UNIVERSITÉ DE SHERBROOKE
Faculté de génie
Département de génie électrique et de génie informatique

MODÉLISATION DE LA SÉCURITE DES TÂCHES COOPÉRATIVES HUMAIN-ROBOT

Mémoire de maîtrise
Spécialité : génie électrique

Sonny TARBOURIECH

Jury : Wael Suleiman (directeur)
François Michaud (rapporteur)
Claudia Esteves (examinateur externe)

Sherbrooke (Québec) Canada Janvier 2016

RÉSUMÉ

L'interaction physique humain-robot est un domaine d'étude qui s'est vu porter beaucoup d'intérêt ces dernières années. Une optique de coopération entre les deux entités entrevoit le potentiel d'associer les forces de l'humain (comme son intelligence et son adaptabilité) à celle du robot (comme sa puissance et sa précision).

Toutefois, la mise en service des applications développées reste une opération délicate tant les problèmes liés à la sécurité demeurent importants. Les robots constituent généralement de lourdes machines capables de déplacements très rapides qui peuvent blesser gravement un individu situé à proximité.

Ce projet de recherche aborde le problème de sécurité en amont avec le développement d'une stratégie dite "pré-collision". Celle-ci se caractérise par la conception d'un système de planification de mouvements visant à optimiser la sécurité de l'individu lors de tâches d'interaction humain-robot dans un contexte industriel. Pour ce faire, un algorithme basé sur l'échantillonnage a été employé et adapté aux contraintes de l'application visée. Dans un premier temps, l'intégration d'une méthode exacte de détection de collision certifie que le chemin trouvé ne présente, a priori, aucun contact indésirable. Ensuite, l'évaluation de paramètres pertinents introduit notre notion de sécurité et définit un ensemble d'objectifs à optimiser. Ces critères prennent en compte la proximité par rapport aux obstacles, l'état de conscience des êtres humains inclus dans l'espace de travail ainsi que le potentiel de réaction du robot en cas d'évènement imprévu. Un système inédit de combinaison d'objectifs guide la recherche et mène à l'obtention du chemin jugé comme étant le plus sûr, pour une connaissance donnée de l'environnement. Le processus de contrôle se base sur une acquisition minimale de données environnementales (dispositif de surveillance visuelle) dans le but de nécessiter une installation matérielle qui se veut la plus simple possible. Le fonctionnement du système a été validé sur le robot industriel Baxter.

Mots-clés : Tâches d'interactions physiques humain-robot, Analyse des dangers, Planification de mouvements sécuritaire

REMERCIEMENTS

Je tiens à remercier tout particulièrement mon directeur de recherche, le professeur Wael Suleiman, qui m'a accordé toute sa confiance et qui a su se rendre disponible tout au long du projet. Je lui suis très reconnaissant vis-à-vis des efforts qu'il a pu faire en ma faveur et de l'aide précieuse qu'il a su m'apporter.

Je remercie également toute ma famille qui a toujours été là pour moi, tant sur le point personnel que financier.

Enfin, je voudrais remercier les nombreux acteurs de l'Université de Sherbrooke qui ont rendu possible la réalisation de cette maîtrise.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

CHAPITRE 1

INTRODUCTION

Les récentes recherches ont commencé à viser une nouvelle manière d'exploiter la robotique afin de rendre le robot capable d'interagir avec l'humain. Au niveau industriel, une évolution serait de faire coopérer la machine et l'humain dans le but de fabriquer des produits spécifiques à la demande du client de manière rapide et précise [Giuliani *et al.*, 2010]. Ceci peut être rendu possible en effectuant des tâches d'interactions physiques qui allient la puissance des systèmes robotiques à l'intelligence de l'humain.

Les robots actuels sont capables de réaliser très efficacement les tâches assignées. En revanche, ceux-ci sont programmés pour effectuer une série d'actions de manière déterminée. Une fois la machine lancée, l'humain n'intervient plus dans le processus de calculs du robot. Cependant, les nouvelles applications citées précédemment impliquent que le robot ne soit plus isolé dans une cellule de travail mais passe dans un environnement non structuré dans lequel l'humain pourrait prendre place. De ce fait, il est indispensable de conférer au robot la capacité d'acquérir et d'interpréter les informations qui l'entourent. La raison principale qui engendre cette nécessité est qu'il faut assurer la sécurité dans un espace où coexistent la machine et l'humain.

Ce projet de recherche aborde le problème de sécurité en amont avec le développement d'une stratégie dite "pré-collision". Celle-ci se caractérise par la conception d'un système de planification de mouvements visant à optimiser la sécurité de l'individu lors de tâches d'interaction humain-robot dans un contexte industriel. Lors de la spécification d'une tâche à effectuer par le robot (une configuration articulaire ou une position de l'effecteur à atteindre), l'algorithme mis au point permet de générer le chemin jugé le plus sécuritaire. L'algorithme de planification de mouvements basé sur l'échantillonnage consiste à explorer l'environnement par le biais d'états (ensemble de valeurs articulaires) prélevés aléatoirement pour ensuite être connectés afin de relier une configuration initiale à une configuration finale. Cet algorithme intègre les critères de sécurités et organise les connections entre les états de manière à obtenir les chemins les plus sécuritaires en tenant compte de la connaissance actuelle de l'environnement.

Dans un premier temps, une analyse des facteurs de danger a permis de définir un ensemble de critères relatifs à la sécurité humaine et matérielle. Ceux-ci tiennent compte de

paramètres liés au robot (proximité des configurations singulières et distances par rapport aux limites articulaires) et à l'environnement (distance par rapport aux obstacles et aux humains, direction du regard humain). Ainsi, il devient possible d'établir une comparaison qualitative entre les différents chemins potentiels.

La méthode standard de détection de collision entre deux états est basé sur l'interpolation. Elle ne permet pas de s'assurer avec certitude que le robot ne va pas percuter un obstacle le long du chemin, en particulier si celui-ci est fin. De ce fait, nous utilisons une méthode exacte de détection de collision qui tient compte des distances entre deux corps rigides et de leur déplacement. Nous avons adapté la méthode aux besoins de cette étude puis l'avons étendu afin qu'elle permette de calculer la distance minimale entre les deux corps le long du chemin (ce qui correspond à un de nos critères).

L'optimisation simultanée de plusieurs objectifs se fait généralement par le biais d'une fonction qui effectue une combinaison linéaire des différents critères. En revanche, en procédant de la sorte, l'influence de chaque objectif n'est pas bien maîtrisé et tous les objectifs ne peuvent pas être combinés ensemble. De ce fait, nous avons développé une nouvelle méthode de combinaison qui pallie ces lacunes.

Le temps de calcul nécessaire à la détermination d'un chemin sécuritaire est une contrainte primordiale pour l'obtention d'un système viable. De ce fait, nous avons mis au point une série de fonctionnalités qui rendent le processus de planification de mouvements plus performants. Celles-ci se caractérisent notamment par l'utilisation d'heuristiques qui permette à l'algorithme de supprimer les opérations inutiles en termes d'optimisation de solution.

Un système au fonctionnement autonome a été développé dans le cadre de cette étude. Il se base sur l'utilisation d'un capteur 3D Asus Xtion PRO pour représenter les obstacles de l'environnement. Son positionnement est guidé par la considération des données de détection de mouvements issues des capteurs ultrasons intégrés sur le robot.

Au chapitre 2, une vue d'ensemble sur les travaux réalisés jusqu'à présent en matière de sécurité des tâches d'interaction humain-robot est présentée. Différentes méthodes d'évaluation du danger sont évoquées. Plusieurs approches destinées à contrer la présence de danger lors d'une telle interaction sont présentées. Les principales techniques d'intégration des mesures de sécurité au sein de systèmes de contrôle font l'objet de la dernière section. Ensuite, un article de journal soumis à "IEEE Transactions on Robotics" est présenté au chapitre 3. Il rassemble le travail effectué dans le cadre de ce projet de recherche. Enfin, le chapitre 4 donne un bilan de la place du projet au sein de la société et de l'intérêt qu'il présente.

CHAPITRE 2

SÉCURITÉ DES INTERACTIONS HUMAIN-ROBOT

Un principe fondamental de la robotique établit que la machine ne doit en aucun cas devenir une menace pour l'humain [Asimov, 2004]. Plus l'humain se rapproche du robot, plus le risque de blessure s'accroît. Pour cette raison, la mise en place de standards de sécurité au niveau de la robotique industrielle a débuté en 1999 [ANSI/RIA, 1999]. À ce moment là, le seul moyen fiable trouvé pour assurer la sécurité en tout temps était de prohiber le partage de l'espace de travail entre les deux entités mises en jeu. Cela est rendu possible par la mise en place de zones de danger, le plus souvent délimitées par une cage rigide dans laquelle l'humain ne peut pas pénétrer lorsque le système est en fonctionnement. Ce type de dispositif constitue un frein à l'amélioration des procédés de production en milieu industriel et entraîne une limite au niveau du champ d'action apporté par les robots, particulièrement au sein des petites et moyennes entreprises où l'espace et les contraintes financières rendent l'installation de robots moins abordable.

Parallèlement à cela, les progrès en intelligence artificielle et les avancées technologiques en termes de contrôle, conception et réalisation de systèmes mécaniques pour la robotique ont permis le développement d'applications nouvelles centrées sur l'humain. Conscient du potentiel mis en jeu, les domaines visés s'élargissent. Parmi eux, les principaux concernés sont les milieux médicaux, domestiques ou encore l'assistance personnelle à domicile pour les personnes âgées ou handicapées [Garcia *et al.*, 2007].

Les besoins de la société ont alors poussé les recherches à se diriger vers l'étude des interactions entre l'humain et le robot afin de casser les barrières qui séparent notre environnement de celui de la machine.

Les études des interactions humain-robot, appelées communément HRI (*Human-Robot Interaction*), ont fait l'objet de nombreux travaux de recherche ces dernières années. À la base, une HRI signifie que le robot et l'humain cohabitent au sein d'un espace de travail partagé. D'un point de vue sécuritaire, cela nécessite une conscience mutuelle vis-à-vis de la présence et des actions effectuées par l'autre contributeur. Il se produit alors une interaction dans le sens où les mouvements de l'un vont affecter les décisions de l'autre. De ce fait, l'humain devient une entrée du système de contrôle du robot. La notion

d'interaction physique humain-robot (pHRI) reprend la définition précédente en ajoutant le fait qu'il se produit, de façon brève ou durable, un contact physique entre l'humain et la machine. Dans le cadre d'interaction physique, il peut s'agir d'un contact direct ou indirect (par le biais d'un objet physiquement partagé). Un scénario généralement étudié est le transfert d'objet du bras du robot vers le bras de l'humain (*object hand-over task*) [Huber *et al.*, 2008]. Quelle que soit la nature de l'interaction à réaliser, elle est rendue possible uniquement dans un contexte où la sécurité est assurée en tout temps.

2.1 Estimation de la sécurité de l'interaction

La sécurité étant un concept vague, il est important de pouvoir caractériser quantitativement le danger d'une situation afin de déterminer la stratégie à adopter en conséquence.

2.1.1 Évaluation des collisions potentielles

Une première méthode destinée à déterminer un indice de sécurité consiste à évaluer la collision potentielle en fonction de paramètres physiques du système. Pour cela, différents indices initialement empruntés des tests de collision de l'industrie automobile sont utilisés. Parmi eux, un des plus communs est le *Head Injury Criterion* (HIC) [Versace, 1971] qui évalue la gravité des lésions encéphaliques lors d'un choc avec le robot. Sa valeur numérique est définie par :

$$HIC = T \left[\frac{1}{T} \int_0^T a(\tau) d\tau \right]^{2.5} \tag{2.1}$$

où a est la valeur d'accélération de la tête et T est la durée totale de l'impact. Lorsque la valeur de HIC est supérieure à 1000, alors la blessure est jugée sévère.

Différents indices de sévérité des blessures céphaliques ont été analysés en simulation dans le cadre d'impacts non contraints avec le bras manipulateur LWRIII [Haddadin *et al.*, 2007]. Cela a permis d'étudier la pertinence de ces critères pour des situations de HRI. La conclusion principale de cette étude est que pour des impacts non tranchants, le robot LWRIII ne cause pas de blessures jugées sévères malgré une vitesse d'impact relativement élevée (1 m/s). Ces résultats ont été confirmés lors d'expériences d'impacts rigides réalisés sur des personnes volontaires.

Ces travaux ont permis de mettre en avant-plan l'inadaptation du standard utilisé en automobile pour des applications de HRI. En effet, l'échelle de vitesse n'étant pas du

tout comparable, les indices indiquent un risque de danger faible dans la plupart des configurations.

Les recherches ont alors été poursuivies dans le but d'établir un nouveau standard plus adapté aux situations de HRI. De nouvelles sources de blessures ont été évaluées [Haddadin *et al.*, 2009]. Il a été montré qu'une cause majeure de blessure grave est issue d'une situation dans laquelle l'humain se retrouve coincé par le robot. De plus, lorsque le robot se trouve proche d'une configuration singulière, les forces qu'il exerce sur l'environnement s'accroissent considérablement et peuvent devenir extrêmement dangereuse pour un humain qui se retrouverait bloqué physiquement par le robot. L'influence de la masse et de la vitesse du robot sur l'impact de la collision a été examinée afin d'élargir le champ des robots considérés. L'étude a permis une détermination plus systématique des situations à risque où de nombreux paramètres supplémentaires ont également été pris en compte : type de contact (rigide/tranchant), proximité d'une configuration singulière, liberté de mouvements de l'humain, type d'outil utilisé.

2.1.2 Modélisation des dangers lors de la phase initiale

La difficulté rencontrée lors de scénarios de HRI, et en particulier pour l'évaluation du danger, réside dans le fait que des robots autonomes prévus pour s'intégrer dans des environnements dynamiques sont utilisés ici. Considérer l'ensemble des situations dangereuses peut s'avérer complexe, voire impossible. Pour faciliter cette phase d'analyse, des méthodes systématiques de modélisation du risque centrée sur l'humain ont été développées. Une approche intéressante se base sur le langage unifié de modélisation (UML) pour tout d'abord représenter l'ensemble des tâches, identifier les situations dangereuses puis estimer les risques associés [Guiochet *et al.*, 2010]. La modélisation des dangers permet d'identifier les situations dangereuses à partir d'une adaptation de la méthode collaborative HAZOP (*HAZard OPerability*). Cette dernière permet une génération systématique des situations à risque à partir de paramètres du système (ex : température, pression, ...) et de mots-clés (moins, plus, pas, ...). L'estimation du risque peut alors être effectuée selon la sévérité potentielle de la blessure et la probabilité d'occurrence.

Un inconvénient majeur de ce genre de méthodes est qu'elles sont appliquées lors des phases initiales de développement du robot. De ce fait, il est nécessaire d'effectuer à nouveau l'analyse de risques lorsque des modifications interviennent dans la tâche à exécuter ou au sein de l'environnement. En outre, estimer la probabilité d'occurrence d'une blessure nécessite une phase d'apprentissage, ce qui ajoute une contrainte supplémentaire à

l'utilisation d'un tel procédé. Enfin, la méthode d'évaluation du risque se base en grande partie sur l'estimation d'un analyste et dépend donc fortement de son expertise.

2.1.3 Évaluation dynamique du danger

L'approche qui présente le plus grand intérêt pour des applications de coopération humain-robot, et particulièrement pour ce projet de recherche, consiste à évaluer dynamiquement un indice de sécurité tout au long du processus de réalisation de la tâche. Récemment, un grand intérêt a été porté vers cette direction tant l'aspect dynamique devient inévitable avec l'intégration d'un élément imprévisible (l'humain) dans l'espace de travail du robot. Le premier indice de danger quantitatif généralisé a été proposé par Ikuta et al. [Ikuta *et al.*, 2003]. Il tient compte des potentielles forces et tensions exercées lors d'un impact involontaire entre un être humain et un robot.

Ce concept d'indice de danger a ensuite été réemployé dans les travaux de Kulic et Croft [Kulić et Croft, 2007]. Il est construit à partir de mesures qui ont un effet sur la force de l'impact potentiel durant une collision. Ce n'est pas seulement l'effecteur du robot qui doit être considéré comme source potentielle de danger, mais le robot tout entier. Pour chaque partie du robot, le point le plus proche de l'individu est considéré : il s'agit du point critique. Le critère de danger est estimé à partir de chaque point critique. Les facteurs évalués sont la distance entre la personne et le robot au niveau du point critique, leur vitesse relative et l'inertie effective au point critique. Pendant l'interaction, une surveillance de l'humain permet d'obtenir des indications qui servent à déterminer plus justement le critère de sécurité : direction des yeux, orientation du visage, expressions du visage.

Un des indices actuellement utilisé et faisant partie des plus développés utilise le concept de *Kinetostatic Danger Field* (KSDF) [Lacevic et Rocco, 2010a]. Le point fort de cet indice est qu'il peut être exprimé sous forme d'expressions algébriques en prenant comme paramètres les positions et vitesses des différents éléments du robot. De ce fait, l'intégration pour des applications temps-réel est rendu possible moyennant un coût de calcul raisonnable. En plus de l'influence de la distance, le KSDF capture deux aspects importants liés au déplacement de la source de danger : (1) la norme du vecteur vitesse de l'obstacle et (2) le sens de déplacement de l'obstacle par rapport au point de l'espace pour lequel l'indice est calculé. Un exemple est donné à la figure 2.1.

L'avantage de cette méthode est que l'évaluation du danger est appliquée sur l'intégralité du robot et non sur un nombre discret de points. De plus, la source du champ potentiel représentant l'indice de danger est issue du robot lui-même plutôt que de l'obstacle. Cela

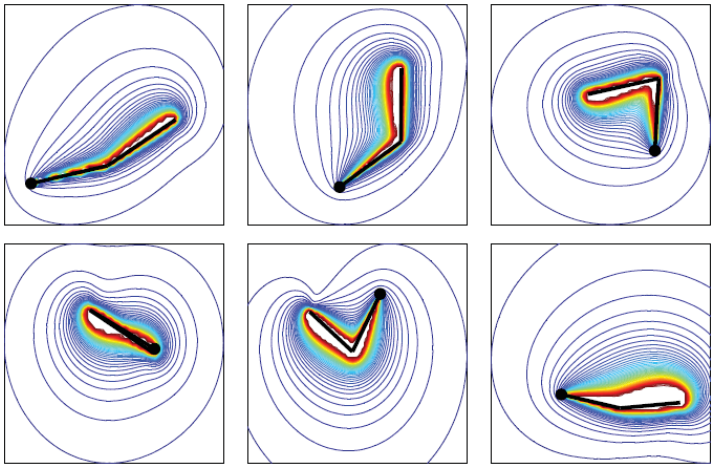


Figure 2.1 Représentation du KSDF pour un bras à deux degrés de liberté [Lacevic et Rocco, 2010a].

permet de connaître directement les déplacements admissibles et ceux considérés comme dangereux.

L'indice de danger est un critère modulable sur lequel peut venir directement s'intégrer de nouvelles propriétés afin d'améliorer la précision de l'évaluation. Ainsi, pour augmenter la sécurité des HRI, de récentes recherches visent à inclure des facteurs liés à l'humain dans la détermination du niveau de danger. Une nouvelle méthode propose une évaluation de l'indice de risque qui, en plus des facteurs physiques de collision, considère également la direction du regard de l'humain et l'orientation de son corps [Najmaei *et al.*, 2010]. Il est aussi possible de se servir de signaux physiologiques qui détiennent des informations particulières sur l'individu, notamment sur sa conscience de l'environnement et son état affectif. Ainsi, en analysant des paramètres tels que son activité électrodermale, son rythme cardiaque ou l'activité de son muscle corrugateur, il serait possible de déterminer son état d'anxiété et moduler la valeur de l'indice de danger en conséquence [Sarkar, 2002].

2.2 Sécurité intrinsèque du robot

Une première mesure effective qui tient compte de la sécurité de l'humain dans les HRI consiste à modifier les propriétés intrinsèques du robot.

L'intérêt grandissant pour le domaine de la coopération humain-robot a généré de nombreux efforts pour la détermination de standards de sécurité dans la robotique industrielle. L'ISO 10218 spécifie les limites de vitesse, la puissance maximale ainsi que la force statique qui permettent de garantir la sécurité d'une collision. Ce standard peut être considéré

comme une mesure de prévention intrinsèque, dans le sens où le robot est bridé en termes de performance afin de répondre aux exigences de sécurité. Son intégration au sein d'installation industrielle n'est pas viable car elle limite considérablement les capacités des robots actuels, notamment dans des situations où la restriction n'est pas nécessaire, ce qui atteint directement la compétitivité d'une entreprise.

Deux axes majeurs en termes de sécurité intrinsèque permettent de contourner cette restriction tout en conservant des performances intéressantes.

2.2.1 Les paramètres de conception du robot

La première méthode consiste à concevoir un manipulateur sécuritaire en établissant un lien direct entre les paramètres de conception du robot et la sévérité des collisions qu'il peut potentiellement générer [Lee *et al.*, 2013]. Les relations entre la sécurité de la collision et les longueurs et masses des liens du robot ont ainsi pu être établies. Il a été montré qu'il était plus efficace de modifier la vitesse de collision par rapport à la masse effective. De ce fait, la meilleure solution pour obtenir une sécurité de collision est de modifier la longueur des liens du robot.

Les contraintes de conception liées à ce principe de modification des paramètres du robot peuvent s'avérer très pénalisantes pour le développement de nombreuses applications industrielles car elles restreignent le volume de travail disponible pour la machine ainsi que les performances de celle-ci.

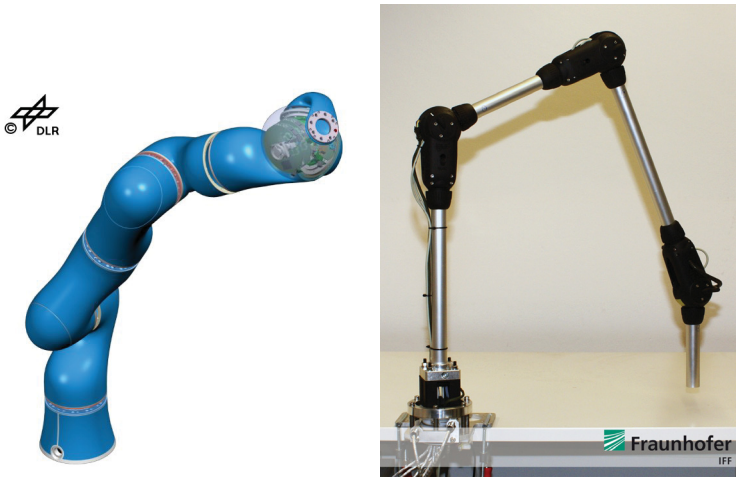
2.2.2 Les technologies de conception

La seconde catégorie de travaux de recherche dans la sécurité intrinsèque vise à intégrer les dernières technologies dans la conception mécanique des systèmes robotiques, tels que ceux montrés à la figure 2.2, afin de les rendre moins nocifs pour l'être humain. Ceci est rendu possible à travers :

- La réduction de la masse des robots grâce à l'utilisation de matériaux modernes très légers. Cependant, cela induit généralement une **perte de puissance** qui empêche l'utilisation de tels robots dans de nombreuses applications.
- L'apport de flexibilité au sein du système. Il peut s'agir de couvrir le robot d'une couche de matériaux viscoélastiques, ou de générer les mouvements des manipulateurs à l'aide de câbles. Cela permet d'éviter les lourds impacts rencontrés habituellement avec des systèmes de haute impédance. En revanche, les robots contrôlés par

câbles admettent généralement une fréquence de résonance faible ce qui **limite les performances de tels systèmes**.

- L'utilisation de nouveaux types d'actionneurs. Cela inclut les articulations basées sur la programmation passive d'impédance, le réglage mécanique d'impédance, l'actionnement contrôlé par le couple articulaire, les actionneurs sériels élastiques, les actionneurs à rigidité variable [Pervez et Ryu, 2008]. Dans cette optique, des chercheurs de l'Institut interdisciplinaire d'innovation technologique (3IT) de Sherbrooke ont mis au point un concept d'actionneur semi-actif à double différentiel rhéologique [Fauteux *et al.*, 2010]. Cet actionneur dispose d'une compacité, d'une simplicité de contrôle et d'une rapidité de réaction qui augmentent la sécurité de l'opérateur. Ces applications tendent à diminuer la rigidité des articulations et à favoriser l'amortissement des chocs lors d'une collision avec un individu.



(a) DLR LWRIII [The German Aerospace Center (DLR), 2014]

(b) ALEXA [European Clearing House for Open Robotics Development (ECHORD), 2013]

Figure 2.2 Des robots conçus pour coopérer avec l'humain.

Aborder le problème de sécurité des HRI en amont, à savoir au niveau de la conception du robot, est indiscutablement une méthode très efficace qui diminue considérablement le risque de blessure grave lors d'une collision avec l'humain. Cependant, cette approche a deux principales faiblesses :

- Elle réduit généralement les capacités offertes par le robot (performances, champs d'applications).
- Elle nécessite un investissement financier élevé. Toute la technologie intégrée dans ces nouveaux robots est coûteuse et n'est pas à la portée de tous les clients qui ont besoin de ce produit.

2.3 Stratégies de réaction post-collision

Au lieu d'agir sur les propriétés intrinsèques du robot, une autre approche réside dans la programmation de la machine. Dans cette optique, deux stratégies sont envisageables : la première vise à maîtriser au mieux une collision humain-robot afin qu'elle ne génère le moins de dégâts possible. La seconde se concentre sur l'anticipation et l'évitement d'un contact indésiré de façon à éliminer, autant que faire se peut, les risques de blessures.

Le risque nul ne pouvant jamais être atteint, des stratégies de réactions aux collisions s'avèrent indispensable. La force de l'impact étant considérée comme la source majeure de blessures lors d'une interaction imprévue avec l'humain, il est d'un grand intérêt de pouvoir en diminuer l'intensité.

Pour cela, l'algorithme doit fonctionner en deux étapes : la première tâche est de détecter l'occurrence d'une collision. Ensuite, une réaction appropriée doit être générée par le contrôleur.

2.3.1 Stratégie de détection d'une collision

Les travaux de Sami Haddadin [Haddadin, 2013] recensent les principales méthodes destinées à détecter une collision. Elles vont de la simple détection de collision basée sur l'énergie, à des méthodes plus poussées qui permettent d'obtenir les valeurs des couples avec précision.

Une méthode de détection de collisions spécifiquement destinée aux pHRI a été mise au point très récemment [Cho *et al.*, 2013]. Un processus en ligne de mise à jour, exécuté avant l'algorithme de détection, permet d'adapter précisément le modèle du robot lors

d'une variation de la charge utile transportée. Ceci est réalisé grâce à un filtrage sur la valeur des couples. Ainsi, lors d'une situation de pHRI (ex : déplacement d'un objet commun) la collision sera détectée plus efficacement. La difficulté rencontrée lors de pHRI est de distinguer une collision d'un contact intentionnel lié à la réalisation de la tâche. Pour remédier à cela, un nouveau module a été spécialement développé pour effectuer cette distinction [Cho *et al.*, 2012]. La résolution du problème se situe dans l'étude des forces exercés lors de l'interaction : un contact intentionnel causera une augmentation plus lente des forces externes que lors d'une collision inattendue. Certains facteurs humains, comme la direction du regard et l'orientation de son corps [Najmaei *et al.*, 2010], sont également susceptible de déceler l'intention humaine lors d'un contact.

2.3.2 Stratégie de réaction à une collision

Aussitôt qu'une détection est détectée, une stratégie de réaction doit être enclenchée afin de diminuer le danger potentiel auquel l'humain est soumis. Jusqu'à maintenant, différentes stratégies ont été élaborées et peuvent conduire à des comportements significativement distincts. Les stratégies mises en œuvre sont les suivants [De Luca *et al.*, 2006] :

- Stratégie 1 : Le robot est stoppé lorsque la collision est détectée.
- Stratégie 2 : Le contrôle en position exécuté pour la réalisation de la tâche laisse place à un contrôle des couples par compensation de gravité. Le robot se comporte alors de manière très docile.
- Stratégie 3 : La stratégie 2 est reprise en utilisant cette fois un retour de couple pour diminuer l'inertie du moteur et du lien du robot. On obtient ainsi un robot encore plus « léger ».
- Stratégie 4 : Le couple externe est estimé pour réaliser un contrôle en admittance. Suite à un choc, la vitesse désirée est définie selon le sens opposé au couple externe de manière à éloigner le robot de la perturbation.

Les expériences ont révélé que les stratégies de compensation de la gravité (2 et 3) étaient les mieux perçues par les utilisateurs car elles donnent l'impression de maîtriser le robot [Haddadin *et al.*, 2008].

2.4 Stratégies pré-collision

De toute évidence, le meilleur moyen de s'assurer que le robot ne puisse pas blesser un individu est de faire en sorte qu'aucune collision inattendue, qu'elle soit directe ou qu'elle mette en jeu un objet tiers, ne se produise lors d'une situation de HRI. Cet aspect d'anticipation du danger se révèle être un défi majeur dans un environnement dynamique où l'humain amène avec lui une grosse part d'imprévisibilité. Malgré la complexité du problème, des pistes de solutions ont été trouvées dans l'optique de développer un robot sécuritaire.

2.4.1 La planification de mouvements

Un besoin fondamental en robotique est de pouvoir convertir la spécification haut-niveau d'une tâche exprimée par l'humain en une description bas-niveau des déplacements à effectuer pour parvenir à réaliser la tâche [LaValle, 2006]. La planification de mouvements et la planification de trajectoires sont des termes souvent employés pour désigner les algorithmes capables de résoudre ce genre de problèmes.

Typiquement, le but d'un algorithme de planification de mouvements est de trouver un chemin sans obstacle dans l'espace des configurations (C-space) du robot qui connecte une configuration initiale q_i à une configuration finale q_f [LaValle, 2006]. L'algorithme exact de planification, basé sur une connaissance a priori de l'intégralité du C-space, admet une complexité qui augmente exponentiellement avec la dimension du C-space [Canny, 1988]. De ce fait, l'application d'une telle méthode pour des espaces de haute dimension n'est pas envisageable. Les méthodes de planification basées sur l'échantillonnage sont bien plus appropriées dans un contexte de HRI [Lindemann et LaValle, 2005].

Parmi elles, les deux méthodes les plus populaires sont *Rapidly-exploring Random Tree* (RRT)[LaValle, 1998] et *Probabilistic Roadmap* (PRM) [Kavraki *et al.*, 1996]. Bien qu'incomplètes, elles permettent la résolution de nombreux problèmes difficiles et de très grande dimension. Ces méthodes reposent sur un échantillonnage aléatoire du C-space. Un graphe est construit à partir de ces échantillons et représente un chemin sans collision qui évolue à partir de la configuration de départ jusqu'au but à atteindre. Depuis leur création, de nombreuses modifications sont venues améliorer leur performance et ont permis la résolution de problèmes spécifiques.

L'idée principale est la suivante : intégrer un indice de sécurité dans l'algorithme de planification de mouvements afin d'orienter la recherche d'un chemin vers les zones jugées sans danger pour l'être humain.

Dans cette optique, le critère de *Kinetostatic Danger Field* a été pris en compte dans plusieurs méthodes de planification de mouvements [Lacevic et Rocco, 2013]. Ainsi, quatre nouveaux algorithmes ont pu être présentés et testés. Deux d'entre eux se basent sur le paradigme RRT et ont été initialement développés dans [Lacevic *et al.*, 2011] :

- SAFE_JR-RRT : utilise un arbre unidirectionnel dont la racine est le point de départ du chemin. Dans un premier mode, l'extension de l'arbre se fait de façon aléatoire en ce qui concerne la direction (sans rechercher à atteindre le but) mais en évaluant deux critères : le futur nœud de l'arbre est celui qui est le plus proche du nœud actuel et qui minimise l'indice de danger. Dans un second mode, l'extension se fait suivant le nœud qui minimise la distance par rapport à la position à atteindre. Dans ce mode, l'information relative au critère de danger est utilisée pour exploiter la redondance du robot en attribuant la posture la plus sûre sans compromettre le déplacement prévu de l'effecteur (projection dans l'espace nul).

- SAFE_RRT-CONNECT : Développe deux RRT basés aux configurations initiale et finale et tente de les joindre. L'extension des arbres se fait en utilisant une heuristique qui tient compte de l'indice de danger.

Parmi les deux approches restantes, l'une d'elle est déterministe et ne fait donc pas partie des algorithmes basés sur l'échantillonnage aléatoire :

- BUBBLES : Pour une configuration q donnée, une bulle $B(q)$ est générée dans le C-space et correspond à toutes les configurations qui permettent de rester à une certaine distance de l'obstacle le plus proche dans l'espace de travail à partir de la configuration actuelle. L'algorithme tente de trouver un chemin entre la configuration initiale et la configuration finale en chaînant simultanément les bulles issues à la fois des configurations initiales et finales jusqu'à trouver une intersection entre ces deux chaînes. À partir d'une configuration actuelle q, la configuration suivante est choisie parmi les points situés aux extrémités de la bulle $B(q)$ selon une fonction heuristique f. Cette fonction se base sur l'algorithme A* dans lequel le critère de sécurité (indice de danger) est intégré. En outre, un critère supplémentaire entre en jeu dans la fonction f et permet de privilégier un point situé dans une zone peu explorée.

La dernière méthode, initialement développée dans [Lacevic et Rocco, 2010b], utilise le paradigme PRM :

- SAFE_PRM_PLANNER : Ici, deux graphes sont simultanément construits à partir des configurations initiales et finales en considérant des échantillons aléatoires comme points de passage. L'évolution des graphes se base également sur une version bidirectionnelle de l'algorithme A* en utilisant la fonction heuristique évoquée précédemment. Lorsque les deux graphes se rejoignent, le chemin sans collision est retourné. La fonction de vérification de collision est uniquement appelée pour le nœud sélectionné et s'assure que la ligne droite entre la configuration actuelle et ce nœud est sans collision.

Le but de l'article [Lacevic et Rocco, 2013] est de valider la performance des planificateurs orientés sécurité et de les comparer aux algorithmes de référence. Les résultats ont montré qu'en plus de permettre une amélioration de la sécurité dans les chemins trouvés, les algorithmes présentés ont également un temps d'exécution réduit. Cela est justifiable par le fait que le critère de sécurité restreint généralement l'espace de configuration exploré en s'éloignant des obstacles.

La comparaison de ces nouveaux algorithmes suggère que le BUBBLE_PLANNER est celui qui mène vers les chemins les plus sûrs. En revanche, son temps de calcul est plus élevé que pour les algorithmes basés sur l'échantillonnage. Parmi les algorithmes basés sur l'échantillonnage, ceux utilisant la méthode RRT présentent l'avantage d'être moins coûteux en temps d'exécution. En outre, pour des espaces de configurations à hautes dimensions, SAFE_PRM_PLANNER a besoin d'un grand nombre d'échantillons pour couvrir l'ensemble de l'espace. Les algorithmes RRT basés sur la sécurité bénéficient quant à eux de tous les avantages des RRT classiques comme l'efficacité et le temps d'exécution.

Il est intéressant de remarquer que l'intégration des contraintes au sein des algorithmes évoqués se fait de façon générique. De manière générale, une fonction de coût est employée pour biaiser la direction du chemin employé. Dans l'étude précédente, le critère de *Kinetostatic Danger Field* était utilisé mais le choix de l'indice peut être modifié aisément. Pour des scénarios de pHRI, des critères supplémentaires destinés au bon déroulement de l'interaction peuvent être ajoutés.

Ainsi, un récent planificateur tient compte à la fois de la sécurité de l'individu mais génère également un chemin prévisible et un point d'échange confortable pour l'individu [Sisbot et Alami, 2012]. La prévisibilité est un critère important dans les scénarios de pHRI car elle permet de comprendre l'intention du robot sans communication directe. Un mouvement

incompris par l'humain pourrait surprendre celui-ci et causer un danger. Le confort est déterminé par rapport à la posture du bras de l'humain durant le scénario et facilite la réalisation de la tâche interactive.

2.4.2 Les méthodes prédictives et l'intelligence artificielle

L'adaptation de la planification de mouvement à des fins sécuritaires a prouvé son efficacité en termes de réduction du danger. En revanche, le processus de calcul d'un chemin est quelque chose de très coûteux en temps et ne peut pas être la solution d'un contrôle dynamique de la sécurité pour une application temps réel. L'environnement étant en perpétuelle évolution lors d'une HRI, le chemin initialement perçu comme sécuritaire peut soudainement présenter un danger pour l'être humain en présence. L'estimation de la sécurité doit être constamment mise à jour pour tenir compte des changements environnementaux. Malheureusement, du fait de l'imprévisibilité de l'humain, il est fréquent que la détection d'une situation dangereuse advienne alors que la collision est rendue inévitable. Il est donc nécessaire d'anticiper l'action à venir afin de disposer du temps requis pour contourner un éventuel danger.

Les techniques d'intelligence artificielles jouent un rôle majeur dans le développement de systèmes prédictifs. Une application intéressante de l'intelligence artificielle pour un système prédictif destiné à un scénario de HRI a été développée dans [Najmaei et Kermani, 2011]. Même si cette étude focalise uniquement sur les déplacements de l'humain et non sur les gestes qu'il effectue, elle constitue une bonne base pour ce genre d'applications. Cet article introduit un nouveau système sensoriel de modélisation de l'humain. Une fois le modèle obtenu, le mouvement que l'individu s'apprête à réaliser peut être prédit. Pour cela, un réseau de neurones artificiels (ANN) est utilisé avec pour données d'entrée les échantillons d'observation des mouvements de l'humain.

Sur le même principe, la méthode décrite dans [Mainprice et Berenson, 2013] permet d'estimer les gestes de l'humain pour une tâche particulière. Pour ce faire, des classes de mouvements spécifiques à cette tâche sont établies et constituent une base de données. Lors du déroulement de l'action, les modèles appris précédemment sont utilisés pour prédire les mouvements de l'humain et simultanément sélectionner une réponse appropriée pour le robot. À partir du mouvement actuellement réalisé par l'individu, l'algorithme permet de générer une probabilité d'occupation du volume pour les instants à venir.

Une étude prédictive spécifiquement destinée aux situations de contact continu humain-robot est présentée dans [Norouzzadeh *et al.*, 2012]. Contrairement à ce qui a été vu jusqu'à

maintenant, l'intention de l'humain n'est pas prédite à partir de méthode d'apprentissages. En effet, ces dernières se basent sur des modèles probabilistes et de ce fait ne garantissent pas une sécurité absolue. L'intention humaine est alors considérée comme inconnue du système. Le problème consiste à déterminer les variables de contrôle d'un robot manipulateur à partir de son modèle linéarisé :

$$\begin{cases} x_{k+1} = Ax_k + B_1 u_k + B_2 f_{hk} \\ y_k = Cx_k + w_k \end{cases} \tag{2.2}$$

où x_k contient les positions et vitesses du manipulateur au temps k, y_k est la position mesurée de l'effecteur, f_{hk} est une inconnue désignant la force exercée par l'humain sur l'effecteur du robot. Le but est de déterminer la force u_k appliquée par le robot de telle sorte que les performances de la tâche soient assurées tout en respectant les contraintes liées à la sécurité. Le schéma de contrôle est donné à la figure 2.3.

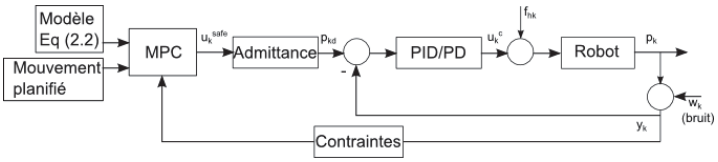


Figure 2.3 Boucle de contrôle du système prédictif assurant la sécurité des pHRI [Norouzzadeh *et al.*, 2012].

Une méthode généralisée de contrôle prédictif (GPC) est utilisée ici. Pour ce faire, on présume dans un premier temps que les valeurs de f_{hk} et u_k restent identiques à celle de l'itération $k-1$. Une prédiction de la sortie $\hat{y}_{k+j|k}$ et de la variable d'état $\hat{x}_{k+j|k}$ est alors réalisée. Ainsi, un modèle de contrôle prédictif (MPC) calcule la force idéale à appliquer u_k^{safe} . Le bloc d'admittance transforme cette force en une position désirée p_{kd}. Une boucle de contrôle en position calcule la commande à envoyer au robot à partir de l'erreur en position $p_{kd} - p_k$. La mesure récente de la position est comparée aux contraintes de sécurité puis un signal est transmis au bloc MPC qui tient compte de ce critère dans la décision de la commande optimale à appliquer.

L'approche suggérée permet au robot d'optimiser ses performances aussi longtemps que la sécurité et le confort de l'interaction ne sont pas compromis. Les expériences ont montré que l'humain se sent plus en sécurité et plus à l'aise lorsque ce module prédictif est employé.

2.5 Systèmes de contrôle

La finalité de l'étude de la sécurité des HRI est de pouvoir intégrer les stratégies énumérés jusqu'à maintenant au sein de systèmes réels.

Kulic et Croft ont proposé un système de contrôle pré-collision basé sur plusieurs niveaux d'exécution durant l'interaction humain-robot [Kulić et Croft, 2007]. Le robot assure la sécurité de l'humain en planifiant et modifiant sa trajectoire sur trois échelles de temps différentes : une planification du chemin à long terme pour éviter toute collision et pour placer le robot dans une posture idéale afin de répondre à un danger soudain ; une planification de la trajectoire à moyen terme comprenant une re-planification locale et une modification de la trajectoire dans le but d'adapter le mouvement du robot pour s'éloigner du danger ; et un contrôle réactif à court terme dans le cas d'un danger imminent permettant d'éviter la collision.

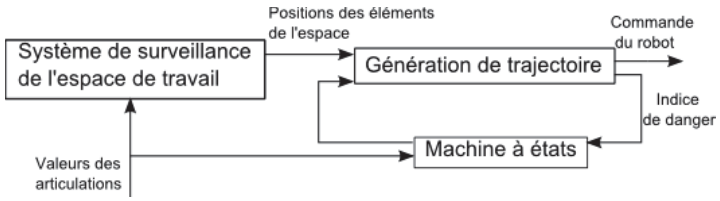


Figure 2.4 Schéma de principe de l'algorithme temps-réel de génération de trajectoire [Zanchettin et Lacevic, 2012].

La génération de trajectoire en ligne, dans le cas d'environnements non structurés, a été abordée dans [Zanchettin et Lacevic, 2012]. Le processus temps-réel fonctionne à l'aide de trois blocs, comme le montre la figure 2.4 :

- Un module de surveillance de l'espace de travail qui génère un indice de danger simple et rapide à partir des informations de positions des éléments de l'environnement.
- Un module de description de la tâche implémenté sous forme d'une machine à états. À partir d'une certaine valeur de l'indice de danger, ce module décide de suspendre la tâche.
- Un module de génération de trajectoire qui fournit les références des articulations à partir des positions/orientations cartésiennes calculés par la machine à états.

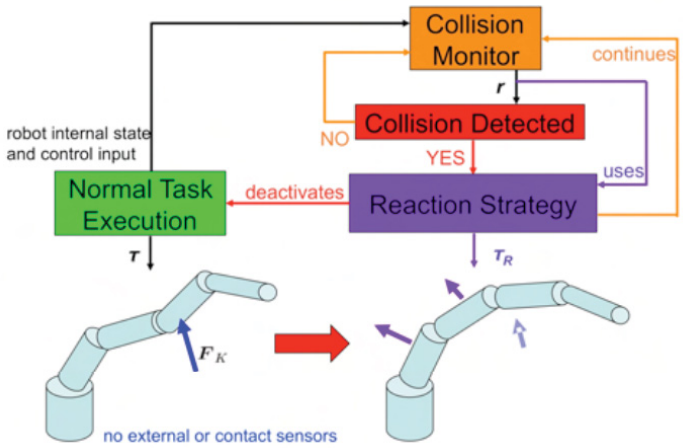


Figure 2.5 Adjonction du module de détection et réaction à une collision au sein du système de contrôle [De Luca et Flacco, 2012].

Un modèle complet de contrôle intégré pour les pHRI a été développé dans [De Luca et Flacco, 2012]. En plus du module d'évitement de collision, un système de détection puis réaction de collision est intégré. Il fait appel à une boucle d'urgence lors d'une interaction indésirable. Sa représentation est donnée à la figure 2.5.

CHAPITRE 3

PLANIFICATION MULTI-OBJECTIFS DE MOUVEMENTS POUR UNE INTERACTION HUMAIN-ROBOT SÉCURITAIRE

Auteurs :

- Sonny Tarbouriech : Étudiant à la maîtrise, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique.
- Wael Suleiman : Professeur, Laboratoire de robotique intelligente, interactive, intégrée et interdisciplinaire (IntRoLab), Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique.

État de l'acceptation : Soumis pour évaluation le 22 février 2016

Titre original : *Multi-objective Motion Planning Approach for Safe Human-Robot Interaction*

Revue : *IEEE Transactions on Robotics*

Contribution au document : Cet article aborde le problème sécuritaire lié à une collaboration humain-robot en proposant une nouvelle stratégie pré-collision. Elle repose sur l'utilisation d'une méthode de planification de mouvements basée sur l'échantillonage aléatoire. L'ajout de nombreuses fonctionnalités et de paramètres pertinents ont mené au développement d'un algorithme qui répond à la fois aux exigences temporelles et aux contraintes sécuritaires. La considération multi-objectifs qui a été mise au point permet de prendre en compte efficacement un ensemble de critères. Ainsi, le chemin qui admet le meilleur compromis (qui satisfait au mieux les différents objectifs) sera sélectionné. Combiné à une méthode en-ligne de modification locale de trajectoire, la stratégie proposée rend possible l'interaction humain-robot sécuritaire.

Résumé français : Cet article présente une nouvelle stratégie de planification de mouvements multi-objectifs de robots manipulateurs axé sur la sécurité. Intégrée dans un algorithme basé sur l'échantillonnage, notre approche réussit à améliorer la sécurité de la tâche en agissant sur deux niveaux : d'abord, une méthode de contrôle dynamique

de collision assure que le chemin obtenu est sans collision, indépendamment de la taille des obstacles environnants. Ensuite, une fonction multi-objectif guide l'expansion du chemin vers les configurations les plus sûres. Notre notion de sécurité consiste à éviter des situations dangereuses, par exemple être proche des obstacles, à prendre en compte la conscience humaine, en restant autant que possible dans son champ de vision, ainsi qu'à maintenir le robot loin des configurations singulières et des limites articulaires pour être en mesure de répondre rapidement à des évènements imprévus. Des simulations effectuées dans deux scénarios et validées sur le robot réel Baxter révèlent l'efficacité de la méthode proposée.

3.1 Abstract

This paper presents a new multi-objective safety-oriented path planning strategy for robotic manipulators. Integrated in a sampling-based algorithm, our approach can successfully enhance the task safety by acting on two levels : first, a dynamic collision checking method ensures that the obtained path is collision-free, regardless of the size of surrounding obstacles. Second, a multi-objective function will guide the expansion of the path towards the safest configurations. Our safety notion consists of avoiding dangerous situations, e.g. being very close to the obstacles, human awareness, e.g. being as much as possible in the human vision field, as well as keeping the robot far from singular and joint limits configurations to be able to promptly respond to unforeseen events. Simulations are conducted in two scenarios and validated on the real Baxter robot. They effectively reveal the efficiency of the proposed method.

3.2 Introduction

In recent years, human-robot collaborative tasks has become a very active research field. In manufacturing, it is particularly interesting to carry out collaborative tasks that involves humans and industrial robots. A simple case of physical human-robot interaction consists in handing over an object from one agent to another, as shown in Fig. 3.1. However, the latter have been largely seen as dangerous machines that are kept inside security fences.

Therefore, before sharing the work space and interacting with a potentially harmful equipment, safety has to be always guaranteed. In that regard, all stages of the manipulator design have been considered. At the hardware level, modifications of the intrinsic properties of the robot can make it safer, for instance by using light-weight materials [Loughlin

et al., 2007], passive compliant systems [Wang *et al.*, 2007] or safe actuator designs [Ham *et al.*, 2009].

Another strategy is aimed at reacting in a fast and reliable way to an unexpected collision. To do this, collision detection methods have been developed [Cho *et al.*, 2013; Haddadin, 2013]. In the case of unforeseen contact, a collision reaction [De Luca *et al.*, 2006] is initiated to escape from the hazardous situation.

Ensuring safety during a human-robot cooperative task can also be done by avoiding any potential collision. The main pre-collision strategy consists in developing a real-time control system that senses the environment and adapts the robot's behavior according to an updated danger assessment [Flacco *et al.*, 2012; Kulić et Croft, 2007; Zanchettin et Lacevic, 2012]. In unstructured environments involving humans, it is clear that a pre-collision real-time control is an indispensable tool to preserve safety. However, the overall efficiency relies on the way the initial trajectory is defined, for instance, by providing a path that guarantees human's safety using a motion planning algorithm, and then modify that path in real-time to deal with the dynamic environment.

Most of the modern motion planning algorithms for robotic systems with high degrees of freedom are, generally, based on sampling methods since an exact consideration of the obstacle space is practically impossible [LaValle, 2006]. Many research in robotics has been done on safety and replanning for car-like model [Bekris et Kavraki, 2007; Parthasarathi et Fraichard, 2007], but our interest lies in the safety issue for arm manipulators.

In [Lacevic et Rocco, 2013; Lacevic *et al.*, 2011], a safety criterion was included in several motion planning algorithms. It is a simple version of the *Kinetostatic danger field* [Lacevic et Rocco, 2010a] which takes into account the overall position of the manipulator with respect to the obstacles. The safety measure is then embedded into a heuristic function that guides the exploration of the free configuration space.

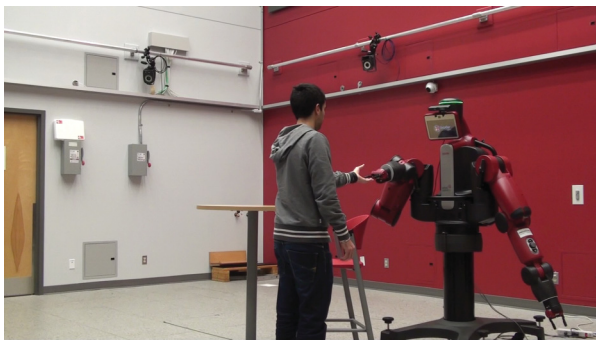


Figure 3.1 Imitation of a hand-over task.

An interesting concept of "Legible motion", which is motion that communicates its intent to a human observer, has been proposed in [Dragan et Srinivasa, 2013]. However, as pointed out by the authors, this concept suffers from several limits. For instance, optimizing legibility leads to learning and optimizing non-convex functions in high-dimensional spaces.

A manipulation planner specifically designed for human-robot interaction was presented in [Sisbot et Alami, 2012]. It generates safe paths that also improve the feeling of comfort perceived by humans during the interaction task. Human-based indications, such as visibility and posture, lead to the development of specific cost functions that help the planner to select an appropriate path.

The aim of this paper is to introduce a new safety-oriented motion planning approach for robotic manipulators operating in unstructured environments. It is based on a multi-objective optimization method combined with an exact collision checker algorithm that helps any sampling-based algorithm to find the safest path. The objectives are mainly evaluated in the Configuration-space (C-space) for reasons of speed and efficiency. Our approach relies on reducing the probability for a hazardous situation to occur, by not only maximizing the clearance of the path, but also maintaining the robot in the best configuration to react in case of emergency. In a manufacturing context, our algorithm finds a compromise between safety and performance of the task. However, since our algorithm provides the safest path for a static environment as it is the result of a motion planning algorithm, this path should be modified online to deal with a dynamic environment in real-time. The latter task will be the focus of our future work, however it is worth to mention a related work in the literature [Quinlan, 1994].

This paper is organized as follows. In Section 3.3, the concept of dynamic collision checking and its extension is introduced. Sections 3.4 describes the developed optimization objectives. Section 3.5 deals with the implementation of a sampling-based algorithm, while conducting experiments in different scenarios are presented in Section 3.6.

3.3 Exact Collision Checking Method

Collision checking is an essential step in motion planning as it ensures the path to be collision-free. However, this is also one of the most computationally expensive operation in the process of finding a free path, especially in a complex environment. The main challenge relies on determining whether the continuous path between two states in C-space is in collision or not. In sampling-based algorithms, a commonly used method consists in testing individual configurations by linear interpolation between the states. The principal

limitation of such an approach is that a compromise must be made when selecting the number of samples. Decreasing it leads to a loss of accuracy and could yield to a collision omission. On the other hand, increasing it could as well increase the computational time for collision checking, which is practically inefficient strategy in situations where the robot is far from obstacles. In a context of unstructured environments where safety of the robot, and possibly of humans interacting with it, is a priority, a computationally efficient method for exact collision checking is primordial. The guarantee of a collision-free path has been introduced by Quinlan in [Quinlan, 1994] and is obtained by using overlapping free bubbles in the C-space along a segment. This concept was then integrated in several motion planning strategies [Vahrenkamp *et al.*, 2007, 2011]. We opted for the dynamic collision checking method developed in [Schwarzer *et al.*, 2004], as it avoid restricting the workspace and can be easily adjusted to meet the application needs. We first adapted it to a 7-DOF robotic manipulator, and then proposed an extension to get information about the minimum distance to obstacles along a collision-free path.

3.3.1 Description of Dynamic Collision Checking Method

The main idea behind this method is to establish a sufficient condition of collision-free by computing the geometric path of rigid bodies in the workspace. Lemma 1 in [Schwarzer *et al.*, 2005] states that a sufficient condition to guarantee that two rigid objects, A_1 and A_2, do not collide at any configuration q located on the path segment π, which is joining two configurations q_a and q_b, is to verify the following inequality :

$$\lambda_1(q_a, q_b) + \lambda_2(q_a, q_b) < \eta_{12}(q_a) + \eta_{12}(q_b) \tag{3.1}$$

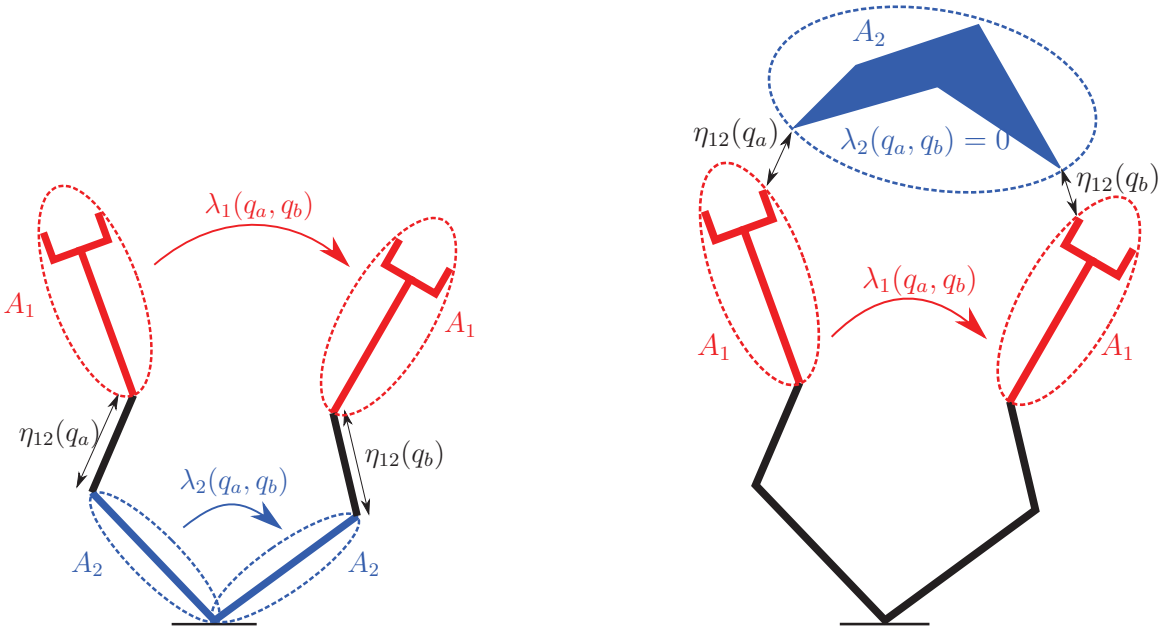
where $\eta_{12}(q_i)$ is the minimum distance between objects A_1 and A_2 for a given configuration q_i, and $\lambda_i(q_a, q_b)$ refers to the maximum Euclidean displacement of all the points in object i along the path segment π.

In the case of a robotic arm, we focus on the two following situations (represented in Fig. 3.2) :

- Auto-collision detection : Both objects A_1 and A_2 are distinct links of the robot. When the kinematic chains of a manipulator has a high degree of freedom, it is possible that two distinct links of the robot could potentially collide in some configurations.

- Detection of collisions with the environment : A_1 is a link of the robot and A_2 is an obstacle in the world. As it is an off-line analysis, we assume that objects in the environment are static. Consequently, in this case, $\lambda_2(q_a, q_b) = 0$ in (3.1).

To make a complete collision check for a given path segment, we have to evaluate all pairs of elements which could potentially collide using (3.1). If the test succeeds for a pair, those two objects do not collide and we move on to the next pair. If the inequality is not verified, the path may not be collision free. However, that does not mean a collision necessarily exists, more tests are needed. To this end, an intermediate state on the path segment π is added. This new configuration divides the initial segment in two parts. If a collision is detected for this state, the motion is then invalid. Otherwise, each segment is evaluated independently in the same manner for the same two objects. If both subsegments satisfy the inequality, the path segment is collision-free for the given pair of elements, if not, the process is repeated for the subsegments that failed the test. More details about the method are given in [Schwarzer *et al.*, 2005].



(a) Auto-collision (b) Collision with an external object

Figure 3.2 Example : Two types of collision analysis for a 3-DOF robotic arm. In each case, an attempt is made to determine whether objects A_1 and A_2 may collide while the robot moves from an initial position q_a to a final one q_b.

The procedure of applying this method to a multi-DOF robotic arm is also presented in [Schwarzer *et al.*, 2005]. In particular, the objective is to compute the term $\lambda_i(q_a, q_b)$ in (3.1) to verify the inequality constraint. Computing the exact displacement of a multi-

DOF robotic arm between two configurations could be a complex process, and therefore a time-consuming operation. However, an upper bound of this distance can be quickly figured out by only considering the displacement range of each joint and the dependency between the links movements.

The original method considers the case of a single link $\mathbb{L}$ of length L, which is controlled by a revolute joint. In this case, an elementary joint variation $d\theta$ leads to a traveling distance for the link of

$$ds = Ld\theta \tag{3.2}$$

Fig. 3.3(a) illustrates the total displacement of the link between configurations q_a and q_b. The point $P \in \mathbb{L}$ which travels the longest distance is located at the end of the link and moves from P_A to P_B. The total length s is :

$$s = \int_{\theta^a}^{\theta^b} Ld\theta = L(\theta^b - \theta^a) \tag{3.3}$$

However, the case where two revolute joints θ_1, θ_2 control the same link has not been considered in [Schwarzer *et al.*, 2005]. This case is particularly interesting for modern redundant robotic arms, and it is the case for the Baxter robot. We therefore developed a method to compute the maximum displacement for the above-mentioned case. First, note that the elementary displacement of a point P now depends on the variation of the two joints [Kay, 1988] and is defined as :

$$ds = L\sqrt{(d\theta_2)^2 + \sin^2\theta_2\,(d\theta_1)^2} \tag{3.4}$$

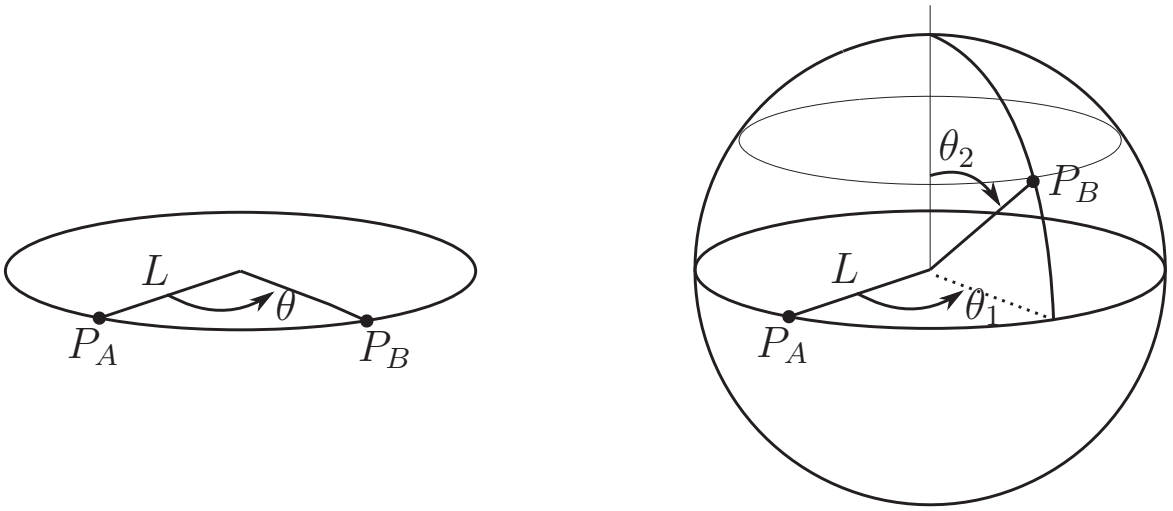
The total traveled distance can be defined by :

$$s = \int_{\theta_1^a}^{\theta_1^b} \int_{\theta_2^a}^{\theta_2^b} L\sqrt{(d\theta_2)^2 + \sin^2\theta_2\,(d\theta_1)^2} \tag{3.5}$$

By linearly interpolating between the configurations q_a and q_b, we obtain $d\theta_2 = \beta d\theta_1$, where β is a constant. Let us suppose that $\sin^2\theta_2^* = \max_{\theta_2^a \leq \theta_2 \leq \theta_2^b}(\sin^2\theta_2)$, thus the integral (3.5)

can be upper bounded by a simple integral :

$$\begin{aligned} s \leq \bar{s} &= \int_{\theta_1^a}^{\theta_1^b} L\sqrt{\beta^2 + \sin^2\theta_2^*}\; d\theta_1 \\ &= L\sqrt{\beta^2 + \sin^2\theta_2^*}\;\left(\theta_1^b - \theta_1^a\right) \\ &= L\sqrt{\left(\theta_2^b - \theta_2^a\right)^2 + \sin^2\theta_2^*\left(\theta_1^b - \theta_1^a\right)^2} \end{aligned} \tag{3.6}$$



(a) Displacement of a link controlled by one joint

(b) Displacement of a link controlled by two joints

Figure 3.3 Relation between displacements in C-space and workspace.

3.3.2 Extension to Minimum Obstacle Distance Computation

In [Schwarzer *et al.*, 2004], it has been proven that a minimum clearance δ between objects A_1 and A_2 at any configuration q between q_a and q_b is guaranteed if :

$$\lambda_1(q_a, q_b) + \lambda_2(q_a, q_b) < \eta_{12}(q_a) + \eta_{12}(q_b) - 2\delta \tag{3.7}$$

This feature is interesting, however it suffers from several disadvantages : I) As the minimum clearance is fixed, the planner might ignore a narrow passage, which would be the only feasible solution ; II) The criterion in (3.7) does not allow to compare two clear paths. To increase safety, we would prefer selecting the one which has the maximum clearance.

For a path where two objects A_1 and A_2 move between two configurations q_a and q_b, we define the following parameter :

$$\delta = \frac{\eta_{12}(q_a) + \eta_{12}(q_b) - \lambda_1(q_a, q_b) - \lambda_2(q_a, q_b)}{2} \tag{3.8}$$

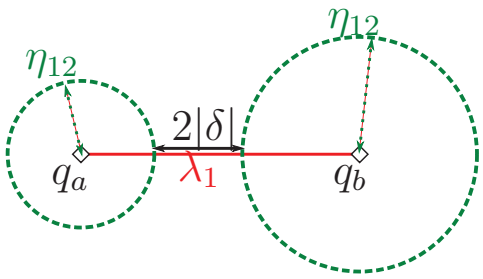
If A_2 is a fixed obstacle, $\lambda_2(q_a, q_b) = 0$ and (3.8) becomes :

$$\begin{aligned} \delta &= \frac{\eta_{12}(q_a) + \eta_{12}(q_b) - \lambda_1(q_a, q_b)}{2} \\ &= dist(q_a, q_b, A_1, A_2) \end{aligned} \tag{3.9}$$

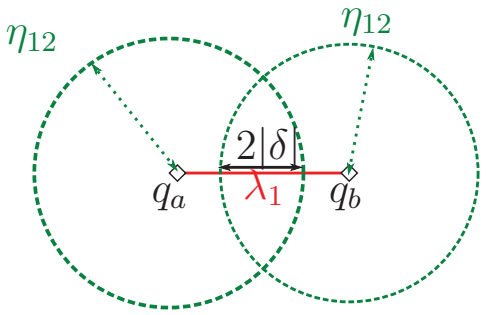
It is worth pointing out that the value of δ can be either negative or positive :

- $\delta \leq 0$ implies that the motion can not be guaranteed to be collision-free, this is because (3.1) is not satisfied. A sub-segmentation is necessary.
- $\delta > 0$, then δ is the minimum distance between A_1 and A_2 along the path.

A schematic example that illustrates the above two cases is given in Fig. 3.4. For more details about this representation, refer to [Schwarzer *et al.*, 2004].



(a) The potentially maximum traveled distance by the link is higher than the sum of minimum distance from the object at each configuration ($\delta \leq 0$)



(b) The two circles fully cover the link displacement and a margin distance from the obstacle is guaranteed ($\delta > 0$)

Figure 3.4 Illustration of the relations between distances in the workspace.

The procedure to compute the minimum clearance along a path segment is given in Algorithm 1. Note that each element of the structure segment refers to a specific pair of link/obstacle evaluated between two states and is used to store the corresponding distance information. Parameter ϵ can be defined as the maximum admissible error on the distance estimation. It is a positive user-defined constant that affects the performance of the algorithm : decreasing it improves the returned distance estimation accuracy whereas increasing it reduces the required computational burden to generate the estimation, the impact of ϵ on the accuracy and the computational complexity can be easily observed in

Algorithm 1 Minimum obstacle distance along a path

procedure CLEARANCE(q_a, q_b)
  priority_queue Q (▷) segments sorted by increasing δ
  for $i \leftarrow 1 : nb\_links$ **do**
    for $j \leftarrow 1 : nb\_obstacles$ **do**
      $\delta \leftarrow dist(q_a, q_b, A_i, A_j)$ (▷) Equation (3.9)
      if $(\eta_{ij}(q_a) \leq 0)||(\eta_{ij}(q_b) \leq 0)$ **then**
        return $COLLISION$
      else
        $Q.Add\_Segment(q_a, q_b, i, j, \delta)$
      end if
    end for
  end for
  repeat
    segment $s \leftarrow Q.Top()$
    $Q.Pop\_Top()$
    $q_{new} \leftarrow Interpolate(s.q_a, s.q_b)$
    $\delta_1 = dist(s.q_a, q_{new}, s.i, s.j)$
    $\delta_2 = dist(q_{new}, s.q_b, s.i, s.j)$
    if $\eta_{ij}(q_{new}) \leq 0$ **then**
      return $COLLISION$
    else
      $Q.Add\_Segment(s.q_a, s.q_{new}, s.i, s.j, \delta_1)$
      $Q.Add\_Segment(s.q_{new}, s.q_b, s.i, s.j, \delta_2)$
    end if
  until $Q.Top().\lambda \leq \epsilon$ (▷) λ is defined in (3.1)
  if $Q.Top().\delta \leq 0$ **then**
    return $COLLISION$
  else
    return $Q.Top().\delta$
  end if
end procedure

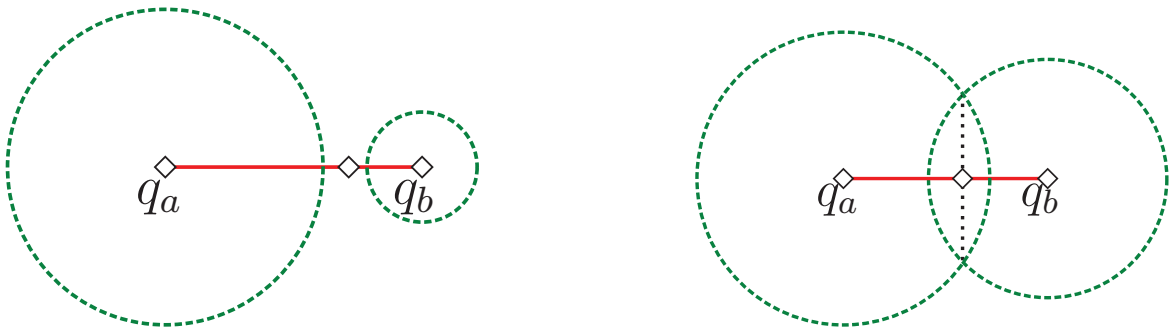
Table 3.1. The value of Δ, the error between the exact value and estimated one, respect the following inequality (see Appendix for the proof) :

$$0 \leq \Delta \leq \frac{\epsilon}{2} \tag{3.10}$$

As it has been mentioned in Section 3.3, if the inequality (3.1) is not satisfied, a subsegmentation by adding an intermediate state is necessary. However, the location of that state can have a significant impact on the algorithm convergence rate. Many covering strategies have been proposed in [Schwarzer *et al.*, 2005], however we selected the strategy that samples toward the part on the path that is most likely to be in collision. This strategy yields to a minimum number of segments evaluation, and can be described as follow :

- When $\delta < 0$, the new sampled state corresponds to the middle of the uncovered length (Fig. 3.5(a)).
- When $\delta \geq 0$, the new sampled state is the projection of the intersection of the two circles onto the path segment (Fig. 3.5(b)).

We refer to this strategy as the $Interpolate(q_a, q_b)$ procedure in Algorithm 1.



(a) Strategy when $\delta < 0$ (b) Strategy when $\delta \geq 0$

Figure 3.5 Illustration of the covering strategy.

3.4 Multi-objective Optimization Path Planning

This section presents several objective functions to be evaluated for each potential path segment connecting two random states. Obviously, the computation time of the algorithm depends on the complexity of the objective functions, therefore we focused our efforts on the development of both relevant and computationally efficient objective functions. All the objective functions are positive functions.

3.4.1 Objectives Combination Strategy

The conventional way of solving a multi-objective optimization problem is to perform a weighted aggregation of all the objective functions in order to obtain a single-objective. Thus, two *positive* objectives, f_1 and f_2, can be linearly combined by the following function lc :

$$\mathrm{lc}(f_1, f_2) = \alpha_1 f_1 + \alpha_2 f_2 \tag{3.11}$$

where α_1 and α_2 are positive constants. This combination represents the cost of a path. Subsequently, the optimization process selects the path that minimizes or maximizes the sum of objective functions depending on the desired performance outcome.

However, this straightforward formulation is not suitable for our application. This is because we aim at minimizing some objective functions, e.g. the path length, and maximizing others, e.g. the clearance to obstacles. However, a simple solution to overcome this issue is to reverse some objectives. For instance, if our objective is to minimize $\mathrm{plc}(q_a, q_b)$, which is the path length cost for a path between states q_a and q_b, then $\mathrm{plc}^*(q_a, q_b) = \frac{1}{\mathrm{plc}(q_a, q_b)}$ can be the new path length objective to maximize. A new issue however arises. Indeed, the possibility to unconditionally control the influence of each objective on the final result depends on several parameters : the value of the weighting factor, the definition of the cost function, and the conditions in which the task is executed. The last point is the main challenge because our objective is to develop a method that can be applied regardless of the environment in which the robot is operating. An example to illustrate the last point is for the clearance objective function : assuming that it is simply defined as the distance to the nearest obstacle, this objective would have less impact in a cluttered space where the robot has no choice but to pass close to obstacles than in a free space. In a clear environment, the objective will tend to prevail over others aside from the weighting factors.

To overcome the above limitations, we developed a cost combination strategy that manages the effect of each objective in an efficient way regardless of the operating environment.

An example to illustrate our cost combination strategy is given in Fig. 3.6, which shows two path segments P^* and $\hat{P}$ in a sampling-based tree planner. Our strategy to compare P^* and $\hat{P}$ can be summarized as follows :

- The set of objective functions is $\{f_1, f_2, \cdots, f_N\}$. The notion of cost is no longer defined as a single scalar value but as a vector $\boldsymbol{v}$ gathering all the objective functions : $\boldsymbol{v} = [f_1, f_2, \cdots, f_N]$.

- Let us suppose that we are interested in maximizing the subset $\{f_1, f_2, \cdots, f_M\}$ and minimizing the subset $\{f_{M+1}, f_2, \cdots, f_N\}$ $(M \leq N)$.

- We choose a set of weighting factors $\{\alpha_1, \alpha_2, \cdots, \alpha_N\}$, where $\alpha_i \geq 0$ and $\sum_{i=1}^{N} \alpha_i = 1$.

- Let us suppose that $\boldsymbol{v}^* = [f_1^*, f_2^*, \cdots, f_N^*]$ is the cost associated to P^* and $\hat{\boldsymbol{v}} = [\hat{f}_1, \hat{f}_2, \cdots, \hat{f}_N]$ is the one associated to $\hat{P}$.

- We compute the following scalar parameters :

$$g^* = \alpha_1 \frac{f_1^*}{\hat{f}_1} + \cdots + \alpha_M \frac{f_M^*}{\hat{f}_M} + \alpha_{M+1} \frac{\hat{f}_{M+1}}{f_{M+1}^*} + \cdots + \alpha_N \frac{\hat{f}_N}{f_N^*}$$
$$\hat{g} = \alpha_1 \frac{\hat{f}_1}{f_1^*} + \cdots + \alpha_M \frac{\hat{f}_M}{f_M^*} + \alpha_{M+1} \frac{f_{M+1}^*}{\hat{f}_{M+1}} + \cdots + \alpha_N \frac{f_N^*}{\hat{f}_N}$$

- If $g^* > \hat{g}$, the cost $\boldsymbol{v}^*$ is considered better than $\hat{\boldsymbol{v}}$ (noted $\boldsymbol{v}^* > \hat{\boldsymbol{v}}$) and the path P^* is preferred over $\hat{P}$, otherwise the contrary applies.

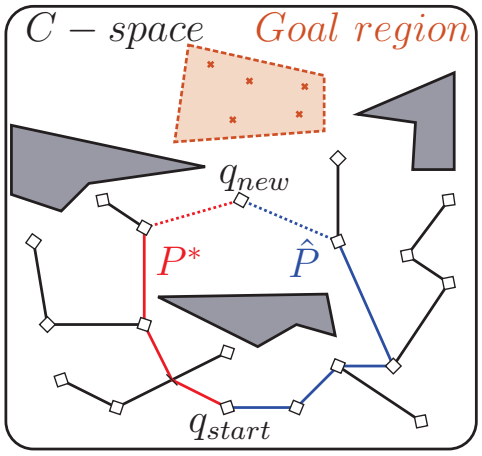


Figure 3.6 Cost function comparison between paths P^* and $\hat{P}$ to figure out which path the state q_{new} will be connected to in a sampling-based tree planner.

3.4.2 Clearance Objective

To ensure safety of the robot as well as humans interacting with it, the first objective function to consider is the distance between the robot and the obstacles or humans. As mentioned in Section 3.3, we can estimate the clearance along a path segment joining q_a and q_b as the distance provided by the developed dynamic method.

When the robot's environment is shared with humans, the clearance objective has to consider human safety at the highest level. To this end, each obstacle evaluated in Algorithm 1 is combined with a weighting factor (wf) that reflects the drive to get away from it. Fig. 3.7 illustrates the purpose of such a consideration. For instance, we classified obstacles in four categories and assigned a specific value to each, by increasing order of safety priority :

- For an object, $wf = 1$
- For a human arm or leg part, $wf = 0.8$
- For a human torso, $wf = 0.6$
- For a human head, $wf = 0.5$

To take this parameter into account, the procedure presented in Algorithm 1 remains exactly the same, but the sorting rules of the priority queue that gathers the segments are determined using Algorithm 2. In doing so, weighting factors are considered only when the two segments are ensured to be collision free. If this is not the case, the most likely colliding segment remains the first being evaluated to avoid computational overhead.

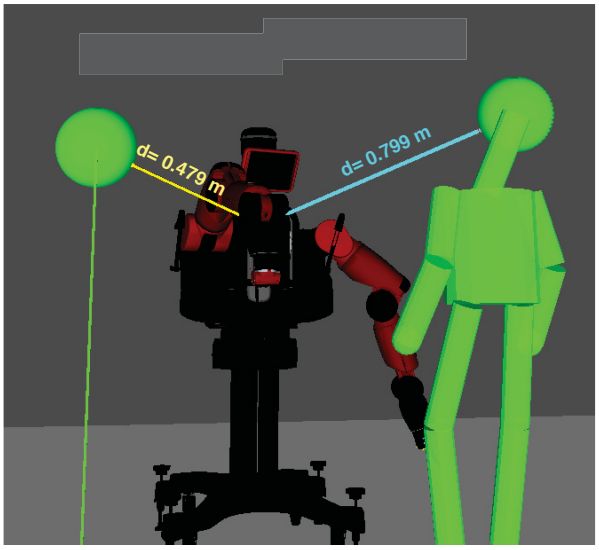


Figure 3.7 Effect of the weighting factor : even if the balloon is closer to the robot than the human's head, the latter is considered more dangerous when the distances are multiplied with the corresponding factors.

Note that the cost returned by Algorithm 1 is no longer necessarily the minimum obstacle distance but it is the one evaluated as the most dangerous. It has to be always associated with its weighing factor, particularly for the cost combination. Indeed, two paths seg-

Algorithm 2 Priority queue sorting rule

```
procedure PRIORITY_SEGMENT(s1, s2)
    if s1.δ > 0 AND s2.δ > 0 then
        if (s1.δ * s1.wf) < (s2.δ * s2.wf) then
            return s1
        else
            return s2
        end if
    else
        if s1.δ < s2.δ then
            return s1
        else
            return s2
        end if
    end if
end procedure
```

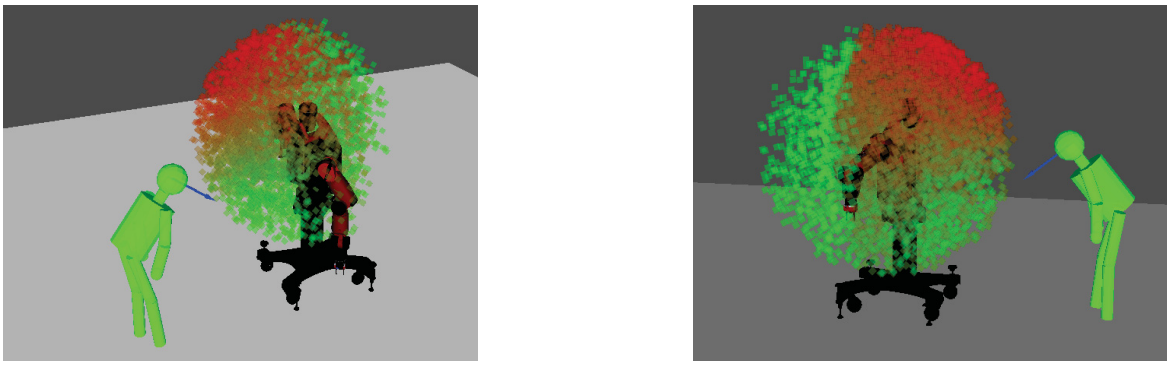
ments, which have costs of c_1 and c_2 and weighting factor of wf_1 and wf_2 respectively, are combined as follow :

$$\text{Combine}_{\text{clearance}}(c_1, wf_1, c_2, wf_2) = \min(c_1 * wf_1, c_2 * wf_2) \tag{3.12}$$

3.4.3 Human Awareness Objective

Another important issue regarding human safety is the human-robot awareness [Corke, 1999]. When one or many people enter in the vicinity of the robot (inside or just outside the reachable space), it is necessary to ensure that the manipulator movements are seen by everyone, otherwise that may increase the risk of creating a dangerous situation. In order to achieve this, we developed an objective function that focuses on keeping the robot arm in the humans' field of view. The objective is to minimize the angular distance between the human gaze direction and the axis defined by the robot's end-effector position and the human head, as illustrated by Fig. 3.8. For the cost of this objective between two configurations q_a and q_b, a linear interpolation is done along the path and return the worst value among the tested states. The cost resulting of two path segments combination, that have costs of c_1 and c_2, is computed by taking the worst case :

$$\text{Combine}_{\text{awareness}}(c_1, c_2) = \min(c_1, c_2) \tag{3.13}$$



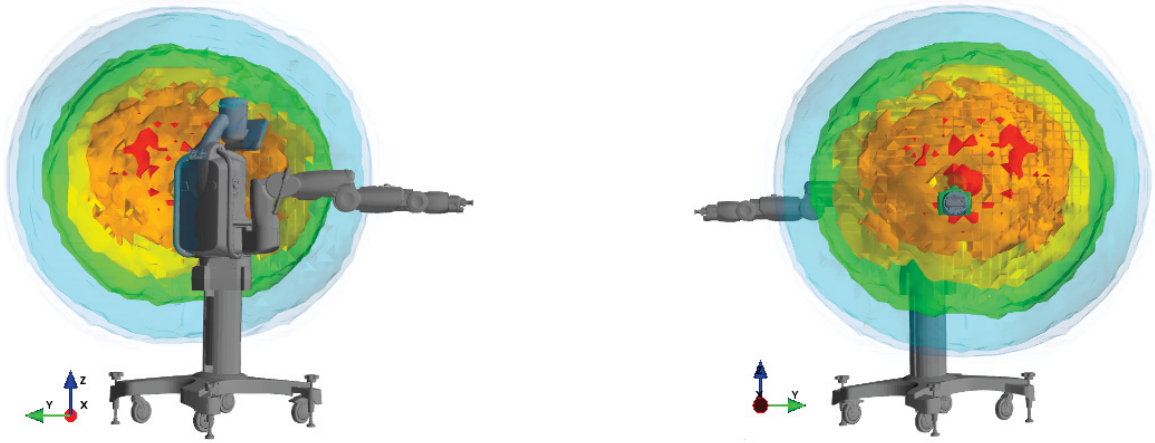
(a) (b)

Figure 3.8 Point cloud representation of the human awareness : colors vary from green to red according to the eye gaze deviation from the end effector. If the distance between the human and the end-effector is greater than a threshold, the criterion is then considered fulfilled.

3.4.4 Manipulability Objective

In a human-robot interaction task in an unstructured environment, we are interested in maximizing the degree of manipulability of the robot to efficiently cope with unforeseen events. Two kinds of manipulability analysis are studied in the literature :

- Manipulability in the workspace : the kinematic reachability space gives the total number of possible configurations at each pose of the end-effector (Fig. 3.9).
- Manipulability in the C-space : a manipulability index could be defined as a combination of the distances from the nearest singularity and the mechanical limits.



(a) (b)

Figure 3.9 6D Kinematic reachability space of Baxter's left arm. The color density increases according to the number of possible configurations.

Singularity objective

A manipulability index for redundant manipulators was originally introduced by Yoshikawa [Yoshikawa, 1985]. For a given state, it describes the distance to singular configurations. The manipulability index m is expressed as :

$$m = \sqrt{det(JJ^T)} = s_1\, s_2 \cdots s_n \tag{3.14}$$

where J is the Jacobian matrix and $\{s_i : 1 \leq i \leq n\}$ are the singular values of J. As our algorithm is operating in C-space, we generated an off-line database of the manipulability index m for all the possible configurations (sampling with a resolution of 0.1 rad). The singular configurations are then associated to hyperspheres in the C-space by defining a configuration as singular if $m \leq \epsilon_m$ $(0 < \epsilon_m \ll 1)$. The cost associated to a path segment is d_{hs}, which is the Euclidean distance to the nearest hypersphere. A simple 2D example is depicted in Fig. 3.10 in which hyperspheres are represented by circles.

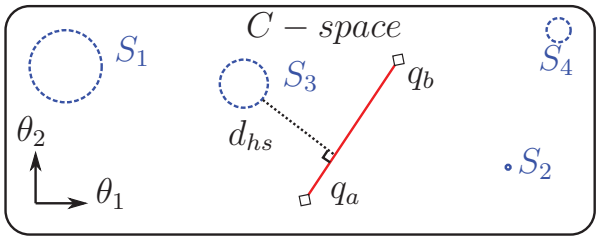


Figure 3.10 Example of distance from a singularity hypersphere in 2D. $S_1\, S_2 \cdots S_4$ contain the singular configurations.

Analogously to (3.13), the corresponding cost of combining two path segments is the worst case.

Joint limits objective

To increase manipulability, the robot also needs to always remain as far as possible from the joint limits. We denote the upper and lower limits of this joint as ${q_i}^{max}$ and ${q_i}^{min}$, respectively. Our joint limits objective f_i for each joint q_i refers to the minimum distance to ${q_i}^{max}$ or ${q_i}^{min}$.

The objective for all the joints is given by :

$$\text{State\_Cost}_{\text{jl}}(q) = \sum_{i=1}^{n} f_i(q_i) = \sum_{i=1}^{n} \min({q_i}^{max} - q_i, q_i - {q_i}^{min}) \tag{3.15}$$

where q_i is the value of the joint i, and $q = [q_1\, q_2 \cdots q_n]^T$. For a given path segment connecting q_a to q_b, the joint limit objective function is defined as follows :

$$\text{Path\_Cost}_{\text{jl}}(q_a, q_b) = \min_{q_a \leq q \leq q_b} \Big(\text{State\_Cost}_{\text{jl}}(q)\Big) \tag{3.16}$$

It can be easily proven that :

$$\text{Path\_Cost}_{\text{jl}}(q_a, q_b) = \min(\text{State\_Cost}_{\text{jl}}(q_a), \text{State\_Cost}_{\text{jl}}(q_b)) \tag{3.17}$$

By combining two path segments, similarly to (3.13), the worst case is also chosen.

3.4.5 Path Length objective

The last objective to be considered is the path length in C-space. The length of a path connecting two states in C-space is defined as the length of the straight line between those states. The path length cost for a path between q_a and q_b is simply defined as the length $d(q_a, q_b)$ of this path :

$$\text{Path\_Cost}_{\text{pl}}(q_a, q_b) = d(q_a, q_b) \tag{3.18}$$

This is the standard cost function of the original RRT* version.

The cost of combing two path segments m_1 and m_2 that connect respectively q_a to q_b and q_b to q_c is given by :

$$\text{Combine}_{\text{pl}}(m_1, m_2) = d(q_a, q_b) + d(q_b, q_c) \tag{3.19}$$

In fact, two compared paths always have the same start state (the initial configuration of the task) but also the same last configuration. As a consequence, it is appropriate to establish a direct relationship between the traveled distances.

In a general case, by combing N segments that connect the states $\{q_0, q_1, \cdots .q_N\}$, the cost function becomes :

$$\text{Combine}_{\text{pl}}(m_1, \cdots, m_N) = d(q_0, q_1) + \cdots + d(q_{N-1}, q_N) \tag{3.20}$$

Contrary to the safety-oriented objectives, the optimization process here consists in minimizing the cost value.

3.5 Integration in Motion Planning Algorithms

The purpose of our work is to integrate the aforementioned safety strategies into a motion planning algorithm. Despite their conceptual simplicity, sampling-based algorithms have been proven as an effective way to solve complex problems in high-dimensional configuration spaces, where the use of deterministic methods is computationally infeasible [Choset, 2005]. The key idea of such an approach is to avoid the exact representation of the C-space by only considering the information provided by randomly sampled states. Then, the problem resolution lies in finding a succession of collision-free connections between the start configuration to a state that respects the goal constraints.

Probably the most widespread sampling-based planners that employ randomization are the rapidly-exploring random tree (RRT) [LaValle et Kuffner, 2001] and the probabilistic roadmap algorithm (PRM) [Kavraki *et al.*, 1996]. Their inherent simplicity and effectiveness in finding a feasible path in complex spaces have made them a reference in many applications, including diverse robotic areas. These algorithms have the property of being probabilistically complete [Kavraki *et al.*, 1998][Kuffner et LaValle, 2000], that means their probability to find a solution, if one exists, approaches one as the number of samples tends toward infinity. However, the quality of the provided solution is not taken into account during the path elaboration.

Recent researches have lead to the development of numerous planners, most of them are extensions of RRT or PRM. In particular, the notion of cost has been introduced to assess the quality of a path. RRT* and PRM* [Karaman et Frazzoli, 2011] add to their respective basic version the property of being asymptotically optimal, that means that the cost of the returned solution converges almost surely to the optimal one.

Yet, these planners might lead to a solution that is far from optimal in the case of applications that are subject to time constraints. This is mainly because of the slow convergence rate. Especially, this would be a major drawback in our case because the high dimension of the C-space and the relatively important computational complexity of the cost functions may strongly impact the convergence process.

The changing needs in many fields that implies motion planning has resulted in the development of algorithms that focus primarily on finding the best compromise between the

quality of the returned solution and the time needed to obtain it. Such algorithms are potentially the most suitable for our application. The next section gives an overview of some of those motion planning algorithms.

3.5.1 Overview

PRM and its extensions belong to the family of multiple-query algorithms. The main principle is to construct a topological graph that connects sampled states of the C-space ; this defines a roadmap. This category of planners is particularly suitable for solving problems where multiple initial state and/or goal states are queried.

Even if PRM* guarantees asymptotic optimality, this condition is fulfilled at the expense of an explosive growth of the constructed graph. Dobson et al. [Dobson *et al.*, 2013] proposed the SPArse Roadmap Spanner (SPARS) algorithm, where the main idea is to relax the optimal property of PRM* to a near-optimal one by using graph spanners [Peleg et Schäffer, 1989]. By doing so, a subgraph that contains only useful states and edges is constructed alongside the denser optimal graph. This has shown that better path quality solution can be provided for queries that are time-constrained. In an advanced version, SPARS2 [Dobson et Bekris, 2013a], the near-optimally feature is preserved without the need for the dense graph to be developed. This allows for a considerable reduction in memory requirement and a production of high-quality path faster than the original version. In [Dobson et Bekris, 2013b], a new stopping criterion for PRM-like methods is proposed to provide a near-optimal solution within a finite time interval.

However, the above-mentioned improvements in roadmap-based algorithms only apply when the path length is the unique criterion to optimize. In fact, some inherent features make them inappropriate within the context of this study. The main drawback comes from the graph representation that uses non-oriented edges to define paths between states. As a result of this drawback, these algorithms have to proceed with a two-phase approach : the construction of the graph in the first instance, and then the query phase that seeks for the best solution. That means that the global cost evolution cannot be extracted during the graph construction. As shown in Section 3.6, this data can significantly improve the performance of the optimization process if it is integrated into heuristic techniques. Moreover, for the same reason, those planners do not allow the specification of a termination condition when a solution that satisfies some criteria is found.

The Fast Marching Trees algorithm (FMT*) [Janson *et al.*, 2015] uses an efficient method for solving complex motion planning problems in high-dimensional configuration

spaces. For a given problem, a single batch of samples is generated initially. Paths are then constructed using a marching method and stored in a tree structure. The sequential structure used here allow for a directly ordered search because the knowledge of the C-space does not evolve while the tree is being build. In addition, the planner uses a "lazy" behavior in its dynamic programming recursion. Thus, faster convergence rate to the optimal solution, compared to RRT* and PRM*, has been put forward by numerical experiments. Nevertheless, FMT* does not have the anytime property and, hence, suffers from two main inconvenients : (1) the whole solving process has to be restarted if the current resolution (i.e., the C-space coverage) is too low, (2) no intermediate solution can be reported and so no time constraint on the planning task can be imposed.

An interesting quasi-anytime version of FMT* has been proposed in [Salzman et Halperin, 2015]. The presented MPLB planner successively restarts the search with a refined resolution and reuses samples and connections that were previously found. It also uses a heuristic that orders the promising nodes and rejects the ones that cannot improve the current best solution. Yet, once again, the planner returns only one solution per batch and the running time is almost twice as long as FMT* for the same final resolution.

Similarly to [Dobson et Bekris, 2013b], tree-based asymptotically near-optimal algorithms have been developed. LBT-RTT [Salzman et Halperin, 2014] is able to find solution paths with cost converges to an approximation factor of $1+\epsilon$ of the optimal one. This algorithm offers a compromise between the speed of RRT and the path quality of RRT*. In this planner, many calls to the collision checker can be avoided, on the other hand a lot of cost estimations on the edges have to be carried out. As, in our case, the collision checking is included in the cost evaluation (via the clearance objective), this approach is not suitable.

A widely used strategy in motion planning consists on setting up a bi-directional search to solve single-query problems. In this sense, a bi-directional variant of RRT , RRT-Connect [Kuffner et LaValle, 2000], is able to find a solution much more quickly than its original version, particularly in complex scenarios that involve high dimensional space and cluttered environment. Upgrading to asymptotically-optimal planner has been proposed in [Akgun et Stilman, 2011] and [Jordan et Perez, 2013]. Both papers present adapted versions of bi-directional RRT* with different heuristics to speed up the search.

The bi-directional RRT* seems to be the most suitable approach in the context of this study, this is because it meets the requirements in terms of computational time and quality of the provided solution and can be easily adapted to support our objective combination strategy. However, heuristics presented in the two previous works are primarily appropriate

for the standard objective that aims at minimizing the Euclidean distance and their impact on the performances of the planner are not totally clear.

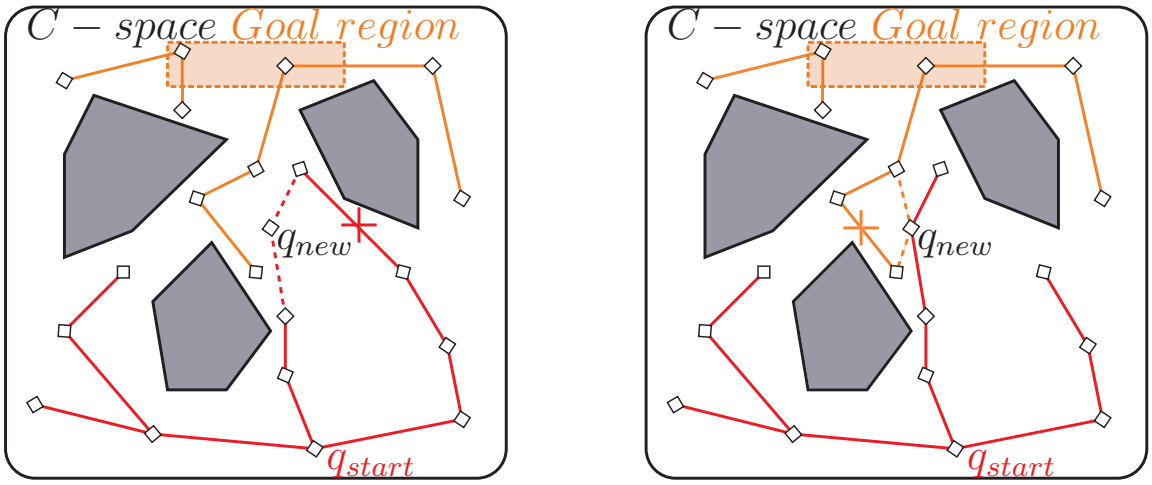
To address this, we introduce a variant of the bi-directional RRT* algorithm. This algorithm uses effective methods that limit as much as possible the inherent computational burden of our cost estimation and provide a high-quality solution in a reasonable time. Some of the implemented heuristics are inspired by those presented in [Akgun et Stilman, 2011] and [Jordan et Perez, 2013], but adapted to multi-objective and greatly improve the performance in this context.

Our main contribution here lies in the development of techniques that are particularly suitable for the use of minimax objective functions (that aim at finding the "best" worst case). Furthermore, we integrated the C-Forest parallelization framework [Otte et Correll, 2013] to take full advantage of the multi-core technology that almost every recent computer is equipped with.

The bi-directional RRT* planner consists primarily of a merger between RRT-Connect and RRT*. In prior works [Akgun et Stilman, 2011; Jordan et Perez, 2013], two trees are grown from the start and goal nodes and expanded in order to establish one or more connections between them. We extended the scope of application to consider the goal not only as a unique configuration but also as a region in the C-space, depending on the task definition. In fact, this feature is particularly useful when planning for redundant arm manipulators because it is often queried to reach a specific pose for the end-effector, and, most of the time, this could be accomplished by different joint states. Thus, we are referring here to a set of goal trees that are rooted in different states, all satisfying the goal constraints. Their data about vertex and edges are stored in a shared structure that let the algorithm behaves in a similar way to the case of a unique tree.

As in previous bi-directional RRT algorithms, two complementary operations are executed at each iteration. First, one of the trees is extended toward a newly sampled state (Fig. 3.11(a)) and, if it succeeded, it is attempted to connect the two trees through this latest configuration (Fig. 3.11(b)).

To keep the optimality property, the following procedure is applied each time an attempt is made to add a state in one of the trees : a connection is attempted between a random state and its nearest state in the tree. If the resulting path segment is allowed, that means that it respects all constraints and remains in the free space, then the state is added to the tree as a leaf. A second phase intends to ascertain the best position of this state in the tree by testing the rewire options with its nearest neighbors, and then keeping the connections



(a) The start tree is extended toward a new sample q_{new}

(b) A connection is made between the trees through q_{new}

Figure 3.11 Illustration of the bi-directional RRT* functioning when the traveled distance is optimized.

that maximize or minimize a given cost function. The rewiring process is illustrated in Fig. 3.11 and is applied for both extension and connection modes.

A description of each specific feature that we developed and is part of the final implementation is given below.

Heuristic rejection

At each iteration, a newly sampled state q_n is treated. Before attempting to integrate it as a candidate waypoint, it is relevant to assess its inherent potential to provide an improvement to the research of a path. Therefore, we use a set of admissible heuristics (one for each objective) to ascertain that a best costs could be obtained if the motion passes through this state. For our application, we need only two different heuristic functions to cover the whole set of objective functions :

- A heuristic function for the path length objective :

$$h_{pl}(q_n) = \text{Path\_Cost}_{\text{pl}}(q_{start}, q_n) + \text{Path\_Cost}_{\text{pl}}(q_n, q_{goal}) \tag{3.21}$$

  If many goal configurations exists, we consider the nearest from q_n.

- A heuristic function for minimax objectives (all the others objectives) :

$$h_{minimax}(q_n) = \text{State\_Cost}(q_n) \tag{3.22}$$

The costs of (3.21) and (3.22) are combined according to our strategy, and the resulting optimistic cost is then compared to the current best solution. If it is worse, the sample is directly rejected and a new iteration is started.

Fast estimation of motion cost

When an attempt is made to insert a new sample, the best position in a given tree is chosen by testing the rewire options with its nearest neighbors, and then keeping the connections that optimize a given cost function. As shown by Karaman and Frazzoli [Karaman et Frazzoli, 2011], this method guarantees asymptotic-optimality of RRT* and this property is preserved for the bi-directional version [Jordan et Perez, 2013]. In this process, the number of selected neighbors for evaluation increases with the number of states in the tree. Because the computation of an exact motion cost is relatively cumbersome in our case, the rewiring operation can be separated into two phases. A fast estimation of motion cost is computed for all potential connections that could link the new state and its neighbors. Similar to the previous feature, this evaluation uses an admissible heuristic, which is based solely on the costs of the states that delimit the motion. For the path length, joint limits and singularity objectives, the real motion cost is directly computed because it only depends on the segment extremities states. For other objectives, however, determining the motion cost usually requires interpolation, therefore the following formula is first applied to give an upper bound estimation of the cost between q_a and q_b :

$$\text{Fast\_Path\_Cost}(q_a, q_b) = \text{Combine}(\text{cost}(q_a), \text{cost}(q_b)) \tag{3.23}$$

In the second phase, the most promising neighbor (the one having the better estimated cost from the root of the tree) is selected and the exact motion cost is computed. If this neighbor remains the most promising (the real motion cost is still higher than the upper bound estimation of other neighbors), then the connection is established without considering the others candidates. Otherwise, the operation is reiterated while another rewiring option can possibly be better. The neighbor selection process is detailed in Algorithm 3. This procedure makes a call to the function Sort(), which employs our combination strategy to reorder the state vectors according to their associated costs in order to get the most

promising neighbor first. The function WorstCost(), that is also called in this algorithm, returns the set of all individual objective functions taken in the worst case. If an objective is to be maximized then the worst value is 0, otherwise an infinite value is returned when the objective is to be minimized.

Algorithm 3 Neighbor selection using fast cost estimation

```
procedure SELECT_BEST_NEIGHBOR(q_new, tree)
    neighbors ← tree.K_Nearest(q_new)                    (▷) k nearest
    for i ← 1 : k do
        inc_cost ← Fast_Path_Cost(neighbors[i], q_new)
        costs[i] ← Combine(neighbors[i].cost, inc_cost)
    end for
    Sort(neighbors, costs)
    best_cost ← Worst_Cost()
    best_nbh ← NULL
    for i ← 1 : k do
        inc_cost ← Path_Cost(neighbors[i], q_new)
        costs[i] ← combine(neighbors[i].cost, inc_cost)
        if Collision_Free_Path(neighbors[i], q_new) then
            if costs[i] > best_cost then
                best_cost ← costs[i]
                best_nbh ← neighbors[i]
                if i ≠ k AND costs[i] > costs[i + 1] then
                    break
                end if
            end if
        end if
    end for
    return best_nbh
end procedure
```

Local biasing

Biasing in sampling-based motion planning is a common practice that aims at steering the samples towards the most interesting regions of the C-space rather than leaving this process totally random. In this regard, numerous techniques have been proposed and their interests strongly depend on the targeted application. As many minimax objectives are used in our framework, the development of a particularly adapted local biasing strategy is necessary. Because the quality of a solution can be highly degraded by a small part of the path, it is appropriate to guide the search towards the part that affects the most the final cost, as depicted in Fig. 3.12.

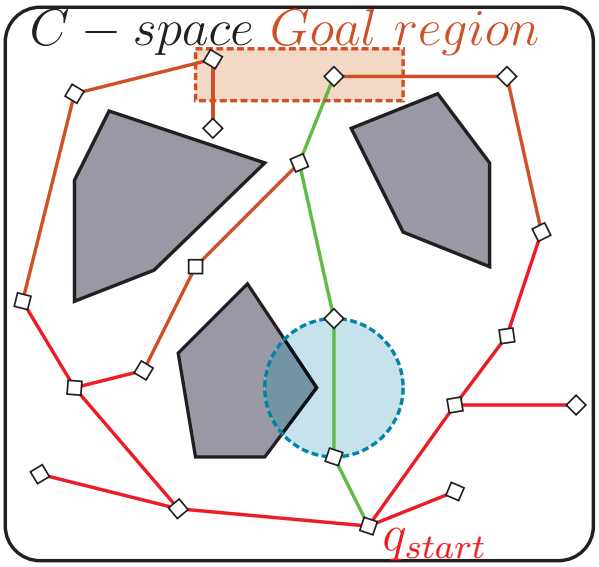


Figure 3.12 Local biasing example when the optimization process combines the clearance and the path length objectives. Considering the current best solution (green), the sampled area (blue) is located in the vicinity of the segment that has the worst clearance (as it is the only minimax objective here). The selected region is defined by a sphere that passes through the two endpoints of the segment.

Anytime behavior

The purpose of this functionality is to obtain a path as fast as possible and using the remaining available time to improve its quality. This is made possible by disabling the optimality research until a first solution is found. Thus, up to that moment, the algorithm behaves exactly like RRT-Connect. Concretely, the planner considers a unique nearest neighbor when trying to connect a new node. Because the aforementioned heuristic rejection feature needs a solution to be found before being activated, the anytime behavior is expected to contribute to improving the convergence rate.

Avoid minimax objective side effects

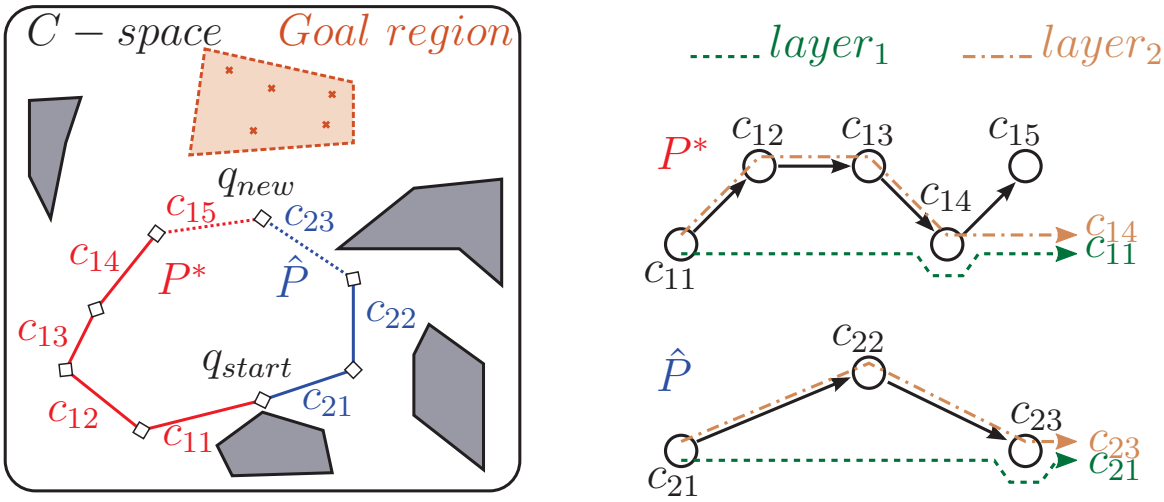
As mentioned before, minimax objective are particularly suitable for our application to avoid any vulnerable situation. However, their efficiency is bounded by the costs of imposed configurations (start or goal states). Thus, if one of these states has a poor cost, the optimization process will have no room to improve the quality of the path ; this is because our cost combination strategy only takes into account the worst case. To get around this problem, we implemented a double-layer cost system which operates as follow : the first layer is the standard one that holds the information about the worst value obtained from

the root of the tree. The second layer considers if the robot is escaping from a undesired posture, i.e., while an objective keeps improving, from the root of the tree, the best value when combining two costs is stored. From the moment at which the objective function starts decreasing, the combination considers the worst case again.

When comparing a minimax objective from two costs, the following cases arise :

- The associated values are distinct : in this case only the first layer is considered.
- The associated values are equal : the second layer is called, in this case, to decide which path has to be prioritized based on the criterion in question.

An illustration example of our two-layer strategy is presented in Fig. 3.13.



(a) Representation of the C-space

(b) Path cost combination behavior

Figure 3.13 Example : Trying to connect the state q_{new} to the tree through either path P^* or $\hat{P}$ when the clearance is the only objective to optimize. Considering the worst case only, no choice can be made because P^* or $\hat{P}$ have the same cost $c_{11} = c_{21}$. However, with the second layer, path P^* has cost c_{14} and $\hat{P}$ has cost c_{23}, so P^* is chosen.

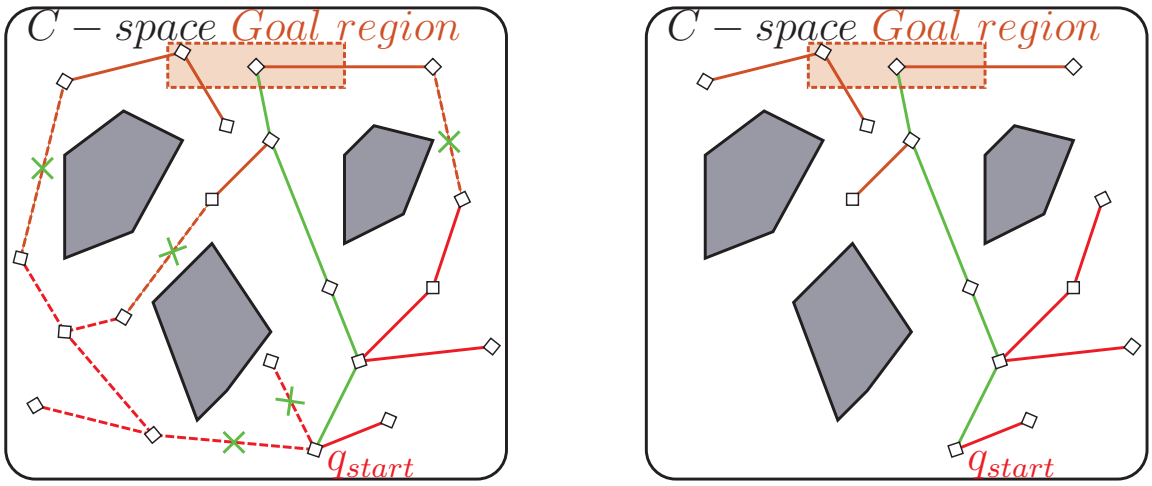
C-Forest framework

We adapted the C-Forest parallelization framework for our bi-directional RRT* implementation. The C-Forest framework consists of data exchange strategy between multiple threads of the planner that are executed in parallel. In particular, the states that constitute the current best solution are shared with the others threads. More information about this framework can be found in [Otte et Correll, 2013]. Implementation details are given

in [Javier V Gomez, 2014].

Pruning process

This functionality is integrated in the C-Forest framework but we present it separately because it can also be used as an independent feature. Once a solution is found, there may exist some states in the tree that become useless because they cannot improve the solution. Thus, it is appropriate to remove them from their respective tree. This allows the algorithm to avoid unnecessary efforts that only slow down the process. In practice, that means that the heuristic rejection presented above is now applied to vertex which are already in the data structure. Note that removing a state also erases the branches that are attached to it. It is therefore likely that a potentially interesting state in the tree could be deleted due to a bad foregoing configuration.



(a) Undesirable paths and their children are selected for removal

(b) New representation of the trees after pruning

Figure 3.14 Illustration of the pruning process applied to both start and goal trees when the clearance is optimized. The cost of the current best solution (green) is used as a reference.

3.6 Experimental Results

The efficiency and performance of the described method have been tested in simulation and on the real robot Baxter from Rethink Robotics Inc. Baxter is equipped with two 7-DOF manipulator arms. In this study, we only focus on the control of one arm but our method can be easily extended to also consider both arms simultaneously. Our algorithm has been implemented using the platform MoveIt ! [Ioan A. Sucan and Sachin Chitta, 2013]

that integrates the motion planning library OMPL [Şucan *et al.*, 2012] and communicates via ROS (Robotic Operating System) API interface [Quigley *et al.*, 2009].

3.6.1 Distance Approximation Function

Even though the dynamic collision checking method tends to limit the number of distance computations, those operations remain widely used during the planning process. Moreover, computing the exact minimum distance between two complex and concave objects is a computationally expensive operation. Moveit ! uses the Flexible Collision Library (FCL) [Pan *et al.*, 2012] as its primary collision checking library. Besides collisions check, FCL can also compute the distance between two non-overlapping objects.

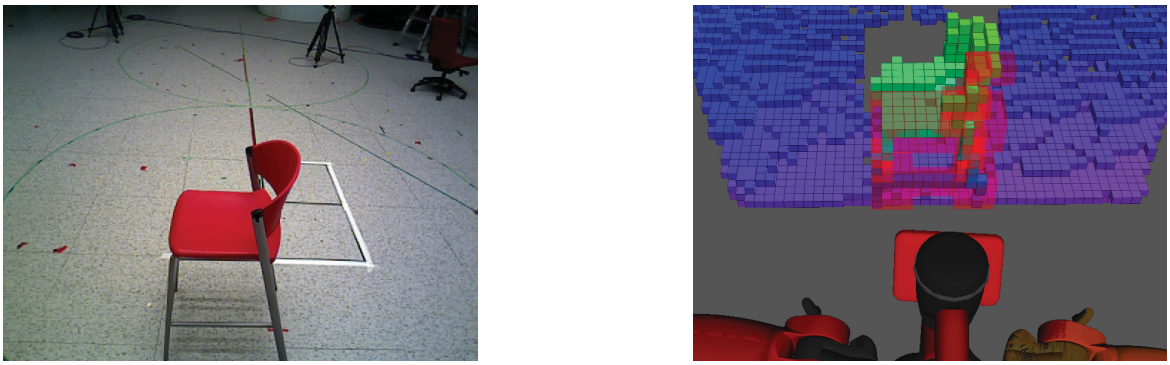
To accelerate the execution of our algorithm, we developed a computationally efficient distance computation function that gives a reasonable approximation of the distance between a link of the robot and an obstacle. This function is based on simplifying the environment representation by transforming each obstacle into its corresponding oriented bounding box (OBB). The Baxter robot is also approximated by its collision geometry model from the Unified Robot Description Format (URDF) file. Thus, the arms are modeled as cylinders and boxes. Each part of the arm is then subdivided into an optimal number of spheres that encompass the initial volume. Finally, the minimum distance between an obstacle and a link becomes the minimum distance between the OBB and the spheres and, hence, can be easily computed.

3.6.2 Environment Representation

In order to create an autonomous system, an Asus Xtion PRO 3D sensor has been attached on top of the Baxter robot's head display. The movement of the Baxter's head pan joint offers the ability to visually scan a large part of the environment. The OctoMap framework [Hornung *et al.*, 2013] integrated into Moveit ! is then used to generate a 3D occupancy grid that gives a collision model of the space. The sonar sensors positioned around the robot's head are used to detect human presence and orientate the camera accordingly. A specific model is provided to represent humans in the environment. In particular, it allows us to distinguish the different parts of the body for the assessment of the hazard level.

In order to reduce computation complexity, we developed a simplified model for the obstacles to reduce the number of distance evaluations. To this end, two operations are applied to refine the OctoMap, which is an input for our method. First, all statics objects that are outside a certain distance from the limits of the robot's workspace are neglected. Then,

we attempt to gather all remaining occupied leaves of the octree nodes into larger boxes. Our model is based on a compromise between a fair representation of the environment and having a minimal number of obstacles. An example of such representation is given in Fig. 3.15.



(a) Camera view (b) Collision model in Moveit !

Figure 3.15 Representation of the scene : the chair and the ground are modeled with occupancy boxes using OctoMap. Only red transparent boxes are considered in the motion planning process.

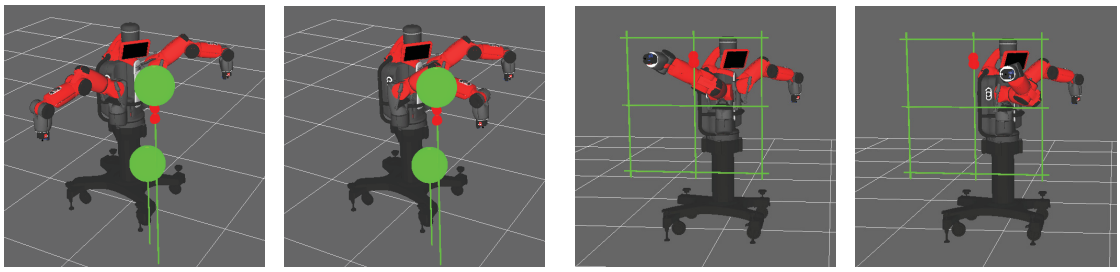
3.6.3 Numerical Experiments

The first validation phase of our approach was made through simulated scenarios using the Gazebo simulator [Koenig et Howard, 2004] as an interface for the Baxter robot. All experiments are performed on Intel(R) Core(TM) i5-3470 CPU @ 3.20 GHz PC with 8 GB RAM. As a first step, we tested each unit independently to figure out the influence of the parameters on the overall performance of our algorithm.

Two scenarios have been considered in this study :

- Scenario 1 : The right arm's is queried to reach successively two particular configurations for the joints. These states are located on the sides of two balloons tethered to the ground by thin rods as shown in Fig. 3.16(a). This scenario has been also tested on the real robot and a snapshot of this experiment is given in Fig. 3.26.
- Scenario 2 : The right arm's end-effector is planned to reach specific poses (position and orientation) passing through a wire grid leaving only narrow free spaces see Fig. 3.16(b).

Table 3.1 shows a comparison between the performances of discrete and dynamic methods in finding the minimum distance from obstacles along a path. Experiments have been



(a) Scenario 1 (b) Scenario 2

Figure 3.16 Motions that connect near states in the C-space could create a large displacement in the workspace. Discrete collision checker does not detect collisions (represented by red spheres) with the thin rod.

conducted in each scenario, using the standard distance function (FCL), and for different values of ϵ. For the discrete case, ϵ is the sampling rate in C-space. It refers to the length of the longest segment that does not require state validation and is defined as a fraction of the space maximum extent. For the dynamic method, ϵ is the maximum acceptable error on the distance evaluation between two objects.

It is worth pointing out that, for a single arm of Baxter robot, the maximum extent in the C-space is approximately 10π (the sum of the joint ranges). Thus, for a standard value $\epsilon = 5\,\%$, a segment is considered as collision free if the total corresponding joint variation is under $\pi/2$. In the worst case, this variation can produce a displacement of more than 1.7 m. Conversely, if we want to ensure a maximum error of 10 mm in the workspace, as done with the dynamic method, we have to set a value of $\epsilon = 0.03\,\%$. This is not an efficient solution as computational time tremendously increases. Therefore, the distance obtained by the discrete method, with an extremely low value of ϵ (0.01 %), is taken as a reference. We indicate by Δ the error between the reference and the estimated distance.

The sign of Δ plays a major role with respect to the safety issue. As it is always positive for the dynamic method, we ensure that no collision could occur.

Computation time is indicative only because accuracy of both methods is not comparable. Note that the use of the distance approximation function can tremendously improve computational time compared to FCL (with an average time gain of 74.6 times in Scenario 1 and 5.5 times in Scenario 2). However, this strategy underestimates the distance between the objects and might add a non-negligible error on the evaluation (7.1 mm for Scenario 1 and 52.5 mm for Scenario 2 on average). This is a good option when planning in a relatively uncluttered areas because it is very fast to compute. On the other hand, that approximation might neglect narrow passages as they could be considered in collision.

Tableau 3.1 Results for minimum distance computation along a motion

Sc.	Method	ϵ	mean(Δ) (mm)	min(Δ) (mm)	max(Δ) (mm)	Avg. time (ms)
1	Discrete	5 %	-2.0	-55.3	0.0	21.06
		1 %	-0.2	-3.2	0.0	90.30
	Dynamic	100 mm	25.1	0.0	48.9	25.23
		10 mm	2.6	0.0	5.0	48.69
2	Discrete	5 %	-3.1	-111.8	0.0	5.15
		1 %	-0.2	-11.4	0.0	22.57
	Dynamic	100 mm	25.2	0.0	49.8	3.96
		10 mm	2.5	0.0	4.9	5.60

The performances of the dynamic collision checker, which is integrated in our bi-SRRT* planner, and discrete collision checking method (via the use of classical RRT*) are compared in Table 3.2. It presents the failure rates as a result of undetected collision. We chose a fixed value $\epsilon = 10\,\text{mm}$ for the dynamic collision checker to enable the robot to operate in a complex environment, like the one in Scenario 2. The tests have been carried out in the two scenarios to point out the environmental influence. Each single test gathers 100 motion planning tasks. The allowed time for task planning is set to 20 seconds for both planners. As anticipated, the dynamic method did not omit any collision and enabled the planner to achieve all tasks. On the other hand, the discrete method tends to find inadmissible solutions due to undetected collisions.

Tableau 3.2 Task validity comparison according to the collision checking method used

Sc.	Method	ϵ	Failure
1	Discrete	5 %	100 %
		1 %	17 %
	Dynamic	10 mm	0 %
2	Discrete	5 %	96 %
		1 %	48 %
	Dynamic	10 mm	0 %

The quality of provided solutions by our bi-SRRT* planner are the subject of the next experiments. The developed framework does not allow a fair comparison with existing planners because the implemented multi-objective strategy is part of our contribution. However, it is worth to investigate the impact of the integrated features on the algorithm performance. As our bi-SRRT* is intended to work in human presence environment, the space in Scenario 1 is now populated by people, as depicted in Fig. 3.17.

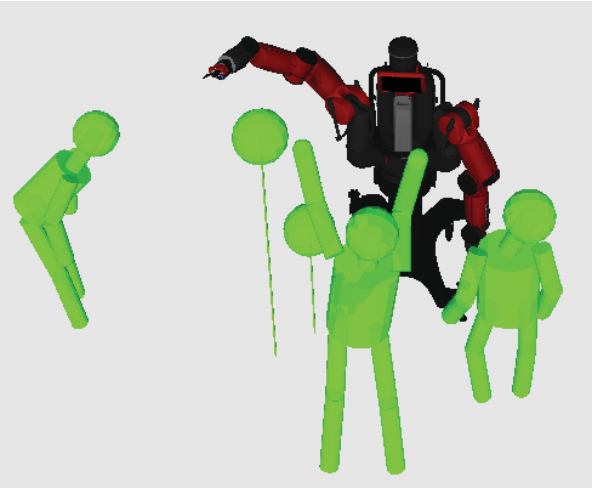


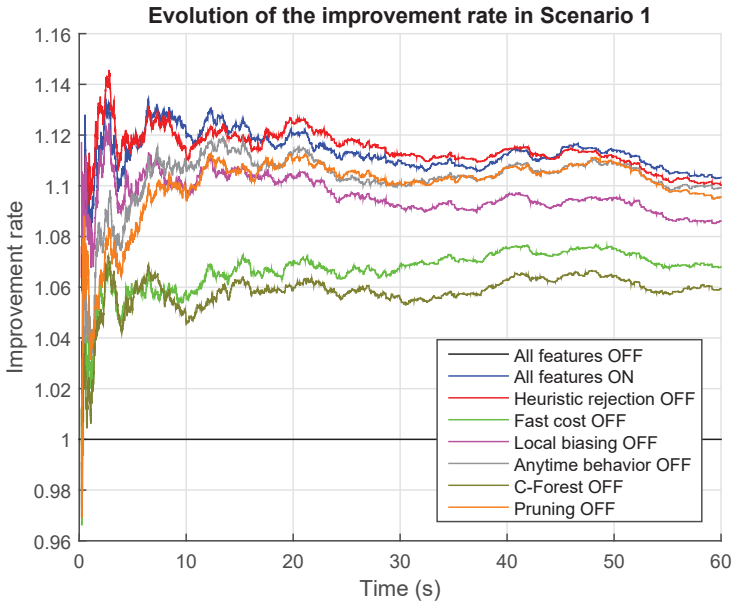
Figure 3.17 Human presence in Scenario 1.

A cost combination is defined as the following set of values : c = Clearance ; Path length ; Joint limits ; Singularities, Awareness. We associated to each scenario an arbitrary cost combination :

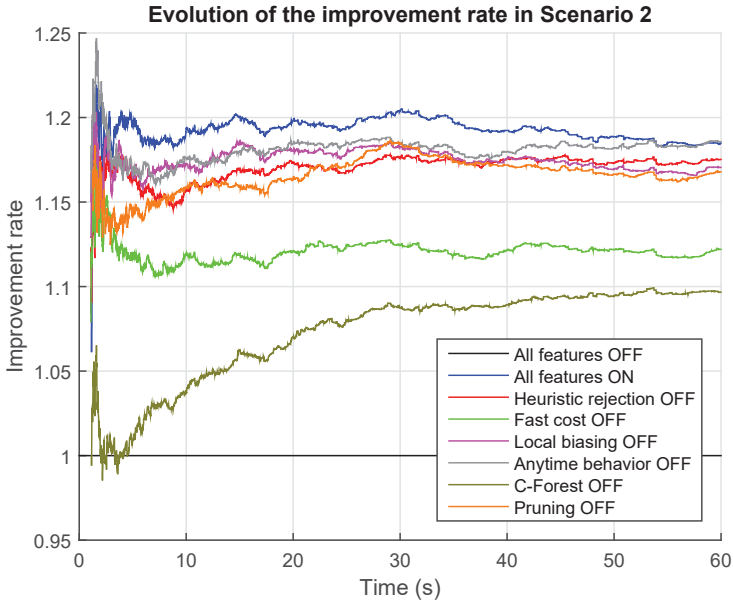
1. In Scenario 1, we prioritized the human-safety : c_1=0.4 ; 0.2 ; 0.1 ; 0.1, 0.2.
2. In Scenario 2, we assigned a balanced cost combination (considering the awareness objective is null as there is no human) : c_2=0.25 ; 0.25 ; 0.25 ; 0.25, 0.

200 motion planning tasks have been conducted for each particular experiment and, if no specification is provided, we always consider the mean value of the collected data to get an idea of the algorithm overall performance. Four threads of the planner are always launched in parallel and they exchange information only if the C-Forest framework is enabled. At a given time, only the best solution among all threads is considered.

Instead of testing the effect of activating only one feature at a time on the bi-SRRT* planner, we have found that it is more significant to study the effect of deactivating one feature at a time. Indeed, the numerical experiments show that there is a considerable dependency between the features. Moreover, a relevant benchmarking can not be made for each objective independently because they are optimized as a set and strongly depend on the value of their associated factor. The comparison is therefore made following the rules of the method presented in Section 3.4.1. First, a function $\mathcal{F}_1$ is computed when all features are deactivated and serve as a reference. Then, a new function $\mathcal{F}_2$ that results from a path cost that integrates some of the features is computed. We evaluate the improvement produced by the later configuration by generating the rate $\frac{\mathcal{F}_2}{\mathcal{F}_1}$, results are reported in Fig. 3.18.



(a) Scenario 1



(b) Scenario 2

Figure 3.18 Evolution of the improvement rate over time.

The first important observation is that the developed bi-SRRT* with all features activated leads to a non-negligible improvement over the basic version in the two scenarios. The global quality of the obtained costs is about 12% better in Scenario 1 and 20% better in Scenario 2. These values does not change a lot over time. It is worth to note that the manner in which we defined most of our objectives (with a minimax optimization) and the compromise imposed by the numerous constraints to be satisfied do not allow to obtain a huge improvement.

From Fig. 3.18, it can also be noticed that when all features are activated, we get the best paths. That means that all features make a contribution of varying degrees to the quality of obtained solutions. The one that has the greatest impact is the C-Forest framework, as its deactivation is the most meaningful in both scenarios.

Moreover, as shown by Fig. 3.19 and Fig. 3.20, the state rejection feature is clearly less efficient than the fast estimation of path cost feature. This makes sense since the first strategy only considers the cost of a unique state to predict the best path that could be generated by including it. This results in an overestimation that generally does not allow to reject the state because it has the inherent potential to be part of a better solution. It gives the corresponding number of states and rewire options inside the trees that have been carried out during the path planning. This shows that the fast cost estimation has a strong influence on those two parameters and allows the planner to have a better knowledge of the space more quickly. Other features are more or less useful depending on context, however the combination of all features provides an interesting tool for applications in unconstrained environments.

The comparisons of the cost evolution when turning all the features ON/OFF, for each objective, are presented in Fig. 3.21 for Scenario 1 and Fig. 3.22 for Scenario 2. They effectively reveal the significant advantage of the complete framework regarding constraint satisfaction. In the two cases, all the criteria are improved simultaneously and significantly.

The feature that aims at avoiding minimax objective side effects has also been evaluated within the two scenarios. Fig. 3.24 and Fig. 3.23 illustrate the evolution of each minimax objective along the motion when this feature is activated/deactivated. Since the planner is bi-directional, the process of escaping from a bad initial state is applied for both of the start and goal trees. That lets the planner produce really safer path when looking at the overall displacement, and avoid the problem of getting stuck in an imposed local minimum.

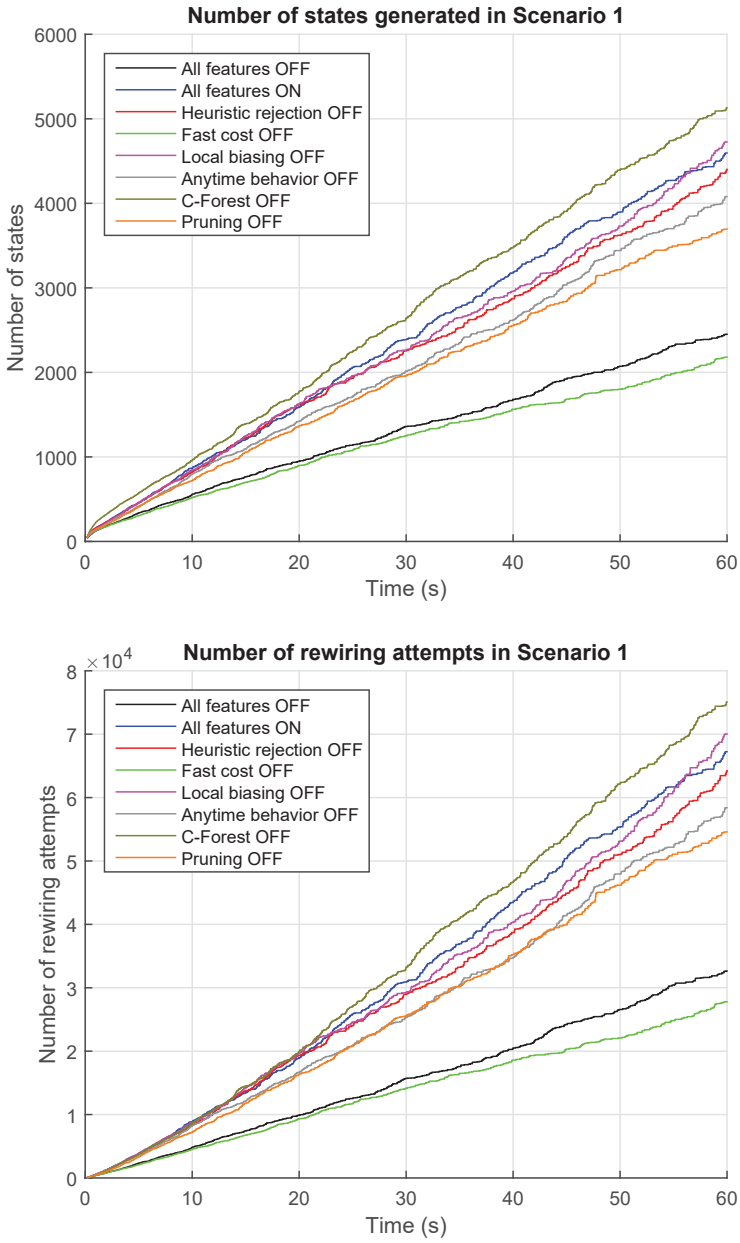


Figure 3.19 Evolution of the number of states generated and rewire options tested over time in Scenario 1.

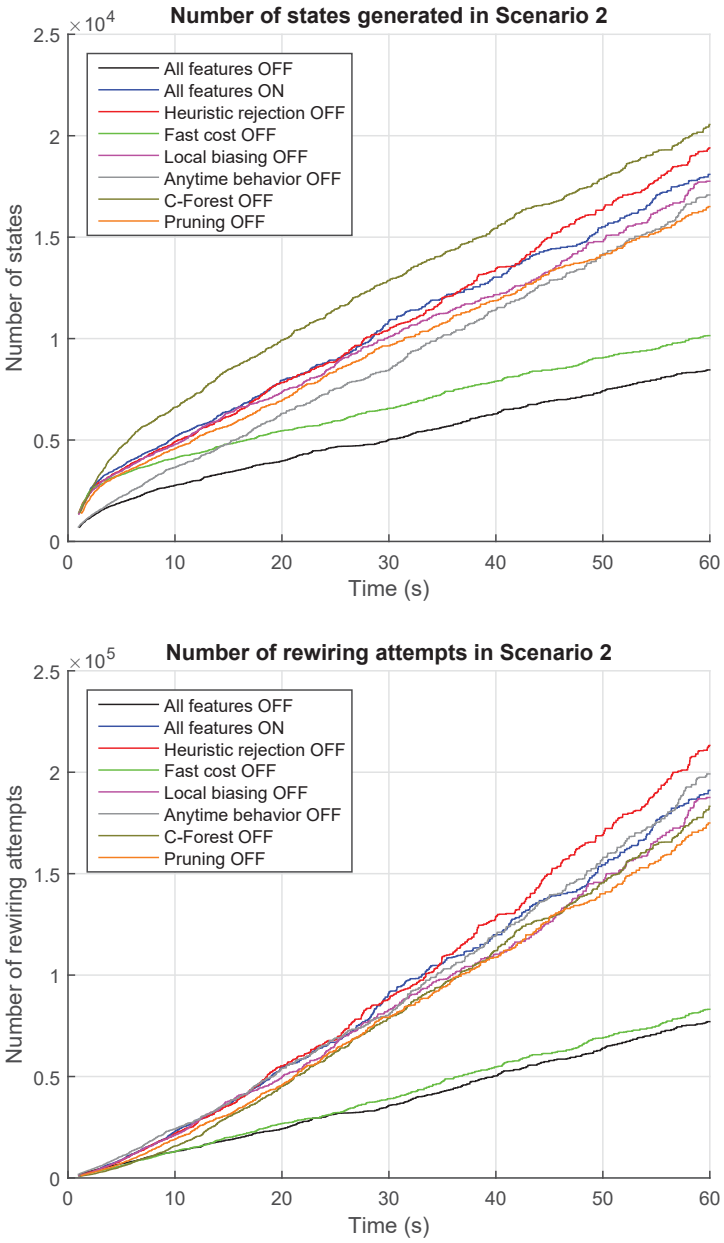


Figure 3.20 Evolution of the number of states generated and rewire options tested over time in Scenario 2.

Even when a complex task is being carried out, like the one in Scenario 2, the planner is able to find a solution quickly. The bi-directional implementation doubtless plays a major role to this end. The anytime behavior also contributes to reducing the time needed to find a first solution. Fig. 3.25 indicates that the usefulness of this feature is more significant when a trivial solution does not exist.

To validate our planner in the presence of humans, we have carried out an experiment where the Baxter robot is executing a motion planning task while considering human avoidance and awareness, a snapshot of this experiment is given in Fig. 3.27. A process that detects an interactive task intention from a human and figures out the corresponding goal to reach for the end-effector has been also developed as shown in Fig. 3.28.

3.7 Conclusion

A complete path planning framework for robotic manipulators that operate in unconstrained environment has been presented. A particular focus is given to ensure the safety of humans that may enter the robot workspace. This is accomplished through the combination of an exact collision checker and several relevant and computationally efficient objectives that guide the planner's exploration.

An adaptation of the bi-directional RRT* planner constitutes the core of our algorithm. It integrates all the safety modules and several features that aim at accelerating the optimization process.

Future work will focus on reducing the computational time to make the planner even more suitable for time-constrained applications, like human-robot interactions. To this end, we will investigate a more appropriate way to generate self-collision checking. This process is the most time-consuming and an efficient approximation method, like the one developed for the collision with obstacles, could be a good alternative to obtain better performances. Also, a more general distance evaluation function will be developed to reduce the approximation error due to bounding boxes. The integration of the second arm of the robot will be also investigated in order to consider bi-manipulation tasks. A comparison with some trajectory optimization approaches, such as [Zucker *et al.*, 2013] and [Kalakrishnan *et al.*, 2011], will be also investiagted.

Finally, our algorithm provides the safest path for a static environment as it is the result of a motion planning algorithm. However, this path should be modified online to deal with a dynamic environment. Real-time aspects will be a major issue in forthcoming work.

The initially optimal path should be deformed, due to changes in the environment, to respect the human safety throughout the whole task execution, physical aspects of the interaction will therefore be examined in depth to allow a complete and fully safe human-robot cooperation.

3.8 Appendix

Let us define δ_{exa}, δ_{est} the exact and estimated minimum distances, respectively, between two objects A_1 and A_2, where A_1 moves from configuration q_a to q_b and A_2 is a fixed obstacle. Let q_{worst} be the configuration, between q_a and q_b, where the distance is minimal.

From (3.9), we can define the error between the exact value and our estimation as :

$$\begin{aligned} \Delta &= \delta_{exa} - \delta_{est} \\ &= \eta_{12}(q_{worst}) - \frac{\eta_{12}(q_a) + \eta_{12}(q_b) - \lambda_1(q_a, q_b)}{2} \end{aligned} \tag{3.24}$$

The proof of Equation (3.10) can now be constructed as follows :

- $\Delta = 0$ **is the minimum error**

By definition, (3.9) gives an underestimation of the minimum clearance along a path segment [Schwarzer *et al.*, 2005]. That means $\delta_{exa} \geq \delta_{est}$ is always verified and thus $\Delta \geq 0$. The value $\Delta = 0$ can be obtained in at least one case : when $q_a = q_b$. That means that, in the best case, our method gives no error on the estimation.

- $\Delta = \frac{\epsilon}{2}$ **is the maximum error**

To get the maximum value of Δ, (3.24) can be rewritten as (3.25).

$$\Delta = \eta_{12}(q_{worst}) - \frac{\eta_{12}(q_a) + \eta_{12}(q_b)}{2} + \frac{\lambda_1(q_a, q_b)}{2} \tag{3.25}$$

We now have two independent parts in the expression of δ_{est}. As η_{12} and λ_1 are always positives, the maximum value of Δ is obtained when the two following conditions are fulfilled :

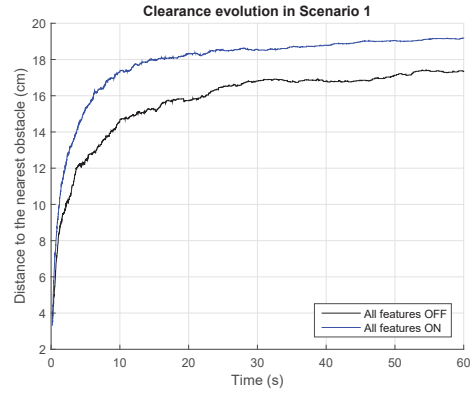
- $\eta_{12}(q_a) + \eta_{12}(q_b)$ is minimal ;
- $\lambda_1(q_a, q_b)$ is maximal.

Because the minimum clearance is obtained at configuration q_{worst}, $\eta_{12}(q_a) = \eta_{12}(q_b) = \eta_{12}(q_{worst})$ satisfies the first condition.

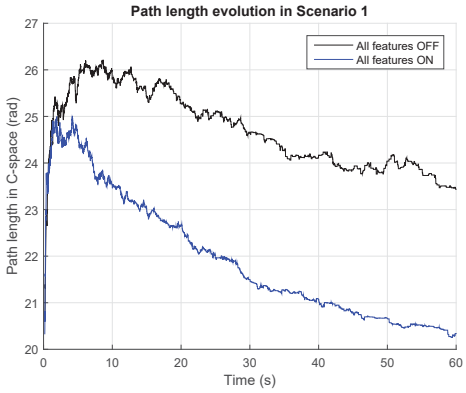
As defined in Algorithm 1, the stopping criterion of the procedure is $\lambda \leq \epsilon$. That means the maximum value that $\lambda_1(q_a, q_b)$ can take is ϵ.

When replacing these terms in (3.25), we get $\Delta = \frac{\epsilon}{2}$. This is the maximum error on the estimated clearance that is obtained in the worst case.

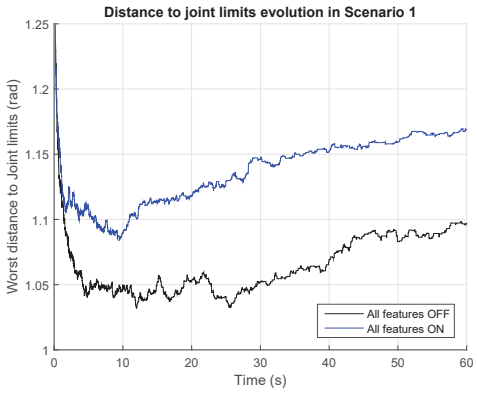
Hence, the condition $0 \leq \Delta \leq \frac{\epsilon}{2}$ is always satisfied.



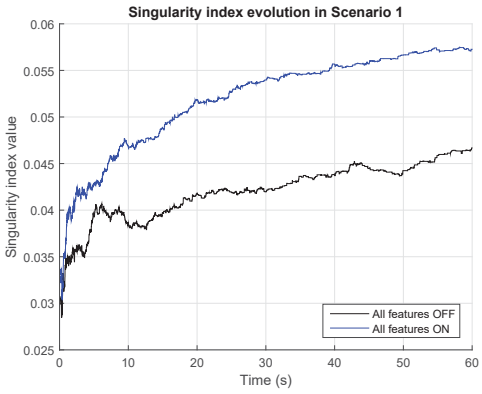
(a) Clearance



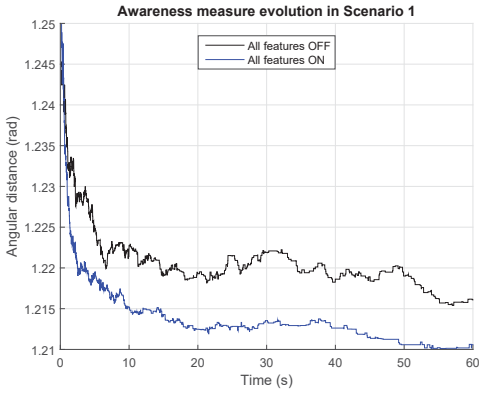
(b) Path length



(c) Joint limits

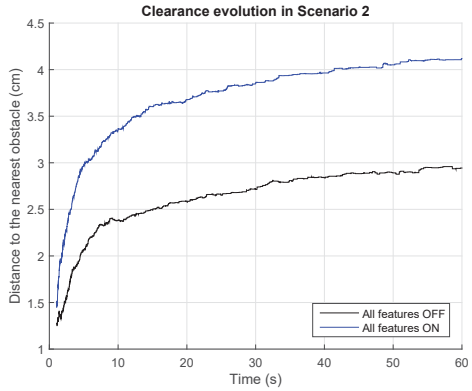


(d) Singularity

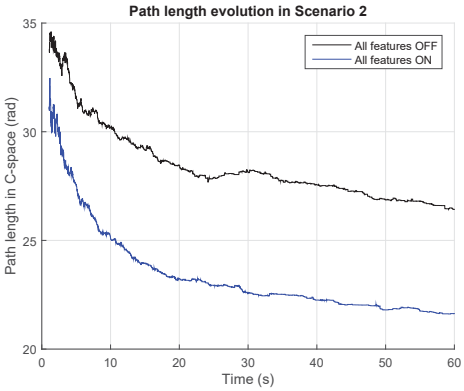


(e) Awareness

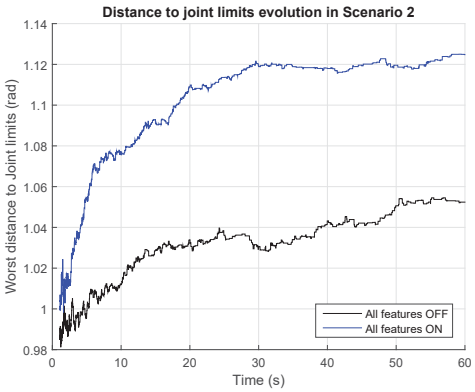
Figure 3.21 Comparative evolution of costs over time in Scenario 1.



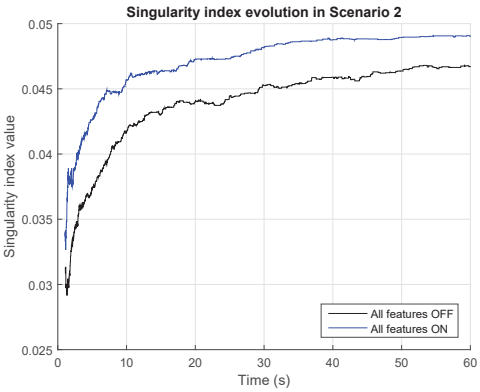
(a) Clearance



(b) Path length

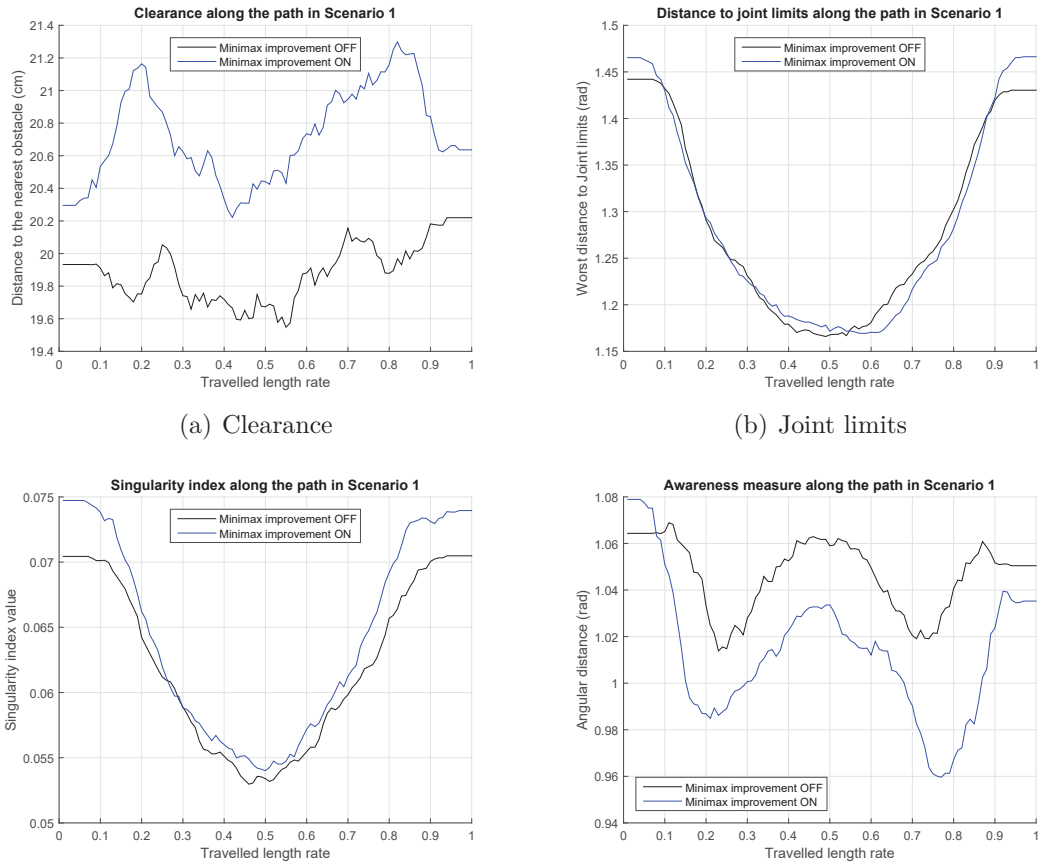


(c) Joint limits



(d) Singularity

Figure 3.22 Comparative evolution of costs over time in Scenario 2.



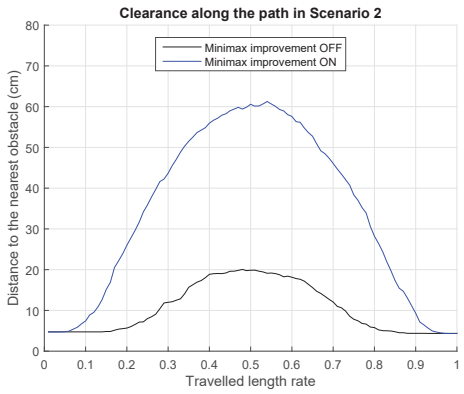
(a) Clearance

(b) Joint limits

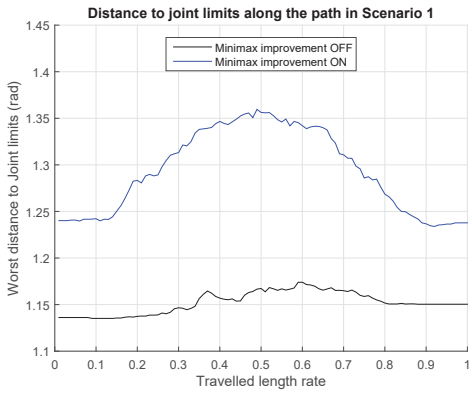
(c) Singularity

(d) Awareness

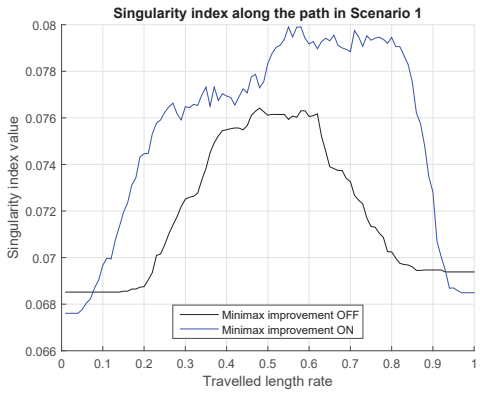
Figure 3.23 Profile of typical cost evolutions along the planned path in Scenario 1.



(a) Clearance

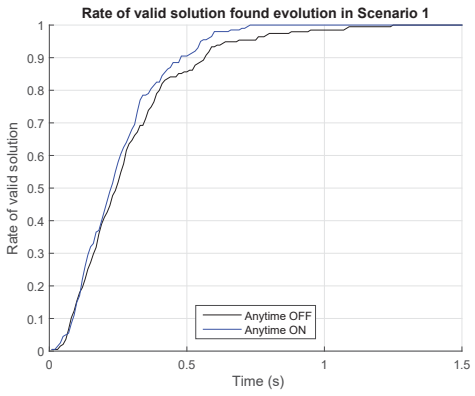


(b) Joint limits

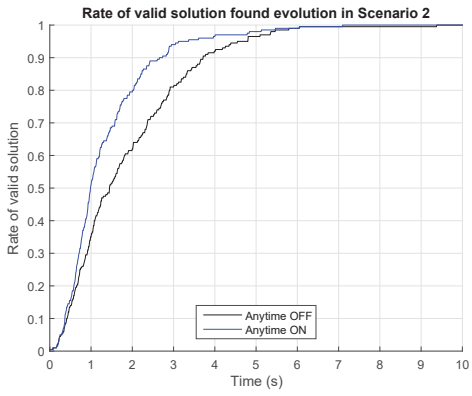


(c) Singularity

Figure 3.24 Profile of typical cost evolutions along the planned path in Scenario 2.



(a) Scenario 1



(b) Scenario 2

Figure 3.25 Rate of valid solution (number of motion planning tasks for which at least one collision-free path have been found) over time for a single thread.

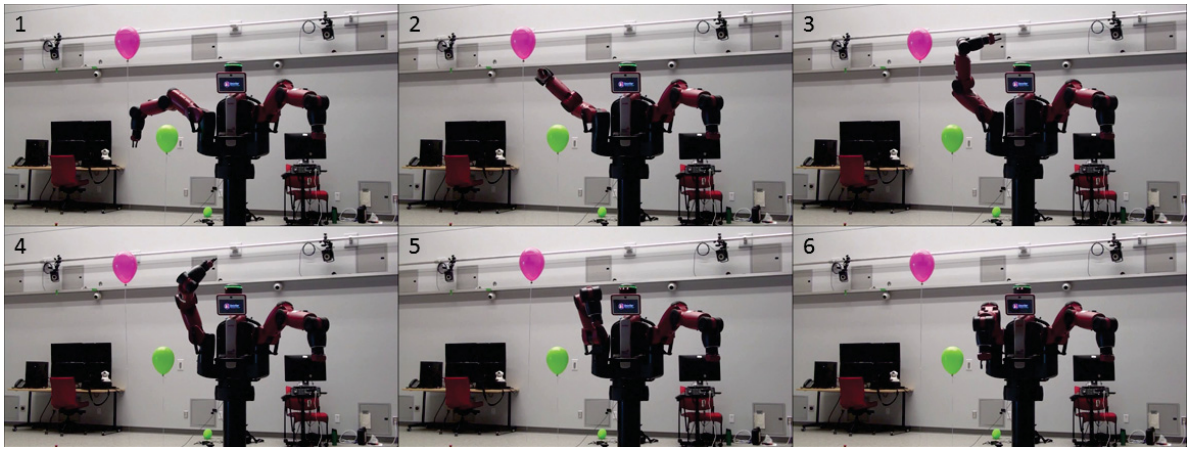


Figure 3.26 Snapshots of the Baxter robot executing a motion planning in an unstable environment.

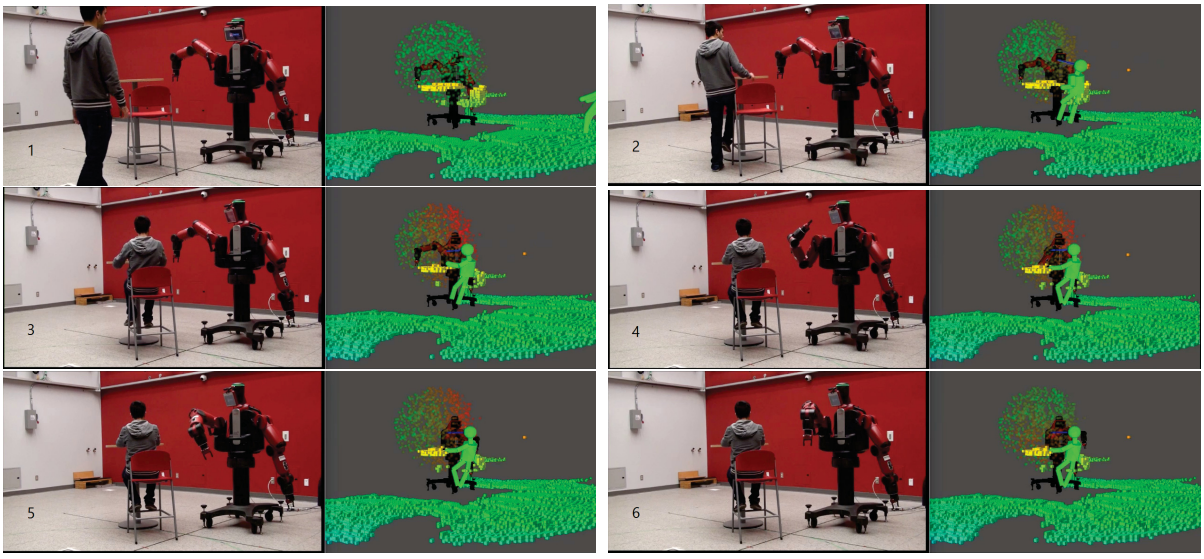


Figure 3.27 Snapshots of the Baxter robot executing a motion planning while avoiding a human.

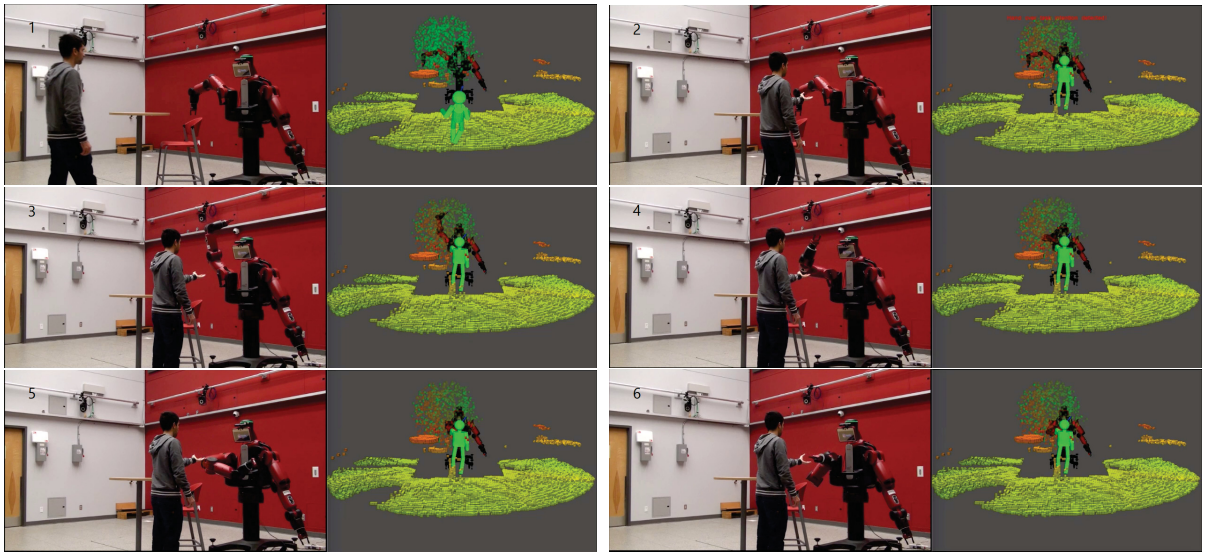


Figure 3.28 Snapshots of the Baxter robot detecting an interactive task intention from a human and executing the corresponding motion planning.

CHAPITRE 4

CONCLUSION

Ce projet s'inscrit dans la lignée des nombreuses avancées technologiques réalisées en matière de robotique ces dernières années. Le robot est en train de prendre une place très importante au sein de la société et de se rapprocher de plus en plus de l'humain. Il est capable d'apporter beaucoup à ce dernier sous condition que cela soit fait de manière sécurisée. Pour cela, cette maîtrise de recherche a pour but de minimiser le facteur de risque existant lors de l'interaction avec la machine.

C'est l'étape de planification de mouvements qui a été considéré dans le cadre de cette étude. En effet, parmi les différents moyens d'intervenir sur la sécurité, les stratégies pré-collision sont à privilégier car elles permettent de détecter et d'anticiper toute situation dangeureuse.

Plusieurs aspects de l'étape de détermination d'une trajectoire pour un robot manipulateur ont ainsi été modifiés. En premieu lieu, l'utilisation d'une méthode de détection de collision dynamique assure l'exactitude des tests effectuées. Ainsi, en considérant l'état initial de l'environement, il est prouvé qu'aucun contact involontaire n'interviendra tout au long du chemin fourni par l'algorithme. Ensuite, l'évaluation de paramètres pertinents liés à la sécurité et leur intégration au sein d'une stratégie d'optimisation multi-objectifs permettent de déterminer le chemin jugé le plus sécuritaire étant donné une certaine connaissance de l'environement. Cette dernière information joue un rôle majeur dans la qualité de la solution fournie par l'algorithme. De ce fait, le développement d'un algorithme RRT* bi-directionnel doté de nombreuses fonctionnalités a pour but de réduire la compléxité des calculs et donc de satisfaire les contraintes temporelles exigées par la plupart des applications industrielles.

Afin de produire un système suffisammenet sécuritaire, la méthode proposée se doit d'être complée à un module de modification en ligne de la trajectoire. En effet, dans un contexte où l'environement évolue dynamiquement, il est impératif de tenir compte de l'état de danger durant l'éxécution de la tâche et d'appliquer des actions correctives lorsque cela est nécessaire. Cet aspect constitue la suite logique du travail effectué dans le cadre de cette maîtrise. De plus, la prise en compte du contact physique entre les deux entités mises

en jeu est un axe d'étude novateur qui devra être étudié afin d'élargir le champ d'action ciblé par ce projet.

LISTE DES RÉFÉRENCES

Akgun, B. et Stilman, M. (2011). Sampling heuristics for optimal motion planning in high dimensions. Dans *IEEE/RSJ International Conference on Intelligent Robots and Systems*. p. 2640–2645.

ANSI/RIA (1999). *ANSI R15.06-1999 : American National Standard for Industrial Robots and Robot Systems-Safety Requirements* (Rapport technique). American National Standards Institute/Robotics Industry Association.

Asimov, I. (2004). *I, Robot*. Random House LLC.

Bekris, K. E. et Kavraki, L. E. (2007). Greedy but safe replanning under kinodynamic constraints. Dans *IEEE International Conference on Robotics and Automation*. p. 704–710.

Canny, J. (1988). *The Complexity of Robot Motion Planning*. MIT Press.

Cho, C.-N., Kim, J.-H., Kim, Y.-L., Song, J.-B. et Kyung, J.-H. (2012). Collision detection algorithm to distinguish between intended contact and unexpected collision. *Advanced Robotics*, volume 26, numéro 16, p. 1825–1840.

Cho, C. N., Kim, Y.-L. et Song, J.-B. (2013). Adaptation-and-collision detection scheme for safe physical human-robot interaction. Dans *IEEE 9th Asian Control Conference*. p. 1–6.

Choset, H. M. (2005). *Principles of Robot Motion : Theory, Algorithms, and Implementation*. MIT Press.

Corke, P. I. (1999). Safety of advanced robots in human environments. *Discussion Paper for International Advanced Robotics Program*.

De Luca, A., Albu-Schaffer, A., Haddadin, S. et Hirzinger, G. (2006). Collision detection and safe reaction with the DLR-III lightweight manipulator arm. Dans *IEEE/RSJ International Conference on Intelligent Robots and Systems*. p. 1623–1630.

De Luca, A. et Flacco, F. (2012). Integrated control for pHRI : Collision avoidance, detection, reaction and collaboration. Dans *4th IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics*. p. 288–295.

Dobson, A. et Bekris, K. E. (2013a). Improving sparse roadmap spanners. Dans *IEEE International Conference on Robotics and Automation*. p. 4106–4111.

Dobson, A. et Bekris, K. E. (2013b). A study on the finite-time near-optimality properties of sampling-based motion planners. Dans *IEEE/RSJ International Conference on Intelligent Robots and Systems*. p. 1236–1241.

Dobson, A., Krontiris, A. et Bekris, K. E. (2013). Sparse roadmap spanners. Dans *Algorithmic Foundations of Robotics X*. Springer, p. 279–296.

Dragan, A. et Srinivasa, S. (2013). Generating legible motion. Dans *Proceedings of Robotics : Science and Systems*.

European Clearing House for Open Robotics Development (ECHORD) (2013). *ECHORD plus plus*. `http://www.echord.eu/` (page consultée le 06 juin 2014).

Fauteux, P., Lauria, M., Heintz, B. et Michaud, F. (2010). Dual-differential rheological actuator for high-performance physical robotic interaction. *IEEE Transactions on Robotics*, volume 26, numéro 4, p. 607 –618.

Flacco, F., Kroger, T., De Luca, A. et Khatib, O. (2012). A depth space approach to human-robot collision avoidance. Dans *IEEE International Conference on Robotics and Automation*. p. 338–345.

Garcia, E., Jimenez, M. A., De Santos, P. G. et Armada, M. (2007). The evolution of robotics research. *IEEE Robotics & Automation Magazine*, volume 14, numéro 1, p. 90–103.

Giuliani, M., Lenz, C., Müller, T., Rickert, M. et Knoll, A. (2010). Design principles for safety in human-robot interaction. *International Journal of Social Robotics*, volume 2, numéro 3, p. 253–274.

Guiochet, J., Martin-Guillerez, D. et Powell, D. (2010). Experience with model-based user-centered risk assessment for service robots. Dans *IEEE 12th International Symposium on High-Assurance Systems Engineering*. p. 104–113.

Haddadin, S. (2013). *Towards Safe Robots : Approaching Asimov's 1st Law*. Springer Publishing Company, Incorporated.

Haddadin, S., Albu-Schaffer, A., De Luca, A. et Hirzinger, G. (2008). Collision detection and reaction : A contribution to safe physical human-robot interaction. Dans *IEEE/RSJ International Conference on Intelligent Robots and Systems*. p. 3356–3363.

Haddadin, S., Albu-Schäffer, A. et Hirzinger, G. (2007). Safety evaluation of physical human-robot interaction via crash-testing. Dans *Robotics : Science and Systems*. volume 3. p. 217–224.

Haddadin, S., Albu-Schäffer, A. et Hirzinger, G. (2009). Requirements for safe robots : Measurements, analysis and new insights. *The International Journal of Robotics Research*, volume 28, numéro 11-12, p. 1507–1527.

Ham, R. v., Sugar, T. G., Vanderborght, B., Hollander, K. W. et Lefeber, D. (2009). Compliant actuator designs. *IEEE Robotics & Automation Magazine*, volume 16, numéro 3, p. 81–94.

Hornung, A., Wurm, K. M., Bennewitz, M., Stachniss, C. et Burgard, W. (2013). OctoMap : An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, volume 34, numéro 3, p. 189–206.

Huber, M., Rickert, M., Knoll, A., Brandt, T. et Glasauer, S. (2008). Human-robot interaction in handing-over tasks. Dans *The 17th IEEE International Symposium on Robot and Human Interactive Communication*. p. 107–112.

Ikuta, K., Ishii, H. et Nokata, M. (2003). Safety evaluation method of design and control for human-care robots. *The International Journal of Robotics Research*, volume 22, numéro 5, p. 281–297.

Ioan A. Sucan and Sachin Chitta (2013). *MoveIt !* `http://moveit.ros.org` (page consultée le 24 février 2015).

Janson, L., Schmerling, E., Clark, A. et Pavone, M. (2015). Fast marching tree : A fast marching sampling-based method for optimal motion planning in many dimensions. *The International Journal of Robotics Research*, volume 34, numéro 7, p. 883–921.

Javier V Gomez (2014). *CForest Parallelization Framework*. `http://ompl.kavrakilab.org/CForest.html` (page consultée le 16 novembre 2015).

Jordan, M. et Perez, A. (2013). *Optimal Bidirectional Rapidly-Exploring Random Trees* (Rapport technique MIT-CSAIL-TR-2013-021). Computer Science and Artificial Intelligence Laboratory, Massachusetts Institute of Technology, Cambridge, MA, 12 p.

Kalakrishnan, M., Chitta, S., Theodorou, E., Pastor, P. et Schaal, S. (2011). Stomp : Stochastic trajectory optimization for motion planning. Dans *IEEE International Conference on Robotics and Automation*. p. 4569–4574.

Karaman, S. et Frazzoli, E. (2011). Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, volume 30, numéro 7, p. 846–894.

Kavraki, L. E., Kolountzakis, M. N. et Latombe, J.-C. (1998). Analysis of probabilistic roadmaps for path planning. *IEEE Transactions on Robotics and Automation*, volume 14, numéro 1, p. 166–171.

Kavraki, L. E., Švestka, P., Latombe, J.-C. et Overmars, M. H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, volume 12, numéro 4, p. 566–580.

Kay, D. (1988). *Tensor Calculus*. Schaum's Outlines, McGraw Hill (USA).

Koenig, N. et Howard, A. (2004). Design and use paradigms for Gazebo, an open-source multi-robot simulator. Dans *IEEE/RSJ International Conference on Intelligent Robots and Systems*. volume 3. p. 2149–2154.

Kuffner, J. J. et LaValle, S. M. (2000). RRT-connect : An efficient approach to single-query path planning. Dans *IEEE International Conference on Robotics and Automation*. volume 2. p. 995–1001.

Kulić, D. et Croft, E. (2007). Pre-collision safety strategies for human-robot interaction. *Autonomous Robots*, volume 22, numéro 2, p. 149–164.

Lacevic, B. et Rocco, P. (2010a). Kinetostatic danger field - a novel safety assessment for human-robot interaction. Dans *IEEE/RSJ International Conference on Intelligent Robots and Systems*. p. 2169–2174.

Lacevic, B. et Rocco, P. (2010b). Sampling-based safe path planning for robotic manipulators. Dans *IEEE Conference on Emerging Technologies and Factory Automation*. p. 1–7.

Lacevic, B. et Rocco, P. (2013). Safety-oriented path planning for articulated robots. *Robotica*, volume 31, numéro 06, p. 861–874.

Lacevic, B., Rocco, P. et Strandberg, M. (2011). Safe motion planning for articulated robots using RRTs. Dans *IEEE XXIII International Symposium on Information, Communication and Automation Technologies (ICAT)*. p. 1–7.

LaValle, S. M. (1998). *Rapidly-Exploring Random Trees : A New Tool for Path Planning* (Rapport technique). Computer Science Dept., Iowa State University.

LaValle, S. M. (2006). *Planning Algorithms*. Cambridge University Press.

LaValle, S. M. et Kuffner, J. J. (2001). Randomized kinodynamic planning. *The International Journal of Robotics Research*, volume 20, numéro 5, p. 378–400.

Lee, S.-D., Kim, B.-S. et Song, J.-B. (2013). Human–robot collision model with effective mass and manipulability for design of a spatial manipulator. *Advanced Robotics*, volume 27, numéro 3, p. 189–198.

Lindemann, S. R. et LaValle, S. M. (2005). Current issues in sampling-based motion planning. Dans *Robotics Research*. Springer, p. 36–54.

Loughlin, C., Albu-Schäffer, A., Haddadin, S., Ott, C., Stemmer, A., Wimböck, T. et Hirzinger, G. (2007). The DLR lightweight robot : Design and control concepts for robots in human environments. *Industrial Robot : An International Journal*, volume 34, numéro 5, p. 376–385.

Mainprice, J. et Berenson, D. (2013). Human-robot collaborative manipulation planning using early prediction of human motion. Dans *IEEE/RSJ International Conference on Intelligent Robots and Systems*. p. 299–306.

Najmaei, N. et Kermani, M. R. (2011). Applications of artificial intelligence in safe human–robot interactions. *IEEE Transactions onSystems, Man, and Cybernetics, Part B : Cybernetics*, volume 41, numéro 2, p. 448–459.

Najmaei, N., Lele, S., Kermani, M. et Sobot, R. (2010). Human factors for robot safety assessment. Dans *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*. p. 539–544.

Norouzzadeh, S., Lorenz, T. et Hirche, S. (2012). Towards safe physical human-robot interaction : An online optimal control scheme. Dans *IEEE International Symposium on Robot and Human Interactive Communication*. p. 503–508.

Otte, M. et Correll, N. (2013). C-forest : Parallel shortest path planning with superlinear speedup. *IEEE Transactions on Robotics*, volume 29, numéro 3, p. 798–806.

Pan, J., Chitta, S. et Manocha, D. (2012). FCL : A general purpose library for collision and proximity queries. Dans *IEEE International Conference on Robotics and Automation*. p. 3859–3866.

Parthasarathi, R. et Fraichard, T. (2007). An inevitable collision state-checker for a car-like vehicle. Dans *IEEE International Conference on Robotics and Automation*. p. 3068–3073.

Peleg, D. et Schäffer, A. A. (1989). Graph spanners. *Journal of Graph Theory*, volume 13, numéro 1, p. 99–116.

Pervez, A. et Ryu, J. (2008). Safe physical human robot interaction-past, present and future. *Journal of Mechanical Science and Technology*, volume 22, numéro 3, p. 469–483.

Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R. et Ng, A. Y. (2009). ROS : an open-source robot operating system. Dans *ICRA Workshop on Open Source Software*. volume 3. p. 5.

Quinlan, S. (1994). *Real-time Modification of Collision-Free Paths*. Thèse de doctorat, Department of Computer Science, Stanford University.

Salzman, O. et Halperin, D. (2014). Asymptotically near-optimal RRT for fast, high-quality, motion planning. Dans *IEEE International Conference on Robotics and Automation*. p. 4680–4685.

Salzman, O. et Halperin, D. (2015). Asymptotically-optimal motion planning using lower bounds on cost. Dans *IEEE International Conference on Robotics and Automation*. p. 4167–4172.

Sarkar, N. (2002). Psychophysiological control architecture for human-robot coordination-concepts and initial experiments. Dans *IEEE International Conference on Robotics and Automation*. volume 4. p. 3719–3724.

Schwarzer, F., Saha, M. et Latombe, J.-C. (2004). Exact collision checking of robot paths. Dans *Algorithmic Foundations of Robotics V*. Springer, p. 25–41.

Schwarzer, F., Saha, M. et Latombe, J.-C. (2005). Adaptive dynamic collision checking for single and multiple articulated robots in complex environments. *IEEE Transactions on Robotics*, volume 21, numéro 3, p. 338–353.

Sisbot, E. A. et Alami, R. (2012). A human-aware manipulation planner. *IEEE Transactions on Robotics*, volume 28, numéro 5, p. 1045–1057.

Şucan, I. A., Moll, M. et Kavraki, L. E. (2012). The Open Motion Planning Library. *IEEE Robotics & Automation Magazine*, volume 19, numéro 4, p. 72–82.

The German Aerospace Center (DLR) (2014). *Robotics and Mechatronics Center*. `http://www.dlr.de/rmc/rm/en/` (page consultée le 06 juin 2014).

Vahrenkamp, N., Asfour, T. et Dillmann, R. (2007). Efficient motion planning for humanoid robots using lazy collision checking and enlarged robot models. Dans *IEEE/RSJ International Conference on Intelligent Robots and Systems*. p. 3062–3067.

Vahrenkamp, N., Kaiser, P., Asfour, T. et Dillmann, R. (2011). RDT+ : A parameter-free algorithm for exact motion planning. Dans *IEEE International Conference on Robotics and Automation*. p. 715–722.

Versace, J. (1971). *A Review of the Severity Index* (Rapport technique). SAE Technical Paper.

Wang, H., Chen, W., Lei, Y. et Yu, S. (2007). Kinematic analysis and simulation of a 7-DOF cable-driven manipulator. Dans *IEEE International Conference on Control and Automation*. p. 642–647.

Yoshikawa, T. (1985). Manipulability of robotic mechanisms. *The International Journal of Robotics Research*, volume 4, numéro 2, p. 3–9.

Zanchettin, A. M. et Lacevic, B. (2012). Sensor-based trajectory generation for safe human-robot cooperation. Dans *IEEE/RSJ International Conference on Intelligent Robots and Systems, Workshop Motion Planning Online, Reactive, Real-Time, Algarve, Portugal*.

Zucker, M., Ratliff, N., Dragan, A., Pivtoraiko, M., Klingensmith, M., Dellin, C., Bagnell, J. A. D. et Srinivasa, S. (2013). Chomp : Covariant hamiltonian optimization for motion planning. *International Journal of Robotics Research*.