Planification d'une tâche de transfert d'objet entre un robot et un humain

Mamoun Gharbi

▶ **To cite this version:**

Mamoun Gharbi. Planification d'une tâche de transfert d'objet entre un robot et un humain. Robotique [cs.RO]. 2011. dumas-00636405

HAL Id: dumas-00636405

https://dumas.ccsd.cnrs.fr/dumas-00636405v1

Submitted on 27 Oct 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

LAAS-CNRS

enib
ÉCOLE NATIONALE
D'INGÉNIEURS DE BREST

Stage de recherche
2011

Planification d'une tâche de transfert d'objet entre un robot et un humain

Mamoun Gharbi

Laboratoire d'Analyse et d'Architecture des Systèmes (LAAS)
7, avenue du Colonel Roche
31077 Toulouse Cedex

Directeur de recherche : M. Rachid ALAMI

Remerciements

Je tiens tout d'abord à remercier M. Rachid ALAMI de m'avoir permis de réaliser ce stage.

Je le remercie aussi ainsi que M. Jim MAINPRICE et tous les membres de l'équipe RIA pour leur disponibilité, leurs conseils et leur sympathie.

Merci enfin à tous les doctorants et stagiaires avec qui j'ai apprécié de travailler et au laboratoire du LAAS-CNRS pour m'avoir accueilli.

Avant propos

Ce mémoire de stage présente la problématique traitée ainsi qu'une solution d'implémentation possible. Cependant, l'implémentation finale de l'algorithme n'est pas encore finie, de fait, la partie analyse des résultats ne sera pas complète. La version finale de ce mémoire comportera, bien sûr, les résultats obtenus ainsi que leur analyse.

Table des matières

1 Introduction

Ces dernières années ont connu une avancé notoire dans le domaine de la robotique, tant au niveau matériel que logiciel permettant ainsi au robot de s'approcher et d'interagir de plus en plus finement avec les humains. Les robots qui étaient jusque là cantonnés aux usines et séparés des humains vont bientôt se retrouver dans nos maisons où ils travailleront, nous aideront et interagiront avec nous. Ce changement de cadre va amener de nouveaux chalenges dans la recherche robotique où la notion de sécurité va prendre un sens nouveau.

Effectivement, dans les usines (fig 1.1 (A)) les robots sont physiquement séparés des humains ce qui assure la sécurité de ces derniers. Cependant cette séparation ne peut pas être assurée pour les utilisations prochaines des robots où ceux-ci se retrouveront dans des situations où il devront assister, coopérer ou même partager une tâche avec l'humain. Dans un scénario ou le robot devra travailler parmi les humains (fig 1.1 (B)), la notion de sécurité devient plus importante et doit être étudiée en détails.

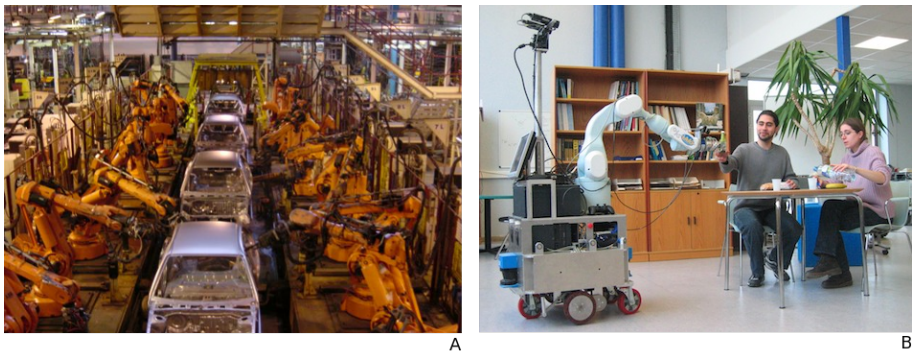

FIGURE 1.1 – (A) un ensemble de robot d'usine travaillant à l'assemblage d'une voiture. (B) un robot en situation domestique. figure de [Sisbot 10]

Ceci dit la grande différence entre ces deux environnements ne vient pas de la définition de leur préoccupation principale, la sécurité physique mais de leur préoccupation secondaire. Dans une usine, une fois la sécurité des humains assurée, la faisabilité de la tâche devient l'élément clé de la structure. Les robots sont synchronisés et parfaitement coordonnés afin de réaliser leur tâche.

A l'inverse, "l'acceptabilité social" de l'action à effectuer peut devenir plus important que la faisabilité de la tâche. Pour un robot qui interagit physiquement avec un humain, accomplir une tâche au

détriment du "confort de l'humain" n'est pas acceptable, même si le robot ne cause aucun dommage à l'humain. Il est préférable que le robot s'abstienne de réaliser une tâche si le seul moyen de la réaliser pourrait créer de la peur, de la surprise ou un inconfort quelconque pour l'humain [Sisbot 10].

Afin d'assurer la sécurité de l'humain au niveau physique, il est important de prendre en compte l'environnement où va évoluer le robot, et ceci avant et pendant sa conception, tant matériel que logiciel. Pendant la conception matériel, les points suivant doivent être abordés [Zinn 04] :

Le design du robot : Un design "human friendly" est nécessaire, autant pour éviter les petits accrochages (en enlevant les points de pincement et les arrêtes vives) que pour assurer un certains confort chez les humains.

Les matériaux du robot : Les matériaux qui constituent le robot doivent être léger afin de diminuer l'inertie de celui-ci (arrêt plus rapide en cas de problème) et le matériel qui le recouvre doit pouvoir absorber les chocs (recouvrement déformable).

Les actionneurs : Les actionneurs utilisés en usine sont rigides et risquent de créer des problèmes lors d'une utilisation domestique. Des actionneurs moins rigides permettent d'éviter certains accidents.

La reconnaissance de l'environnement : Le robot évoluant dans un environnement inconnu ou peu connu, doit pouvoir "voir" celui-ci à travers un certain nombre de capteurs.

Le bras KUKA [Hirzinger 02] est un bon exemple qui respecte les différentes propriétés citées ci-dessus fig 1.2.

FIGURE 1.2 – Le bras KUKA Light Weight Robot 4 ayant un design "human-friendly" [Hirzinger 02]

Pendant la conception logiciel d'autre éléments entre en considération : Les capacités cognitive du robot doivent être conçus de manière à respecter certaines contrainte, tel que la sécurité lors des déplacements, la réalisation de mouvements fiables et efficace respectant les normes social de l'humain. Autrement dit, il est claire que la présence de l'humain doit être prise en compte explicitement à tous les niveau de la conception logiciel. Du plus haut niveau jusqu'aux plus bas, le logiciel du robot doit prendre en compte la présence de l'humain, non seulement en tant qu'obstacle mobile mais surtout en tant qu'entité à part entière avec : une structure cinématique, un champ de vision, différentes préférences, états et postures.

Dans le groupe de travail où mon stage s'est déroulé, et afin de satisfaire les obligations citées ci-dessus, la structure de la figure 1.3 est utilisée.

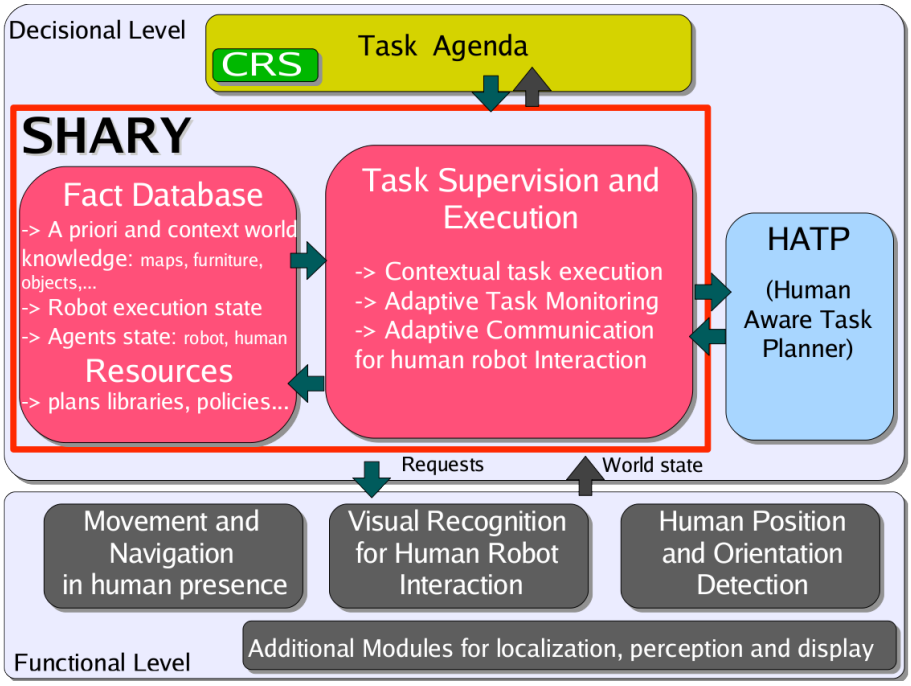

FIGURE 1.3 – Structure conceptuelle des logiciels utilisés au cours du stage [Clodic 09]

La structure se divise en 2 partie :

La partie décisionnelle : qui elle même est composée de trois sous parties : le task agenda, SHARY et HATP.

La partie fonctionnelle : qui est divisée en deux sous parties : le planificateur 3D et la couche bas niveaux.

Voici la description de chacune de ces parties :

Task agenda : S'occupe de toutes les activités décisionnelles reliées aux objectifs de haut niveau du robot.

SHARY : (Supervision for Human Aware Robot Ynteraction) constitue le noyaux décisionnel, basée sur un raffinement incrémental de tâches basée sur le contexte de l'humain. [Alili 09]

HATP : (Human Aware Task Planner) Un système de planification de tâches capable de synthétiser des plans du robot socialement acceptable qui peuvent impliquer des actions human-robot coordonnées.

Le planificateur 3D : Outre les calculs nécessaires à la réalisation d'une action, ce planificateur s'occupe de la gestion de l'espace : L'évitement des obstacles et le confort de l'humain par exemple [Gharbi 08].

La couche bas niveau : La partie qui s'occupe de l'asservissement des actionneurs et de l'acquisition des données à travers les capteurs [Broquere 08].

Lors de mon stage je me suis intéressé au planificateur 3D, ceci en me concentrant sur un exemple spécifique : l'échange de petits objets entre humain et robot dans un milieu moyennement encombré.

Dans notre vie quotidienne, nous réalisons des échanges d'objets avec nos semblables un nombre incalculable de fois et ceci de manière complètement intuitive et inconsciente. Afin d'intégrer un robot dans cet univers domestique, il doit être capable, entre autre, de réaliser cette tâche simple, mais qui soulève néanmoins nombres de questions : Où doit se dérouler l'échange ? Comment y accéder ? Quand doit-il avoir lieu ? Que faire de cet objet si l'humain n'en veut plus ?

Nous allons tenter de répondre à travers cette étude à quelques-unes de ces questions.

Ce problème, qui est en fait un problème décisionnel, a déjà été traité dans quelques études tel que [Mainprice 10] et [Sisbot 08] où le confort de l'humain est mis en avant à l'aide de grille de coût permettant de trouver des emplacements d"échange confortable relatif à l'humain. Une partie de ces études (le calcul des coûts de confort de l'humain) était dors et déjà implémentée sur le logiciel "move3d" [Simeon 01] qui gère le planificateur 3D cité précédemment.

Le contenu de mon stage se divise en trois volets : Le premier qui consiste à formaliser le problème (section 2), le second qui consiste à proposer et implémenter une solution (section 3) tout en utilisant des études utilisateur (disponible en annexe)

2 Problématique

Afin de résoudre le problème nous allons le formaliser sous forme d'une recherche de placement dans un espace de configurations.

Le but de cette section est de répondre aux questions suivantes : qu'est ce qu'un espace de configurations ? quel est l'espace que nous allons étudier et comment le déterminer ? sur quel critères effectuer la recherche dans cet espace ? et enfin, quelles sont les données en entrée et quelles sont celles attendues en sortie ?

Un espace de configurations est l'ensemble des configurations possibles d'un système mécanique. Une configuration est un vecteur contenant l'ensemble des valeurs de chaque degrés de liberté d'un système mécanique.

Afin de déterminer l'espace étudié, nous devons au préalable prendre connaissance de certains espaces indispensables pour le définir :

C_{Hum} **:** L'espace de configurations de l'homme.

C_{Rob} **:** L'espace de configurations du robot.

C **:** Le produit cartésien de l'espace des configurations de l'homme et celui du robot :
$C = C_{Hum} \times C_{Rob}$

C_{col} **:** Cet espace est inclus dans C, il représente les configurations où l'homme et le robot ne sont pas en collision (ni entre eux, ni avec l'environnement)

C_{apt} **:** Cet espace est inclus dans C, il représente les configurations où l'homme et le robot peuvent tous deux accéder à un point de l'espace avec un de leur préhenseur. En d'autres termes, pour toute configuration de C_{apt}, les deux protagonistes peuvent saisir avec leur préhenseur un objet de petite taille (un témoin par exemple)

C_{ecc} **:** Cet espace est inclus dans C. Quelque soit la configuration dans C_{ecc}, il existe une trajectoire pour le robot et une trajectoire pour l'humain qui leurs permettent d'y accéder. Un exemple est donné dans la fig 2.1

C_{stb} **:** Cet espace est inclus dans C, il représente les configurations où le robot et l'humain sont dans des configurations stables. Pour le robot une position stable est une position d'équilibre stable. Pour l'humain une position stable est une position où il est, soit assis, soit debout, les deux pieds au sol.

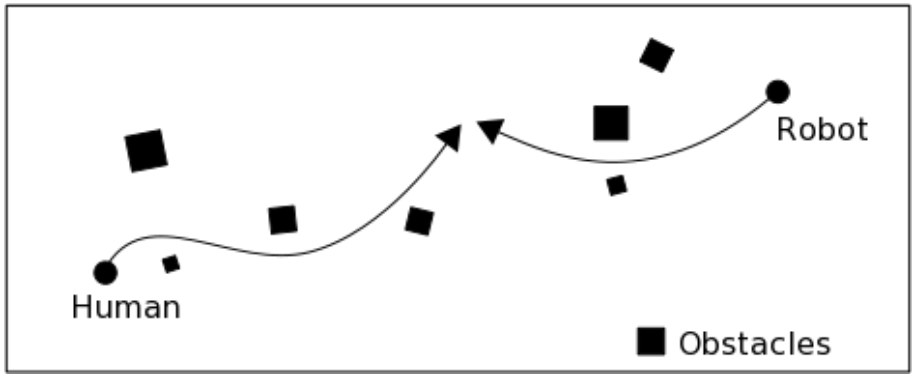

FIGURE 2.1 – L'espace C_{ecc}, inclus dans C, correspond à l'espace où il existe un chemin sans collision entre les positions initiales du robot et de l'humain et leurs positions finales.

L'espace de configurations que nous allons étudier est l'espace des configurations C_e qui est inclus dans C et qui se trouve dans l'intersection des espaces de configurations C_{col}, C_{apt}, C_{ecc} et C_{stb}. En d'autres termes : $C_e = \{C_e \subset C | C_e = C_{col} \cap C_{apt} \cap C_{ecc} \cap C_{stb}\}$ (fig 2.2)

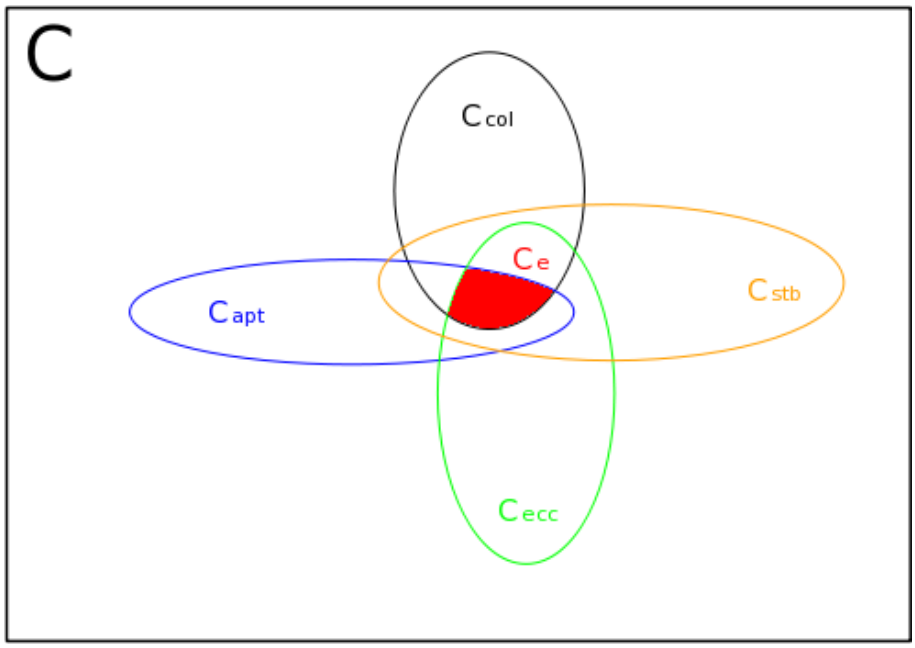

FIGURE 2.2 – L'espace C_e, inclus dans C, est l'intersection des espaces de configurations C_{col}, C_{apt}, C_{ecc} et C_{stb}. Cette figure est schématique : les espaces représentés ne sont pas forcément connexes.

Pour effectuer une recherche dans cet espace des configurations, nous allons nous baser sur deux principaux critères :

Le confort de l'humain : Ou la tension musculo-squelettique. Ce paramètre quantifie la qualité de la configuration de l'humain d'un point de vue anatomique ; plus la configuration est confortable pour l'humain plus elle aura de chance d'être retenue [Mainprice 10], [Sisbot 07a].

La participation de l'homme à la tâche : Ce paramètre quantifie la qualité de la trajectoire à parcourir par l'humain et le robot. Selon son besoin de l'objet (*objectNecessity*) que le robot va lui donner, l'humain s'impliquera plus où moins dans la tâche. Effectivement, si l'humain est occupé, où que son besoin n'est pas urgent, le robot peut se permettre de choisir une trajectoire où l'humain devra se déplacer le moins possible. A l'inverse, si le besoin est urgent, le robot va chercher une trajectoire qui sera plus contraignante pour l'humain (il devra peut-être se déplacer un peu plus) afin de réaliser l'échange au plus vite (fig. 2.3). D'autres paramètres doivent aussi être pris en compte lors du choix de la trajectoire : Si l'humain est debout, ces déplacements seront plus faciles que si il est assis. La position initiale du robot peut aussi influencer le choix de sa trajectoire [Marin-Urias 09], le robot préférera des trajectoires minimalistes, à des trajectoires trop longues, même si cela implique un peu plus d'effort de la part de l'humain (fig 2.4).

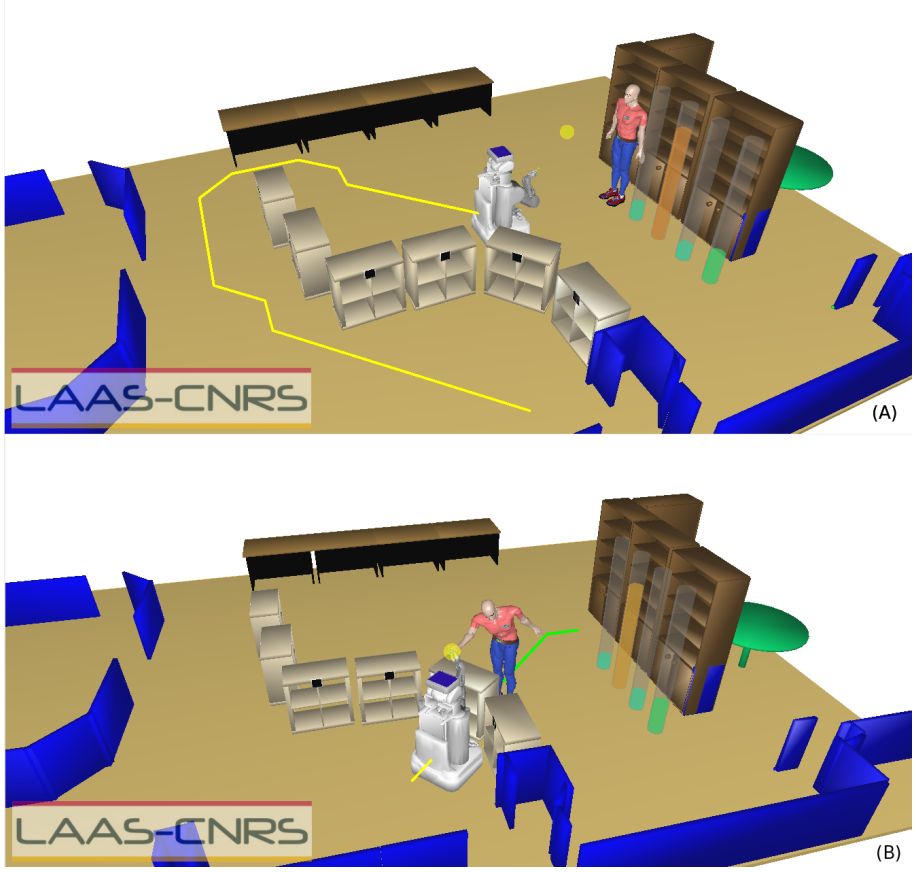

FIGURE 2.3 – (A) représente une trajectoire où on ne tient pas compte du temps d'attente de l'humain et (B) une trajectoire où le temps global de l'action est minimal. Figure réalisée grâce à move3d [Simeon 01]

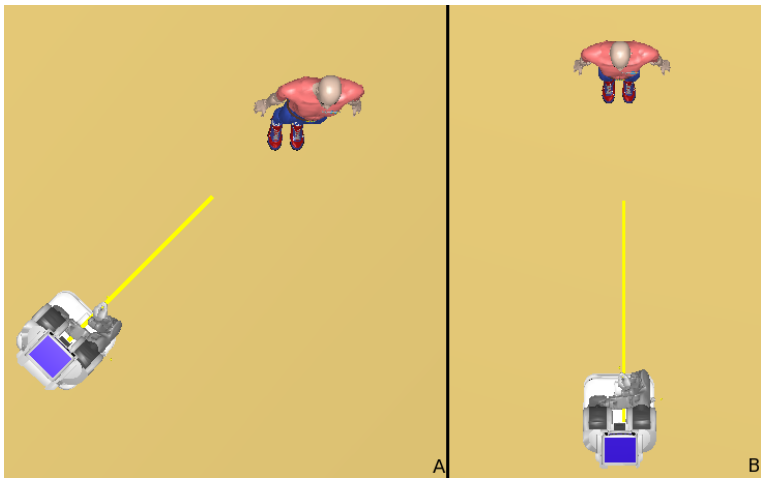

FIGURE 2.4 – Un exemple de l'influence de la position initiale du robot. En jaune, la trajectoire que le robot va suivre afin d'atteindre la configuration d'échange. Figure réalisée grâce à move3d [Simeon 01]

Dans le but de réaliser la tâche, nous avons les données suivantes en entrée :

$$inputs = \begin{cases} Q_{hum} \\ humanInitPos \\ robotInitPos \\ isStanding \\ objectNecessity \end{cases}$$

Q_{hum} et Q_{rob} sont, respectivement, les configurations initiales de l'humain et du robot. $humanInitPos$ et $robotInitPos$ représentent respectivement les positions initiales de l'humain et du robot dans l'espace 3D : leurs positions (en x et y) et leurs rotations (en R_z). $isStanding$ est un boolean qui détermine si l'humain est assis ou debout (vrai si il est debout). $objectNecessity$ varie entre 0 et 1, et représente son degré de besoin de l'objet.

Nous attendrons en sortie les données suivantes :

$$outputs = \begin{cases} Q_{hGoal} \\ humanGoalPos \\ humanTrajectory \\ Q_{rGoal} \\ robotGoalPos \\ robotTrajectory \end{cases}$$

Q_{hGoal} et Q_{rGoal} sont, respectivement, les configurations d'échange de l'humain et du robot. $humanGoalPos$ et $robotGoalPos$ sont leurs positions en x, y et R_z pour cette même configuration d'échange. $humanTrajectory$ et $robotTrajectory$ sont les trajectoires (au départ de la position initiale) qui correspondent à ces positions.

Plusieurs études ont été menées sur l'échange d'objets entre humain, [Cakmak 11], par exemple, pose le problème de la continuité durant l'échange, cela signifie que, dans l'idéal, ni le robot, ni l'humain ne doivent attendre pour recevoir/donner l'objet. [Kajikawa 95] quant à elle, se base sur l'analyse des trajectoires et des vitesses d'un échange d'objet entre humain pour le reproduire sur le robot afin d'avoir un comportant qui ressemble à celui de l'humain (human-like). La nouveauté apportée par ce travail est la planification de trajectoires pour l'humain. Durant un échange d'objet entre humain, il est rare qu'un seul des deux protagonistes fasse tout le travail or cette étude essaye de déterminer dans quelle mesure l'humain participera à la tâche afin de réaliser un mouvement coopératif homme-robot.

3 Solution proposé

La solution la plus simple consisterait à parcourir l'espace des configurations C, puis de choisir la meilleure configuration appartenant à C_e possible. Cette solution n'est pas utilisable car la nature de l'action nous impose des calculs en temps réel, or le nombre important d'éléments de C ne permettrait pas un parcours exhaustif. Cette section présente une solution possible au problème. (Rappel : C est un espace de 68 éléments continue)

Quelle que soit la posture initiale de l'humain, le planificateur (et donc le robot) doit lui proposer une configuration propre au transfert d'objet, tout en lui évitant les postures trop inconfortables (fig 3.1), ceci même si il se voit obliger d'effectuer un déplacement supplémentaire. Ce déplacement peut être plus ou moins grand selon l'urgence à récupérer l'objet que le robot va lui donner. La variable $objectNecessity$ permettra de déterminer l'amplitude de ce mouvement. Afin de traiter ce problème, nous allons suivre les étapes suivantes :

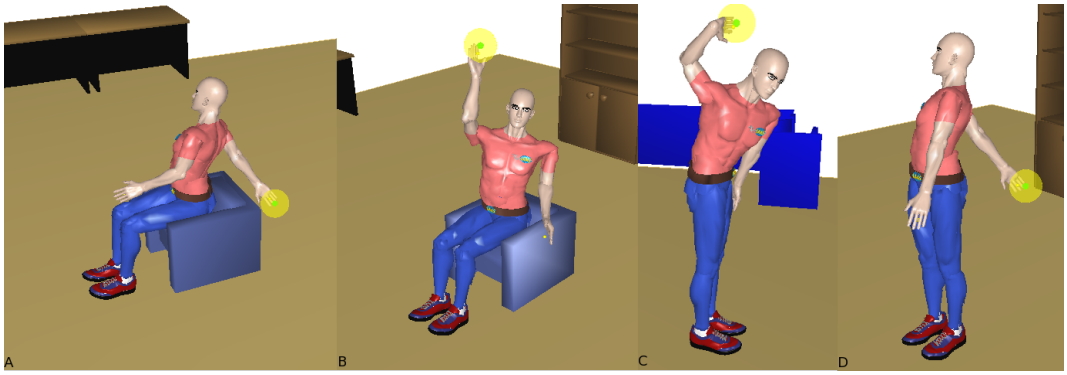

FIGURE 3.1 – Configurations inconfortables de l'humain. Figure réalisée grâce à move3d [Simeon 01]

- Discrétiser l'espace des configurations afin de ne garder que celles qui sont pertinentes (qui n'impliquent pas de mouvements trop inconfortables), sans prendre en compte l'environnement de l'humain.
- Donner un coût peu élevé aux petits mouvements telle que la rotation ou les déplacements locaux si l'humain est debout. A noter que la position assise augmente fortement le coût en ce qui concerne les mouvements.
- Chercher la solution faisable dont le coût est minimal.

3.1 Discrétisation de l'espace des configurations

Pour ne pas prendre en compte l'environnement, nous allons ignorer les déplacements en x et y ainsi que la rotation en R_z. De ce fait, les configurations retenues lors de la discrétisation seront liées à ces trois paramètres et pourront être déplacées en fonction de l'humain. En d'autres termes, pour chaque combinaison (x, y, R_z), un ensemble de configurations ϵ C_{conf} homme-robot sera disponible.

L'espace de configuration C_{conf} représente une discrétisation des configurations d'échange possibles entre le robot et l'humain. De ce fait, cet espace est fortement lié au robot utilisé : lors de ce stage, le robot mis en situation est le PR2 (http://www.willowgarage.com/pages/pr2/overview, fig 3.2). Afin de trouver les éléments de cet espace, nous allons choisir un ensemble de points, centrés sur l'humain, qui représenteront les endroits (emplacements 3D) où se dérouleront les échanges. A chaque point correspondra une configuration (calculée par cinématique inverse) qui donnera une idée approximative de la configuration que peut prendre l'humain pour effectuer l'échange à cet endroit. Cette approximation nous permet de déterminer une configuration du robot qui correspond à chaque point.

FIGURE 3.2 – Le robot PR2.

Cette discrétisation nous donne un début de solution à la problématique. Effectivement, l'espace C_{conf} est le résultat de l'intersection des espaces C_{apt} et C_{stb} (section 2) respectivement l'espace des configurations accessibles et l'espace des configurations stables. De plus elle nous permet de résoudre indirectement deux problématiques :

La sécurité : Les points doivent se trouver à une distance suffisante de l'humain pour assurer sa sécurité.

La visibilité : Les points doivent se trouver dans le champs de vision de l'humain afin qu'il puisse comprendre les intentions du robot.

Ainsi, pour notre étude, un certain nombre de points ont été retenus, la fig 3.3 permet de les situer dans l'espace. Le choix des différentes distances et angles a été fait à travers le simulateur de BioMove3d, se basant sur un humain de 1m80, droitier. Ces valeurs sont, bien entendu, à adapter selon les caractéristiques de l'humain avec lequel l'interaction doit être réalisée. De plus, deux études utilisateur portant sur le placement du robot par rapport à l'humain et sur les différentes configurations que celui-ci peut prendre seront réalisées afin de déterminer la pertinence de nos choix. (plus de détails en annexe.)

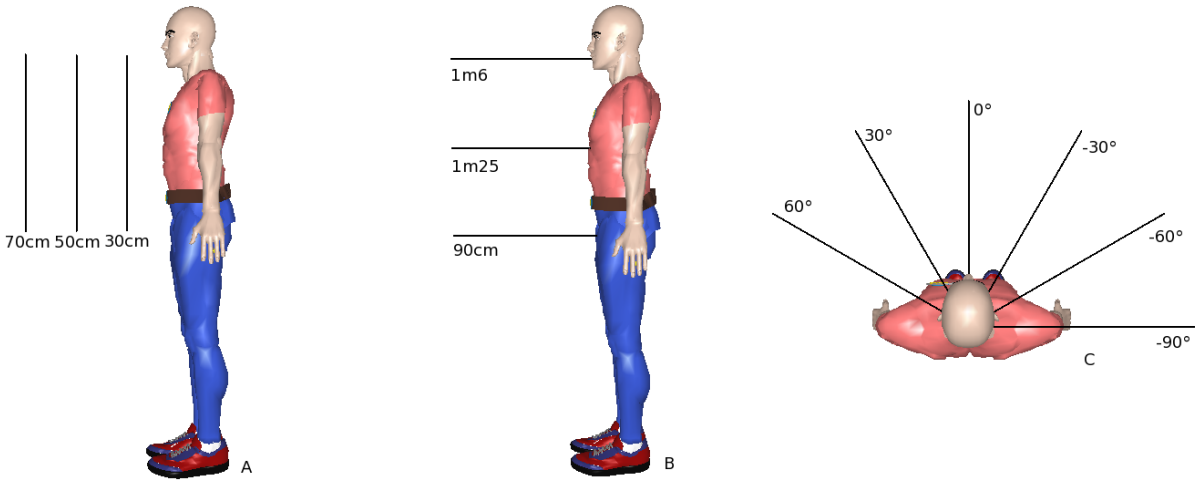

FIGURE 3.3 – (A) représente la discrétisation en profondeur ; les distances y sont données par rapport au centre de l'humain. (B) représente la discrétisation en hauteur, les distances y sont données par rapport au sol, et (C) la discrétisation angulaire. Le résultat du croisement de (A) et (B) ajouté à leur rotation selon (C) donne les points choisis. Figure réalisée grâce à move3d [Simeon 01]

Ainsi, la fig 3.4 montre les configurations du robot PR2 correspondantes à chacun des points calculés précédemment.

Cette discrétisation est indépendante de l'environnement, or l'environnement où évolue l'humain et le robot est encombré par un certain nombre d'obstacles. Avant de pouvoir quantifier la qualité des configurations, nous devons d'abord nous assurer de leur faisabilité.

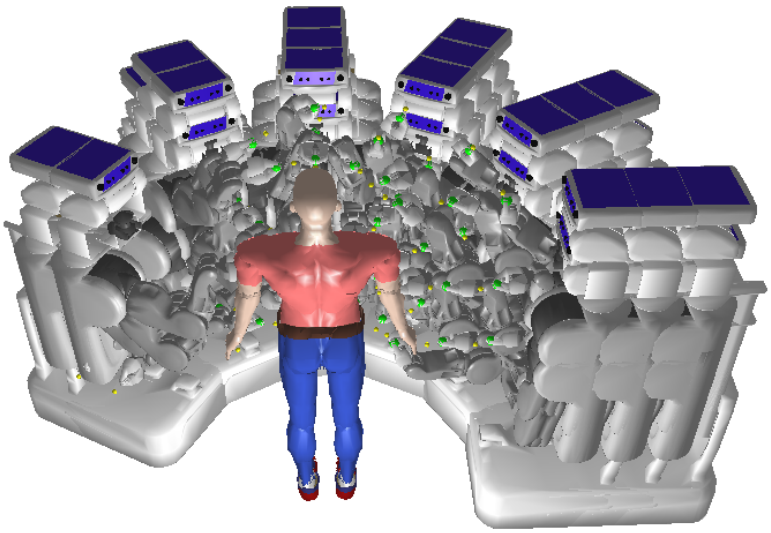

FIGURE 3.4 – Configurations du robot PR2 pour un ensemble de points discrétisés centré sur l'humain. Figure réalisée grâce à move3d [Simeon 01]

3.2 Faisabilité des configurations

L'espace C_e est égal à l'intersection de quatre espaces (section 2), notamment les espaces C_{col} et C_{ecc}, respectivement, l'espace libre de collision, et l'espace où il existe des chemins vers les configurations. Afin de définir ces deux espaces nous allons utiliser les techniques suivantes :

3.2.1 Détection de collision

Afin de déterminer l'espace de configurations C_{col}, nous allons utiliser une technique basée sur les *ghost* [Gregory 05]. Un *ghost* est un ensemble de boites englobantes, chacune d'entre elle est alignée sur l'axe principal de la partie de l'objet qu'elle représente. Pour détecter les collisions, un calcul de distance euclidienne est fait : la distance euclidienne entre deux *ghost* est la distance qui sépare les deux boites englobantes les plus proches (chacune appartenant à un *ghost* différent)fig. 3.5. La polarité de cette distance détermine l'existence ou non de collision dans la configuration analysée.

Afin de détecter les collisions, nous avons besoins en entrée seulement de la configuration à tester.

3.2.2 Existence d'un chemin vers les configurations

Le dernier espace dont l'identification est nécessaire afin de déterminer C_e est l'espace C_{ecc}. pour toute configuration dans C_{ecc}, il existe deux trajectoires : la première qui relie la position initiale de l'humain et sa position dans la configuration qui appartient à C_{ecc} et la deuxième qui relie la position initiale du robot et sa position dans cette même configuration.

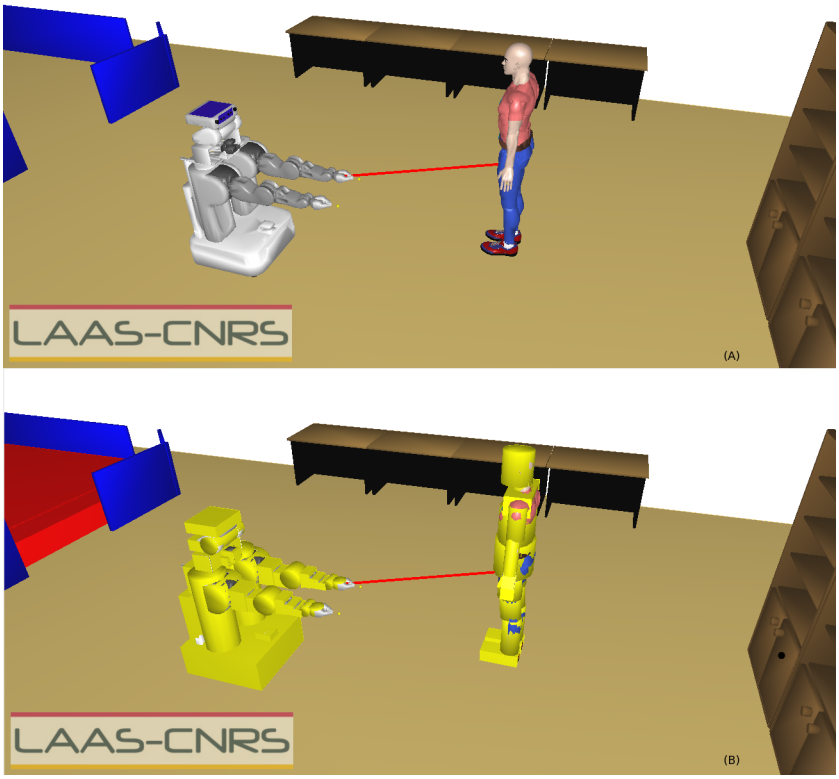

FIGURE 3.5 – La distance entre le robot et l'humain. En (A) la scène est représentée sans les *ghost*, et avec en (B). Figure réalisée grâce à move3d [Simeon 01]

La grille de coût est rudimentaire (1 si il y a collision, 0 sinon). Quelques améliorations peuvent être apportées à cette grille :

L'encombrement du robot : Dépendant de l'encombrement du robot et de sa rotation, certains endroits où il pouvait se rendre sans collision peuvent devenir inaccessibles, et inversement certains endroits jusque là hors d'atteinte peuvent devenir accessibles. [Marin-Urias 09]

Obstruction du champs de vision : Passer derrière des obstacles et donc sortir du champs de vision de l'humain peut provoquer un sentiment d'anxiété, voir de peur chez l'humain. De la même manière, passer derrière son dos peut provoquer la même chose. [Sisbot 07b]

La recherche de chemin prend en entrée la position initiale de l'humain ainsi que celle du robot.

Les quatre espaces de configurations qui forment (par leur intersection) C_e sont maintenant déterminés. La recherche des meilleures configurations dans cet espace devra prendre en compte non seulement le confort de l'humain, mais aussi son urgence d'obtenir l'objet.

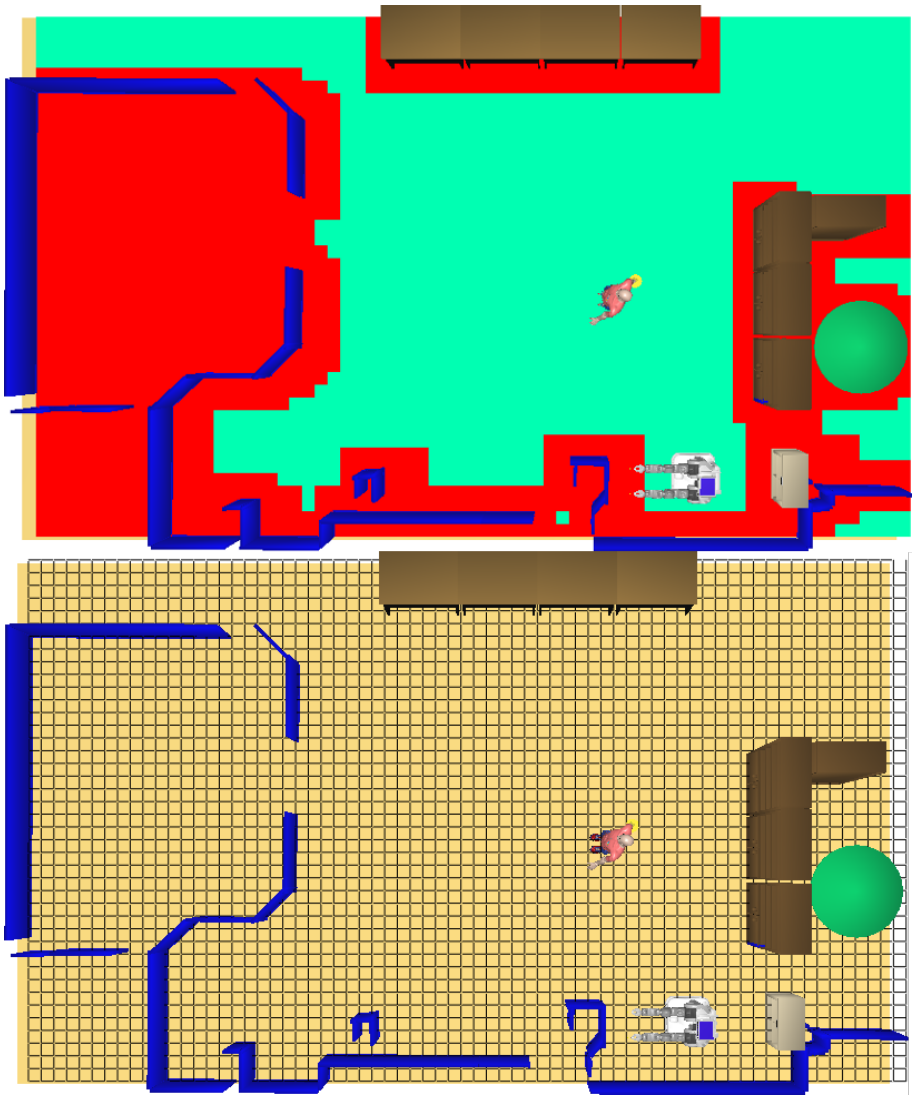

FIGURE 3.6 – En haut, la grille des coûts du graph 2D (représenté en bas) que l'algorithme A* parcourra. Figure réalisée grâce à move3d [Simeon 01]

3.3 Qualification des configurations

Afin d'effectuer la recherche dans l'espace des configurations C_e, nous devons d'abord pouvoir qualifier autant la configuration en elle même que la trajectoire qui y est associée.

3.3.1 Tension musculo-squelettique

Cette tension représente le confort de l'humain pour atteindre une configuration donnée à partir d'une configuration initiale. Elle est exprimée par un coût qui est égal à la somme pondérée des trois composantes suivantes [Mainprice 10] :

– La distance entre les configurations : La distance euclidienne qui sépare la configuration finale et la configuration initiale. Cette composante traduit l'effet "effort minimum", c'est à dire que

les petits déplacements seront favorisés par rapport aux déplacements de grande envergure. l'équation 3.1 illustre cette distance.

$$Dist_{cost} = \sqrt{\sum_{k=0}^{conf.size} (initConf.at(k) + finalConf.at(k))^2} \tag{3.1}$$

$initConf$ et $finalConf$ sont respectivement une configuration initiale et une configuration finale appartenant à C_e

– L'énergie potentielle : Toujours dans un souci d' "effort minimum", cette composante permet d'éviter les configurations d'échange où le point d'échange se trouve trop en hauteur (par rapport à l'humain). Ceci en utilisant la formule :

$$En_{cost} = \sum_{k=0}^{conf.size} (conf.at(k).mass * g * conf.at(k).height) \tag{3.2}$$

– Le confort des articulations [Katayama 03] : La position optimale d'un joint, pour cette composante, est la position milieu de son rayon d'action. De fait, cette composante représente la distance entre la configuration analysée et la configuration où les joints sont dans leur position optimale. L'équation 3.3 illustre ce confort.

$$Confort_{cost} = \sqrt{\sum_{k=0}^{conf.size} (conf.at(k) + conf.at(k).middle)^2} \tag{3.3}$$

le coût d'une configuration est donc :

$$Config_{cost} = \alpha * Dist_{cost} + \beta * En_{cost} + \gamma * Confort_{cost} \tag{3.4}$$

Les valeurs de α, β et γ sont à identifier selon les besoins.

Ce coût nous a permis de créer une grille de coût (disponible en annexe) centrée sur l'humain et son confort.

Le calcul de ce coût nécessite en entrée la configuration initiale de l'humain.

Cette qualification est indépendante de l'environnement (x, y et R_z ne sont pas pris en compte lors du calcul). Ces paramètres sont pris en compte lors du calcul du coût des déplacements et de leur temps d'exécution.

3.3.2 Coûts des déplacements de l'humain

Notre étude est centrée sur le confort de l'humain lors d'un échange d'objet. Pour que le confort soit maximal, le robot doit faire en sorte que l'humain se déplace le moins possible. Cependant, dans certains cas, ce déplacement devient inévitable, il faut donc pouvoir le qualifier, cela en lui donnant un coût.

Dans cette section, nous allons considérer seulement les déplacements en x, y et R_z ainsi que les changements de posture (debout,assis). De ce fait, nous pouvons diviser les déplacements en trois parties ayant des coûts distincts :

Les translations : en x et y, plus l'humain se déplacera, plus ce coût sera élevé :

$$trans_{cost} = \psi * dist \tag{3.5}$$

Les rotations : en R_z, plus l'humain devra se tourner plus ce coût sera élevé :

$$rot_{cost} = \delta * rot \tag{3.6}$$

Les changements de posture : Un seul changement de posture peut être réalisé, se lever. Ainsi, si l'humain est assis et doit se déplacer, un décalage sera ajouté à tous ses déplacements :

$$temp_{cost} = \begin{cases} \psi + \delta & \text{si} \quad \text{not } isStanding \quad \text{et} \quad dist > 0 \\ 0 & \text{sinon} \end{cases} \tag{3.7}$$

ψ représente la pente de la fonction de coût des translations, plus il est grand, plus les translations seront coûteuses. δ représente la même chose pour les rotations. Nous pouvons constater que les rotations sont beaucoup moins couteuses que les translations, δ doit donc être inférieure à ψ . Le décalage en cas de changement de posture est égal à la somme des deux pour garder une homogénéité dans les coûts.

Le coût final de déplacement est donc :

$$deplacement_{cost} = \frac{trans_{cost} + rot_{cost} + post_{cost}}{3} \tag{3.8}$$

Ce calcul de coût nécessite en entrée la position initiale la trajectoire de l'humain ainsi que sa rotation, en plus de la posture ($isStanding$).

3.3.3 Temps global d'exécution

L'urgence de l'humain à obtenir l'objet que le robot veut lui donner peut être traduite par une minimalisation du temps global que l'action (l'échange d'objet) va prendre. Ceci en considérant le début de l'action le moment où l'un des deux protagonistes commence à se déplacer et la fin, le moment où l'humain récupère l'objet.

Le coût temporel est donc :

$$post_{cost} = \begin{cases} min(rDist * Vrob, hDist * Vhum + const) & \text{si} \quad \text{not}\, isStanding \quad \text{et} \quad hDist > 0 \\ min(rDist * Vrob, hDist * Vhum) & \text{sinon} \end{cases} \tag{3.9}$$

$rDist$ et $hDist$ sont respectivement les distances parcourues par le robot et l'humain pour effectuer la trajectoire en entrée. $Vrob$ et $Vhum$ sont leurs vitesses de déplacement. $const$ est le temps que met l'humain à se lever.

La fig 3.7 montre un exemple de l'utilisation de ce coût.

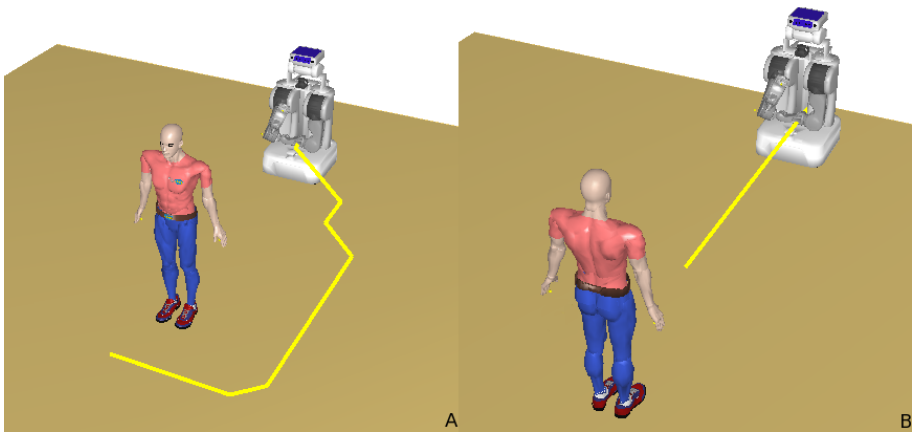

FIGURE 3.7 – En A, la configuration initiale de l'humain pousse le robot à contourner celui-ci. Si l'humain se déplace légèrement (une rotation en B), il réduit considérablement le temps d'exécution de l'action. Figure réalisée grâce à move3d [Simeon 01]

Ainsi les données nécessaires au calcul de ce coût sont la posture de l'humain et sa trajectoire ainsi que celle du robot.

3.4 Synthèse

Dans la suite, nous allons utiliser la structure $confData$ définie comme :

$$confData = \begin{cases} humanConf & \text{La configuration de l'humain} \\ humaonPos & \text{La position de l'humain en x, y, et } R_z \\ humanTraj & \text{La trajectoire de l'humain pour rejoundre } humanConf \\ robotConf & \text{La configuration du robot} \\ robotPos & \text{La position du robot en x, y, et } R_z \\ robotTraj & \text{La trajectoire du robot pour rejoindre } robotConf \\ cost & \text{Le cout de la configuration} \end{cases}$$

Afin de choisir la meilleure configuration pour effectuer l'échange d'objet, nous allons procéder en deux étapes :

Une partie hors ligne : Avant d'implémenter l'algorithme sur le robot, nous allons d'abord effectuer la discrétisation citée en section 3.1. Les configurations ainsi obtenues seront ensuite placées dans une liste ($confList$) de $confData$. Le coût de chaque configuration pourra alors être calculé en ce basant sur les calculs décrits en section 3.3.1. Finalement, la liste sera triée, par ordre croissant de coût. La fonction $getSortedConfigList$ de l'algorithme 3.4 prend en entrée la configuration initiale de l'humain et renvoie une liste triée de $confData$. Au lancement du robot, la liste $confList$ contient un ensemble de configurations (homme-robot) indépendantes de la position (x, y et R_z) de l'humain.

Une partie en ligne : En temps réel, le robot va d'abord créer une liste de positions choisies aléatoirement dans l'espace tridimensionnel représenté par x, y et R_z en y ajoutant la position initiale de l'humain. Ensuite, pour chaque position de cette liste, il va associer une configuration qu'il choisira dans $confList$. Le choix se portera sur la configuration au coût minimal et libre de collision. Ainsi une nouvelle liste ($posList$) de $confdata$ sera formée contenant les différentes positions associées à leurs configurations. Par la suite, pour chaque configuration de $posList$ les trajectoires (de l'homme et du robot) seront calculées (en utilisant la technique citée en section 3.2.2). Si une des deux trajectoires n'existe pas pour cette configuration, celle-ci sera enlevée de la liste. Si les deux trajectoires sont trouvées, le coût final de la configuration sera calculé, ceci en utilisant les résultats de la section 3.3.2, liés au déplacement de l'humain, et de la section 3.3.3, liés au temps d'exécution global de l'action, en les combinant avec le coût déjà calculé (celui décrit dans la section 3.3.1 lié au confort de l'homme.). Cette combinaison se fait selon l'équation 3.10. Finalement, la configuration retenue sera celle qui aura le coût minimal. l'algorithme 3.4 récapitule ces différentes étapes.

La fonction $getDiscretisation()$ renvoie une liste de $confData$ où seuls les membres $confData.humanConf$ et $confData.robotConf$ ont été remplis (en utilisant les données de la section 3.1). La fonction $getConfortCost$ quant à elle, calcule le coût de la configuration au niveau du confort de l'humain (ceci en utilisant les calculs de la section 3.3.1) et prend donc en entrée la configuration à évaluer et la configuration initiale de l'homme.

Algorithm 3.4.1 $getSortedConfigList(Q_{hum})$

Require: la configuration initial de l'humain Q_{hum}

1: $confList = getDiscretisation()$
2: **for** all c in $confList$ **do**
3: $\quad c.cost = getConfortCost(c, Q_{hum})$
4: **end for**
5: $confList = sortByCosts(confList)$
6: **return** $confList$

$$globalCost = (1 - objectNecessity) * (\xi * deplacement_{cost} + \rho * Config_{cost}) + objectNecessity * post_{cost} \tag{3.10}$$

ξ et ρ sont des degrés de liberté permettant d'adapter la solution au problème rencontré.

Algorithm 3.4.2 $getBestConf(humanInitPose, robotInitPos, isStanding, objectNecessity)$

Require: La position initiale de l'homme $humanInitPose$
Require: La position initiale du robot $robotInitPos$
Require: La posture de l'humain $isStanding$
Require: L'urgence de l'humain à récupérer l'objet $objectNecessity$

```
randomPos = getRandomPosition()
randomPos.push(humanInitPos)
for all pos in randomPos do
    setHumanToPos(pos)
    for all conf in confList do
        setHumanToConf(conf.humanConf)
        setRobotTopos(conf.robotPos)
        setRobotToConf(conf.robotConf)
        if human.isCollisionFree() and robot.isCollisionFree then
            conf.humanPos = pos
            goalNode = conf.humanPos.getNode()
            initNode = humanInitPos.getNode()
            path = findPath(initNode, goalNode)
            if path == NULL then
                break
            end if
            conf.humanTraj = path
            goalNode = conf.robotPos.getNode()
            initNode = robotInitPos.getNode()
            path = findPath(initNode, goalNode)
            if path == NULL then
                break
            end if
            conf.robotTraj = path
            DepCost = ComputeDepCost(conf.humanTraj, isStanding)
            tmpCost = ComputeTempCost(conf.humanTraj, conf.robotTraj, isStanding)
            objNes = objectNecessity
            conf.cost = (1 − objNes) ∗ (ξ ∗ DepCost + ρ ∗ conf.cost) + objNes ∗ tmpCost
            posList.push(conf)
            break
        end if
    end for
end for
return minCost(posList)
```

La fonction $getRandomPosition$ renvoie une liste aléatoire de position 3D.
Les fonctions $setHumanToPos$, $setHumanToConf$, $setRobotTopos$ et $setRobotToConf$ permette de changer la position est la configuration de l'humain et du robot afin d'effectuer les calculs.
La fonction $getNode$ permet de récupérer le nœud dans la grille de la fig 3.6 à partir d'une position en x, y et R_z.

La fonction *ComputeDepCost* utilise les calculs décrits en section 3.3.2 pour calculer le coût du temps global nécessaire à l'exécution de l'action.
La fonction *ComputeTempCost* utilise les calculs décrits en section 3.3.3 afin de calculer le coût.
La fonction *minCost* permet de récupérer la structure *confData* dont le coût est le plus petit dans une liste constituée d'un ensemble de *confData*.

3.5 Analyse des résultats

L'algorithme n'étant pas encore complètement implémenté, seule la partie implémentée sera analysée, la version finale de ce dossier comprendra l'analyse de l'algorithme complet.

La partie implémentée est la partie hors ligne, composée de la discrétisation de l'espace et du calcul du coût de confort de l'humain. La fig 3.8 représente les dix meilleures configurations (par rapport à leurs coûts) qui ont été calculées. Le tableau 3.1 montre les coûts qui leurs sont associés.

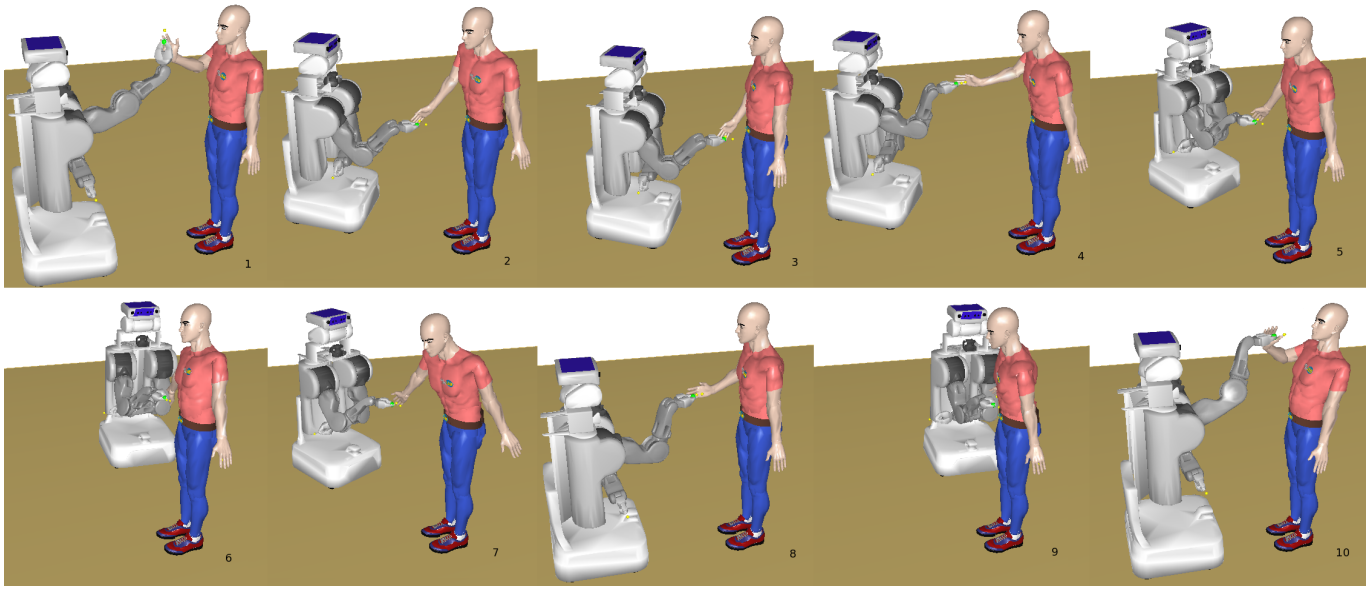

FIGURE 3.8 – Les meilleures configurations calculées. Figure réalisée grâce à move3d [Simeon 01]

TABLE 3.1 – Coût des meilleures configurations

classements	0	1	2	3	4	5	6	7	8	9
coûts	0.505	0.529	0.530	0.533	0.543	0.558	0.580	0.593	0.594	0.600

Afin d'obtenir ces configurations, nous avons identifié certaines variables (dont les valeurs sont résumées dans le tableau 3.2).

TABLE 3.2 – Choix des variables

variable	α	β	γ
valeur	0.8	0.6	10

Les Configurations que nous obtenons sont plausibles, le confort de l'humain est respecté, le robot se trouve dans son champs de vision et ne s'approche pas de manière exagérée. Ces différentes configurations sont stockées dans un fichier XML, est seront utilisées par la suite dans la partie en ligne.

4 Conclusion

L'algorithme décrit dans les sections précédentes permet de trouver une configuration d'échange d'objet entre le robot et l'humain dépendante du confort de l'humain et de l'urgence qu'il a de récupérer l'objet. Bien que pas encore totalement implémenté, on peut dors et déjà remarquer certaines limitations qui pourraient être améliorées :

- Le choix des positions adjacentes à la position initiale est aléatoire, ce choix peut être amélioré à l'aide de méta-heuristiques.
- Cet algorithme ne traite pas la navigation au long-cour, mais pourrait être amélioré dans ce sens.
- La recherche de chemin se fait en 2 dimensions sur une grille simplifiée. Ajouter les rotations et améliorer la grille permettrait de trouver des chemins plus intéressants.

Annexes

A* pseudocode

Algorithm 4.0.1 $findPath(initialNode, goalNode)$

```
1: vector openList, closedList
2: openList.push(initialNode)
3: closedList.clean()
4: currentNode = NULL
5: while currentNode! = goalNode do
6:    tmpNode = openList.pop()
7:    if tmpNode == goalNode then
8:       currentNode = tmpNode
9:    else
10:      closedList.push(tmpNode)
11:      n = NEIGHBORS(tmpNode)
12:      for all x in n do
13:         if x ϵ closedList and EVAL(tmpNode) < closedList.find(x).eval then
14:            closedList.find(x).eval = EVAL(tmpNode)
15:            closedList.find(x).parent = tmpNode
16:         else if x ϵ openList and EVAL(tmpNode) < openList.find(x).eval then
17:            openList.find(x).eval = EVAL(tmpNode)
18:            openList.find(x).parent = tmpNode
19:         else if x ∉ closedList and x ∉ openList then
20:            openList.push(x)
21:            openList.find(x).eval = EVAL(tmpNode)
22:            openList.find(x).parent = tmpNode
23:         end if
24:      end for
25:   end if
26: end while
27: return openList
```

Grille de confort

La figure 4.1 représente une grille de coût relatif au confort de l'humain. Trois paramètres entrent dans le calcul du coût de chaque point de cette grille [Sisbot 08] :

Le confort physique : Celui exprimé en section 3.3.1.

La distance du point par rapport à l'humain : La théorie de la proxémique [Hall 63] montre l'existence d'un seuil autour de l'humain qui délimiterait une zone d'intimité. L'introduction de tout objet dans cette zone peut créer un sentiment d'anxiété voire de peur. Dans ce cadre, ce paramètre aura un coût bas si le point est loin de l'humain et un coût élevé sinon.

La visibilité : Plus l'objet est visible (plus il se trouve proche de la direction du regard de l'humain) plus son coût sera bas par rapport à ce paramètre.

La somme pondérée des coûts résultants de ces trois paramètres nous permet de calculer la grille de la figure 4.1.

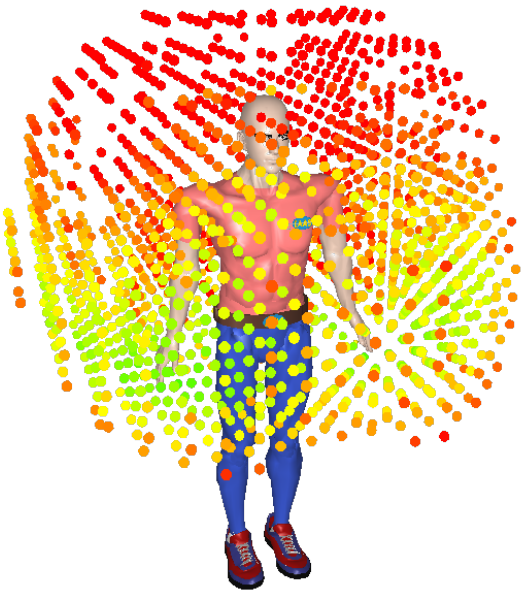

FIGURE 4.1 – La grille de coût de confort calculée. Les points verts ont un coût bas, alors que les points rouges ont un coût élevé. Figure réalisée grâce à move3d [Simeon 01]

Nous pouvons constater que les points les moins coûteux se trouvent à proximité des mains dans la position de repos (celle représentée dans la fig 4.1) or dans la vie quotidienne, nous remarquons que les échanges d'objet ne s'effectuent pas à cet endroit mais s'effectuent en face de l'humain en général.

Études d'utilisateurs : Distance optimale

Présentation de l'étude

Nous allons mener une étude d'utilisateur sur la distance qui doit séparer le robot et l'humain lors d'un échange d'objet. Cette distance doit être suffisamment grande pour respecter la théorie de la proxémique [Hall 63] : si la distance d'un objet à l'humain est inférieure à un certain seuil, l'objet peut créer un sentiment d'anxiété voire de peur chez l'humain. Elle ne doit pas non plus être trop grande pour ne pas perturber l'humain et lui permettre de récupérer l'objet. Cette distance est très fortement liée au robot : elle dépend de son encombrement, ainsi que de sa portée. Comme précédemment, nous allons utiliser le robot PR2 (fig 3.2) pour mener cette étude.

Procédure

Durant cette étude, l'humain restera dans une position fixe, prêt à recevoir un objet du robot. Celui-ci devra quant à lui se rendre à des positions (ainsi qu'aux configurations correspondantes) qui seront échantillonnées le long de la droite correspondante à l'orientation de l'humain. La figure 4.2 montre les différentes positions ainsi que les configurations que devra prendre le robot.

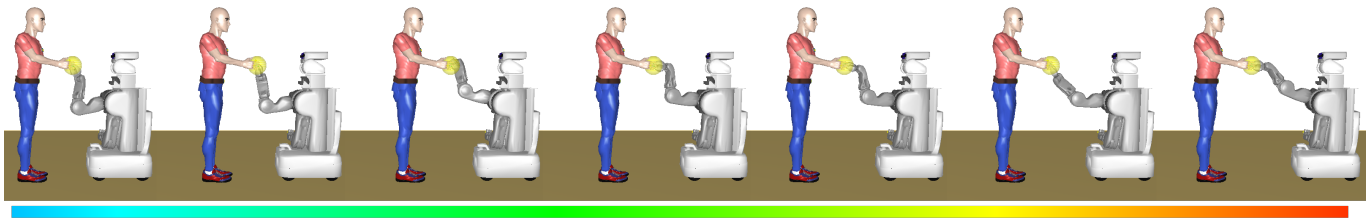

FIGURE 4.2 – Différentes positions du robot par rapport à l'humain. De la plus proche (à gauche) à la plus éloignée (à droite). Figure réalisée grâce à move3d [Simeon 01]

Ce test sera reproduit sur un nombre n de personnes qui ne sont pas liées à cette étude, mais qui connaissent le but de l'expérience. Ensuite ils rempliront le formulaire 4.3 avec leurs impressions.

	1	1.05	1.1	1.15	1.2	1.25	1.3
trop loin							
loin							
bien							
proche							
trop proche							

FIGURE 4.3 – Tableau à remplir par les candidats de l'étude. Les distances correspondent aux distances de la fig 4.2.

Une fois les résultats obtenus, nous pourrons les utiliser afin de déterminer une courbe d'impression par rapport aux distances et ainsi avoir une approximation de la distance optimale entre l'homme et le robot.

Études d'utilisateurs : Comparaison entre les configurations réelles et calculées

Présentation de l'étude

Nous allons, au cours de cette étude, comparer différentes configurations calculées par cinématique inverse, avec des configurations recueillies directement à partir de la position et de la posture de l'humain. Cette comparaison permettra par la suite d'améliorer notre estimation du coût de confort de l'humain et ainsi de trouver des configurations d'échanges plus pertinentes.

Procédure

La discrétisation de l'espace en section 3.1 permet de récupérer un nombre i de configurations. Grâce au calcul de leur coût de confort en section 3.3.1, nous pouvons retenir les 10 meilleures configurations par rapport au coût de confort (fig 3.8). Cette études se déroulera donc de la manière suivante :

- Pour chacune des 10 configurations, le robot va se placer en fonction de l'homme en lui présentant l'objet.
- L'homme va se mettre en position pour récupérer l'objet.
- Nous allons récupéré la configuration d'échange de l'humain grâce au kinect (le capteur visuel de Microsoft `http://www.xbox.com/en-US/kinect`) installer sur le robot.
- L'utilisateur notera chacune de ces configurations de 1 à 10. tableau 4.1

TABLE 4.1 – Coût des meilleures configurations

classements	0	1	2	3	4	5	6	7	8	9
notes										

L'étude sera répétée sur un nombre n de personnes qui ne sont pas liées à l'étude. Ces personnes seront néanmoins mises au courant du cadre du problème ainsi que l'utilisation des résultats de cette expérience.

Apport sur le plan personnel

Ce stage m'a permis de découvrir la recherche en général, et le domaine de la robotique en particulier, domaine qui me tient à cœur depuis longtemps. De plus, j'ai pu évoluer dans une équipe de recherche très sympathique dont le spectre d'étude s'étend de la perception à la planification de mouvement en passant par l'étude des comportements humains et sociaux.

Durant cette période, j'ai aussi réalisé la différence entre l'état d'esprit dans le monde de la recherche et celui que j'ai côtoyé durant ma scolarité ou encore durant mes précédents stages. Dans ce cadre, et après une période d'adaptation, j'ai pu évoluer de manière positive et constante.

Bibliographie

[Alili 09] Matthieu Warnier ; Muhammad Ali ; Rachid Alami ; Samir Alili. *Planning and Plan-execution for Human-Robot Cooperative Task achievement.* Conference on Automated Planning, 2009. 8

[Broquere 08] D. Sidobre ; I. Herrera-Aguilar ; X. Broquere. *Soft motion trajectory planner for service manipulator robot.* Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on, 2008. 8

[Cakmak 11] Siddhartha Srinivasa ; Min Kyung Lee ; Sara Kiesler ; Jodi Forlizzi ; Maya Cakmak. *Using Spatial and Temporal Contrast for Fluent Robot-Human Hand-overs.* 6th ACM/IEEE International Conference on Human-Robot Interaction,, 2011. 13

[Clodic 09] Aurélie Clodic, Hung Cao, Samir Alili, Vincent Montreuil, Rachid Alami & R. Chatila. *SHARY : A Supervision System Adapted to Human-Robot Interaction.* In Experimental Robotics, Springer Tracts in Advanced Robotics. Springer Berlin / Heidelberg, 2009. 7

[Gharbi 08] J. Cortés ; T. Siméon ; M. Gharbi. *A sampling-based path planner for dual-arm manipulation.* Advanced Intelligent Mechatronics, 2008. AIM 2008. IEEE/ASME International Conference on, 2008. 8

[Gregory 05] Ming C. Lin ; Stefan Gottschalk ; Russell Taylor ; Arthur Gregory. *A Framework for Fast and Accurate Collision Detection for Haptic Interaction.* Proceeding SIGGRAPH '05 ACM SIGGRAPH 2005 Courses, 2005. 17

[Hall 63] EDWARD T. Hall. A system for the notation of proxemic behavior. American Anthropologist, 1963. 30, 31

[Hirzinger 02] G. Hirzinger, N. Sporer, A. Albu-Schäffer, M. Hähnle, R. Krenn, A. Pascucci & M. Schedl. *DLR's Torque-Controlled Light Weight Robot III - Are We Reaching the Technological Limits Now ?* pages 1710–1716. IEEE, 2002. 6

[Kajikawa 95] S. Kajikawa, T. Okino, K. Ohba & H. Inooka. *Motion planning for hand-over between human and robot.* In Intelligent Robots and Systems 95. 'Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on, 1995. 13

[Katayama 03] Hiroyuki Hasuura ; Masazumi Katayama. *Optimization principle determines human arm postures and "comfort"*. SICE 2003 Annual Conference, 2003. 20

[Mainprice 10] Jim Mainprice, E. A. Sisbot, Thierry Siméon & Rachid Alami. *Planning Safe and Legible Hand-over Motions for Human-Robot Interaction*. In IARP Workshop on Technical Challenges for Dependable Robots in Human Environments, 2010. 8, 10, 19

[Marin-Urias 09] Luis Felipe Marin-Urias. *Reasoning About Space for Human-Robot Interaction*. PhD thesis, Université Toulouse III - Paul Sabatier Ecole Doctorale Systemes, 2009. 11, 18

[Simeon 01] T. Simeon, J p. Laumond & F. Lamiraux. *Move3D : a generic platform for path planning*. In in 4th Int. Symp. on Assembly and Task Planning, pages 25–30, 2001. 8, 11, 12, 14, 16, 17, 18, 19, 22, 26, 30, 31

[Sisbot 07a] E.A. Sisbot, L.F. Marin & Alami R. *Spatial reasoning for human robot interaction*. In IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007. 10

[Sisbot 07b] E.A. Sisbot, L.F. Marin-Urias, R. Alami & T. Siméon. *A Human Aware Mobile Robot Motion Planner*. IEEE Transactions on Robotics, vol. 23, 2007. 18

[Sisbot 08] Emrah Akin Sisbot, Aurelie Clodic, Rachid Alami & Maxime Ransan. *Supervision and motion planning for a mobile manipulator interacting with humans*. In Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction, HRI '08, pages 327–334, New York, NY, USA, 2008. ACM. 8, 30

[Sisbot 10] Emrah Sisbot, Luis Marin-Urias, Xavier Broquère, Daniel Sidobre & Rachid Alami. *Synthesizing Robot Motions Adapted to Human Presence*. International Journal of Social Robotics, vol. 2, pages 329–343, 2010. 10.1007/s12369-010-0059-6. 5, 6

[Zinn 04] M. Zinn, O. Khatib, B. Roth & Salisbury J.K. *Playing it safe [human-friendly robots]*. IEEE Robotics and Automation Magazine, vol. 11, 2004. 6

Résumé

L'interaction homme-robot est un domaine de recherche récent. Dans ce cadre, une des problématique qui est posée est l'échange d'objets entre humain et robot. Cette étude se centre sur cette problématique en se concentrant sur le transfert de petits objets (un témoin par exemple) du robot vers l'humain. Elle prend en compte les positions initiales des deux protagonistes ainsi que la posture de l'humain et son besoin de l'objet que veut lui donner le robot pour planifier un mouvement pour le robot,bien sûr, mais surtout pour l'humain afin qu'ils puissent réaliser l'échange de manière coopérative.

Mots clefs : interaction homme-robot, HRI, Confort de l'humain, échange d'objet (hand-over), planification de trajectoire.