FACULTÉ DES SCIENCES ET DE GÉNIE

Département de génie mécanique

PLANIFICATION DE TRAJECTOIRE D'UN MANIPULATEUR SÉRIEL REDONDANT DANS UN ENVIRONNEMENT ENCOMBRÉ

THIERRY LALIBERTÉ

Mémoire
présenté
pour l'obtention
du grade de maître ès sciences (M.Sc.)

FACULTÉ DES ÉTUDES SUPÉRIEURES
UNIVERSITÉ LAVAL

MARS 1994

Résumé

Ce mémoire présente une stratégie de planification de trajectoire d'un manipulateur sériel redondant dans un environnement encombré. Développée dans un contexte pratique de télémanipulation, elle est adaptée aux applications en temps réel avec interaction humaine. Elle utilise un modèle de l'environnement fourni par un système de vision. Un processus incrémentiel guide l'effecteur vers son objectif à l'aide de champs de potentiel sans minimums locaux. Le temps de calcul de ces champs de potentiel est réduit en éliminant les expansions inutiles. À chaque incrément, le mouvement global du manipulateur est calculé par l'inversion en vitesse exploitant la redondance pour optimiser la distance aux obstacles sans nuire à la trajectoire de l'organe terminal. Un algorithme tient compte des limites articulaires et s'assure d'éviter toute collision. Des heuristiques favorisent la résolution de cas types complexes, afin d'augmenter le taux de réussite. Un simulateur a été construit pour vérifier le comportement des algorithmes.

Thierry Laliberté
Étudiant

Clément M. Gosselin
Directeur de recherche

Avant-Propos

Je tiens à remercier mon directeur de recherche, Monsieur Clément Gosselin pour son dévouement, sa grande disponibilité et les connaissances qu'il m'a apportées. Je remercie aussi mes proches de m'avoir encouragé et aidé tout au long de cette maîtrise.

Je tiens à souligner le travail Monsieur Denis Laurendeau et Monsieur Marc Richard qui ont bien voulu examiner ce mémoire. Je souligne aussi le travail de Chantale qui a grandement amélioré la qualité du texte.

Je remercie finalement le Conseil de Recherches en Sciences Naturelles et en Génie (CRSNG) et l'Institut de Recherche d'Hydro-Québec (IREQ) pour leur support financier.

Ces différents apports on aidé à faire de cette maîtrise une expérience enrichissante et très appréciée.

Table des matières

Liste des Figures

Liste des Tableaux

Chapitre 1

Introduction

1.1 Problématique

Pour des raisons de sécurité, Hydro-Québec veut utiliser un télémanipulateur [24], pour effectuer des opérations d'entretien de son réseau de distribution sur des lignes sous tension, tel que le remplacement d'isolateurs en céramique. Un environnement de travail typique est illustré à la figure 1.1. Les lignes sont gardées sous tension, car l'interruption du service priverait les usagers d'électricité pendant une très longue période de temps (jours), ce qui est inacceptable. Le concept du télémanipulateur est le suivant : l'opérateur, installé dans une cabine de sécurité fermée et électriquement isolée, dirige deux manipulateurs (des bras Sarcos) avec glissières à la base. Le bras Sarcos, présenté à l'annexe A, est un manipulateur à 7 degrés de liberté ayant la morphologie du bras humain. Avec une articulation prismatique à sa base, un 8^{ieme} degré de liberté devient disponible. Le manipulateur est donc redondant par rapport à des tâches conventionnelles à 3 ou 6 degrés de liberté.

Étant donné que les robots ajoutent un élément de travail supplémentaire et augmentent le niveau de difficulté de la tâche, il est intéressant d'automatiser le plus possible cet élément pour faciliter et rendre plus sécuritaire le travail de l'opérateur. Les bras sont présentement commandés à l'aide de pantins. Une structure attachée au bras de l'opérateur interprète ses mouvements et les transmet au contrôleur du manipulateur pour être exécutés. Ce système fonctionne bien en laboratoire, mais avec

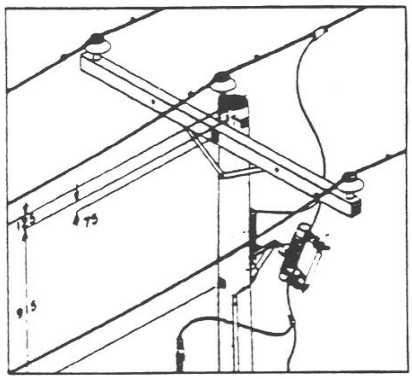


Figure 1.1: Environnement de travail typique.

le temps, il devient encombrant et fatigant pour le travailleur, malgré la compensation de la gravité. L'utilisation de manettes (joysticks) est donc préférée. La commande en mode articulaire, telle que généralement utilisée pour une pelle mécanique (4 degrés de liberté) est très difficile pour un robot à 8 degrés de liberté. Il faut donc permettre à l'opérateur de commander les robots à un niveau supérieur à celui de la simple commande articulaire. Par exemple, l'opérateur demande au robot d'amener son effecteur près d'une céramique sans produire de collisions. L'étude du sujet a permis de découvrir un autre mode de commande intéressant, soit la commande partagée (ou commande Cartésienne). L'opérateur dirige l'organe terminal et le système se charge du mouvement du reste du manipulateur.

Ce paragraphe énumère les conditions particulières à rencontrer. Il faut d'abord empêcher la collision du robot avec les obstacles pour éviter un bris ou une décharge électrique. Il est donc demandé au système de s'assurer de l'évitement de collision avec les obstacles. L'environnement, qui n'est pas connu à l'avance, peut varier d'une trajectoire à l'autre, mais demeure fixe pour une trajectoire donnée. Les calculs à effectuer pour l'automatisation doivent donc être effectués en temps réel. Ainsi, le calcul de la trajectoire doit se faire rapidement pour être utile sinon, l'opérateur a avantage à l'exécuter lui-même. Puisqu'il y a toujours un opérateur, il n'est pas nécessaire de déterminer automatiquement toutes les tâches théoriquement réalisables. On doit cependant obtenir un bon taux de réussite pour justifier l'utilité de l'algorithme. Pour

être utilisable, la trajectoire générée doit éviter les problèmes dus aux singularités, conserver une distance aux obstacles acceptable, se révéler assez douce pour tenir compte de l'inertie du robot et ne pas faire de détours inutiles. La dynamique du manipulateur n'est pas explicitement étudiée. Cependant, on cherche à limiter les accélérations. L'opérateur ne demande que la pose de l'organe terminal. Il n'a pas à fournir la configuration finale, ce qui serait inutilement exigeant. L'algorithme doit donc s'en occuper. Les algorithmes doivent être rapides pour des cas simples et capables de résoudre des cas plus difficiles, même si le temps de calcul s'avère plus long.

Pour effectuer ces opérations, l'algorithme doit connaître son environnement. Pour ce faire, un système de vision 3D permet d'obtenir un modèle de l'espace de travail qui est utilisé lors de l'automatisation de tâches [2]. Le système de planification de trajectoire doit donc être compatible avec le modèle de l'environnement fourni par le système de vision 3D. Celui-ci est disponible sous deux formes, soit l'*octree* [17] [9] et la grille d'occupation. L'environnement pouvant comporter des obstacles quelconques, la description de la scène a avantage à être générale.

En pratique, l'application consiste à automatiser la phase d'approche des bras Sarcos vers le lieu de travail voulu. Dans un contexte plus général, le système doit effectuer la planification de trajectoire d'un manipulateur redondant dans un environnement encombré d'une configuration de départ à une position (et orientation) demandée de l'organe terminal, en évitant toute collision. Dans ce contexte, il est intéressant de généraliser la stratégie à un manipulateur sériel quelconque, surtout qu'il n'est pas assuré que le bras Sarcos soit le manipulateur finalement utilisé.

1.2 Méthodologie utilisée

L'approche utilisée est la suivante :

1. La problématique est établie.

2. Une revue de la littérature portant sur la planification de trajectoire dans un environnement encombré et sur les manipulateurs redondants est effectuée.

3. La stratégie la plus intéressante est choisie selon les critères établis.

4. Les algorithmes de la stratégie sélectionnée sont implantés et évalués.

5. Autour des algorithmes de base de la stratégie choisie, des algorithmes sont remplacés, modifiés et ajoutés pour être adaptés à la situation et améliorer les performances.

Pour pouvoir évaluer les algorithmes, un simulateur est construit en version 2D et 3D. Les algorithmes sont d'abord implantés en 2D, puis en 3D si les résultats sont positifs. On procède ainsi étant donné qu'il est plus simple d'implanter et d'évaluer les algorithmes en 2D. Ce simulateur permet de visualiser la trajectoire du manipulateur dans son environnement et la trajectoire des coordonnées articulaires.

1.3 Plan du mémoire

Au chapitre 2, ce qui est proposé dans la littérature est considéré. La stratégie générale étant introduite dès le chapitre 3, le lecteur pourra situer les éléments présentés ultérieurement dans leur contexte. Par la suite, le chapitre 4 présente quelques éléments théoriques sur la robotique, dont un élément clé, la cinématique inverse d'un manipulateur redondant. Au chapitre 5, le deuxième concept clé, soit les champs de potentiel discrets, est présenté en deux sections. La première section explique le concept proposé par [4] et la deuxième discute des modifications apportées pour adapter et améliorer le concept. Le chapitre 6 contient la description détaillée de la stratégie de planification de trajectoire. Des concepts originaux, permettant un bon fonctionnement du système, s'y retrouvent. Le chapitre 7 décrit le simulateur permettant d'observer le comportement de la stratégie. De plus, des exemples de résultats et les performances du système y sont discutés. Finalement, la conclusion résume les grandes lignes de ce mémoire et ouvre la porte à de futurs travaux.

Chapitre 2

Revue de la littérature

2.1 Méthodologie

Cette revue de la littérature se veut représentative, mais pas forcément exhaustive. Le but de cet exercice est d'obtenir des algorithmes utilisables et répondant aux besoins demandés, et de connaître les principes généraux de la planification de trajectoire. Les articles choisis sont préférablement récents (5 dernières années). On considère que les articles pertinents publiés avant cette période ont déjà été utilisés pour les travaux récents. Ces articles se retrouvent donc en réference dans les articles plus récents sélectionnés.

La recherche effectuée s'est concentrée principalement au niveau des sources de renseignements suivantes: Compendex 86-91 (banque informatisée de Engineering Index), Engineering Index 92, IEEE Transactions on Robotics and Automation, The International Journal of Robotics Research et divers livres.

La revue de la littérature a principalement porté sur la planification de trajectoire dans un environnement encombré, puisque cet algorithme constitue la base du système. On a aussi mis l'accent sur les algorithmes tenant compte de la redondance, puisque le manipulateur utilisé est redondant.

La résolution de la cinématique inverse d'un robot redondant a ultérieurement été étudiée de façon détaillée, suite aux premiers résultats qui ont démontré son utilité.

2.2 Stratégies proposées dans la littérature

Cette section présente les principales stratégies de planification de trajectoire retrouvées dans la littérature.

Barraquand et Latombe [3] [4] : La statégie est adaptée pour des robots très redondants. Elle a un caractère local dans l'espace des configurations, celui-ci n'étant exploré que près de la trajectoire. L'espace de travail et l'espace des configurations sont discrétisés. Les obstacles sont modélisés par une grille d'occupation. Le robot est modélisé par des points de contrôle influencés par des champs de potentiel agissant dans l'espace de travail. Chaque point de contrôle est poussé par un champ de potentiel vers un objectif qui correspond à la position de ce point lorsque le robot est à la configuration finale demandée. Pour diriger tout le robot, on combine les potentiels des points de contrôle dans l'espace des configurations, ce qui dicte le mouvement de chacune des articulations. Ces champs de potentiel attractifs, construits à l'aide d'une méthode d'expansion par vague, n'ont pas de minimums locaux, diminuant de beaucoup les chances de rencontrer ces minimums locaux dans l'espace des configurations. Pour éviter des temps de calcul trop longs, on ne calcule le potentiel combiné que pour un sous-ensemble des voisins de la configuration présente choisis de manière aléatoire. Lorsque l'algorithme tombe dans un minimum local, il utilise un mouvement Brownien (aléatoire) pour en sortir. Cette partie de l'algorithme peut être plus longue en temps de calcul, mais elle permet de sortir tôt ou tard du minimum local. Tout ceci ne garantit pas l'absence de collisions. Pour s'assurer de l'évitement des obstacles, on modélise le robot par des points de vérification. Ces points de vérification sont localisés par une table de distances aux obstacles, construite par la méthode d'expansion par vague. Si la distance entre deux points voisins est plus grande que la distance à un obstacle de l'un de ces deux points, il y a risque de collision et une nouvelle configuration est suggérée. Le chemin initialement calculé comporte des changements de direction brusques et est inutilement trop long. Une méthode est présentée pour modifier la trajectoire et éliminer ces problèmes.

Faverjon et Tournassoud [11] [12] [13] : Les articles décrivent trois algorithmes de planification de trajectoire. Un premier est local, un deuxième est global, et le dernier représente la combinaison des deux premiers.

Éléments de la partie locale : Les obstacles et le robot dans l'espace de travail sont modélisés par des formes primitives (cubes, cylindres, cônes, sphères, etc.). La représentation est hiérarchique, c'est-à-dire que des primitives de plus en plus complexes sont utilisées seulement lorsque nécessaires, permettant ainsi de diminuer les temps de calcul. La tâche, exprimée par la minimisation d'une fonction, est réalisée récursivement. Cette fonction peut être la minimisation de la distance entre les angles des articulations et les angles objectifs correspondants. L'algorithme est basé sur un calcul rapide de la distance minimale entre deux solides, que l'on peut dériver par rapport au temps (il est donc possible de connaître la vitesse d'approche). La fonction d'évitement d'obstacles voit à respecter une vitesse d'approche entre deux objets à ne pas dépasser, selon la distance entre ceux-ci, pour éviter toute collision. Ceci permet à deux primitives de s'approcher l'une de l'autre jusqu'à une distance limite arbitraire, sans collision.

Éléments de la partie globale utilisée seule : La stratégie travaille dans l'espace des configurations, discrétisé sous forme de 2^n *tree* (*octree* si n=3). Elle utilise un modèle hierarchique du manipulateur pour permettre un calcul plus rapide des obstacles dans l'espace des configurations. Une méthode de recherche "A*" (voir [23]) permet de trouver un chemin. La stratégie globale utilisée seule est très longue en temps de calcul et ne peut raisonnablement travailler avec plus de 3 ou 4 degrés de liberté.

Méthode mixte : La méthode globale fournit des objectifs intermédiaires à la méthode locale lorsque celle-ci tombe dans des minimums locaux. La méthode globale ne tient plus compte des obstacles : c'est la méthode locale seulement qui se charge de la vérification des collisions. La méthode globale utilise un graphe d'états et ne calcule pas les obstacles dans l'espace des configurations. Celui-ci est plutot divisé en cellules. Pour chacune d'entre elles, on évalue la probabilité de succès à déplacer le robot dans celle-ci. L'évaluation de la probabilité est réalisée par apprentissage. On doit donc procéder par essais et erreurs pour que le graphe d'états devienne efficace. Cet algorithme devient donc intéressant si le robot travaille dans un environnement

qui ne change pas.

Gupta [15] : L'algorithme est spécialement intéressant pour les manipulateurs très redondants. Le problème est abordé en traitant la trajectoire des membres séquentiellement, de la base vers l'organe terminal : on trouve une trajectoire au membre le plus près de la base; puis, à partir de la trajectoire générée par ce premier membre, on génère une trajectoire pour le deuxième membre et ainsi de suite. Ainsi, au lieu de travailler un problème dans l'espace des configurations en N dimensions (N est le nombre de degrés de liberté), on traite N-1 problèmes en 2 dimensions de l'espace des configurations. L'approche présentée ici conduit à un temps de calcul linéaire en fonction de N au lieu d'exponentiel en fonction de N pour une recherche dans tout l'espace de travail. Les trajectoires sont planifiées en deux dimensions dans l'espace $t \times \theta$, où t est un paramètre en fonction duquel la trajectoire des i-1 premiers membres est définie et θ est la configuration du i^{eme} membre. Une fois le processus appliqué à tous les membres, on obtient une trajectoire en fonction d'un seul paramètre, soit t. Pour planifier la trajectoire dans cet espace $t \times \theta$, un graphe de visibilité, décrit dans [18], est utilisé. Les obstacles de ces espaces en deux dimensions doivent être calculés à partir de l'espace de travail (méthode utilisée très semblable à celle présentée dans [25]). La configuration finale doit être déterminée à priori. L'algorithme peut manquer certains chemins à cause de son caractère local.

Lozano-Pérez [25] : L'article décrit une stratégie travaillant dans l'espace des configurations à N dimensions où N est le nombre de degrés de liberté du manipulateur. Le robot et les obstacles sont modélisés par des volumes polyhédriques dans l'espace de travail. La construction de l'espace des configurations est basée sur le calcul des angles des articulations au contact des volumes polyhédriques. Le calcul de l'espace des configurations pouvant être relativement long, l'auteur suggère de ne travailler qu'avec les trois premiers membres du manipulateur si le manipulateur est découplable et que les dimensions de l'organe terminal sont négligeables. La position et l'orientation de l'organe terminal ne sont traitées en détail qu'au départ et à l'arrivée. Une fois les obstacles calculés dans l'espace des configurations, ce qui n'est

pas un obstacle constitue un espace légal pour le mouvement du robot. Cet espace est divisé en régions. Pour calculer une trajectoire, on voyage d'une région à l'autre à l'aide d'une recherche "A*" (voir [23]). Le temps de calcul augmente très rapidement avec le nombre de degrés de liberté du manipulateur.

Khatib [19] : L'article décrit un algorithme basé sur le concept de champ de potentiel. L'algorithme travaille dans l'espace de travail Cartésien. Le manipulateur bouge dans une combinaison de champs de potentiel. Un champ de potentiel attractif attire le manipulateur vers son objectif et des champs de potentiel répulsifs éloignent le manipulateur des obstacles. Les obstacles sont représentés par des primitives sous forme de n-ellipsoïdes. Selon les valeurs des coefficients, les primitives peuvent s'approcher de la forme d'un ellipsoïde, d'un parallépipède, d'un cylindre ou d'un cône. Il est très facile de créer un champ potentiel répulsif à partir d'une telle modélisation. Un potentiel répulsif est utilisé pour éviter d'atteindre les limites des articulations. L'algorithme est rapide, mais il peut facilement tomber dans des minimums locaux.

Kondo [22] : L'article décrit un algorithme qui calcule un chemin pour un robot à 6 degrés de liberté parmi des obstacles fixes. Les configurations initiale et finale sont données. L'espace des configurations est divisé en hypercubes de dimensions égales. L'idée de base de l'algorithme vise à vérifier s'il y a collision, c'est-à-dire de calculer s'il y a un obstacle dans l'espace des configurations, seulement pour une très faible partie de l'espace des configurations et ainsi diminuer considérablement le temps de calcul. La méthode a donc un aspect local. L'algorithme fonctionne comme suit. À partir d'une configuration, on regarde ses voisins immédiats. Pour chacun de ces voisins, on vérifie s'il y a collision et on évalue une fonction heuristique. La configuration suivante du chemin sera l'hypercube sans collision qui sera le plus prometteur selon l'heuristique. La configuration de départ peut être la configuration initiale ou finale ou les deux à la fois. La vitesse de calcul dépend beaucoup de la qualité de l'heuristique. Malheureusement, on doit connaître tout l'espace des configurations

pour connaître la qualité de celui-ci. Un compromis permettant d'améliorer les performances consiste à essayer plusieurs heuristiques à la fois et à mettre l'accent sur celles qui donnent les meilleurs résultats à mesure que la recherche du chemin se réalise. Selon l'auteur, les heuristiques les plus efficaces sont à déterminer expérimentalement.

Kim et Khosla [20] : L'organe terminal est dirigé par un point de contrôle primaire dans un champ potentiel attractif harmonique. Celui-ci n'a pas de minimums locaux et est basé sur des principes de la mécanique des fluides. Le reste du robot est dirigé par des points de contrôle secondaires qui tiennent compte de la distance aux obstacles. La partie secondaire de contrôle n'influence pas la partie primaire. L'algorithme a des minimums locaux même si le champ potentiel attractif n'en a pas, parce que l'on dirige un manipulateur. Les obstacles sont modélisés par des "panels", c'est-a-dire une série de panneaux, chacun d'eux créant une répulsion.

Warren et al. [33] [34] : L'article suggère des améliorations au calcul des obstacles dans l'espace des configurations et à la planification de trajectoire. L'algorithme travaille dans l'espace des configurations. Une méthode de calcul de l'espace des configurations basée sur des points à la surface des obstacles est brièvement présentée. On y fait les hypothèses suivantes: le robot est réduit à des lignes et les obstacles sont des polygones convexes. Pour la planification d'un chemin, on trace une trajectoire directe entre les points de départ et d'arrivée sans tenir compte des obstacles, dans l'espace des configurations. Puis, à l'aide de champs de potentiel répulsifs, on modifie cette trajectoire pour la sortir et l'éloigner des obstacles. Cette méthode a pour effet de diminuer les possibilités de minimums locaux. Pour empêcher la génération d'une trop longue trajectoire, on cherche aussi à minimiser la longueur du trajet. Etant donné que l'on doit calculer l'espace des configurations, on ne peut travailler qu'avec peu de degrés de liberté (environ 3) pour avoir des temps de calcul acceptables.

Dupont et Derby [8] [9] [10] : La stratégie fonctionne en deux étapes et travaille dans l'espace des configurations. D'abord, elle trace un chemin entre la configuration initiale et la position finale. Ce chemin peut passer dans des obstacles. Ensuite,

elle modifie le chemin pour qu'il contourne les obstacles à l'aide d'heuristiques basées sur l'espace de travail. L'espace des configurations n'est explicitement modélisé que lorsque l'algorithme en a besoin, ce qui réduit beaucoup les temps de calcul.

2.3 Considérations générales

La planification de trajectoire peut être divisée en deux grandes approches : locale et globale. L'approche locale n'utilise que l'environnement au voisinage du robot pour modifier *localement* sa trajectoire, selon les nouvelles informations recueillies sur les obstacles, tout en se dirigeant préférablement vers l'objectif final. Cette approche est rapide en temps de calcul puisqu'elle repose sur l'analyse d'une partie seulement des informations recueillies sur l'environnement. Cependant, elle ne garantit pas de trouver une trajectoire réalisable, même si une solution existe théoriquement. A l'opposé, l'approche globale tient compte de toute l'information connue sur l'environnement pour calculer *globalement* une trajectoire. Cette approche est très lourde en temps de calcul, mais elle garantit la génération d'une trajectoire faisable, s'il en existe une.

Les principales représentations des robots sont les volumes polyhédriques [25], les points de contrôle [4], la combinaison de formes primitives [17] et l'*octree* [9].

Les principales représentations des obstacles dans l'espace de travail sont l'*octree* [17] [9], la grille d'occupation [4], les ellipsoïdes [19], les volumes polyhédriques [25], la combinaison de formes primitives [12] et les "panels" [20].

À sa configuration de départ, le manipulateur est complètement décrit. Cependant, l'objectif final peut être la configuration finale ou seulement la pose finale de l'organe terminal. Le deuxième objectif a l'avantage d'être moins exigeant pour l'utilisateur et d'augmenter les chances de succès de l'algorithme.

La description et l'utilisation de l'espace des configurations peut être complète (approche globale) [25], aux environs de la trajectoire (approche locale) [22] ou par traitement séquentiel des articulations [15].

Pour les chemins générés par des champs de potentiels, deux approches sont rencontrées. La première génère progressivement un chemin du point de départ au point d'arrivée. La seconde trace un chemin hypothétique initial qui est ensuite modifié.

Un livre de Latombe [23] donne une synthèse intéressante de ce qui existe en planification de trajectoire.

2.3.1 Éléments d'un planificateur de trajectoire

La revue de la littérature a permis d'identifier les principaux éléments qui composent un planificateur de trajectoire.

- L'acquisition des données initiales.
  - Le modèle des obstacles dans l'espace de travail.
  - Le modèle du manipulateur.
    - Paramètres Hartenberg-Denavit.
    - Modèle géométrique compatible avec le modèle des obstacles.
  - La description du manipulateur aux états initial et final.
- Calcul d'un espace de recherche à partir des données initiales (l'espace des configurations du robot, un champ de potentiel, etc).
- La recherche d'un chemin sans collision. Le chemin crée un lien entre les points de départ et d'arrivée, mais il n'est pas utilisable.
- La recherche d'une trajectoire réalisable par un manipulateur réel. Celle-ci est continue et tient compte de critères tel que le temps, la longueur et la distance aux obstacles le long du parcours. La trajectoire peut être produite directement, provenir d'un post-traitement du chemin établi ou être une combinaison des deux.

2.4 Choix de la stratégie

Tout d'abord, on constate qu'une méthode totalement globale est inadéquate pour les manipulateurs redondants, car leur complexité augmente exponentiellement avec

le nombre de degrés de liberté. En effet, puisque l'on doit travailler avec des manipulateurs ayant un très grand nombre de degrés de liberté (7 ou plus en admettant que la tâche requière 6 degrés de liberté), il devient impensable d'utiliser des algorithmes qui demandent un temps de calcul important pour des manipulateurs à 4 degrés de liberté ou plus. Cela est surtout dû au calcul de l'image des obstacles dans l'espace des configurations.

Certains auteurs apportent l'argument que ces algorithmes peuvent être utilisés pour les articulations entre la base et le poignet, si le manipulateur est découplable [25]. Bien que ceci diminue effectivement le nombre de degrés de liberté à traiter (3 degrés de liberté de moins), l'algorithme résultant serait trop long en temps de calcul pour des manipulateurs à 7 degrés de liberté et plus. De plus, cette méthode repose sur l'hypothèse d'un organe terminal de dimension relativement petite. Ceci diffère de notre cas, surtout lorsque le manipulateur peut transporter un outil quelconque.

L'utilisation d'algorithmes ayant un caractère partiellement global tel que [11] pourrait être intéressante, mais ils sont adaptés à des cas dans un environnement répétitif, du fait qu'ils demandent un apprentissage.

On se concentrera donc sur les méthodes locales. Bien que celles-ci aient des capacités limitées à trouver un chemin, elles peuvent donner rapidement des résultats acceptables pour des cas simples, ceci correspondant aux critères demandés.

Parmi les méthodes locales, les stratégies utilisant les champs de potentiels sont les plus intéressantes, vu leur simplicité et leur manière naturelle de diriger le manipulateur. Les champs de potentiel analytiques présentés dans [19] comportent beaucoup de minimums locaux et sont donc bloqués régulièrement. Une autre approche utilise les champs de potentiel discrets [4] et se révèle des plus intéressante car leur construction permet de ne pas avoir de minimum local pour un espace de travail donné, ce qui diminue les minimums locaux de la fonction globale et augmente la capacité à trouver des solutions. Sa compatibilité avec une représentation de l'espace de travail par grille d'occupation, qui est la représentation utilisée dans notre application, est à son avantage.

Un autre champ de potentiel ne produisant pas de minimums locaux et étant éventuellement compatible avec la représentation des obstacles par *octree* est basé sur

des fonctions potentielles harmoniques [20]. Celui-ci pourrait se comparer à l'approche de [4], mais on ne connaît pas bien ses capacités en 3 dimensions et il semble plus complexe.

Une autre approche, ne calculant qu'une partie de l'espace des configurations, consiste à rechercher séquentiellement dans l'espace $t \times \theta$ [15]. Bien que plus courte en temps de calcul que les approches globales, elle demeure relativement longue comparativement aux méthodes locales. De plus, il faut déterminer la configuration finale à priori, ce que l'on cherche à éviter, puisque l'objectif final est une pose de l'organe terminal seulement.

L'utilisation d'un chemin hypothétique corrigé par un champ de potentiel [8] [35] diminue de beaucoup la possibilité de tomber dans un minimum local. Mais pour être efficace, il doit travailler dans l'espace des configurations [35], qui demande trop de temps de calcul.

Les représentations des obstacles dans l'espace de travail par *octree* [16] [17] [29] et par grille d'occupation [4] sont très avantageuses. Puisqu'elles sont compatibles avec la représentation fournie par la caméra 3D, elles ne demandent pas de transformations qui peuvent être très coûteuses en temps de calcul.

En résumé, l'algorithme regroupant les éléments les plus intéressants est celui de Barraquand et Latombe [4]. C'est donc à partir de cette stratégie que la planification de trajectoire dans un environnement encombré est construite.

2.5 Cinématique inverse d'un manipulateur redondant

Suite à des essais préliminaires, nous avons envisagé l'utilisation de la cinématique inverse d'un manipulateur redondant, contrairement à ce qui est fait en [4]. Les raisons motivant cette utilisation sont expliquées au chapitre 4. Plusieurs articles discutent de l'utilisation de la cinématique inverse d'un manipulateur redondant [1] [5] [6] [7] [21] [26] [28] [32] . Ils présentent des applications, dont l'évitement d'obstacles, et proposent des variantes adaptées à différentes situations. Plusieurs proposent de

l'utiliser en coopération avec un module qui fournit la trajectoire de l'organe terminal . L'implantation initiale des champs de potentiels discrets conduit justement à un module adéquat après quelques modifications. On utilise la cinématique inverse en vitesse, car elle est bien adaptée à la situation et assez flexible pour y apporter des modifications. Suffisante, la version la plus simple est utilisée. La cinématique inverse d'un manipulateur redondant est présentée en détail au chapitre 4.

Chapitre 3

Description générale de la stratégie

On présente ici une vue d'ensemble de la stratégie de planification de trajectoire qui permettra de situer globalement la description détaillée des différents éléments fournie dans les chapitres ultérieurs. La stratégie regroupe des éléments appartenant aux algorithmes sélectionnés au chapitre précédent, soit les champs de potentiel discrets et la cinématique inverse d'un manipulateur redondant. De plus, elle utilise des algorithmes originaux, développés pour ce projet. L'ensemble vise les objectifs cités dans l'introduction, principalement la rapidité de calcul et l'efficacité.

Pour réaliser la planification de trajectoire, le manipulateur est modélisé par des points de contrôle, eux-mêmes influencés par des champs de potentiel discrets, tel qu'illustré à la figure 3.1.

Tout d'abord, il y a acquisition des paramètres se rapportant à l'environnement et au manipulateur. Puis, la construction d'une *table préparatoire* précède toute planification de trajectoire.

Pour chaque nouvel environnement, fourni par un système de vision, une *table des distances* et un *champ de potentiel attractif préparatoire* sont construits à partir de la table préparatoire. Une version modifiée du champ potentiel attractif préparatoire est ensuite construite à l'aide d'une heuristique qui tient compte d'un certain type d'obstacles, tels que des fils. Un *champ de potentiel répulsif* est aussi construit à partir de la table des distances.

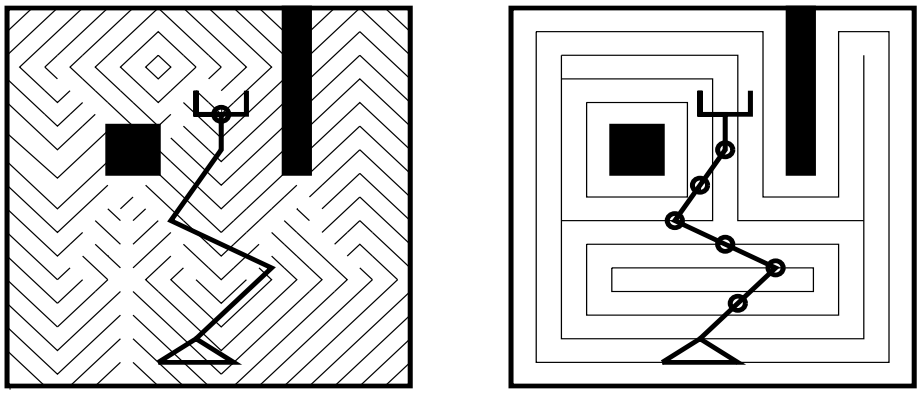


Figure 3.1: Manipulateur modélisé par des points de contrôle dans des champs de potentiel a) attractif et b) répulsif.

Pour chaque nouvelle trajectoire, on vérifie les implications de l'objectif demandé et un *champ de potentiel attractif* est construit à partir du champ potentiel attractif préparatoire et de l'objectif.

On est alors prêt à générer la trajectoire. Un point de contrôle attaché à l'organe terminal est poussé par le champ potentiel attractif vers l'objectif demandé en proposant par incréments de nouvelles poses de l'organe terminal qui rapprochent de l'objectif. À chaque incrément, les autres points de contrôle attachés au reste du robot sont éloignés des obstacles par le champ de potentiel répulsif. Les deux tâches sont possibles grâce à la présence de redondance. Pour agencer adéquatement celles-ci, l'algorithme utilise la cinématique inverse en vitesse d'un manipulateur redondant qui permet de réaliser une tâche primaire (organe terminal vers son objectif) et une tâche secondaire (éloignement par rapport aux obstacles), où la tâche secondaire ne nuit pas à la tâche primaire. À mesure que les configurations de la trajectoire sont générées, une vérification de collision (à l'aide de la table des distances) et de respect des limites articulaires s'effectue, et une correction est apportée en cas de problème. Si un blocage de l'algorithme de planification de trajectoire survient, des *heuristiques de dégagement* proposent des séquences de trajectoire pour permettre de dégager le robot de la situation problématique et ainsi trouver une solution. Une fois les trajectoires articulaires générées, un lissage s'effectue sur celles-ci pour diminuer les accélérations. Finalement, un intervalle de temps entre les pas est calculé à partir des limites du

manipulateur. La trajectoire est alors prête à être envoyée au contrôleur du robot.

À la figure 3.2, un organigramme général de la stratégie de planification de trajectoire illustre sa structure. La figure 3.3 présente l'organigramme de la séquence de génération de la trajectoire. Le lecteur pourra se référer à ceux-ci pour situer les éléments de la stratégie présentés dans les chapitres suivants.

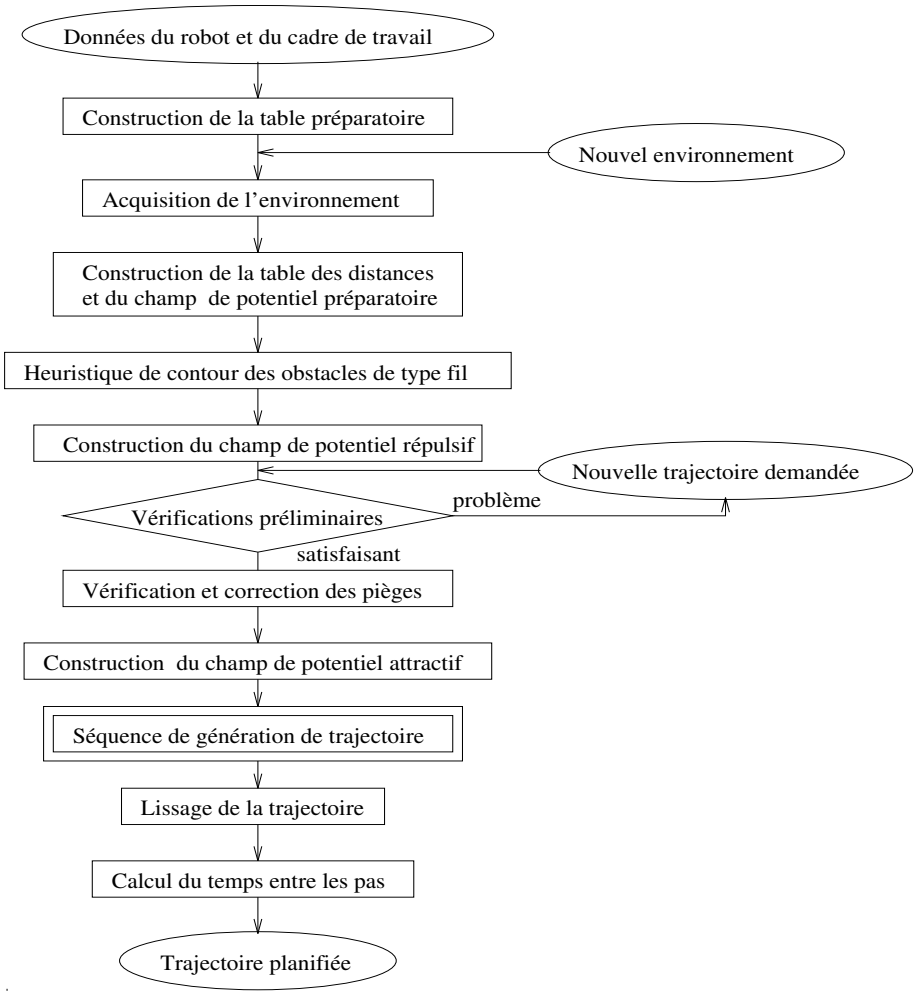


Figure 3.2: Organigramme général de la stratégie.

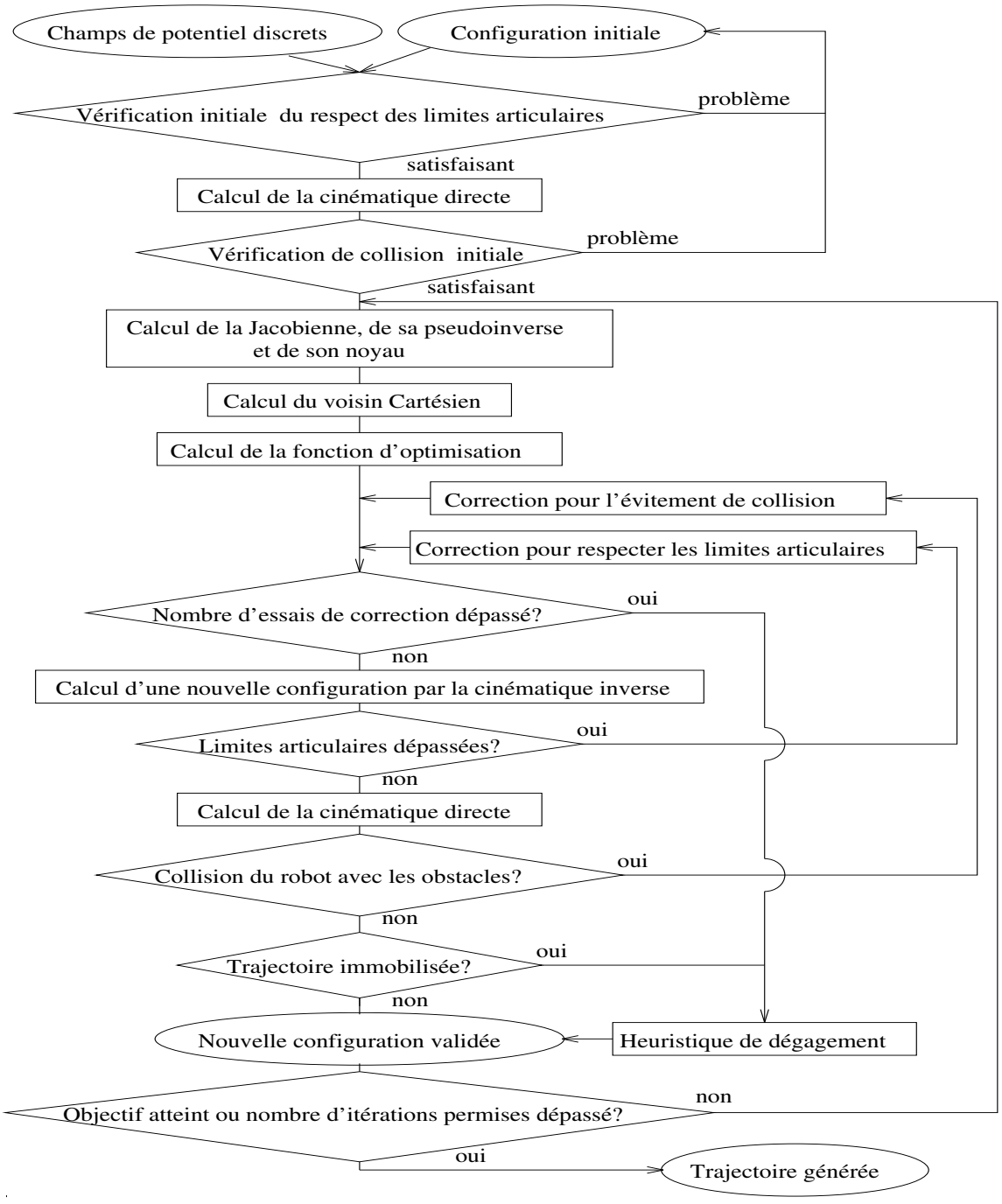


Figure 3.3: Organigramme de la séquence de génération de trajectoire.

Chapitre 4

Concepts de base en robotique

4.1 Définitions

Cette section donne les définitions de notions, se rapportant à la robotique, utilisées dans le mémoire. Elles sont tirées de [14].

Un *manipulateur* désigne une machine pouvant aider l'homme dans des tâches de manipulation. On dit le manipulateur *robotique* lorsqu'il est placé sous la commande d'un ordinateur.

L'*organe terminal* identifie le membre d'un manipulateur auquel est attaché l'effecteur. L'*effecteur*, qui réalise la tâche, comporte généralement un outil, tel qu'une main.

La *télémanipulation* consiste à faire reproduire par un manipulateur les mouvements d'un humain. Un pantin, ou une manette, capte les mouvements de l'humain et le système demande au manipulateur de les reproduire à distance.

Une *chaîne cinématique* consiste en un système mécanique composé de corps rigides, ou membres, reliés entre eux par des liaisons cinématiques. Les liaisons cinématiques principalement utilisées en robotique sont les liaisons rotoïde et prismatique. Deux corps couplés par une liaison *rotoïde* peuvent tourner, l'un par rapport à l'autre autour d'un axe de rotation, mais ne peuvent subir de mouvements de translation relatifs. Deux corps couplés par une liaison *prismatique* ne peuvent subir que des mouvements relatifs de translation et ce dans une seule direction, soit celle de l'axe de translation.

Un manipulateur *sériel* est constitué d'une chaîne cinématique simple et ouverte. Une chaîne cinématique est dite simple si chacun des corps rigides qui la constituent est couplé à au plus deux autres corps. Elle est ouverte si deux de ses membres, le premier et le dernier, ne sont couplés qu'à un seul corps.

La notation de *Hartenberg-Denavit* (H-D) est une convention utilisée pour décrire l'architecture d'un manipulateur sériel avec liaisons rotoïdes et prismatiques. Elle associe un repère cartésien à chaque membre. C'est la relation entre les différents repères en position et en orientation qui donne l'architecture du manipulateur. Le présent mémoire utilise cette notation et on la considère connue du lecteur. Pour une description détaillée, se référer à [14].

L'*architecture*, fixe pour un mécanisme donné, est l'arrangement des corps qui détermine la chaîne cinématique. Pour un manipulateur, elle est définie par les paramètres H-D.

La *configuration*, variable pour un mécanisme donné, désigne la position et orientation relative instantanée des corps de la chaîne entre eux. Pour un manipulateur, elle est définie par ses coordonnées articulaires.

Le *degré de liberté* d'une chaîne cinématique représente le nombre de paramètres indépendants qu'il faut spécifier pour en déterminer exactement la configuration.

Un manipulateur est dit *redondant* par rapport à une tâche donnée lorsque le nombre de degrés de liberté du manipulateur s'avère plus grand que le nombre de degrés de liberté nécessaire pour effectuer la tâche.

4.2 Cinématique directe

La cinématique directe permet d'obtenir la pose (position et orientation) de l'organe terminal, connaissant les coordonnées articulaires du manipulateur. La position est donnée par le vecteur position $\mathbf{p}_b$, et l'orientation, par la matrice de rotation $\mathbf{Q}_b$, ceux-ci représentant la pose de l'organe terminal par rapport à la base. Les équations de cinématique directe sont exprimées comme [14]

$$\mathbf{Q}_1\mathbf{Q}_2\mathbf{Q}_3...\mathbf{Q}_n = \mathbf{Q}_b \tag{4.1}$$

$$\mathbf{a}_1 + \mathbf{Q}_1\mathbf{a}_2 + \mathbf{Q}_1\mathbf{Q}_2\mathbf{a}_3 + ... + \mathbf{Q}_1\mathbf{Q}_2...\mathbf{Q}_{n-1}\mathbf{a}_n = \mathbf{p}_b \tag{4.2}$$

où $\mathbf{Q}_i$ est la matrice de rotation amenant le repère i au repère $(i+1)$ et $\mathbf{a}_i$ le vecteur position reliant l'origine du repère i à celle du repère $(i+1)$. Ces quantités sont exprimées dans le repère i. La valeur n donne le nombre de membres (ou le nombre de degrés de liberté pour un manipulateur sériel).

En fonction des paramètres H-D et des coordonnées articulaires, $\mathbf{Q}_i$ et $\mathbf{a}_i$ sont calculés ainsi :

$$\mathbf{Q}_i = \begin{bmatrix} \cos\theta_i & -\cos\alpha_i \sin\theta_i & \sin\alpha_i \sin\theta_i \\ \sin\theta_i & \cos\alpha_i \cos\theta_i & -\sin\alpha_i \cos\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i \end{bmatrix} \tag{4.3}$$

$$\mathbf{a}_i = \begin{bmatrix} a_i \cos\theta_i \\ a_i \sin\theta_i \\ b_i \end{bmatrix} \tag{4.4}$$

où θ_i est la i^{eme} coordonnée articulaire et où α_i, a_i et b_i sont les paramètres H-D de la i^{eme} articulation [14] (dans le cas d'une articulation prismatique, le rôle de θ_i et b_i est inversé). Dans cette application, les valeurs sont exprimées par rapport à un repère global. Les équations de cinématique directe se trouvent donc modifiées par deux termes: la matrice $\mathbf{P}_1$, qui exprime l'orientation du repère de base du manipulateur par rapport au repère global, et le vecteur $\mathbf{o}_1$, qui exprime la position du repère de base par rapport à la position du repère global. Les équations deviennent alors

$$\mathbf{P}_1\mathbf{Q}_1\mathbf{Q}_2\mathbf{Q}_3...\mathbf{Q}_n = \mathbf{Q} \tag{4.5}$$

$$\mathbf{o}_1 + \mathbf{P}_1\mathbf{a}_1 + \mathbf{P}_1\mathbf{Q}_1\mathbf{a}_2 + \mathbf{P}_1\mathbf{Q}_1\mathbf{Q}_2\mathbf{a}_3 + ... + \mathbf{P}_1\mathbf{Q}_1\mathbf{Q}_2...\mathbf{Q}_{n-1}\mathbf{a}_n = \mathbf{p} \tag{4.6}$$

En pratique, on veut aussi obtenir la position et l'orientation des repères intermédiaires, en plus d'un calcul rapide. Pour ce faire, le calcul s'effectue de manière récursive dans l'algorithme.

La matrice de rotation amenant le repère global au repère i, $\mathbf{P}_i = \mathbf{P}_1\mathbf{Q}_1...\mathbf{Q}_{i-1}$ est calculée récursivement comme

$$\mathbf{P}_{i+1} = \mathbf{P}_i\mathbf{Q}_i \quad \text{pour} \quad i = 1, ..., n \tag{4.7}$$

La matrice $\mathbf{P}_1$ se retrouve dans le fichier de description du manipulateur. Une séquence de calcul efficace de $\mathbf{P}_i$, présentée en détail dans [14], est utilisée dans l'algorithme.

Le vecteur position de l'origine du repère i, $\mathbf{o}_i$ se calcule récursivement comme

$$\mathbf{o}_{i+1} = \mathbf{o}_i + \mathbf{P}_i \mathbf{a}_i \quad \text{pour} \quad i = 1, ..., n \tag{4.8}$$

Le vecteur $\mathbf{o}_1$ est aussi fourni dans le fichier de description du manipulateur.

4.3 Jacobienne et équations de vitesse

La relation entre les vitesses articulaires $\dot{\theta}_i$, pour $i = 1, 2, ..., n$, et la vitesse de l'organe terminal, soit le vecteur vitesse Cartésienne du manipulateur $\mathbf{t}$, est donnée par la matrice Jacobienne $\mathbf{J}$. La relation qui permet d'obtenir la vitesse de l'organe terminal s'écrit [14]

$$\mathbf{t} = \mathbf{J}\dot{\boldsymbol{\theta}} \tag{4.9}$$

avec

$$\dot{\boldsymbol{\theta}} = \left[\dot{\theta}_1, \dot{\theta}_2, ..., \dot{\theta}_n\right]^T \tag{4.10}$$

et

$$\mathbf{t} = \left[\dot{\mathbf{p}}^T, \boldsymbol{\omega}^T\right]^T \tag{4.11}$$

où $\dot{\mathbf{p}}$ désigne le vecteur vitesse du point de référence sur l'organe terminal et $\boldsymbol{\omega}$ est le vecteur vitesse angulaire de l'organe terminal.

Le vecteur $\mathbf{j}_i$ désignant la i-ième colonne de $\mathbf{J}$, soit

$$\mathbf{J} = [\mathbf{j}_1, \mathbf{j}_2, ..., \mathbf{j}_n] \tag{4.12}$$

on a

$$\mathbf{j}_i = \begin{bmatrix} \mathbf{e}_i \times \mathbf{r}_i \\ \mathbf{e}_i \end{bmatrix} \tag{4.13}$$

si l'articulation i est rotoïde et

$$\mathbf{j}_i = \begin{bmatrix} \mathbf{e}_i \\ 0 \end{bmatrix} \tag{4.14}$$

pour une articulation prismatique. Le vecteur $\mathbf{e}_i$ représente le vecteur propre de $\mathbf{Q}_i$ associé à la valeur propre +1 (vecteur unitaire sur l'axe de rotation i) et $\mathbf{r}_i$ est le vecteur qui relie O_i, l'origine du repère i, à O_{n+1}, l'origine du repère attaché à l'organe terminal. Ceux-ci sont exprimés dans le repère global.

Les vecteurs $\mathbf{e}_i$ sont donnés par la dernière colonne des matrices $\mathbf{P}_i$. Puisque la cinématique directe est d'abord calculée pour les besoins de l'algorithme, on utilise directement les $\mathbf{P}_i$ obtenus lors du calcul de la cinématique directe.

Les vecteurs $\mathbf{r}_i$ sont donnés par

$$\mathbf{r}_i = \mathbf{p} - \mathbf{o}_i \tag{4.15}$$

où $\mathbf{p}$ et $\mathbf{o}_i$ proviennent de la cinématique directe. Les $\mathbf{r}_i$ seraient obtenus différemment si la cinématique directe n'était pas d'abord calculée [14].

On note que $\mathbf{J}$ peut être séparé en deux blocs correspondant aux vecteurs $\dot{\mathbf{p}}$ et $\boldsymbol{\omega}$. On peut ainsi exprimer $\mathbf{J}$ par

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}_A \\ \mathbf{J}_B \end{bmatrix} \tag{4.16}$$

avec

$$\dot{\mathbf{p}} = \mathbf{J}_A \dot{\boldsymbol{\theta}} \tag{4.17}$$

$$\boldsymbol{\omega} = \mathbf{J}_B \dot{\boldsymbol{\theta}} \tag{4.18}$$

4.4 Cinématique inverse en vitesse d'un manipulateur redondant

4.4.1 Utilité de la cinématique inverse

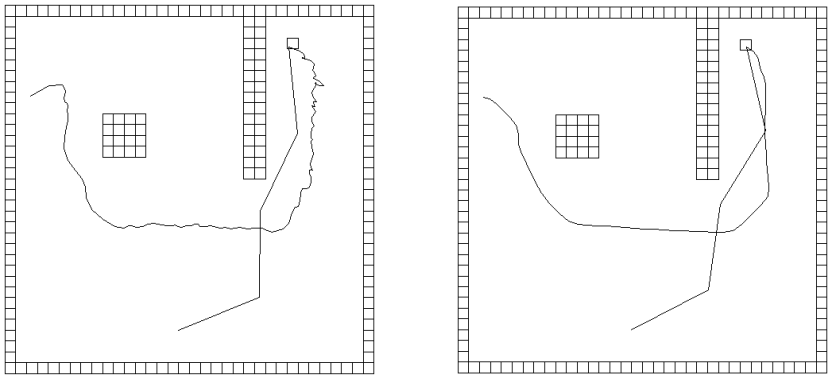
La stratégie choisie, soit la méthode de recherche dans l'espace des configurations à l'aide des champs de potentiel discrets, a d'abord été implantée. Contrairement à la référence [4], la configuration finale n'est pas fournie. Seule la pose de l'organe terminal est demandée à l'opérateur, ce dernier n'ayant pas à déterminer où sera le reste du manipulateur pour la tâche. Déterminer la cinématique inverse du manipulateur sans un bon estimé initial devient difficile et incertain lorsqu'il y a des obstacles.

Cette étape peut être longue et donner de mauvais résultats. De plus, elle diminue les chances de succès de l'algorithme de planification de trajectoire puisqu'elle restreint les possibilités. Il s'avère donc avantageux de laisser à l'algorithme de planification de trajectoire le soin de déterminer une configuration finale qui satisfait les contraintes de pose demandée de l'organe terminal et d'évitement d'obstacles.

Pour positionner l'organe terminal, il n'y a qu'un point de contrôle dans un champ de potentiel attractif. Il est à remarquer que l'utilisation de plusieurs champs potentiels attractifs augmente beaucoup les temps de calcul. L'orientation de l'organe terminal n'est pas obtenue par des champs potentiels. Elle est plutôt obtenue par une méthode utilisant les repères attachés à l'objectif et à l'organe terminal, qui est expliquée au chapitre 6. Pour le reste du manipulateur, on pourrait se fier au vérificateur de collisions, mais celui-ci frôlerait les obstacles, ce qui provoquerait de nombreux minimums locaux. Un champ de potentiel répulsif, qui éloigne les points de contrôle attachés au reste du manipulateur, est donc construit à partir de la table des distances rendant ainsi les collisions prévisibles et maintenant le robot éloigné des obstacles.

L'utilisation de plusieurs points de contrôle occasionne quand même des minimums locaux à cause de l'interaction entre ceux-ci. De plus, la recherche aléatoire dans l'espace des articulations accentue l'*indécision* de l'algorithme, c'est-à-dire sa tendance à se diriger un peu partout au lieu d'aller directement vers l'objectif. La trajectoire obtenue est chaotique (voir figure 4.1a) et l'objectif n'est pas toujours atteint, même pour des cas simples, sans l'algorithme d'échappement des minimums locaux.

On veut éviter l'utilisation de l'algorithme d'échappement de minimums locaux utilisant le mouvement Brownien pour deux raisons. Premièrement, bien qu'il soit efficace à cent pour cent après une période de temps infinie, il s'avère trop long pour être utile, l'algorithme devant trouver une solution en quelques secondes (environ 10 s), alors que celui-ci peut régulièrement prendre plus d'une minute, particulièrement à cause de la recherche non-guidée. Deuxièmement, ceci oblige à utiliser l'algorithme de lissage de [4] qui est relativement plus long et complexe que celui que l'on propose. De plus, la trajectoire obtenue répondrait à d'autres critères que ceux recherchés.



a b

Figure 4.1: Exemples de trajectoires de l'effecteur : a) trajectoire chaotique, b) trajectoire obtenue à l'aide de la cinématique inverse.

L'indécision de l'algorithme provient surtout du conflit entre la tâche d'atteindre l'objectif et celle d'éviter les obstacles. Elle est aussi due à l'utilisation d'un élément aléatoire pour diminuer l'espace de recherche, afin de trouver une configuration voisine.

Il est donc intéressant d'ordonner ces deux tâches pour diminuer leur interaction, principalement l'influence de la deuxième tâche sur la première. La figure 4.1b illustre un exemple de trajectoire générée en utilisant la cinématique inverse.

La cinématique inverse d'un manipulateur redondant permet de séparer ces deux tâches. En effet, celle-ci permet d'effectuer une seconde tâche à l'aide des degrés de liberté redondants du manipulateur (éviter les obstacles) sans affecter la première tâche (atteindre l'objectif). De plus, la recherche se fait dans l'espace cartésien pour $3^3-1 = 26$ voisins, ce qui ne nécessite pas de nombres aléatoires (l'algorithme prend *la* meilleure direction), au lieu de se faire parmi 3^n-1 voisins (6560 voisins pour 8 degrés de liberté). Il est aussi beaucoup plus facile de contrôler la vitesse de l'organe terminal , puisqu'elle est donnée directement. Il en résulte une trajectoire plus rapidement trouvée et moins chaotique. De plus, il y a beaucoup moins de minimums locaux, ce qui entraine un taux de réussite beaucoup plus élevé sans système d'évasion des minimums locaux.

Ainsi, une implantation préliminaire de l'algorithme fondée sur les champs de potentiel discrets a permis de mettre en évidence l'utilité de la cinématique inverse dans la planification de trajectoire des manipulateurs redondants dans un environnement encombré.

4.4.2 Équations de la cinématique inverse

Pour effectuer la planification de trajectoire, l'algorithme utilise la cinématique inverse en vitesse d'un manipulateur redondant, c'est-à-dire que l'on cherche à trouver des vitesses articulaires (ou des variations des positions articulaires) pour une vitesse cartésienne (ou variation de la pose cartésienne) donnée. La relation entre les vecteurs vitesse cartésienne et vitesse articulaire est donnée à l'équation 4.9.

Vu la redondance du manipulateur, on se retrouve face à un système sous-déterminé. Pour le résoudre, on calcule une solution particulière qui minimise la norme du vecteur solution $\dot{\boldsymbol{\theta}}$, plus un vecteur arbitraire, orthogonal au premier, qui permet de minimiser une fonction potentielle p quelconque.

La relation utilisée est souvent exprimée sous la forme [26][28]

$$\dot{\boldsymbol{\theta}} = \mathbf{J}^I\mathbf{t} + (\mathbf{1} - \mathbf{J}^I\mathbf{J})\mathbf{z} \tag{4.19}$$

où

$$\mathbf{J}^I = \mathbf{J}^T(\mathbf{J}\mathbf{J}^T)^{-1} \tag{4.20}$$

est la pseudoinverse de la matrice Jacobienne permettant de calculer la solution à norme minimale et

$$\mathbf{z} = [z_1, z_2, ..., z_n]^T = -\frac{\partial p}{\partial \boldsymbol{\theta}} \tag{4.21}$$

désigne le terme d'optimisation, c'est-à-dire la dérivée partielle de la fonction p par rapport aux coordonnées articulaires.

La matrice $(\mathbf{1} - \mathbf{J}^I\mathbf{J})$ projette donc ce vecteur d'optimisation dans le noyau de la matrice $\mathbf{J}$. Ainsi, le mouvement généré par le second terme de l'équation(4.19) n'affecte pas la trajectoire cartésienne de l'effecteur.

Pour un calcul stable et rapide, la pseudoinverse $\mathbf{J}^I$ est calculée à l'aide d'un algorithme de décomposition en valeurs singulières (DVS) [14] [30]. Une version plus

rapide de la (DVS), adaptée à l'algorithme de type incrémental, proposée dans [27], pourra éventuellement être implantée.

Il est souvent souhaitable de pondérer le mouvement relatif de chacune des articulations. Ceci se révèle surtout vrai pour les cas où un même manipulateur comporte des articulations rotoïdes et prismatiques ou des débattements articulaires différents. Pour ce faire, on utilise une matrice de pondération diagonale $\mathbf{W}$.

L'importance des deux parties de l'équation(4.19) peut être ajustée en modifiant des facteurs de corrections C_1 et C_2.

Suite à ces ajouts, l'équation de cinématique inverse devient [14]

$$\dot{\boldsymbol{\theta}} = C_1 \mathbf{W}_0^{-1} \mathbf{U}^I \mathbf{t} + C_2 \mathbf{W}_0^{-1} (\mathbf{1} - \mathbf{U}^I \mathbf{U}) \mathbf{z} \tag{4.22}$$

où

$$\mathbf{W} = \mathbf{W}_0 \mathbf{W}_0^T \tag{4.23}$$

et

$$\mathbf{U} = \mathbf{J} \mathbf{W}_0^{-1} \tag{4.24}$$

4.4.3 Applications

Dans le cadre de ce projet, le terme d'optimisation sert principalement à éviter les obstacles. Pour ce faire, la fonction p est un champ de potentiel dont la valeur est inversement proportionnelle à la distance aux obstacles. Une autre application dans le cadre de ce projet consiste à amener l'organe terminal vers une orientation voulue. Ces applications sont expliquées en détail au chapitre 6.

Le terme d'optimisation a d'autres applications telles que l'optimisation du conditionnement, la minimisation de l'énergie potentielle, etc.

Chapitre 5

Champs de potentiel discrets

5.1 Description de l'environnement

Pour réaliser la planification de trajectoire, l'environnement du manipulateur doit être connu, du moins partiellement. Dans le contexte de ce projet, l'environnement est fixe pour une trajectoire donnée. L'algorithme utilise un modèle de l'environnement fourni par un système de vision 3D, développé au Laboratoire de vision numérique du département de génie électrique de l'Université Laval [2]. Le modèle de l'environnement est une grille d'occupation. Dans la représentation par grille d'occupation, l'espace de travail est discrétisé en cubes de grandeur égale (en carrés si l'environnement est en 2D), que l'on appellera *noeuds* par la suite. Si un noeud est occupé en tout ou en partie par un obstacle, il est considéré occupé. Si un noeud est exempt de tout obstacle, il est considéré libre. Ce qui est à l'extérieur du *cadre* de l'espace de travail discrétisé est considéré occupé, puisque rien ne garantit l'absence d'obstacles dans cet espace inconnu. Le manipulateur ne peut circuler que dans la zone constituée par les noeuds libres.

Les dimensions de l'espace de travail discrétisé et le niveau de discrétisation de la grille d'occupation peuvent être choisis. Le choix est alors un compromis entre la qualité du modèle de l'environnement, qui augmente les chances de réussite de la stratégie, et le temps de calcul.

Ce mode de représentation des obstacles a l'avantage d'être très général et flexible,

puisqu'il peut représenter n'importe quoi, en plus d'être très simple.

Pour établir le modèle de l'environnement, une caméra sera placée sur le manipulateur pour obtenir les points de vue voulus. Pour déplacer la caméra, le manipulateur doit effectuer une trajectoire. Le planificateur de trajectoire sert donc entre autres à positionner la caméra lors de l'acquisition du modèle de la scène. L'interdépendance des composantes planification de trajectoire et vision entraîne la présence d'une boucle. Un point de départ à cette boucle est rendu possible grâce au concept d'espace préparatoire discuté plus loin.

5.1.1 Génération artificielle d'une grille d'occupation

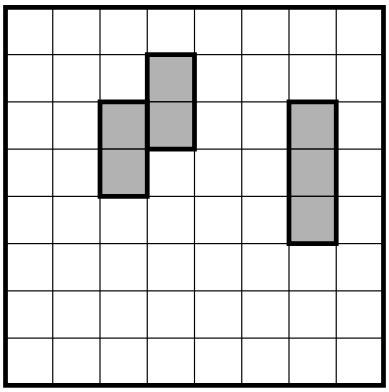


Figure 5.1: Environnement sous forme de grille d'occupation.

Le simulateur a besoin d'obstacles pour vérifier les algorithmes. Pour obtenir rapidement des obstacles quelconques, il faut les créer artificiellement. Un module permettant d'aider à leur création a donc été conçu. L'idée générale consiste à construire un environnement sous forme de grille d'occupation à l'aide de parallépipèdes à angles droits (rectangles en 2D). L'utilisateur donne les coordonnées d'un coin et les dimensions de chaque parallépipède qui compose les obstacles de la scène. Le module considère occupés les noeuds englobés par les parallépipèdes et les autres noeuds sont considérés vides. Le module crée ainsi une grille d'occupation semblable à celle éventuellement fournie par le système de vision. Les coordonnées des obstacles

parallépipèdes sont fournies par fichiers, que le système peut lire à la demande de l'utilisateur. On obtient alors une méthode simple et flexible de créer des obstacles artificiels. La figure 5.1 illustre un exemple simple en 2D d'un environnement sous forme de grille d'occupation dans laquelle les rectangles servent à créer les obstacles.

5.2 Champs de potentiel discrets

Cette section reprend en détail les éléments du concept de champs de potentiel discrets présenté dans [4] et utilisé dans la stratégie développée ici.

Les champs de potentiel discrets servent à diriger le manipulateur dans son environnement, afin qu'il atteigne ses objectifs. Le manipulateur est modélisé par des points de contrôle qui sont influencés par ces champs de potentiel (ce concept est illustré à la figure 3.1). Des points de vérification de collision attachés au manipulateur sont utilisés pour vérifier si le robot est en collision avec un obstacle à l'aide d'une table des distances.

Les champs de potentiel discrets sont construits à l'aide d'une expansion par vagues. D'abord, une table des distances aux obstacles L_1 est créée. La distance L_1 équivaut à la somme scalaire des écarts de position des noeuds en X, Y et Z. En parallèle, un *squelette* de l'espace de travail, semblable, en 2D, à un diagramme de Voronoï, est extrait. À partir de la table des distances et du squelette, un champ de potentiel *attractif*, qui pousse un point de contrôle vers son objectif, est construit.

5.2.1 Table des distances

La distance aux obstacles L_1 est d'abord calculée. Premièrement, les points aux frontières des obstacles sont identifiés et se voient attribuer la valeur 0. Les points à la frontière de l'espace de travail possèdent aussi la valeur 0, puisque la frontière est considérée comme un obstacle. Par la suite, une expansion par vague s'effectue à partir de ces points, dans la zone libre. Les voisins non-traités des noeuds de valeur 0 se voient attribuer la valeur 1. Ensuite, les voisins non-traités des noeuds de valeur 1 obtiennent la valeur 2. Le processus se répète jusqu'à ce que tout l'espace libre soit

traité. Ce processus est illustré à la figure 5.2.

0	0	0	0	0	0	0	0	0	0
0	1	1	1	0	1	1	1	1	0
0	1	2	1	0	1	1	2	1	0
0	1	2	1	0	0	1	2	1	0
0	1	2	1	1	1	1	2	1	0
0	1	2	2	2	2	2	2	1	0
0	1	2	3	3	3	3	2	1	0
0	1	2	2	2	2	2	2	1	0
0	1	1	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0	0

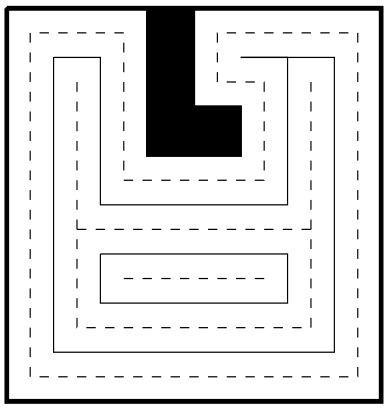


Figure 5.2: Construction de la table des distances (la figure de droite illustre le résultat par des lignes de niveau).

5.2.2 Squelette de l'espace de travail

En même temps, le squelette de l'espace de travail est construit. Celui-ci constitue un schéma général de l'espace libre et trace un *réseau de routes* pour faire voyager un point sécuritairement dans l'espace de travail. Il est constitué des points de rencontre des vagues d'expansion qui servent à créer la table des distances. Pour détecter ces points de rencontre, les positions d'origine de l'expansion sont propagées. Si un noeud d'expansion tombe sur un voisin traité, il vérifie si l'expansion provient d'un autre secteur que lui, c'est-à-dire que la distance L_1 entre les origines d'expansion est plus grande qu'un certain seuil. Si tel est le cas, il y a donc rencontre de deux vagues et le noeud fait partie du squelette. Ce squelette servira lors de la construction de champs potentiels attractifs. La figure 5.3 illustre un exemple de squelette.

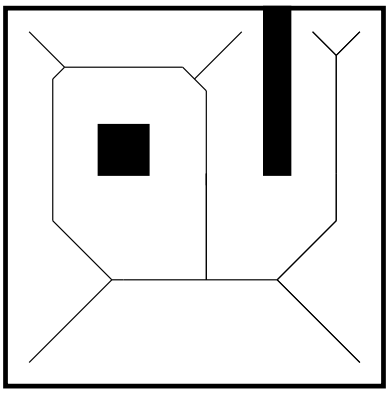


Figure 5.3: Exemple de squelette de l'espace de travail.

5.2.3 Champ de potentiel attractif

Par la suite, un champ de potentiel attractif sans minimums locaux est créé. Deux versions sont présentées. Une simple et une autre plus évoluée, qui donne de meilleurs résultats lors de la planification de trajectoire proprement dite.

La plus simple se réalise comme suit. Le principe d'expansion par vague est utilisé. Le noeud de départ de l'expansion est l'objectif à atteindre. L'algorithme lui attribue la valeur 1. Aux voisins non-traités de ce noeud, l'algorithme attribue la valeur 2. Par la suite, aux voisins non-traités des noeuds de valeur 2, l'algorithme accorde la valeur 3. Le processus, illustré à la figure 5.4, se répète jusqu'au traitement de tout l'espace de travail libre.

Si on demande itérativement au point de contrôle de se diriger vers une valeur de potentiel plus faible, il se rendra à son objectif, s'il existe un chemin. Cependant, le point aura tendance à frôler les obstacles, ce qui diminue les chances de trouver une configuration valide du manipulateur lors de la génération de trajectoire.

La version plus évoluée permet de régler le problème puisqu'elle garde le point de contrôle le plus loin possible des obstacles. Pour ce faire, le squelette de l'espace de travail est utilisé. Tout d'abord, un chemin est créé entre le noeud objectif et le squelette pour obtenir un *squelette augmenté*. Par la suite, une expansion dans ce squelette augmenté s'effectue à partir du point de départ. Une fois cette expansion

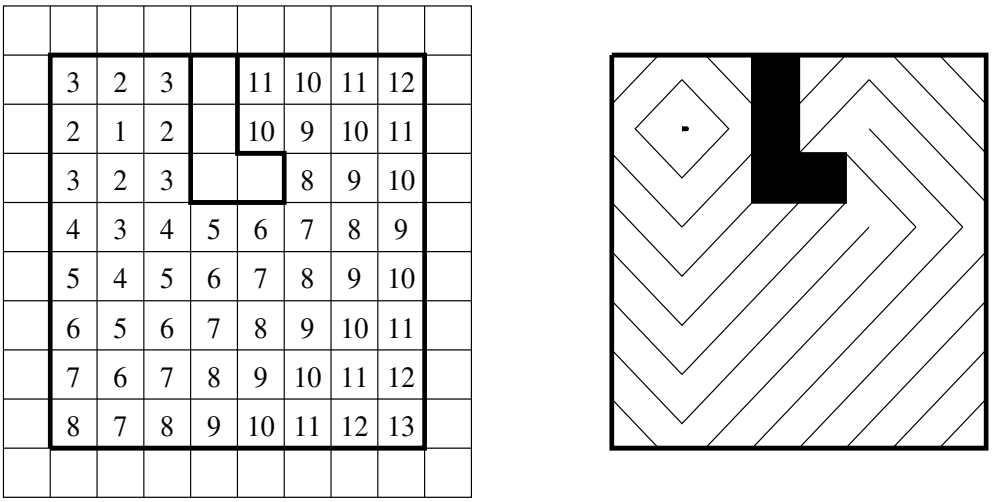


Figure 5.4: Construction du champ de potentiel attractif (la figure de droite illustre le résultat par des lignes équipotentielles).

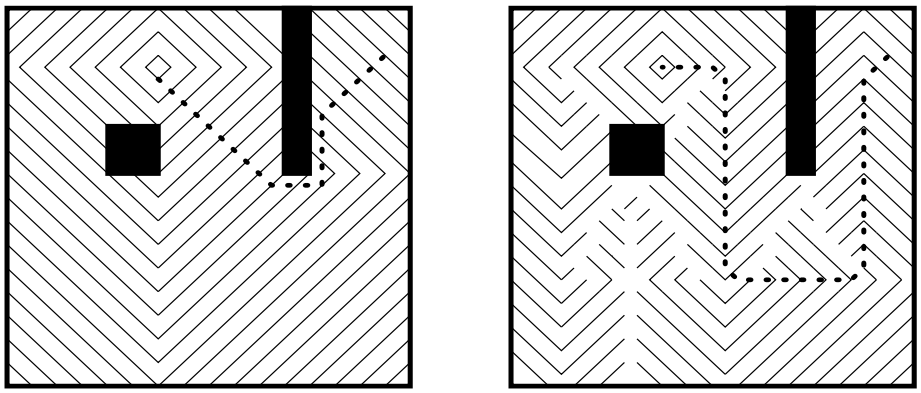
réalisée, une expansion à partir des noeuds du squelette est réalisée pour le reste de l'espace libre. Un champ qui pousse un point vers l'objectif tout en le repoussant le plus loin possible des obstacles est ainsi obtenu. Des exemples de champs de potentiel attractifs simple et évolué, avec des exemples de trajectoires obtenues, sont illustrés à la figure 5.5.

5.2.4 Remarques sur les champs de potentiel discrets

Cette section regroupe diverses remarques se rapportant aux champs de potentiel discrets. Dans [4], les auteurs proposent de se servir d'un niveau de discrétisation grossier et de raffiner, si une solution demeure introuvée, jusqu'à une résolution limite. L'utilisation d'un ordinateur parallèle permettrait d'explorer le problème avec différents niveaux de discrétisation en même temps.

Le principal avantage de ce type de champs de potentiels provient du fait qu'il ne contient pas de minimums locaux, contrairement aux champs de potentiels conventionnels.

La complexité de l'algorithme est linéairement proportionnelle au nombre de



a b

Figure 5.5: Exemples de champs de potentiel attractifs a) simple et b) évolué.

noeuds de l'espace de travail.

L'approximation du squelette et des distances par L_1 suffit pour les besoins de l'algorithme de génération de trajectoire.

5.3 Implantation des champs de potentiel discrets adaptée à l'application présente

Pour une meilleure adaptation à la présente application, les champs de potentiel présentés ont subi des modifications et de nouveaux champs ont été construits. La figure 5.6 présente un exemple de champ de potentiel attractif en 2D résultant de ces modifications.

5.3.1 Espace préparatoire

Tout d'abord, le concept d'espace préparatoire est introduit (voir illustration à la figure 5.7). Cet espace, libre de tout obstacle, englobe le manipulateur lorsqu'il est près de sa base. Ceci permet au manipulateur de se déplacer en toute sécurité lors de la prise des premières images par le système de vision 3D pour l'acquisition du modèle de l'environnement. En effet, la caméra étant placée sur le robot, celui-ci ne peut pas

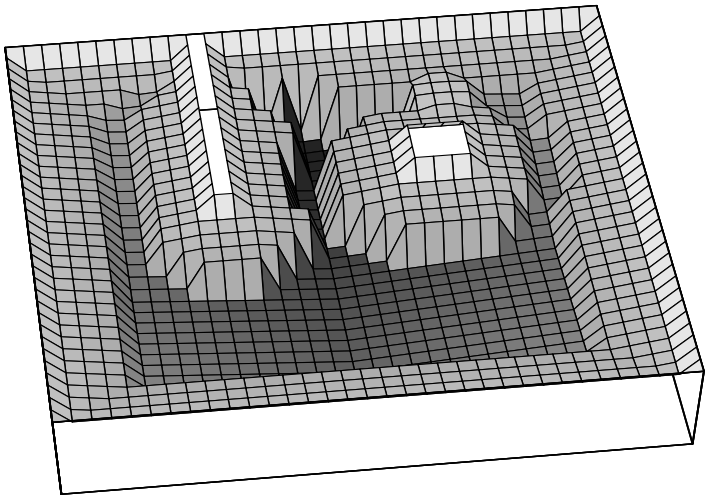


Figure 5.6: Exemple de champ de potentiel attractif en 2D résultant des modifications.

se voir. Cette zone permet aussi d'améliorer les performances de la planification de trajectoire. Bien positionnée, cette zone ne réduit pas significativement l'espace de travail utile puisqu'un robot n'est généralement pas utilisé pour travailler près de sa base.

5.3.2 Concept de voisin d'un noeud et types d'expansion

Dans la grille d'occupation, il y a plusieurs niveaux de voisins. Selon le type de voisin utilisé, l'expansion adoptera un aspect différent.

En 2D, il y a deux niveaux de voisins : 4 voisins de côtés et 4 voisins de coins. Si l'expansion est effectuée par les voisins de côté, on obtient une expansion de type losange, et lorsqu'effectuée par les voisins de côté et de coins (8 voisins), on obtient une expansion de type carrée. Si on utilise arternativement les deux types d'expansion, on produit une expansion de forme octogonale. Ceci est illustré à la figure 5.8.

En 3D, il y a trois niveaux de voisins : 6 voisins de face, 12 voisins d'arrête et 8 voisins de coin. Selon les voisins utilisés, l'expansion prendra différentes formes. Ici, l'expansion avec les voisins de face est assimilée à une expansion de type losange et celle avec tous les voisins est assimilée à une expansion de type carrée.

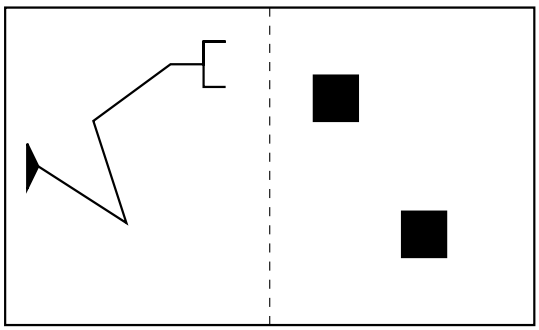


Figure 5.7: Illustration du concept d'espace préparatoire.

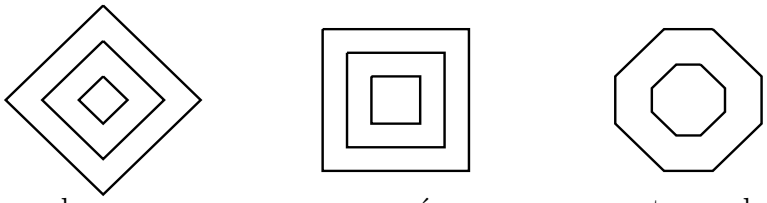


Figure 5.8: Types d'expansion.

5.3.3 Table des distances

Initialement, l'expansion de la table des distances se réalise avec 4 voisins en 2D et 6 voisins en 3D, soit de type losange. Avec cette expansion, pour qu'un point soit à au moins un noeud de distance d'un obstacle ($Secur = 1$), on a besoin de deux vagues d'expansion et le nombre de noeuds interdits (noeuds où le point ne doit pas aller pour qu'il soit à au moins un noeud de distance de l'obstacle) est de 12, en 2D. Avec l'expansion carrée, pour la même contrainte, on a besoin d'une vague d'expansion et le nombre de noeuds interdits est de 8, comme illustré à la figure 5.9. Ce phénomène se produit aussi en 3D. Puisque l'expansion carrée est moins restrictive, elle permet une zone de liberté plus grande pour le même niveau de sécurité. Elle est donc utilisée au début de l'expansion, dans la zone de sécurité. L'expansion de type losange est effectuée pour le reste de la zone de travail, puisqu'elle est plus adéquate pour la construction du squelette et la préparation des autres champs, vu son meilleur raffinement.

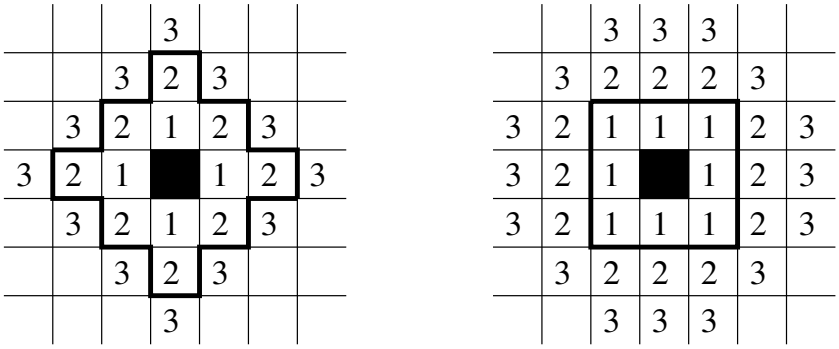


Figure 5.9: Zones de sécurité selon le type d'expansion initiale utilisée, exemple en 2D.

5.3.4 Champ de potentiel répulsif

Contrairement à [4], où tous les points de contrôle sont influencés par des champs de potentiel attractifs, l'application présente n'associe qu'un point de contrôle attractif à l'organe terminal. Le reste du manipulateur est simplement repoussé des obstacles. Pour ce faire, un champ de potentiel *répulsif* qui éloigne des obstacles est créé. Lorsque le robot (ou un point de contrôle) est loin d'un obstacle, il n'est pas impératif de tendre à s'en éloigner fortement, et il s'avère même important de ne pas le faire pour ne pas nuire aux autres effets des champs de potentiel. Par contre, lorsque le manipulateur se trouve près d'un obstacle, il faut le repousser fortement. La table des distances sert à obtenir rapidement un champ de potentiel répulsif ayant cet effet. Les valeurs du champ de potentiel répulsif *Rep* sont calculées comme

$$Rep = \frac{K}{L_1 - Secur} \quad \text{si} \quad L_1 > Secur \tag{5.1}$$

et

$$Rep \gg K \quad \text{si} \quad L_1 \leq Secur \tag{5.2}$$

où K correspond à la valeur maximum du champ de potentiel répulsif (pour l'implantation, $K = 100$) et $Secur$ est la distance de sécurité entre les points de contrôle et les obstacles. La figure 5.10 illustre un champ potentiel répulsif en coupe.

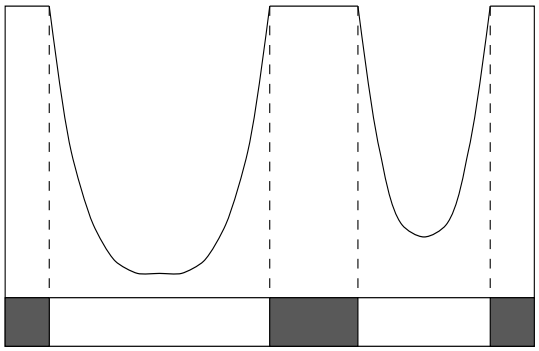


Figure 5.10: Champ potentiel répulsif vu en coupe (les zones sombres constituent les obstacles).

5.3.5 Qualité de la trajectoire

Plutôt que de faire passer le point de contrôle attractif le plus loin possible ou près des obstacles, il est préférable qu'il passe à une distance *raisonnable* des obstacles. Faire passer l'organe terminal le plus loin possible des obstacles provoque des replis du robot sur lui-même impossibles à réaliser à cause des limites articulaires. De plus, la trajectoire devient souvent inutilement longue (l'optimisation de trajectoire décrite dans [4] n'est pas utilisée pour les raisons décrites plus tôt). Faire passer l'organe terminal près des obstacles encombre plus sérieusement la zone autour du robot et diminue les chances de trouver une solution. Pour faire passer l'organe terminal à une distance intermédiaire des obstacles, l'algorithme effectue une première expansion, non seulement dans le squelette semblable à un diagramme de Voronoï, mais aussi dans l'ensemble des noeuds dont la distance aux obstacles est plus grande qu'une distance intermédiaire réduisant les problèmes cités plus tôt. On appellera la combinaison de ces deux zones le *squelette élargi*. Avec l'expansion effectuée dans cette zone, le point de contrôle aura naturellement tendance à frôler les bords de cette zone du côté qui minimise la longueur du trajet, tel qu'illustré à la figure 5.11. On obtient donc un trajet court, et qui facilite la génération de trajectoire.

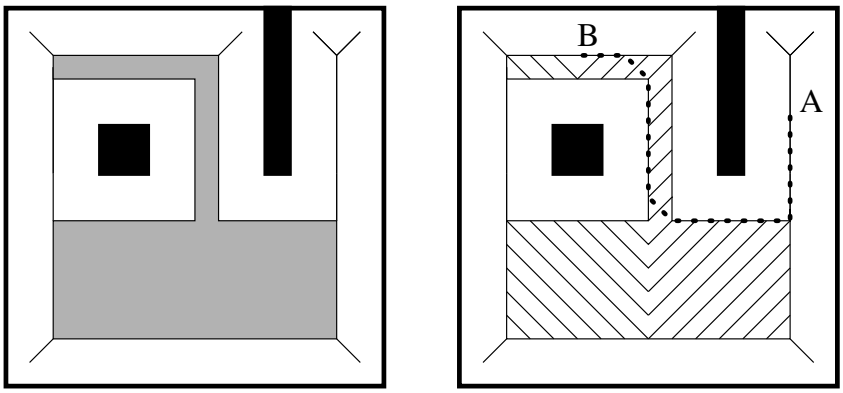


Figure 5.11: Exemple de squelette élargi et de trajectoire générée.

5.3.6 Réduction des temps de calcul

Étant donné notre objectif de calcul en temps réel, on cherche à diminuer les temps de calcul au strict minimum. L'idée générale vise à se servir au maximum des champs de potentiel déjà créés et d'éviter d'étendre l'expansion par vagues dans des zones non-utilisées. Pour ce faire, une stratégie qui évite de créer des champs de potentiel inutilement et réduit les expansions par vague au minimum est instaurée. Trois solutions particulières sont utilisées : une pour la table des distances et deux pour le champ de potentiel attractif.

Pour la table des distances, l'algorithme construit d'abord une *table des distances préparatoire*, qui donne la distance aux frontières d'un environnement vide. Pour chaque nouvel environnement, on se sert de cette table des distances préparatoire comme valeurs initiales. L'expansion n'est effectuée qu'à partir des nouveaux obstacles, et cela, tant que la nouvelle distance d'expansion demeure plus petite que l'ancienne. Le squelette est construit comme avant à l'aide des origines de l'expansion. On obtient le même résultat qu'avec l'ancienne méthode, sauf que le temps de calcul est considérablement diminué, puisqu'il n'y a pas d'expansion à partir du contour de l'espace de travail à chaque nouvel environnement. La figure 5.12 illustre la zone traitée pour un nouvel environnement.

Pour l'implantation, on remarque qu'il faut d'abord faire une copie de la table

des distances préparatoire afin de la garder inacte et de la réutiliser au besoin. Cette copie est toutefois très rapide.

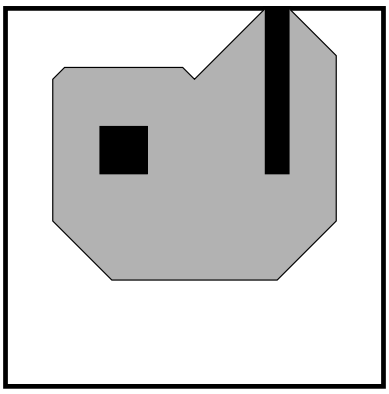


Figure 5.12: Exemple de zone traitée pour la création de la table des distances d'un nouvel environnement.

Pour le champ de potentiel attractif, un champ de potentiel préparatoire *cpp* est d'abord construit à partir de la table des distances L_1. Celui-ci a pour effet d'éloigner un point de contrôle des obstacles. Rapidement construit, ses valeurs sont simplement

$$cpp = K - L_1 \tag{5.3}$$

où K représente une valeur assez grande pour s'assurer que les valeurs d'expansion dans le squelette élargi sont plus petites que les valeurs du champ de potentiel préparatoire. Ce champ de potentiel préparatoire est construit une fois par trajectoire puisqu'il faut nécessairement générer un autre champ pour ne pas affecter l'original, appelé à servir plusieurs fois. L'expansion du champ de potentiel attractif n'est ensuite effectuée que dans la zone du squelette élargi, en ne modifiant qu'une partie du champ de potentiel préparatoire, ce qui est plus court en temps de calcul. Ainsi, si un point de contrôle est situé en dehors du squelette élargi, il est d'abord poussé dans le squelette élargi (c'est-à-dire éloigné des obstacles). Par la suite, il est poussé vers son objectif par le champ de potentiel attractif créé dans le squelette élargi.

Il est possible de diminuer davantage la zone d'expansion à partir de l'objectif en arrêtant celle-ci lorsqu'elle a atteint le point de départ de l'éventuelle trajectoire

de l'organe terminal. Au delà de ce point, le champ de potentiel attractif n'est normalement pas utilisé. Si le point de départ n'est pas dans la zone du squelette élargi, l'algorithme fait voyager le point de départ dans le champ potentiel préparatoire jusque dans le squelette élargi comme si la trajectoire était générée. Le nouveau point ainsi généré devient la nouvelle borne qui arrête l'expansion, puisqu'il est le premier point de la trajectoire dans le squelette élargi. Un exemple de la zone traitée pour la construction du champ potentiel attractif est illustré à la figure 5.13.

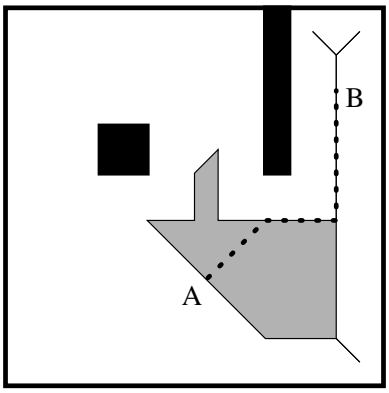


Figure 5.13: Exemple de la zone traitée pour la création d'un champ de potentiel attractif.

Ces dernières modifications permettent de diminuer considérablement les temps de calcul, surtout pour les trajectoires simples, sans altérer significativement les champs de potentiel attractifs par rapport à leur application. Une vue d'ensemble de la structure de construction des différents champs est présentée aux organigrammes des figures 3.2 et 3.3.

5.3.7 Interpolation des champs potentiels discrets

Étant donné le niveau de discrétisation utilisé, on obtient des plateaux de valeurs de potentiel sur des plages assez grandes. Pour éliminer cet effet, on effectue localement une interpolation linéaire (double en 2D et triple en 3D) lorsque c'est nécessaire. Les champs de potentiel peuvent ainsi adopter un aspect continu, malgré la discrétisation

grossière utilisée.

Pour obtenir une double interpolation $inter_2$, on utilise l'équation

$$inter_2 = (a(1-x)+bx)(1-y)+(c(1-x)+dx)y \tag{5.4}$$

qui a été simplifiée pour obtenir

$$inter_2 = bx+(d-b)xy+(a+(c-a)y)(1-x) \tag{5.5}$$

qui demande une multiplication de moins. Les valeurs a, b, c et d sont les valeurs de potentiel aux points servant à l'interpolation, tel qu'illustré à la figure 5.14. Les valeurs x et y sont les coordonnées, de zéro à un (les valeurs zéro et un sont situées au centre des noeuds), de la position du point où on désire la valeur interpolée, tel qu'illustré à la figure 5.14. Un résultat de l'interpolation en 2D est illustré à la figure 5.15.

Pour obtenir une triple interpolation $inter_3$, le concept utilisé en 2D est généralisé en 3D, pour obtenir l'équation

$$\begin{aligned} inter_3 = \quad & ((hx+g(1-x))y+(fx+e(1-x))(1-y))z \\ & +((dx+c(1-x))y+(bx+a(1-x))(1-y))(1-z) \end{aligned} \tag{5.6}$$

qui a été simplifiée pour obtenir

$$\begin{aligned} inter_3 = \quad & ((d+(h-d)z)x+(c+(g-c)z)(1-x))y \\ & +((b+(f-b)z)x+(a+(e-a)z)(1-x))(1-y) \end{aligned} \tag{5.7}$$

qui demande quatre multiplications de moins. Cette simplification est importante car l'interpolation peut être utilisée très souvent (des milliers de fois pour une trajectoire). Les valeurs a, b, c, d, e, f, g et h sont les valeurs de potentiel aux points servant à l'interpolation. Les valeurs x, y et z sont les coordonnées, de zéro à un (les valeurs zéro et un sont situées au centre des noeuds), de la position du point où on désire la valeur interpolée.

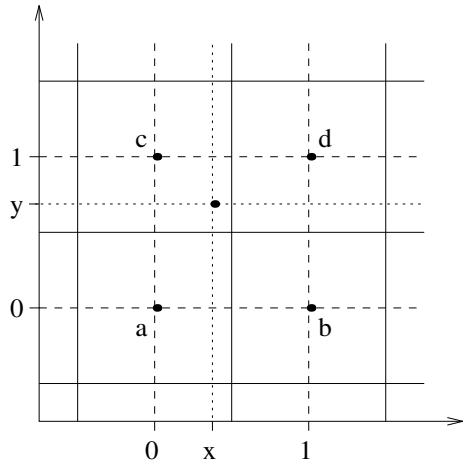


Figure 5.14: Explication de l'interpolation en 2D.

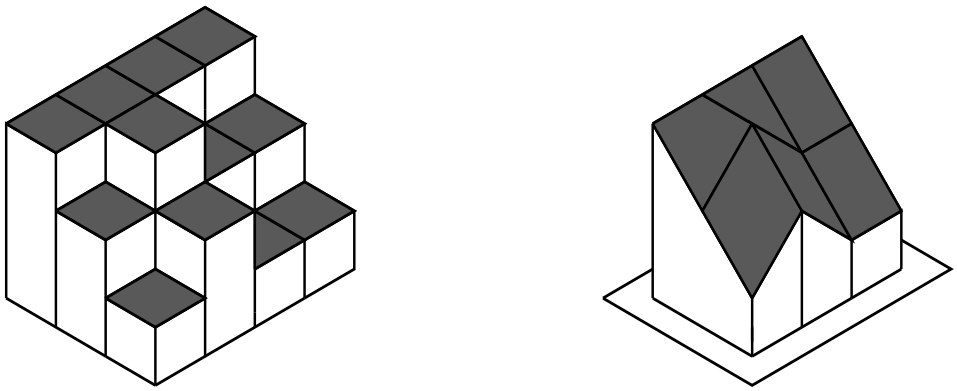


Figure 5.15: Illustration de l'interpolation en 2D.

5.4 Heuristique de contour d'obstacles de type *fil*

Le champ de potentiel qui dirige l'organe terminal vers son objectif permet de trouver une solution pour la plupart des cas d'encombrement moyen. Il y a cependant un cas peu encombré relativement fréquent qu'il ne résoud pas. Supposons la situation décrite à la figure 5.16. Le champ de potentiel tel que décrit jusqu'à maintenant ne tient pas compte que l'algorithme travaille avec un manipulateur et force l'organe terminal à passer par le chemin le plus court pour lui, soit en arrière de l'obstacle, par rapport au robot. Le manipulateur ne contourne pas l'obstacle et l'algorithme reste coincé dans un minimum local d'où il est très difficile de s'échapper. En 3D, une situation similaire est rencontrée lorsqu'il y a des fils, des poteaux, etc. Elle peut donc être très fréquente.

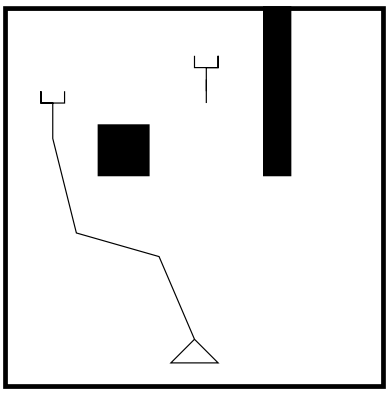


Figure 5.16: Illustration du probleme dû aux obstacles de type *fil*.

Pour réussir la trajectoire, il faut faire passer l'organe terminal devant les obstacles lorsque ceux-ci sont éloignés. L'idée consiste à créer des murs imaginaires derrière les obstacles avant de construire le champ de potentiel attractif. Pour ce faire, on doit d'abord détecter l'arrière des obstacles. Cette détection est réalisée en utilisant le même concept que celui utilisé pour construire le squelette. L'expansion part d'un point à la base du manipulateur ou à une position environnante avantageuse pour le manipulateur en question. Elle n'est effectuée que dans la zone du squelette élargi. Les zones de rencontre de l'expansion sont situées derrière le type d'obstacles qui

cause problème. Une fois l'arrière des obstacles repéré, il est identifié comme étant près des obstacles. Lors de la création du champ de potentiel attractif, ces zones coupent le squelette élargi et ne seront donc pas traversées par la première phase de l'expansion, qui détermine la trajectoire générale de l'organe terminal, tel qu'illustré à la figure 5.17. Celui-ci aura donc tendance à contourner l'obstacle adéquatement. Pour ne pas créer de murs nuisibles lorsque les obstacles sont très près de la base du manipulateur, on ne détecte la rencontre de l'expansion qu'à partir d'une certaine distance du point de départ.

On remarque qu'il ne se crée pas de murs qui empêcheraient l'accès à des zones, puisque les murs artificiels ne sont créés qu'en présence d'au moins deux chemins pour se rendre derrière l'obstacle. De plus, l'objectif peut se situer sur un mur imaginaire sans problème. En effet, c'est l'équivalent d'avoir un objectif situé près d'un obstacle. Le manipulateur va se rendre derrière l'obstacle par le côté le plus court, l'objectif ayant une accessibilité équivalente par les deux côtés.

Ce traitement permet de générer une table des distances avec obstacles imaginaires, une fois par environnement. Elle servira de base à la construction de champs de potentiel attractifs. La table des distances originale est conservée, puisqu'elle sert entre autres pour la vérification de collision.

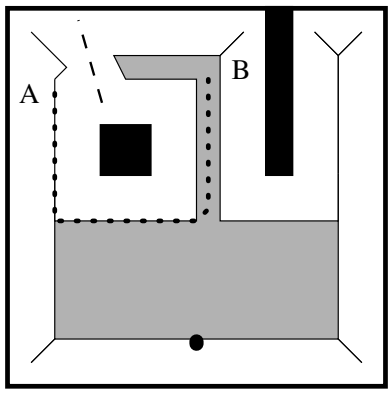


Figure 5.17: Illustration de l'effet du contour des fils.

Lors de l'implantation de l'expansion pour détecter l'arrière des obstacles, on constate qu'une modification s'impose par rapport à ce qui est fait pour créer le

squelette de l'espace de travail. En effet, la distance entre les noeuds d'origine de l'expansion est le critère pour détecter les zones de rencontre de cette expansion. Dans ce cas présent, on n'a qu'un point origine. Il n'y aurait donc jamais de détection de renconte d'expansion puisque la distance entre les noeuds d'origine de l'expansion est toujours nulle. On doit alors garder une liste des ancêtres de chaque noeud et vérifier la distances entre les ancêtres d'une même génération. Cette liste n'a pas besoin de contenir toute la *lignée*. La comparaison des ancêtres à chaque trois générations suffit pour savoir s'il y a rencontre d'expansions.

En trois dimensions, la gestion de cette lignée demande un temps de calcul non-négligeable (environ cinq fois le temps de calcul des autres champs de potentiels). Pour ne pas avoir à gérer cette lignée, l'algorithme effectue donc une expansion à partir d'un plan formé par la frontière entre la zone préparatoire et la zone de travail. L'expansion n'est bien sûr réalisée que dans la zone de travail. On obtient ainsi une approximation acceptable du derrière des murs avec un temps de calcul raisonnable. Ceci est illustré à la figure 5.18.

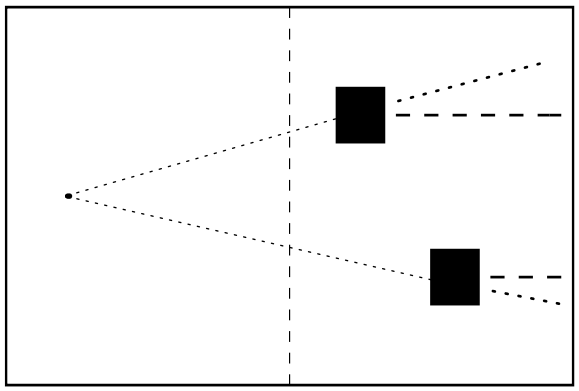


Figure 5.18: Illustration du contour des fils simplifié (la version originale est en pointillé et la version simplifiée est en tirets).

Chapitre 6

Génération de trajectoire

6.1 Stratégie générale de la génération de trajectoire

Une fois les champs de potentiel nécessaires créés, la génération de trajectoire utilise itérativement une routine qui génère les valeurs articulaires de la trajectoire jusqu'à l'atteinte de l'objectif ou le dépassement du nombre d'itérations permises.

La routine effectue les tâches suivantes :

1. Calcul de la variation de position de l'organe terminal.
2. Calcul de la dérivée de la fonction potentielle pour obtenir le vecteur d'optimisation.
3. Calcul de la Jacobienne et de sa pseudoinverse.
4. Calcul des valeurs articulaires à l'aide de la cinématique inverse.
5. Vérification des limites articulaires. Correction apportée et retour à l'étape 4 en cas de problème.
6. Calcul de la cinématique directe.

7. Vérification de collision. Correction apportée et retour à l'étape 4 en cas de problème.

8. Vérification s'il y a situation de blocage. Si oui, algorithme de dégagement et retour à l'étape 1.

9. Validation du nouveau point et vérification de l'atteinte de l'objectif. Si oui, étape 10. Si non, retour à l'étape 1.

10. Lissage de la trajectoire.

11. Ramener la trajectoire dans le domaine temporel.

On note que les étapes 1 à 10 ne tiennent pas compte explicitement du temps lors de l'implantation. Le travail s'effectue plutôt en nombre de pas ou nombre d'itérations, dans un domaine normalisé. On ne considère le temps qu'à l'étape 11, en calculant l'intervalle de temps entre chaque pas de la trajectoire. Pour passer du domaine normalisé au domaine temporel, il n'y a donc qu'un facteur. Dans ce contexte d'explication théorique, les équations sont exprimées dans le domaine du temps.

On peut se référer à l'organigramme de la figure 3.3 pour visualiser la structure de la stratégie de génération de trajectoire.

6.2 Calcul de la variation de pose de l'organe terminal

6.2.1 En position

Le point de contrôle attractif, attaché à l'organe terminal, voyage dans le champ de potentiel attractif. Pour se déplacer dans ce champ, l'algorithme vérifie la valeur des voisins du noeud où se situe présentement le point de contrôle. Parmi ces voisins, il choisit celui possédant le potentiel le plus bas. Une fois le voisin choisi, il calcule une

variation de position (vitesse) de l'organe terminal $\dot{\mathbf{p}}$ qui permet de se déplacer en direction de ce voisin. Pour diminuer le temps de calcul, la variation de position a avantage à être la plus grande possible. Cependant, si elle est trop grande, l'algorithme éprouvera des difficultés dans les situations serrées et il survient un risque de collision entre les pas, surtout après le lissage. Des essais ont démontré que l'algorithme donnait de bons résultats tant que l'organe terminal ne sautait pas par dessus un noeud de discrétisation fine. On permet donc un pas légèrement inférieur à la dimension d'un noeud d'une discrétisation fine, quel que soit le niveau de discrétisation utilisé. Cette stratégie, utilisée à plusieurs reprises, dirige l'organe terminal vers le potentiel minimum, correspondant à l'objectif à atteindre.

6.2.2 En orientation

Ce qui précède présente le déplacement cartésien de l'organe terminal. Pour l'orientation, le champ de potentiel n'est pas utilisé. L'algorithme cherche plutôt à réduire l'écart entre l'orientation du repère attaché à l'objectif et l'orientation du repère attaché à l'organe terminal. L'orientation du repère attaché à l'objectif est donnée par une matrice de rotation $\mathbf{Q}_O$. Celle-ci est spécifiée en utilisant les angles d'Euler.

Soit les rotations successives par les angles β, θ et ψ. La matrice $\mathbf{Q}_O$ est obtenue par [14]

$$\mathbf{Q}_O = \begin{bmatrix} c(\beta)c(\theta) & -c(\beta)s(\theta)c(\psi) - s(\beta)s(\psi) & c(\beta)s(\theta)s(\psi) - s(\beta)c(\psi) \\ s(\theta) & c(\theta)c(\psi) & -c(\theta)s(\psi) \\ s(\beta)c(\theta) & -s(\beta)s(\theta)c(\psi) + c(\beta)s(\psi) & s(\beta)s(\theta)s(\psi) + c(\beta)c(\psi) \end{bmatrix} \tag{6.1}$$

où c(·) représente le cosinus de l'angle et s(·) le sinus de l'angle.

Posons $\mathbf{Q}_R$ la matrice de rotation de l'organe terminal du robot par rapport au repère global et $\mathbf{Q}$ la matrice de rotation, exprimée dans le repère global, qui fait passer l'orientation de $\mathbf{Q}_R$ à $\mathbf{Q}_O$, tel qu'illustré à la figure 6.1. On obtient $\mathbf{Q}$ par

$$\mathbf{Q} = \mathbf{Q}_O \mathbf{Q}_R^T \tag{6.2}$$

Par la suite, les invariants naturels de $\mathbf{Q}$, c'est-à-dire un vecteur unitaire $\mathbf{e}$ dans

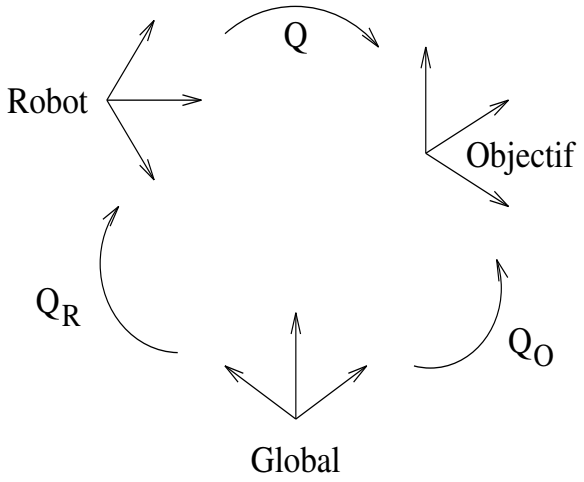


Figure 6.1: Illustration des repères pour le calcul de l'orientation de l'organe terminal.

la direction de l'axe de rotation et l'angle de rotation ϕ, sont calculés comme

$$\text{vect}(\mathbf{Q}) = \mathbf{e} \sin \phi \tag{6.3}$$

et

$$\text{tr}(\mathbf{Q}) = 1 + 2 \cos \phi \tag{6.4}$$

où vect$(\cdot)$ est l'invariant vectoriel de la matrice et tr$(\cdot)$ est la trace de la matrice.

En posant ϕ positif, $\boldsymbol{e}$ est parallèle et de même sens que $\boldsymbol{\omega}$, le vecteur vitesse angulaire de l'organe terminal, qui pousse le repère d'orientation $\mathbf{Q}_R$ vers le repère $\mathbf{Q}_O$.

La vitesse angulaire est donc évaluée comme

$$\boldsymbol{\omega} = k\mathbf{e} \tag{6.5}$$

où k est un facteur d'amplification. Si l'angle de rotation ϕ est petit, alors on met $k = \phi$, ce qui pousse l'organe terminal très près de son orientation finale.

6.2.3 Vecteur résultant

Les deux vecteurs $\dot{\mathbf{p}}$ et $\boldsymbol{\omega}$ forment le vecteur de variation de pose de l'organe terminal (vitesse Cartésienne du manipulateur) $\mathbf{t}$ (voir équation 4.11) qui est inclus dans

le premier terme de l'équation 4.22. Il est aussi possible d'inclure l'orientation de l'organe terminal dans le second terme de l'équation 4.22, tel qu'expliqué à la section suivante. Dans ce cas, le vecteur vitesse cartésienne devient

$$\mathbf{t} = \dot{\mathbf{p}} \tag{6.6}$$

et la Jacobienne

$$\mathbf{J} = \mathbf{J}_A \tag{6.7}$$

6.3 Calcul de la dérivée de la fonction potentielle

6.3.1 Pour l'éloignement des obstacles

Le champ potentiel répulsif utilisé éloigne les points de contrôle des obstacles. Une fonction potentielle p est posée comme étant la somme des valeurs de potentiel répulsif de tous les points de contrôle attachés au manipulateur. Pour calculer la dérivée partielle de la fonction potentielle en fonction d'une variable articulaire i, $\partial p/\partial\theta_i$, il s'agit de faire varier artificiellement la variable articulaire d'une petite valeur positive $+\Delta\theta_i$, puis d'une petite valeur négative $-\Delta\theta_i$ (les autres variables articulaires restant fixes), de calculer la cinématique directe de chacune des configurations fictives, ainsi que la valeur de la fonction potentielle résultante. Il s'agit ensuite de calculer la différence entre les deux valeurs de fonction potentielle, par rapport à la différence entre les deux valeurs articulaires fictives, soit

$$\frac{\partial p}{\partial \theta_i} \simeq \frac{p_{\theta_i+} - p_{\theta_i-}}{2\Delta\theta_i}, \quad i = 1, \ldots, n \tag{6.8}$$

où $p_{\theta_i+} = p(\theta_i + \Delta\theta_i)$ et $p_{\theta_i-} = p(\theta_i - \Delta\theta_i)$.

On obtient le vecteur des dérivées de la fonction potentielle répulsive en répétant cette démarche pour les n valeurs articulaires. Ce vecteur est inséré dans le vecteur d'optimisation $\mathbf{z}$.

Étant donné la discrétisation, il existe des plateaux de valeurs qui peuvent provoquer des dérivées nulles ne correspondant pas à la situation réelle. Pour obtenir un champ de potentiel continu, on effectue l'interpolation linéaire présentée au chapitre 5.

6.3.2 Pour l'orientation

On peut aussi minimiser l'écart entre l'orientation de l'organe terminal et l'orientation de l'objectif, pour tendre vers l'orientation voulue. En pratique, l'algorithme effectue la dérivée de ϕ par rapport à la configuration pour minimiser ϕ, où ϕ correspond à l'angle de rotation entre le repère relié à l'organe terminal et celui relié à l'objectif. On calcule alors $\partial\phi/\partial\theta_i$ de la même manière qu'à l'équation 6.8. Le vecteur ainsi obtenu est ajouté au vecteur obtenu de la dérivée de la fonction potentielle répulsive, dans le vecteur $\mathbf{z}$. Ceci permet de ramener la tâche d'atteinte de l'orientation voulue de l'organe terminal au même niveau que la tâche évitement d'obstacles, dont on verra l'utilité à la section suivante.

6.4 Spécification de l'orientation

La spécification de l'orientation par le vecteur vitesse de l'organe terminal $\mathbf{t}$ limite les possibilités d'évitement d'obstacles de l'algorithme en lui enlevant 3 degrés de liberté, alors qu'il n'est généralement pas nécessaire de spécifier prioritairement l'orientation le long de la trajectoire. Le mode de spécification de l'orientation par le vecteur d'optimisation $\mathbf{z}$, avec $\mathbf{t} = \dot{\mathbf{p}}$ et $\mathbf{J} = \mathbf{J}_A$ est donc utilisé au cours de la trajectoire. L'algorithme peut ainsi spécifier l'orientation de l'organe terminal à un niveau qui permet à l'évitement d'obstacles d'influencer l'orientation de l'organe terminal lorsque c'est nécessaire. Une fois l'objectif cartésien atteint, l'algorithme spécifie l'orientation par le vecteur des vitesses Cartésiennes $\mathbf{t}$ pour s'assurer de l'atteinte exacte et rapide de l'orientation.

L'opérateur peut aussi ne spécifier aucune orientation lorsque ce n'est pas nécessaire, ce qui augmente les chances de succès et diminue les temps de calcul.

6.5 Boucles de correction

À chaque itération, l'algorithme vérifie la nouvelle configuration, car celle-ci n'est pas nécessairement valide. Elle peut provoquer une collision du manipulateur avec les

obstacles ou dépasser les limites articulaires en position ou en variation de position. Lors de la détection d'un de ces problèmes, l'algorithme apporte une correction. La configuration corrigée est ensuite vérifiée à nouveau. Elle n'est pas nécessairement valide, car les corrections peuvent se nuire les unes aux autres ou être un processus itératif. Il y a donc trois boucles de correction imbriquées, soit une pour chaque problème possible. Les boucles dont la vérification est rapide, soit la vérification des limites articulaires, sont à l'intérieur de la boucle de vérification de collision, elle-même plus longue. On minimise ainsi les temps de calcul à ce niveau. L'algorithme sort de la boucle lorsqu'une nouvelle configuration valide est générée ou lorsqu'un nombre limite d'itérations de correction est atteint. Dans ce dernier cas, l'algorithme de base est coincé et une heuristique est alors utilisée.

6.6 Limites articulaires

6.6.1 En position

Pour chaque nouvelle configuration proposée, l'algorithme vérifie si la position articulaire dépasse les limites articulaires du manipulateur. Si tel est le cas, l'algorithme met à zéro la variation de la position des articulations fautives et bloque ainsi les articulations à leur dernière valeur valide. Il y a trois manières principales de bloquer les articulations fautives.

Remplacement de la variation des articulations fautives par zéro

La première consiste à remplacer directement la variation de position des articulations fautives par zéro. Ceci a pour effet d'altérer la trajectoire de l'organe terminal et l'effort d'évitement d'obstacle, ce qui provoque une trajectoire cahotique de l'organe terminal et des difficultés à bien s'éloigner des obstacles.

Recalcul de la cinématique inverse en vitesse avec un manipulateur réduit

La deuxième consiste à recalculer la cinématique inverse en vitesse en n'incluant pas les articulations fautives, la variation de celles-ci étant fixée à zéro. Le blocage des

articulations revient à travailler avec un manipulateur avec n_f degrés de liberté de moins, n_f étant le nombre d'articulations fautives. Ces articulations n'étant pas utiles à ce moment, le manipulateur réduit qui en résulte devient le meilleur disponible. La trajectoire de l'organe terminal n'est pas altérée, et l'évitement d'obstacles travaille de son mieux avec les articulations disponibles, sans être induit en erreur par la mise à zéro ultérieure des articulations fautives.

Pour ce faire, on remplace $\dot{\boldsymbol{\theta}}$ par $\dot{\boldsymbol{\theta}}_R$, $\mathbf{J}$ par $\mathbf{J}_R$ et $\mathbf{z}$ par $\mathbf{z}_R$ dans les équations 4.22 et 4.24, où on a alors

$$\dot{\boldsymbol{\theta}}_R = \left[\dot{\theta}_1, ..., \dot{\theta}_{fi-1}, \dot{\theta}_{fi+1}, ..., \dot{\theta}_n\right]^T \quad \text{pour} \quad i = 1, ..., n_f \tag{6.9}$$

$$\mathbf{J}_R = [\mathbf{j}_1, ..., \mathbf{j}_{fi-1}, \mathbf{j}_{fi+1}, ..., \mathbf{j}_n] \quad \text{pour} \quad i = 1, ..., n_f \tag{6.10}$$

$$\mathbf{z}_R = [z_1, ..., z_{fi-1}, z_{fi+1}, ..., z_n]^T \quad \text{pour} \quad i = 1, ..., n_f \tag{6.11}$$

fi désignant la i^{eme} articulation fautive.

La détection du dépassement d'une limite articulaire ne survenant qu'après un calcul complet d'une nouvelle configuration, on ne peut effectuer ce calcul qu'après un premier essai. Cette méthode s'avère longue en temps de calcul, car il faut recalculer presque toute la cinématique inverse à chaque fois qu'une limite articulaire est dépassée.

Recalcul du vecteur d'optimisation

Le recalcul du vecteur d'optimisation est une méthode qui permet d'obtenir le même effet que la deuxième méthode. Cette autre méthode est toutefois moins longue en temps de calcul, bien que conceptuellement plus complexe.

Soit θ_{fi}, une articulation fautive pour $i = 1$ à n_f. Pour bloquer les n_f articulations fautives sans altérer l'atteinte des objectifs, l'algorithme remplace les éléments z_{fi} (avec $i = 1$ à n_f) du vecteur $\mathbf{z}$ par une valeur qui permet de mettre à zéro la variation de position des n_f articulations fautives, soit $\dot{\theta}_{fi} = 0$ pour $i = 1$ à n_f.

Les éléments remplacés sont les z_{fi} qui correspondent aux $\dot{\theta}_{fi}$ que l'on veut mettre à zéro, puisque ceux-ci n'ont plus de signification pour diminuer la fonction potentielle.

En effet, z_{fi}, qui représente la variation de p pour une variation de θ_{fi}, n'a plus de signification puisque $\dot{\theta}_{fi}$ est justement fixée à zéro.

Ceci permet de trouver une nouvelle solution qui se trouve dans le noyau de la matrice Jacobienne. La trajectoire de l'organe terminal n'est donc nullement modifiée. Les éléments non modifiés du vecteur $\boldsymbol{z}$ cherchent à éloigner le robot des obstacles avec les degrés de liberté restants. Tout comme pour la deuxième méthode, la combinaison des degrés de liberté restants devient la meilleure disponible, puisque ce sont les seuls degrés de liberté encore utiles.

Pour calculer les valeurs des z_{fi}, l'algorithme extrait les n_f lignes de l'équation 4.22 qui calculent les $\dot{\theta}_{fi}$. Il pose $\dot{\theta}_{fi} = 0$ pour $i = 1$ à n_f et les z_{fi} pour $i = 1$ à n_f sont les inconnues. On obtient donc un système linéaire

$$\mathbf{Ax} = \mathbf{b} \tag{6.12}$$

à n_f équations et n_f inconnues qui se résoud aisément.

En pratique, le système linéaire à résoudre s'obtient ainsi : soit l'équation 4.22 réécrite sous la forme

$$\dot{\boldsymbol{\theta}} = \mathbf{d} + \mathbf{Nz} \tag{6.13}$$

Les éléments du système d'équations linéaire $\mathbf{Ax} = \mathbf{b}$ sont

$$A_{ij} = N_{fi\ fj} \quad \text{pour} \quad i = 1, ..., n_f \quad \text{et} \quad j = 1, ..., n_f \tag{6.14}$$

$$x_i = z_{fi} \quad \text{pour} \quad i = 1, ..., n_f \tag{6.15}$$

$$\begin{aligned} b_i = \quad & -d_{fi} - (N_{fi\ 1}\ z_1 + ... + N_{fi\ fi-1}\ z_{fi-1} + N_{fi\ fi+1}\ z_{fi+1} + ... + N_{fi\ n}\ z_n) \\ & \text{pour} \quad i = 1, ..., n_f \end{aligned} \tag{6.16}$$

où fi désigne la i^{eme} articulation fautive.

On note que cette troisième méthode permet plus de flexibilité que la deuxième. En effet, on pourrait spécifier une autre valeur que zéro pour $\dot{\theta}_{fi}$. Par exemple, on peut vouloir $\dot{\theta}_{fi}$ pour positionner θ_{fi} à mi-chemin entre la dernière valeur valide de θ_{fi} et la limite articulaire de l'articulation fi.

En pratique, le blocage d'une première articulation peut provoquer le dépassement d'une seconde articulation. À l'itération de correction suivante, la seconde articulation va généralement provoquer le dépassement de la première articulation à son tour, puisque celle-ci se situe très près de sa limite articulaire. L'algorithme reste alors coincé dans la boucle parce qu'il demande à une articulation non-disponible (puisque sur le point d'être hors limites) de participer à la cinématique inverse. Pour éviter ce phénomène, une articulation bloquée reste ainsi bloquée pendant plusieurs itérations de correction ou jusqu'à ce qu'une configuration valide soit trouvée.

6.6.2 En vitesse (variation de position)

On limite les variations de position pour éviter d'avoir des mouvements démesurés. Les vitesses articulaires maximales (dans le domaine temporel) ne sont considérées que lorsque l'on détermine t, l'intervalle de temps entre les pas. Des mouvements démesurés obligeraient à utiliser un intervalle de temps très long pour satisfaire aux limites de vitesses lors de ces mouvements démesurés, ce qui ralentirait la trajectoire globalement. Ceci est aussi vrai pour des articulations avec des limites de vitesse différentes. Dans ce cas, il devient avantageux de poser des limites de variation de position en proportion avec les limites de vitesse. De plus, des pas démesurés ne pourraient plus garantir l'absence de collision entre deux pas de la trajectoire.

Pour chaque nouvelle configuration proposée, l'algorithme vérifie si la variation de position de chaque articulation dépasse les limites permises. Si c'est le cas, l'algorithme calcule un ratio de dépassement entre la variation de position proposée et la variation de position permise. Une modification du module du vecteur variation de position des articulations change seulement la vitesse d'atteinte des objectifs et non pas les objectifs eux-mêmes. Pour s'assurer que toutes les articulations sont à l'intérieur des variations permises, il s'agit de diviser la variation de position de chaque articulation par le ratio de dépassement le plus élevé. En pratique, on réduit simplement C_1 et C_2, les coefficients définis à l'équation 4.22. Ainsi, la plus grande variation de position (par rapport à sa limite) sera à la limite de sa vitesse permise, alors que les autres articulations seront à l'intérieur de leur limite permise. Cette correction est

annulée lorsqu'une autre correction est effectuée, car l'effet cumulatif dans la boucle de correction donne des valeurs de variation de vitesse articulaire inutilement petites. Le peu de calculs nécessaire rend cette modification très rapide.

6.6.3 En accélération

Tout comme la vitesse, l'accélération articulaire n'est considérée que lors du calcul de l'intervalle de temps t entre les pas. L'accélération est liée à la variation de la variation de la position articulaire. Les limites articulaires en accélération étant généralement les limites les plus contraignantes, on a de fortes raisons de croire que celles-ci le sont aussi dans cette application. Les accélérations sont significativement réduites lors du lissage de la trajectoire qui sera présenté plus loin.

6.7 Vérification de collision

Lors de la génération de trajectoires, il est parfois possible que les configurations proposées fassent entrer le manipulateur en collision avec les obstacles. Puisque l'algorithme doit s'assurer qu'il n'y ait jamais de collision, il vérifie la validité de chaque nouvelle configuration proposée avant de l'accepter. Si la vérification confirme l'absence de collision, la nouvelle configuration est acceptée. À la détection d'une collision, la nouvelle configuration est rejetée et l'algorithme recherche une autre configuration acceptable en modifiant la suggestion initiale. Il est possible que l'algorithme reste coincé dans un minimum local, mais on a la certitude qu'il n'y aura jamais de collisions, si l'environnement n'a pas évolué depuis la dernière prise d'information par le système de vision 3D.

Le manipulateur est modélisé par des points de vérification. Ceux-ci sont localisés le long des segments, entre les repères attachés au manipulateur. Les points sont générés entre chaque repère en tenant compte d'un intervalle maximum L_{max} entre eux et de la distance entre les repères. Ceci permet de générer des points pour un manipulateur théorique dont les membres longent les segments entre les repères, tel que le bras Sarcos.

On note qu'il serait possible d'avoir une modélisation plus détaillée, pour un manipulateur de forme quelconque, en ajoutant des points de vérification spécifiques, par des modifications au programme. Puisque la modélisation simple permet l'étude du bras Sarcos et de beaucoup de manipulateurs théoriques, cette modélisation détaillée n'a pas été implantée. Le programme pourrait être modifié si le besoin s'en faisait sentir.

À chacun des points de vérification est associée une distance minimale aux obstacles à respecter R. Ceci crée des sphères de sécurité autour de points de vérification. L'ensemble de ces sphères constitue une enveloppe de sécurité le long des segments du manipulateur qui englobe complètement celui-ci, tel qu'illustré à la figure 6.2.

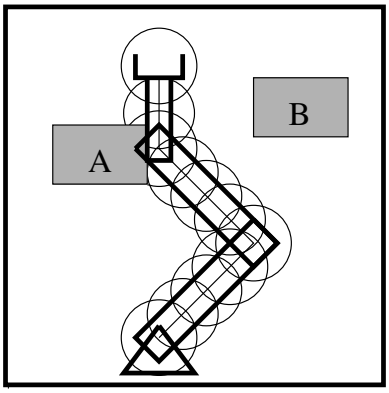


Figure 6.2: Illustration des sphères de sécurité pour la vérification de collision.

Pour le paragraphe suivant, on se réfère à la figure 6.3. La grosseur des sphères est ajustée selon le manipulateur utilisé et la marge de sécurité voulue. Il y a des *creux* entre les sphères qui causent des variations de la distance de sécurité de l'enveloppe S. Ces variations obligent à prendre des sphères de rayon R plus grand que la distance de sécurité minimale S_{min}, ce qui diminue inutilement l'espace libre du manipulateur. On cherche à minimiser cet effet en gardant l'intervalle entre le centre des sphères L_{max} suffisamment petit. Ici, on considère que cet effet est raisonnablement faible lorsque $R = 1,1S_{min}$. Il faut aussi tenir compte de cet effet lors de la détermination

du rayon des sphères de sécurité. La relation entre L_{max}, R et S_{min}, soit

$$R^2 = S_{min}^2 + \frac{L_{max}^2}{4} \tag{6.17}$$

est utile lors du choix des paramètres. Par exemple, si la situation demande $S_{min} = 10$ et que l'on veut respecter $R = 1,1S_{min}$, alors $R = 11$ et on obtient $L_{max} = 9,17$ par l'équation 6.17.

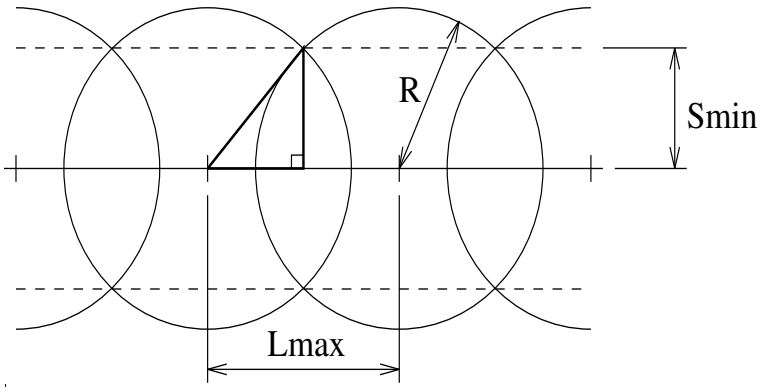


Figure 6.3: Illustration des valeurs L_{max}, R et S_{min}.

Pour effectuer la vérification de collision, l'algorithme évalue la distance aux obstacles des différents points de vérification en obtenant la valeur de chacun des points dans la table des distances. On peut garantir que le manipulateur n'est pas en collision avec un obstacle si tous les points de vérification sont à une distance aux obstacles plus grande que leur distance minimale à respecter, puisque les obstacles se trouvent alors à l'extérieur des sphères de sécurité. Dès qu'un point de vérification se situe à une distance d'un obstacle inférieure à sa distance minimale à respecter, on considère qu'il y a collision. À la figure 6.2, on considère l'obstacle A en collision avec le manipulateur alors que l'obstacle B ne l'est pas.

S'il y a détection de collision, c'est que la partie de calcul qui permet de s'éloigner des obstacles n'a pas eu assez d'importance par rapport à la partie qui permet d'atteindre le point suivant. Pour corriger la situation, il s'agit donc de pondérer plus fortement la partie permettant de s'éloigner des obstacles (augmenter C_2) et

de pondérer moins fortement celle permettant de se diriger vers la position finale (diminuer C_1). Ceci entraîne l'amélioration de la situation sans avoir à refaire tous les calculs de cinématique inverse. Ainsi, le robot s'éloignera plus vigoureusement des obstacles, et l'organe terminal n'effectuera qu'une partie de l'élément de trajectoire prévu, en conservant la direction voulue.

En plus des collisions avec les obstacles, il est possible que le robot entre en collision avec lui-même. Souvent, l'architecture d'un manipulateur et ses limites articulaires empêchent ce problème et il n'est pas néssessaire d'en tenir compte. Mais si le manipulateur est très redondant ou que l'effecteur maintient une pièce de grandes dimensions, il est possible qu'il y ait collision.

Pour les prévenir, on utilise les sphères de sécurité. Il s'agit de s'assurer que les sphères de membres non-consécutifs du manipulateur ne s'intersectent pas. On désigne par membre, le segment entre deux repères consécutifs dont la longueur n'est pas nulle. Un objet maintenu par l'effecteur est aussi considéré comme un membre. Deux sphères ne s'intersectent pas si la distance entre les centres des sphères, dont les positions sont C_1 et C_2, est plus grande que la somme de leur rayons R_1 et R_2, soit

$$\sqrt{(c_{2x} - c_{1x})^2 + (c_{2y} - c_{1y})^2 + (c_{2z} - c_{1z})^2} > R_1 + R_2 \tag{6.18}$$

où c_{ix}, c_{iy} et c_{iz} représentent les coordonnées du centre de la sphère i, pour $i = 1, 2$. Pour éviter la racine carrée, longue à calculer, on utilise

$$(c_{2x} - c_{1x})^2 + (c_{2y} - c_{1y})^2 + (c_{2z} - c_{1z})^2 > (R_1 + R_2)^2 \tag{6.19}$$

ce qui est équivalent.

On est ainsi assuré qu'aucun membre ne peut entrer en collision. Il est à noter que deux membres consécutifs, qui se touchent naturellement par l'articulation les liant, ne peuvent entrer en collision si les limites articulaires sont respectées.

Si une collision est détectée, il s'agit d'inclure une fonction qui éloigne l'une de l'autre les deux sphères qui s'intersectent. Cette fonction est simplement la distance entre les sphères trop rapprochées que l'on dérive comme la fonction qui sert à s'éloigner des obstacles. La fonction de répulsion entre les membres n'est activée que lorsqu'il y a risque imminent de collision. On veut ainsi maintenir au minimum les

calculs reliés à cette action de répulsion. De plus, l'effet de cette répulsion mutuelle est inutile lorsque deux sphères sont très distancées.

6.8 Heuristiques

6.8.1 Vérifications préliminaires

Lorsque l'utilisateur demande au système de planification de trajectoire une pose que le manipulateur ne peut atteindre, le système va quand même chercher une solution et ne va constater l'échec qu'après avoir dépassé le nombre de pas alloué, ce qui est relativement long. Il est donc intéressant de détecter dès le départ l'inaccessibilité de la pose demandée. Il est impossible de déterminer à tout coup s'il y a une solution, mais il est possible de détecter cette inaccessibilité pour des cas simples. Ici, deux cas sont étudiés.

Le premier cas détecte si l'objectif est trop près ou en collision avec un obstacle. Cette situation est possible, car l'objectif peut passer par les obstacles, pour éviter à l'opérateur d'avoir à contourner ceux-ci. Ceci est réalisé en effectuant une vérification de collision du point de contrôle attractif lorsque l'objectif est une position. Si l'orientation est aussi spécifiée, alors la vérification de collision s'effectue sur l'organe terminal imaginaire au complet.

Le deuxième cas détecte si l'objectif se trouve dans l'espace atteignable du manipulateur, en supposant qu'il n'y ait pas d'obstacle. L'espace atteignable du manipulateur est modélisé par un volume. Ici, une sphere modélise l'espace atteignable du bras Sarcos. C'est une modélisation inexacte, mais elle permet la détection rapide de plusieurs cas. La position du centre et les rayons de la sphère sont calculés préalablement. Le centre de la sphère se situe à l'épaule du bras Sarcos. Il s'agit de calculer la distance entre la position du centre de la sphère et la position de l'objectif, et de la comparer au rayon de la sphère. Si la distance est plus grande que le rayon, l'objectif est alors hors d'atteinte. Si l'objectif est une position de l'organe terminal, la distance entre le point de contrôle attractif attaché à l'extrémité de l'organe terminal et le centre de la sphère est comparée à un rayon modélisant la distance maximale atteignable

par l'extrémité de l'organe terminal du manipulateur. Si l'objectif est en position et orientation, la distance entre le point au poignet du manipulateur et le centre de la sphère est comparée à un rayon modélisant la distance maximale atteignable par le poignet du manipulateur, ce qui modélise en quelque sorte un espace atteignable agile.

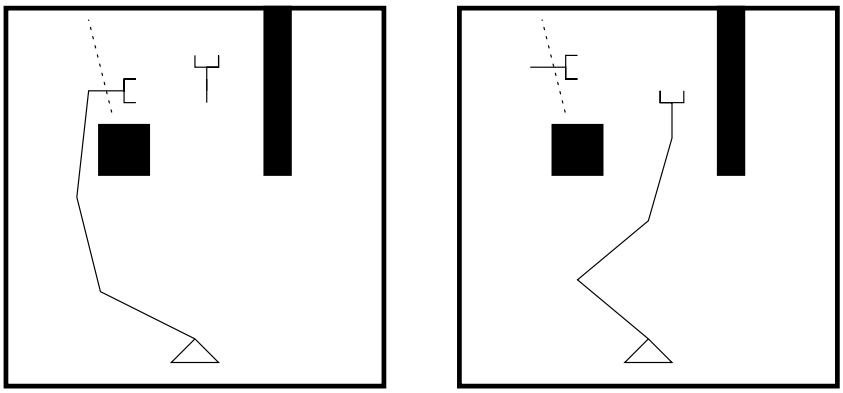
Un volume différent pourrait remplacer la sphère pour des robots dont l'espace atteignable a une forme très différente d'une sphère. Par exemple, on utiliserait un cylindre pour un manipulateur Scara.

6.8.2 Contour des obstacles de type *fil*

Dans la section sur les champs de potentiel, une heuristique permettant de contourner des fils a été présentée. Utilisée tel quel, l'heuristique comporte deux cas pathologiques.

Le premier est illustré à la figure 6.4a. Si le manipulateur traverse un mur à sa configuration initiale, il sera porté à s'enrouler autour de l'obstacle. Cette situation se détecte rapidement en vérifiant s'il y a intersection entre le manipulateur et le mur imaginaire. Si tel est le cas, l'algorithme demande au manipulateur de se rendre à un point lui permettant de sortir du piège, situé le long de la chaîne cinématique de sa configuration initiale, avec un champ potentiel attractif qui ne tient pas compte des murs imaginaires. Une fois sorti du piège, l'algorithme peut se rendre à l'objectif en tenant compte des murs imaginaires.

Le deuxième cas, présenté à la figure 6.4b, se produit lorsque l'on veut spécifier l'orientation de l'objectif de l'organe terminal alors que celui-ci intersecte un mur artificiel. En effet, l'organe terminal peut atteindre sa position, mais il est très mal placé pour atteindre la bonne orientation. On voit que le manipulateur aurait dû passer du côté du poignet. L'algorithme peut encore une fois détecter rapidement le piège en vérifiant s'il y a intersection entre l'objectif et le mur imaginaire. Si tel est le cas, on demande au manipulateur de se rendre au poignet au lieu de se rendre à son objectif. Par la suite il se rend à son objectif à l'aide d'un champ potentiel qui ne tient pas compte des murs imaginaires.



a b

Figure 6.4: Illustration des deux cas pathologiques reliés au contour des fils.

6.8.3 Maintien de la base en retrait

Pour les manipulateurs qui ont une articulation prismatique permettant un mouvement avant-arrière à leur base, l'expérience a démontré qu'il est avantageux de maintenir cette articulation en retrait le plus possible. Ceci maintient le robot dans une meilleure configuration et diminue les oscillations de la base.

Pour ce faire, une fonction qui pousse la base vers l'arrière est insérée dans le vecteur **z**. Plus l'organe terminal est en retrait et plus la fonction pousse la base en retrait.

6.8.4 Heuristiques de dégagement à des cas problèmes typiques

Lorsque le manipulateur se situe près des obstacles, ou dans des situations serrées, le coude est souvent mal placé pour exécuter la tâche ou encore, la trajectoire de l'organe terminal empêche la cinématique inverse de trouver une solution.

Pour ces cas, il s'avère possible de construire des stratégies particulières pour trouver une solution.

La stratégie générale est la suivante:

1. Détecter la présence d'un problème

2. Rassembler l'information disponible sur la situation problématique
3. Déterminer le cas problème à partir de l'information disponible
4. Effectuer la correction appropriée selon le cas problème
    - Déterminer à partir de quel pas de la trajectoire le robot repart pour faciliter l'action de l'heuristique
    - À partir d'une banque d'actions créée à l'avance, déterminer la séquence d'actions pertinentes à chaque cas
    - Déterminer quand changer d'action à l'aide d'une condition à réaliser
    - À mesure que les pas de dégagement sont générés, effectuer les vérifications et corrections nécessaires pour satisfaire les contraintes de collisions et de limites articulaires
5. Construire un nouveau champ de potentiel attractif pour continuer la trajectoire en mode normal. Ce dernier est reconstruit, car le nouveau point de départ peut être à l'extérieur de la zone attractive traitée.

Toutes les actions sont réalisées à l'aide de fonctions potentielles. Chacune des articulations est poussée individuellement dans la direction qui aide à accomplir l'action de dégagement.

Un exemple d'application de stratégie est donné en annexe B pour le bras Sarcos avec glissière à la base.

6.9 Lissage des trajectoires articulaires

Les valeurs articulaires générées par l'algorithme présenté donnent une trajectoire courte, sans mouvements cahotiques, mais qui comporte des changements de direction brusques et des oscillations pouvant causer de fortes accélérations et des difficultés pour le robot à suivre la trajectoire. Tel que mentionné précédemment, on a de fortes raisons de croire que l'accélération est la limite la plus contraignante. Ainsi,

les capacités du manipulateur en vitesses articulaires sont sous-utilisées. Pour qu'un robot réel puisse accomplir la trajectoire avec une vitesse raisonnable, l'algorithme doit d'abord diminuer ces mouvements brusques en effectuant un lissage. Celui-ci modifie légèrement les valeurs articulaires pour adoucir la trajectoire. Le lissage est de la forme

$$l_i = \frac{p_1 x_{i-2} + p_2 x_{i-1} + p_3 x_i + p_4 x_{i+1} + p_5 x_{i+2}}{\sum_{j=1}^{5} p_j} \quad \text{pour} \quad i = 1, ..., n_p \tag{6.20}$$

où, pour le i^{eme} point de la trajectoire, l_i correspond à une valeur articulaire lissée, x_i à une valeur articulaire avant lissage, les p_j, pour $j = 1, ..., 5$, à des facteurs de pondération et n_p au nombre de pas de la trajectoire.

Au début et à la fin de la trajectoire, le manipulateur part du repos. Pour reproduire cette situation, plusieurs pas dont les valeurs sont les mêmes que la première et la dernière configuration sont ajoutés à chaque extrémité, avant qu'il y ait lissage. Pour obtenir un lissage aux extrémités équivalent au lissage le long de la trajectoire, il faut lisser deux pas avant la première configuration et deux pas après la dernière configuration. Il y a donc quatre pas de plus que pour la trajectoire initiale.

La sélection des p_j est très importante pour obtenir de bons résultats. Pour faciliter l'étude du lissage, un programme a été réalisé. Il permet de lisser une séquence de valeurs quelconque avec les paramètres p_1 à p_5 quelconques. Il affiche les séquences sans et avec lissage, les variations de variations de position (accélération) et l'écart entre les séquences avec et sans lissage. Finalement, il calcule la variation de variation de position maximale et l'écart maximal entre les séquences avec et sans lissage. La figure 6.5 illustre un exemple de résultat fourni par le programme de lissage.

La sélection des p_j est faite selon trois contraintes :

- Le lissage est symétrique, c'est-à-dire que

$$p_1 = p_5 \tag{6.21}$$

et

$$p_2 = p_4 \tag{6.22}$$

- Diminution des oscillations, particulièrement celles de premier ordre. L'étude du lissage a permis de déterminer une règle, que la combinaison doit respecter, qui permet d'éliminer les oscillations de premier ordre, soit

$$p_1 + p_3 + p_5 = p_2 + p_4 \tag{6.23}$$

Elle tient aussi compte de deux critères :

- Réduire la variation de la variation de position articulaire, qui influence directement l'accélération.
- Garder un écart entre la trajectoire lissée et la trajectoire non-lissée assez petit pour éviter de causer des collisions après le lissage.

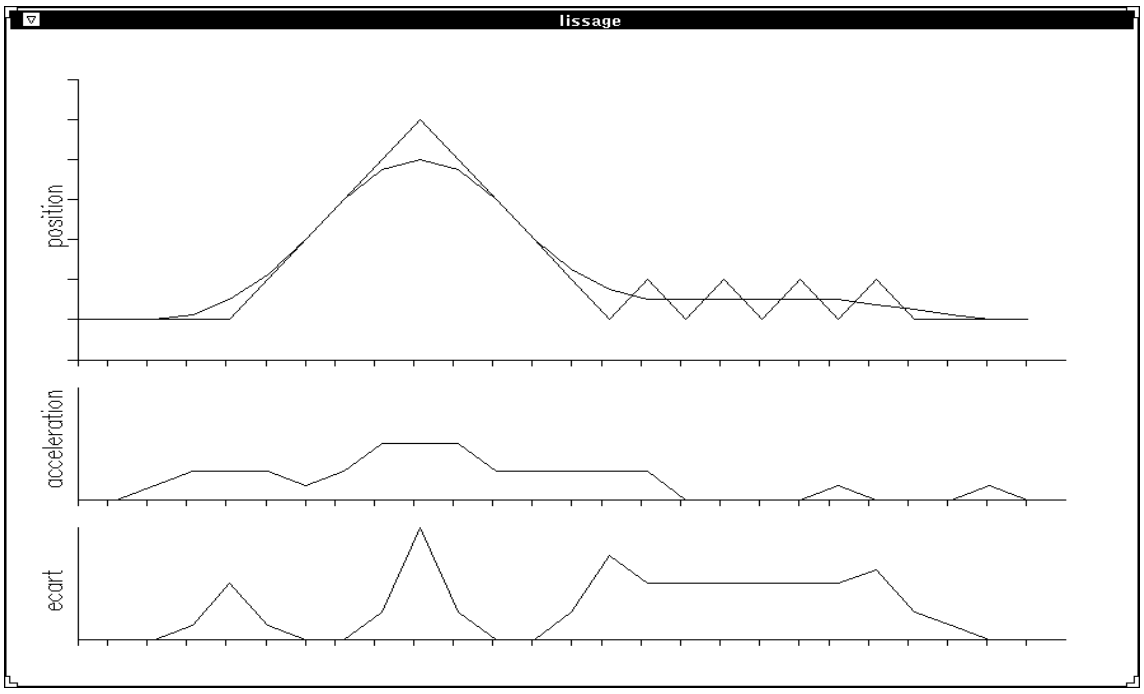


Figure 6.5: Exemple de trajectoire typique lissée, à l'aide du programme de lissage, avec les paramètres (1,2,2,2,1).

Pour déterminer les coefficients, on a utilisé la procédure suivante. Tout d'abord, on constate que ces coefficients ont des valeurs relatives. Par exemple, la combinaison (1,3,4,3,1) aura le même effet que la combinaison (2,6,8,6,2). On peut donc poser $p_1 = p_5 = 1$ sans perdre de généralité. On remarque ainsi qu'il ne reste qu'un degré de liberté à cause des équations 6.22 et 6.23. On peut ainsi effectuer des essais représentatifs en ne faisant varier qu'un seul paramètre, par exemple p_3. Les résultats des essais, pour la pire situation possible, sont visualisés à la figure 6.6. On observe d'abord qu'il est inutile de choisir $p_3 < 2$ puisque l'écart augmente sans diminuer l'effet sur les accélérations. Les autres combinaisons possibles sont des compromis entre les deux critères. On peut donc utiliser la combinaison minimisant l'effet sur l'accélération, quitte à utiliser une combinaison avec un p_3 plus élevé s'il y a détection de collision après lissage. Comme on peut le voir, la combinaison des p_j donnant les meilleurs résultats selon les critères est

$$p_1 = 1 \quad p_2 = 2 \quad p_3 = 2 \quad p_4 = 2 \quad p_5 = 1 \tag{6.24}$$

Un exemple de l'effet du lissage avec ces coefficients est illustré à la figure 6.5. Dans ce cas, la variation de la variation de position est réduite par un facteur quatre pour le pire cas. L'écart maximum entre la trajectoire lissée et la trajectoire non-lissée équivaut à une fois la valeur de la variation maximum. En pratique, ces coefficients donnent de meilleurs résultats que les coefficients choisis auparavant par essai et erreur, tels que (1,2,3,2,1).

Le lissage effectué sur les valeurs articulaires permet naturellement d'obtenir une trajectoire de l'organe terminal adéquatement adoucie.

Puisque la trajectoire est modifiée, l'algorithme doit effectuer une seconde vérification de collision pour des raisons de sécurité. Il ne se produit pratiquement jamais de collision avec les paramètres utilisés et si les pas de l'organe terminal sont raisonnablement petits. Toutefois, si le lissage provoque une collision, il est possible de diminuer son effet en augmentant p_2, p_3 et p_4 tout en respectant l'équation 6.23, tel que mentionné plus tôt. S'il y a quand même collision, ce qui est très peu probable, le système va simplement avertir l'opérateur du problème et considérer qu'il s'agit d'un cas d'échec.

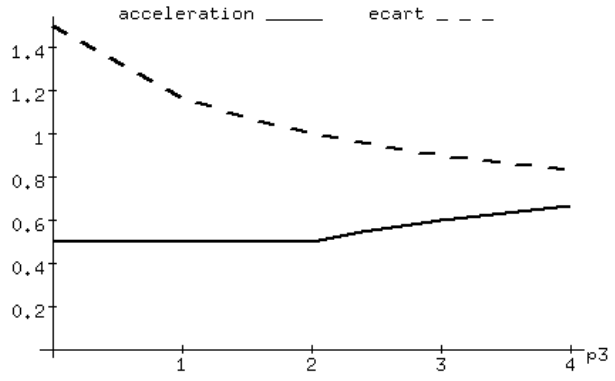


Figure 6.6: Graphique permettant de sélectionner les coefficients du lissage.

6.10 Passage au domaine temporel

La génération de trajectoire décrite jusqu'à maintenant ne tient pas compte du temps. Tout au plus, on émet l'hypothèse que l'intervalle de temps entre les pas sera constant. On cherche à minimiser la durée de la trajectoire. Donc, on a avantage à prendre l'intervalle de temps entre les pas, t, le plus petit possible. Cependant, le manipulateur a des limites en vitesse et en accélération qui demandent un intervalle de temps minimum. Tel que mentionné précédemment, il est justifié de supposer que les limites en accélération sont les plus contraignantes et que les articulations du manipulateur sont sous-utilisées en vitesse. C'est dans ce contexte que t_{min} est calculé. Puisque la notion de temps est explicitement abordée dans cette section, la notation distinguera les grandeurs qui comportent ou non le temps, contrairement à ce qui précède.

Tout d'abord, la relation entre la variation de position maximum $\Delta\theta_{i,max}$ et la vitesse maximum $\dot{\theta}_{i,max}$ est

$$\dot{\theta}_{i,max} = \frac{C_v \ \Delta\theta_{i,max}}{t_{i,min}} \tag{6.25}$$

et la relation entre la variation de la variation de position maximum $\Delta(\Delta\theta)_{i,max}$ et l'accélération maximum $\ddot{\theta}_{i,max}$ est

$$\ddot{\theta}_{i,max} = \frac{C_a \ \Delta(\Delta\theta)_{i,max}}{t_{i,min}^2} \tag{6.26}$$

pour la trajectoire d'une articulation i donnée, en valeur absolue. Les coefficients C_v et C_a sont des valeurs qui prennent en compte que la vitesse et l'accélération locales peuvent être plus grandes que les vitesses et accélérations moyennes entre deux pas. Ces valeurs sont principalement dictées par la façon dont le contrôleur agit. Elles peuvent aussi inclure un facteur de sécurité.

Les valeurs $\dot{\theta}_{i,max}$ et $\ddot{\theta}_{i,max}$ constituent les limites du manipulateur utilisé et sont considérées connues. Les valeurs $\Delta\theta_{i,max}$ et $\Delta(\Delta\theta)_{i,max}$ sont calculées lors du lissage de la trajectoire et sont donc connues. On note que le lissage permet de diminuer significativement C_v, C_a et $\Delta(\Delta\theta)_{max}$. Cela permet de diminuer $t_{i,min}$ et d'équillibrer l'utilisation des capacités du manipulateur, surtout pour mieux exploiter ses capacités en vitesse, lorsque l'accélération est limitative.

Les $t_{i,min}$ étant les seules inconnues, ils sont aisément obtenus. Finalement, t_{min} est calculé comme

$$t_{min} = \max_i \ (t_{i,min}) \tag{6.27}$$

pour satisfaire toutes les limites du manipulateur.

On note que d'autres limites, telles que l'accélération maximale de l'organe terminal pourraient aisément être prises en considération de la même manière que les limites articulaires.

6.11 Commande partagée

On peut utiliser des algorithmes de génération de trajectoire pour créer un mode de commande partagée. Celui-ci consiste à laisser l'opérateur commander l'organe terminal au lieu de faire appel au champ de potentiel attractif. L'algorithme se charge alors de calculer la cinématique inverse, d'éviter les collisions et de gérer les limites articulaires, tel que présenté précédemment.

Pour obtenir un contrôle du manipulateur aussi complet qu'en mode articulaire, on peut laisser à l'opérateur la capacité de modifier la position du coude. Pour ce faire, un terme supplémentaire ajouté au vecteur $\mathbf{z}$ déjà présent, modifie la position du coude. Ceci facilite la résolution des situations que la stratégie de planification de trajectoire ne peut résoudre.

Selon les besoins, l'opérateur peut vouloir conserver ou non l'orientation de l'organe terminal lorsqu'il modifie la position de celui-ci ou qu'il bouge le coude. Les termes qui peuvent influencer l'orientation, soit l'évitement de collision et le coude, sont dans le noyau de la matrice Jacobienne $\mathbf{J}$. La spécification de l'orientation dans $\mathbf{J}$ lui permet donc de ne pas être influencée par les actions $\mathbf{z}$. Il s'agit donc d'utiliser $\mathbf{J} = \mathbf{J}_A$ dans les calculs de la cinématique inverse pour laisser l'orientation libre, et d'utiliser les deux blocs $\mathbf{J}_A$ et $\mathbf{J}_B$ dans $\mathbf{J}$ pour fixer l'orientation.

L'algorithme de lissage est utilisé lors de la commande partagée. Il permet d'obtenir les mêmes avantages que pour la commande automatique. On note que l'action demandée par l'opérateur est envoyée au contrôleur deux pas plus tard à cause du lissage. On doit faire attention d'utiliser un pas assez petit pour s'assurer que le lissage ne provoque pas de collision. L'algorithme travaillant en temps réel, il n'a pas le temps de corriger un problème après le lissage s'il n'y a que deux pas de retard.

Tout comme la commande automatique, la commande partagée subit les limites de vitesses et accélérations maximum atteignables du manipulateur. Puisque l'on n'est pas dans le contexte d'une trajectoire finie, $\Delta\theta_{i,max}$ et $\Delta(\Delta\theta)_{i,max}$ ne sont pas connus à l'avance. On doit donc utiliser la variation de position maximale globale $\Delta\theta_{i,max,g}$ et la variation de la variation de position globale $\Delta(\Delta\theta)_{i,max,g}$, qui tiennent compte de la pire situation possible, dans les équations 6.25 et 6.26. La valeur $\Delta\theta_{i,max,g}$ est fournie par le fichier de description du robot. Lors de la génération de trajectoire, l'algorithme s'assure de ne jamais dépasser cette valeur. La valeur $\Delta(\Delta\theta)_{i,max,g}$ utilisée dépend de l'effet du lissage dans la pire situation possible. Pour un lissage avec les paramètres (1,2,2,2,1), on a

$$\Delta(\Delta\theta)_{i,max,g} = \frac{1}{2}\ \Delta\theta_{i,max,g} \tag{6.28}$$

Une contrainte s'ajoute pour la commande partagée. Il s'agit du temps nécessaire au système pour générer une nouvelle configuration t_{syst}. On a donc

$$t_{min} = \max\ (\max_i\ (t_{i,min}), t_{syst}) \tag{6.29}$$

On croit que ce mode de commande peut significativement simplifier et rendre plus sécuritaire le travail de l'opérateur.

6.12 Singularités

Les singularités n'ont pas été explicitement étudiées, car elles n'ont pas causé de problèmes. On explique ce phénomène par l'utilisation que l'on fait de la Jacobienne $\mathbf{J}$. On rencontre des singularités lorsque $\mathbf{J}$ est singulière. Plus $\mathbf{J}$ contient des éléments et plus les chances de rencontrer une singularité augmentent. Puisque le long de la trajectoire on utilise $\mathbf{t} = \dot{\mathbf{p}}$ et $\mathbf{J} = \mathbf{J}_A$, et que l'orientation est spécifiée par $\mathbf{z}$, $\mathbf{J}$ a peu de chances d'être singulière.

Dans un contexte plus général, on suggère d'utiliser la cinématique inverse en vitesse pour pouvoir traverser les singularités lors du suivi d'une trajectoire Cartésienne générale. Si une singularité est rencontrée, il s'agit de détecter les lignes de l'équation 4.9 responsables de cette singularité et de les éliminer pour quelques pas. Les demandes formulées par ces lignes peuvent être transférées dans le vecteur d'optimisation $\mathbf{z}$ qui cherchera à suivre la trajectoire demandée du mieux qu'il peut, sans toutefois causer de singularité. L'organe terminal du manipulateur déviera légèrement de la trajectoire demandée, mais il parviendra à traverser la singularité.

Chapitre 7

Simulation

7.1 Caractéristiques du simulateur

La stratégie est implantée dans un simulateur 3D, en langage C [31], dans l'environnement Openwindows, sur des stations de travail SUN SPARC. Pour faciliter le développement, une version 2D a aussi été implantée. Le simulateur permet d'évaluer la qualité des trajectoires produites en affichant les mouvements du robot dans son environnement et les valeurs articulaires de la trajectoire. Pour la visualisation, il utilise la librairie graphique VOGL, qui est une banque de fonctions d'affichage écrites en langage C.

Le simulateur permet de modéliser un manipulateur sériel ayant jusqu'à neuf degrés de liberté avec liaisons prismatiques ou rotoïdes. Neuf degrés de liberté représentent le nombre de touches disponibles pour l'interface clavier et est suffisant pour ce projet. On pourrait ajouter d'autres degrés de liberté, l'algorithme étant général. En plus du mode de commande automatique, un mode de commande partagée a été implanté. Un mode de commande articulaire est aussi disponible pour faciliter les essais et comparer l'utilité des modes de commande de haut niveau par rapport à la commande articulaire.

Les trajectoires articulaires calculées sont affichées en position avec et sans lissage, en variation de position (vitesse) et en variation de variation de position (accélération).

On retrouve aussi l'écart entre les trajectoires avec et sans lissage. Lors de la visualisation de la trajectoire, il est possible d'arrêter, de reculer, ou de faire avancer pas à pas la séquence de la trajectoire. La scène peut être observée de n'importe quel point de vue. La représentation du robot en fil-de-fer favorise une bonne vitesse d'animation et permet de voir à travers les objets. Les obstacles sont représentés cube par cube, pour garder la généralité de la représentation, par une simple diagonale, pour les mêmes raisons que la représentation fil-de-fer. L'objectif, que l'opérateur peut déplacer à sa guise, est représenté par un cube s'il est en position et par une *pince* orientable, s'il est décrit en position et orientation. Il est possible d'afficher à l'écran les résultats des étapes de calcul, avec la quantité de détails désirée. On peut aussi mettre en fichier les champs de potentiel discrets.

Pour la commande du simulateur, deux versions sont disponibles. La première version du simulateur est principalement commandée à partir du clavier. Celle-ci a servi lors de l'implantation des algorithmes, puisque sa programmation très simple facilitait les nombreuses modifications. Bien que peu conviviale pour un nouvel usager, elle s'avère rapide et simple d'utilisation pour un usager initié. Pour satisfaire des besoins d'utilisation plus généraux et en simplifier l'apprentissage, une deuxième version a été créée. Celle-ci est commandée à l'aide de la souris et d'une interface de commande utilisant la librairie XView. La fenêtre principale de cette version plus conviviale du simulateur est illustrée à la figure 7.1.

Les dimensions de la scène, les caractéristiques du manipulateur, la description des obstacles et les paramètres du système sont conservés en fichiers que le système va lire au besoin. Ceci permet une grande flexibilité, vu la facilité à modifier ou créer un fichier, puis à le faire lire par le système, pendant que le simulateur fonctionne. Ces caractéristiques en font un outil performant pour les besoins de développement du système.

Étant donné la grande flexibilité du système, on suggère que celui-ci pourrait servir lors du design d'un manipulateur. Il permet en effet d'évaluer facilement et rapidement les performances de manipulateurs sériels quelconques dans un environnement avec ou sans obstacles, pour les trois modes disponibles.

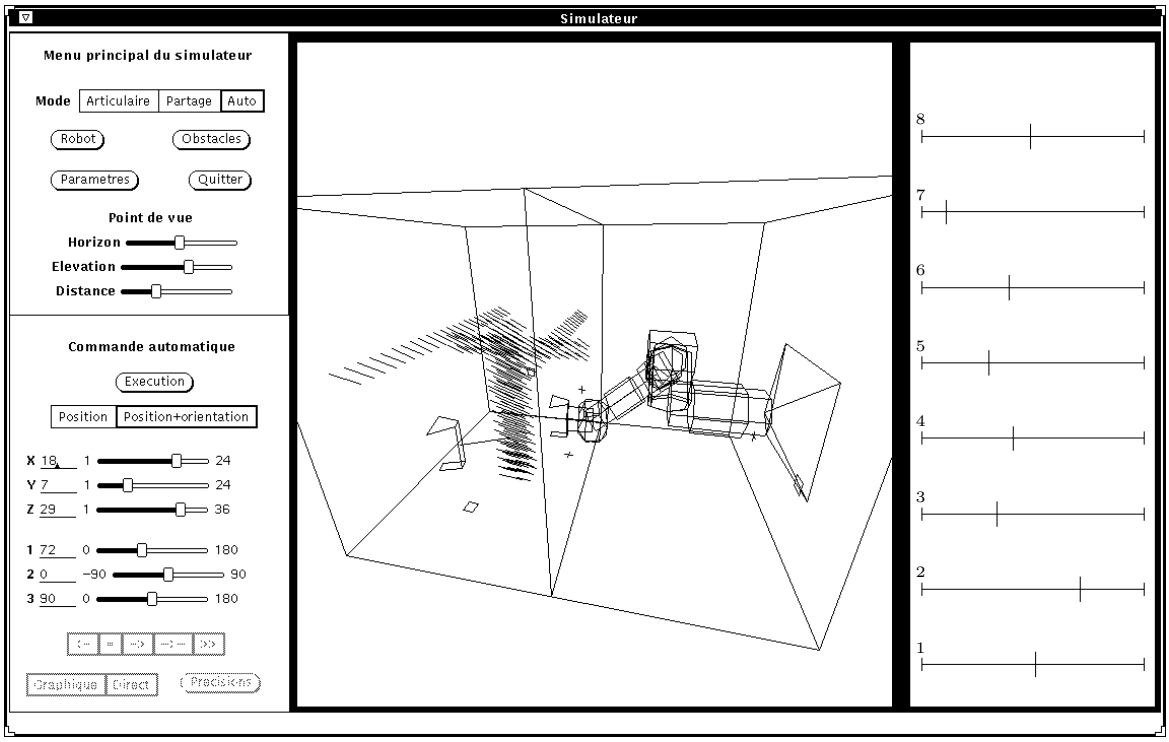


Figure 7.1: Fenêtre principale du simulateur de planification de trajectoire.

7.2 Résultats

Le simulateur 3D a principalement servi à évaluer les performances de la commande automatique pour le bras Sarcos avec glissière à sa base, dans des environnements semblables à celui présenté à la figure 1.1, puisqu'il s'agissait de l'application principale. Dans un cadre plus général, des essais ont aussi été réalisés avec d'autres manipulateurs, tels que le Puma et une version modifiée du bras Sarcos, dans divers environnements.

Le simulateur 2D a surtout servi à faire une vérification initiale des nouveaux algorithmes pendant le développement du système, car les algorithmes sont plus rapidement implantés et leur comportement est plus facile à observer. Divers robots et environnements ont été utilisés.

Pour présenter les caractéristiques du système, quelques exemples de trajectoires sont illustrées et commentées. Celles-ci sont réalisées sur une station SUN SPARC LX. Les temps de calcul des exemples de trajectoires sont présentés au tableau 7.1.

L'exemple 2D-A (figure 7.2) présente une trajectoire d'un manipulateur 2D à 6 degrés de liberté, avec 3 liaisons rotoïdes et 3 liaisons prismatiques. La trajectoire comporte 50 pas. Cet exemple illustre l'utilité des articulations prismatiques pour ce type de planification de trajectoire. La généralité du système facilite l'essai de ce type de manipulateur original.

L'exemple 2D-B (figure 7.2) présente une trajectoire d'un manipulateur 2D à 4 degrés de liberté, avec 4 liaisons rotoïdes. Cet exemple illustre une situation où l'heuristique de contour des fils a son utilité. En effet, l'organe terminal aurait tendance à passer du côté éloigné des obstacles et le manipulateur resterait coincé, sans l'heuristique. La trajectoire comporte 80 pas. On obtient un temps de calcul égal à l'exemple 2D-A, même s'il y a plus de pas, car un manipulateur avec moins de degrés de liberté demande moins de temps de calcul. Ceci s'explique par la dimension des matrices utilisées pour la cinématique inverse.

La courbe sur la dernière image de chaque séquence de la figure 7.2 représente la trajectoire de l'organe terminal. Pour les deux trajectoires, remarquer la tendance de l'organe terminal à se tenir loin des obstacles, sans toutefois garder la plus grande

distance possible de ceux-ci. Les trajectoires sont assez douces, malgré un faible lissage (paramètres (0,1,2,1,0)), car la cinématique inverse est utilisée.

Les exemples en 3D présentent des trajectoires effectuées par le bras Sarcos avec une liaison prismatique à sa base. La grille d'occupation utilisée est de dimensions $32 \times 32 \times 48$ noeuds. Il est à noter que le temps de calcul des champs de potentiel est proportionnel au nombre de noeuds de la grille d'occupation.

L'exemple 3D-A (figure 7.3) présente une trajectoire simple, comportant environ 40 pas. L'objectif demande une position et une orientation de l'organe terminal. L'organe terminal n'a pas à contourner d'obstacles, mais le reste du manipulateur doit bouger de manière à éviter l'obstacle. On note que le temps de calcul est très court, ce qui démontre la capacité de la stratégie à résoudre des trajectoires simples rapidement. Les trajectoires articulaires, représentées par la série de courbes, sont relativement simples.

L'exemple 3D-B (figure 7.4) présente une trajectoire où les heuristiques de contour des fils sont utiles. Les obstacles de l'environnement modélisent un ensemble de support pour les fils électriques, semblable à celui illustré à la figure 1.1. On voit que l'objectif, modélisé par un cube, est inaccessible si le manipulateur cherche à passer au dessus du fil, mais l'heuristique du contour des fils le force naturellement à contourner ce fil. On remarque qu'un piège associé au contour des fils est rencontré. Au départ, l'organe terminal traverse un mur imaginaire situé derrière le fil le plus près du manipulateur. Un mouvement de retrait (de l'image 1 à l'image 2) est donc effectué en premier lieu. Puisque l'objectif ne prescrit pas d'orientation c'est le mouvement naturel du manipulateur et la tendance qu'a celui-ci à s'éloigner des obstacles qui dictent l'orientation de l'organe terminal. Les trajectoires articulaires sont plus sollicitées qu'au premier exemple, étant donné la trajectoire plus complexe demandée. On note qu'il n'y a toutefois pas de mouvements brusques, grâce à un lissage adéquat.

La figure 7.5 présente l'affichage détaillé d'une trajectoire articulaire. Cet affichage est disponible pour toutes les articulations. Ici, c'est la trajectoire de l'articulation 2 de l'exemple 3D-B qui est illustrée. Ceci permet d'obtenir l'information voulue sur la sollicitation de chaque articulation.

L'exemple 3D-C (figure 7.6) présente une trajectoire où les heuristiques de dégagement

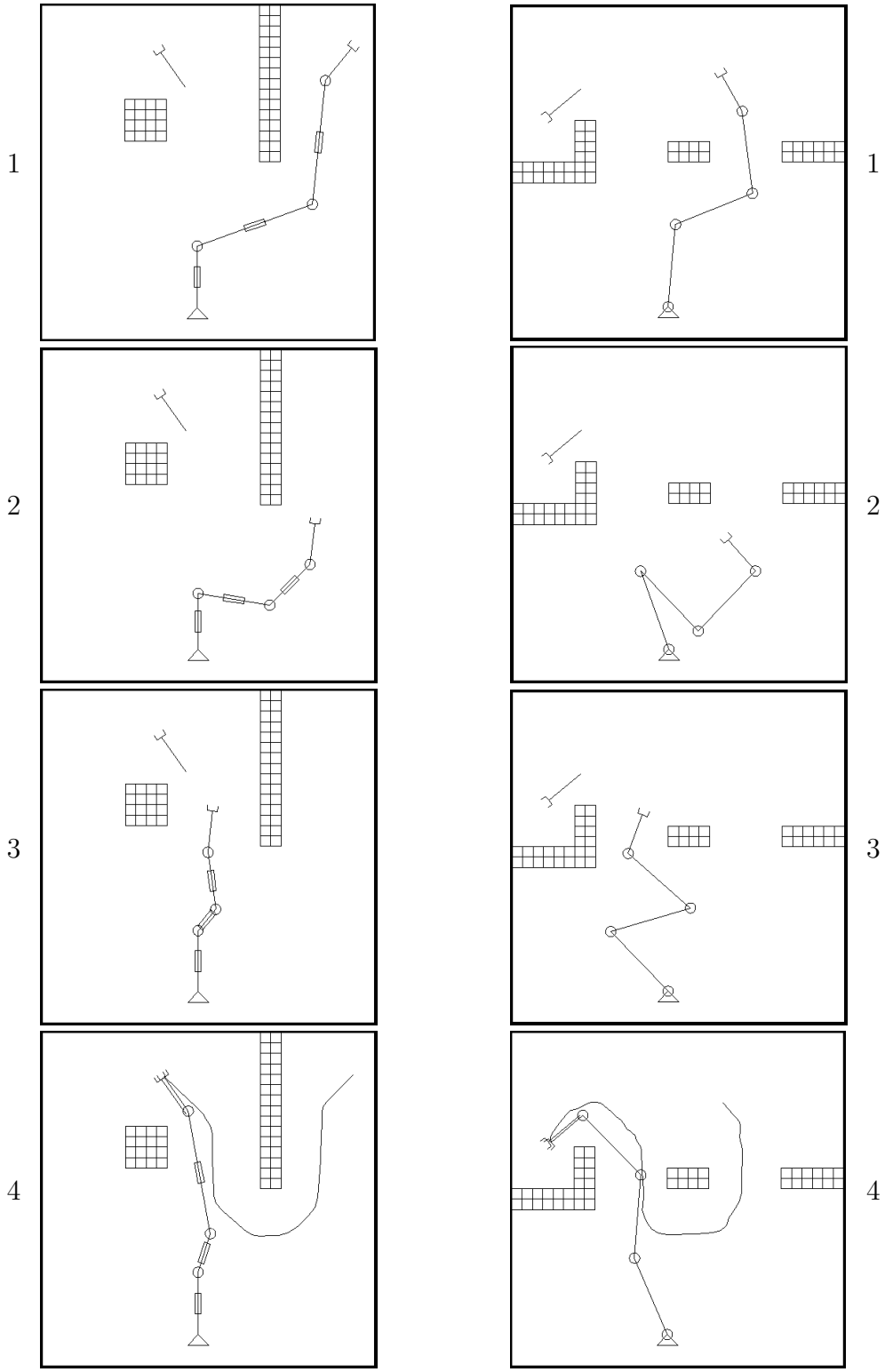


Figure 7.2: Exemples de trajectoire 2D-A et 2D-B.

doivent servir. Le champ de potentiel attractif prescrit une trajectoire de l'organe terminal qui est trop à la gauche du manipulateur. Ceci force le coude et l'avant-bras à passer vis-à-vis la poutre, ce qui provoque un blocage et déclenche l'heuristique de dégagement. D'abord, les derniers pas de la trajectoire sont éliminés car ils conduisent vers une impasse. Pour se dégager, le manipulateur va tendre à reculer son coude, ce qui tire l'organe terminal vers la droite, puis diriger l'organe terminal et le coude dans la direction générale de la trajectoire (vers le bas, dans le cas présent) vers une configuration ayant un dégagement adéquat. Finalement, le manipulateur se dirige vers son objectif, une fois l'heuristique de dégagement utilisée. Cette trajectoire comporte environ 100 pas.

La figure 7.7 illustre l'utilité du lissage en comparant les trajectoires sans et avec lissage de l'articulation 3 de l'exemple 3D-C. Cette articulation aurait été particulièrement sollicitée, surtout lors de l'heuristique de dégagement, sans le lissage. L'élimination des oscillations est à remarquer.

Les essais ont démontré que le manipulateur employé influence les chances de trouver une trajectoire. Par exemple, l'articulation prismatique à la base du bras Sarcos permet de contourner les obstacles les plus rapprochés tout en permettant d'atteindre des objectifs éloignés. Ceci est d'autant plus vrai lorsque le débattement de l'articulation prismatique est grand. Bien sûr, le degré de redondance du manipulateur influence aussi les possibilités de réussite.

En général, le système permet de solutionner une large gamme de trajectoires, dont des cas complexes, tel qu'illustré par les exemples. Toutefois, des cas d'échec peuvent survenir, puisque la stratégie utilisée est locale. Les temps de calcul moyens obtenus, soit environ 3 secondes pour une trajectoire de difficulté moyenne et 2.5 secondes pour le traitement d'un nouvel environnement, sont assez courts pour qu'une implantation en temps réel soit utile. La pertinence de la commande de haut niveau devient évidente lorsqu'on veut essayer de réaliser les trajectoires présentées en contrôlant chacune des 8 articulations séparément, ce qui est très difficile.

La commande partagée est aussi très utile. Elle l'est particulièrement lorsque les trajectoires sont trop complexes pour la commande automatique. La fréquence de calcul et d'envoi de nouvelles configurations, qui est d'environ 30 Hz, et le respect des

limites mécaniques du manipulateur pour la pire situation possible limitent cependant la rapidité de ce mode de commande. On croit que la commande partagée pourrait aussi servir lors des manipulations, une fois la phase d'approche effectuée.

Il reste à connaître le temps nécessaire pour réaliser les trajectoires, compte tenu des capacités mécaniques du manipulateur. On n'a pu obtenir de résultats, n'ayant pas les données nécessaires. Cependant, les trajectoires obtenues après lissage permettent de croire à des performances raisonnables.

Exemple	2D-A	2D-B	3D-A	3D-B	3D-C
Calcul des champs de l'environnement	< 0.1	< 0.1	2.4	2.6	2.6
Calcul des champs de la trajectoire	< 0.1	< 0.1	0.5	1.0	0.8
Génération de la trajectoire	0.5	0.5	1.0	1.3	3.2
Total de la trajectoire	0.5	0.5	1.5	2.3	4.0
Total global	0.5	0.5	3.9	4.9	6.6

Table 7.1: Temps de calcul des exemples (en secondes).

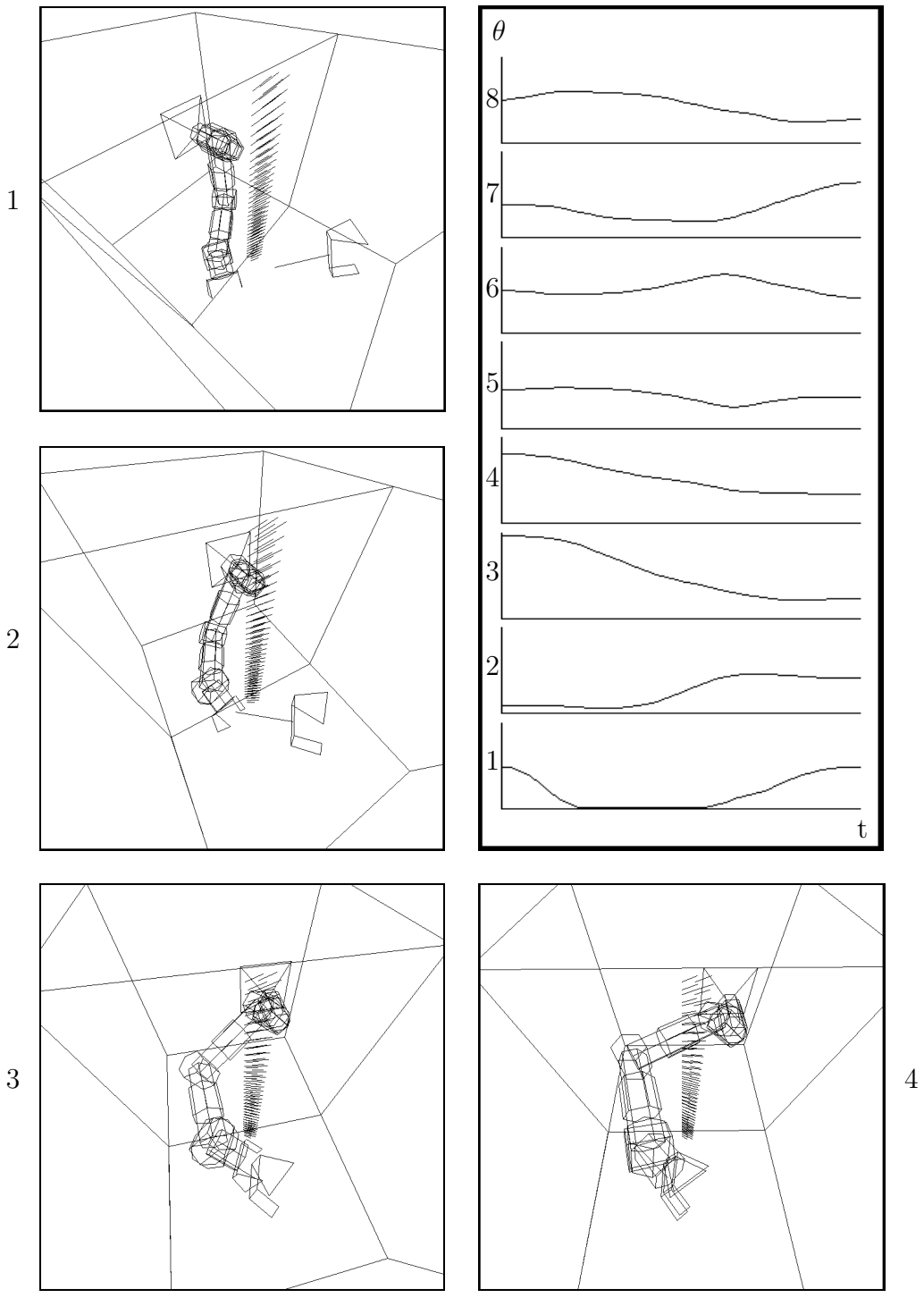


Figure 7.3: Exemple 3D-A.

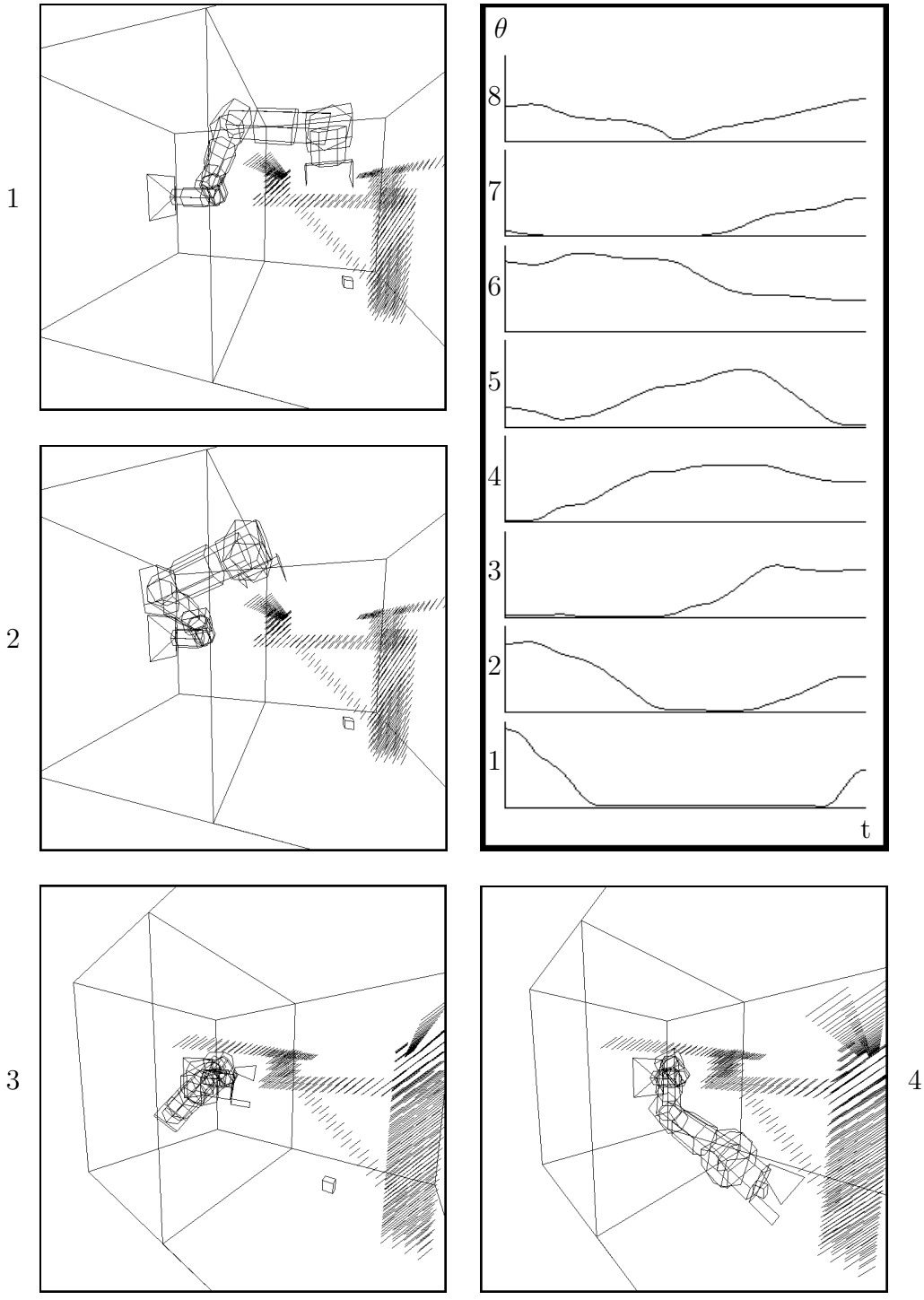


Figure 7.4: Exemple 3D-B.

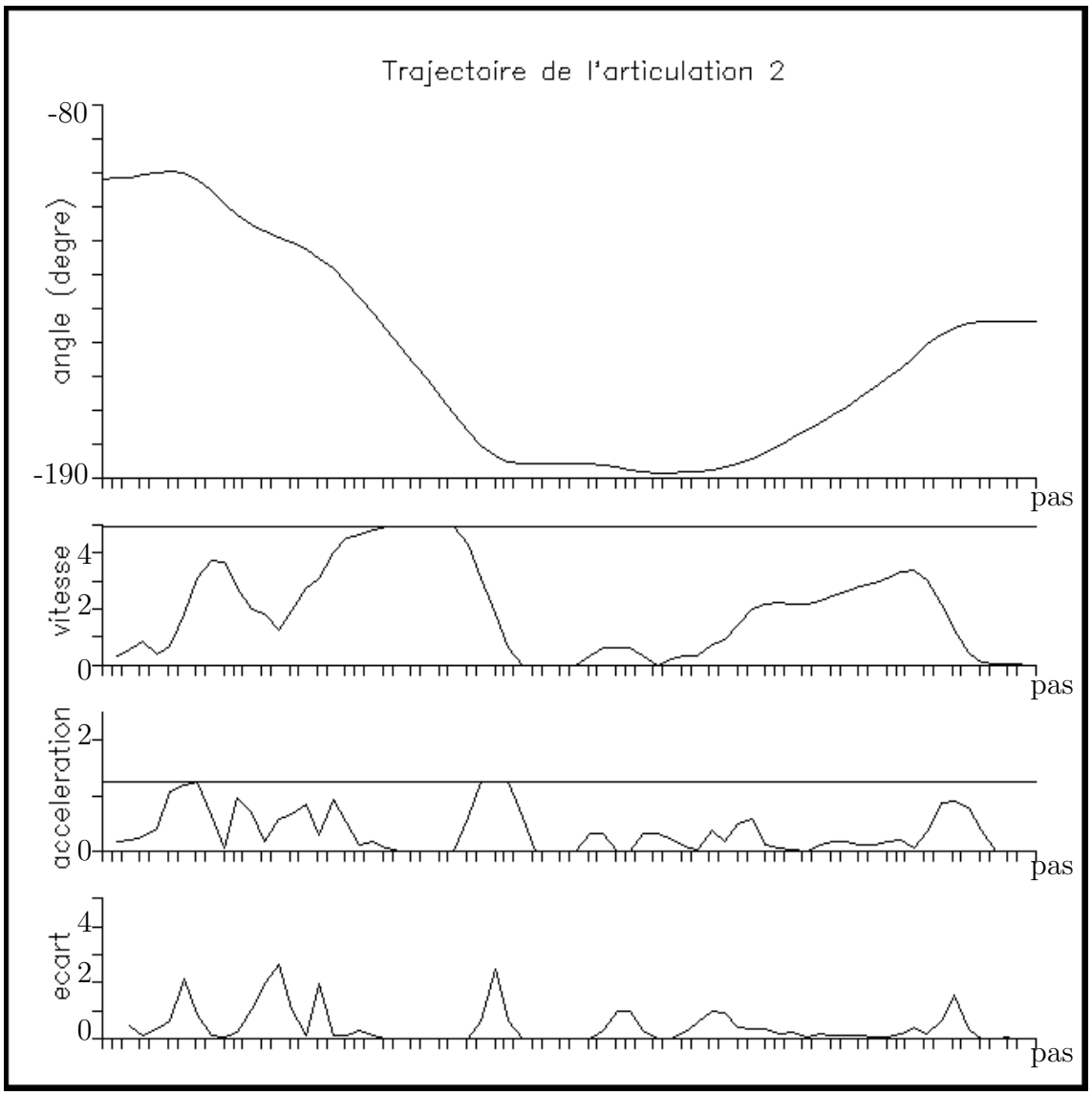


Figure 7.5: Description détaillée de la trajectoire articulaire de l'articulation 2 de l'exemple 3D-B.

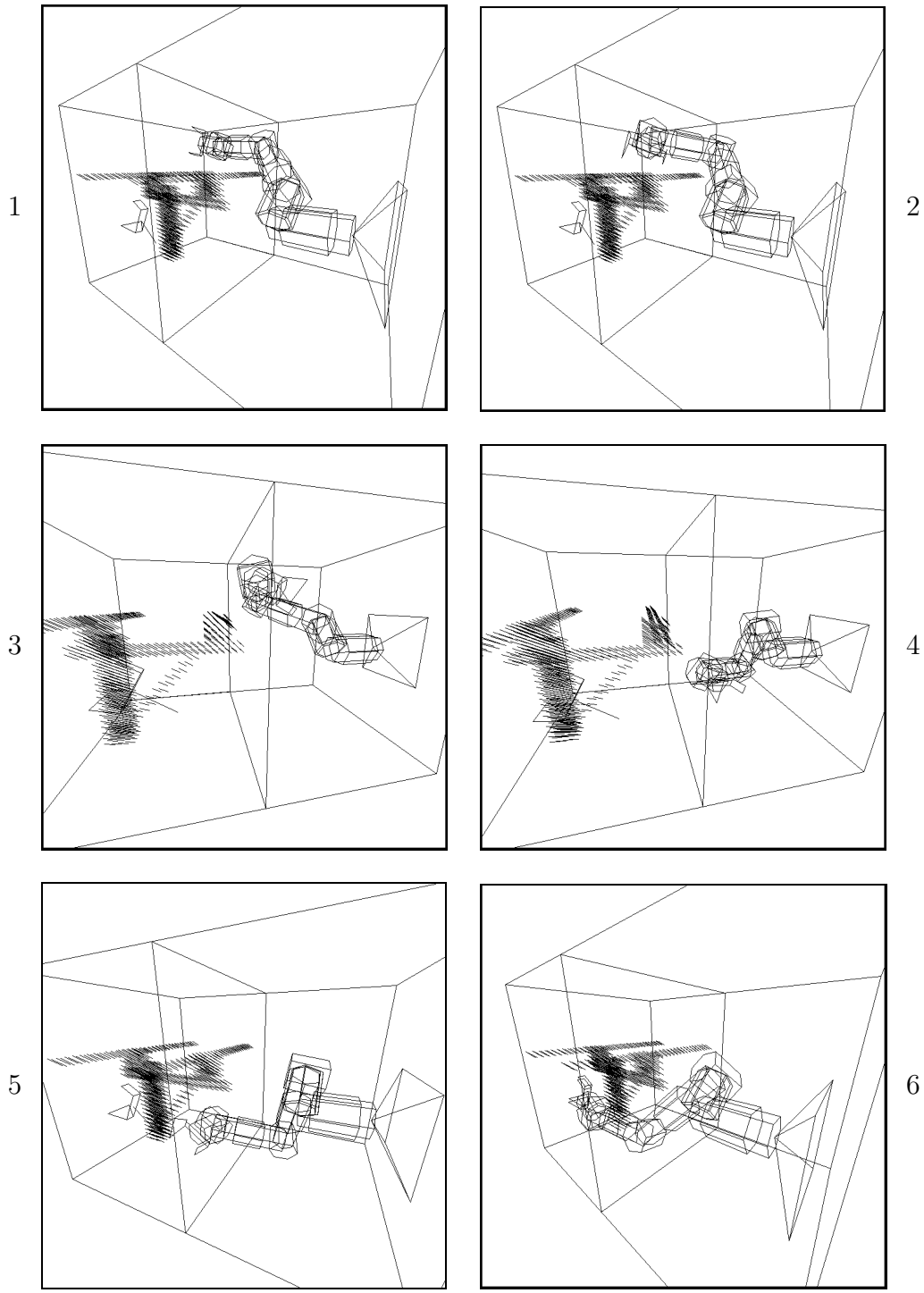


Figure 7.6: Exemple 3D-C.

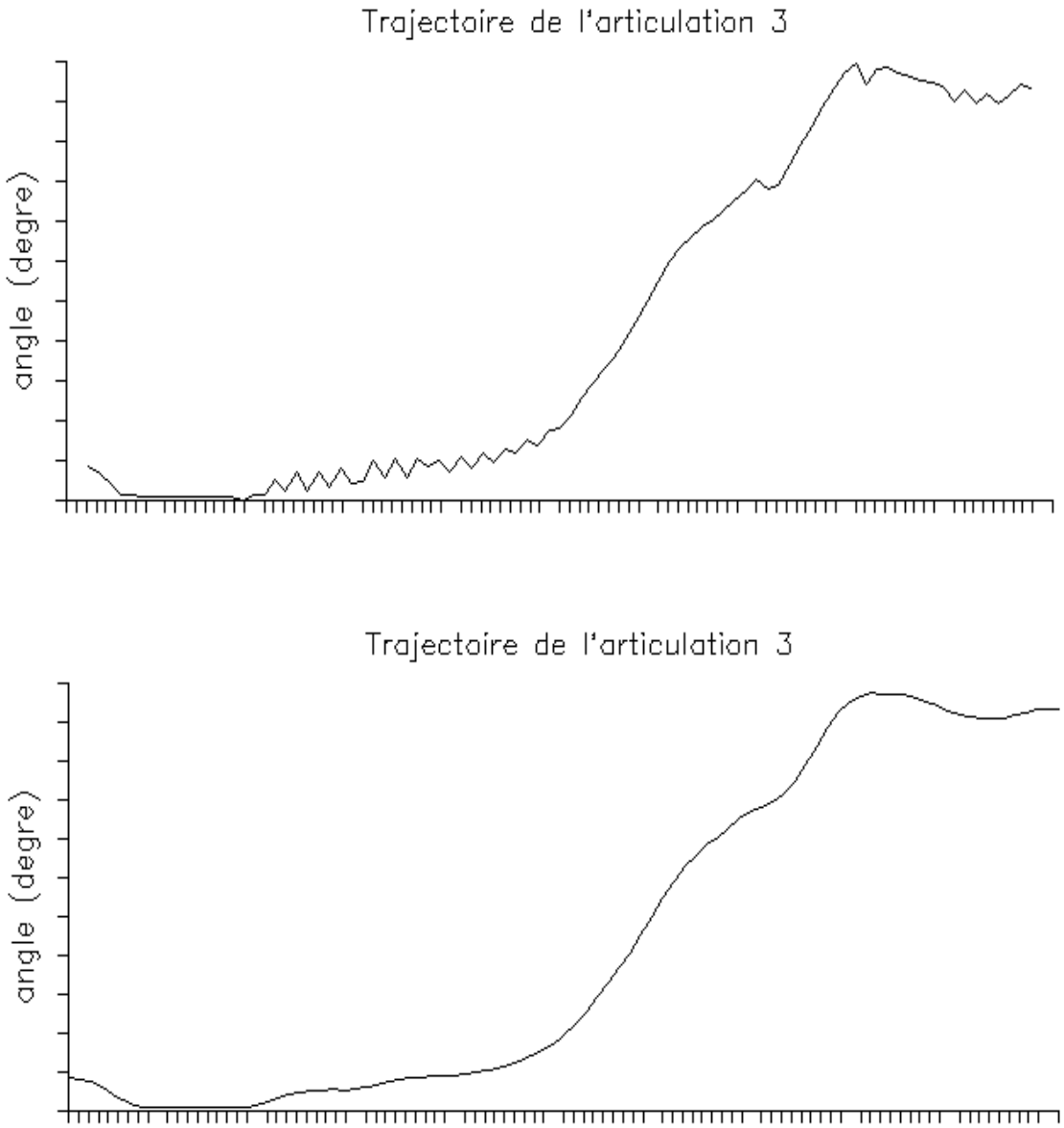


Figure 7.7: Illustration de l'utilité du lissage.

Chapitre 8

Conclusion

Ce mémoire présente une stratégie de planification de trajectoire d'un manipulateur redondant dans un environnement encombré. La stratégie découle d'un compromis entre le taux de réussite et le temps de calcul, et s'avère adaptée aux cas qui demandent une réponse rapidement avec présence d'un opérateur. En plus d'une combinaison des concepts de champ de potentiel discret et de cinématique inverse en vitesse, tirés de la revue de la littérature, elle utilise des algorithmes originaux.

Le système implanté représente un outil particulièrement utile pour les cas de télémanipulation. En permettant de commander à un plus haut niveau, il simplifie la tâche de l'opérateur et rend les opérations d'approche plus sécuritaires. Les essais effectués ont clairement démontré l'utilité de la commande automatique, surtout en présence de nombreux degrés de liberté en 3D. La commande partagée est aussi très utile et sécuritaire. Elle complète bien la commande automatique lorsque celle-ci échoue. L'effort fourni par l'opérateur est moindre par rapport au mode de commande articulaire, ce qui lui permet de se concentrer sur le travail à effectuer.

Le taux de réussite du système est élevé pour des environnements peu et moyennement encombrés, tels que ceux présentés au chapitre 7, mais plus faible pour des cas très encombrés. Ces derniers ne sont résolvables que par des stratégies globales, trop longues en temps de calcul pour la présente application. Pour les cas d'échec, l'opérateur doit intervenir. Dans le contexte d'entretien d'un réseau de distribution, le système se révèle particulièrement bien adapté aux cas où il y a des fils

grâce aux heuristiques implantées à cet effet. Les heuristiques de dégagement implantées permettent de solutionner certaines situations complexes. La spécification souple de l'orientation le long de la trajectoire et l'introduction du concept d'espace préparatoire favorisent aussi l'augmentation du taux de réussite. La qualité de la redondance du manipulateur utilisé et le niveau de discrétisation utilisé influencent ce taux de réussite.

Les trajectoires sont généralement courtes et sans détours inutiles. Ceci est surtout vrai pour les trajectoires générées à l'aide de la cinématique inverse. Les heuristiques de dégagement génèrent des trajectoires de qualité inférieure, bien qu'acceptables. Dans tous les cas, la trajectoire ne causera pas de collision, si l'environnement demeure inchangé après la prise d'information. Le lissage simple effectué sur les valeurs articulaires permet une utilisation plus efficace du manipulateur. La configuration finale de la trajectoire est automatiquement générée par la cinématique inverse, ce qui simplifie la tâche de l'opérateur.

L'algorithme est suffisamment rapide pour une implantation utile. En moyenne, des trajectoires, incluant le calcul de tous les champs potentiels nécessaires et le calcul de trajectoire, sont générées en environ 6 secondes, sur une station SUN SPARC LX. Les heuristiques permettant de réduire le traitement des champs de potentiel, la simplicité du modèle du robot et les vérifications préliminaires ont particulièrement réduit les temps de calcul. Le niveau de discrétisation employé et les performances de l'ordinateur influencent significativement le temps de calcul.

En plus des contributions mentionnées précédemment, ce mémoire introduit un traitement original des limites articulaires en position et l'utilisation d'une interpolation pour contourner le problème de la résolution grossière des champs de potentiel.

D'un point de vue général, plusieurs aspects restent à étudier :

- L'étude plus approfondie du conditionnement et des singularités dans le contexte de la planification de trajectoire.
- Détailler la modélisation des manipulateurs, pour étudier des cas plus généraux.
- Étudier la planification de trajectoire en tenant compte de la dynamique du manipulateur.

- Utiliser les algorithmes pour faire de la planification de trajectoire dans un environnement dynamique.

D'un point de vue pratique, ce travail n'est qu'une étape pour arriver à une implantation sur le terrain. Plusieurs aspects restent donc à travailler :

- Expérimenter le système pour un robot réel.
- Intégrer la planification de trajectoire dans le système général. Principalement, établir l'interface avec le système de vision.
- Continuer l'étude des heuristiques de dégagement dans un cadre plus pratique.
- Diminuer le temps de calcul des routines. L'utilisation d'un ordinateur parallèle est entre autres suggéré.

Bibliographie

[1] Anderson, K. and Angeles, J., "Kinematic Inversion of Robotic Manipulators in the Presence of Redundancies", The International Journal of Robotics Research, Vol. 8, No. 6, pp. 80–96, December 1989.

[2] Auclair, F.,"Modélisation de l'encombrement de l'espace de travail d'un robot à partir de données 3D obtenues d'un capteur", Mémoire de maîtrise, Département de génie électrique, Université Laval, Septembre 1993, 69 p.

[3] Barraquand, J. and Latombe, J.-C.,"A Monte-Carlo Algorithm for Path Planning with Many Degrees of Freedom", Proceedings of the IEEE International Conference on Robotics and Automation, Cincinnati, pp. 1712–1717, May 1990.

[4] Barraquand, J. and Latombe, J.-C.,"Robot Motion Planning: A Distributed Representation Approach", The International Journal of Robotics Research, Vol. 10, No. 6, pp. 628–649, December 1991.

[5] Chang, P. H.,"A Closed-Form Solution for Inverse Kinematics of Robot Manipulators with Redundancy", IEEE Journal of Robotics and Automation, Vol. RA-3, No. 5, pp. 393–403, October 1987.

[6] Cheng, F.-T., Chen, T.-H. and York-Yih, S.,"Efficient Algorithm for Resolving Manipulator Redundancy – The Compact QP Method", Proceedings of the IEEE International Conference on Robotics and Automation, Nice, France, pp.508–513, May 1992.

[7] Chiacchio, P., Chiaverini, S., Sciavicco, L. and Siciliano, B.,"Closed-Loop Inverse Kinematics Schemes for Constrained Redundant Manipulators with Task

Space Augmentation and Task Priority Strategy", The International Journal of Robotics Research, Vol. 10, No. 4, pp. 410–425, August 1991.

[8] Dupont, P. E. and Derby S.,"A Two Phase Path Planning Algorithm for Robots with Six or More Joints", Proceedings of the ASME Mechanisms Conference, Kissimmee, pp. 415–428, September 1988.

[9] Dupont, P. E. and Derby S.,"A Simple Heuristic Path Planner for Redundant Robots", Proceedings of the ASME Mechanisms Conference, Kissimmee, pp. 429–440, September 1988.

[10] Dupont, P. E. and Derby S.,"Planning Collision Free Paths for Redundant Robots Using a Selective Search of Configuration Space", Proceedings of the Design Engineering Conference, Columbus, 1986.

[11] Faverjon, B.,"Hierarchical Object Models for Efficient Anti-collision Algorithms", Proceedings of the IEEE International Conference on Robotics and Automation, Scottsdale, pp. 333–340, May 1989.

[12] Faverjon, B. and Tournassoud, P.,"A Local Based Approach for Path Planning of Manipulators With a High Number of Degrees of Freedom", Proceedings of the IEEE International Conference on Robotics and Automation, Raleigh, pp. 1152–1159, March-April 1987.

[13] Faverjon, B. et Tournassoud, P.,"Techniques de la robotique, Tome 2, (Chap. 7)", (Boissonnat, Faverjon et Merlet éditeurs), Hermès, 1988, 496 p.

[14] Gosselin, C. M.,"Mécanique des manipulateurs", Notes de cours, Département de génie mécanique, Université Laval, 1992, 162 p.

[15] Gupta, K. K.,"Fast Collision Avoidance for Manipulator Arms: A Sequential Search Strategy", IEEE Transactions on Robotics and Automation, Vol. 6, No. 5, pp. 522–532, October 1990.

[16] Hayward, V.,"Fast Collision Detection Scheme by Recursive Decomposition of a Manipulator Workspace", Proceedings of the IEEE International Conference on Robotics and Automation, San Francisco, pp. 1044–1049, April 1986.

[17] Herman, M.,"Fast, Three-Dimensional, Collision-Free Motion Planning", Proceedings of the IEEE International Conference on Robotics and Automation, San Francisco, pp. 1056–1063, April 1986.

[18] Kant, K. and Zucker, S. W.,"Toward Efficient Trajectory Planning: The Path-Velocity Decomposition", The International Journal of Robotics Research, Vol. 5, No. 3, pp. 72–89, Fall 1986.

[19] Khatib, O.,"Real-Time Obstacle Avoidance for Manipulators and Mobile Robots", The International Journal of Robotics Research, Vol. 5, No. 1, pp. 90–98, Spring 1986.

[20] Kim, J.-O. and Khosla, P.,"Real-Time Obstacle Avoidance Using Harmonic Potential Functions", Proceedings of the IEEE International Conference on Robotics and Automation, Sacremento, pp. 790–796, April 1991.

[21] Kirćanski, M. and Vukobratović, M.,"Contribution to Control of Redundant Robotic Manipulators in an Environment with Obstacles", The International Journal of Robotics Research, Vol. 5, No. 4, pp. 112–119, Winter 1986.

[22] Kondo, K.,"Motion Planning with Six Degrees of Freedom by Multistrategic Bidirectional Heuristic Free-Space Enumeration", IEEE Transactions on Robotics and Automation, Vol. 7, No. 3, pp. 267–277, June 1991.

[23] Latombe, J.-C.,"Robot Motion Planning", Kluwer Academic Publishers, 1991, 651 p.

[24] Lessard, J., Lavallée, J., Girard, P. and McGee, J.-Y.,"Research and Developments in Telerobotics at Hydro-Québec for the Introduction of a Live Line Telemanipulation System", Rapport Technique, IREQ, Hydro-Québec, Varennes, Québec, 1992.

[25] Lozano-Pérez, T.,"A Simple Motion-Planning Algorithm for General Robot Manipulators", IEEE Journal of Robotics and Automation, Vol. RA-3,No. 3, pp. 224–238, June 1987.

[26] Maciejewski, A. A. and Klein, C. A.,"Obstacle Avoidance for Kinematically Redundant Manipulators in Dynamically Varying Environnements", The International Journal of Robotics Research, Vol. 4, No. 3, pp. 109–117, Fall 1985.

[27] Maciejewski, A. A. and Klein, C. A., "The Singular Value Decomposition: Computation and Applications to Robotics", The International Journal of Robotics Research, Vol. 8, No. 6, pp. 63–79, December 1989.

[28] Nakamura, Y., Hanafusa, H. and Yoshikawa, T.,"Task-Priority Based Redundancy Control of Robot Manipulators", The International Journal of Robotics Research, Vol. 6, No. 2, pp. 3–15, Summer 1987.

[29] Noborio, H., Fukuda, S. and Arimoto, S.,"Fast Interference Check Method Using Octree Representation", Advanced Robotics, Vol. 3, No. 3, pp. 193–212, 1989.

[30] Press, W. H., Flannery, B. P., Teukolsky, S. A. and Vetterling, W. T.,"Numerical Recipes in C, Second Edition", Cambridge University Press, Cambridge, 1992, 994 p.

[31] Schildt, H.,"C: The Complete Reference, Second Edition", Osborne Mc Graw-Hill, Berkeley, 1990, 823 p.

[32] Sciavicco, L. and Siciliano, B.,"A Solution Algorithm to the Inverse Kinematic Problem for Redundant Manipulators", IEEE Journal of Robotics and Automation, Vol. 4, No. 4, pp. 403–410, August 1988.

[33] Warren, C. W.,"Global Path Planning Using Artificial Potential Fields", Proceedings of the IEEE International Conference on Robotics and Automation, Scottsdale, pp. 316–321, May 1989.

[34] Warren C. W., Danos, J. C. and Mooring B. W.,"An Approach To Manipulator Path Planning", The International Journal of Robotics Research, Vol. 8, No. 5, pp. 87–95, October 1989.

[35] Wikman, T. S. and Newman W. S.,"A Fast, On-Line Collision Avoidance Method for a Kinematically Redundant Manipulator Based on Reflex Control", Proceedings of the IEEE International Conference on Robotics and Automation, Nice, France, pp. 261–266, May 1992.

ANNEXE A

Description du bras Sarcos

Le bras Sarcos est un manipulateur à 7 degrés de liberté ayant la morphologie du bras humain. La version utilisée, le *General Robotic Large Arm*, est actionnée hydrauliquement. Les couples aux actionneurs et les débattements sont présentés au tableau A.1.

Dans cet projet il y a une articulation prismatique à la base du Sarcos. C'est donc avec ses 8 degrés de liberté qu'il est présenté. On note que l'articulation prismatique (axe Z1) a un débattement avant-arrière d'environ 21 cm. Les paramètres H-D du bras Sarcos utilisés pour ce projet sont présentés dans le tableau A.2 et illustrés à la figure A.1.

articulation (axe Z)	couple (lb-po)	débattement (degrés)
flexion-extension de l'épaule (Z2)	24 000	110
abduction-adduction de l'épaule (Z3)	24 000	110
rotation humérale (Z4)	6 000	180
coude (Z5)	6 000	110
rotation du poignet (Z6)	2 000	180
flexion-extension du poignet (Z7)	2 000	180
abduction-adduction du poignet (Z8)	2 000	90

Table A.1: Couples aux actionneurs et débattements du bras Sarcos.

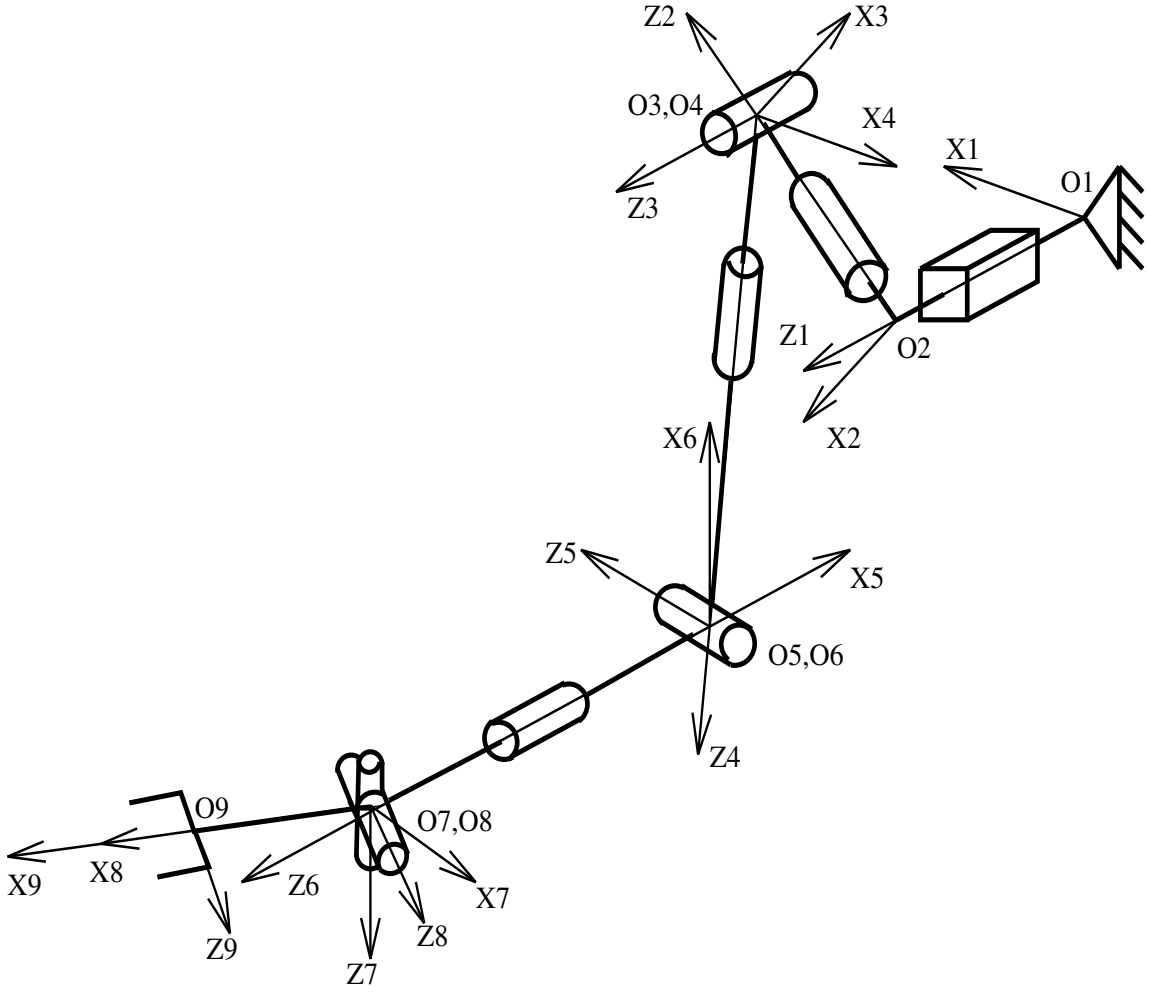


Figure A.1: Illustration des paramètres H-D du bras Sarcos avec glissière.

i	a_i (cm)	b_i (cm)	α_i (degrés)	θ_i (degrés)
1	0.0	b_1	90.0	45.0
2	0.0	50.0	90.0	θ_2
3	0.0	0.0	90.0	θ_3
4	0.0	87.9	90.0	θ_4
5	0.0	0.0	90.0	θ_5
6	0.0	76.2	90.0	θ_6
7	0.0	0.0	90.0	θ_7
8	25.0	0.0	0.0	θ_8

Table A.2: Paramètres H-D du bras Sarcos avec glissière.

ANNEXE B

Heuristiques de dégagement pour le Sarcos

Les heuristiques de dégagement pertinents peuvent différer selon le manipulateur utilisé. Ainsi, ces heuristiques ne sont pas généraux. Cependant, la stratégie d'implantation est quant à elle générale et sert de base à l'implantation d'heuristiques pour n'importe quel manipulateur. La présente annexe décrit les résultats d'une étude plus approfondie pour le bras Sarcos avec glissière. Elle sert d'exemple pratique pour comprendre le concept et faciliter l'implantation d'heuristiques pour d'autres manipulateurs. Ce qui suit donne donc la description détaillée des opérations effectuées par les heuristiques de dégagement pour le bras Sarcos avec glissière.

Tout d'abord, l'algorithme considère qu'il y a problème si la boucle de correction de la cinématique inverse ne trouve pas de solution ou si les configurations fournies restent à peu près les mêmes pendant plusieurs pas. Dans ces situations, l'heuristique de dégagement est appelée.

Par la suite, le cas problème est identifié à l'aide des indices disponibles. Les cas illustrés ici représentent la plupart des problèmes rencontrés après plusieurs mises en situation.

Cas A : Si l'objectif en position est atteint, mais que celui en orientation ne l'est

pas, c'est que ce dernier a pu être mal abordé. En effet, il se peut que l'orientation de l'objectif soit près de l'orientation de l'organe terminal, mais qu'il ne soit pas atteignable à cause des limites articulaires. Dans ce cas, il est souvent possible de trouver une solution en abordant autrement l'orientation.

Cas B : Si l'objectif se trouve dans l'espace préparatoire, il est possible que le coude soit mal placé (par analogie avec le bras humain, le coude reste près du corps alors que la main cherche à rejoindre le milieu du dos).

Cas C : Si l'organe terminal et le coude sont dans l'espace préparatoire, la trajectoire de l'organe terminal demande alors un repli impossible pour le manipulateur.

Cas D : Si aucune des conditions précédentes n'est rencontrée, on considère avoir un cas où le coude est mal placé pour contourner un obstacle. Soit qu'il est trop avancé, ou que la répulsion des obstacles l'a poussé dans la mauvaise direction, conduisant ainsi les articulations critiques à leur limite articulaire.

Ensuite, le mouvement latéral de l'organe terminal avant le blocage est déterminé. On l'appellera *mouvement général*. Pour ce faire, on compare les positions de l'organe terminal à la mi-trajectoire et au blocage. On en déduit aussi un mouvement latéral perpendiculaire au mouvement latéral de l'organe terminal, que l'on nommera *mouvement perpendiculaire*. On note qu'il y a deux sens possibles à ce mouvement et qu'ils seront utilisés à tour de rôle, étant donné l'impossibilité de déterminer le plus adéquat d'entre eux.

Aussi, l'algorithme met en mémoire les articulations aux limites lors du blocage de la trajectoire.

Finalement, une configuration de départ est déterminée. L'algorithme repart à la moitié de la séquence générée, évitant ainsi un détour inutile dans une impasse et favorisant le dégagement d'une situation difficile.

Pour chaque cas problème, une séquence d'actions est réalisée. Les actions sont exécutées par chaque articulation séparément. Tout comme pour le calcul du vecteur d'optimisation $\mathbf{z}$, on effectue une dérivée numérique sur la fonction à optimiser, correspondant à l'action voulue, en faisant varier artificiellement la valeur de chaque

articulation.

Cas A :

L'action consiste à repousser de leur limite les articulations bloquées, et ce, pendant 20 pas. Pour éviter d'éloigner l'organe terminal de son objectif en position, une autre action maintient l'organe terminal aux environs de la position demandée. Ces actions combinées procurent une rotation de l'organe terminal qui permet généralement d'attaquer l'orientation avec plus de succès, si une solution existe.

Cas B :

L'action pousse le coude dans le sens inverse de la direction du mouvement général. De plus, elle pousse l'organe terminal dans la direction du mouvement général avec les articulations qui n'influencent pas la position du coude (articulations 4 à 8), pendant 15 pas.

Cas C :

La première action pousse l'organe terminal dans la direction du mouvement perpendiculaire, tout en le maintenant dans la zone préparatoire, pendant 15 pas. Si l'heuristique de dégagement est rappelée une autre fois, l'action pousse dans l'autre sens.

La deuxième action pousse le coude dans la direction du mouvement général, tout en maintenant le coude reculé et l'organe terminal dans l'espace préparatoire, pendant 10 pas.

La troisième action consiste à pousser l'organe terminal et le coude dans la direction du mouvement général, tout en maintenant leur position en profondeur. Ceci est effectué jusqu'à ce que la position de l'organe terminal, dans la direction du mouvement général, atteigne la position de l'objectif, ou pendant 25 pas.

La dernière action pousse l'organe terminal vers l'avant tout en dirigeant l'organe terminal et le coude dans la direction du mouvement général. Ceci est effectué jusquà ce que l'organe terminal entre dans la zone de travail ou pendant 10 pas.

Cas D :

La première action recule l'organe terminal et le coude dans la zone préparatoire jusqu'à ce que l'organe terminal atteigne la zone préparatoire ou pendant 15 pas.

La deuxième action consiste à ramener les articulations aux limites vers le centre de leur zone atteignable pendant 5 pas. Ceci permet de faciliter les actions suivantes.

La troisième action pousse le coude dans la direction du mouvement général, tout en maintenant le coude reculé et l'organe terminal dans l'espace préparatoire, pendant 10 pas.

La quatrième action consiste à pousser l'organe terminal et le coude dans la direction du mouvement général, tout en maintenant leur position en profondeur. Ceci est effectué jusqu'à ce que la position de l'organe terminal, dans la direction du mouvement général, atteigne la position de l'objectif, ou pendant 25 pas.

La dernière action pousse l'organe terminal vers l'avant tout en dirigeant l'organe terminal et le coude dans la direction du mouvement général. Ceci est effectué jusquà ce que l'organe terminal entre dans la zone de travail ou pendant 10 pas.

Pendant la génération de la trajectoire de dégagement, il y a vérification de collision et des limites articulaires. Pour la vérification de collision et les limites articulaires en vitesse, l'algorithme fonctionne comme l'algorithme standard. Pour les limites articulaires en position, l'algorithme bloque simplement l'articulation fautive, puisque l'on ne travaille plus avec la cinématique inverse.

L'heuristique de dégagement terminée, un nouveau champ de potentiel attractif est construit en considérant la nouvelle situation.

L'heuristique peut être utilisée le nombre de fois voulu. Généralement, on peut considérer que l'heuristique ne parviendra pas à dégager le robot après trois essais. De plus le temps nécessaire devient trop long pour être utile à l'opérateur.