigma_i = 0$, et prismatique quand $\sigma_i = 1$.

La matrice de transformation qui décrit le repère R_i par rapport au repère R_{i-1} est donnée par :

$$\begin{aligned} {}^{i-1}T_i &= Rot(X,\alpha_i)Trans(X,d_i)Rot(Z,\theta_i)Trans(Z,r_i) \\ &= \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & d_i \\ \cos\alpha_i\sin\theta_i & \cos\alpha_i\cos\theta_i & -\sin\alpha_i & -r_i\sin\alpha_i \\ \sin\alpha_i\sin\theta_i & \sin\alpha_i\cos\theta_i & \cos\alpha_i & r_i\cos\alpha_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{1.2}$$

Alors, la matrice de transformation peut être décrite comme une combinaison de l'orientation du repère R_i par rapport au repère R_{i-1}, définie par la matrice de rotation ${}^{i-1}A_i$,

et de la position de l'origine O_i définie par le vecteur ${}^{i-1}P_i$ où la matrice de rotation est donnée par:

$$ {}^{i-1}A_i = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{1.3} $$

Aussi, le vecteur de position est donné par :

$$ {}^{i-1}P_i = \begin{bmatrix} P_x \\ P_y \\ P_z \end{bmatrix} \tag{1.4} $$

Donc, la matrice de transformation peut être écrite sous sa forme compressée comme:

$$ {}^{i-1}T_i = \begin{bmatrix} & {}^{i-1}A_i & & {}^{i-1}P_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.5} $$

1.3. Modèle géométrique direct

Le modèle géométrique direct est la représentation de la situation de l'organe terminal, par rapport au repère de base, au moyen des coordonnées articulaires [Khalil 91, Khalil 07]. Cette représentation peut être décrite par la matrice de transformation qui donne la position et l'orientation de l'organe terminal en termes des coordonnées articulaires. En générale, le modèle géométrique direct est représenté par la relation suivante:

$$ {}^{0}T_{\mathrm{n}} = {}^{0}T_{1}(\mathrm{q_1})\,{}^{1}T_{2}(\mathrm{q_2}) \ldots {}^{\mathrm{n-1}}T_{\mathrm{n}}(\mathrm{q}_n) \tag{1.6} $$

Sous forme compacte, le modèle géométrique direct est défini par:

$$ \mathrm{X} = f(\mathrm{q}) \tag{1.7} $$

Où, q décrit les coordonnées articulaires données par:

$$ \mathrm{q} = \begin{bmatrix} q_1 & q_2 & q_3 & \ldots & q_n \end{bmatrix}^T \tag{1.8} $$

Et, X représente les coordonnées opérationnelles de l'organe terminal, désignées par:

$$ \mathrm{X} = \begin{bmatrix} P_x & P_y & P_z & s_x & s_y & s_z & n_x & n_y & n_z & a_x & a_y & a_z \end{bmatrix}^T \tag{1.9} $$

1.4. Méthodes de la description de l'orientation

Le déplacement du robot d'une configuration à une autre est généralement décrit par le changement de position et d'orientation du mécanisme par rapport à un repère de référence. Contrairement à la position où un vecteur à trois composantes est utilisé pour décrire la translation selon les axes du repère de référence, différentes représentations sont utilisées pour l'orientation. Les plus connus représentations sont les matrices de rotation, les angles d'Euler, les angles de Roulis-Tangage-Lacet et la représentation des quaternions [Kenneth 08].

1.4.1. Les matrices de rotation

Dans cette représentation, l'orientation du repère de coordonnées *i* par rapport au repère de coordonnées j est représentée par une matrice de rotation ${}^{j}R_{i}$ à neuf composantes. Les neuf composantes sont les résultats de l'orientation des trois axes x_i, y_i, z_i du repère de coordonnées i par rapport aux trois axes x_j, y_j, z_j du repère de coordonnées *j*. Cette relation est déterminé par:

$$ {}^{j}R_{i} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \tag{1.10} $$

Comme il y a trois axes, trois rotations élémentaires peuvent être distinguées. La rotation θ du repère *i* autour de l'axe x_j, la rotation θ du repère *i* autour de l'axe y_j et la rotation θ du repère *i* autour de l'axe z_j.

La première est représentée par $R_x(\theta)$ selon l'équation:

$$ R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} \tag{1.11} $$

La deuxième, autour de l'axe y_j, est représentée par $R_y(\theta)$ selon l'équation:

$$ R_y(\theta) = \begin{bmatrix} c\cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{1.12} $$

La troisième, $R_z(\theta)$ est la représentation de la rotation autour de z_j, et elle est défini par:

$$R_y(\theta) = \begin{bmatrix} c\cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{1.13}$$

1.4.2. Les angles d'Euler

Dans la représentation des angles d'Euler, le but est de minimiser le nombre de paramètres utilisés pour décrire une rotation. Par conséquent, trois angles α, β, γ sont utilisés à la place des neuf paramètres utilisés dans la représentation des matrices de rotation. Chacun d'eux signifie une rotation autour de l'un des axes du repère mobile. Dans les angles d'Euler Z-Y-X, le premier paramètre α correspond à l'angle de rotation autour d'un axe Z du repère mobile. Le second paramètre β est équivalent à une rotation autour de l'axe Y du repère résultant d'une rotation autour de l'axe Z. Les rotations α autour de l'axe Z, et β autour du nouvel axe Y déplacent l'axe X deux fois. L'angle de rotation autour de l'axe X résultant fournit le troisième paramètre γ.

La transformation entre la représentation des angles d'Euler et la représentation des matrices de rotation peut simplement être donnée par les équations suivantes:

$$\beta = \text{Atan2}\left(-r_{31}, \sqrt{r_{11}^2 + r_{21}^2}\right) \tag{1.14}$$

$$\alpha = \text{Atan}\left(\frac{r_{21}}{\cos\beta}, \frac{r_{11}}{\cos\beta}\right) \tag{1.15}$$

$$\gamma = \text{Atan}\left(\frac{r_{32}}{\cos\beta}, \frac{r_{33}}{\cos\beta}\right) \tag{1.16}$$

Cette transformation montre que la représentation des angles d'Euler souffre du problème de singularités. Ce problème se produit lorsque β correspond à un angle perpendiculaire.

1.4.3. Les angles de Roulis-Tangage-Lacet

Puisque l'orientation définie par trois rotations autour des trois axes d'un repère mobile peut être obtenue par trois rotations dans l'ordre inverse autour des trois axes d'un repère fixe, une représentation alternative qui utilise trois angles peut être employée pour représenter l'orientation d'un repère *i* par rapport à un autre repère *j*. Au lieu de faire référence à un repère mobile, une rotation autour d'un repère fixe est employée. Par

conséquent, la rotation X-Y-Z est généralement définie par trois angles ψ, θ, φ. De cette manière, l'orientation du repère i par rapport au repère fixe j est représentée par trois angles. Celle-ci est équivalente à une rotation ψ autour de l'axe fixe x_j, une rotation θ autour de l'axe fixe y_j et une rotation φ autour de l'axe fixe z_j. En effet, puisque cette représentation est semblable à celle de l'angle d'Euler, elle souffre aussi du problème de singularités**.**

1.4.4. Axe-angle

Dans la représentation axe-angle, au lieu d'utiliser trois paramètres pour définir l'orientation du repère i par rapport au repère j, quatre paramètres sont utilisés. Elles sont l'angle de rotation θ et le vecteur u, avec trois composantes (u_x, u_y, u_z), à travers lesquelles le repère i est tourné. Cette représentation est dite redondante du fait que soit la rotation par un angle négatif autour d'un vecteur négatif ou la rotation par un angle positif autour d'un vecteur positif oriente le repère i par rapport au repère j de la même manière. Généralement, la représentation axe-angle est donnée par θu ou (θu_x, θu_y, θu_z).

1.4.5. Représentation par quaternions

Comme leur nom l'indique, les quaternions utilisent quatre composantes pour représenter l'orientation. Ces composantes sont les quatre scalaires Q_0, Q_1, Q_2, Q_3 qui sont définis en terme de trois opérateurs i, j, k. De cette façon, un quaternion est donné par:

$$Q = Q_0 + Q_1 i + Q_2 j + Q_3 k \tag{1.17}$$

Contrairement aux autres représentations, la représentation par les quaternions ne souffre pas du problème de singularités. Donc, les quaternions sont devenus un outil très utile pour la représentation de l'orientation.

1.5. Modèle géométrique inverse

Contrairement au modèle géométrique direct, qui décrit les coordonnées opérationnelles de l'organe terminal du robot en fonctions des coordonnées articulaires, le modèle géométrique inverse implique la recherche des coordonnées articulaires nécessaires pour une situation désirable de l'organe terminal [Khalil 91**,** Khalil 07]. Cela consiste à trouver la solution au problème défini par l'équation suivante:

$$^{f}T_{E}^{\,d} = Z^{0}T_{n}(q)E \tag{1.18}$$

Où:

Z est la matrice de transformation qui définit le repère de la base en fonction du repère de référence.
E est la matrice de transformation représentant le repère de l'outil en termes du repère de l'organe terminal.
${}^{0}T_{n}$ est la matrice de transformation qui regroupe la position et l'orientation du repère de l'organe terminal en fonction du repère de la base.

En regroupant les variables inconnues d'un côté et les variables connues de l'autre côté, le problème géométrique inverse défini par l'équation (1.18) sera écrit comme suit:

$$ {}^{0}T_{n}(q) = Z^{-1} {}^{f}T_{E}{}^{d}E^{-1} \tag{1.19} $$

On considère que la matrice de transformation d'une situation désirable de l'organe terminal du robot est donnée par:

$$ U_0 = Z^{-1} {}^{f}T_{E}{}^{d}E^{-1} = \begin{bmatrix} s_x & n_y & a_z & p_x \\ s_x & n_y & a_z & p_y \\ s_x & n_y & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{1.20} $$

Par conséquent, en remplaçant les équations (1.6) et (1.19) dans l'équation (1.20), le problème est le suivant:

$$ U_0 = {}^{0}T_{1}(q_1) {}^{1}T_{2}(q_2) \ldots {}^{n-1}T_{n}(q_n) \tag{1.21} $$

Pour résoudre ce système d'équation, Paul [Paul 81] a proposé une méthode basée sur la pré-multiplication de l'équation (1.21) par la matrice ${}^{i}T_{i-1}$ pour différente valeur de *i* allant de 1 jusqu'à *n*-1. Cela permet d'isoler et identifier les coordonnées articulaires les uns après les autres. L'équation qui en résulte avec un robot à six degrés de liberté est la suivante:

$$ \begin{aligned} U_0 &= {}^{0}T_{1} {}^{1}T_{2} {}^{2}T_{3} {}^{3}T_{4} {}^{4}T_{5} {}^{5}T_{6} \\ {}^{1}T_{0}U_0 &= {}^{1}T_{2} {}^{2}T_{3} {}^{3}T_{4} {}^{4}T_{5} {}^{5}T_{6} \\ {}^{2}T_{1}U_0 &= {}^{2}T_{3} {}^{3}T_{4} {}^{4}T_{5} {}^{5}T_{6} \\ {}^{3}T_{2}U_0 &= {}^{3}T_{4} {}^{4}T_{5} {}^{5}T_{6} \\ {}^{4}T_{3}U_0 &= {}^{4}T_{5} {}^{5}T_{6} \\ {}^{5}T_{4}U_0 &= {}^{5}T_{6} \end{aligned} \tag{1.22} $$

1.6. Modèle cinématique direct

Une méthode simple pour obtenir les vitesses de l'organe terminal du robot est la dérivation de l'équation du modèle géométrique directe (1.7) par rapport au temps. Cela donne l'équation suivante [Khalil 99]

$$\frac{dX}{dt} = \dot{X} = J(q)\dot{q} \tag{1.23}$$

Comme il est clair, cette équation, nommée modèle cinématique direct, relie les vitesses opérationnelles de l'organe terminal du robot aux vitesses des coordonnées articulaires au moyen de la matrice jacobéenne.

1.6.1. Calcul de la matrice jacobéenne

Comme la vitesse de l'organe terminal du robot est composée à la fois du vecteur de vitesse linéaire V_n et la vitesse angulaire ω_n du repère dans laquelle ils sont exprimés, l'équation (1.23) se réécrit comme suit:

$$\begin{bmatrix} V_n \\ \omega_n \end{bmatrix} = J_n \dot{q} \tag{1.24}$$

La vitesse linéaire et angulaire de l'organe terminal résulte de la contribution de chaque $k^{ième}$ articulation par une vitesse linéaire $V_{k,n}$ et une vitesse angulaire $\omega_{k,n}$. Par conséquent, la description de la vitesse linéaire et angulaire est présentée par l'équation suivante:

$$\begin{cases} V_n = \sum_{k=1}^{n} V_{k,n} = \sum_{k=1}^{n} \sigma_k \mathrm{a}_k + \bar{\sigma}_k (\mathrm{a}_k \times L_{k,n}) \dot{q}_k \\ \omega_n = \sum_{k=1}^{n} \omega_{k,n} = \sum_{k=1}^{n} \bar{\sigma}_k \mathrm{a}_k \dot{q}_k \end{cases} \tag{1.25}$$

Où

a_k est le vecteur unitaire de l'articulation k selon l'axe z_k, et $L_{k,n}$ est le vecteur d'origine O_k et d'extrémité O_n.

La matrice jacobéenne est alors écrite comme:

$$J_n = \begin{bmatrix} \sigma_1 \mathrm{a}_1 + \bar{\sigma}_1 (\mathrm{a}_1 \times L_{1,n}) & \dots & \sigma_n \mathrm{a}_n + \bar{\sigma}_n (\mathrm{a}_n \times L_{n,n}) \\ \bar{\sigma}_1 \mathrm{a}_1 & \dots & \bar{\sigma}_n \mathrm{a}_n \end{bmatrix} \tag{1.26}$$

A partir de l'équation précédente, la $k^{ième}$ colonne de la matrice jacobéenne du repère n dans le repère i est donnée par:

$$ {}^{i}J_{n;k} = \begin{bmatrix} \sigma_k \, {}^{i}a_k + \bar{\sigma}_k ({}^{i}a_k \times {}^{i}L_{k,n}) \\ \bar{\sigma}_k \, {}^{i}a_k \end{bmatrix} = \begin{bmatrix} \sigma_k \, {}^{i}a_k + \bar{\sigma}_k ({}^{i}A_k \, {}^{k}\hat{a}_k \, {}^{k}L_{k,n}) \\ \bar{\sigma}_k \, {}^{i}a_k \end{bmatrix} \quad (1.27) $$

Où

$${}^{k}\hat{a}_k = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T , \; {}^{k}L_{k,n} = {}^{k}P_n$$

En remplaçant dans l'équation (1.26), le résultat sera comme suit :

$$ {}^{i}J_{n;k} = \begin{bmatrix} \sigma_k \, {}^{i}a_k + \bar{\sigma}_k \, {}^{i}\hat{a}_k ({}^{i}P_n - {}^{i}P_k) \\ \bar{\sigma}_k \, {}^{i}a_k \end{bmatrix} \quad (1.28) $$

Où

$\hat{a} = \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix}$ est la matrice symétrique.

1.7. Modèle cinématique inverse

Deux approches peuvent être utilisées pour définir le modèle cinématique inverse. La première approche fait référence au modèle différentiel. Elle permet la détermination de la variation articulaire nécessaire qui conduit à la variation opérationnelle souhaitée. La deuxième approche décrit le modèle cinématique inverse comme une relation entre la vitesse de l'organe terminal et la vitesse des coordonnées articulaire. Elle offre la possibilité de calculer la vitesse articulaire requise pour une vitesse opérationnelle désirée [Nakamura 86**,** Khalil 07]. Cette approche est décrite par l'équation suivante:

$$ \dot{q} = J^{-1} \dot{X} \quad (1.28) $$

Aussi, le modèle différentiel est décrit par l'équation:

$$ \frac{dq}{dt} = J^{-1} \frac{dX}{dt} \quad (1.29) $$

Dans les deux modèles, la détermination de la matrice jacobéenne inverse est cruciale. Cela peut être fait en fonction du cas rencontré. D'une part, lorsque la matrice

jacobéenne est une matrice carrée de rang plein, ce qui est le cas de la plupart des robots à six degrés de liberté, sa forme générale est donnée par [GOR 84]:

$$J = \begin{bmatrix} A & 0 \\ B & C \end{bmatrix} \tag{1.30}$$

Donc son inverse est donné par :

$$J^{-1} = \begin{bmatrix} A^{-1} & 0 \\ -C^{-1}BA^{-1} & C^{-1} \end{bmatrix} \tag{1.31}$$

D'un autre côté, lorsque le déterminant de la matrice jacobéenne est nul, le pseudo-inverse la matrice jacobéenne noté J^{+} est couramment utilisé [Whitney 69]. Donc, l'équation (1.28) est écrite comme:

$$\dot{q} = J^{+}\dot{X} \tag{1.32}$$

1.8. Modèle dynamique

Il y a deux cas importants dans lesquels le modèle dynamique est requis. Le premier est considéré lorsqu'on veut tester les stratégies de planification de mouvement. Dans ce cas, les informations fournies, obtenues à partir de la trajectoire générée, sont les positions, les vitesses et les accélérations articulaires, et l'information requise est le vecteur des couples. L'objectif est donc de trouver les couples nécessaires pour fournir les variables articulaires données. Le deuxième cas dans lequel le modèle dynamique est appelé apparaît lors de la simulation du mouvement du robot. En simulation, il est utile d'étudier le comportement du système en appliquant un vecteur des couples. Le but est d'obtenir les variables articulaires en sortie, à partir des couples appliqués à l'entrée [Siciliano 09, Khali 99].

Pour tester les stratégies de planification de mouvement, le modèle communément appelé, le modèle dynamique inverse, est la relation qui permet de calculer le couple nécessaire en termes des variables articulaires et des forces externes appliquées au robot. Cette relation est défini par:

$$\Gamma = f(q, \dot{q}, \ddot{q}, \mathrm{f_e}) \tag{1.33}$$

La version utilisée en simulation est appelé le modèle dynamique direct. Ce modèle est représenté par la relation qui permet de déterminer les accélérations articulaires au

moyen du reste des variables du système, y compris les positions, les vitesses, les couples et les forces extérieures. Cette relation est généralement exprimée par:

$$\ddot{q} = f(q, \dot{q}, \Gamma, \mathrm{f}_e) \tag{1.34}$$

1.9. Calcul du modèle dynamique

Les formulations les plus utilisées pour déterminer le modèle dynamique des robots sont la formulation lagrangienne et la formulation de Newton-Euler.

1.9.1. Formulation lagrangienne

La formulation lagrangienne est basée sur l'accumulation de l'énergie totale des corps du robot. Cette énergie est généralement calculée en utilisant les énergies cinétiques et potentielles élémentaires des corps qui composent l'ensemble du système.

Supposant que les forces extérieures appliquées sur l'organe terminal sont nulles, la description des équations du mouvement utilisant la formulation lagrangienne est donnée par [Siciliano 09, Khali 99]:

$$\Gamma_i = \frac{d}{dt}\frac{\partial L}{d\dot{q}_i} - \frac{\partial L}{dq_i} \tag{1.35}$$

Où:

L: Lagrangien du système, donné par E-U

E: l'énergie cinétique totale du système.

U: l'énergie potentielle totale du système.

Comme l'énergie cinétique est une fonction quadratique des vitesses articulaires et que l'énergie potentielle est aussi fonction des variables articulaires, la forme générale de l'équation du mouvement sans prise en compte des forces de frottement est défini par:

$$\Gamma = M(q)\ddot{q} + C(q, \dot{q})\dot{q} + G(q) \tag{1.36}$$

Où

$M(q)$ est la matrice d'inertie;

$C(q, \dot{q})$ est le vecteur des forces de Coriolis et des forces centrifuges;

$G(q)$ est la matrice de gravité;

Γ est le vecteur des couples.

1.9.2. Formulation de Newton-Euler

La formulation de Newton-Euler est établie sur une récurrence vers l'avant suivie d'une récurrence vers l'arrière. En partant de la base jusqu'au terminal, le but de la récurrence vers l'avant est de déterminer les forces et les moments totaux des articulations en se basant sur les vitesses et les accélérations des articulations initialement calculées. En retournant de l'organe terminal à la base, la récurrence vers l'arrière permet de calculer le couple requis pour chaque articulation.

Un moyen utile et pratique pour trouver l'équation du mouvement en utilisant la formulation de Newton est donné par Khalil [Khalil 99] sur la base du travail de Luh et al. [Luh 80]. Selon cette formulation, les équations de la récurrence vers l'avant sont données par:

$$^{j}\omega_{j-1} = {}^{j}R_{j-1}\,{}^{j-1}\omega_{j-1} \tag{1.37}$$

$$^{j}\omega_{j} = {}^{j}\omega_{j-1} + \bar{\sigma}_j \dot{q}_j\,{}^{j}\mathrm{a}_j \tag{1.38}$$

$$^{j}\dot{\omega}_{j} = {}^{j}R_{j-1}\,{}^{j}\dot{\omega}_{j-1} + \bar{\sigma}_j(\ddot{q}_j\,{}^{j}\mathrm{a}_j + {}^{j}\omega_{j-1} \times \dot{q}_j\,{}^{j}\mathrm{a}_j) \tag{1.39}$$

$$^{j}U_{j} = {}^{j}\hat{\dot{\omega}}_{j} + {}^{j}\hat{\omega}_{j}\,{}^{j}\hat{\omega}_{j} \tag{1.40}$$

$$^{j}\dot{V}_{j} = {}^{j}R_{j-1}({}^{j-1}\dot{V}_{j-1} + {}^{j-1}U_{j-1}\,{}^{j-1}P_j) + \sigma_j(\ddot{q}_j\,{}^{j}\mathrm{a}_j + 2\,{}^{j}\omega_{j-1} \times \dot{q}_j\,{}^{j}\mathrm{a}_j) \tag{1.41}$$

$$^{j}F_{j} = M_j\,{}^{j}\dot{V}_{j} + {}^{j}U_j\,{}^{j}MS_j \times {}^{j}\dot{V}_j \tag{1.42}$$

$$^{j}F_{j} = M_j\,{}^{j}\dot{V}_{j} + {}^{j}U_j\,{}^{j}MS_j \times {}^{j}\dot{V}_j \tag{1.43}$$

$$^{j}\mathrm{M}_{j} = {}^{j}J_j\,{}^{j}\dot{\omega}_j + {}^{j}\omega_j \times ({}^{j}J_j\,{}^{j}\omega_j) + {}^{j}MS_j \times {}^{j}\dot{V}_j \tag{1.44}$$

$$\text{Avec } \omega_0 = 0, \quad \dot{\omega}_0 = 0, \quad \dot{V}_0 = -g \tag{1.45}$$

Les équations de la récurrence vers l'arrière sont données par:

$$^{j}\mathrm{f}_{j} = {}^{j}F_j + {}^{j}\mathrm{f}_{j+1} + {}^{j}\mathrm{f}_{ej} \tag{1.46}$$

$$^{j-1}\mathrm{f}_{j} = {}^{j-1}R_j + {}^{j}\mathrm{f}_j \tag{1.47}$$

$$^{j}\mathrm{m}_{j} = {}^{j}M_j + {}^{j}R_{j+1}\,{}^{j+1}\mathrm{m}_{j+1} + {}^{j}P_{j+1} \times {}^{j}\mathrm{f}_{j+1} + {}^{j}\mathrm{m}_{ej} \tag{1.48}$$

$$\Gamma_j = (\sigma_j {}^j\mathrm{f}_j + \bar{\sigma}_j {}^j\mathrm{m}_j)^{T\,j}\mathrm{a}_j + F_{sj}sign(\dot{q}_j) + F_{vj}\dot{q}_j + \mathrm{Ia}_j\ddot{q}_j \quad (1.49)$$

1.10. Planification des chemins et planification des trajectoires

En robotique, le terme planification des chemins fait référence à la courbe géométrique qui relie deux points différents. Cependant, la planification des trajectoires décrit cette courbe géométrique en termes de temps. Habituellement, la planification des chemins précède la planification des trajectoires dans le sens que la planification des chemins trouve une courbe réalisable qui relie la position initiale à la cible, tandis que la planification des trajectoires assigne le variable de temps à chaque point le long de cette courbe [Gasparetto 15].

1.10.1. Planification des chemins

La planification des chemins est la construction de la forme géométrique, que le robot peut suivre, sans aucune référence à l'échelle de temps. Généralement, la difficulté de ce problème dépend de la situation. Dans le cas où le chemin est destiné aux manipulateurs, la forme géométrique est déterminée par la tâche du robot. Cependant, pour les robots mobiles, le problème de planification des chemins est plus exigeant. Puisque ces types de robots ont besoin de plus d'autonomie, la caractéristique du planificateur de chemin est étendue pour inclure le comportement d'évitement d'obstacle. Néanmoins, le type de planification des chemins est fortement affecté par l'environnement dans lequel le robot navigue. Dans les environnements statiques, la planification de chemin est principalement effectuée hors ligne, mais dans un environnement dynamique inconnu, le chemin nécessite une planification en ligne.

1.10.2 Planification des trajectoires

Contrairement à la phase de planification des chemins où le chemin planifié est laissé sans association avec l'échelle de temps, la phase de planification des trajectoires considère la spécification du temps pour la forme géométrique. Cependant, ce n'est pas la seule tâche pour réaliser le mouvement souhaité. En plus de la spécification de temps, les contraintes imposées par l'environnement et celles imposées par la dynamique du robot doivent également être satisfaites. Cela signifie que, pour résoudre le problème de planification des trajectoires et trouver les valeurs d'entrée nécessaires pour le robot, il est important de prendre en compte tous ces aspects.

1.10.3. Planification des trajectoires dans l'espace articulaire

Dans ce type des trajectoires, le mouvement souhaité est obtenu en générant une trajectoire pour chaque articulation afin de déplacer l'organe terminal du robot d'une situation à une autre. Au début, les seules informations données sont la situation de départ et la situation d'arrivée, et peuvent être étendues pour inclure certaines situations intermédiaires que l'organe terminal doit traverser. Par conséquent, une sorte de transformation doit être introduite pour calculer l'image des cordonnées opérationnelles des situations de l'organe terminal dans les cordonnées articulaires. Cette transformation est généralement effectuée en employant le modèle cinématique inverse. L'étape suivante consiste à générer les trajectoires articulaires qui relient les images obtenues. A ce niveau, les trajectoires obtenues doivent être synchronisées pour traverser l'image de chaque point de passage en même temps. Ainsi, l'objectif principal de déplacer l'effecteur d'une situation initial à une situation finale en passant par les situations intermédiaires sera atteint. Cependant, comme la trajectoire est conçue dans l'espace articulaire, la trajectoire de l'organe terminal qui en résulte pourrait être indésirable [Siciliano 09].

De nombreux chercheurs abordent la résolution du problème de génération des trajectoires dans l'espace articulaire. Par exemple, le modèle «générateur de commandes» proposé par Vaccaro [Vaccaro 88], qui était une extension de celui suggéré par Wolovich [Wolovich 84], est basé sur l'introduction d'une fonction linéaire dans la boucle non linéaire pour obtenir la trajectoire nécessaire. L'entrée dans ce modèle est la trajectoire cartésienne requise définie seulement par la situation de l'organe terminal. Leurs sorties sont les positions, les vitesses et les accélérations articulaires. Le modèle est appliqué au bras CMU 'Direct Drive I', où la trajectoire requise est générée à l'aide d'un polynôme de cinquième ordre. Cependant, la conclusion est qu'un polynôme quantique est préférable en raison des grandes accélérations transitoires résultantes.

L'une des méthodes suggérée par Taylor [Taylor 79] pour résoudre le problème de suivi d'une ligne droite est basée sur la génération des trajectoires dans l'espace articulaires. Dans cette méthode, d'abord un ensemble des nœuds est défini, puis des points supplémentaires sont mis entre ces nœuds. Le nombre des points supplémentaires est lié à la déviation de l'organe terminal par rapport à la droite. Par conséquent, chaque fois qu'une erreur tolérée est violée, plus de points sont ajoutées, et leurs configurations équivalentes dans l'espace articulaires sont calculées et interpolées. Le critère d'arrêt de ce processus est

lié au déplacement et à l'écart de rotation qui doivent être minimisés. L'objectif principal de cette méthode est de réduire l'erreur entre la trajectoire désirée et la trajectoire produite.

1.10.4. Planification des trajectoires dans l'espace cartésien

Pendant la planification dans l'espace articulaire, le mouvement est garanti en déplaçant les articulations à travers un ensemble de configuration. Les configurations initiales sont généralement les résultats de l'application de la cinématique inverse aux points de trajectoire requis de l'organe terminal. La règle du planificateur est de générer les configurations intermédiaires qui lient les configurations initiales. C'est-à-dire que le planificateur doit générer les angles articulaires nécessaires qui passent par les configurations articulaires désirés. Bien que la trajectoire générée assure l'inclusion des configurations intermédiaires dans le mouvement, elle ne garantit pas le profil que l'organe terminal peut prendre. Par conséquent, la meilleure façon d'éviter les profils indésirables de l'effecteur est de générer la trajectoire dans l'espace cartésien. Pour atteindre ce but, il est nécessaire d'affecter les positions et l'orientation aux points de la trajectoire désirée. Ensuite, la règle du planificateur est de générer la trajectoire cartésienne qui inclut ces points. L'étape finale, qui est généralement réalisée pendant le fonctionnement, consiste à calculer les angles articulaires requis en introduisant le modèle cinématique inverse [Siciliano 09].

Pour générer un mouvement linéaire dans l'espace opérationnel, Taylor [Taylor 79] propose une méthode qui consiste à décomposer la trajectoire désirée en morceaux égaux. Les points intermédiaires reliant ces morceaux définissent les repères de l'organe terminal, et comprennent sa position et son orientation. L'orientation, définie par la matrice de rotation, est transformée en un vecteur en utilisant la représentation des quaternions. L'interpolation du vecteur résultant, qui comprend à la fois la position et l'orientation, permet une translation uniforme ainsi qu'une rotation uniforme. Le chemin généré en utilisant l'interpolation est ensuite converti à son équivalent dans l'espace articulaire au moyen d'une cinématique inverse.

1.11. Critères d'optimisation pour la planification des trajectoires

Différentes approches ont été proposées pour résoudre le problème de planification des trajectoires. Ces approches sont conçues pour optimiser les critères formulés. Généralement, les critères à optimiser sont basés sur le temps d'exécution, l'énergie, le jerk ou la combinaison entre eux [Gasparetto 15].

1.11.1. Durée d'exécution

Dans l'industrie, les performances des systèmes robotiques tels que la productivité et la vitesse sont généralement prioritaires. Par conséquent, un critère important à prendre en compte lors de la planification des trajectoires est le temps d'exécution. En minimisant le temps d'exécution, la vitesse et la productivité peuvent être améliorées.

Une option pour obtenir des trajectoires qui peuvent être exécutées en temps minimale consiste à approcher les chemins en utilisant des segments primitifs. De cette manière, le chemin entier est défini par les points intermédiaires reliant ces segments. Par conséquent, la façon la plus simple de trouver une trajectoire minimale est d'interpoler les points intermédiaires tout en satisfaisant certaines caractéristiques imposées comme la souplesse des entrées de commande du robot. Les approches les plus commun et largement utilisées pour cette effet sont basées sur l'optimisation des courbes splines où le temps d'exécution est la somme totale des intervalles de temps entre les points intermédiaires.

1.11.2. Minimisation de l'énergie

Bien que les trajectoires avec un temps d'exécution minimum, qui est un des facteurs principaux pour améliorer la productivité industrielle, prennent une grande importance, d'autres critères peuvent être considérés durant le traitement du problème de planification des trajectoires. La minimisation de l'énergie est parmi les critères remarquables qui peuvent améliorer plus que la souplesse de la trajectoire. En prenant l'énergie comme fonction objective, d'autres caractéristiques peuvent être améliorées. En effet, la minimisation de l'énergie améliore le rendement dynamique des actionneurs et diminue les contraintes mécaniques. En outre, l'optimisation de l'énergie peut être utilisée pour permettre aux robots plus d'autonomie. Tant que l'énergie est maintenue, les robots n'ont pas besoin de l'intervention de l'être humain, surtout quand ces robots naviguent dans des environnements inaccessibles.

1.11.3. Minimisation du jerk

Une autre approche pour résoudre le problème de planification des trajectoires est basée sur l'utilisation du jerk comme critère d'optimisation. Cette approche s'appuie sur le fait qu'il existe une relation entre les couples appliqués aux actionneurs et le jerk. Par conséquent, un moyen facile pour satisfaire les contraintes des actionneurs consiste à minimiser le jerk. La minimisation du jerk est très utile surtout lorsqu'il s'agit d'un

problème de suivi de trajectoire car elle permet d'éviter les vibrations de la structure mécanique. De plus, la planification de la trajectoire basée sur la minimisation du jerk produit des couples réguliers avec un ordre de grandeur accepté. Cela permet de limiter la charge appliquée aux actionneurs et à la structure mécanique du robot.

1.12. Détection de collision

Différents tests peuvent être envisagés pour la détection de collision qui sert à vérifier l'existence de l'intersection entre des objets en mouvement. La forme la plus simple pour la détection de collision ne regarde que la présence ou l'absence d'interférence des objets. Cependant, un test plus avancé peut prendre en compte le moment de la première interférence ou même la séquence temporelle de tous les points d'intersection. Selon Jiménez [Jiménez 98], toutes les formes des problèmes de détection de collision peuvent être résolues en utilisant quatre approches principales.

1.12.1. Détection d'interférences multiples

La détection d'interférences multiples est basée sur l'échantillonnage des trajectoires des objets en mouvement. Les échantillons résultants sont soumis à un test d'interférence statique pour détecter toute collision entre eux.

L'efficacité de l'approche de détection d'interférences multiples dépend fortement du pas d'échantillonnage. D'une part, des pas d'échantillonnage moyens peuvent être moins exigeants dans le calcul, mais il n'est pas possible de s'assurer que la trajectoire est sans collision. D'autre part, malgré les exigences de temps, les plus petits pas d'échantillonnage offrent une grande précision pour la détection de collision.

1.12.2. Interférence de volume balayé

Dans la méthode des interférences des volumes balayés, les objets mobiles sont décrits par l'espace qu'ils occupent pendant une période de temps. Typiquement, le volume de cet espace, appelé volume balayé, est déterminé par les points atteints par les objets pendant leur mouvement. Par conséquent, la détection de collision peut être formée comme un test de l'interférence des volumes. Bien que l'absence de toute interférence entre les volumes assure un mouvement sans collision, sa présence n'est pas suffisante pour la vérification de la collision.

1.12.3. Extrusion dans l'espace à quatre dimensions

Dans cette approche, au lieu d'utiliser trois paramètres, pour représenter le mouvement des objets, un paramètre supplémentaire, le temps, est ajouté. Cet arrangement spatial et temporel fournit des volumes dans l'espace appelés volumes extrudés. L'absence d'intersection entre ces volumes correspond à un mouvement sans collision. Contrairement à l'interférence des volumes balayés où l'interférence entre les volumes balayés est suffisante mais pas une condition nécessaire pour la collision, toute présence d'intersection des volumes extrudés confirme la collision des objets.

1.12.4. Paramétrage de la trajectoire

Dans l'approche de paramétrage de la trajectoire, le temps est utilisé comme variable unique pour décrire les paramètres géométriques des objets. Généralement, cette approche est basée sur la formulation du problème de détection de collision comme un problème analytique plutôt qu'un problème géométrique. Les chemins des objets en mouvement sont exprimés en fonction du temps. Ensuite, ces fonctions sont résolues pour déterminer les instances de temps où les collisions sont survenues.

1.13. Évitement de collision

Comme les robots partagent habituellement leur environnement avec d'autres objets et agents, différentes situations peuvent être envisagées. Dans certains cas, un robot peut coopérer avec d'autres robots dans le même environnement. Par conséquent, le contact entre eux est inévitable, car cette coopération joue un rôle important dans la réalisation de toute tâche. Cependant, dans d'autres situations, la meilleure façon d'éviter de blesser des agents comme les être humains ou d'endommager des objets comme le robot lui-même est de planifier l'évitement des collisions.

Pour traiter le problème d'évitement de collision, deux approches sont couramment employées [Petric 15].

1.13.1. Approche globale pour la planification des trajectoires

L'approche globale pour planifier un mouvement sans collision repose sur l'hypothèse que l'environnement est connu à l'avance. Lorsque le robot travaille dans un environnement où des informations complètes sur les structures géométriques des obstacles sont données, le problème d'évitement d'obstacle est réduit à la recherche d'une trajectoire réalisable qui relie la position initiale à la cible. Par conséquent, toute collision possible est

évitée en effectuant une planification hors ligne. L'inconvénient majeur de cette approche est associé au re-calcul de trajectoire qui est inévitable lorsque l'environnement change.

1.13.2. Approche locale pour la planification des trajectoires

Le défi principal de l'approche locale est de trouver un mouvement sans collision dans un environnement dynamique. Contrairement à l'approche globale, où l'objectif est d'éviter les obstacles statiques, l'approche locale vise à empêcher le robot de heurter des obstacles mobiles inattendus. Cela signifie qu'au lieu d'utiliser une technique de planification hors ligne pour trouver une trajectoire réalisable, une technique alternative est nécessaire pour éloigner le robot des obstacles dynamiques. Habituellement, cette technique consiste à équiper le robot avec des capteurs et à les exploiter pour détecter les obstacles. Enfin, une décision en ligne est effectuée pour assurer un mouvement sans collision.

Conclusion

Dans ce chapitre, nous avons présenté les modèles fondamentales pour décrire la géométrique, la cinématique ainsi que la dynamique des robots. Nous avons aussi montré une classification des trajectoires en fonction de l'espace utilisé pour leur planification. Cette classification concerne la planification des trajectoires dans l'espace articulaires ou l'espace cartesian. Ensuite, nous avons indiqué les métriques les plus communs dans le domaine d'optimisation des trajectoires. Nous avons décrit les outils les plus utilisés pour le traitement du problème d'évitement des collisions ainsi que les. Enfin, nous avons présenté les approches adoptées pour la planification des trajectoires dans des environnements statique ou dynamique.

Chapitre 2

Les Méthodes Conventionnelles pour la Planification des Trajectoires

2.1. Introduction

Dans ce chapitre nous présentons les méthodes conventionnelles les plus connues pour la planification des trajectoires. Ces méthodes incluent les algorithmes bug, le graphe de visibilité, les roadmaps, le champ potentiel et la décomposition cellulaire. Pour chacune de ces méthode nous donnons leurs principes et nous abordons leur sous classe si existe. Aussi nous montrons un état de l'art sur les travaux orientés vers la planification des trajectoires en utilisant ces méthodes.

2.2. Les algorithmes bug

Les algorithmes bug de planification de trajectoire considèrent une représentation simplifiée du robot et de son environnement. Ils supposent que le robot est une structure ponctuelle à base capteur. Ce point est capable de se déplacer de la source à la cible sans quitter l'environnement. Respectivement, l'environnement prend la forme d'un plan avec un nombre limité d'obstacles statiques. A leur tour, les obstacles sont considérés comme des polygones fermés et séparés. Cela signifie que la distance entre eux est suffisante pour que le point mobile puisse passer [Lumelsky 86].

2.2.1. Bug1

De la source au but, le robot suit une ligne droite. Lorsque le capteur signale un contact avec un obstacle, le point est classé comme le premier point de frappe. Ensuite, le robot contourne l'obstacle tout en stockant le point avec la moindre distance à la cible. En rencontrant ce point une autre fois, le robot quitte l'obstacle au premier point de départ. De même, pendant que le robot avance et rencontre un obstacle récent, un autre point de frappe est défini, le périmètre de l'obstacle est couvert et le nouveau point de départ est déterminé. Ce processus est répété autant de fois qu'il y a d'obstacles jusqu'à ce que le robot atteint le but (figure2.1). Cependant, si le robot retourne à l'un des obstacles laissés, aucun chemin n'est trouvé, et le but est dit inaccessible [Lumelsky 86, 87].

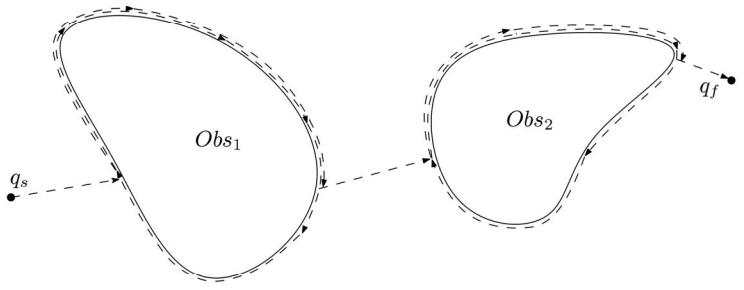


Figure 2.1. Planification des chemins par l'algorithme Bug1

La longueur du chemin planifié par l'algorithme bug1 est déterminée par la distance entre la source et la cible, et le nombre d'obstacles rencontrés. Avec un environnement plein d'obstacles, la première partie du chemin n'est plus une ligne droite, mais elle est composée d'un ensemble de segments reliant la source à la cible sans que ces segments franchissent les obstacles. Cela signifie que le chemin parcouru à travers ces segments n'atteint jamais la distance initiale de la ligne droite. La deuxième partie du chemin est décrit par le trajet parcouru autour des obstacles. A partir du point de frappe, chaque limite d'obstacle est couverte et, au plus, la moitié de cette distance est nécessaire pour atteindre le point de départ. En d'autres termes, le chemin formé autour de chaque obstacle ne dépasse jamais un et demi de son périmètre. En conséquence, toute la distance du chemin est combinée par les segments reliant les obstacles et ceux qui entourent ses périmètres. Cela peut être donné par:

$$D_t = D_l + 1,5\sum_i p_i \tag{2.1}$$

Où:

D_l est la distance linéaire de la source à la cible.

p_i est le périmètre du $i^{ème}$ obstacle.

2.2.2. Bug2

Avec l'algorithme Bug2, le robot utilise une manière différente pour trouver le chemin à la cible [Lumelsky 86, 87]. Il commence par définir une ligne droite qui relie la source à la cible. Comme il n'y a pas d'obstacle, le robot avance directement sur cette ligne. Cependant, lorsqu'un obstacle se bloque le chemin, le robot définit un point de frappe et diverge pour suivre les limites de l'obstacle. Lorsque le robot rencontre une autre fois la ligne droite, il définit le point de départ, d'où il retourne à la bonne route. Cette procédure est répétée jusqu'à ce que l'objectif soit atteint ou aucun chemin ne soit trouvé (figure2.2).

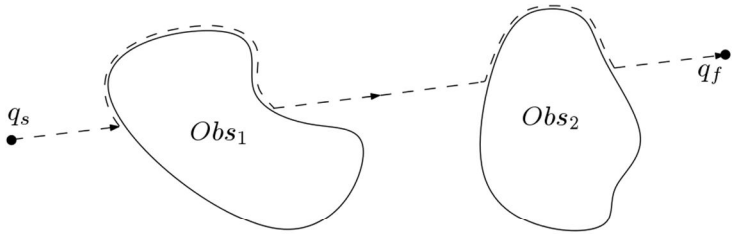


Figure 2.2. Planification des chemins par l'algorithme Bug2

La longueur du chemin de l'algorithme Bug2 est déterminée par de nombreuses conditions. D'une part, le nombre d'intersection n_i entre la ligne reliant la source à la cible et les obstacles dans l'environnement. Chaque obstacle peut être franchi au moins deux fois. En outre, le pire des cas conduit à couvrir tout le périmètre p_i de l'obstacle. D'autre part, la distance initiale D_l de la source à la cible n'atteint jamais la longueur d'origine de la ligne. Lorsque les obstacles bloquent son chemin, certaines de ses parties sont soustraites. Par conséquent, la longueur totale du chemin est donnée par l'équation suivante:

$$D_t = D_l + \sum_i \frac{n_i p_i}{2} \tag{2.2}$$

Où:

D_l est la distance linéaire de la source à la cible.

p_i est le périmètre du $i^{ème}$ obstacle.

n_i le nombre d'intersection avec le $i^{ème}$ obstacle.

L'algorithme Bug2 diffère de l'algorithme Bug1 en plusieurs points. D'une part, pour l'algorithme Bug1, le robot contourne la circonférence de tous les obstacles, dans sa direction vers la cible, au moins une fois. A chaque obstacle, le robot ne croise pas plus d'un point de frappe et d'un point de départ avant de trouver la cible. Cependant, la cible est inaccessible quand le robot rencontre le même obstacle deux fois. D'autre part, l'algorithme Bug2, le robot n'est pas obligé de contourner les obstacles pour atteindre la cible. De plus, à chaque obstacle rencontré, le robot peut définir plus d'un point de frappe et plus d'un point de départ. Cependant, il y'a une probabilité que le robot trouve la cible.

2.2.3. TangentBug

Contrairement aux algorithmes Bug1 et Bug2 qui sont basés sur les capteurs de contact, l'algorithme TangentBug est fondé sur l'utilisation de plusieurs capteurs de distance qui sont distribué uniformément sur le périmètre du robot. Cela signifie qu'au lieu d'utiliser le sens de toucher pour détecter la présence d'obstacles, le sens de vue est utilisée. Les capteurs de distance montés sur le robot ponctuel créent un champ centré sur sa position et s'annule à la fin de la limite des capteurs. Ce champ n'est plus qu'un cercle avec le robot ponctuel comme centre et un rayon égale à la portée des capteurs. Lorsque la distance entre le robot et un obstacle dans son environnant est supérieur à ce rayon,

l'obstacle est au-delà de l'horizon du robot. Cependant, lorsque cette distance est inférieure au rayon du cercle, tout obstacle est localisé [Kamon 96].

Dans une position donnée, les capteurs de distance construisent tous les chemins possibles. Initialement, les obstacles détectés sont déterminés. Ce sont les parties situées dans le champ de vison du robot, à partir des obstacles qui se chevauchent avec le cercle. Ensuite, les extrémités de ces parties sont connectées au point où le robot est positionné. Le graphe résultant crée tous les chemins possibles. Comme ses segments sont la tangente des points d'extrémité, ce graphe est appelé le graphe local tangentiel.

Sous l'algorithme TangentBug [Kamon 96], le mouvement du robot est caractérisé par deux comportements différents. Initialement, le robot se déplace vers la cible en suivant le chemin le plus court. Ce chemin est choisi parmi tous les chemins disponibles construits au moyen du graphe local tangentiel. Le chemin garantit que la distance est réduite tant que le robot se dirige vers la cible. Ensuite, lorsque le robot est piégé dans un emplacement donné, le mouvement vers la cible est terminé et un suivi des limites est initié. Ici, le robot peut entrer en mouvement illimité autour de l'obstacle. Par conséquent, la cible est inaccessible. Dans d'autres cas, la cible est accessible et un chemin court est disponible. Encore une fois, ce chemin est trouvé en employant le graphe local tangentiel. Enfin, le robot bascule entre les deux comportements jusqu'à ce que le robot atteint le but visé ou la cible est jugé inaccessible (figure2.3).

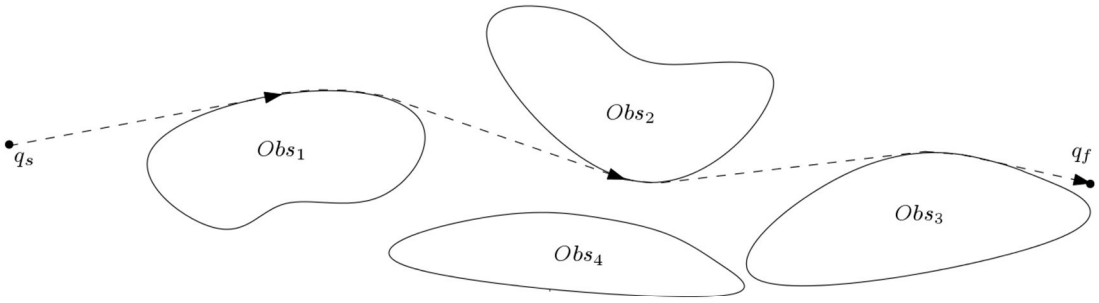


Figure 2.3. Planification des chemins par l'algorithme TangentBug

2.3. Les roadmaps

2.3.1. Silhouette

L'approche des roadmaps silhouette réduit le problème de planification de chemin à la recherche d'une courbe continue qui relie deux points donnés [Canny 88]. Elle commence par trouver les points extrêmes dans une surface donnée. Ensuite, Elle détermine s'il y a une déconnexion entre eux. D'une part, les points isolés, lorsqu'ils sont

trouvés, sont liés ensemble au moyen des segments supplémentaires. D'un autre part, les courbes ne sont faites que lorsque des points voisins sont connectés. Les segments et les courbes sont combinés pour former l'ensemble du squelette de la silhouette. Enfin, au lieu d'examiner la surface initiale entière, on cherche le chemin dans le squelette de la silhouette.

Le squelette de la silhouette est déterminé par une ligne droite appelée tranche. La tranche est déplacée dans tout le plan pour se croiser avec n'importe quel objet. Lorsque elle touche un objet, elle est appelée une tranche critique. De même, le point reliant la tranche critique à l'objet est appelé point critique. Cela signifie que toute nouvelle tranche critique est formée en faisant face à un autre point critique ou en le quittant. Cependant, au fur et à mesure que la tranche avance, les points extrêmes des objets sont repérés. Enfin, pour former l'ensemble du squelette de la silhouette, les points voisins sont reliés entre eux et reliés aux isolées, si elles existent, par les coupes critiques elles-mêmes (figure 2.4).

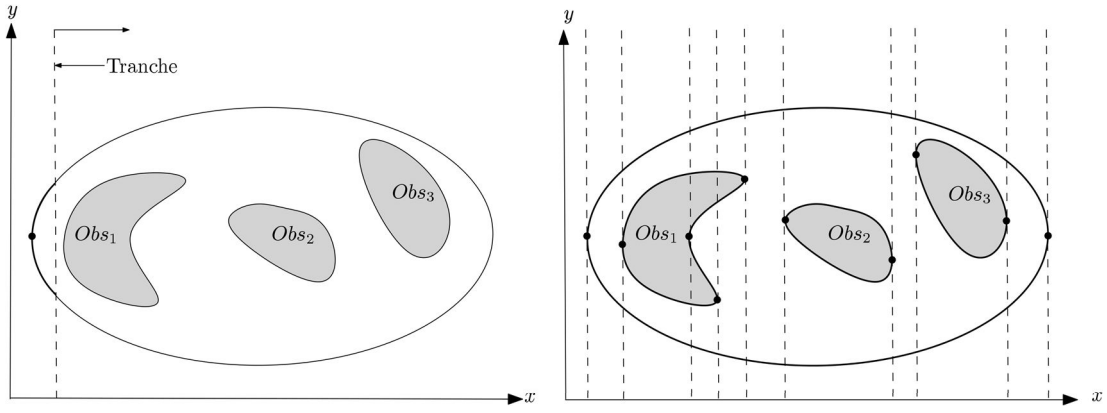


Figure 2.4. Construction d'une silhouette dans le plan

Au lieu d'utiliser une ligne droite, un plan est utilisé pour construire le squelette de la silhouette dans l'espace. Le processus commence en sélectionnant une direction spécifique. Ensuite, selon cette direction, le plan se déplace pour traverser n'importe quel objet sur son chemin. Les points extrêmes résultants de l'intersection entre le plan et les objets forment ce qu'on appelle la roadmap de base. Les points critiques, lorsqu'ils existent, sont connectés à cette roadmap de base. La combinaison des deux ensembles de points, constituée au moyen des plans critiques, forme la roadmap complète (figure 2.5).

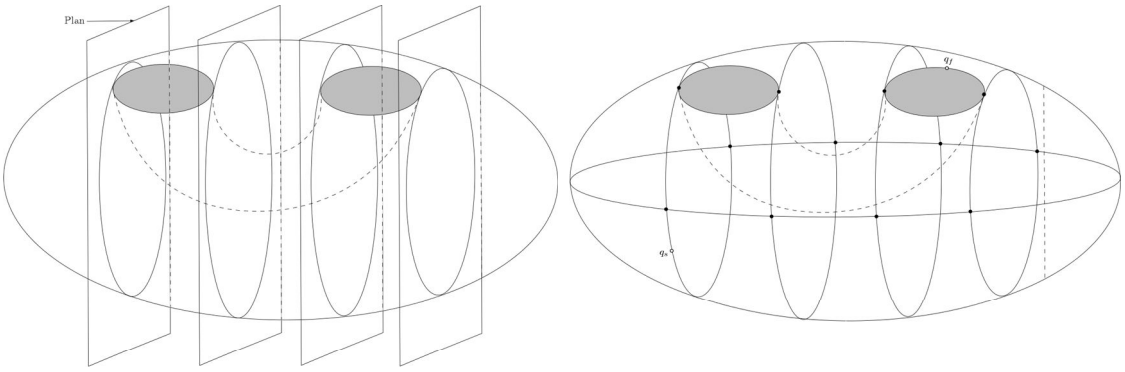


Figure 2.5. Construction d'une silhouette dans l'espace

2.3.2. Planificateur de chemin opportuniste

Pour éviter le contact du robot avec les obstacles voisins lors de déplacement vers la destination, [Canny 93] propose une autre alternative à l'approche de la silhouette. Celle-ci est connue sous le nom du planificateur de chemin opportuniste. Cette approche intègre à la fois la technique du champ potentiel et la technique de silhouette. La première assure que le robot est tenu à l'écart des obstacles voisins, tandis que la deuxième détermine les chemins locaux.

Le champ potentiel est utilisé pour s'assurer que le robot garde une certaine distance avec les obstacles. Il comprend uniquement des forces répulsives qui garantissent que le robot ne touche aucun des obstacles voisins. Les forces répulsives sont définies en fonction de la distance euclidienne. Lorsque cette distance est réduite, les forces répulsives augmentent et vice versa. Cela signifie qu'un potentiel uniforme est généré lorsqu'une distance maximale donnée est imposée.

Comme l'approche de la silhouette, une tranche balayée est utilisée pour déterminer le chemin. La tranche est déplacée dans une direction donnée. Tous les points extrêmes du champ potentiel qui se croisent avec cette tranche construisent le chemin sans collision. Cependant, en certains points appelés points critiques intéressants, le champ potentiel disparaît et le chemin devient divisé. Entre ces points, l'espace divisé par l'obstacle est appelé un canal.

Pour former l'ensemble du chemin de la roadmap, toutes les étapes de l'algorithme proposé sont suivies. Tout d'abord, le point de départ et la cible sont connecté au chemin. Cette connexion est constituée par les tranches traversant ces points. Deuxièmement, la discontinuité de la roadmap est vérifiée. Si aucune n'est trouvée, le processus est terminé, sinon les parties isolées sont connectées. Encore une fois, la connexion est construite, à

proximité des points critiques intéressants, à travers des segments appelés ponts. Finalement, la première étape est répétée jusqu'à que le point de départ et la cible sont connectées, ou aucun chemin n'est trouvé (Figure2.6).

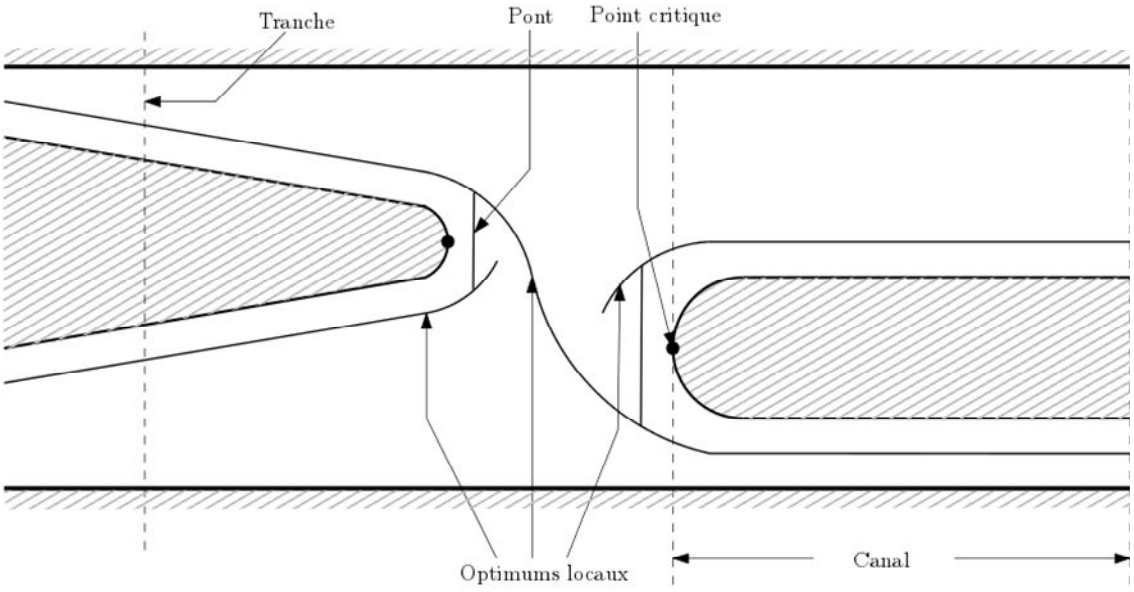


Figure 2.6. Planificateur de chemin opportuniste

2.3.3. Diagramme de Voronoi

2.3.3.1. Diagramme de Voronoi classique

La construction du diagramme de Voronoi pour un ensemble de points dans le plan consiste à diviser ce plan en différentes régions [Shamos 75]. Habituellement, les éléments de base dans cette division, les points, sont appelés sites. De même, les régions associées à ces sites sont appelées polygones de Voronoi. Ces polygones sont composés d'un groupe de points plus proches du site associé que d'autre site.

La construction du diagramme de Voronoi dépend de la manière dont le plan est divisé. Le moyen le plus simple de diviser le plan consiste à utiliser des segments pour connecter les sites les uns aux autres. Ensuite, des lignes perpendiculaires qui divisent chaque segment en deux parties équidistantes sont dessinées. Lorsque toutes ces lignes bissectrices sont dessinées, les segments sont supprimés, ce qui permet de former le squelette du diagramme de Voronoi. Ce n'est que lorsque les lignes bissectrices sont réunies, aux points où elles sont coupées, que le diagramme de Voronoi est construit (figure 2.7).

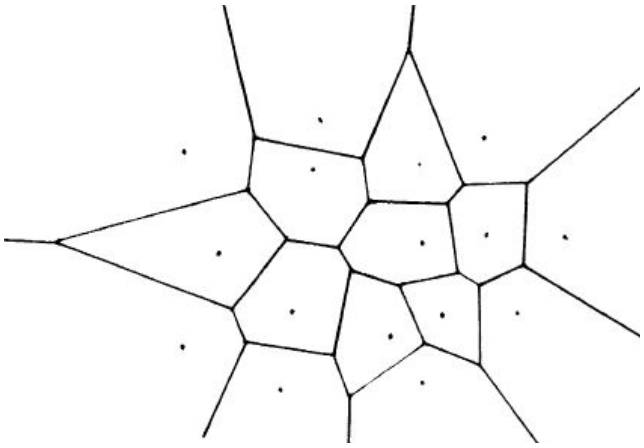


Figure 2.7. Diagramme de Voronoi classique

2.3.3.2. Diagramme de Voronoi généralisé

Le diagramme de Voronoi classique peut être étendu pour donner lieu à une forme plus générale appelée diagramme de Voronoi généralisé. Contrairement au diagramme de Voronoi classique, où les sites sont des points simples, le diagramme de Voronoi généralisé a des sites plus complexes tels que les polygones et les segments. Cela ne constitue pas un problème car la plupart des formes sont des structures à base de points. Par conséquent, la planification du diagramme de Voronoi généralisé est réduite à la construction de l'image des éléments de base de ses sites. Cela signifie que le diagramme de Voronoi généralisé des sites polygonaux consiste à planifier le squelette des sommets des points et des bords des segments. En conséquence, le squelette entier est produit par la combinaison de l'ensemble de points équidistant provenant de ces sites élémentaires (figure 2.8).

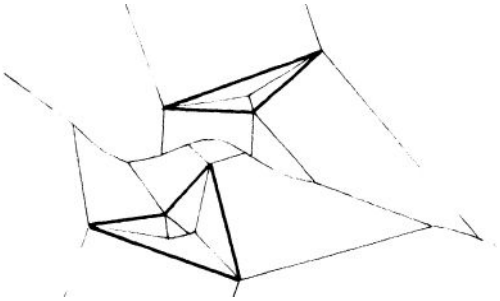


Figure 2.8. Diagramme de Voronoi généralisé

Dans le diagramme de Voronoi généralisé, le squelette entier est composé de trois parties élémentaires différentes [Lee 81, O'Duinlaing 87]. Habituellement, il y a la séquence équidistante de points à deux sommets. C'est une bissectrice qui divise le plan en

deux moitiés égales avec un sommet chacune. C'est la même que celle utilisée dans le diagramme de Voronoi classique. De même, la bissectrice de deux segments non parallèles étirés constitue un autre élément central. Le dernier constituant est la courbe qui sépare entre un sommet et un segment. C'est un arc de rayon égal à la moitié de la distance entre le sommet et le segment (figure 2.9).

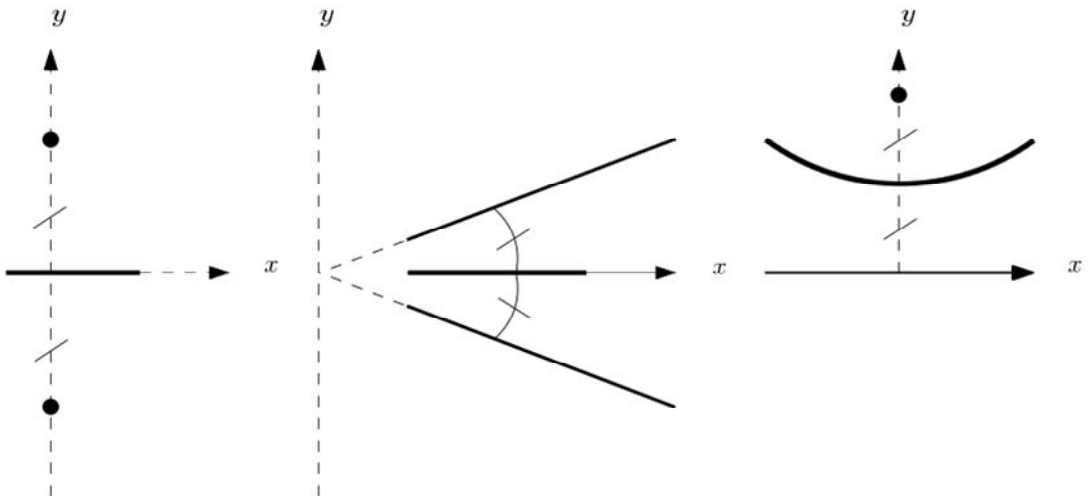


Figure 2.9. Composantes de base d'un diagramme de Voronoi généralisé

Pour le problème de la planification des trajectoires, il est clair que les sites représentent les obstacles et que le diagramme de Voronoi correspond à l'ensemble des chemins possibles. En conséquence, de nombreux travaux orientés sur les diagrammes de Voronoi sont proposés pour résoudre le problème de planification de trajectoire. O'Dunlaing [O'Dunlaing 85] suggère l'utilisation d'un diagramme de Voronoi généralisé pour obtenir un mouvement continue, sans collision, d'un disque. L'espace de travail est une chambre polygonale ouverte. Elle comprend des obstacles avec des limites spécifiques. Le squelette complet du diagramme de Voronoi généralisé est formé par un ensemble de points équidistants à partir de ces limites. La phase suivante corresponde à la recherche d'un chemin continu qui conduit le disque de la position initiale à la position finale. En fait, le chemin n'existe que lorsque le disque navigue sans toucher les limites des obstacles, et que son centre se déplace librement dans le squelette de Voronoi. Lorsque le chemin existe, la planification du mouvement est dite rétractée sur le diagramme de Voronoi.

2.3.4. Graphe de visibilité

Comme indiqué dans [Latombe 91], la méthode la plus ancienne pour la planification des trajectoires est le graphe de visibilité. Cette méthode est appliquée par

[Nilsson 69] pour la planification de trajectoire d'un robot mobile. D'une manière générale, cette méthode est appropriée surtout pour la configuration bidimensionnelle. Dans ce cas, pour planifier un chemin pour un robot, d'une position initiale à une position finale, dans un environnement plein d'obstacles, de nombreuses étapes doivent être suivies. Premièrement, les obstacles doivent être modélisés comme des polygones. Deuxièmement, l'espace libre, le reste de la soustraction de l'espace de configuration de tous les obstacles liés ensemble, doit être déterminé. Ensuite, toutes les lignes s'étirent dans l'espace libre, reliant les sommets des obstacles et la position initiale et finale, doivent être dessinés. Enfin, le chemin de la roadmap est produit à partir de la méthode de graphe de visibilité en connectant la position initial, passant par des sommets d'obstacles dans l'espace libre, et en terminant à la position finale. Ces étapes devraient être conçues comme le montre la figure 2.10.

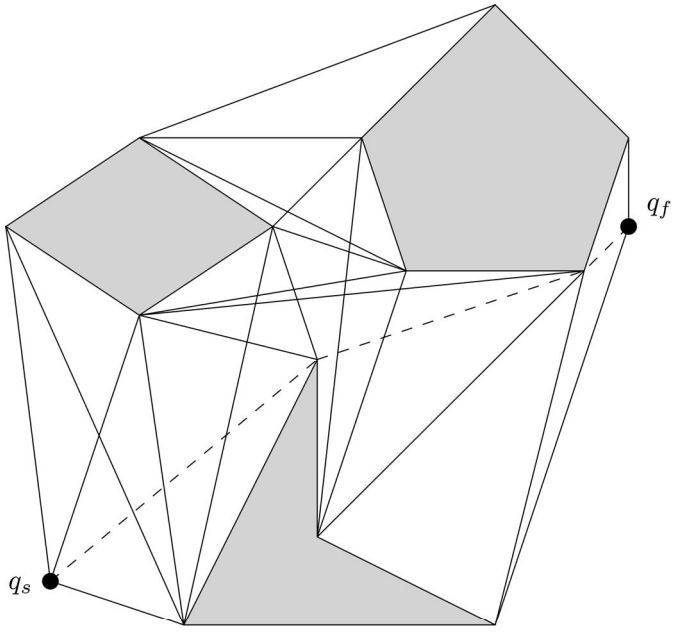


Figure 2.10. Construction du graphe de visibilité

La méthode de graphe de visibilité, que ce soit dans sa version originale ou modifiée, est utilisée par de nombreux chercheurs pour résoudre le problème de la planification des chemins. Jason [Jason 95] suggère l'utilisation de vecteurs traversable comme approche de graphe de visibilité pour la planification de mouvement globale. Ce concept est basé sur l'élimination de toutes les lignes, reliant l'une des extrémités du polygone d'obstacle à tout autre sommet visible des autres obstacles polygonaux, différent du sommet le plus éloigné qui est prétendu redondant. Après une étape de simulation,

l'algorithme est implémenté dans un robot mobile autonome pour les expériences de guidage.

Lozano-Pérez [Lozano-Pérez 79] propose un algorithme pour la planification de trajectoire dans l'environnement polyédrique inoccupé en utilisant la méthode de graphe de visibilité. Son algorithme est inspiré à partir de celui utilisé par Udupa [Udupa 77]. L'idée principale de l'algorithme est d'étendre toutes les obstacles dans l'environnement, en réduisant l'objet mobile à un point élémentaire, puis de mettre en œuvre la méthode de graphe de visibilité pour trouver le plus court chemin. Cette idée est clairement expliquée dans l'espace cartésien en étendant les obstacles polygonaux proportionnellement à un point de référence dans l'objet en mouvement. Les régions d'expansion résultantes des obstacles polygonaux sont appelées les régions interdites, et leurs sommets mutuellement vues sont reliés ensemble à l'aide de la méthode de graphe de visibilité. Ensuite, le chemin le plus court est recherché dans ce graphe. Après, la méthode est étendue à un espace de travail à trois dimensions et mise en œuvre pour planifier la trajectoire d'un manipulateur à sept degrés de liberté. Cependant, il est constaté que les obstacles dans l'environnement sont plus complexes et qu'il fallait beaucoup de temps pour construire le graphe de visibilité, mais le chemin qui en résulte n'est jamais le plus court.

Huang [Huang 04] introduit un algorithme de graphe de visibilité modifié, appelé visibilité graphique dynamique. L'idée est basée sur l'utilisation de ce qui est appelé une région active, définie comme le rectangle joignant les extrémités les plus éloignées des obstacles traversés par la ligne droite qui relie le point de départ au point final. Lorsque le chemin qui passe par l'extrémité externe la plus éloignée d'un obstacle est plus court que tout autre chemin à l'intérieur et l'obstacle est inclut dans la région active mais ne croise pas avec la ligne précitée, cette région est mise à jour pour l'inclure complètement. De plus, les chemins de la visibilité graphique sont construits et l'algorithme de Dijkstra est utilisé pour rechercher le chemin le plus court dans cette région. Pour la planification des trajectoires, l'algorithme de la visibilité graphique dynamique est comparé à l'algorithme de visibilité graphique standard, et la notion de cercle de vertex est utilisée pour résoudre ce problème avec des objectifs multiples. L'algorithme dynamique est prouvé très efficace dans la situation précédente.

Tran [Tran 13] propose une planification de trajectoire globale pour les robots autonomes en utilisant une méthode de graphe de visibilité modifiée, qu'il nomme la visibilité graphique parallèle orientée. L'idée est basée sur la subdivision de l'espace de

travail en plusieurs sous-espaces de travail, puis un graphe de visibilité synchrone est dessiné pour chaque sous-espace de travail. Le chemin total est l'accumulation des sous-chemins graphiques résultant de la roadmap. L'objectif principal est de minimiser la distance parcourue par le robot, depuis la position initiale jusqu'à la position finale.

2.4. Champs potentiels

L'une des méthodes les plus utilisées pour résoudre le problème de la planification des trajectoires en présence d'obstacles est la méthode de champ potentiel. Le principe de cette méthode est basé sur la diffusion des charges électriques sur le robot et son environnement. En général, le robot et les obstacles reçoivent des charges positives, tandis que la cible reçoit des charges négatives. Les charges positives entraînent des forces répulsives qui repoussent le robot des obstacles proches. Les charges négatives ont des forces d'attraction qui tirent le robot vers la destination. De cette manière, le chemin prévu est créé par l'accumulation des forces qui éloignent le robot des obstacles et de ceux qui l'attirent vers la destination (figure 2.11) [Choset 05].

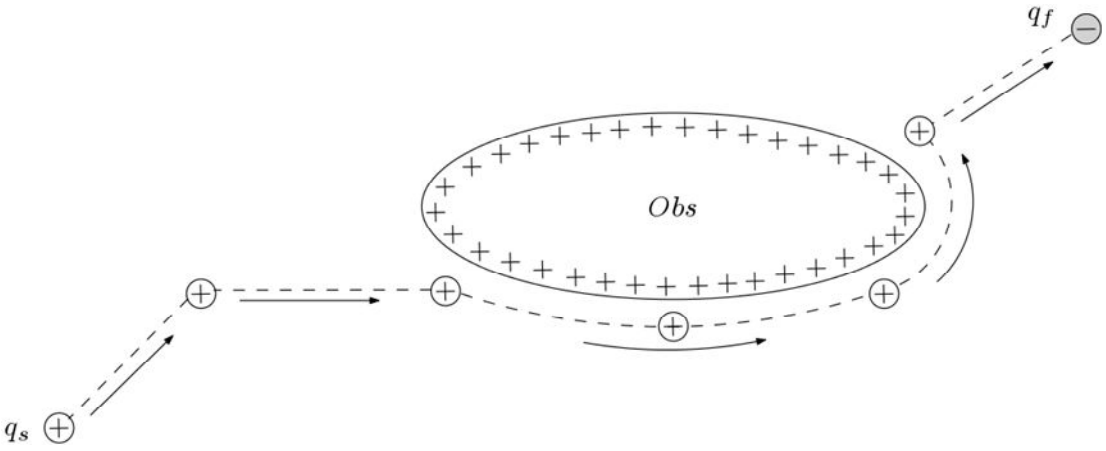


Figure 2.11. Planification des chemins par la méthode du champ potentiel

La construction des forces attractives et répulsives peut être faite par l'utilisation de l'équation suivante :

$$U(q) = U_{att}(q) + U_{rep}(q) \tag{2.3}$$

Les forces d'attraction dépendent fortement de la destination finale. Par conséquent, la formulation appropriée pour calculer les forces d'attraction relie l'amplitude des forces à la distance entre le robot et la cible. Plus la distance est grande, plus l'amplitude devrait

être grande. Le vecteur résultant peut être approché en utilisant une fonction quadratique pour assurer une distance positive. Ceci est donné par l'équation suivante:

$$U_{att}(q) = \frac{1}{2}\xi d^2(q, q_f) \tag{2.4}$$

Comme les forces d'attraction sont déterminées par la distance entre le robot et la cible, les forces de répulsion sont également influencées par la proximité du robot aux obstacles. Plus la distance est faible, plus la force doit être grande et vice versa. Chaque fois que cette distance dépasse une certaine valeur, l'obstacle doit être ignoré. La relation qui donne les forces de répulsion en termes de distance entre le robot et les obstacles voisins pourrait être définie comme suit:

$$U_{rep}(q) = \begin{cases} \frac{1}{2}\mu\left(\frac{1}{d(q)} - \frac{1}{\delta}\right)^2 & d(q) \le \delta \\ 0 & d(q) > \delta \end{cases} \tag{2.5}$$

Où

μ est le gain répulsif.

δ est la distance limite entre le robot et l'obstacle. Chaque fois que cette limite est dépassée, l'obstacle est ignoré.

Un travail influent dans l'utilisation du champ potentiel pour résoudre le problème de la planification des trajectoires est celui de [Khatib 86]. Son approche est une approche cartésienne en ligne. Cela signifie que le chemin est généré en temps réel dans l'espace cartésien. Au début, toutes les articulations du robot sont supposées être chargées positivement, tandis que la cible est estimée être chargée négativement. Les charges opposées produisent une force d'attraction qui conduit l'effecteur du robot vers la cible. Au cours de mouvement vers la cible, le mécanisme entier du robot est soumis à des forces répulsives. Ces forces sont le résultat des charges positives des obstacles voisins. Par conséquence, le champ potentiel total créé par des forces attractives et répulsives est ajouté à l'énergie cinétique de la structure pour fournir le formalisme lagrangien et donc l'équation du mouvement. L'approche implique également de résoudre le problème des contraintes articulaires en utilisant le champ potentiel. Comme les articulations sont chargées positivement, des forces répulsives sont créées chaque fois que les articulations violent les

limites. Ces forces répulsives agissent comme une barrière pour satisfaire les contraintes articulaires.

Warren [Warren 89] propose une autre stratégie basé sur l'utilisation du champ potentiel pour la planification globale de chemin. Dans son approche, l'espace de travail du robot est défini en tant qu'un espace de configuration dans lequel la position et l'orientation du mécanisme ont une représentation ponctuelle. Un chemin initial est généré entre les obstacles statiques. Tous les points du chemin généré sont soumis au champ potentiel créé par les forces répulsives des obstacles voisins. Ensuite, le chemin d'origine est optimisé pour obtenir la valeur minimale du champ potentiel, en considérant les points globaux formulant le chemin, et en tenant le but principal d'éviter le problème des minima locaux. Finalement, pour montrer l'efficacité de cette stratégie, un chemin sécurisé est généré pour un manipulateur à deux degrés de liberté.

2.5. Décomposition cellulaire

Dans la technique de planification des chemins par décomposition cellulaire, l'espace de configuration original est initialement divisé en deux: l'espace d'obstacles et l'espace de configuration libre. Encore, l'espace de configuration libre est décomposé en un ensemble de cellules adjacentes. Ces cellules sont représentées par des nœuds et connectées les unes aux autres au moyen de leurs bordures pour créer ce qui est appellé un graphe adjacent. Le chemin planifié est construit par le graphe final qui connecte tous les nœuds.

Puisque la technique de décomposition cellulaire est utile pour la couverture, le problème de la planification des chemins est défini pour trouver une façon de se déplacer à travers les cellules en passant par leurs nœuds. Pour résoudre ce problème, différentes représentations de la décomposition cellulaire sont utilisées. Ces représentations incluent la décomposition cellulaire trapézoïdales, la décomposition cellulaire boustrophédon et la décomposition cellulaire de Morse.

2.5.1. Décomposition cellulaire trapézoïdales

L'une des décompositions cellulaires bien connues est la méthode trapézoïdale. Cette méthode est basée sur la division de l'espace de configuration en cellules trapézoïdales. Le processus commence par la représentation des obstacles dans l'espace de configuration en utilisant des polygones. Les sommets de ces polygones sont étirés, lorsque cela est possible, verticalement de haut et bas. Les lignes étirées sont terminées lorsqu'elles

sont coupées avec un bord d'un autre obstacle ou un bord de l'espace de configuration. De cette manière, les formes résultantes sont soit des cellules trapézoïdales, soit des triangles. Dans le cas où les cellules ont une forme triangulaire, elles sont considérées comme des cellules trapézoïdales sans bord parallèle [Nora 99]. Le problème de planification de chemin est, par conséquent, réduit à trouver la route qui relie ces cellules adjacentes. Un moyen facile pour trouver cette route est de connecter le milieu des lignes reliant les cellules les unes aux autres. L'ensemble du processus peut être résumé dans la figure 2.12.

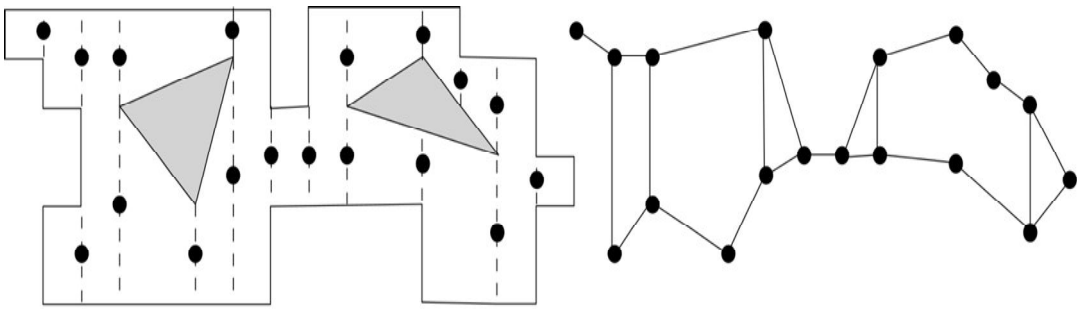


Figure 2.12 Construction des cellules trapézoïdales

2.5.2. Décomposition cellulaire boustrophédon

L'origine du mot boustrophédon remonte au 1699, où il a été introduit pour signifier «le chemin du bœuf». Pour couvrir tout le champ, le bœuf tire la charrue en allant dans des lignes droites adjacentes vers l'avant et vers l'arrière. Sur la base de cette idée, la décomposition cellulaire de boustrophédon est apparue comme une technique de planification de trajectoire orientée pour la couverture. Comme toute méthode de décomposition cellulaire, l'espace de configuration libre dans la méthode de boustrophédon est divisé en cellules adjacentes. Tout l'espace de configuration libre est franchi lorsque le robot se déplace d'une cellule à une autre en couvrant chacun allant en avant puis en arrière [Choset 86].

La génération des cellules dans la méthode de boustrophédon est basée sur le déplacement d'une tranche verticale vers les sommets des obstacles. A un type de ces sommets, la tranche ne peut pas être prolongée dans les deux côtés sans être diviser en deux morceaux. Ce type de sommets ne contribue à aucune génération de cellules. Au deuxième type de ces sommets, la tranche est encore reliée en s'étirant sur ses deux côtés. Ces genres de sommets s'appellent les sommets critiques. Pour construire les cellules complètes de décomposition de boustrophédon, on dessine l'ensemble des sommets critiques (figure 2.13).

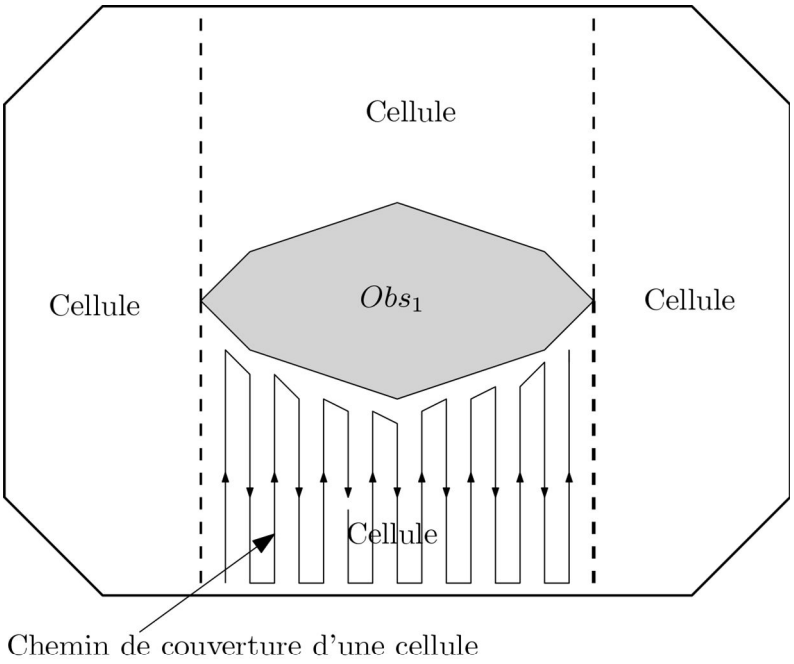


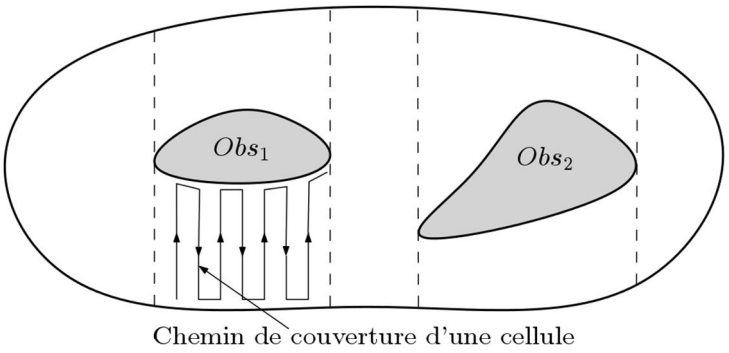
Figure 2.13. Construction des cellules boustrophédon

2.5.3. Décomposition cellulaire de Morse

La décomposition cellulaire de Morse est une forme générale de la décomposition cellulaire boustrophédon. La raison est que les obstacles peuvent prendre des formes autres que des polygones. Ces formes peuvent même être lisses, et n'ont donc aucun sommet. Par conséquent, au lieu d'utiliser les sommets d'obstacle pour la construction des cellules, une autre technique est proposée. Cette technique s'appelle la décomposition cellulaire de Morse. Elle est basée sur la définition d'une tranche comme une fonction appelé fonction de Morse. Ensuite, le balayage de cette tranche au-dessus de l'espace où le robot devrait se déplacer détermine la manière dont les cellules sont construites [Acar 02].

La tranche balayée change de connectivité toutes fois qu'elle rencontre ou laisse un obstacle. Dans l'espace libre, on considère que la tranche a une connectivité simple. Allant avant, la connectivité de la tranche change de simple en double au point où elle rencontre un obstacle. Ce point s'appelle un point critique et la tranche qui le traverse est une tranche critique. Pendant que la tranche continue son mouvement, elle part de l'obstacle. Au point de départ, sa connectivité change de double au simple. Ce point définit un autre point critique, tandis que la tranche qui le traverse est une autre tranche critique. En décomposition de Morse, ces tranches critiques, en plus du bord de l'espace et les limites des obstacles créent les cellules entières.

Le chemin de couverture est créé en deux étapes. La première consiste à générer l'ordre de passage des cellules. La deuxième étape détermine la façon de couvrir chaque cellule. Par conséquent, pour chaque cellule, le chemin est généré en avançant le long de la tranche et en revenant au bord d'un obstacle ou de la limite de l'espace de travail. Cependant, la largeur entre le chemin vers l'avant et vers l'arrière est habituellement déterminée soit par la taille du robot mobile, soit par l'organe terminal du manipulateur utilisé.



2.14. Décomposition cellulaire de Morse

2.6. Conclusion

Nous avons abordé, dans ce chapitre, une classe des méthodes conventionnelles de planification des trajectoires. D'abord, nous avons présenté les différentes catégories de cette classe qui sont le graphe de visibilité, les roadmaps, le champ potentiel et la décomposition cellulaire, puis nous avons indiqué leurs sous classes. Ensuite, nous avons expliqué leurs mécanismes. Enfin nous avons donné une vision sur les travaux qui ont employé ces méthodes dans le domaine de planification des trajectoires.

Chapitre 3

Les Approches Métaheuristiques pour la Planification des Trajectoires

3.1. Introduction

Dans ce chapitre nous montrons deux classes des métaheuristiques. La première est la classe des métaheuristiques à base population. Cette classe regroupe les algorithmes génétiques, les essaimes particulaires et les colonies de fourmis. La deuxième est la classe des métaheuristiques à base individu. Elle inclut le recuit simulé et la recherche tabou. Pour chaque métaheuristique, d'abord nous présentons leur origine et ensuite nous décrivons leur mécanisme. Puis nous présentons quelques travaux qui emploient ces méthodes pour résoudre le problème de planification des trajectoires.

3.2. Les algorithmes génétiques

3.2.1. Qu'est-ce qu'un algorithme génétique?

Inspirés de la génétique biologique, les algorithmes génétiques sont des algorithmes de recherche. Ils emploient des capacités humaines inventives pour former les algorithmes de recherche. Les algorithmes génétiques génèrent d'abord un ensemble aléatoire d'informations structurées. Ensuite, des nouveaux ensembles sont créés en suivant le mécanisme de la sélection naturelle. Les structures générées à l'origine échangent des informations entre elles. Les structures individuelles les plus fortes survivent alors que les plus faibles disparaissent. Les structures survivantes sont combinées pour former le nouvel ensemble. Encore une fois, les structures de cet ensemble suivent le même mécanisme de la génétique naturelle. Des informations sont échangées et le mécanisme de sélection naturelle est utilisé. La scène se termine par un ensemble récent de structures efficaces. Au cours de cette recherche, les algorithmes génétiques améliorent la performance de la solution potentielle grâce à une exploitation efficace des informations précédentes [Goldberg 89].

3.2.2. Principe d'un algorithme génétique

En 1975, John Holland et ses assistants [Holland 75] ont proposé les algorithmes de recherche génétique. Le but est de conceptualiser un algorithme évolutif similaire à celui du système d'évolution biologique et de fournir une interprétation facile à tout processus qu'ils emploient. Le système artificiel est conçu pour maintenir les processus significatifs du système naturel. Il imite la sélection naturelle ainsi que les mécanismes de réplication. De même, les bases théoriques sont dérivées du système naturel. La façon dont les

informations sont structurées, combinées, sélectionnées et régénérées est adoptée à partir de la manière dont le système naturel évolue.

Comme les algorithmes génétiques biologiques, les algorithmes génétiques artificiels utilisent un principe similaire. Pour un problème donné, un ensemble de solution potentielle est généré. Il s'appelle la population initiale. En général, pour chaque solution potentielle, un individu est créé. Chaque individu est représenté par un ou plusieurs chromosomes. À leur tour, ces chromosomes stockent l'information génétique structurée dans des petits composants appelés gènes. Ces informations génétiques sont soit mutuellement échangées grâce à une opération de croisement, soit modifiées au hasard par mutation. Le cycle est répété en sélectionnant les individus les plus adaptés pour former la population de la nouvelle génération. Ce processus de reproduction est terminé lorsque des solutions appropriées sont trouvées.

3.2.3. Mécanismes d'un algorithme génétique

Les algorithmes génétiques sont basés sur des processus d'évolution similaire des mécanismes de la génétique naturelle. Ces mécanismes concernent la reproduction génétique, le croisement et la mutation.

3.2.3.1. Reproduction génétique

Fondamentalement, le processus de reproduction des algorithmes génétiques est une forme non naturelle de son équivalent biologique. C'est un mécanisme par lequel les générations suivantes sont créées. Naturellement, la reproduction darwinienne implique que les organismes luttent pour survivre de la création à la production de nouveaux individus. En d'autres termes, les organismes doivent résister contre les obstructions empêchant les générations suivantes. Cela peut être vu comme la capacité d'empêcher les organismes de disparaître lorsqu'ils affrontent des maladies et des prédateurs. En effet, certains des organismes prospèrent tandis que d'autres souffrent. De même, la capacité de croissance des individus artificiels est mesurée en termes de leur fonction objective. Cela signifie que la valeur de la fonction objective des individus est déterminante. Les individus ayant une meilleure valeur de fonction objective ont plus de chances de survivre alors que les individus ayant une mauvaise valeur de fonction objective disparaissent. En conséquence, les individus favorisés sont copiés et leurs informations génétiques sont répliquées.

3.2.3.2. Croisement

L'opérateur de reproduction permet de sélectionner des individus favorisés et de reproduire exactement leur information génétique. L'opération suivante, le croisement, combine l'information génétique des individus sélectionnés pour créer les enfants de la nouvelle génération. Cette opération consiste à échanger des parties de matériel génétique entre individus parents pour permettre la création d'enfants. C'est une autre imitation de l'opération naturelle. Les enfants artificiels, comme les biologiques, combinent les qualités de leur parent.

3.2.3.3. Mutation

La mutation est une modification aléatoire d'une partie de l'information génétique des individus favorisée. Il joue un rôle important pour maintenir une exploration globale. Pendant la reproduction et le croisement, les individus sont sélectionnés et combinés pour former une nouvelle génération, parfois les individus peuvent partager des informations identiques. À ce stade, l'algorithme de recherche n'assure pas la convergence vers un meilleur optimum. Pour éviter ce problème, la mutation est utilisée comme opération secondaire pour empêcher l'homogénéité de la population. Par conséquent, lorsque l'algorithme génétique maintien des altérations accidentelles, la recherche se termine avec des résultats optimaux.

3.2.4. Planification des trajectoires par les algorithmes génétiques

Les algorithmes génétiques ont vu de nombreuses améliorations dans le domaine de la planification des trajectoires des robots. Certains d'entre eux sont orientés vers la planification des trajectoires des robots mobiles, tandis que d'autres se concentrent sur les manipulateurs. Dans [Luis 13], le chemin du robot manipulateur est construit comme un groupe de primitives géométriques. Ces primitives peuvent prendre la forme de lignes, d'arcs, de segments circulaires et de courbes de Bézier. Chacune de ces primitives est considérée comme un vecteur et libellé par ses extrémités. Le manipulateur est demandé de se déplacer à travers un chemin donné tout en minimisant la distance parcourue. Les extrémités des éléments de base du chemin doivent être visitées par le point central de l'organe terminal au moins une fois. Malheureusement, le chemin peut avoir des parties détachées où certains points sont isolés les uns des autres, et le manipulateur doit se déplacer dans l'espace libre pour les rejoindre. Pour résoudre ce problème, un algorithme

génétique est utilisé. Les individus sont représentés par un vecteur, avec un ensemble des extrémités des segments comme composants. Pour mesurer la qualité de chaque individu, la distance euclidienne est utilisée comme fonction objective. C'est la distance totale où le robot n'effectue rien d'autre que de se repositionner dans point d'extrémité du vecteur. Pour minimiser la fonction objective, il est naturel de suivre une séquence et une direction optimale des parties du chemin. Par conséquent, le croisement des individus avec un codage réel est utilisé pour assurer une combinaison optimale. Pour éviter de piéger dans les minima locaux, le deuxième opérateur, la mutation, est utilisé pour basculer entre deux composants dans la même séquence. Enfin, l'algorithme génétique s'arrête lorsque le nombre maximal de générations est atteint.

La planification de la trajectoire de deux manipulateurs partageant le même espace de travail est décrite dans [Victor 98]. Chaque mouvement est construit sur la base de la configuration du robot voisin, où une seule articulation de robot est déplacée à chaque fois. Le chemin complet est structuré comme une séquence des articulations tournantes et de leurs directions. Ces informations sont codées comme chaînes d'individus, et utilisées par un algorithme génétique coopératif pour trouver un chemin sans collisions pour les deux manipulateurs. Contrairement à l'algorithme génétique standard qui emploie une seule population, l'algorithme génétique coopératif utilise deux populations différentes. Le meilleur individu à chaque génération de la première population est envoyé à l'autre. La deuxième population considère cet individu comme un obstacle en mouvement. Quand l'algorithme coopératif recherche un chemin optimal, il exclut l'individu reçu car il décrit l'obstacle. L'évitement de collision est maintenu en utilisant une fonction de pénalité. Cependant, un chemin optimal n'est pas défini seulement par la fonction de pénalité, mais aussi par la distance parcourue et l'erreur entre la cible désirée et celle atteinte. Ainsi, pour mesurer la qualité d'un individu, tous ces paramètres sont combinés dans la fonction objective principale. Enfin, l'approche est testée avec succès en utilisant deux robots planaires qui partagent le même espace de travail.

Dans [Ahmad 91] Les algorithmes génétiques sont utilisés pour trouver un chemin optimal pour un manipulateur redondant à trois degrés de liberté. Le manipulateur doit se déplacer du point de départ au but tout en évitant les obstacles proches. Ce mouvement est assuré en incrémentant les articulations du robot. Le chemin est la courbe représentée par l'effecteur du manipulateur. Le rôle de l'algorithme génétique est de rechercher la séquence

des mouvements réalisables qui permettent au robot d'atteindre la cible en toute sécurité. Initialement, les mouvements sont codés par trois bits avec trois états chacun. La fonction objective est utilisée pour tester la qualité de chaque mouvement. Les critères principaux de la fonction objective sont l'évitement des obstacles et la distance entre l'effecteur et la cible. L'algorithme génétique tente de combiner un nombre maximum de dix mouvements pour que l'effecteur atteint la cible. Malheureusement, les dix mouvements ne sont pas toujours suffisants, et l'algorithme génétique est obligé d'utiliser à plusieurs reprises le dernier mouvement comme point de départ pour initier un autre essai. Le processus est arrêté, lorsque le but est atteint. Cependant, dans les cas où les déplacements n'apportent aucune amélioration, le robot est considéré comme piégé. L'algorithme génétique bascule le dernier mouvement dans l'autre direction et suit le même cycle de dix mouvements jusqu'à ce que l'effecteur atteint la cible.

3.3. Les essaims particulaires

3.3.1. Origine des essaims particulaires

À l'origine, les essaims particulaires ont été créés pour interpréter les comportements des créatures. Comme les chercheurs étaient ravis par les comportements naturels des créatures, ils suggèrent différents modèles pour les imiter. La simulation par modèle graphique est utilisée pour imiter le mouvement harmonisé des essaims d'oiseaux [Reynolds 87]. Le modèle indique que le comportement des oiseaux est déterminé par l'instinct de suivre leurs voisins respectant leurs vitesses. De même, un modèle de jeu est utilisé pour étudier les règles qui contrôlent le comportement des oiseaux [Heppner 90]. Ce modèle est basé sur la coordination du mouvement des oiseaux artificiels, où les actions naturelles des oiseaux sont artificiellement imitées par un programme informatique. En plus de suivre leurs voisins et de garder la distance avec eux, les règles de contrôle supposent l'utilisation des forces attractives qui permettent aux oiseaux d'atteindre leurs perchoir.

Pas loin de l'explication du comportement des oiseaux, le socio biologiste [Wilson 75], émet l'hypothèse que, durant leur recherche de nourriture, les membres d'espèces de poissons profitent de l'expérience de leurs partenaires. Les découvertes de certains membres sont utilisées par l'ensemble d'espèces. En d'autres termes, les informations partagées entre les individus du même groupe permettent aux autres de trouver la nourriture. Ce comportement suggère que les poissons évitent la compétition en faveur de

l'objectif principal de la recherche de la nourriture. Cela implique également que les membres de la même espèce emploient une sorte de mécanisme de recherche évolutif.

3.3.2. Les essaims particulaires et l'optimisation

Les travaux des chercheurs de la science naturelle sur les essaims d'oiseaux et sur les espèces de poisson fournissent des aperçus utiles à Kennedy et Eberhart [Kennedy 95]. A l'origine, les chercheurs ont essayé d'imiter les mouvements habiles des oiseaux. Les agents générés ont été guidés par les règles qui les font se déplacer simultanément à leur perchoir artificiel. Pour eux, l'imitation de la recherche d'oiseaux du perchoir connu est fascinante, mais le défi est de trouver comment les oiseaux découvrent un lieu de nourriture imprévisible. Certainement, c'était le point de transformation, parce qu'il permet aux chercheurs de décaler leur attention d'imitation de comportement aux problèmes d'optimisation.

Stimulés par le comportement de recherche de nourriture des oiseaux et des poissons, Kennedy et Shi [Kennedy 95, Shi 98] ont proposé l'approche d'optimisation par les essaims particulaires. C'est un paradigme de recherche d'aliments artificiels. Cela signifie que les agents modélisés suivent le même principe de recherche naturel des oiseaux et des poissons. Comme certains agents trouvent la nourriture, les autres les suivent. Ainsi, l'algorithme génère initialement un ensemble de solutions potentielles appelées particules. Ensuite, il évalue la qualité de ces particules en utilisant la fonction coût. Comme leurs analogues naturels, les particules artificielles améliorent leur position et leur vitesse pour suivre les autres membres de la population. Ce processus se poursuit tant que la solution optimale n'est pas découverte. Cependant, une fois trouvé, la solution optimale attire d'autres particules. Enfin, cette étape fournit non seulement une solution optimale mais un ensemble de solutions.

3.3.3. Planification des trajectoires par les essaims particulaires

L'algorithme d'optimisation par les essaims particulaires, en version standard ou en version améliorée, est utilisé par de nombreux chercheurs pour résoudre le problème de la planification des trajectoires. Wen [Wen 09] propose un algorithme des essaims particulaires modifiée pour la planification de mouvement d'un manipulateur à deux degrés avec couplage dynamique. Initialement, l'algorithme des essaims particulaires standard est modifié pour limiter la vitesse des particules. Cette modification est basée sur

l'introduction de ce qui est appelé une contrainte dynamique de la vitesse. Le but est d'assurer une distribution uniforme des particules et d'accélérer la convergence de l'algorithme. Ensuite, le problème de planification de chemin est utilisé pour tester l'efficacité de l'algorithme. Comme le couplage dynamique du manipulateur nécessite des variables articulaires élevées, les limites des articulations sont utilisées comme des contraintes dans la planification de la trajectoire. De plus, la fonction objective est formulée en incluant les couples appliqués au robot. Puisque ces couples sont décrits par une fonction B-spline du troisième ordre, le problème de planification de trajectoire est transformé en un problème d'optimisation des points de contrôle de cette fonction. En simulation, la performance de l'algorithme proposé est comparée à celle de la programmation quadratique séquentielle. Les résultats montrent que l'algorithme d'optimisation par les essaims particulaires génère des couples plus souples par rapport au second algorithme qui dépend fortement des conditions initiales.

Guo [Guo 10] examine l'utilité de l'algorithme d'optimisation par les essaims particulaires pour la planification de la trajectoire d'un manipulateur redondant. Le problème fondamental consiste à concevoir une trajectoire, dans un environnement qui inclut des obstacles, pour un manipulateur redondant à trois degrés. Dans un premier temps, au lieu de la trajectoire cartésienne, une trajectoire articulaire est utilisée pour éviter le problème de redondance. Cette trajectoire est composée de deux segments différents. Le segment reliant la position initiale à la position intermédiaire est modélisé par un polynôme d'ordre quatre, tandis que l'autre, continuant de la position intermédiaire à la position finale, est décrit par un polynôme quantique. L'ensemble de la trajectoire est optimisée par une version améliorée des essaims particulaires appelée 'quantum particle swarm optimization' «QPSO». Les principaux changements dans cette version concernent la génération chaotique de la population initiale et le mécanisme d'adaptation des particules. Cependant, le mécanisme de mesure des particules basé sur la fonction objective reste inchangé. Il est formulé pour inclure différents aspects, qui sont la quantité de violation du couple maximum, les distances articulaires et cartésiennes totales, le temps total de mouvement et l'évitement d'obstacle. En conséquence, cette fonction est optimisée par QPSO, et des trajectoires optimales sont générées même en présence d'un ou deux obstacles dans l'environnement. Enfin, le QPSO est comparé à l'algorithme génétique et son efficacité est prouvée.

Lin [Lin 13] suggère d'utiliser l'optimisation par les essaims particulaires pour trouver une trajectoire de jerk minimale pour un manipulateur à six degrés de liberté. Initialement, la trajectoire de chaque articulation est décrite comme une interpolation point à point des nœuds de la spline cubique. Ensuite, le jerk de la trajectoire est simplement obtenu de la troisième dérivée de la trajectoire articulaire où il est conclu qu'il dépend fortement des intervalles du temps entre les nœuds. Cette conclusion conduit à la formation de deux fonctions objectives différentes mais avec les mêmes paramètres de la variable jerk. En conséquence, la technique d'optimisation par essaims particulaires est employée pour trouver le jerk optimal. Le processus commence par former une population initiale en utilisant une technique de projection. Comme les paramètres des fonctions objectives sont des intervalles de temps, les particules ayant des valeurs négatives sont considérées comme irréalisables et donc éliminées. La qualité du reste de la population est mesurée et l'algorithme de classification K-means est introduit pour trouver la meilleure particule globale. Dans le cas où le reste des particules est proches de la meilleure, les particules sont groupées et testées avec les meilleures des générations précédentes pour vérifier si le critère d'arrêt est satisfait. Cependant, lorsque le meilleur est différent du reste de la population, les positions des particules sont mises à jour au moyen de la technique du facteur de constriction [Clerk 99]. A ce niveau, des positions aléatoires sont générées pour remplacer les particules irréalisables. De nouveau, la fonction objective est utilisée pour mesurer la qualité des particules de la nouvelle population et le cycle continue jusqu'à ce que le critère d'arrêt soit satisfait. Une fois trouvée, la meilleure particule définit les intervalles de temps minimum entre les nœuds. Finalement, avec les deux fonctions objectives, la technique proposée découvre avec succès la trajectoire de jerk minimum, avec cinq nœuds, du manipulateur utilisé.

Yueqiang [Yueqiang 14] propose une autre version de l'algorithme des essaims particulaires pour résoudre le problème de planification des trajectoires du manipulateur de soudage. Le chemin est décrit par des points de soudure qui doivent être traversés par le manipulateur de soudage. Il part de la position initiale, traverse tous les points de soudure intermédiaires et revient à la position initiale. Par conséquent, pour s'assurer que chaque point de soudure intermédiaire n'est franchi qu'une seule fois, la fonction objective est formulée de façon à inclure toute la distance entre les points de soudure. De cette manière, le problème de planification des trajectoires est transformé en un problème de recherche de la meilleure séquence dans le chemin de soudage. Les solutions potentielles de ce

problème sont les particules où la qualité de chaque particule est mesurée en fonction des facteurs assignés par cette particule aux points de soudure. Plus les facteurs sont appropriés, meilleure est la séquence des points traversés par le robot. Enfin, pour accélérer le mécanisme de recherche, au lieu d'utiliser l'algorithme des essaims particulaires standard, l'algorithme des essaims particulaires amélioré est utilisé pour mettre à jour les emplacements des particules.

3.4. Les colonies de fourmis

La nature inspire les chercheurs, même avec les insectes minuscules. De nombreuses observations naturelles motivent les biologistes à trouver le secret derrière le comportement social collaboratif des insectes en général et la colonie de fourmis en particulier. Ils mènent de nombreuses expériences pour étudier comment les fourmis parviennent à structurer leurs œuvres, notamment lorsqu'elles cherchent de la nourriture. La plupart des résultats démontrent qu'être aveugle ou avoir des sens primitifs n'empêche pas les fourmis d'interagir. En effet, les fourmis utilisent la communication implicite plutôt que l'interaction explicite. Ce comportement social n'est rien d'autre que l'échange d'informations partagées entre les fourmis de la même espèce. Plus précisément, les fourmis maintiennent la communication à travers la substance chimique appelée phéromone. Quand elles vont de leur nid à la recherche d'un endroit de la nourriture, ou quand elles y reviennent, certaines fourmis forment un chemin entier en laissant derrière elles les phéromones. En suivant les traces de phéromone par leurs partenaires, le reste de la colonie peut atteindre la destination à la nourriture [Marco 04].

3.4.1. Les colonies de fourmis et la planification des chemins

Un bon exemple de planification de chemin naturel est l'expérience bien connue du double pont [Goss 89], qui illustre comment des insectes aussi petits que les fourmis peuvent trouver le chemin le plus court entre leur nid et la destination à la nourriture. Cette expérience commence en reliant le nid des fourmis Argentines et la destination à la nourriture avec un pont à deux segments. Chaque segment est composé de deux branches, de sorte que la longueur de l'une est deux fois la longueur de l'autre. Les segments sont connectés ensemble de manière opposée en trois points: le point de départ, le point milieu et le point final. Cela signifie que la longue branche de chaque segment fait face à la branche courte de l'autre. Lorsque les fourmis commencent à se déplacer, elles choisissent uniformément les branches à partir desquelles elles peuvent atteindre la nourriture. Ce

comportement dure quelques instants. Cependant, après un certain temps, le nombre de fourmis prenant le chemin le plus court commence à augmenter jusqu'à ce qu'aucune fourmi ne prenne les autres branches.

Le changement de comportement des fourmis est interprété comme suit. Au départ, il n'y a pas de phéromone sur les branches. Par conséquent, les fourmis choisissent librement leurs voies à la nourriture. En se dirigeant vers la destination, les fourmis en mouvement marquent tous les chemins disponibles par leurs phéromones. Mais, une fois la destination est atteinte, la fourmi qui revient à son nid ajoute plus de phéromone. À ce moment, la phéromone sur le plus court chemin est augmentée en premier. Pendant ce temps, plus de fourmis sont attirées par le niveau d'augmentation de la phéromone. Cette action modifie les préférences du reste des fourmis pour choisir le même chemin le plus court. Enfin, en raison de l'accumulation de phéromones, le chemin le plus court est favorisé par toutes les fourmis.

Bien que les fourmis choisissent parfaitement le chemin le plus court, leurs préférences sont représentatives d'une situation spécifique. Lorsque des fourmis sont proposées, en même temps, par des chemins différents, leurs interactions favorisent le chemin le plus court. Cependant, après avoir été choisis, les fourmis ne peuvent pas passer à un autre chemin encore plus court. Cette situation est étudiée dans une expérience différente où les fourmis déjà établissent leur chemin le plus court à partir des chemins disponibles. Une demi-heure plus tard, un nouveau chemin encore plus court est ajouté. Néanmoins, les fourmis ne peuvent pas échapper du chemin déjà choisi. Le niveau élevé de phéromone, sur ce chemin, empêche les fourmis de choisir le nouveau chemin. Ceci élimine le choix du chemin optimal, sauf si la phéromone est totalement évaporée.

3.4.2. Le système des colonies de fourmis

Pour analyser les mécanismes qui conduisent les fourmis à trouver un chemin optimal à travers leur comportement social naturel, différents modèles sont suggérés. Le modèle du système des colonies de fourmis qui est utilisé pour résoudre le problème d'optimisation bien connu du voyageur de commerce est parmi les modèles les plus ancien [Colorni 92]. Ce modèle définit les fourmis comme des simples agents mobiles qui décrivent des solutions potentielles du problème. En outre, la probabilité qu'un agent visite une ville donnée est basée sur trois paramètres fondamentaux qui sont l'intensité de la trace de phéromone, son facteur d'évaporation et le paramètre de visibilité.

A un instant donné (t + 1), l'intensité de la trace de phéromone $\tau_{ij}(t+1)$ est donné par :

$$\tau_{ij}(t+1) = \rho\tau_{ij}(t) + \Delta\tau_{ij}(t, t+1)$$

Où ρ est le facteur d'évaporation.

Cette intensité dépend aussi de la quantité de phéromone laissée derrière la k^{ième} fourmi sur le chemin *ij* dans le temps t, t + 1 qui est décrit par:

$$\Delta\tau_{ij}(t, t+1) = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}(t, t+1)$$

La probabilité pour chaque fourmi de se déplacer d'un site *i* vers un autre site *j* est défini par:

$$p_{ij}(t) = \frac{\left[\tau_{ij}(t)\right]^{\alpha}\left[\eta_{ij}\right]^{\beta}}{\sum_{j=1}^{n}\left[\tau_{ij}(t)\right]^{\alpha}\left[\eta_{ij}\right]^{\beta}}$$

Où η_{ij} est la visibilité du site *j* pour le site *i*, et α, β sont des paramètres de pondération.

Poursuivant ce modèle, différentes améliorations sont apportées. Par exemple, dans [Dorigo 96], la version initiale est mise à jour pour que la meilleure fourmi laisse derrière elle plus de phéromone que les autre. Cette version vise à influencer les préférences des autres fourmis pour choisir le meilleur chemin. De même, une autre version de mise à niveau de la version initiale est proposée par [Dorigo 97a, 97b]. Cette version est basée sur l'introduction d'un facteur aléatoire par lequel l'algorithme est orienté vers la diversification ou l'intensification. Cela signifie que la valeur du facteur peut favoriser l'intensité de la phéromone déposée ou la visibilité des sites.

3.4.3. Planification des trajectoires par les colonies de fourmis

Brand [Brand 10] propose l'utilisation d'un algorithme de colonie de fourmis pour résoudre le problème de planification des trajectoires dans un environnement dynamique. Initialement, ce problème est formulé pour trouver le chemin le plus court, entre une position initiale et une position cible, dans un modèle de réseau de grille d'un environnement sans obstacle. Différents chemins sont générés et mis à jour de manière aléatoire selon les mécanismes des colonies de fourmis. Ensuite, le chemin avec un niveau élevé de phéromone est considéré comme le chemin le plus court. Cependant, dès que les

obstacles sont ajoutés à l'environnement, le niveau de phéromone sur le chemin le plus court est ajusté et le processus de colonie de fourmis est initié. D'une part, lorsque l'ajustement local est adopté, le niveau de phéromone est modifié uniquement autour de l'obstacle. Ce type d'ajustement nécessite peu d'itération supplémentaire pour trouver la nouvelle route la plus courte vers la cible. D'un autre côté, une fois l'ajustement global effectué, le niveau de phéromone de l'environnement entier est réinitialisé à sa valeur initiale. Par conséquent, beaucoup d'itérations supplémentaires sont nécessaires pour trouver le nouveau chemin le plus court. En effet, plus la taille du réseau est grande, plus la différence entre les itérations supplémentaires, requises par l'initialisation locale et globale, est importante.

Zho [Zho 16] emploie l'algorithme des colonies de fourmis pour la planification de trajectoire d'un manipulateur hyper redondant. L'objectif principal est de résoudre la cinématique inverse du manipulateur. En d'autres termes, générer les articulations nécessaires pour déplacer l'effecteur du robot vers la position cible. Cependant, le manipulateur redondant peut atteindre la même cible à travers différentes configurations. Par conséquent, pour trouver la configuration appropriée, le problème de planification de mouvement est formulé de manière à inclure de nombreuses exigences telles qu'un mouvement minimal des articulations, un test d'auto collision, l'évitement d'obstacles et l'erreur entre la cible et l'effecteur. Ces exigences sont définies comme une fonction objective et utilisées par l'algorithme de colonie de fourmis pour trouver une trajectoire optimale. Initialement, de nombreux chemins sont générés de manière aléatoire. Ces chemins sont équivalents aux positions des fourmis. Ensuite, les chemins qui satisfont toutes les contraintes de mouvement sont mis à jour. Ce processus est répété jusqu'à ce qu'une trajectoire optimale soit trouvée. Enfin, l'algorithme de colonies de fourmis, comprenant des populations de trois tailles différentes, est implémenté pour rechercher une trajectoire optimale pour un manipulateur à dix degrés de liberté. Les résultats montrent que plus la taille de la population est élevée, meilleure est la trajectoire.

Jin [Jin 16] utilise un algorithme d'optimisation hybride pour résoudre le problème de planification de trajectoire d'un manipulateur à six degrés de liberté. Cet algorithme injecte les meilleurs caractéristiques des essaims particulaires et de l'algorithme génétique dans l'algorithme de la colonie de fourmis. Au début, le problème est formulé pour trouver la distance de déplacement minimum qui permet au robot de traverser tous les points de

soudure dans l'espace de travail. Ensuite, cette formulation est introduite comme la fonction objective principale du problème d'optimisation. En conséquence, le processus d'optimisation se déroule comme suit. Une fois les chemins sont générés, les mécanismes des essaims particulaires sont introduits et les meilleures positions personnelles et globales des fourmis sont déterminées en utilisant la valeur de la fonction objective. Ces informations et le niveau de phéromone dans chaque segment sont utilisés par les fourmis pour trouver leur cible. Le meilleur chemin de tous les chemins disponibles est déterminé et le mécanisme de vérification est introduit pour mesurer sa validité. A ce moment, il n'y a pas d'amélioration du trajet à moins qu'une augmentation significative de la valeur objective soit mesurée. Par conséquent, le mécanisme de mutation de l'algorithme génétique est utilisé pour apporter une nouvelle diversification au processus de recherche. Encore une fois, la meilleure solution est déterminée à partir des chemins disponibles et sa validité est vérifiée. L'ensemble du processus est répété jusqu'à ce qu'un nombre maximum d'itérations est atteint.

Afin de vérifier l'efficacité de l'algorithme hybride par rapport à l'algorithme standard de colonies de fourmis, les deux algorithmes reçoivent les mêmes données à l'entrée. Cela comprend une population initiale de trente fourmis, un nombre maximum de cinquante itérations et cinquante points de soudure. Le chemin de sortie obtenu par l'algorithme standard comprend des intersections entre les points de soudure. Cela signifie que d'autres améliorations peuvent être utilisées. Cependant, l'algorithme hybride fournit un chemin optimal qui inclut aucune intersection entre ses segments. En fait, quand le chemin obtenu est exécuté par le robot, Puma 560, de soudage, sa faisabilité est facilement confirmée. Par conséquent, la conclusion est que l'algorithme hybride surpasse complètement l'algorithme standard.

Baghli [Baghli 17] utilise l'algorithme de colonie de fourmis pour trouver un chemin optimal pour un manipulateur à deux bras. A l'origine, l'environnement du manipulateur est rempli de cinq obstacles cubiques. Ensuite, cet environnement est modélisé comme un réseau de grille avec cinq régions rectangulaires interdites. Ces régions d'obstacles sont dispersées dans le réseau de grille de sorte que des espaces libres sont laissés entre eux. L'algorithme de colonie de fourmis est utilisé pour trouver un chemin optimal qui évite les obstacles statiques et relie le point de départ à la cible. Ce but est atteint en employant simplement tous les mécanismes de la colonie de fourmis, depuis

la génération aléatoire des positions initiales des fourmis jusqu'à la mise à jour du niveau de phéromone. Par conséquent, une population initiale de vingt fourmis est générée. Ensuite, les positions des fourmis sont mises à jour en fonction de leur visibilité et de l'intensité de la phéromone déposée. D'une part, la visibilité des fourmis est mesurée en utilisant l'exigence principale de la fonction objective, qui est la distance parcourue. D'autre part, le niveau de phéromone est ajusté proportionnellement aux facteurs d'évaporation et d'intensité. Enfin, un compromis entre intensification et diversification permet à l'algorithme de trouver un chemin optimal après cent générations en prenant deux secondes.

3.5. Recuit simulé

En physique de la matière condensée, les analyses de l'effet de l'ajustement de la température sur l'état final des matériaux conduisent à distinguer deux processus différents. D'une part, la trempe est le processus par lequel un matériau est chauffé en augmentant sa température jusqu'au niveau où il devient liquide, suivi d'un refroidissant rapide où il forme un solide. Dans la première phase, en raison de la température élevée, les atomes du liquide sont répartis de manière aléatoire. Le refroidissement rapide ne permet aucune stabilité thermique. Par conséquent, l'état final du matériau est un solide amorphe métastable. D'autre part, le recuit est le processus par lequel, au lieu de refroidissement rapide, une diminution de température progressif est suivie. De la même manière que la trempe, à haute température, le matériau fondu est constitué d'un grand nombre d'atomes non structurés. Cependant, en refroidissant lentement le liquide et laissant le matériau atteint un équilibre thermique, l'état final qui en résulte est un cristal solide avec une énergie minimale [Kirkpatrick 83].

L'algorithme le plus ancien pour simuler l'équilibre thermique d'un ensemble de particules est proposé par Metropolis [Metropolis 53]. Initialement, les particules sont placées dans un réseau régulier à une température donnée. Ensuite, toutes les particules, dans la configuration d'origine, sont déplacées consécutivement dans le plan. Dans chaque cycle, une particule donnée est exposée à un petit déplacement arbitraire et le changement de l'énergie du système est calculé. Dans cette phase, lorsque l'énergie du système est réduite, la nouvelle position de la particule est acceptée. Cependant, lorsque l'énergie du système est augmentée, la nouvelle position de la particule peut également être acceptée. La probabilité d'accepter ce mouvement est égale à exp ($-\Delta E$ / $K_B T$). Par conséquent,

lorsque la probabilité d'acceptation est supérieure à une probabilité générée au hasard, la nouvelle position est autorisée; sinon, l'ancien est maintenu. Enfin, le système continue d'évoluer, suivant la distribution de Boltzmann, jusqu'à atteindre l'équilibre thermique.

Le recuit simulé pour un problème d'optimisation fonctionne de manière analogue à l'algorithme de Metropolis qui imite le processus d'un système physique [Kirkpatrick 83]. L'énergie du système physique est remplacée par la fonction objective tandis que les configurations sont remplacées par les paramètres de la fonction objective. Par conséquent, la simulation du processus de recuit d'un problème donné implique d'élever la température jusqu'à atteindre le niveau de fusion et de la réduire graduellement jusqu'à aboutir le niveau de congélation. A chaque température, la configuration d'origine est perturbée et la valeur de la fonction objective est calculée. Quand elle est réduite, la nouvelle configuration est acceptée. Cependant, avec une augmentation de la valeur de la fonction objective, cette nouvelle configuration est acceptée si la probabilité est supérieure à une probabilité arbitraire et rejetée autrement. Ce cycle continue jusqu'à ce que l'équilibre thermique soit atteint. En suivant ce processus jusqu'à la température de congélation, la configuration du système à un équilibre thermique est considérée comme la solution au problème d'optimisation.

3.5.1. Planification des trajectoires par recuit simulé

Garg [Garg 02] applique deux techniques d'optimisation différentes pour trouver une solution optimale au problème de planification de trajectoire dans deux situations différentes. D'une part, le problème de trouver un chemin optimal avec un couple minimum d'un manipulateur à deux degrés de liberté est considéré. Le chemin de chaque articulation est défini comme un polynôme d'ordre quatre où ses facteurs sont combinés en termes d'un seul facteur. Les facteurs dépendants sont considérés comme les paramètres de la fonction objective. Par conséquent, la recherche d'un couple minimal à appliquer aux articulations du robot correspond à la recherche des valeurs optimales de ces paramètres. D'autre part, le problème de planification de trajectoires de deux robots coopérants, avec deux degrés de liberté chacun, est étudié. L'objet manipulé par les deux robots ajoute plus de paramètres au problème. En plus des paramètres du polynôme qui définit la trajectoire, deux paramètres supplémentaires, résultant des forces appliquées par l'un des robots sur l'autre, sont considérés. Tous ces paramètres sont pris en compte dans les fonctions objectives qui considère le couple total des deux robots. Une fois les fonctions objectives

sont formulées, les algorithmes génétiques ainsi que le recuit simulé adaptatif sont utilisés pour résoudre le problème. Comme les algorithmes génétiques basent leur recherche des paramètres optimaux, qui donnent un couple minimal, sur les opérateurs génétiques, le recuit simulé adaptatif considère l'ajustement de ses paramètres comme une clé pour résoudre le problème. Enfin, la comparaison entre les deux techniques montre que le recuit simulé adaptatif surpasse les algorithmes génétiques en termes de convergence.

Peng [Peng 06] utilise une approche hybride d'algorithme génétique et de recuit simulé pour déterminer une trajectoire optimale pour un manipulateur planaire à trois degrés de liberté. Dans le nouveau algorithme, les opérateurs génétiques standards sont soumis à de nombreuses modifications où les procédures de recuit simulé sont adoptées. Par exemple, la sélection proportionnelle est transformée de sorte que les meilleurs individus ne peuvent pas être favorisés. Le nouveau mécanisme de sélection prend en compte la génération de l'individu, les valeurs extrêmes de la fonction objective dans la génération et le nombre maximum de générations. Aussi, après le croisement, les individus des nouvelles générations sont sélectionnés différemment. Lorsque les valeurs de la fonction objective des enfants sont meilleures que celles de leurs parents, les premières sont sélectionnées, sinon, le mécanisme de Boltzmann, utilisé dans le recuit simulé, est mis en œuvre pour choisir entre chaque enfant et son parent. De même, une procédure identique est suivie pour déterminer qui reste entre l'individu muté et son précédent. Suite à cette adaptation, le problème de planification de trajectoire est transformé en un problème de minimisation du déplacement total des articulations. Ce problème est résolu en utilisant l'algorithme hybrid, ainsi que l'algorithme génétique standard. Enfin, les performances des deux algorithmes sont comparées, où il est conclu que l'algorithme hybrid surpasse l'algorithme standard.

L'algorithme de recuit simulé est utilisé par Miao [Miao 08] pour résoudre le problème de planification du chemin d'un robot mobile dans un environnement dynamique. Comme l'environnement comprend à la fois des obstacles statiques et mobiles, deux paradigmes de planification des trajectoires sont mis en œuvre. D'une part, une planification hors ligne est établie avant le départ du robot. A cette étape, l'algorithme de recuit simulé utilise les informations disponibles sur les sommets des obstacles statiques pour générer une trajectoire optimale. L'algorithme commence par évaluer une solution générée aléatoirement par rapport à une autre nouvelle solution. Cette évaluation est

équivalente au calcul des valeurs de la fonction objective des solutions qui sont basées sur la distance totale des sommets des obstacles choisis. Selon le mécanisme de Boltzmann introduit, même la mauvaise solution peut survivre à la génération suivante. Cette solution est mise à jour, à chaque séquence, jusqu'à ce que la température finale soit atteinte. Cependant, il est important de noter que la solution aléatoire de chaque génération est principalement basée sur la suppression de certains sommets plutôt que sur la commutation entre eux. D'un autre côté, quand le robot se déplace, il peut activer l'algorithme de recuit simulé pour effectuer un ajustement en ligne de la trajectoire en fonction des obstacles mobile. A cette étape, la vitesse et la direction des obstacles sont obtenues par le capteur monté sur le robot. Par conséquent, ces informations sont utilisées par le robot pour continuer avec la même trajectoire si aucune collision n'est anticipée, ou effectuer un ajustement en ligne de la trajectoire lorsqu'une collision est prévue. En conclusion, les performances de cette technique sont testées efficacement en utilisant quatre environnements différents.

Jianjun [Jianjun 17] mis en œuvre l'algorithme de recuit simulé pour trouver une trajectoire à temps optimale pour un robot à roues. Dans un premier temps, le robot est modélisé comme un mécanisme simplifié de quatre degrés de liberté. Ensuite, la trajectoire de chaque articulation est définie par un polynôme quantique. Comme les vitesses, accélérations et jerk articulaires sont simplement dérivées de l'équation polynomiale, les contraintes sur les variables articulaires sont utilisées pour gérer les limites de la dynamique du robot. En conséquence, ces contraintes sont définies comme des fonctions de pénalité et ajoutées au temps de parcours pour formuler la fonction objective principale. L'algorithme de recuit simulé est appliqué pour trouver la trajectoire à temps optimale. Le processus itératif de l'algorithme se déroule comme suit. Une solution initiale et une température d'échauffement élevée sont établies. Ensuite, une autre solution aléatoire est générée. Les deux solutions sont évaluées et comparées. Lorsque la valeur la fonction objective de la nouvelle solution est meilleure, elle remplace l'ancienne; sinon, sa probabilité d'être remplacé est calculée. Lorsque cette valeur est supérieure à une valeur aléatoire, la nouvelle solution est sélectionnée pour la génération suivante; sinon l'ancienne est maintenue. Ce processus est répété, tandis que la température est réduite linéairement, jusqu'à ce qu'une température finale soit atteinte. Enfin, une trajectoire à temps optimal inférieur à trois secondes est trouvée.

3.6. Recherche tabou

La recherche tabou est une approche métaheuristique proposée par Glover [Glover 86]. L'approche évite les optima locaux sur la base d'un mécanisme fondamental qui favorise le contrôle de nouvelles solutions par rapport aux anciennes. Ce mécanisme enregistre les déplacements vers les solutions récemment examinées et les marque comme des mouvements interdits ou tabou. De cette façon, la recherche tabou commence par l'utilisation du même principe que les méthodes de gradient, mais, dès que la recherche est piégée dans un optimum local, elle favorise le choix de déplacer vers une mauvaise solution sur la visite d'une solution déjà examinée. Généralement, la restriction des déplacements vers des solutions précédents est flexible dans le sens que ces solutions ont un statut tabou temporaire et peuvent donc être visités à nouveau.

3.6.1. Principes de la recherche tabou

Un moyen efficace de faciliter la mise en œuvre de la recherche tabou, après le succès obtenu, consiste à décrire les principes fondamentales de l'approche. Ces principes sont établis par Glover [Glover 86, 89] où les plus importants sont : la mémoire à court terme, la mémoire à long terme et le critère d'aspiration.

3.6.1.1 Mémoire à court terme

Pendant le processus de recherche, les déplacements vers des solutions qui ont déjà été visités sont stockés dans une liste appelée tabou liste. Généralement, la longueur de cette liste, appelée tabou ‘‘tenure’’, détermine le type de mémoire utilisée. La mémoire à court terme signifie qu'une quantité limitée de déplacements tabou peut être stockée. Par conséquent, la mémoire est rapidement occupée et le statut tabou de ces déplacements peut facilement être oublié. La mémoire à court terme est couramment utilisée pour orienter le processus de recherche vers les régions locales. Cette stratégie est connue sous le nom d'intensification.

3.6.1.2. Mémoire à long terme

Contrairement à la mémoire à court terme, qui vise à intensifier la recherche des régions locales, la mémoire à long terme vise à diversifier la recherche vers des régions non découvertes. Une façon de favoriser la diversification est de conserver le statut tabou des déplacements pénalisés pour une période beaucoup plus longue. L'augmentation de la

longueur de la mémoire aide à guider le processus de recherche vers les régions non explorées.

3.6.1.3. Critère d'aspiration

Puisque la recherche tabou est basé sur la définition des mouvements récents comme des mouvements interdits, certains de ces mouvements sont autorisés lorsque ils améliorent la valeur de la fonction objective. Cette autorisation est connue sous le nom du critère d'aspiration.

3.6.2. Planification de trajectoire à l'aide de la recherche tabou

Masehian [Masehian 06] utilise l'approche de la recherche tabou pour concevoir un planificateur de mouvement en ligne pour un robot mobile. La planification en ligne est basée sur la perception du robot de son environnement à travers les capteurs situés sur son périmètre. Ces capteurs jouent un rôle important dans la recherche tabou, car ils offrent les informations nécessaires sur l'environnement. L'algorithme de recherche tabou utilise ces informations pour éviter de revenir à la zone déjà visitée en mettant à jour deux types de listes tabou. Il s'agit de la mémoire à court terme qui inclut l'enveloppe des directions interdites, et la mémoire à long terme qui contient les sommets déjà examinés et les points trompeurs. De plus, l'algorithme de recherche tabou favorise les déplacements, aux sommets des obstacles, à faible coût. La connexion de ces sommets permet de générer la trajectoire optimale du robot. Cependant, lorsque le robot est piégé dans une impasse, la diversification est activée et l'ensemble du processus est initié. Enfin, le traitement des différentes situations prouve l'efficacité de l'algorithme utilisé.

Hussein [Hussein 12] mène une étude comparative entre l'efficacité des métaheuristiques à base individu et des métaheuristiques à base population pour résoudre le problème de la planification des trajectoires. Les premiers incluent le recuit simulé et la recherche tabou mais les seconds comprennent l'algorithme génétique. Les trois algorithmes sont utilisés pour trouver un chemin optimal dans un environnement de grille qui représente une situation réelle dans un campus. Les chemins dans cet environnement sont composés d'un ensemble de nœuds et représentés par une séquence de valeurs entières. Par conséquent, dans un chemin donné, la diversification est maintenue en remplaçant aléatoirement un sous-chemin, entre deux nœuds choisis, par un autre. Le critère permettant de mesurer ce changement est la distance globale du chemin. Cette

métrique est utilisée comme fonction objective à minimiser par les trois algorithmes. Pour chaque algorithme, en plus de ce critère, la solution est évaluée en fonction d'autres métriques qui sont le temps nécessaire pour parcourus le chemin, le nombre d'itération requis pour le trouver, le temps requis pour chaque itération et le chemin court trouver après cinq itérations. D'une part, la recherche tabou utilise le moindre nombre d'itérations tandis que le recuit simulé emploi le nombre maximum d'itérations. D'autre part, le recuit simulé surpasse l'algorithme génétique et la recherche tabou en termes de temps requis pour trouver le chemin optimal.

Imen [Imen 14] étudie l'efficacité de l'approche de recherche tabou pour le problème de planification global du chemin d'un robot mobile dans un environnement de grille. Basé sur la distance euclidienne, un chemin initial est généré en utilisant une méthode gourmande. Ensuite, pour réduire le temps d'exploration, la recherche tabou est appliquée aux différents sous-chemins composant le chemin initial. Le nouveau chemin est créé en employant différents déplacements tels que l'insertion, la suppression et l'échange de déplacements. Ces déplacements sont équivalents à l'ajout, l'élimination et l'échange de cellules. Par conséquent, pour éviter de revenir au même déplacement pendant une durée spécifiée, deux types de liste tabou sont créés. Ces listes sont ''TabulistOut'' pour les mouvements de suppression et ''TablistIn'' pour les déplacement d'ajout. Malgré l'utilisation des listes tabou, un critère d'aspiration est parfois utilisé pour permettre de revenir à certains déplacements tabou où la fonction objective, distance euclidienne, est améliorée. Afin d'accélérer l'évaluation de la fonction objective, un mécanisme incrémental est adopté. Cependant, lorsqu'il n'y a pas d'amélioration dans le processus de recherche, le mécanisme de diversification est utilisé. Ce mécanisme génère un nouveau chemin et initie un autre processus de recherche tabou. L'algorithme de recherche tabou proposé est testé et comparé à deux autres algorithmes différents dans un environnement de grille de différentes tailles et complexités. La conclusion est que la recherche tabou fonctionne mieux dans le processus de post-optimisation. Cela signifie que la recherche tabou peut être utilisée pour améliorer les solutions d'autres métaheuristiques.

Panda [Panda 16] propose la combinaison entre différentes métaheuristiques pour résoudre le problème de la planification des trajectoires. L'approche hybride combine l'algorithme des essaims particulaires standard et le même algorithme de recherche tabou utilisé dans [Imen 14]. Initialement, cette approche commence par l'algorithme des

essaims particulaires pour rechercher des chemins possibles dans un modèle d'environnement de grille. Puis, à la fin de chaque génération, la recherche tabou est utilisée pour améliorer ces chemins. Ce cycle est répété jusqu'à ce que le nombre maximum d'itérations est atteint. L'efficacité de cette approche est testée et comparée à l'algorithme des essaims particulaires et l'algorithme de recherche tabou utilisant plusieurs robots dans un environnement statique. Les résultats montrent que l'algorithme hybride surpasse les autres algorithmes en termes de convergence et de qualité des solutions.

3.7. Conclusion

Ce chapitre présente quelques métaheuristiques selon leur classe. Nous avons commencé par les métaheuristiques à base population qui sont les algorithmes génétique, les essaimes particulaires et les colonies de fourmis. Nous avons décrit les mécanismes d'évolution, d'une génération à une autre, employés par ces méthodes. Puis, nous avons présenté quelques travaux qui ont utilisé ces métaheuristiques pour la planification des trajectoires. Ensuite, nous avons abordé les métaheuristiques à base individu qui sont le recuit simulé et la recherche tabou. Nous avons donné les principes de ces deux méthodes et quelques recherches concernant leurs utilisations pour la planification des trajectoires.

Chapitre 4

Applications et Résultats

4.1. Introduction

Dans ce chapitre, nous présentons d'abord les outils par lesquels le chemin d'un robot manipulateur est décrit. Cela concerne la courbe NURBS, sa formulation et comment obtenir sa dérivée. Ensuite, nous décrivons comment le problème de planification de trajectoire est formulé et converti en un problème d'optimisation. Dans les parties suivantes, nous donnons une description complète de trois approches qui sont proposées pour résoudre le problème de planification des trajectoires. Enfin, nous montrons les résultats de la mise en œuvre de ces approches pour générer des trajectoires lisses pour un robot manipulateur dans trois situations différentes.

4.2. Non-Uniform, Rational, B-Splines (NURBS)

Les NURBS sont des fonctions polynomiales par morceaux utilisées pour définir des courbes de formes différentes telles que des lignes, des arcs, des cercles, des paraboles, des hyperboles, et des ellipses. La caractéristique la plus importante des NURBS, qui sont des généralisations de B-splines, est le contrôle local de la courbe [Biagotti 08].

La courbe NURBS peut être obtenue en utilisant l'expression suivante :

$$n(u) = \frac{\sum_{j=0}^{m} p_j w_j B_j^p(u)}{\sum_{j=0}^{m} w_j B_j^p(u)}, \quad u_{\min} \le u \le u_{\max} \tag{4.1}$$

Ce qui est équivalent à :

$$n(u) = \sum_{j=0}^{m} p_j N_j^p(u), \quad u_{\min} \le u \le u_{\max} \tag{4.2}$$

avec

$$N_j^p(u) = \frac{w_j B_j^p(u)}{\sum_{j=0}^{m} w_j B_j^p(u)}, \quad u_{\min} \le u \le u_{\max} \tag{4.3}$$

$p_j, j = 0,...,m$ sont les points de contrôle, w_j sont les poids propres, $B_j^p(u)$ sont les fonctions B-spline de base de degré p défini, en termes de vecteur de nœud non-uniforme u, comme suite:

$$B_j^0(u) = \begin{cases} 1, & \text{if } u_j \le u \le u_{j+1} \\ 0, & \text{otherwise} \end{cases} \tag{4.4}$$

$$B_j^p(u) = \frac{u - u_j}{u_{j+p} - u_j} B_j^{p-1}(u) + \frac{u_{j+p+1} - u}{u_{j+p+1} - u_{j+1}} B_{j+1}^{p-1}(u), \quad p > 0. \tag{4.5}$$

avec u est défini par :

$$u = [\underbrace{u_{\min},...,u_{\min}}_{p+1}, u_{p+1},...,u_{n_{knot}-p-1}, \underbrace{u_{\max},...,u_{\max}}_{p+1}] \tag{4.6}$$

4.3. Dérivé d'une courbe NURBS

Les dérivées de la courbe NURBS jouent un rôle important dans la détermination de la forme finale de la courbe. Généralement, les dérivés sont nécessaires pour imposer certaines conditions telles que les conditions aux limites sur la tangente et la courbure.

Puisque la courbe NURBS est définie en terme des fonctions de base B-splines, ses dérivées sont également exprimées avec ces fonctions. La première dérivée de la courbe NURBS est donnée par:

$$n(u)^{(1)} = \left(\frac{\sum_{j=0}^{m} p_j w_j B_j^p(u)}{\sum_{j=0}^{m} w_j B_j^p(u)} \right)^{(1)} \tag{4.7}$$

En utilisant la règle de division , la dérivée peut être écrite comme suit:

$$n(u)^{(1)} = \frac{\sum_{j=0}^{m} p_j w_j B_j^p(u)^{(1)} \sum_{j=0}^{m} w_j B_j^p(u) - \sum_{j=0}^{m} p_j w_j B_j^p(u) \sum_{j=0}^{m} w_j B_j^p(u)^{(1)}}{\left(\sum_{j=0}^{m} w_j B_j^p(u) \right)^2} \tag{4.8}$$

Où la première dérivée de la fonction de base est donnée par

$$B_j^{p(1)}(u) = \frac{p}{u_{j+p} - u_j} B_j^{p-1}(u) - \frac{p}{u_{j+p+1} - u_{j+1}} B_{j+1}^{p-1}(u) \tag{4.9}$$

Alors que, la k^{ème} dérivée de la fonction de base est, généralement, donnée par:

$$B_j^{p(k)}(u) = \frac{p!}{(p-k)!} \sum_{i=0}^{k} c_{k,i} B_{j+i}^{p-k}(u) \tag{4.10}$$

Avec

$$\begin{aligned} c_{0,0} &= 1 \\ c_{k,0} &= \frac{c_{k-1,0}}{u_{j+p-k+1} - u_j} \\ c_{k,i} &= \frac{c_{k-1,i} - c_{k-1,i-1}}{u_{j+p+i-k+1} - u_{j+i}} \quad i = 1,...,k-1 \\ c_{k,k} &= \frac{-c_{k-1,k-1}}{u_{j+p+1} - u_{j+k}} \end{aligned} \tag{4.11}$$

4.4. Problème de planification des trajectoires

4.4.1. Planification des trajectoires par NURBS

La planification de trajectoires est basée sur l'utilisation des NURBS pour interpoler un ensemble de points donnés. Les NURBS sont choisis pour générer le chemin car elles offrent un haut degré de continuité. De plus, cette représentation assure plus de souplesse et donne plus de flexibilité au chemin. En raison du contrôle local qu'elles présentent, la forme du chemin peut être changée d'une forme convexe à une forme concave et vice versa. Le contrôle local est très important, car il signifie que la modification des points de contrôle permet d'empêcher toute interférence du chemin avec les obstacles.

Comme la technique, basée sur B-spline, employée dans [Biagotti 08], l'approche NURBS consiste à définir le problème d'interpolation d'un ensemble des points par un système linéaire représenté par :

$$q_k^T = \left[N_0^p(\overline{u}_k), N_1^p(\overline{u}_k), ..., N_m^p(\overline{u}_k)\right] \begin{bmatrix} p_0^T \\ p_1^T \\ \vdots \\ p_m^T \end{bmatrix} \qquad k = 0, ..., n \tag{4.12}$$

Où q_k sont les points de passage. p_j sont les points de contrôle qui assure que q_k sont interpolées.

La contrainte supplémentaire pour affecter la direction à chaque point de passage est donnée par :

$$t_k^T = \left[N_0^{p(1)}(\overline{u}_k), N_1^{p(1)}(\overline{u}_k), ..., N_m^{p(1)}(\overline{u}_k)\right] \begin{bmatrix} p_0^T \\ p_1^T \\ \vdots \\ p_m^T \end{bmatrix} \qquad k = 0, ..., n \tag{4.13}$$

Où t_k sont les directions assignées à chaque point de passage.

De plus, l'utilisation d'un NURBS de degré quatre avec des nœuds spécifiques implique l'insertion de deux contraintes supplémentaires pour obtenir un système carré et donc une solution unique [Biagotti 08]. En conséquence, les courbures au point de démarrage et d'arrêt sont choisis pour être ajoutés. Elles sont défini comme suit :

$$n_k^T = \left[N_0^{p(2)}(\overline{u}_k), N_1^{p(2)}(\overline{u}_k), ..., N_m^{p(2)}(\overline{u}_k)\right] \begin{bmatrix} p_0^T \\ p_1^T \\ \vdots \\ p_m^T \end{bmatrix} \qquad k = 0, n \tag{4.14}$$

Où n_k sont les courbures à chacun des points extrêmes.

Alors, le système résultant est défini comme :

$$N\ P = R \tag{4.15}$$

Avec : $P = [p_0, p_1, ..., p_{m-1}, p_m]$

$$N = \begin{bmatrix} N_0^p(\overline{u}_0) & N_1^p(\overline{u}_0) & ... & N_m^p(\overline{u}_0) \\ N_0^{p^{(1)}}(\overline{u}_0) & N_1^{p^{(1)}}(\overline{u}_0) & ... & N_m^{p^{(1)}}(\overline{u}_0) \\ N_0^{p^{(2)}}(\overline{u}_0) & N_1^{p^{(2)}}(\overline{u}_0) & \cdots & N_m^{p^{(2)}}(\overline{u}_0) \\ N_0^p(\overline{u}_1) & N_1^p(\overline{u}_1) & \cdots & N_m^p(\overline{u}_1) \\ N_0^{p^{(1)}}(\overline{u}_1) & N_1^{p^{(1)}}(\overline{u}_1) & \cdots & N_m^{p^{(1)}}(\overline{u}_1) \\ N_0^p(\overline{u}_2) & N_1^p(\overline{u}_2) & \cdots & N_m^p(\overline{u}_2) \\ \vdots & \vdots & & \vdots \\ N_0^p(\overline{u}_{n-1}) & N_1^p(\overline{u}_{n-1}) & \cdots & N_m^p(\overline{u}_{n-1}) \\ N_0^{p^{(1)}}(\overline{u}_{n-1}) & N_1^{p^{(1)}}(\overline{u}_{n-1}) & \cdots & N_m^{p^{(1)}}(\overline{u}_{n-1}) \\ N_0^{p^{(2)}}(\overline{u}_n) & N_1^{p^{(2)}}(\overline{u}_n) & \cdots & N_m^{p^{(2)}}(\overline{u}_n) \\ N_0^{p^{(1)}}(\overline{u}_n) & N_1^{p^{(1)}}(\overline{u}_n) & \cdots & N_m^{p^{(1)}}(\overline{u}_n) \\ N_0^p(\overline{u}_n) & N_1^p(\overline{u}_n) & \cdots & N_m^p(\overline{u}_n) \end{bmatrix}, R = \begin{bmatrix} q_0^T \\ t_0^T \\ n_0^T \\ q_1^T \\ t_1^T \\ q_2^T \\ \vdots \\ q_{n-1}^T \\ t_{n-1}^T \\ n_n^T \\ t_n^T \\ q_n^T \end{bmatrix} \tag{4.16}$$

4.4.2. Fonction coût

Afin de résoudre le problème de planification de trajectoires, en utilisant l'approche globale, différents critères sont considérés. Tout d'abord, on pense que le temps total de déplacement est un facteur important dans l'amélioration de la vitesse du robot. Par conséquent, cette métrique est considérée comme l'un des critères de base dans la fonction coût. Deuxièmement, les deux critères, la distance de déplacement articulaire et la distance de déplacement cartésienne sont utilisées pour éviter les manœuvres inutiles. Ces derniers sont considérés comme des critères fondamentaux pour l'optimisation d'énergie.

En plus des critères de temps et d'énergie, la fonction coût comprend également des stratégies d'évitement des singularités et d'obstacles. La première est introduite pour éviter toute configuration inaccessible, tandis que la deuxième est utilisée pour empêcher le robot de heurter tout obstacle dans son environnement.

Le but est de résoudre le problème énoncé par l'équation (4.15), tout en minimisant la fonction coût définie par:

$$f = f_t + f_q + f_{dis} + f_{\text{sing}} + f_{obs} \tag{4.17}$$

Où

f_t décrit le temps d'exécution ;

f_q représente la fonction de la distance articulaire totale parcourue, en radian ;

f_{dis} signifie la fonction de la distance de déplacement cartésienne totale, en mètres ;

f_{sing} désigne la fonction liée à la stratégie d'évitement des singularités ;

f_{ob} fait référence à la fonction liée à la stratégie d'évitement d'obstacles ;

4.4.2.1. Critère de temps d'exécution

Le temps d'exécution est défini comme le temps nécessaire pour déplacer le robot de la configuration initiale jusqu'à atteindre la cible. La fonction décrivant ce critère est donnée par :

$$f_t = t \tag{4.18}$$

Où t est le temps de déplacement en seconde.

4.4.2.2. Critère de la distance articulaire parcourue

Le mouvement des articulations nécessaire pour que le robot atteint la cible est défini comme la distance articulaire totale de déplacement. Cette métrique est décrite comme suit :

$$f_q = \sum_{i=1}^{n} \sum_{j=2}^{m} \left| q_{i,j} - q_{i,j-1} \right| \tag{4.19}$$

4.4.2.3. Critère de la distance de déplacement cartésien

Contrairement à la distance de déplacement articulaire, où le but est de calculer le mouvement articulaire nécessaire pour réaliser une tâche, le critère de la distance de déplacement cartésienne a pour but le calcul de la distance parcourue par l'organe terminal du robot. Cette distance est donnée par :

$$f_{dis} = \sum_{i=1}^{n} \sqrt{x_i^2 + y_i^2 + z_i^2} \tag{4.20}$$

4.4.2.4. Stratégie d'évitement des singularités

Puisque la trajectoire est générée dans l'espace cartésien, il est inévitable d'utiliser le modèle cinématique inverse pour déterminer les positions articulaires nécessaires pour le mouvement. Pour cette raison, une méthode basée sur le pseudo inverse de la matrice

jacobéenne est utilisée pour trouver la solution pour chaque configuration de l'organe terminal. D'une part, lorsque la méthode trouve une solution dans un nombre donné d'itérations, la configuration est considérée comme non singulière. Dans cette situation, la fonction d'évitement des singularités n'a aucun effet sur la fonction coût. D'autre part, lorsque la méthode ne converge pas pendant le nombre donné d'itérations, la configuration est considérée comme singulière. Par conséquent, la valeur de la fonction d'évitement des singularités est ajoutée comme fonction de pénalité dans la fonction de coût.

Toute la procédure pour la stratégie d'évitement des singularités est donnée comme suit :

Tant que (la variation articulaire est trop petite)

1. Calculez la cinématique inverse.
    a. Calculer la variation articulaire en utilisant le pseudo inverse du jacobéenne.
    b. Vérifiez si la variation articulaire est trop petite.
    c. Mettre à jour la position articulaire en ajoutant la variation articulaire.
    d. Fonction d'évitement des singularités f_s=0.
2. ***Si*** (le nombre d'itérations max est dépassé)
    a. Affecter une valeur très grande à la fonction d'évitement des singularités.
    b. Arrêter.

Fin Si

3. Incrémenter le nombre d'itérations

Fin tant que

4.4.2.5. Stratégie d'évitement d'obstacles

La stratégie utilisée pour éviter les obstacles est basée sur la pénalisation de toute configuration qui entraîne des interférences entre le robot manipulateur et les obstacles. Par conséquent, la collision du robot avec les obstacles est vérifiée. Lorsqu'un mouvement sans collision est détecté, la fonction d'évitement des obstacles n'a aucun effet sur la fonction coût ; en cas de collision, le nombre de tous les points d'interférence est calculé et pondéré. Cela nous permet d'ajouter la somme comme une valeur de pénalité dans la fonction coût.

Dans cette stratégie, la fonction d'évitement d'obstacle est défini par :

$$\begin{cases} f_{ob} = 0 & if \quad (R_t \notin R_{ob} \wedge n(u) \notin R_{ob}) \\ f_{ob} \neq 0 & otherwise \end{cases} \tag{4.21}$$

Avec

R_t est l'espace occupé par le robot pendant son mouvement.

$n(u)$ est la trajectoire générée en utilisant la courbe NURBS.

R_{ob} est l'espace occupé par les obstacles définis comme des cylindres :

$$R_{ob} = \sqrt{(x - x_0)^2 + (y - y_0)^2} \tag{4.22}$$

Puisque le but principal est de produire un mouvement sans collision et sans singularités, le temps de déplacement, la distance de déplacement articulaire et la distance de déplacement cartésienne ne sont pas pondérés, alors que la fonction d'évitement des singularités et la fonction d'évitement d'obstacle sont pondérées. Par conséquent, le problème de planification de trajectoire peut être traité comme un problème de minimisation, défini par l'équation (4.17), ou comme un problème de maximisation, défini par une fonction d'adaptation donnée par :

$$adapt(f) = \frac{1}{1+f} \tag{4.23}$$

4.5. Optimisation des trajectoires

4.5.1. Approche NURBS-algorithmes génétiques

Dans cette approche, les algorithmes génétiques sont utilisés pour résoudre le problème de planification des trajectoires représentés par les courbes NURBS. Ces algorithmes commencent par générer aléatoirement une population initiale. Cette population est constituée d'un ensemble d'individus représentant les poids des NURBS. Pour former les chromosomes, les individus de la population sont décrits par un code binaire, où chaque gène est représenté par un bit distinct. Tous les chromosomes de la population initiale suivent un processus d'évolution itératif pour trouver les solutions optimales. Ce processus est représenté par trois procédures génétiques : sélection des parents, croisement et mutation.

4.5.1.1. Sélection des parents

La sélection des parents est considérée comme l'étape fondamentale pour former la nouvelle génération. Par conséquent, il est décidé d'utiliser la méthode de sélection par tournois pour choisir les parents de la génération suivante. La première étape de cette méthode consiste à choisir deux individus au hasard. Ensuite, l'étape suivante implique d'effectuer un tournoi entre les individus choisis. Le plus adapté gagne la compétition et est donc choisi comme premier parent. Ce processus est répété jusqu'à ce que la taille de la population soit atteinte.

4.5.1.2. Croisement

Le but principal de la réalisation de croisement est de créer de nouveaux individus et donc de faire une exploration intensive de l'espace de recherche. L'opération de croisement est réalisée en échangeant des parties des chromosomes des parents choisis. Cette opération produit un pair d'enfants dont les caractéristiques sont dérivées de leurs parents. Ensuite, les quatre individus, les parents et leurs enfants sont mis dans une compétition de tournoi où le pair le plus adapté est sélectionné pour être dans la prochaine génération.

L'aspect important de l'opération de croisement est la façon dont il est exécuté. Au début, deux points sont générés de manière aléatoire. Ces points sont les lieus du croisement. Puisque les chromosomes des parents sont codés en binaire, tous les bits, entre les deux points, du premier parent sont transformés au deuxième et vice versa. Cette méthode est connue sous le nom de croisement à deux points.

4.5.1.3. Mutation

Bien que la probabilité de mutation soit très faible, cette opération joue un rôle important en empêchant le processus d'évolution de prendre une seule direction. Grâce au changement aléatoire des gènes dans les chromosomes, de nouveaux individus sont créés et, par conséquent, plus d'espace est exploré.

Puisque l'interaction entre ces chromosomes a lieu à la phase de croisement, il est important de noter que la mutation est également effectuée au cours de cette phase. Cela signifie qu'il peut arriver que, pendant l'opération de croisement, certains gènes dans les chromosomes soient inversés.

Algorithme NURBS- Algorithmes génétiques

L'approche de planification des trajectoires NURBS-Algorithmes génétiques proposée est résumée dans la procédure suivante:

1. Générer aléatoirement la population initiale.
2. Initialisez le numéro de génération.

Tant que (le nombre maximum de générations n'est pas atteint)

3. Calculez la fonction coût pour chaque individu dans la population.
    a. Générer le chemin NURBS.
    b. Vérifiez les singularités possibles.
    c. Vérifiez les collisions possibles avec l'obstacle.
    d. Calculer la longueur de la trajectoire.
4. Initialiser la taille de la population

Tant que (la taille maximale de la population n'est pas atteinte)

a. Sélectionnez au hasard deux parents.
b. Effectuer une opération de croisement avec probabilité Pc
c. Effectuer une opération de mutation avec probabilité Pm
d. Sélectionnez les deux individus adaptés parmi les parents et leurs enfants.
e. Ajouter les deux individus dans la nouvelle population
f. Incrémenter la taille de la population

Fin tant que

5. Incrémenter le nombre de génération

Fin tant que

Les étapes de cet algorithme peuvent être décrites par l'organigramme suivant :

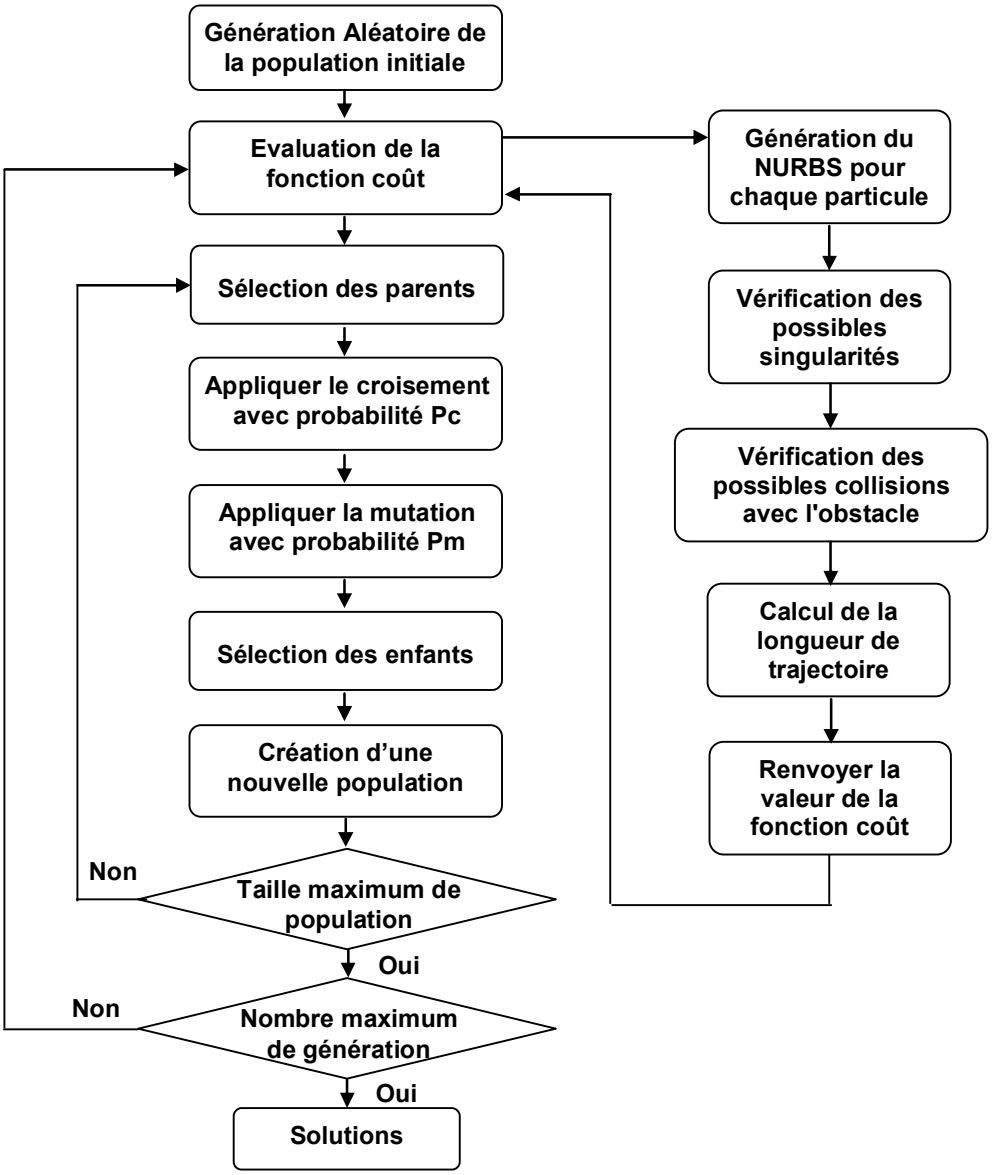


Figure 4.1 Organigramme de l'approche NURBS- Algorithmes Génétiques

4.5.2. Approche NURBS-essaims particulaires

Dans cette approche, une autre méthode à base population, les essaims particulaires standard, est utilisé pour résoudre le problème de planification des trajectoires représentés par les courbes NURBS. La phase initiale consiste à générer aléatoirement les particules, dans l'espace de recherche, selon une distribution uniforme. De même, la vitesse de chaque particule est produite au hasard également selon une distribution uniforme. A chaque itération, chaque particule dans l'espace de recherche est définie par son position $x_i(t)$, sa vitesse $v_i(t)$, la meilleure position visitée, donnée par un vecteur $p_i(t)$, et la meilleure position trouvée par les informateurs de la particule représentée par un vecteur $g_i(t)$. Par conséquent, la position de la particule et sa vitesse sont mises à jour en utilisant les équations suivantes:

$$\begin{cases} v_i(t) = v_i(t-1) + c_1\varphi_1(p_i - x_i) + c_2\varphi_2(g_i - x_i) \\ x_i(t) = x_i(t-1) + v_i(t) \end{cases} \tag{4.24}$$

Où c_1 et c_2 sont des constantes positives, φ_1 et φ_2 sont deux variables aléatoires avec une distribution uniforme entre 0 et 1.

Au cours de l'évolution des essaims particulaires, si une coordonnée $x_i(t)$ calculée selon l' équation du mouvement (4.24) est inférieure à $x_{\min}$ ou supérieure $x_{\max}$, la valeur correspondante est respectivement remplacée par $x_{\min}$ ou $x_{\max}$, et la vitesse de la particule est remise à zéro, le mécanisme complet utilisé pour empêcher une particule de quitter l'espace de recherche est décrit par les équations suivantes:

$$x_i \notin [x_{\min}, x_{\max}] \Rightarrow \begin{cases} v_i = 0 \\ x_i < x_{\min} \Rightarrow x_i = x_{\min} \\ x_i > x_{\max} \Rightarrow x_i = x_{\max} \end{cases} \tag{4.25}$$

Algorithme NURBS-essaims particulaires

L'approche de planification des trajectoires NURBS-PSO proposée est résumée en quatre étapes dans la procédure suivante:

1. Générer aléatoirement les particules initiales et leurs vitesses.

Tant que (le nombre maximum de générations n'est pas atteint)

2. Calculez la fonction coût.
   a. Générer le chemin NURBS.
   b. Vérifiez les singularités possibles.
   c. Vérifiez les interférences possibles avec l'obstacle.

d. Calculer la longueur de la trajectoire.

3. Sélectionnez les meilleures particules.

4. Mettre à jour la position et la vitesse de chaque particule.

Fin tant que

Cette procédure est transformée en organigramme représenté sur la Figure 4.2. Les deux blocs clarifient la mise en œuvre de l'algorithme NURPS- essaims particulaires

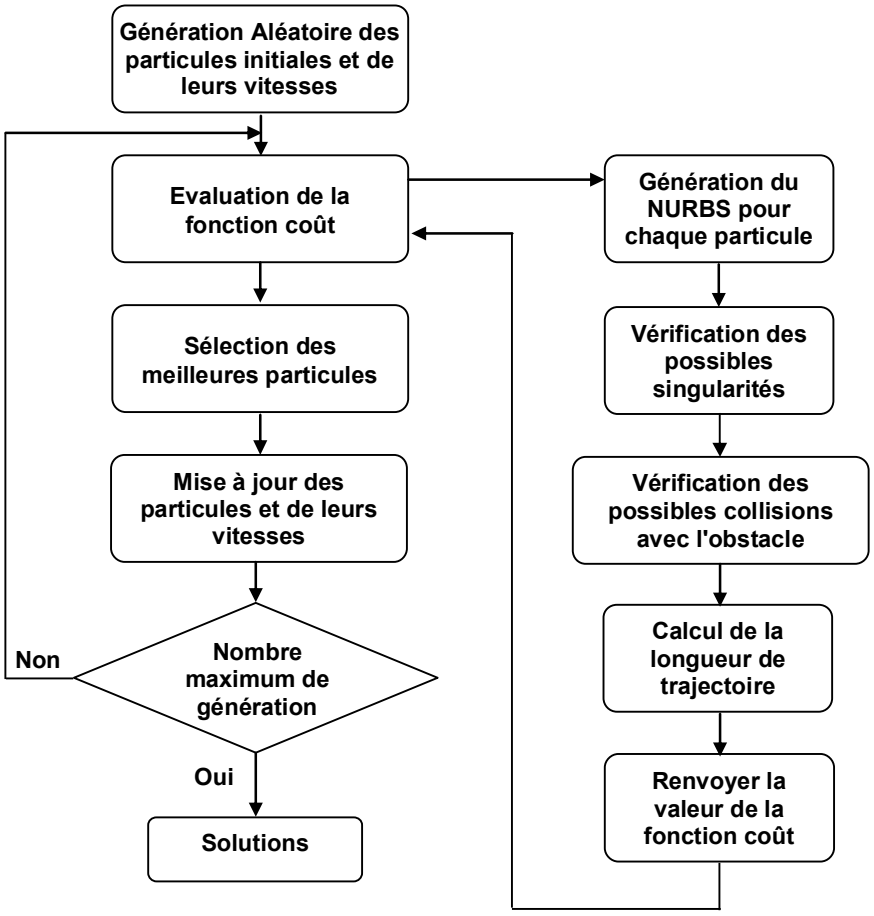


Figure 4.2 Organigramme de l'approche NURBS-Essaims particulaires

4.5.3. Approche NURBS-recuit simulé

Afin de résoudre le problème de planification des trajectoires, les poids de la courbe NURBS sont utilisés comme variable de contrôle pour l'algorithme de recuit simulé. Comme le recuit réel, le but de l'utilisation de cet algorithme est de stabiliser la fonction coût à une valeur minimale en commençant par une haute température, et de l'abaisser progressivement, jusqu'à atteindre la température finale. Premièrement, une solution initiale, un vecteur des poids du NURBS, est générée aléatoirement et sa fonction coût est évaluée. Suite à cette initialisation, un nouveau vecteur est généré aléatoirement et sa

fonction coût est évaluée et itérativement comparée à l'ancienne. Lorsque la valeur de la fonction coût du nouveau vecteur est meilleure que l'ancienne, il est évident que ce vecteur est accepté comme la nouvelle solution. Cependant, une caractéristique intéressante du recuit simulé est que ce vecteur peut être accepté, en tant qu'une nouvelle solution, même lorsque sa valeur de fonction évaluée est mauvaise que l'ancienne, mais cette fois avec une probabilité égale à $\exp(-\Delta E / T)$.

Une étape importante pour l'algorithme de recuit simulé, pour converger vers la solution optimale, est de maintenir la température fixée, durant le processus itératif, pendant un certain temps. Cette procédure est similaire à celle du recuit réel où la température est maintenue inchangée jusqu'à ce que le système se stabilise. Généralement, à cette phase, le processus continue avec la génération d'une nouvelle solution et la comparaison avec l'ancienne. Cependant, lorsque cette phase se termine, le processus itératif se poursuit avec une valeur de température réduite. Enfin, l'algorithme se termine lorsque le critère d'arrêt de la température finale est atteint.

Algorithme de NURBS-recuit simulé

L'approche de NURBS-recuit simulée utilisée pour la planification de la trajectoire, est résumée dans les étapes suivantes:

1. Initialisation de la température.
2. Générer aléatoirement un vecteur des poids NURBS V_i comme solution initiale.
3. Évaluez la solution initiale $E_i = f(V_i)$.

Tant que (la température finale n'est pas atteinte)

Initialiser le Compteur

Tant que (Compteur <Seuil)

a. Générer aléatoirement un nouveau vecteur des poids NURBS Vn

b. Évaluez la solution initiale $E_n = f(V_n)$.

c. ***si*** $(E_n < E_i)$

$V_n = V_i$

sinon si $(\exp(-\Delta E / T))$ > probabilité

$V_n = V_i$

Fin si

Incrément compteur;

Fin tant que

Réduire la température

Fin tant que

Cet algorithme est transformé en organigramme représenté sur la Figure 4.3

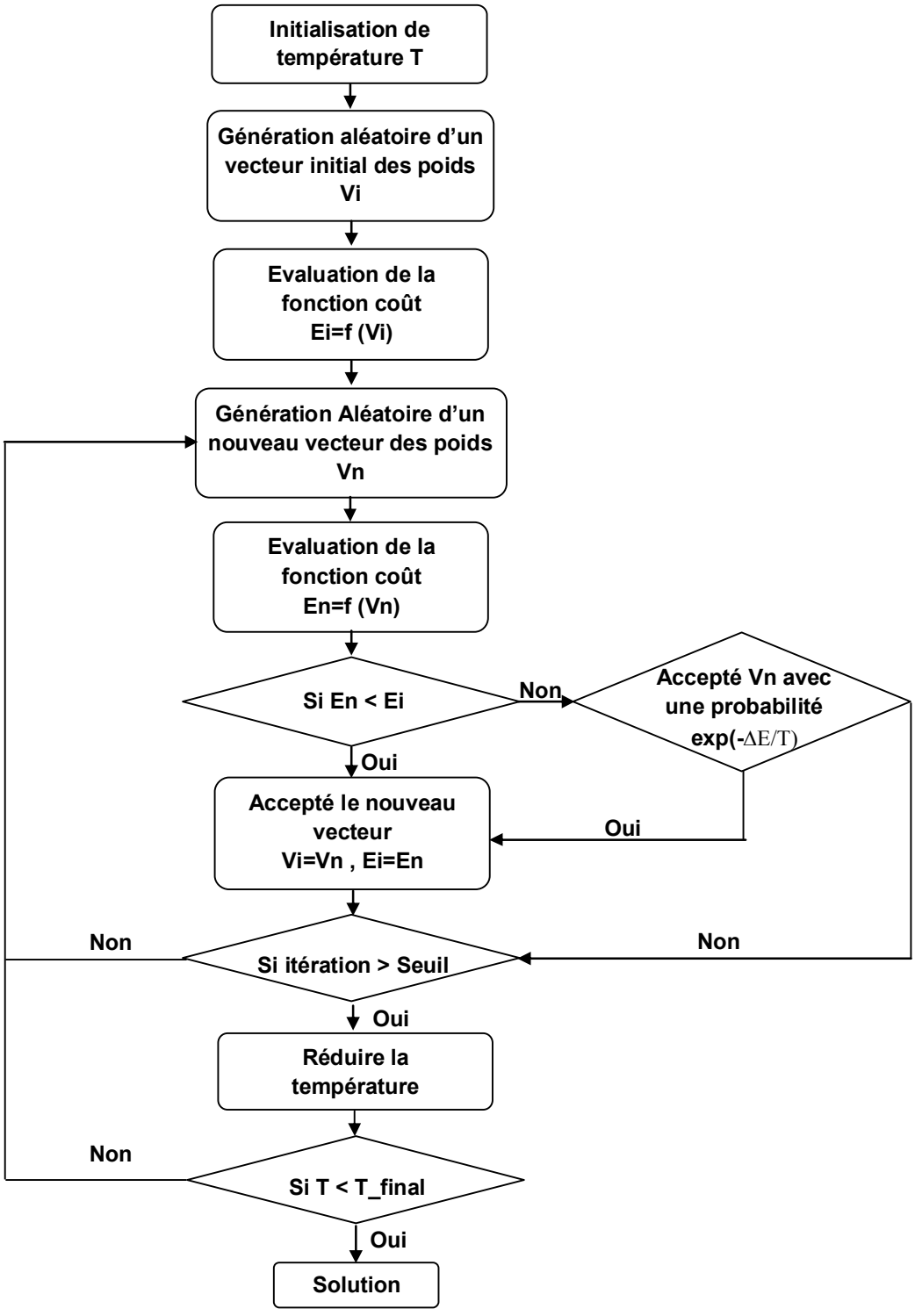


Figure 4.3 Organigramme de l'approche NURBS-Recuit Simulé

4.6. Génération des trajectoires pour un robot manipulateur

Il existe deux approches principales pour la planification des trajectoires pour les robots manipulateurs. Dans la première approche la trajectoire est définie dans l'espace opérationnel. Bien que cette approche assure que la trajectoire de l'effecteur souhaitée, il est nécessaire d'utiliser le modèle cinématique inverse pour calculer les positions articulaires requises. Dans la deuxième approche la trajectoire est exprimée directement dans l'espace articulaire sans avoir besoin du modèle cinématique inverse. Cependant, l'inconvénient principal de cette approche est que la trajectoire exacte de l'effecteur ne peut être prédite.

Afin d'évaluer les performances des approches proposés, nous considérons le robot à six dégrées de liberté, Kuka KR15, montré dans la Figure 4.4, avec le modèle mathématique montré dans le tableau 4.1 [Verdonck 04], où la courbe NURBS est choisie pour décrire le chemin dans l'espace opérationnelle pour permettre à l'effecteur du robot manipulateur de se déplacer selon un chemin précis.

Tableau 4.1. Paramètres Denavit-Hartenberg du robot KUKA KR15

j	$\alpha_{i-1}[rad]$	$a_{i-1}[m]$	$\theta_i[rad]$	$d_i[m]$
1	π	0	0	0
2	$\pi/2$	$0,3$	0	0
3	0	$0,65$	0	0
4	$\pi/2$	$0,155$	0	$-0,6$
5	$-\pi/2$	0	0	0
6	$\pi/2$	0	0	0

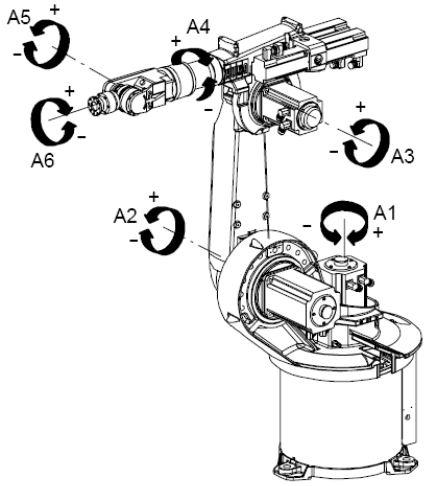


Figure 4.4 Le robot industriel KUKA KR15

Nous commençons par définir un ensemble de points de passage qui délimitent la courbe NURBS, où chaque point est défini dans l'espace. De plus, les obstacles sont considérés comme des objets tridimensionnels avec une largeur et une hauteur définies pour couper les segments NURBS reliant les points de passage.

Dans le but de déplacer l'effecteur du robot d'une situation initiale à une situation finale, différentes représentations peuvent être utilisées pour exprimer l'orientation. Pour éviter le problème des singularités d'orientation, la représentation des quaternions est utilisée [Itzhack 00, Funda 90]. Par conséquent, le vecteur des coordonnées opérationnelles est composé de sept composantes, trois composantes du vecteur de position et quatre composantes des quaternions:

$$X = \begin{bmatrix} P \\ Q \end{bmatrix} = \begin{bmatrix} x & y & z & n & i & j & k \end{bmatrix}^T \tag{4.26}$$

La planification de la trajectoire consiste à déterminer une courbe NURBS dans l'espace de travail, qui interpole les points de passage imposés et qui évite les obstacles en respectant les vitesses assignées à chaque point de passage. La première étape consiste à exprimer la trajectoire dans l'espace opérationnel. L'étape suivante consiste à appliquer le modèle cinématique inverse pour obtenir les articulations nécessaires d'une position et d'une orientation donnée de l'effecteur.

En simulation, nous planifions une trajectoire multipoint pour la position et l'orientation en fonction de la représentation des quaternions et en utilisant une courbe NURBS de degré quatre avec sept points de passage donnés par :

$$\begin{aligned}
q_0 &= [0 \ \tfrac{-\pi}{2} \ 0 \ 0 \ 0 \ 0]; \\
q_1 &= [\tfrac{\pi}{3} \ \tfrac{-\pi}{3} \ \tfrac{-\pi}{12} \ \tfrac{\pi}{12} \ \tfrac{-\pi}{12} \ \tfrac{\pi}{18}]; \\
q_2 &= [\tfrac{\pi}{3} \ \tfrac{-\pi}{4} \ \tfrac{\pi}{3} \ \tfrac{\pi}{3} \ \tfrac{-\pi}{3} \ \tfrac{\pi}{9}]; \\
q_3 &= [\tfrac{5\pi}{36} \ \tfrac{-\pi}{4} \ \tfrac{\pi}{18} \ \tfrac{\pi}{4} \ \tfrac{-\pi}{4} \ \tfrac{\pi}{6}]; \\
q_4 &= [\tfrac{-5\pi}{36} \ \tfrac{-\pi}{4} \ \tfrac{\pi}{18} \ \tfrac{\pi}{6} \ \tfrac{-\pi}{6} \ \tfrac{\pi}{9}]; \\
q_5 &= [\tfrac{-\pi}{3} \ \tfrac{-\pi}{4} \ \tfrac{\pi}{6} \ \tfrac{\pi}{12} \ \tfrac{-\pi}{12} \ \tfrac{\pi}{18}]; \\
q_6 &= [\tfrac{-\pi}{3} \ \tfrac{-\pi}{3} \ \tfrac{-\pi}{12} \ 0 \ 0 \ 0];
\end{aligned} \tag{4.27}$$

4.7. Résultats et discussions

Pour résoudre le problème de planification des trajectoires, trois situations sont étudiées. Premièrement, un environnement libre est considéré. Puis, un environnement avec obstacle est adopté. Enfin, un environnement occupé par deux obstacles est étudié. Pour chaque situation, les approches proposées basées sur les algorithmes génétiques, les essaims particulaires et le recuit simulé sont testés. Ils sont codés dans MATLAB 8.1.0.604 et implémentés sur un ordinateur i5 avec 2.30 GHz et 6 Go de RAM sous Windows 8. La courbe NURBS à quatre degrés avec vingt et un nœuds est choisie pour construire un système de seize équations avec seize points de contrôle à optimiser. En forçant la courbe à respecter les directions imposées, on a assuré le franchissement des points de passage dans l'espace libre et un mouvement sans collision dans l'environnement avec obstacles.

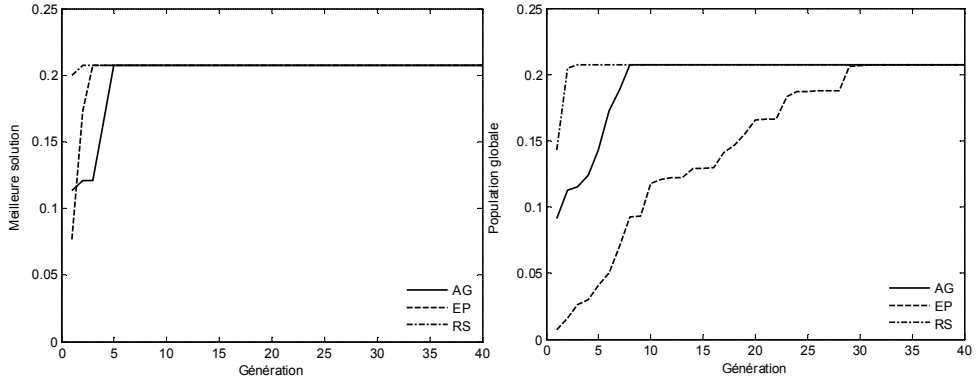
Nous exécutons les approches proposées en utilisant une population de vingt individus et quarante générations pour les algorithmes génétiques et les essaims particulaires, qui est équivalente à huit cents solutions potentielles de recuit simulé. Nous fixons la limite inférieure pour les poids des NURBS à un et une limite supérieure à dix, puis nous exécutons les algorithmes plusieurs fois. Les trois algorithmes sont évalués en termes de temps d'exécution moyen, de vitesse de convergence de la meilleure solution et de la population globale vers la solution optimale. Enfin, nous sélectionnons le vecteur des poids qui donne les points de contrôle les plus appropriés pour chaque algorithme, et par conséquent, le chemin le plus approprié avec le temps de déplacement minimum.

4.7.1. Planification des trajectoires dans un environnement sans obstacle

Le but de la planification des trajectoires dans un environnement sans obstacle est de déplacer l'organe terminal du robot manipulateur de la situation initiale, à travers les points intermédiaires, jusqu'à la situation finale. Dans ce cas, l'évaluation des approches en termes de convergence de la meilleure solution et de convergence de la population globale sont données dans la figure 4.5. Cependant, le temps moyen d'exécution est illustré dans le tableau 4.2. De plus, le mouvement du robot est représenté sur les figures 4.6, 4.9 et 4.12, tandis que l'orientation par quaternions est représentée sur les figures 4.7, 4.10 et 4.13. En fin, les positions articulaires nécessaires sont montrés sur les figures 4.8, 4.11 et 4.14.

Tableau 4.2. Performance des approches dans un environnement sans obstacle

Critère	Algorithme Génétique	Essaim particules	Recuit simulé
temps moyen d'exécution (min)	15	49	16



(a) (b)

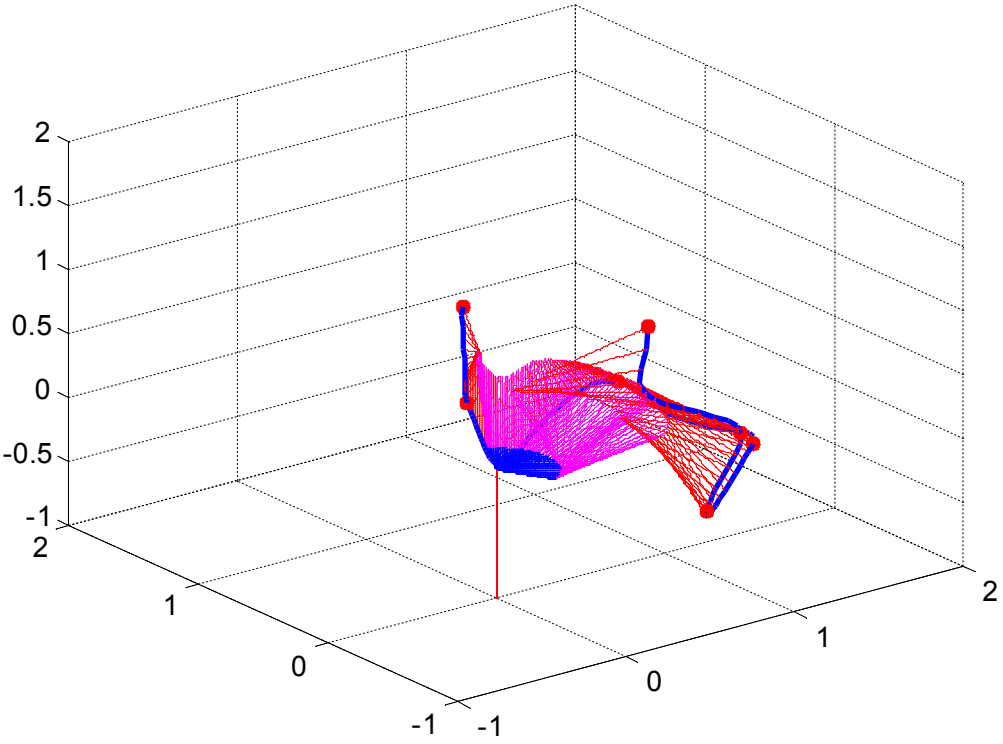
Figure 4.5. Convergence des approches (a: meilleure solution, b : population globale) dans un environnement sans obstacle

À partir du tableau 4.2, on peut voir que les essaims particulaires ont pris plus de temps que les deux autres algorithmes. Cela est causé par la complexité de son mécanisme de recherche. D'autre part, le temps minimale nécessaire à l'algorithme génétique par rapport aux autres approches est dû à sa structure simple et à sa tendance vers des solutions homogènes qui conduisent à moins de nombre d'itérations dans la fonction de test des singularités.

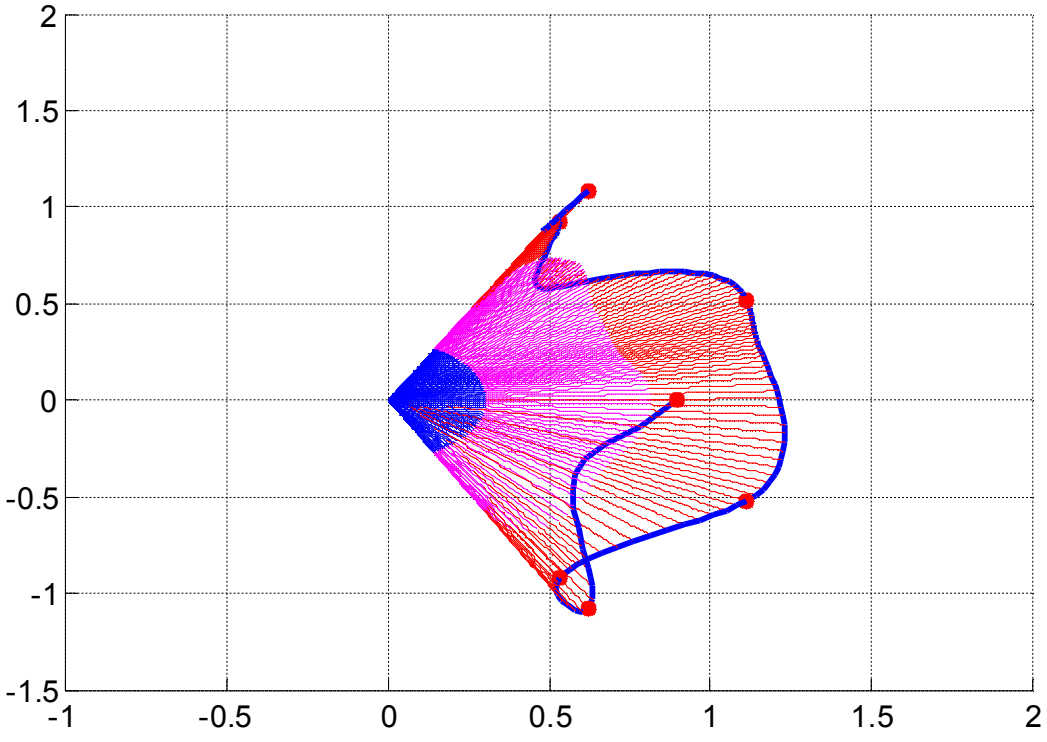
De la figure 4.5 on peut voir que les trois approches convergent vers la même solution optimale. Cependant, la population globale converge d'une façon rapide vers la solution optimale pour l'algorithme génétique et le recuit simulé seulement. Mais pour les essaims particulaires, elle atteint la valeur maximale dans les dernières générations. Ceci indique que les mécanismes utilisés dans l'algorithme génétique et le recuit simulé permettent l'homogénéité des solutions. En revanche, le mécanisme utilisé dans les essaims particulaires permet plus de diversité.

Les figures 4.6, 4.9 et 4.12 montrent les trajectoires générées, pour le robot manipulateur dans un environnement sans obstacle, en utilisant les trois approches. Ces trajectoires sont présentées avec le mouvement du robot, dans un espace à trois dimensions et dans un plan respectivement. On peut observer que tous les points intermédiaires, du point de départ au point final, sont interpolés.

Les figures 4.8, 4.11 et 4.14 illustrent les positions articulaires requises pour chaque trajectoire planifiée. Ces positions articulaires sont obtenues en utilisant le modèle cinématique inverse. On peut remarquer que les trajectoires articulaires obtenues soit pour la position ou pour l'orientation sont très souples. Ces derniers sont confirmés par leurs équivalents quaternions représentés par les 4.7, 4.10 et 4.13 respectivement.



(a)



(b)

Figure 4.6. Trajectoire optimisée par les algorithmes génétiques dans un environnement sans obstacle. (a) vue 3D. (b) vue 2D.

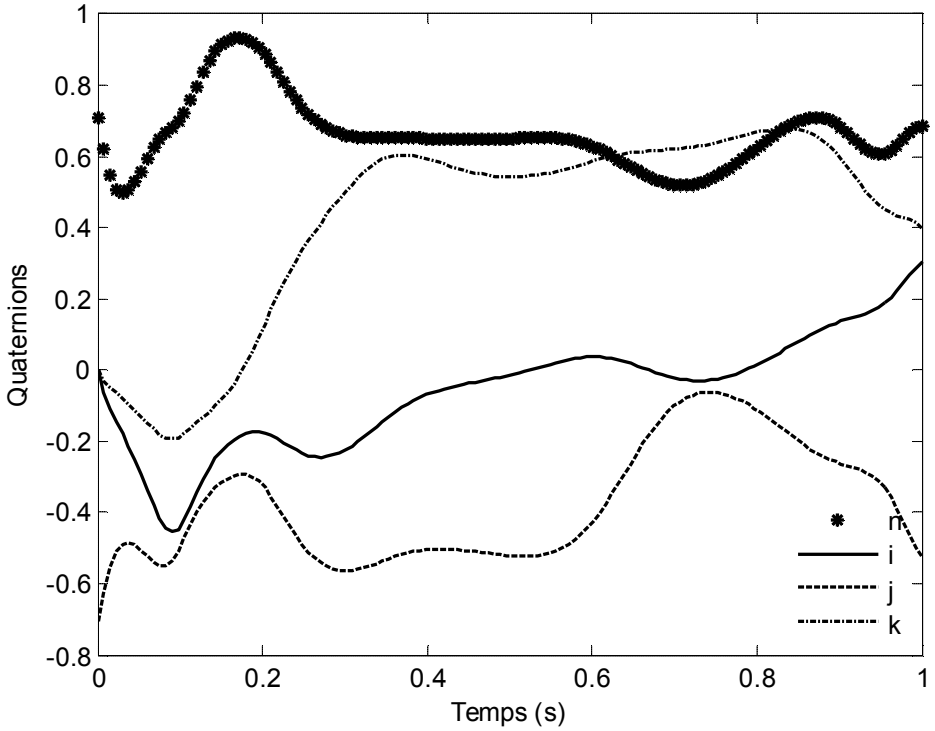


Figure 4.7 Quaternions résultants de l'optimisation par l'approche NURBS-algorithmes génétiques dans un environnement sans obstacle.

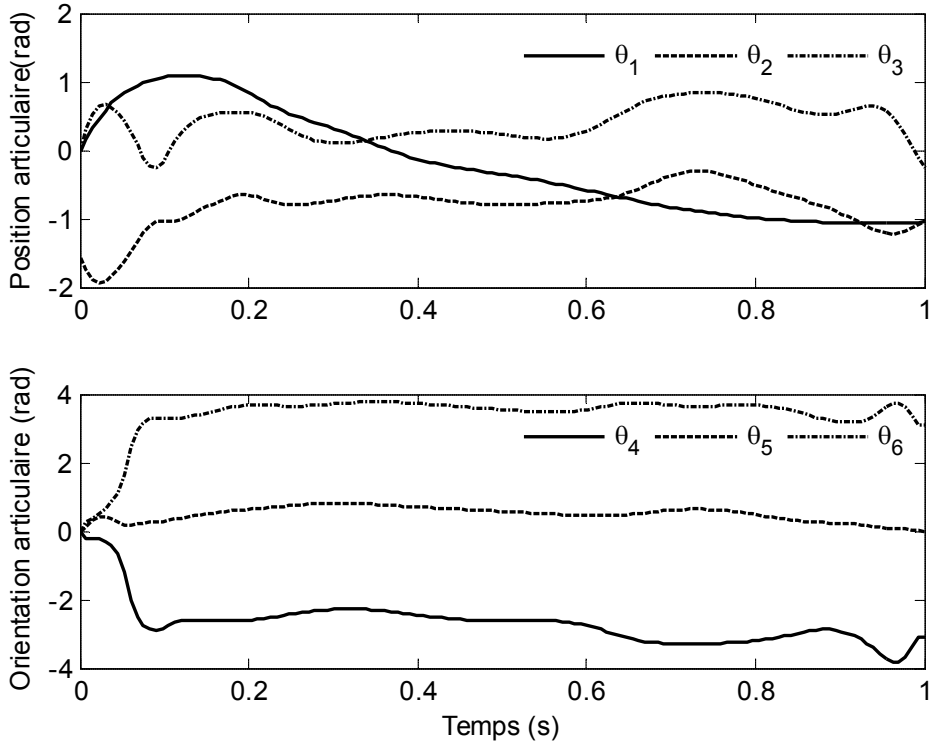
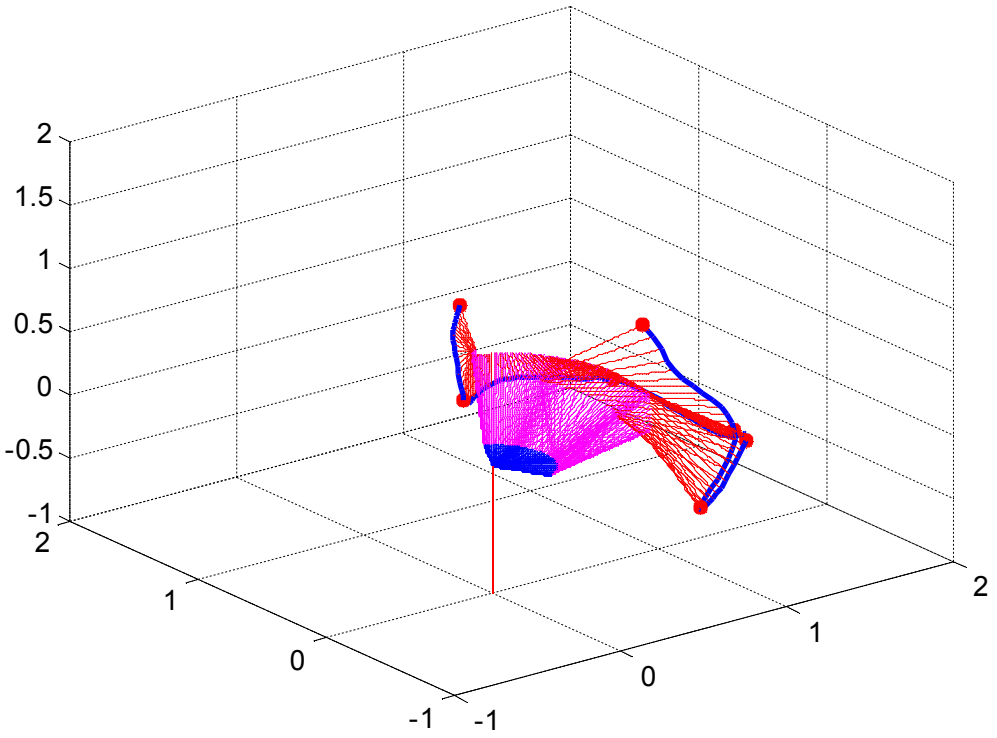
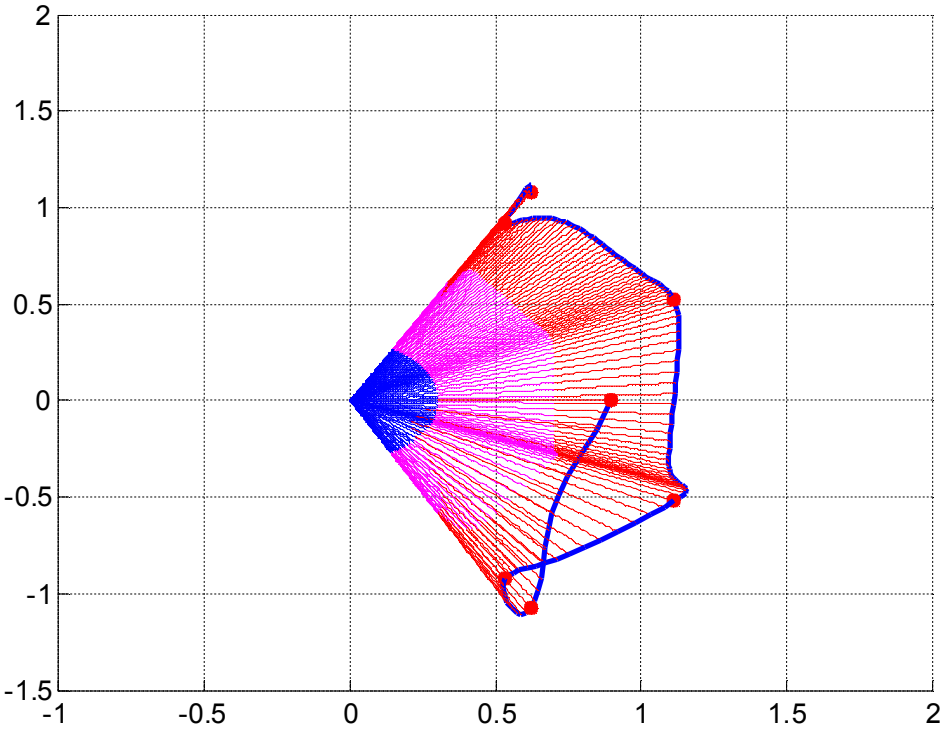


Figure 4.8. Positions articulaires résultantes de l'optimisation par l'approche NURBS-algorithmes génétiques dans un environnement sans obstacle



(a)



(b)

Figure 4.9. Trajectoire optimisée par les essaims particulaires dans un environnement sans obstacle. (a) vue 3D. (b) vue 2D.

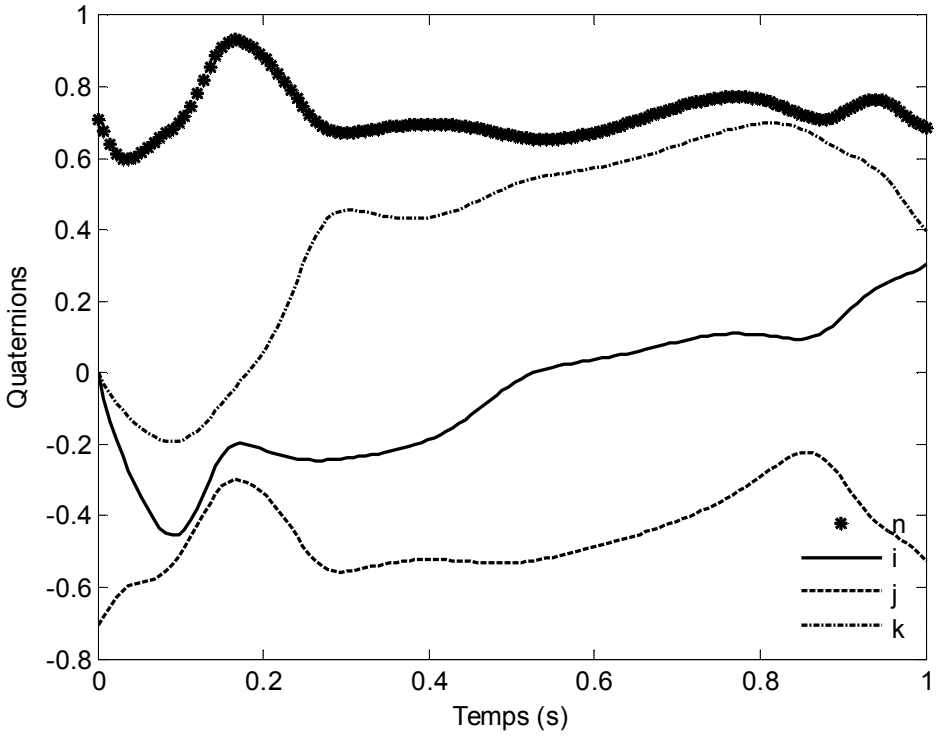


Figure 4.10 Quaternions résultants de l'optimisation par l'approche NURBS-essaims particulaires dans un environnement sans obstacle

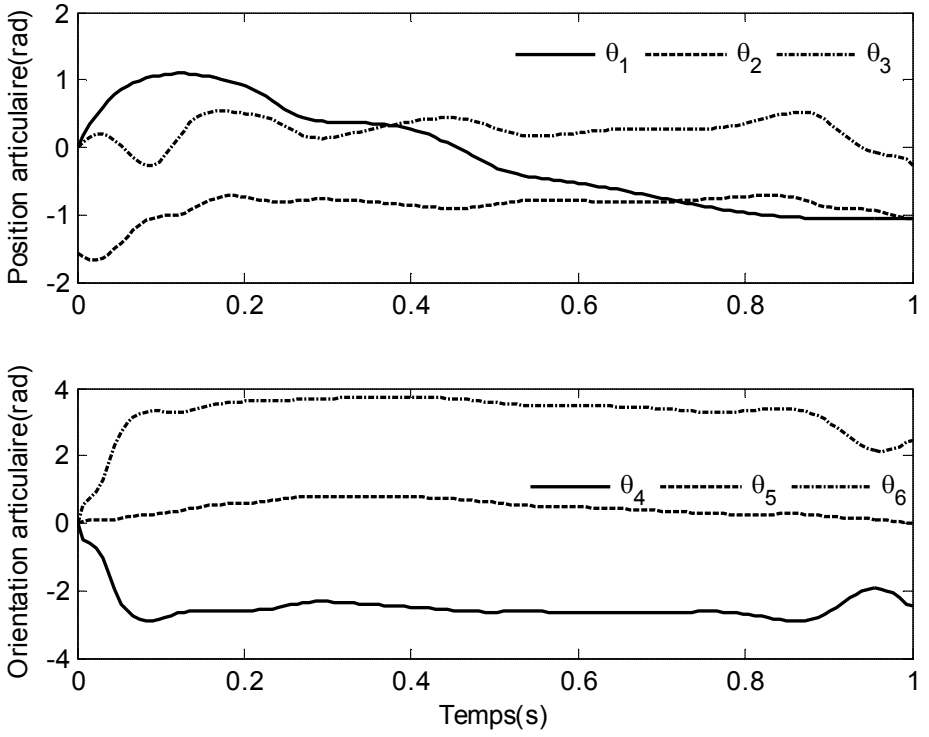
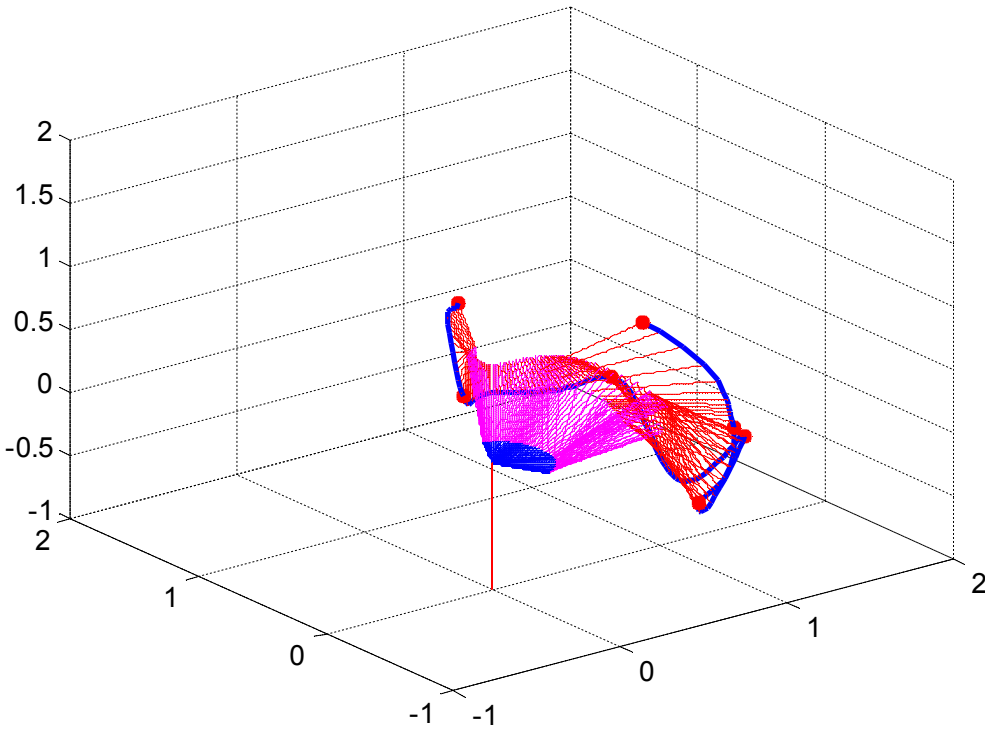
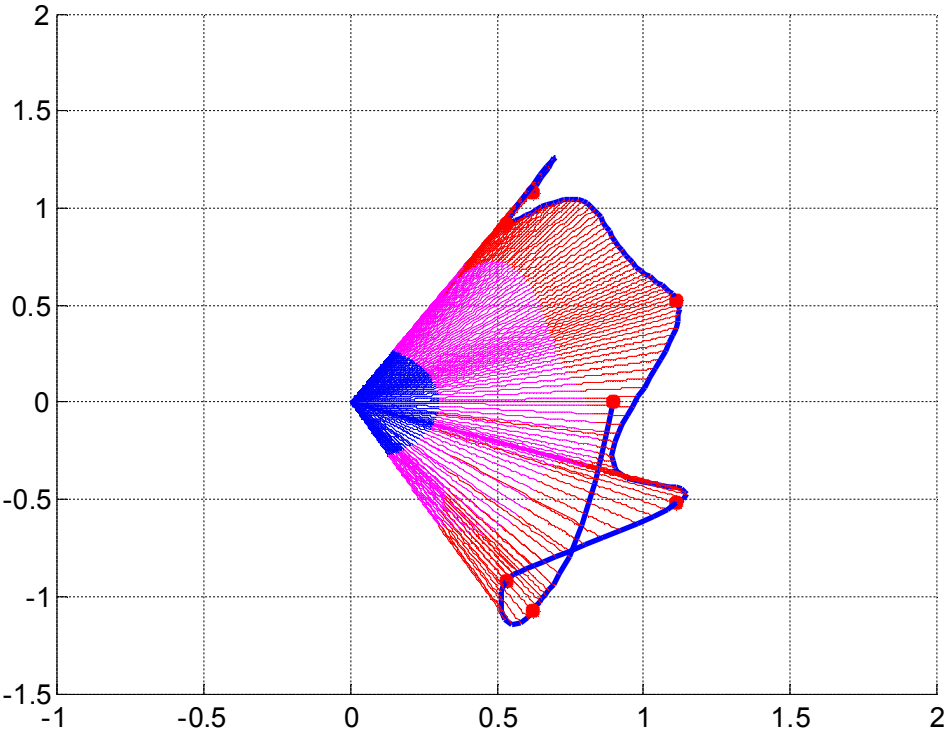


Figure 4.11. Positions articulaires résultantes de l'optimisation par l'approche NURBS-essaims particulaires dans un environnement sans obstacle



(a)



(b)

Figure 4.12. Trajectoire optimisée par recuit simulé dans un environnement sans obstacle. (a) vue 3D. (b) vue 2D.

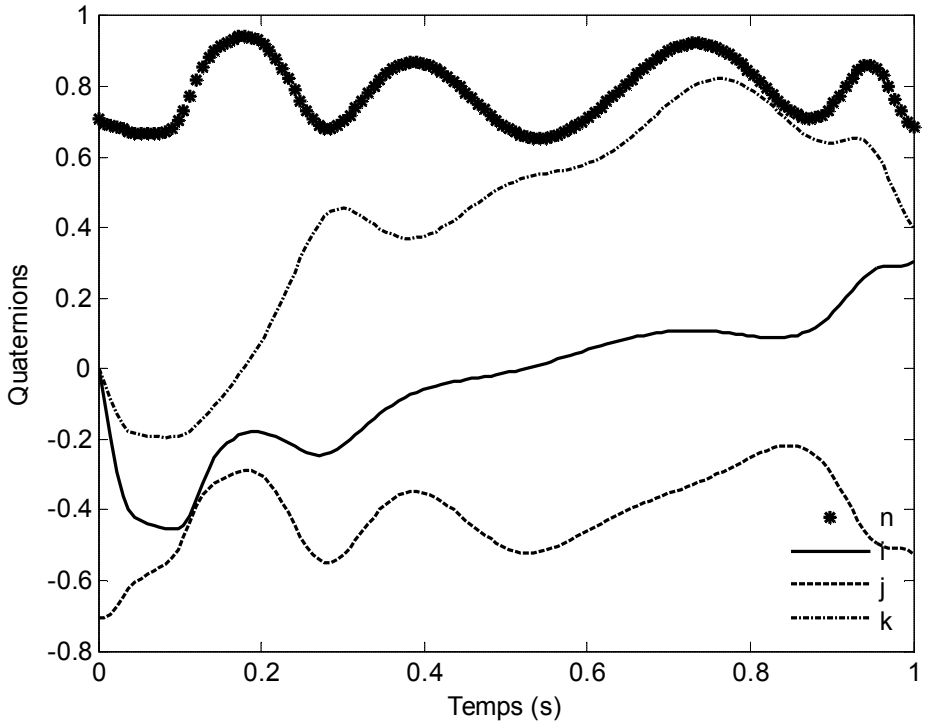


Figure 4.13. Quaternions résultants de l'optimisation par l'approche NURBS-recuit simulé dans un environnement sans obstacle

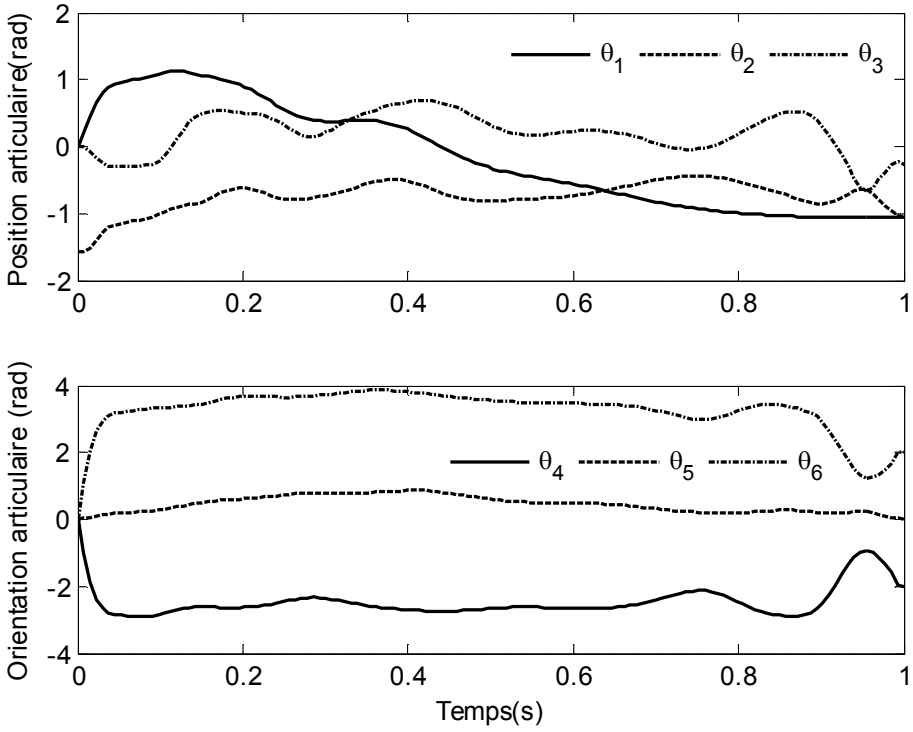


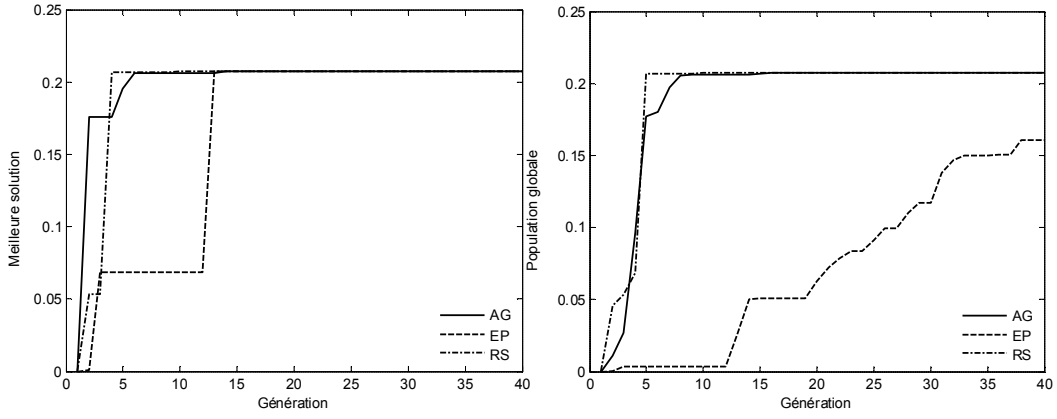
Figure 4.14. Positions articulaires résultantes de l'optimisation par l'approche NURBS-recuit simulé dans un environnement sans obstacle

4.7.2. Planification des trajectoires dans un environnement avec un obstacle

Dans ce cas, le but est de générer des trajectoires qui permettent à l'organe terminal du robot de déplacer de la situation initiale, passant par les points intermédiaires, jusqu'à la situation finale, mais sans avoir aucune collision avec l'obstacle. Encore, les trois approches utilisées précédemment pour la planification des trajectoires dans un environnement sans obstacle sont employées pour cet effet. De même, ces approches sont évaluées en termes de convergence de la meilleure solution et de convergence de la population globale vers la solution optimale. Les performances de ces approches sont montrées dans la figure 4.15. Cependant, le temps moyen d'exécution est donné dans le tableau 4.3. De plus, le mouvement du robot est représenté sur les figures 4.16, 4.19 et 4.22, alors que l'orientation de l'organe terminal du robot à l'aide des quaternions est représentée sur les figures 4.17, 4.20 et 4.23. Enfin, les positions articulaires nécessaires pour le mouvement du robot sont illustrées sur les figures 4.18, 4.21 et 4.24.

Tableau 4.3. Performance des approches dans un environnement avec un obstacle

Critère	Algorithme Génétique	Essaim particules	Recuit simulé
temps moyen d'exécution (min)	12	45	17



(a) (b)

Figure 4.15. Convergence des approches (a: meilleure solution, b : population globale) dans un environnement avec un obstacle

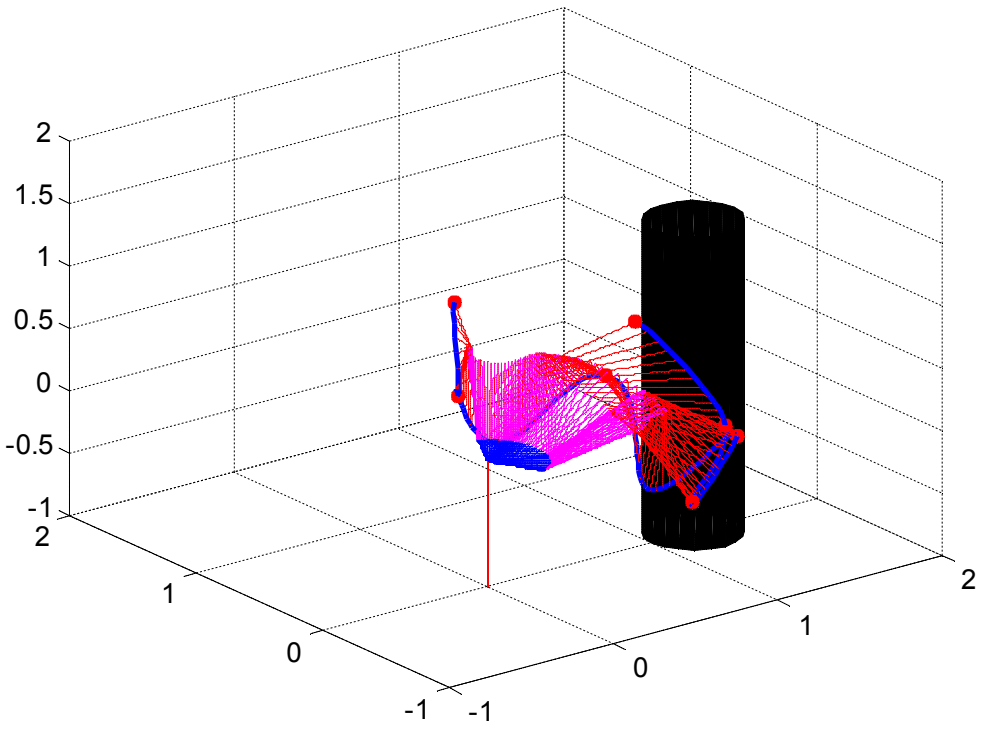
Du tableau 4.3, nous pouvons voir que l'algorithme génétique et le recuit simulé ont pris moins de temps que les essaims particulaires. Cela peut être interprété sur la base de la complexité des stratégies utilisées dans le mécanisme de recherche de chaque approche. Tandis que l'algorithme génétique et le recuit simulé incorporent des stratégies simples qui nécessitent moins de temps d'exécution, les essaims particulaires impliquent des stratégies un peu plus complexes et nécessitent plus de temps d'exécution.

De la figure 4.15, nous pouvons voir que les trois approches montrent un comportement similaire à celui présenté par elles dans un environnement sans obstacle. Encore une fois, la meilleure solution des trois approches atteint la même solution optimale obtenue dans un environnement sans obstacle. Cependant, pour la population globale, les essaims particulaires représentent l'exception. Cela signifie que la moyenne de toute la population n'atteint jamais la solution optimale. Généralement, ce comportement est dû au mécanisme de diversité utilisé par les essaims particulaires.

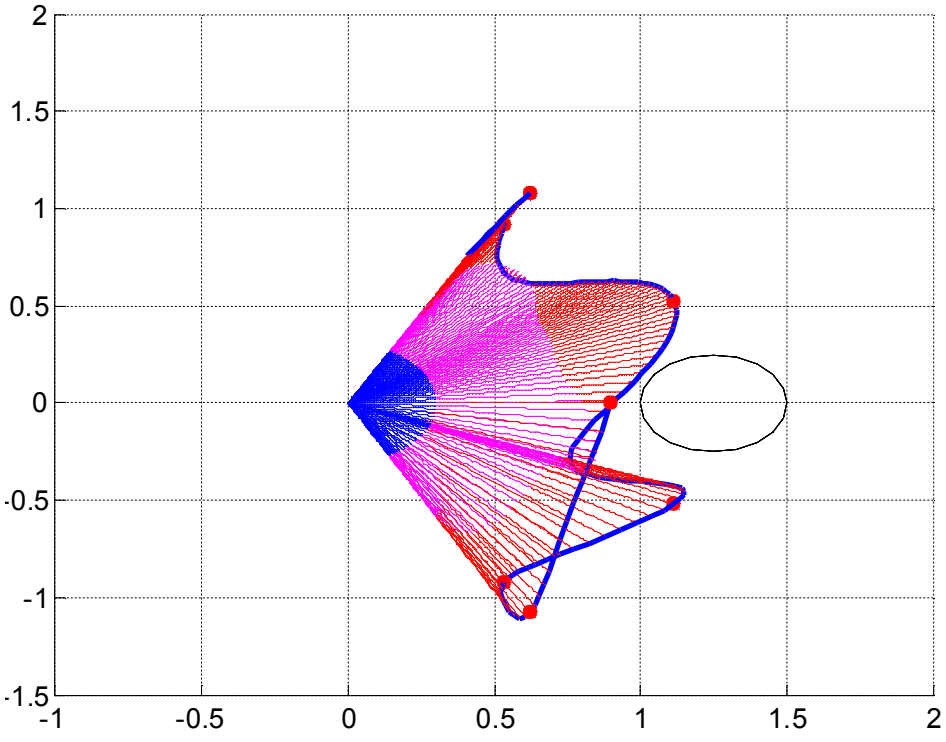
Les figures 4.16, 4.19 et 4.22 montrent les mouvements du robot manipulateur dans un environnement avec obstacle. Ces mouvements sont basés sur les trajectoires planifiées en utilisant les différentes approches d'optimisation. A partir des deux vues différentes, de l'espace à trois dimensions et du plan, on peut observer que l'obstacle est évité. De plus, on peut également remarquer que tous les points intermédiaires, du point de départ au point final, sont traversés.

Les figures 4.17, 4.20 et 4.23 montrent la variation de l'orientation de l'effecteur du robot en utilisant la représentation par quaternions. À partir de ces figures, nous pouvons observer que l'orientation de l'effecteur du robot change de manière très souple. Ceci peut être confirmé par leurs équivalents articulaires.

Les figures 4.18, 4.21 et 4.24 montrent les positions articulaires nécessaires pour les trajectoires générées. On peut remarquer que les angles articulaires requises pour positionner l'organe terminal du robot dans des situations qui permettent au robot lui-même d'éviter l'obstacle restent dans la même plage que celles présentés dans un environnement sans obstacle. Cela signifie que la présence de l'obstacle n'affecte pas l'efficacité des approches proposées pour générer des trajectoires souples.



(a)



(b)

Figure 4.16. Trajectoire optimisée par les algorithmes génétiques dans un environnement avec un obstacle. (a) vue 3D. (b) vue 2D.

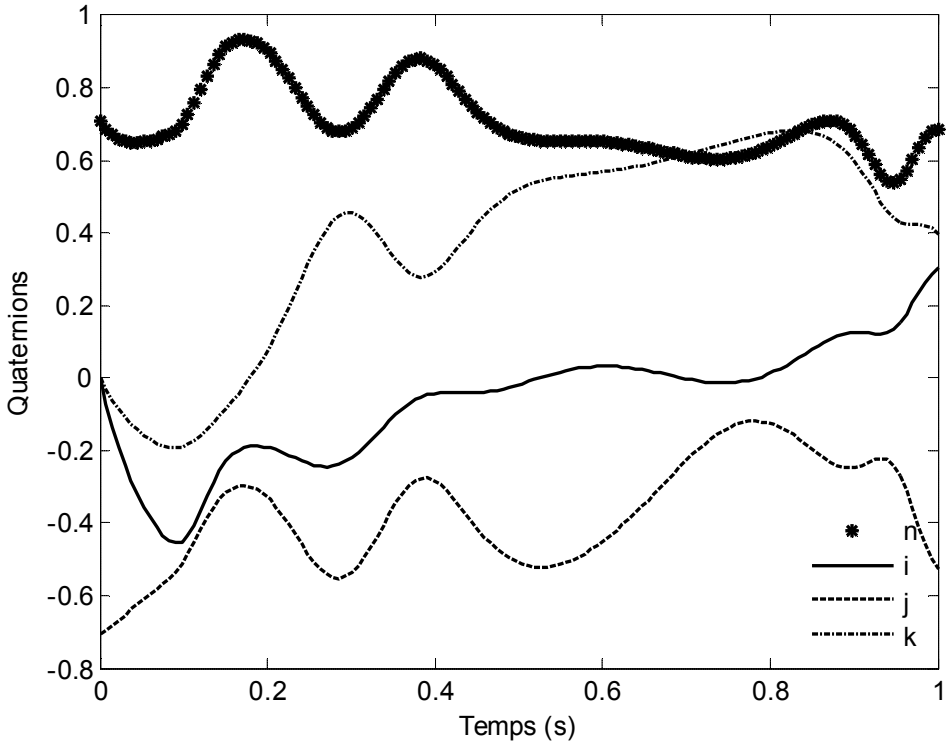


Figure 4.17. Quaternions résultants de l'optimisation par l'approche NURBS-algorithme génétique dans un environnement avec un obstacle

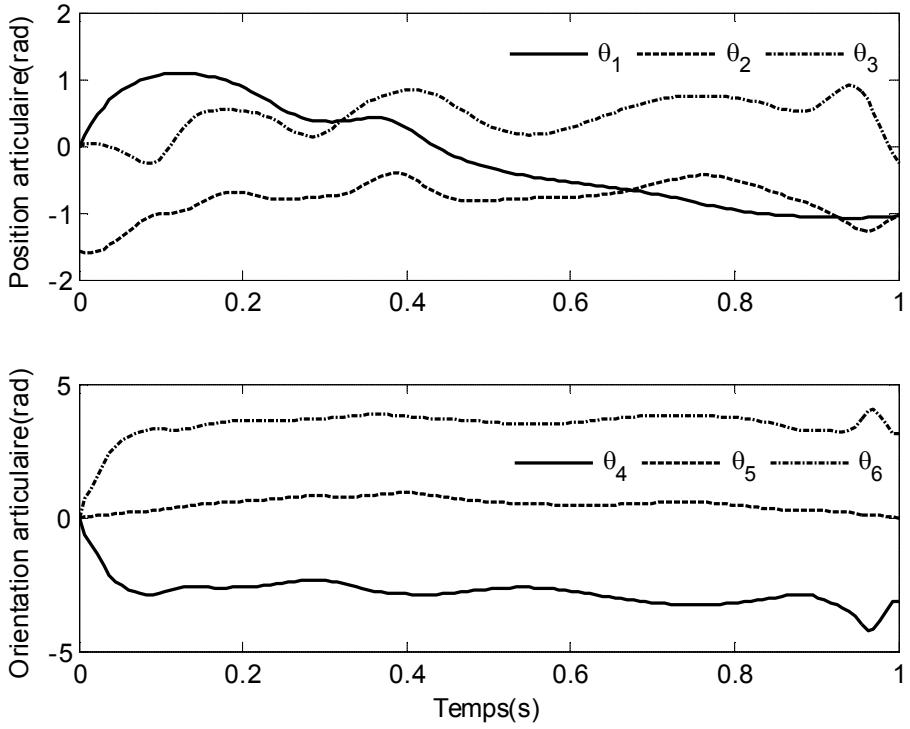
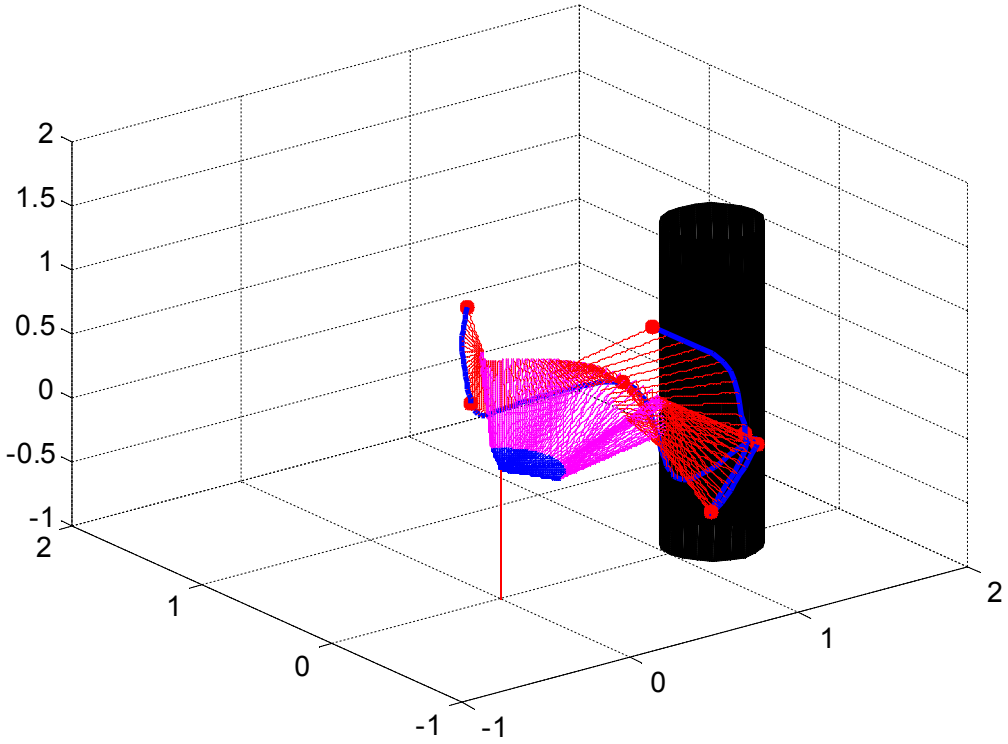
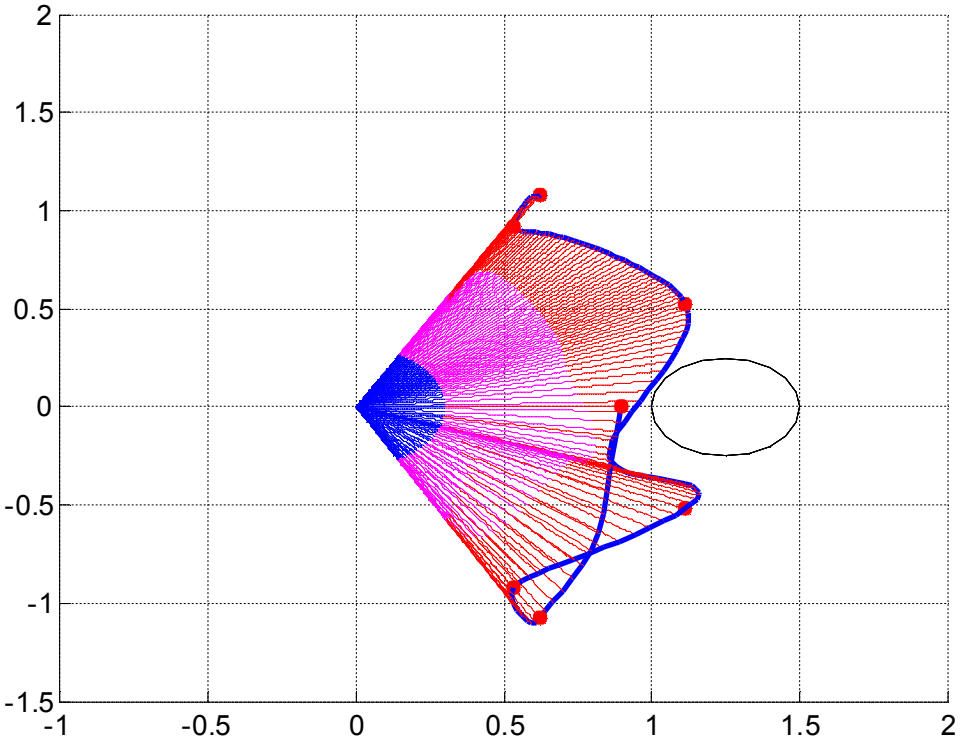


Figure 4.18. Positions articulaires résultantes de l'optimisation par l'approche NURBS-algorithme génétique dans un environnement avec un obstacle



(a)



(b)

Figure 4.19. Trajectoire optimisée par les essaims particulaires dans un environnement avec un obstacle. (a) vue 3D. (b) vue 2D.

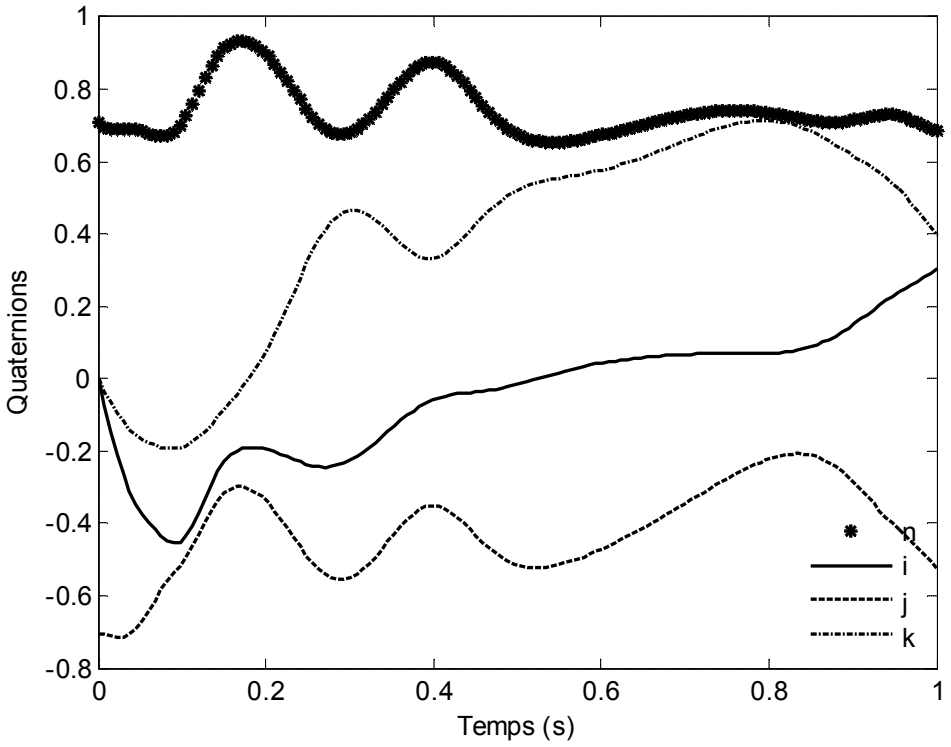


Figure 4.20. Quaternions résultants de l'optimisation par l'approche NURBS-essaims particulaires dans un environnement avec un obstacle

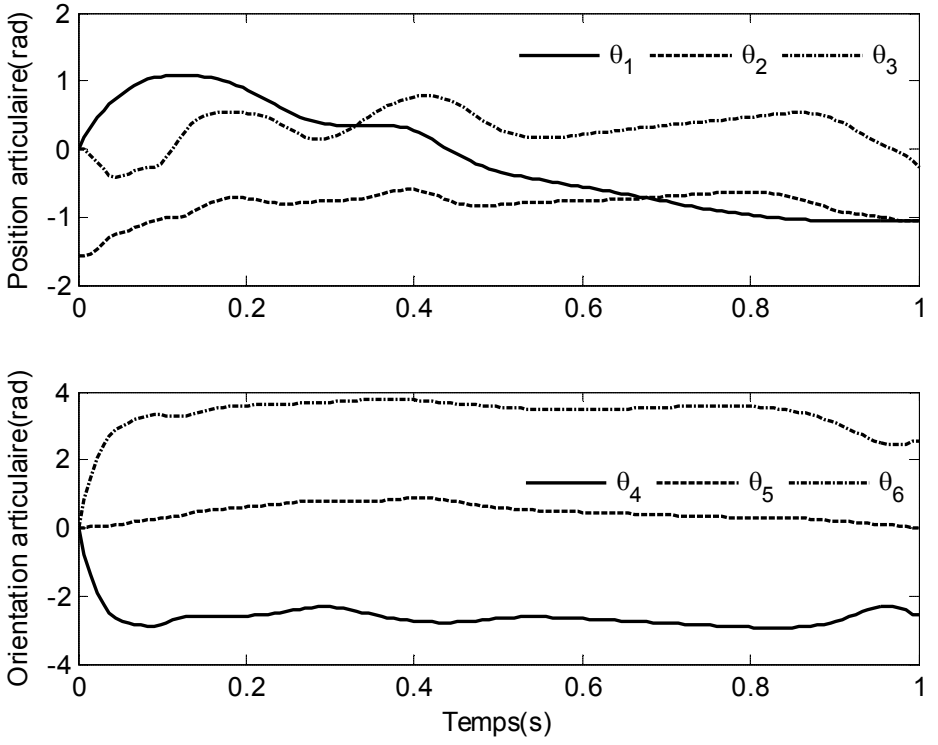
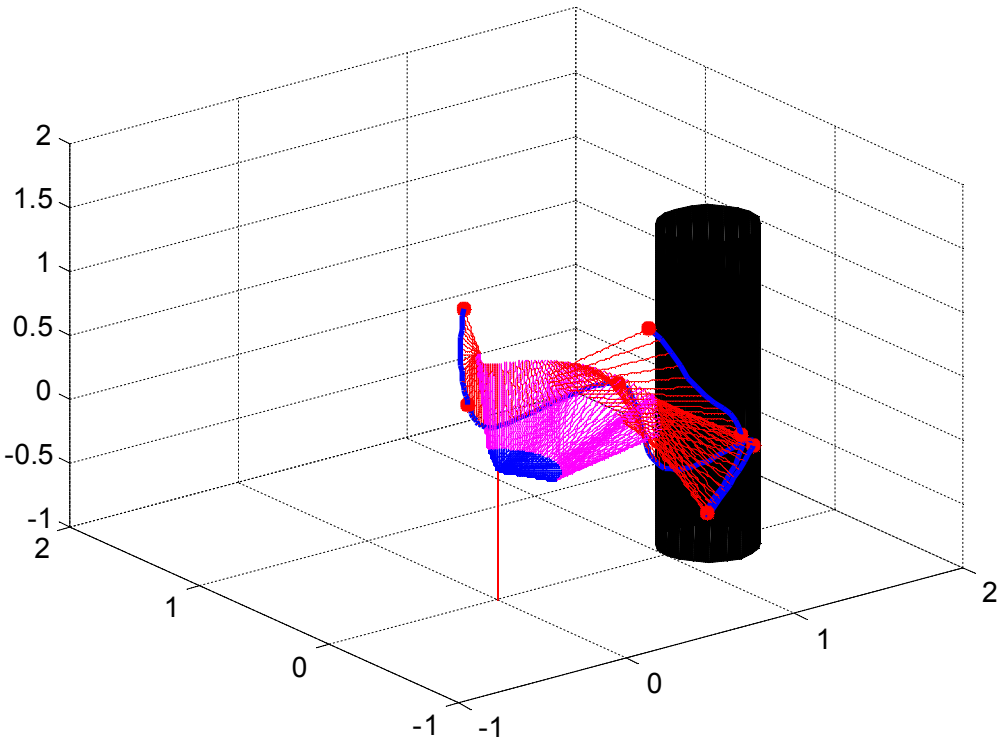
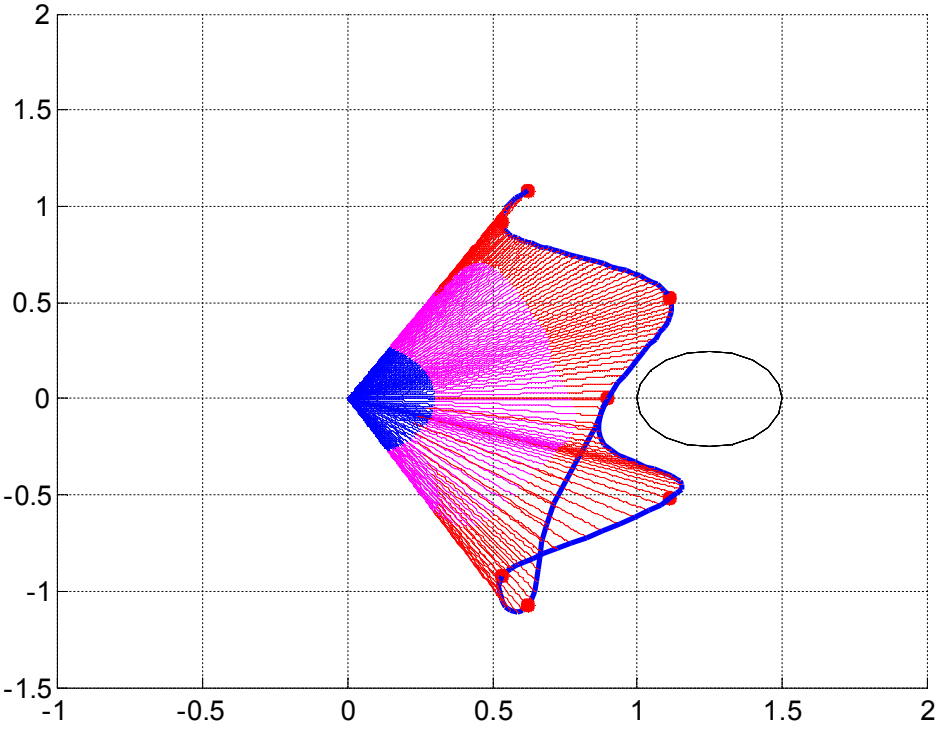


Figure 4.21. Positions articulaires résultantes de l'optimisation par l'approche NURBS-essaims particulaires dans un environnement avec un obstacle



(a)



(b)

Figure 4.22. Trajectoire optimisée par recuit simulé dans un environnement avec un obstacle. (a) vue 3D. (b) vue 2D.

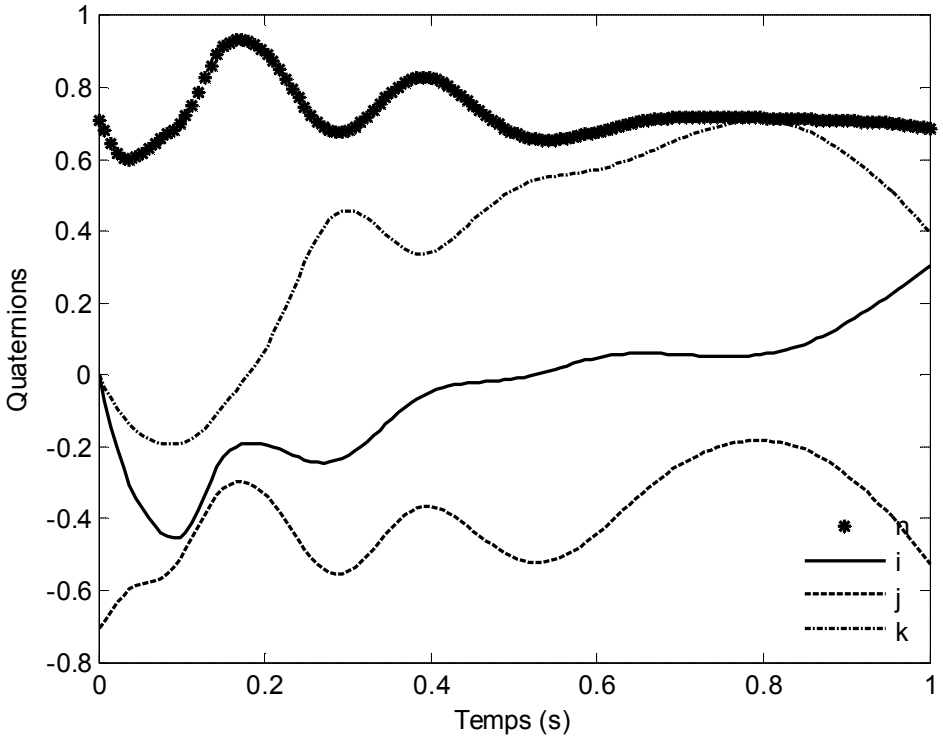


Figure 4.23. Quaternions résultants de l'optimisation par l'approche NURBS-recuit simulé dans un environnement avec un obstacle

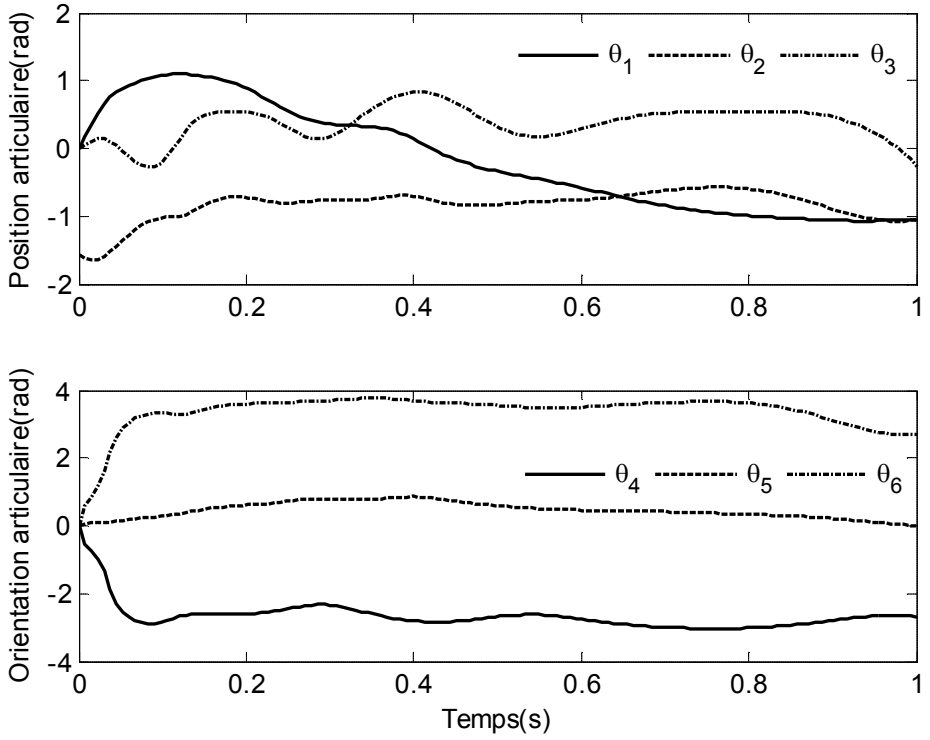


Figure 4.24. Positions articulaires résultantes de l'optimisation par l'approche NURBS-recuit simulé dans un environnement avec un obstacle

4.7.3. Planification des trajectoires dans un environnement avec deux obstacles

Dans un environnement avec deux obstacles, il ’y a plus de contraintes à satisfaire pour planifier les trajectoires. D’abord, les trajectoires planifiées doivent permettre à l’organe terminal du robot de traverser toutes les points de passage à partir de la situation initiale jusqu’au l’arrivée au but. En plus, ces trajectoires doivent aussi permettre à l’effecteur du robot de passer par l’un de ces points de passage qui est situé entre les deux obstacles sans que la structure du robot les touches. Pour cette raison, nous adoptons les mêmes approches utilisées dans les deux environnements précédents. Cependant, cette situation, où deux obstacles sont présentés dans l’environnement de travail du robot, nous permet de mieux évaluer les approches utilisées. Encore une fois, l’évaluation concerne la convergence de la meilleure solution et de la moyenne de la population globale vers la solution optimale, et le temps moyen d'exécution de chaque approche utilisée. Les résultats de cette évaluation sont présentées dans le tableau 4.4 et la figure 4.25.

Le mouvement résultant de la trajectoire planifiée par l’utilisation de chaque approche est montré sur les figures 4.26, 4.29 et 4.32. En plus, l’orientation de l’organe terminal du robot, en termes de la représentation par quaternions, est montrée dans les figures 4.27, 4.30 et 4.33. Enfin, les mouvements articulaires requises pour réaliser les trajectoires planifiées sont exprimés par le changement de position et d’orientation articulaires et ils sont représentés par les figures 4.28, 4.31 et 4.34.

Tableau 4.4. Performance des approches dans un environnement avec deux obstacles

Critère	Algorithme Génétique	Essaim particules	Recuit simulé
temps moyen d'exécution (min)	17	80	20

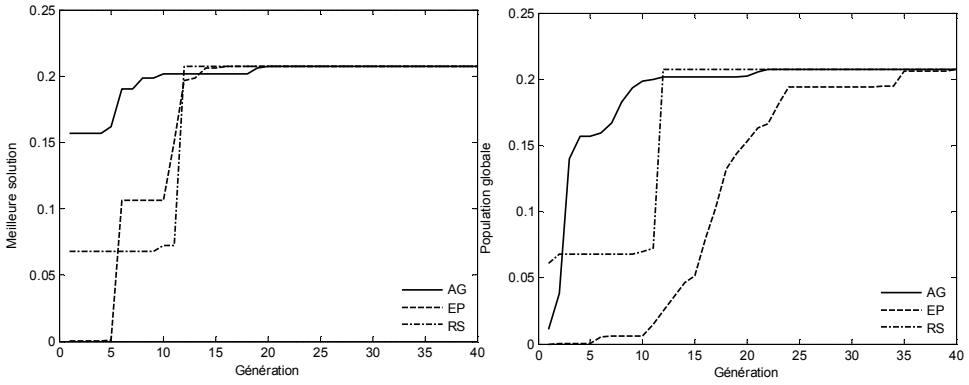


Figure 4.25. Convergence des approches (a: meilleure solution, b : population globale) dans un environnement avec deux obstacles

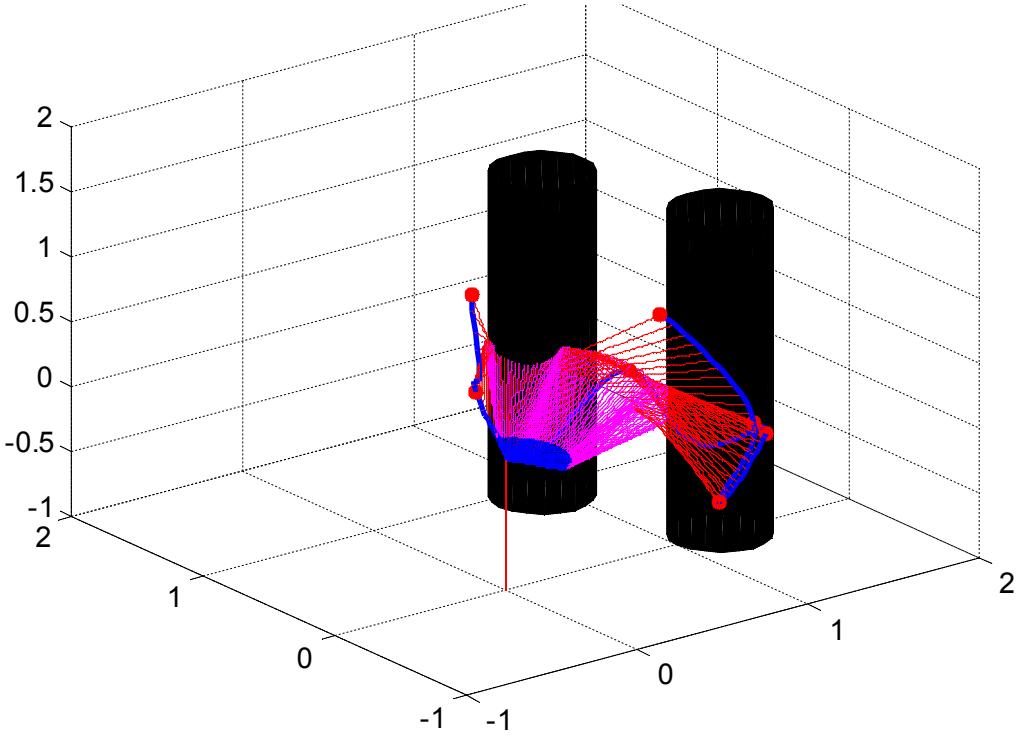
Comparant les performances des approches obtenues dans un environnement avec un seul obstacle (tableau 4.3) et celles obtenues dans un environnement avec deux obstacles (tableau 4.3), nous pouvons conclure que l'addition du deuxième obstacle dans l'environnement de travail est traduit par une augmentation du temps d'exécution pour les trois approches. Cependant, dans cet environnement, l'approche basée sur les algorithmes génétiques reste la meilleure en termes de temps d'exécution, alors que l'approche des essaims particulaires nécessite un temps maximale d'exécution. Ce critère de temps d'exécution est dépendant des mécanismes de recherche utilisés dans chaque approche ainsi que l'environnement de travail du robot.

Les performances des approches utilisées, qui sont représentés dans la figure 4.25, montrent que la meilleure solution des trois approches atteint toujours la solution optimale même si l'environnement de travail du robot est changé. Cependant, la moyenne de la population globale aboutie la solution optimale seulement par les approches basées sur les algorithmes génétiques et le recuit simulé. Pour l'approche basée sur les essaims particulaires, la moyenne de la population globale a vu le même comportement que celle obtenue dans un environnement avec un seul obstacle. Par conséquence cette moyenne n'atteint jamais la solution optimale. Cela est causé par l'hétérogénéité de solutions générées par les essaims particulaires.

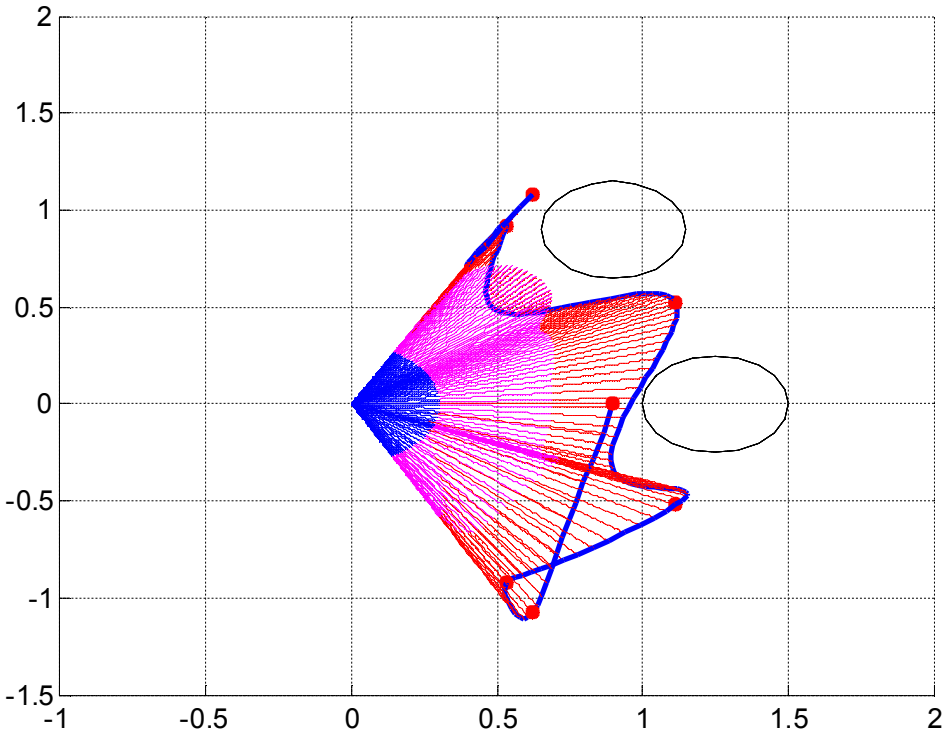
Les figures 4.26, 4.29 et 4.32 montrent les différentes configurations que le robot doit passer pour réaliser la trajectoire planifiée dans l'espace cartésien où deux obstacles sont présentés. Ces configurations sont présentées dans l'espace à trois dimensions et dans un plan pour montrer comment la trajectoire planifiée permet au robot d'évité les deux obstacles toutes on traversant les points de passage.

Les trajectoires planifiées, par l'utilisation des approches proposées, pour orienter l'effecteur du robot sont montrées en termes des quaternions dans les figures 4.27, 4.30 et 4.33. On constate de ces courbes que l'orientation de l'organe terminal du robot varie d'une façon très souples ce qui est confirmé par les angles d'orientation correspondantes.

Les différentes configurations que le robot doit réaliser, en termes des positions et des orientations articulaires, durant son mouvement sont montées dans les figures 4.28, 4.31 et 4.34. Encore une fois, on peut voir que les positions articulaires nécessaires pour permettre au robot de réaliser la trajectoire planifiée dans un environnement avec deux obstacles varient dans la même plage que celles montrés dans les deux environnements précédents. Cela signifie que l'efficacité des approches utilisées pour planifier des trajectoires souples est n'est pas influencer par le changement d'environnements.



(a)



(b)

Figure 4.26. Trajectoire optimisée par les algorithmes génétiques dans un environnement avec deux obstacles. (a) vue 3D. (b) vue 2D.

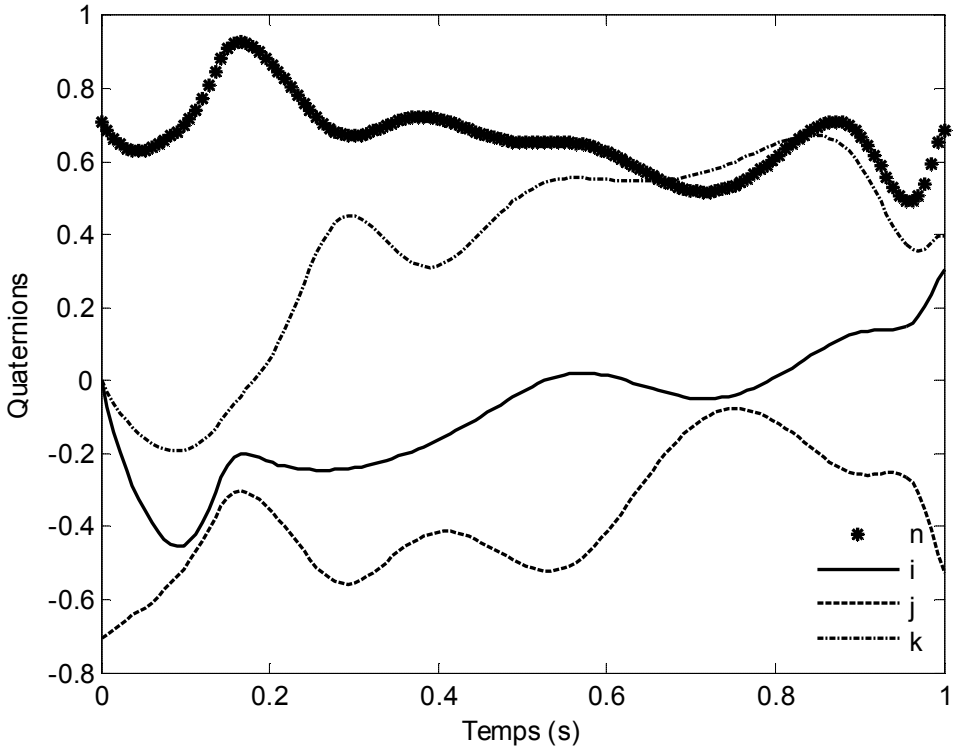


Figure 4.27. Quaternions résultants de l'optimisation par l'approche NURBS-algorithmes génétiques dans un environnement avec deux obstacles

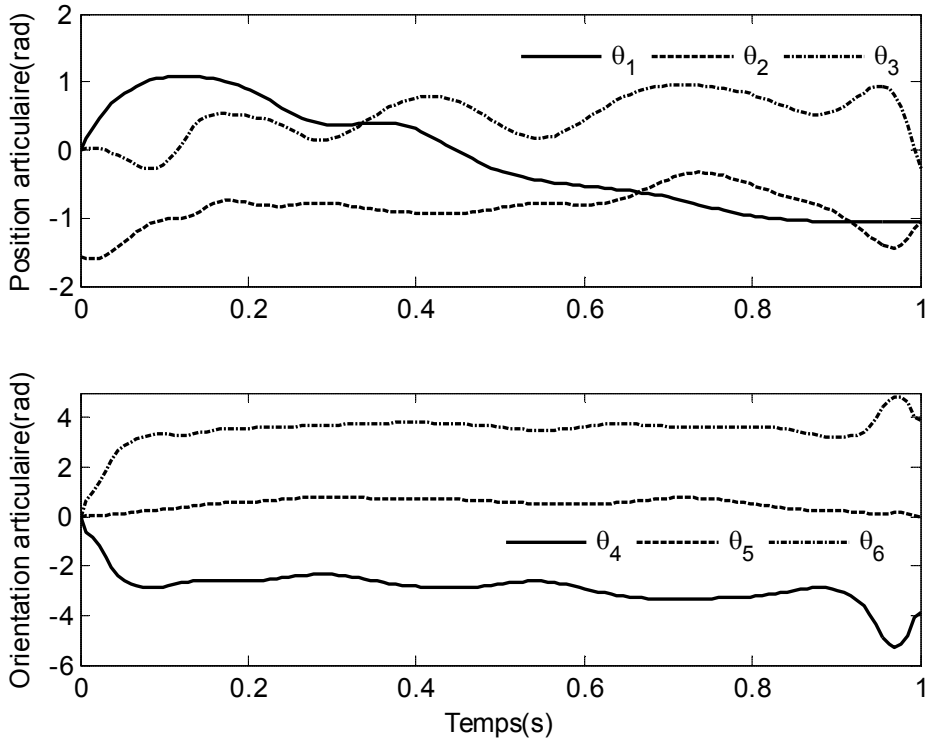
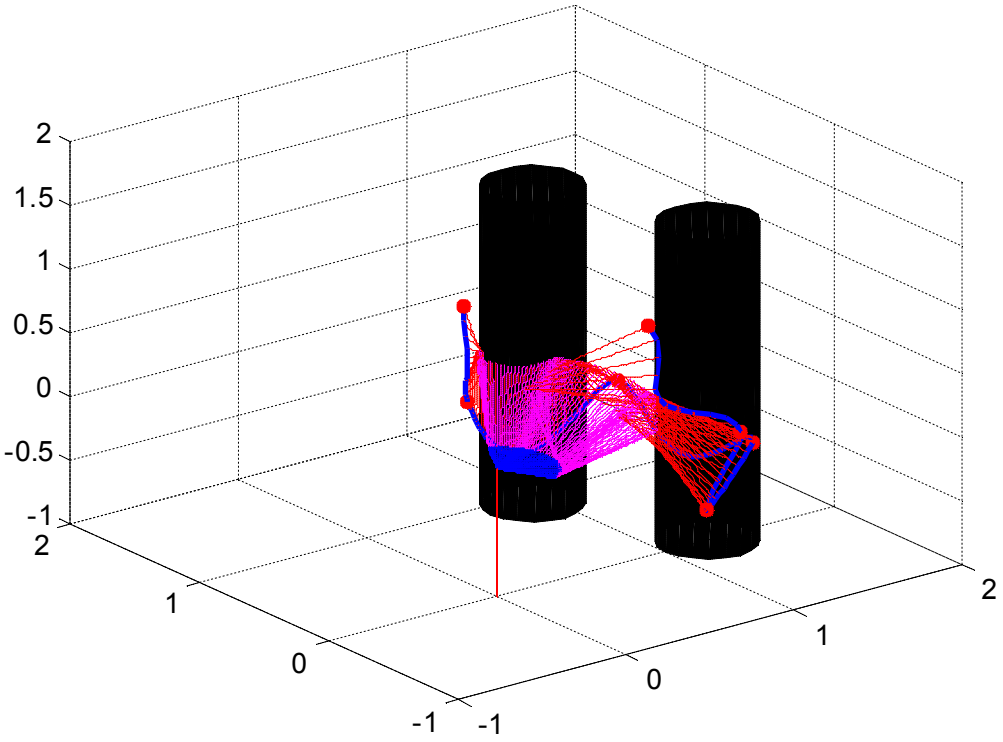
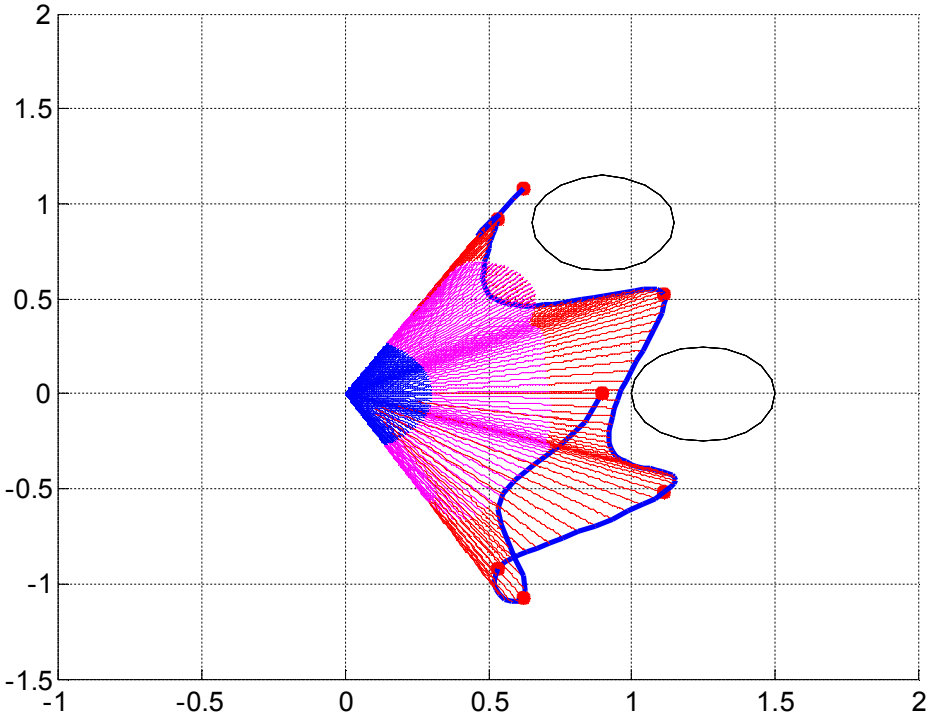


Figure 4.28. Postions articulaires résultants de l'optimisation par l'approche NURBS-algorithme génétique dans un environnement avec deux obstacles



(a)



(b)

Figure 4.29. Trajectoire optimisée par les essaims particulaires dans un environnement avec deux obstacles. (a) vue 3D. (b) vue 2D.

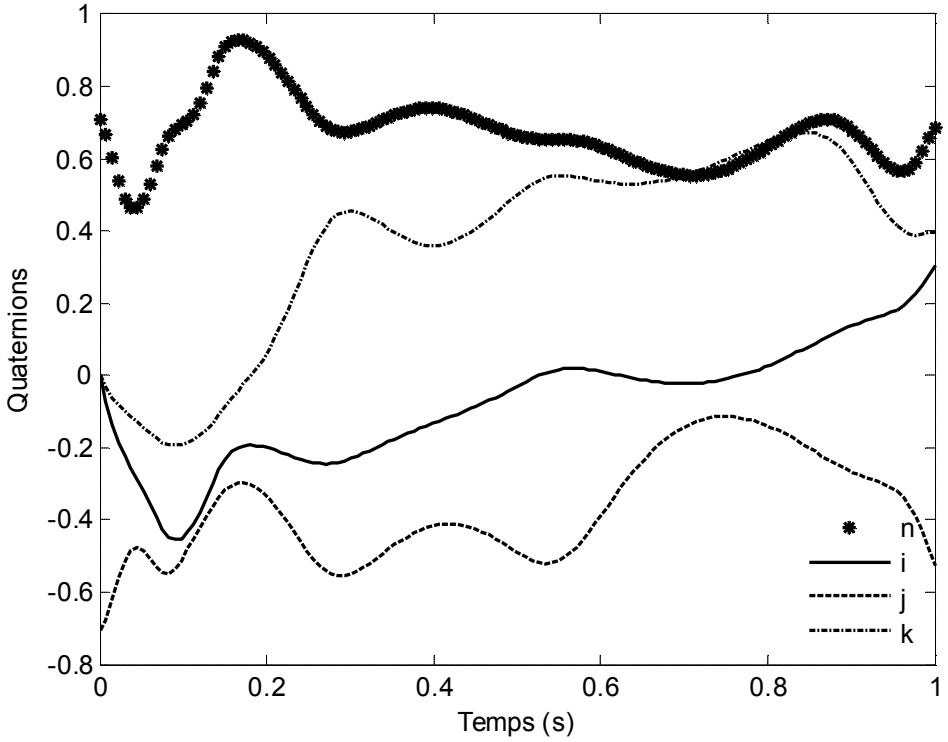


Figure 4.30. Quaternions résultants de l'optimisation par l'approche NURBS-essaims particulaires dans un environnement avec deux obstacles.

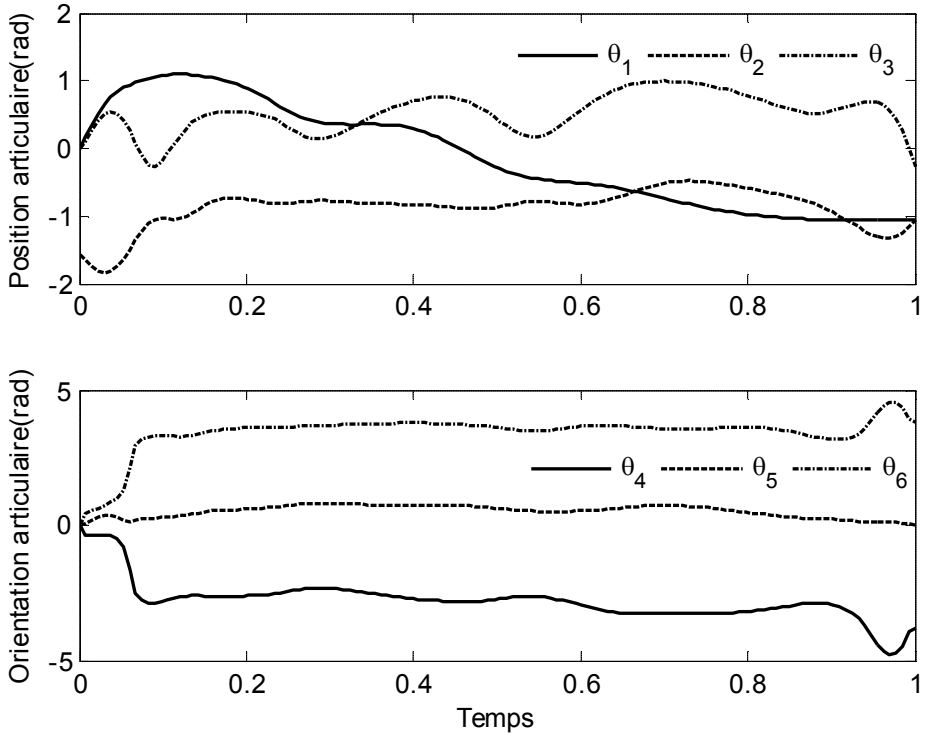
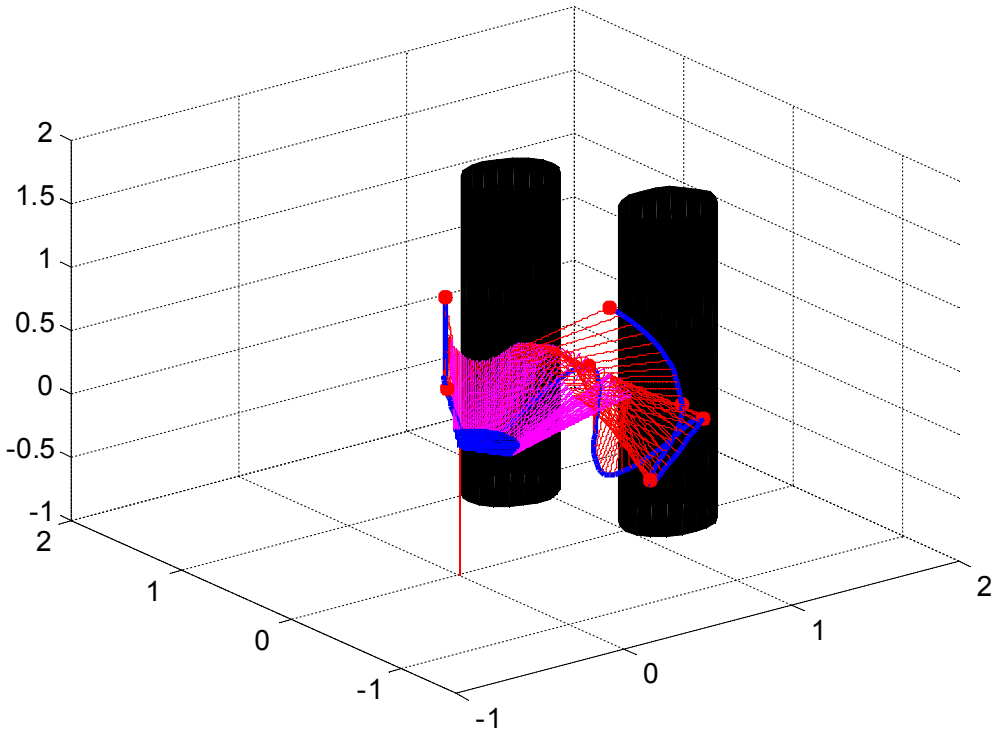
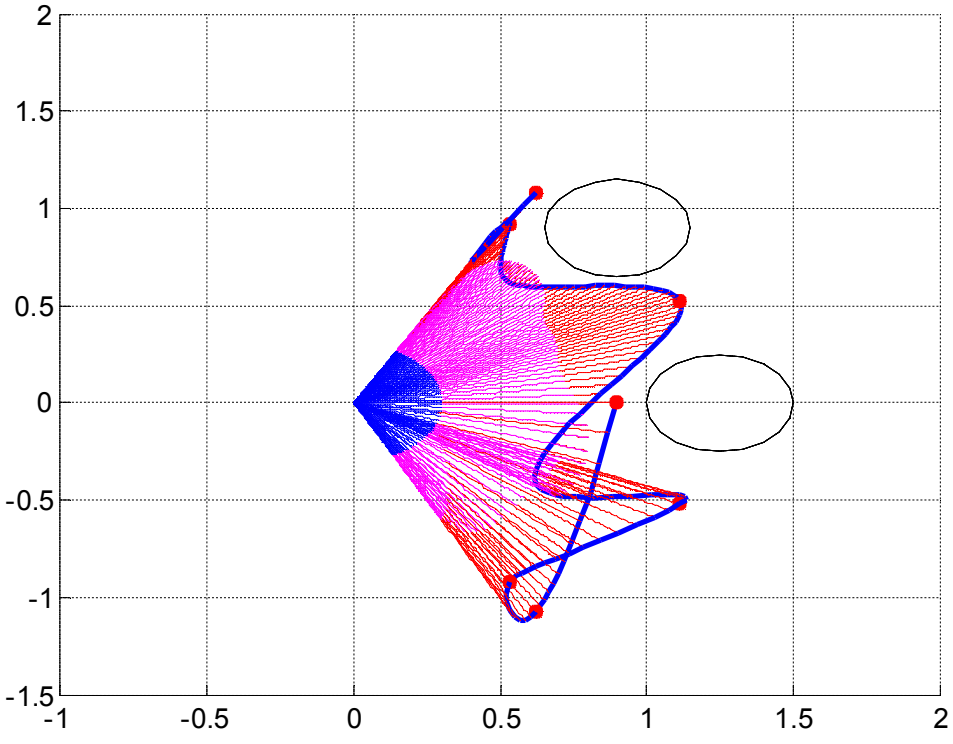


Figure 4.31. Positions articulaires résultantes de l'optimisation par l'approche NURBS-essaims particulaires dans un environnement avec deux obstacles



(a)



(b)

Figure 4.32. Trajectoire optimisée par recuit simulé dans un environnement avec deux obstacles. (a) vue 3D. (b) vue 2D.

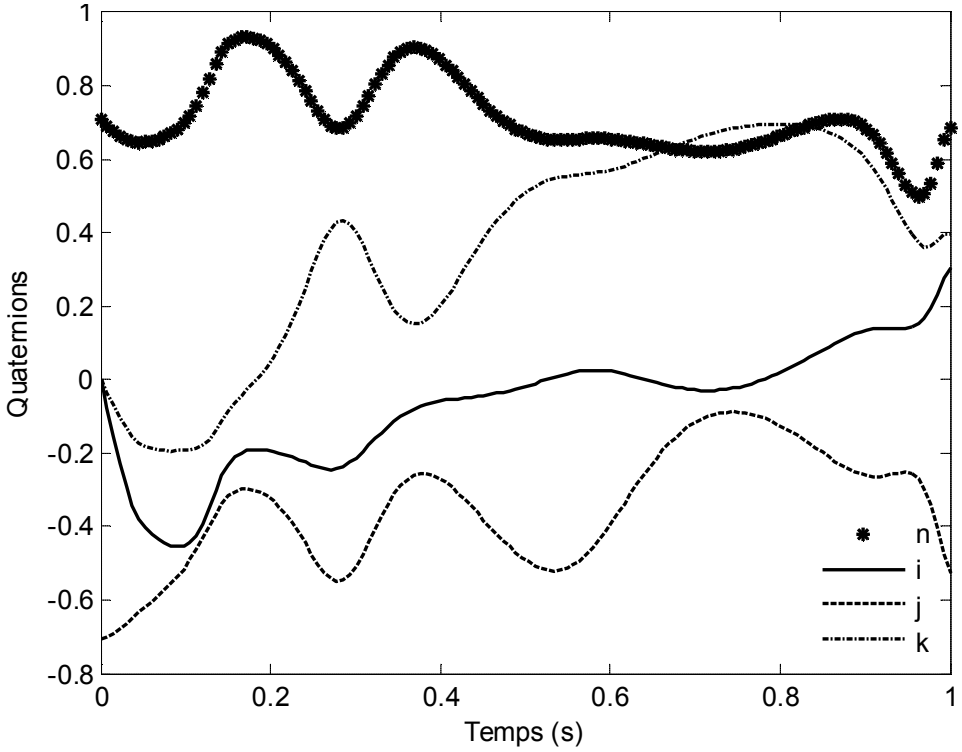


Figure 4.33. Quaternions résultants de l'optimisation par l'approche NURBS-recuit simulé dans un environnement avec deux obstacles

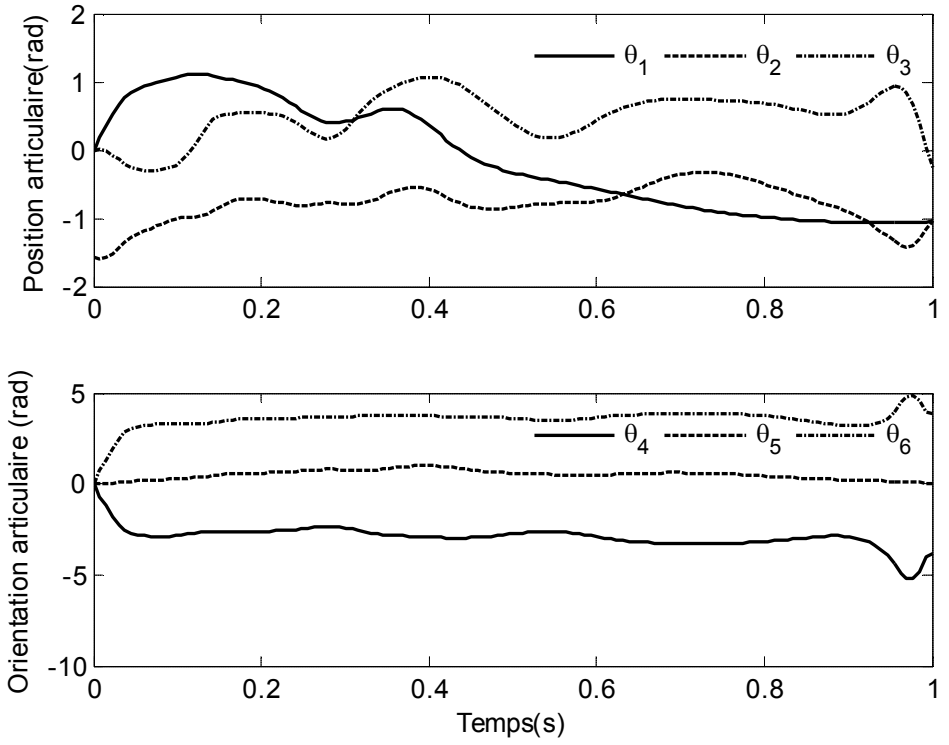


Figure 4.34. Positions articulaires résultantes de l'optimisation par l'approche NURBS-recuit simulé dans un environnement avec deux obstacles.

Conclusion

Dans ce chapitre, nous avons présenté trois approches pour résoudre le problème de planification des trajectoires d'un robot manipulateur. Ce problème a été converti en un problème d'optimisation des poids du chemin qui est décrit en utilisant une courbe NURBS d'ordre quatre. De cette manière, de nombreux critères ont été utilisés pour définir la fonction objective du problème d'optimisation. Ces critères concernent le temps d'exécution, les distances articulaire et cartésienne parcourues, l'évitement des singularités et des collisions. Les approches proposées ont été employées pour trouver les poids appropriés qui font passer l'effecteur du robot à travers les points intermédiaires en respectant les contraintes supplémentaires. Dans ce contexte, trois situations ont été étudiées. Dans la première, il n'y avait aucun obstacle dans l'environnement de travail. Cependant, dans la deuxième, un obstacle a été défini pour couper le segment reliant les points intermédiaires. Un environnement à deux obstacles a été adopté dans la troisième situation. Les résultats de simulation montrent que les trois approches résolvent efficacement le problème de planification des trajectoires. Les trajectoires planifiées soit dans l'environnement libre ou dans les environnements avec obstacles étaient suffisamment lisses pour permettre un mouvement approprié du robot.

Conclusion Générale

Conclusion générale

Dans cette thèse nous avons présenté trois approches pour la planification des trajectoires dans l'espace cartesian. Ces approches sont basées sur l'utilisation de trois métaheuristiques pour l'optimisation des trajectoires décrites à l'aide des courbes NURBS. Cette formulation permet de considérer plusieurs contraintes liées à la planification des trajectoires comme le temps d'exécutions, l'énergie consommée, les singularités et l'évitement d'obstacles.

La première approche que nous avons développée est basée sur l'emploi des algorithmes génétiques pour l'optimisation des poids de la courbe NURBS qui définit la trajectoire du robot manipulateur. Dans cette approche nous avons décrit un ensemble des situations intermédiaires dans l'espace Cartesian, où le mouvement de l'organe terminal du robot est défini. Chacune de ces situations est définie par un ensemble de sept composantes. Les trois premières composantes sont reliées à la position de l'organe terminale du robot dans l'espace, alors que les autres composantes concernent son orientation et elles sont définies à l'aide de la représentation par quaternions pour éviter le problème de singularité d'orientation. Ensuite, nous avons employé une courbe NURBS pour interpoler les points de passage de l'organe terminal. Cette courbe présente une propriété importante qui nous permet la déformation locale de son forme à l'aide de la manipulation de ses poids. On basant sur cette propriété, la population initiale des algorithmes génétiques est générer de façon que chaque individu représente un vecteur de poids de la courbe NURBS et alors une solution potentiel au problème de la planification des trajectoires. La qualité de ces individus est ensuite mesurée à l'aide de la fonction objective employée. Cependant, les individus sont ajustés en employant les opérateurs de croissement et de mutation. A chaque génération les meilleurs survivent et les mauvaises disparaissent. A la fin du cycle d'évolution des algorithmes génétiques, un ensemble de solution homogène est proposé.

La deuxième approche que nous avons proposé pour la manipulation des poids de la courbe NURBS est basé sur l'utilisation d'une autre métaheuristique à base population, appelée les essaimes particulaires. Dans cette approche chaque particule représente un vecteur des poids de la courbe NURBS où la fonction objective est utilisée pour juger son qualité. Ensuite l'ensemble des particules sont déplacés dans l'espace de recherche en fonction des positions et des vitesses des meilleures particules. Après un certain nombre de

génération, les positions des particules sont considérées comme solutions potentielle au problème de planification des trajectoires.

Contrairement aux premières approches qui utilisent des métaheuristiques à base population, la troisième approche que nous avons suggéré est fondé sur l'utilisation d'une métaheuristique à base individu nommée le recuit simulé. Pour cette approche chaque configuration des atomes représente un vecteur des poids de la courbe NURBS. Alors, cette approche est basée sur la génération aléatoire d'une configuration initiale et la mesure de son qualité à l'aide de la fonction objective. Ensuite, un processus itératif, basé sur la comparaison de la configuration initiale avec une autre configuration qui est généré aussi aléatoirement, est employé pour guider le système vers la stabilisation à une énergie minimale. La configuration qui conduit à cette stabilisation est considéré comme la seule solution au problème de planification des trajectoires.

Les trois approches que nous avons proposées sont utilisées pour résoudre le problème de planification des trajectoires pour un robot manipulateur à six dégréé de liberté, KUKA KR 15, dans trois environnements différent. Le premier environnement ne contient aucun obstacle. Le but est de déplacer l'organe terminal du robot à travers les situations intermédiaires. Pour atteindre ce but, la fonction objective, employé dans les trois approches, est définit de façons à inclut les critères de temps d'exécution, d'énergie et d'évitement de singularité.

Dans la deuxième situation, les approches que nous avons proposées sont employées pour résoudre le problème de planification des trajectoires du même robot mais dans un environnement qui contient un obstacle statique. Pour cet effet, la fonction objective contient en plus des critères de temps d'exécution, d'énergie et d'évitement de singularité, le critère d'évitement d'obstacle.

Dans la troisième situation, nous avons considéré un environnement à deux obstacles. Donc, la fonction objective utilisée dans un environnement avec un seul obstacle est ajustée pour prendre en compte l'évitement du deuxième obstacle. Encore une fois, nous avons employé les mêmes approches pour résoudre le problème de planification des trajectoires dans cet environnement.

Comme perspectives on s'intéresse à la validation des résultats obtenues, l'amélioration des performances des approches utilisées pour les implémenter en temps réel et l'utilisation des mêmes approches pour les robots mobiles.

Références bibliographiques

[Acar 02] Acar, E. U. Choset, H. Rizzi, A. A. Atkar, P. and Hull, D. Morse decompositions for coverage tasks. *International Journal of Robotics Research,* 21:331–344, April 2002.

[Ahmad 91] Ahmad, R. K and Joey K. P. Obstacle avoidance of redundant manipulators using genetic algorithms. In IEEE Proceedings of the Southeast SOUTHEASTCON'91, volume 1, 1991: 317-320.

[Ahmed 12] Ahmed, H. Heba, M. Mohamed, B. Osama, S and Alaa, K (2012). "Metaheuristic Optimization Approach to Mobile Robot Path Planning". In proceedings of International Conference on Engineering and Technology (ICET2012).

[Biagiotti 08] Biagiotti, L., Melchiorri, C. "Trajectory Planning for Automatic Machines and Robots." Springer-Verlag Berlin Heidelberg. 2008.

[Brand 10] Brand, M., Masuda, M., Wehner, N., Yu, X.H. Ant colony optimization algorithm for robot path planning In: Proceedings of the 2010 International Conference on Computer Design and Applications (ICCDA); 25–27 June 2010; Qinhuangdao. New York: IEEE; 2010. pp. V3–436–V3–440. DOI: 10.1109/ICCDA.2010.5541300.

[Canny 93] Canny, J. F. and Lin, M. An opportunistic global path planner. *Algorithmica,*10:102–120, 1993.

[Canny 98] Canny, J. F. *The Complexity of Robot Motion Planning*. MIT Press, Cambridge, MA, 1988.

[Choset 98] Choset H, Pignon P (1998) Coverage path planning: the boustrophedon cellular decomposition. *In: Zelinsky A (ed) Field and service robotics. Springer,* London, pp 203–209

[Choset 05] Choset, H. Lynch, K. M. Hutchinson, S. Kantor, G. A. Burgard, W. Kavraki, L. E. and Thrun. S. Principles of Robot Motion: Theory, Algorithms, and Implementations. MIT Press, Cambridge. MA, June 2005.

[Clerc 99] Clerc, M. "The Swarm and the Queen Towards the Deterministic and Adaptive Particle Swarm Optimization," Proceedings of the Congress on Evolutionary Computation, Washington, DC, 1999, pp. 1951-1957.

[Colorni 92] Colorni A, Dorigo M and Maniezzo V. (1992). Distributed optimization by ant colonies Proc. 1st European Conf. on Artificial Life (Paris, France,) pp 134-142.

[Denavit 55] Denavit J., Hartenberg R.S., "A kinematic notation for lower pair mechanism based on matrices", *Trans. of ASME, J. of Applied Mechanics*, Vol. 22, June 1955, p. 215-221.

[Dorigo 97a] Dorigo, M , Gambardella, L. M. (1997a). Ant colonies for the traveling salesman problem. BioSystems, 43(2), 73–81.

[Dorigo 96] Dorigo, M. Gambardella, L. M. (1996). A study of some properties of

Ant-Q. In H. Voigt, W. Ebeling, I. Rechenberg, and H. Schwefel (Eds.), Proceedings of PPSN-IV, Fourth International Conference on Parallel Problem Solving from Nature, vol. 1141 of Lecture Notes in Computer Science (pp. 656–665). Berlin, Springer-Verlag.

[Dorigo 97b] Dorigo, M., & Gambardella, L. M. (1997b). Ant Colony System: A cooperative learning approach to the traveling salesman problem. IEEE Transactions on Evolutionary Computation, 1(1), 53–66.

[Fatima 16] Fatima, Z. B. Larbi, E. B, Yassine L. Optimization of Arm Manipulator Trajectory Planning in The Presence of Obstacles by Ant Colony Algorithm. 10th International Conference Interdisciplinarity in Engineering, INTER-ENG 2016.

[Fogel 66] Fogel, L. J., Owens, A. J., and Walsh, M. J. (1966). *Artificial Intelligence through Simulated Evolution*. New York: John Wiley and Sons.

[Funda 90] Funda, J., Taylor, R. H., Paul, R. P. "On homogeneous transforms, quaternions and computational efficiency." IEEE Transactions on Robotics and Automation. 6, pp. 382–388. 1990.

[Gasparetto 15] Gasparetto, A., Boscariol, P., Lanzutti, A., and Vidoni, R. Path Planning and Trajectory Planning Algorithms: A General Overview. in Motion and Operation Planning of Robotic Systems, Part I Theoretical Background, pages 3-27. Marco Ceccarelli, Cassino, Italy, 2015.

[Glover 89] Glover, F. "Tabu Search — Part I", ORSA Journal on Computing 1989 1: 3, 190-206.

[Glover 86] Glover, F. “Future Paths for Integer Programming and Links to Artificial Intelligence,” Computers & Operations Research, Vol. 13, No. 5, 1986, pp. 533-549.

[Goldberg 89] Goldberg, D. E. (1989). Genetic algorithms in search, optimization, and machine learning. Reading, MA: Addison-Wesley. 74,527

[Gorla 84] Gorla B., Renaud M., *Modèles des robots-manipulateurs; application à leur commande*, Cepadues Editions, Toulouse, 1984.

[Goss 89] Goss, S., Aron, S., Deneubourg, J. L., and Pasteels, J. M. (1989). Self-organized shortcuts in the Argentine ant. Naturwissenschaften, 76, 579–581.

[Grag 02] Grag, D. P. Kumar, M. “Optimization Techniques Applied To Multiple Manipulators for Path Planning and Torque Minimization”, Engineering Applications of Artificial Intelligence, Volume 15, Number3, June 2002, pp. 241-252(12).

[Guo 10] Guo, J. Wang, X. and Zheng, X. Trajectory planning of redundant robot manipulators using QPSO algorithm. In : Intelligent Control and Automation (WCICA), 2010 8th World Congress on. IEEE, 2010. p. 403-407. Jinan, China

[Heppner 90] Heppner, F., and Grenander, U. (1990). A stochastic nonlinear model for coordinated bird flocks. In S. Krasner (Ed.), The Ubiquity of Chaos. Washington, DC: AAAS Publications.

[Holland 75] Holland, J. H. (1975). *Adaptation in natural and artificial systems*. Ann Arbor: the university of Michigan Press

[Huang 04] Huang, H and Chung, S. Dynamic visibility graph for path planning. In Intelligent Robots and Systems, 2004. (IROS 2004). 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566), pp.2813-2818, 2004.

[Imen 14] Imen, C. Anis, K. Hachemi, B. Adel, A. Sahar, T. Mohamed, T. Elhadi, S and Habib, Y. On the Adequacy of Tabu Search for Global Robot Path Planning Problem in Grid Environments. 5th International Conference on Ambient Systems, Networks and Technologies (ANT-2014).

[Itzhack 00] Itzhack, B., Itzhack, Y. "New Method for Extracting the Quaternion from a Rotation Matrix." AIAA Journal of Guidance, Control, and Dynamics. 23(6), pp. 1085-1087. 2000.

[Jason 95] Jason, A. Janet, Ren C. Luo, and Michael G.Kay. The essential visibility graph: An approach to global motion planning for autonomous mobile robots. In IEEE International Conference on Robotics and Automation, 1958-1963, 1995.

[Jianjun 17] Jianjun, Y. Cheng,S. Le, Z. Chenguang, X. Ming, Y and Shiqi, Z. Time Optimal Trajectory Planning Based on Simulated Annealing Algorithm for a Train Uncoupling Robot. 2017 29th Chinese Control And Decision Conference (CCDC 2017).

[Jiménez 98] Jiménez, P.Thomas, F. and Torras,C. Collision detection algorithms for motion planning. In J.-P. *Laumond, editor, Robot Motion Planning and Control,* pages 305-343. Springer-Verlag, Berlin, 1998.

[Jin 16] Jin, X. Kang, J. F. Zhang, J. J. Yang, X. Trajectory Planning of a Six-DOF Robot based on a Hybrid Optimization Algorithm. 9th International Symposium on Computational Intelligence and Design Dec. 2016.

[Kamon 96] Kamon, I. Rivlin, E. and Rimon, E. A new range-sensor based globally convergent navigation for mobile robots. In *IEEE Int'l. Conf. on Robotics and Automation,* Minneapolis, MN, April 1996.

[Kennedy 95] Kennedy, J., and Eberhart, R. C. (1995). :Particle Swam Optimization; Roc. IEEE International Conference on Neural Networks (Path, Australia), IEEE Service Center, Piscataway, NJ, N: 1942-1948

[Kenneth 08] Kenneth, W and *James, S*. Kinematics. In Handbook of Robotics, chapter A.1, pages 9–33. *Springer*-Verlag, Berlin, Heidelberg, *2008*.

[Khalil 99] Khalil W., Dombre E., *Modélisation, identification et commande des robots*, Hermès, 1999.

[Khalil 91] Khalil W., Gautier M., "Calculation of the identifiable parameters for robot calibration", *Proc. IFAC Symp. on Identification and System Parameter Estimation*, Budapest, 1991, p. 888-892.

[Khalil 86] Khalil W., KLEINFINGER J.-F., "A new geometric notation for open and closed-loop robots", *Proc. IEEE Int. Conf. on Robotics and Automation*, San Francisco, April 1986, p. 1174-1180.

[Khalil 07] Khalil, W. And Dombre, E. (2007) Modeling And Identification Of Serial Robots, In Modeling,. Performance Analysis And Control Of Robot Manipulators (eds E. Dombre And W. Khalil), ISTE, London, UK.

[Khatib 86] Khatib, O. Real-time obstacle avoidance for manipulators and mobile robots. International Journal of Robotics Research, 5:90–98, 1986.

[Kirkpatrick 83] Kirkpatrick, S. Gelatt Jr., C. D. and Vecchi, M. P. "Optimization by Simulated Annealing,". Science, Vol. 220, No. 4598, 1983, pp. 671-680.

[Koza 92] Koza, J. R. *Genetic Programming. On the Programming of Computers byMeans of Natural Selection.* Cambridge, MA: MIT Press, 1992.

[Latombe 91] Latombe ,J. C. Robot Motion Planning. Kluwer Academic Publishers, Boston, MA, 1991.

[Lee 81] Lee, D.T. and Drysdale, R.L.S.. Generalization of Voronoi diagrams in the plane. SIAM Journal on Computing 10 (1981), 73–87.

[Lin 14] Lin, H.I. "A Fast and Unified Method to Find a Minimum-. Jerk Robot Joint Trajectory Using Particle Swarm. Optimization," Journal of Intelligent & Robotic Systems, vol. 75, no. 3-4, pp. 379-392, 2014.

[Lozano-Perez 79] Lozano-Perez, T. and Wesley, M.A. (1979). An Algorithm for Planning Collision Free Paths among Polyhedral Obstacles. Communications of the ACM, 22, 560-570.

[Luh 80] Luh, J.Y.S., Walker, M.W., PAUL R., "On-line computational scheme for mechanical manipulators", *ASME, J. of Dyn. Syst., Measurements and Control*, Vol. 102, No. 2, p. 69-76, 1980.

[Luis 13] Luis E. G. M, Angelica N. L., Jose L. M. L. "A Genetic Algorithm for Optimizing Vector-based Paths of Industrial Manipulators". 11th. IEEE International Conference on Industrial Informatics. 2013.

[Lumelsky 86] Lumelsky, V. J and Stepanov, A. A. "Dynamic Path Planning for a Mobile Automation with Limited Information on the Environment", IEEE Trans. Automation Control, vol. 31, no. 11, pp. 1058-1063, Nov. 1986.

[Lumelsky 87] Lumelsky, V. J and Stepanov, A.A . Path planning strategies for point mobile automaton moving amidst unknown obstacles of arbitrary shape. *Algorithmica,* 2:403–430, 1987.

[Marco 04] Marco, D and Thomas, S. Ant Colony Optimization. The MIT Press, Massachusetts, USA, 2004.

[Masehian 06] Masehian, E, and Amin Naseri, M. R. "A Tabu Search based Approach for Online Motion Planning", IEEE Int. Conf. Industrial Tech. (2006), Mumbai, India.

[Metropolis 53] Metropolis, N. and Rosenbluth, A. and Rosenbluth, M. N and Teller, A. H and Teller, E. Equations of state calculations by fast computing machines, J. Chem. Phys., 21, 1953, 1087–1092

[Miao 08] Miao, H .Tian, Y.C. Robot path planning in dynamic environments using a simulated annealing based approach. In International Conference on Control, Automation, Robotics and Vision (ICARCV08), 17-20 December 2008, Hanoi, Vietnam.

[Nakamura 86] Nakamura, Y, Hanafusa, H (1986) Inverse kinematic solutions with singularity robustness for robot manipulator control. Journal of dynamic systems, measurement, and control 108(3): 163–171.

[Nilsson 69] Nilsson, N.J. A mobile automaton: An application of artificial intelligence techniques. Proc. International Joint Conference on Artificial Intelligence, 1969, pp. 509-520.

[Nora 99] Nora, H. S and Nadine, T.G, (1999). Exact Cell Decomposition of Arrangements used for Path Planning in Robotics. Swiss Federal Institute of Technology. https://doi.org/10.3929/ethz-a-006653440

[O'Dunlaing 85] O'Dunlaing, C. and Yap, C. K., A retraction method for planning the motion of a disc, J. Algorithms, Vol. 6, 1985, pp. 104-111.

[O'Dunlaing 87] O'Dunlaing, C. Sharir, M. and Yap, C. K. Generalized Voronoi diagrams for a ladder: II. Efficient construction of the diagram. Algorithmica, 2:27-59, 1987.

[Panda 16] Panda, M.R. Priyadarshini, R. Pradhan, S.K . Autonomous mobile robot path planning using hybridization of particle swarm optimization and Tabu search. IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), 2016

[Paul 81] Paul R.C.P., *Robot Manipulators: Mathematics, Programming and Control*, MITPress, Cambridge, 1981.

[Peng 06] Peng, Y. Wei, W. A New Trajectory Planning Method of Redundant Manipulator Based on Adaptive Simulated Annealing Genetic Algorithm (ASAGA). Proc. of the IEEE Int. Conf. on Computational Intelligence and Security 2006; 262 – 265.

[Petric 15] Petric, T. Gams, A. Likar, *N.*and Zlajpah, L. "Obstacle avoidance with industrial robots," *Mechanisms and Machine Science*, vol. 29, pp. 113–145, 2015.

[Piegl 95] Piegl, L and Tiller, W. The NURBS Book, Springer-Verlag, Berlin Heidelberg, 1995.

[Procházková 06] Procházková, J. Procházka, D. "The application of NURBS surfaces in engineering practice," Proceedings of conference, Modern mathematical method in engineering, 2006.

[Rechenberg 65] Rechenberg, I. (1965). Cybernetic solution path of an experimental problem. Royal Aircraft Establishment, library translation 1122, Farnborough, Hants, U.K.

[Reynolds 87] Reynolds, C.W Flocks, Herds and Schools: A Distributed Behavioral Model, Computer Graphics, 21(4), pp. 25-34, 1987.

[Shamos 75] Shamos, M. I and Hoey, D. Closest-point problems. In 16th Annual Symposium on foundations of Computer Science, p 151-162, Berkeley, California, October 1975, IEEE Computer Society Press.

[Shi 98] Shi, Y. and Eberhart, R. (1998) A Modified Particle Swarm Optimizer. IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence, 69-73.

[Siciliano 09] Siciliano, B. Sciavicco, L. Villani, L and Oriolo, G. Robotics - Modelling, Planning and Control. *Advanced Textbooks in Control and Signal Processing series.* Springer, 2009.

[Taylor 79] Taylor, Russell H., "Planning and Execution of Straight Line Manipulator Trajectories," *IBM J. Res. Develop*. 23, No. 4, July 1979, 424-436.

[Tran 13] Tran, N. Nguyen, D.T. Vu, D. L. and Truong, N.V. "Global path planning for autonomous robots using modified visibility-graph," in Proceedings of the 2nd International Conference on Control, Automation and Information Sciences (ICCAIS '13), pp. 317–321, NhaTrang, Vietnam, November 2013.

[Udupa 77] Udupa, S. Collision detection and avoidance in computer controlled manipulators. Ph.D. Th., Calif. Inst. of Technology, Pasadena, Calif., 1977.

[Vaccaro 88] Vaccaro, R.J. and Hill, S.D., "A joint-space command generator for Cartesian control of robotic manipulators*" IEEE J. Robotics and Automation RA-4,* No. 1, 70–76 (1988).

[Verdonck 04] Verdonck, W. "Experimental robot and payload identification with application to dynamic trajectory compensation." PhD Thesis, Department of Mechanical Engineering, KU Leuven, Belgium, 2004.

[Victor 98] Victor, D. L. C, Fernando. R. Cooperative genetic algorithms: a new approach to solve the path planning problem for cooperative robotic manipulators sharing the same work space. Proceedings of IEEE Int Conference on Intelligent Robotics and Systems. New York:IEEE,1998:267-272.

[Warren 89] Warren, C. W. "Global path planning using artificial potential fields," in. Proc. IEEE Conf. Robotics and Automation, 1989, pp. 316–321.

[Wen 09] Wen, Z. Luo, J. Li, Z. On the global optimum motion planning for dynamic coupling robotic manipulators using particle swarm optimization technique, Proceedings of the 2009 international conference on Robotics and biomimetics, December 19-23, 2009, Guilin, China.

[Whitney 69] Whitney D.E., "Resolved motion rate control of manipulators and human prostheses", *IEEE Trans. on Man Machine Systems*, Vol. MMS-10(2), June 1969, p. 47-53.

[Wilson 75] Wilson, E. O. Sociobiology: the new synthesis, Belknap Press,. Cambridge, MA, 1975.

[Wolovich 84] Wolovich W, Elliot H. (1984). A computational technique for inverse kinematics. *In: Proc. 23rd Conf. on Decision and Control,* Las Vegas 1359–1363

[Yueqiang 14] Yueqiang,Y. Zhiwei, T. The Study of Path Planning of Welding Manipulator Based on Improved QPSO. 2014 Sixth International Conference on Measuring Technology and Mechatronics Automation. IEEE, 2014.p. 774-777. Hubei, China

[Zhao 16] Zhao, J.D. Zhao, L.L. Liu, H. Motion Planning of Hyper-Redundant Manipulators Based on Ant Colony Optimization. Proceedings of the 2016 IEEE. International Conference on Robotics and Biomimetics. Qingdao, China, December 3-7, 2016

Annexe

1. Commande par couple calculé dans l'espace articulaire

La loi de commande utilisée est basée sur l'emploi des trois gains : proportionnel, intégrale et dérivé pour permettre au robot manipulateur de suivi les positions articulaires désirées dans trois environnements différents. Le schéma bloc de cette loi de commande est monté ci-dessous.

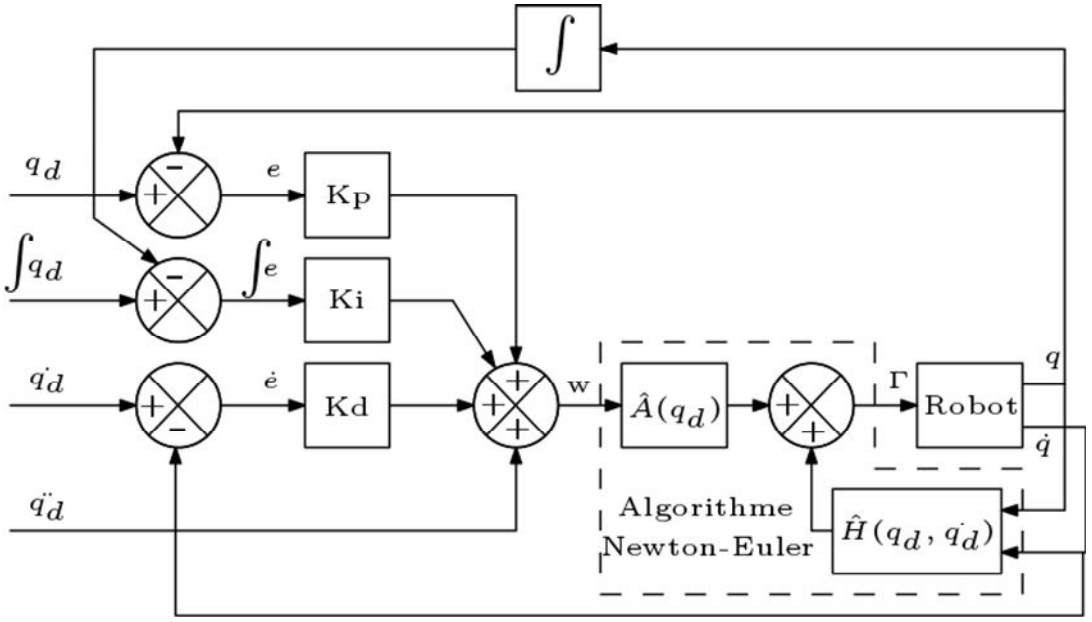


Figure A.1. Schéma bloc de la commande par couple calculé dans l'espace articulaire

Dans les trois situations les algorithmes génétiques sont employés pour l'optimisation des gains Kp, Ki et Kd.

1.1. Commande du robot dans un environnement libre

Afin d'évaluer les performances des algorithmes génétiques, les erreurs entre les positions calculées et désirées sont considérées comme fonction coût. Le résultat de simulation présenté dans la figure 2 montre la convergence de la meilleure solution ainsi que la moyenne de l'ensemble des solutions, en fonction du nombre de génération, vers la solution optimale.

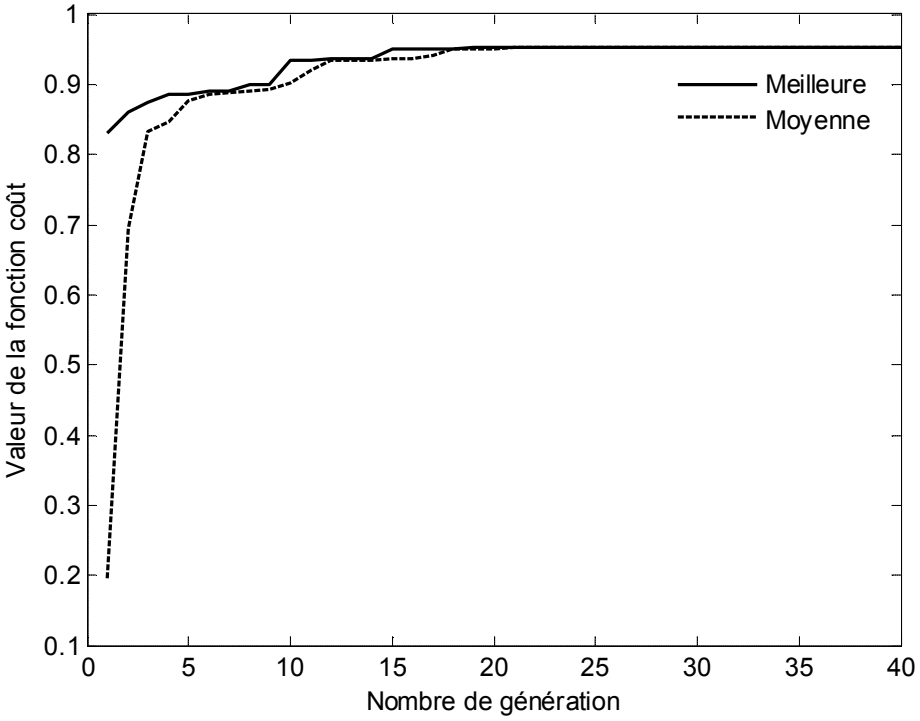


Figure A.2. Convergence des algorithmes génétiques dans un environnement sans obstacle

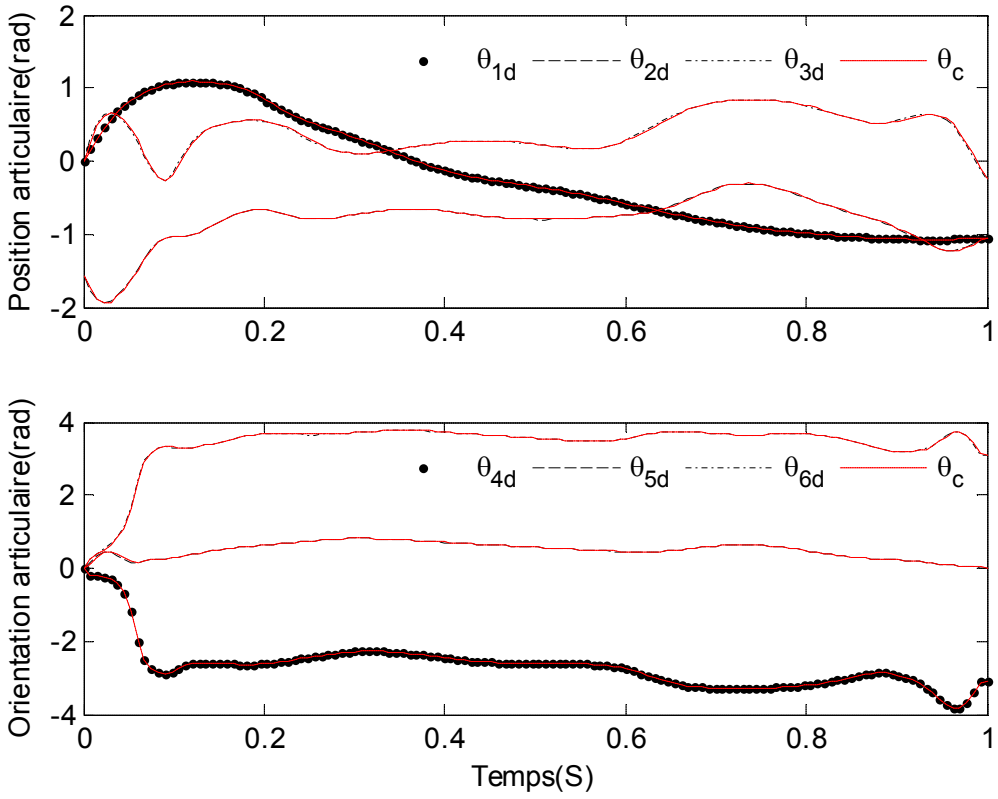


Figure A.3. Poursuite des positions articulaires désirées dans un environnement sans obstacle

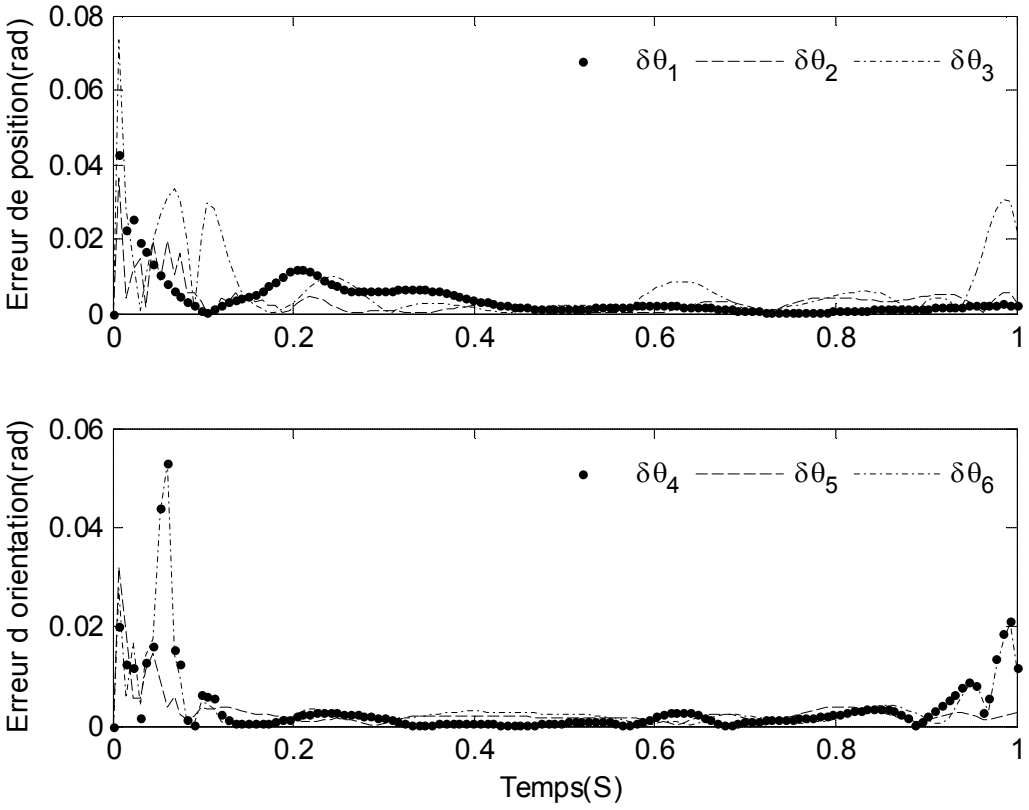


Figure A.4. Erreurs de poursuite dans un environnement sans obstacle

1.2. Commande du robot dans un environnement avec un obstacle

Les positions articulaires obtenues par l'utilisation des algorithmes génétiques dans la phase de planification des trajectoires sont considérées comme des consignes dans la boucle de commande. Le rôle des algorithmes génétiques dans la phase de commande est de trouver le vecteur optimale des gains qui assure la moindre erreur entre les positions désirées et calculées.

Après exécution des algorithmes génétiques, leurs performances sont évaluées en termes de convergence de la meilleure solution et de la moyenne des solutions vers la solution optimale (figure 5). Ensuite, la solution optimale est utilisée pour évaluer le degré de poursuite des positions calculées à celles désirées (figure 6) ainsi que les erreurs entre les deux (figure 7).

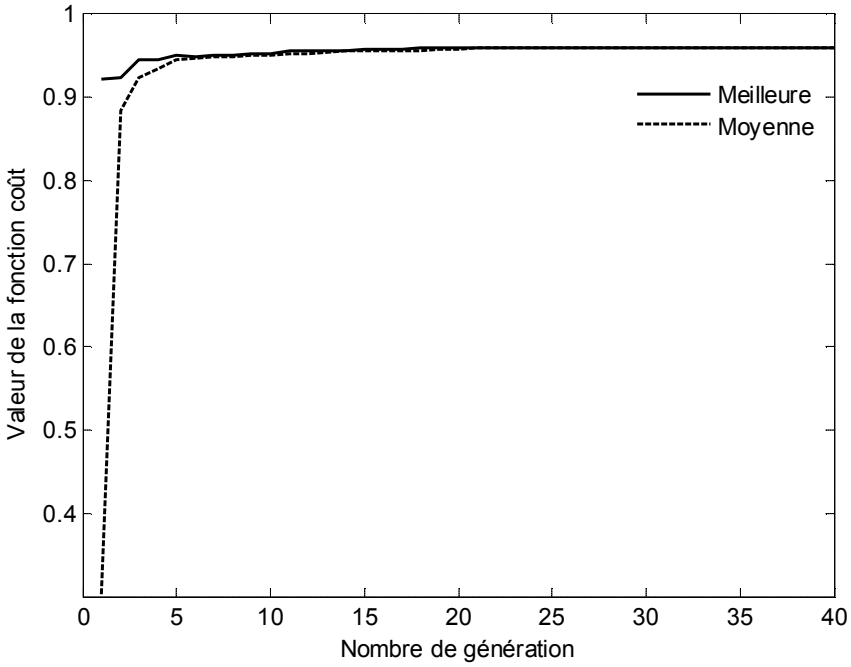


Figure A.5. Convergence des algorithmes génétiques dans un environnement avec un obstacle

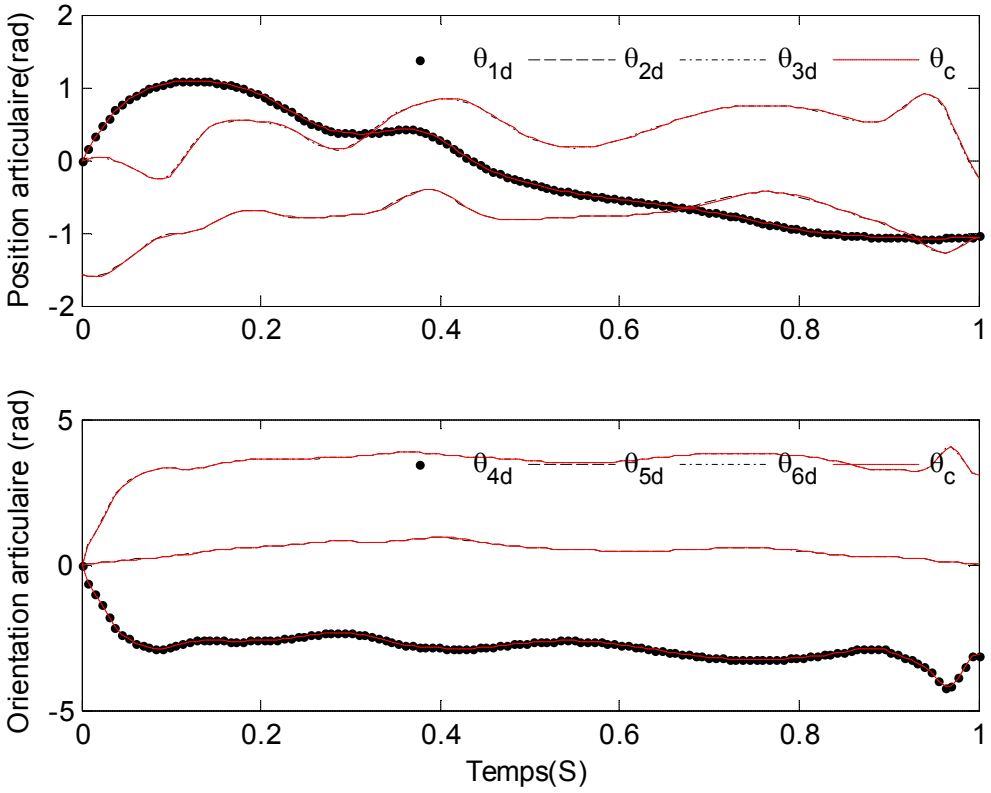


Figure A.6. Poursuite des positions articulaires désirées dans un environnement avec un obstacle

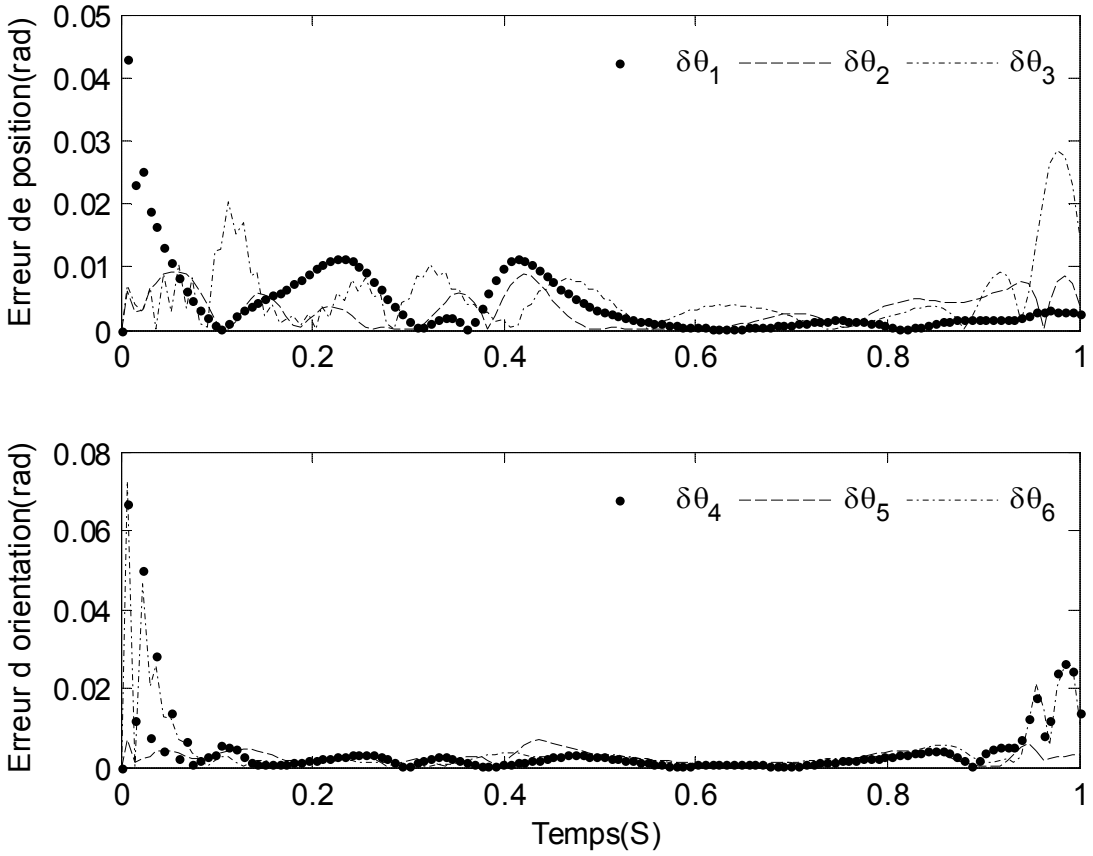


Figure A.7. Erreurs de poursuite dans un environnement avec un obstacle

1.3. Commande du robot dans un environnement avec deux obstacles

Pour commander le robot dans un environnement qui inclut deux obstacles, la boucle de commande reçoit les consignes qui sont les positions articulaires générés dans la phase de planification des trajectoires. Ensuite, les algorithmes génétiques sont employés pour rechercher le vecteur optimale des gains qui minimise la fonction d'erreur entre les consignes et les positions articulaires calculées.

L'évaluation des performances des algorithmes génétiques pour la résolution du problème de commande du robot dans un environnement avec deux obstacles est montrée en termes de convergence de la meilleure solution et de la moyenne des solutions vers la solution optimale (figure 8). Cette dernière est employée pour juger la fiabilité de la commande en termes de poursuite des consignes (figure 9) et des erreurs entre les consignes et les positions calculées (figure 10).

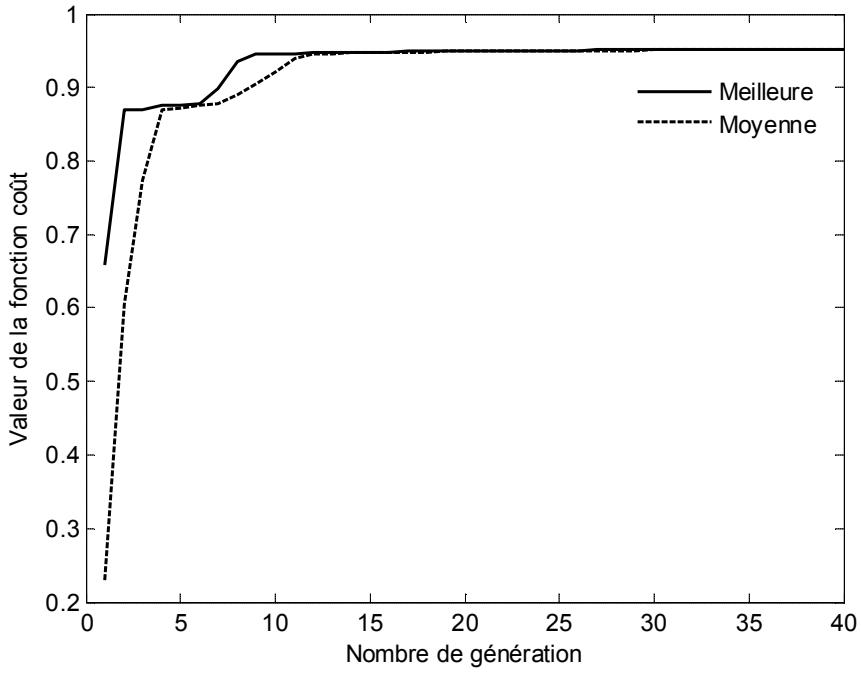


Figure A.8. Convergence des algorithmes génétiques dans un environnement avec deux obstacles

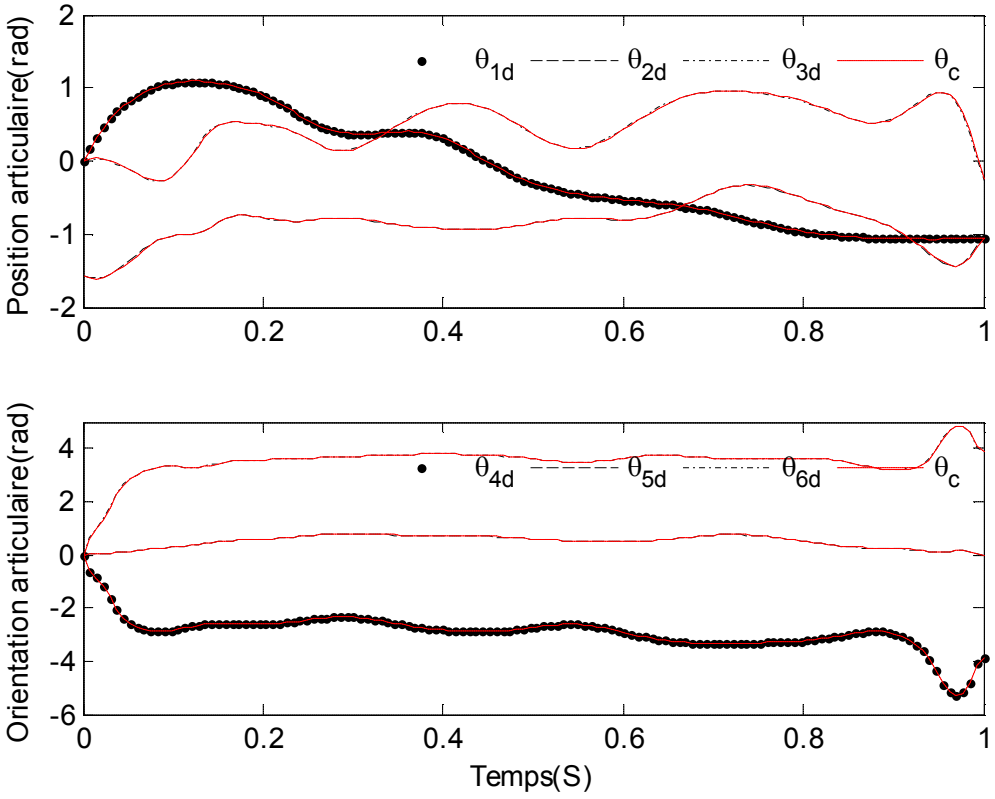


Figure A.9. Poursuite des positions articulaires désirées dans un environnement avec deux obstacles.

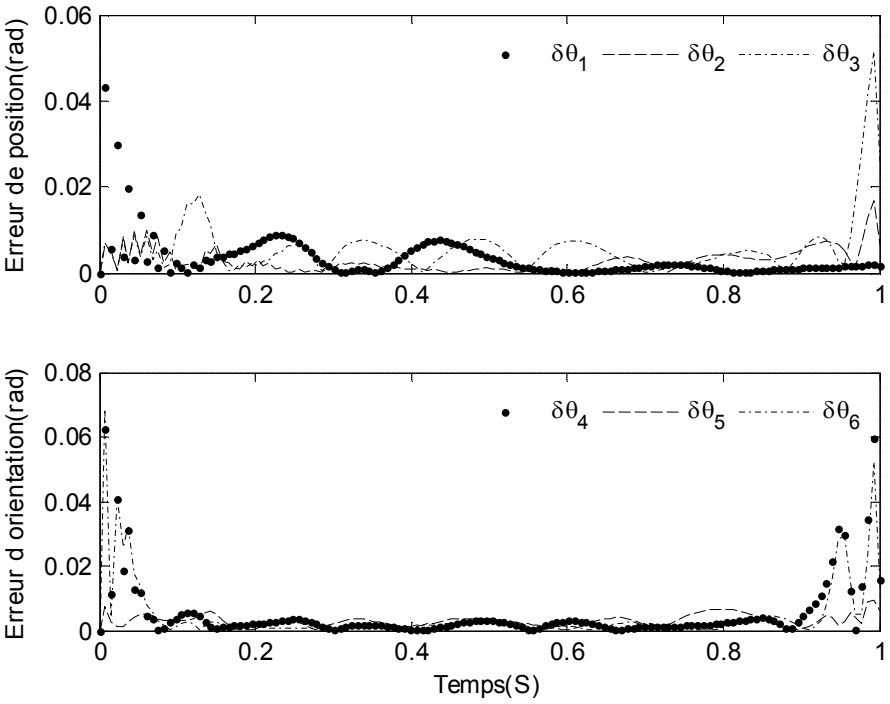


Figure A.10. Erreurs de poursuite dans un environnement avec deux obstacles.