Exécution réactive de trajectoires pour robots mobiles non-holonomes

David Bonnafous

Thèse

préparée au

Laboratoire d'Analyse et d'Architecture des Systèmes

en vue de l'obtention du titre de

Docteur de l'Institut National Polytechnique de Toulouse

Systèmes Automatiques

Execution réactive de trajectoires pour robots mobiles non-holonomes

David BONNAFOUS

Soutenue le 22 décembre 2003 devant le jury

Président	**M. LAUMOND**
Rapporteurs	**M. LAUGIER** **M. DE LUCA**
Directeurs de thèse	**M. LAMIRAUX** **M. CHATILA**

Table des matières

Introduction **5**

1 Problématique et état de l'art **7**
1.1 Problématique 7
1.1.1 Histoire et contexte 7
1.1.2 La navigation autonome 8
1.1.3 Les difficultés de la navigation 9
1.2 État de l'art 11
1.2.1 Les algorithmes de replanification rapide. 12
1.2.2 L'évitement réactif en vitesse 13
1.2.3 Utilisation de trajectoires d'évitement 15
1.2.4 La bande élastique 15
1.2.5 Planification de trajectoires à partir de données proximétriques 17
1.3 Notre approche et notre contribution 17

2 Déformation de trajectoire **19**
2.1 Présentation du problème 19
2.1.1 Système non-holonome et trajectoires 19
2.1.2 La déformation élémentaire de trajectoire 20
2.1.3 La déformation de trajectoire comme système asservi . 23
2.1.4 Champ de potentiel et produit scalaire 23
2.1.5 Résumé du problème 25
2.2 Résolution : l'algorithme de déformation 26

2.2.1 Les étapes de la déformation 26
2.2.2 L'espace de dimension finie des perturbations 27
2.2.3 Champ de potentiel et déformation élémentaire 29
2.2.4 Prise en compte des conditions aux limites 30
2.2.5 Produit scalaire et base orthonormée 31
2.2.6 Dérive des contraintes non-holonomes 36
2.2.7 Résumé : l'algorithme de déformation de trajectoire non-holonome 38
2.3 Exemple : le robot Hilare 2 39

3 Définition du potentiel d'une trajectoire 45
3.1 Introduction 45
3.2 Évitement d'obstacles 46
3.3 Exemples 50
3.3.1 Exemple 1 : le robot Hilare 2 50
3.3.2 Exemple 2 : le robot Hilare 2 muni d'une remorque . . 52
3.3.3 Exemple 3 : camion à remorque avec timon 55
3.4 Utilisation de segments 57

4 Mise en oeuvre à bord du robot Hilare 2 61
4.1 Difficultés d'implantation à bord du robot Hilare 2 61
4.2 Hypothèses et contraintes 62
4.3 Algorithme mis en œuvre 65
4.3.1 cCheck : recherche des collisions 66
4.3.2 tFollow, ou comment et quand arrêter Hilare 2 69
4.3.3 tDeform : déformons l'intervalle de trajectoire 72
4.4 Expérimentation : suivi d'une trajectoire 73
4.5 Expérimentation : déformation d'une trajectoire 77

Conclusion 83

Bibliographie 84

Introduction

Mes travaux, présentés dans ce mémoire, s'intéressent à la navigation d'un robot mobile à roues soumis à des contraintes de roulement sans glissement dans son environnement.

Plus précisément la problématique abordée est l'exécution de trajectoires planifiées pour de tels robots.

Le mémoire est organisé de la manière suivante.

Dans le premier chapitre, nous présentons le rôle de la navigation pour une machine mobile intelligente et les problèmes qu'elle soulève. Dans ce même chapitre, nous faisons un état de l'art des méthodes existantes et nous présentons notre approche de la navigation.

Le deuxième chapitre décrit en détail notre algorithme d'adaptation de la trajectoire du robot en fonction des obstacles qu'il perçoit. Il s'agit d'une méthode de déformation locale de la trajectoire basée sur la connaissance de la loi de mouvement du robot et des obstacles perçus.

Dans le chapitre trois, nous développons les techniques utilisées pour modéliser l'interaction entre les obstacles perçus et la trajectoire.

Le dernier chapitre présente l'implantation de la méthode de déformation de trajectoire à bord du robot d'expérimentation Hilare 2. Nous présentons les objectifs à atteindre puis les contraintes à respecter et enfin les algorithmes que nous avons développés pour le suivi de la trajectoire déformée.

I
Problématique et état de l'art

1.1 Problématique

1.1.1 Histoire et contexte

Des milliers d'années après l'invention de la roue, vers 3000 ans avant Jésus Christ, l'homme a enfin trouvé le moyen de se déplacer plus rapidement et plus loin sans attendre que sa monture soit reposée. La machine à vapeur va révolutionner la vie des hommes. L'industrie va se développer très rapidement et les premiers véhicules entièrement contrôlés par l'homme, les trains par exemple, vont lui permettre d'aller encore plus vite et plus loin.

La dernière grande invention de l'homme pour se déplacer c'est le moteur à explosion. Sa relative facilité de mise en oeuvre a eu raison de son médiocre rendement et de sa nocivité. Il a rapidement équipé toutes sortes de véhicules : les voitures particulières, les autobus, les camions de transport, les engins de chantier, etc. Malgré de nombreuses améliorations pour rendre le pilotage de ces engins moins difficile et plus sûr comme la boîte de vitesse automatique, le régulateur de vitesse, les systèmes d'aide à la navigation, etc. le conducteur humain reste encore indispensable au guidage du véhicule.

La communauté scientifique et les industriels travaillent depuis longtemps à automatiser les véhicules.

La robotique mobile cherche depuis des années à rendre une machine mobile autonome face à son environnement pour qu'elle puisse sans intervention humaine accomplir les missions qui lui sont confiées. Le spectre des missions que les roboticiens veulent voir accomplir par leurs machines est im-

mense : exploration en terrain inconnu, manipulation d'objets, assistance aux personnes handicapées, transport automatisé, etc. De grand progrès ont été accomplis dans tous les domaines de la robotique : perception et modélisation de l'environnement, commande automatique des actionneurs, planification de mouvements, ordonnancement de tâches, gestion de l'énergie,...

Bien que la machine autonome parfaite, capable de s'adapter à de nombreuses situations et capable d'estimer ses possibilités d'actions, n'existe pas encore, des applications concrètes des techniques de robotique mobile ont été faites. On peut citer par exemple la solution entièrement robotisée de transport collectif pour le tourisme culturel mise en place par la société ROBOSOFT dans le fort du Simserhof en Moselle (voir communiqué de presse ROBOSOFT du 4 avril 2003). La société allemande FOX GmbH utilise aussi des techniques de robotique (détection laser d'obstacles, manoeuvres automatiques,...) pour automatiser entièrement ou partiellement des véhicules industriels.

Ces applications très remarquables ne sont toutefois que des cas particuliers pour lesquels des solutions plus ou moins spécifiques ont été développées. En effet, les problèmes posés par la navigation autonome d'un véhicule sont très nombreux et font l'objet de recherches soutenues. En simplifiant, une machine autonome c'est l'association d'une intelligence artificielle (la capacité de raisonner) avec des capacités de perception et de modélisation de son environnement et de son propre état et des capacités d'actions sur son propre état (mouvement) et sur son environnement (manipulation).

Toutes ces capacités sont nécessaires mais chacune amène ses difficultés. Alors, pour progresser, les chercheurs de chaque spécialité isolent les problèmes, proposent des solutions et les comparent. Les robots d'expérimentations des laboratoires deviennent alors les cobayes des sciences naturelles. On implante sur leurs calculateurs un planificateur de mouvement à réseau probabiliste, on connecte sur leurs bus d'acquisition une paire de caméras ou un capteur télé-métrique laser, etc. et on étudie les possibilités d'actions obtenues.

1.1.2 La navigation autonome

Intéressons nous maintenant au problème de la navigation d'un robot mobile dans son environnement. C'est à dire la capacité d'aller d'une position initiale à une position finale de manière autonome. C'est là, encore, un vaste thème. En effet, les méthodes utilisées pour un robot du type Masokhod [Bonnafous 2001] en environnement d'extérieur (non structuré) inconnu ne sont pas les mêmes que pour un robot cylindrique omnidirectionnel en environnement d'intérieur (structuré).

Considérons la cas d'un robot mobile en environnement structuré.

Dans le cas idéal d'un environnement exactement modélisé et où la cinématique du robot est parfaitement maîtrisée les étapes de la navigation se résument simplement : le robot planifie une trajectoire puis l'exécute.

Mais alors pourquoi nos voitures ne sont-elles pas entièrement automatiques ?

Dans la réalité nous faisons, sans nous en rendre compte, des actions extrêmement complexes lorsque nous nous déplaçons : nous estimons notre position, nous analysons en permanence les objets qui nous entourent et nous nous mouvons. Un robot doit faire de même.

1.1.3 Les difficultés de la navigation

Le mouvement

En robotique, il existe deux grandes catégories de robots mobiles à roues. Je ne parlerai pas des robots à pattes.

Les véhicules non-holonomes

Les véhicules dit non-holonomes sont ceux que l'on rencontre le plus dans la vie courante : voiture particulière, bus, camion,... Ces véhicules ont une structure mécanique relativement simple : des roues motrices, des roues directrices et des roues porteuses. Une roue peut avoir une, deux ou les trois fonctions. Mais tous ces véhicules ont une caractéristique commune : la direction de la vitesse d'avance (ou vitesse linéaire) est imposée par la direction des roues directrices. Pour fixer les idées prenons un exemple : pour qu'une voiture particulière aille de sa position initiale à un mètre sur sa droite elle est obligée de faire une manoeuvre : une marche avant puis une marche arrière.

Les véhicules holonomes

Le deuxième type de véhicules, beaucoup plus rare dans notre vie quotidienne, s'appelle les véhicules holonomes. Ils ont une structure mécanique complexe qui leur permet de se déplacer dans toutes les directions sans manoeuvre !

Il existe, néanmoins, un exemple commun (et bien pratique) : le chariot de magasin. Vous pouvez prendre un chariot et le tirer sur la droite, les roues s'orientent alors dans la bonne direction.

Un chariot n'a pas une structure mécanique compliquée mais ses articulations sont passives. La société Nomadic, disparue en l'an 2000, a conçu un robot holonome : le XR4000. Il dispose de 4 roues motrices et directrices montées comme des roues de chariot [Holmberg 2000]. La synchronisation des 8 axes (2 par roue, rotation et orientation) est assurée par une carte dédiée

basée sur le micro-contrôleur motorola 68332 et des circuits FPGAs et la structure mécanique est composée d'engrenages coniques.

Robot holonome ou non-holonome : les différences

La relative simplicité mécanique des robots non-holonomes face aux robots holonomes à un prix. Les algorithmes d'exécution de trajectoire, de commandes et de planification de mouvements sont plus complexes. Mais leur facilité de construction en font des véhicules très répandus dont l'automatisation présente un intérêt socio-économique évident.

La localisation

Quel que soit le type de robot, holonome ou pas, la connaissance de sa position dans son environnement (son espace de travail) est importante pour s'y déplacer.

Il existe principalement deux types de localisation. La localisation topologique et la localisation métrique. Dans le cas de la localisation topologique, le robot se situe de manière abstraite. Il utilise des primitives de haut niveau pour se situer : je suis dans le bureau E46 ou dans le couloir C2. A l'opposé, la localisation métrique donne les coordonnées du robot dans un repère fixe. Dans le cas où l'espace de travail est le plan, il s'agit donc de l'abscisse, de l'ordonnée et de l'orientation du corps principal du robot.

Nous parlons ici uniquement de localisation métrique car elle est nécessaire à nos travaux.

La localisation métrique d'un robot dans son espace de travail est un thème de recherche très actif [Hayet 2003]. La difficulté de connaître la position exacte d'un robot vient de l'impossibilité de mesurer précisément ses déplacements. Différentes techniques existent parmi lesquelles l'odométrie mécanique par intégration du mouvement des roues du robot, l'odométrie optique ou estimation visuelle du mouvement par stéréo vision ou vision monoculaire, utilisation d'un GPS...

Mais chaque méthode présente des inconvénients. L'odométrie, par exemple, intègre au cours du temps le déplacement du robot mesuré par des roues odométriques placées sur les roues motrices du robot. Ces mesures ont une certaine précision et donc l'odométrie mécanique est fiable sur des petites distances. Sur de longues distances la dérive devient trop importante.

Il existe des techniques qui tentent de pallier les défauts de telle ou telle méthode en fusionnant les données issues de plusieurs capteurs. Elles sont difficiles à mettre oeuvre car il faut prendre en compte les incertitudes de chaque mesure, la date d'acquisition des données, la date de production du résultat, la période d'acquisition,...

Quoi qu'il en soit, les algorithmes de navigation doivent, par une méthode

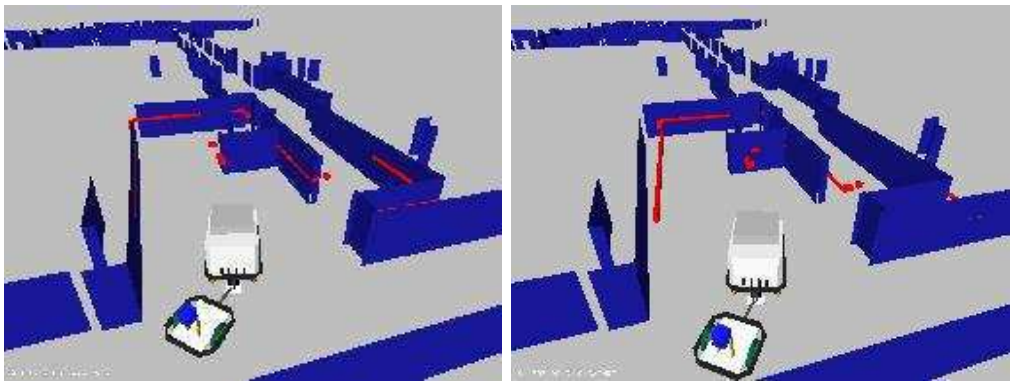

FIG. 1.1 – Sur la figure de gauche, le robot est bien localisé, les obstacles perçus (segments clairs) coïncident avec les obstacles du plan initial. Sur la figure de droite, le robot est mal localisé, les obstacles perçus ne coïncident pas avec les obstacles du plan initial.

ou une autre, prendre en considération le fait qu'il ne connaît jamais sa position avec exactitude.

Exécution du mouvement

Pour naviguer dans son environnement, une machine autonome doit pouvoir se le représenter.

Le robot peut utiliser une carte globale de son environnement pour planifier une trajectoire. Cette carte peut-être complète ou partielle, construite par le robot ou par l'homme. Le robot pourra la mettre à jour pendant ses missions. Dans la grande majorité des cas la trajectoire créée dans un environnement modélisé ne peut être exécutée, ainsi, sans collision. Il existe toujours des artefacts de mesure ou des imprécisions que le robot doit être capable de prendre en compte pendant l'exécution du mouvement.

Les primitives utilisées pour la planification de trajectoire et celles utilisées pendant l'exécution ne sont pas nécessairement les mêmes. Il faut utiliser des primitives adaptées à la tâche considérée.

1.2 État de l'art

Dans l'exposé de la problématique abordée dans ce mémoire, nous avons décrit les principales difficultés de la navigation d'un robot mobile. Il en découle une certitude : le robot doit pendant l'exécution d'une trajectoire

prendre en compte son environnement. Il doit le faire pour plusieurs raisons :

- compenser les erreurs de localisation,
- compenser les imprécisions de modélisation de l'environnement initial,
- éviter les collisions avec les obstacles inconnus ou mobiles.

De nombreuses méthodes ont été étudiées pour permettre à un robot (holonome ou non-holonome) de se déplacer sans collision dans un environnement partiellement connu ou inconnu. Nous allons présenter, dans les sections suivantes, cinq techniques majeures développées dans ce but.

La première technique utilise des algorithmes de replanification rapide pour trouver un nouveau chemin après avoir détecté une modification de l'environnement. La suivante calcule des vitesses de manière réactive pour éviter les obstacles tout en essayant de suivre une trajectoire de référence. La troisième calcule des trajectoires d'évitement. La quatrième change de modalités de déplacement suivant l'encombrement de l'environnement. La dernière considère la trajectoire comme un élastique qui se tend sous la poussée des obstacles.

Enfin, la dernière section présente deux techniques de planification de trajectoires basées sur des données proximétriques qui utilisent des techniques proches de celles que nous avons utilisées dans notre méthode de suivi de trajectoires réactif.

1.2.1 Les algorithmes de replanification rapide.

Les algorithmes de replanification rapide sont basés sur l'algorithme A* de recherche de chemin optimal dans un graphe grâce à une heuristique [Nilsson 1971].

L'espace de travail du robot $\mathcal{W}$ est découpé en cellules. Chaque cellule peut être qualifiée soit de "libre" si le robot peut la franchir soit d'obstacle si elle est infranchissable.

Les arcs du graphe représentent un mouvement du robot d'une cellule à une autre.

La trajectoire obtenue est toujours optimale pour le critère utilisé pour valuer les arcs du graphe. Ce critère est généralement la distance parcourue mais il peut aussi être défini par l'utilisateur.

L'apport de ces méthodes par rapport à l'algorithme A* est la possibilité de modifier la solution lorsque le robot détecte des modifications de l'environnement sans avoir à recommencer toute la recherche. Ainsi, ces algorithmes

peuvent être utilisés en ligne contrairement à l'algorithme A* qui est trop coûteux en temps.

Deux approches existent. Elles se distinguent par la façon de faire évoluer le graphe au fur et à mesure que le robot acquiert de nouvelles informations.

- Dans la première approche, le coût des arcs du graphe évolue. On parle alors de l'algorithme D* [Stentz 1994] et plus particulièrement la version avec heuristique de recherche "Focussed D*" [Stentz 1995]. Plus récemment [Koenig 2002] propose une méthode équivalente mais algorithmiquement plus simple.
- Dans la deuxième approche, les noeuds du graphe sont supprimés ou rajoutés. On peut citer IGPRR (Intelligent Global Path Planner with Replanning) [Podsdkowski 1998] et [Podsdkowski 2001].

Un inconvénient majeur de ces méthodes réside dans le fait que la cinématique des robots est très peu prise en compte. Par exemple, dans IGPRR la vitesse du robot est choisie parmi un ensemble fini de couples (vitesse linéaire, vitesse angulaire).

De plus les possibilités de planification de ces méthodes sont bien inférieures aux méthodes utilisant un échantillonage aléatoire de l'environnement (en particulier pour des systèmes non-holonomes complexes).

1.2.2 L'évitement réactif en vitesse

Ces méthodes associent une méthode d'évitement local d'obstacles avec une information globale sur la connectivité de l'espace de travail (ou des configurations) du robot pour fournir la vitesse linéaire et angulaire du robot.

La première méthode, appelée "Global Dynamic Window Approach" est décrite dans [Brock 1999]. Elle a été développée pour un robot holonome mais devrait pouvoir s'adapter à un robot non-holonome. Elle associe la méthode d'évitement local d'obstacles appelée "Dynamic Window Approach" [Fox 1997] à une carte de potentiels sans minimum local dans l'espace des configurations grâce à la fonction de navigation NF1 [Latombe 1991].

La deuxième méthode est décrite dans [Xu 2003]. Elle a été développée dans le cadre d'un projet industriel pour équiper un transpalette. Elle utilise une carte locale des obstacles et un chemin de référence que doit suivre le robot. Lors d'un évitement, le robot utilise des ancres virtuelles sur les obstacles et des sous buts le long du chemin pour ne pas se perdre.

La troisième méthode est présentée dans [Mínguez 2000]. Cette technique appelée "Nearness Diagram navigation" (ND) s'appuie sur l'analyse des obstacles perçus par le robot (avec plus ou moins de mémoire) pour en déduire les conditions de navigation : navigation en espace contraint ou en espace libre. A chacune de ces conditions de navigation correspond une stratégie de navigation qui, dans tous les cas, produit la vitesse linéaire jugée la meilleure pour la stratégie utilisée. Dans [Mínguez 2002a] les auteurs proposent une architecture logicielle permettant d'étendre l'utilisation de "ND" à des systèmes non-holonomes. La vitesse linéaire calculée par "ND" est transformée en une force virtuelle qui tire le robot selon un modèle cinématique et dynamique du mouvement du robot. Des expériences ont été faites sur différents robots non-holonomes relativement simples (espace des configuration de dimension 3) avec succès.

Dans [Mínguez 2002b] les mêmes auteurs décrivent un espace appelé "Ego-Kinematic Space" dans lequel les points obstacles et les configurations atteignables par le robot en un seul mouvement (ligne droite ou arc de cercle) sont transformés en un couple de réels par une transformation bijective. Dans cet espace, n'importe quel algorithme d'évitement d'obstacles développé pour un robot holonome peut être utilisé pour des robots non-holonomes.

La quatrième méthode est décrite originalement dans [Fiorini 1998]. Dans cette méthode, la vitesse des obstacles est supposée connue ou mesurable et le concept de "Velocity Obstacle" permet de définir l'ensemble des vitesses qui assure, sur un horizon de temps donné, la non-collision du robot avec les obstacles. Dans le cas où la vitesse des obstacles est connue à l'avance, cette technique permet de planifier des trajectoires dans un environnement fortement dynamique. Dans le cas contraire, la vitesse des obstacles est estimée et supposée constante sur des petits horizons de temps. On détermine alors pour le prochain intervalle de temps la meilleure vitesse qui assure la non collision et le rapprochement du but. Dans sa thèse, Frédéric Large [Large 2003] étend cette méthode au cas où la vitesse des obstacles n'est pas constante par morceaux. Il présente alors l'exemple simulé de l'insertion d'une voiture, modélisé par un point, sur une voie à circulation rapide.

La dernière méthode est décrite dans [Bemporad 1996]. Elle s'applique à un robot du type voiture ou unicycle. À partir de deux forces dans le plan de travail appliquées en un point du robot, la première qui tire le robot vers le but et la seconde qui éloigne le robot des obstacles, elle détermine les commandes en vitesse à appliquer au robot. Cette détermination est faite en utilisant la pseudo-inverse du modèle cinématique.

Néanmoins, l'extension de ces techniques à des robots non-holonomes plus complexes ayant des configurations de dimension 4 et plus, semble difficile. De plus l'évitement purement réactif d'obstacles simplement guidé par une trajectoire initiale pour de tels systèmes ne donnerait pas de bons résultats. Il est nécessaire de considérer la trajectoire entière pour franchir des passages difficiles sans quoi le robot a de fortes chances de se placer dans une configuration dans laquelle il ne peut poursuivre son mouvement.

1.2.3 Utilisation de trajectoires d'évitement

Dans [Laugier 1999], les auteurs proposent une architecture de contrôle pour une voiture autonome évoluant sur le réseau routier. Les éléments de base sont des SMB, Sensor-Base Manoeuvre. Ce sont des capacités "élémentaires", comme par exemple suivre une trajectoire ou se garer, que le robot ordonnance pour effectuer une mission.

La capacité de suivi de trajectoire se décompose en deux sous-capacités : le suivi de trajectoire et l'évitement d'obstacle.

Le véhicule suit une trajectoire initiale produite par le planificateur décrit dans [Scheuer 1997] et utilise une famille de trajectoires d'évitement pour contourner les éventuels obstacles qui obstruent la trajectoire initiale.

Lorsque le véhicule détecte un obstacle immobile ou mobile, les paramètres de la trajectoire d'évitement sont calculés et, selon ses possibilités le robot décide soit de suivre cette trajectoire, soit de ralentir ou même de s'arrêter.

On peut dire que cette méthode d'évitement travaille dans un sous-espace des trajectoires faisables du robot.

Cette méthode a été utilisée sur deux véhicules d'expérimentations : une Ligier et le Cycab [Large 2000] dans des environnements routiers dégagés.

Dans cette méthode, la cinématique et les possibilités d'accélération des véhicules sont rigoureusement respectées mais l'utilisation d'une famille de trajectoires d'évitement empêche l'utilisation de cette méthode pour des environnements contraints qui exigent des capacités de manœuvres plus fines.

1.2.4 La bande élastique

Cette méthode, développée dans [Quinlan 1993] pour des robots holonomes, utilise une trajectoire dans l'espace des configurations du robot produite par un planificateur quelconque.

La trajectoire est considérée comme un élastique sur lequel des forces virtuelles externes, créées par les obstacles, s'exercent et le repoussent au loin alors que des forces internes sont créées pour assurer la cohésion de la trajectoire (conserver la classe d'homotopie de la trajectoire).

Pour être implantée sur un calculateur, la trajectoire est discrétisée en une liste de configurations. Pour être efficace, c'est à dire minimiser le nombre de configurations à traiter, la discrétisation est effectuée en utilisant le concept de "bulle".

Une bulle est définie comme le plus grand sous-espace libre dans l'espace des configurations qui forme une sphère autour du centre de la bulle dans l'espace métrique associé. Si $\rho(\mathbf{r})$ est une fonction qui donne la distance minimale entre le robot dans la configuration $\mathbf{r}$ et les obstacles de l'environnement la bulle $\mathcal{B}(\mathbf{r})$ est l'ensemble des configurations $\mathbf{q}$ tel que $\|\mathbf{r} - \mathbf{q}\| < \rho(\mathbf{r})$.

Ainsi, une trajectoire peut être discrétisée en un nombre minimal de configurations. De plus, pour garantir qu'une trajectoire est faisable, il suffit qu'elle soit entièrement recouverte de bulles.

Dans le cas de robots holonomes la définition des bulles est simple. Quelques exemples sont traités dans [Quinlan 1994] et [Khatib 1996].

Pour des robots non-holonomes la définition d'une métrique non-holonome appropriée est beaucoup plus difficile à définir.

Le cas particulier du robot-voiture a été particulièrement étudié. Les chemins les plus courts entre deux configurations pour un tel robot ont été définis par J.A. Reeds et R.A. Shepp dans [Reeds 1990] : les chemins de Reeds et Shepp. Par la suite, de nombreux travaux résumés dans [Vendittelli 1999] ont défini les plus courtes distances entre une configuration d'un tel robot et des obstacles.

Utilisant ces résultats, Maher Khatib dans [Khatib 1997] définit des bulles pour des robots voitures et crée ainsi la "bande élastique non-holonome". Bien que les métriques utilisées soient définies pour des robots ponctuels il obtient de bons résultats en minorant la taille des bulles utilisées.

Le principal inconvénient de cette méthode est que pour de nombreux systèmes non-holonomes complexes la plus courte distance entre une configuration et les obstacles n'est pas connue.

1.2.5 Planification de trajectoires à partir de données proximétriques

Bien que cette méthode soit présentée comme une méthode de planification de trajectoires, les techniques utilisées sont proches de celles que nous avons utilisées dans notre méthode d'évitement réactif. C'est pourquoi elle est décrite ici.

Cette méthode est décrite dans [Divelbiss 1997]. Les auteurs décrivent un algorithme pour trouver un chemin cinématiquement faisable pour un système non-holonome quelconque en présence d'obstacles.

La méthode transforme le problème en un système d'équations non-linéaires résolues de manière itérative par la méthode de Newton-Raphson. Les inconnues du système d'équations sont les commandes du système non-holonome considéré. Pour pouvoir résoudre le problème, ces commandes sont aproximées pas des séries de Fourier à un ordre donné. Les inconnues sont donc les coefficients de ces séries de Fourier.

Les obstacles sont pris en compte par des fonctions nulles lorsque la trajectoire est sans collision, positives sinon. Ces fonctions appartiennent au système non-linéaire résolu par la méthode de Newton-Raphson.

La principale différence avec notre méthode, présentée dans la section suivante 1.3, est que la nôtre utilise une trajectoire initiale que nous déformons en temps réel lors de l'exécution du mouvement. La méthode présentée semble difficile à utiliser dans ce contexte.

1.3 Notre approche et notre contribution

Notre approche se base sur une trajectoire initiale calculée par un planificateur utilisant des réseaux probabilistes, Move3D [Simeon 2001], à partir d'un plan de l'environnement préétabli.

Cette trajectoire est ensuite partagée entre deux tâches. L'une assure la progression du robot le long de celle-ci. L'autre la déforme en fonction des obstacles perçus par le robot afin d'éliminer les collisions tout en respectant les contraintes cinématiques.

Notre contribution se situe dans cette phase d'exécution. Dans ce mémoire, nous proposons une méthode générique permettant à un robot non-holonome de suivre une trajectoire planifiée sans collision avec son environnement.

L'atout majeur de notre approche de déformation de trajectoire est la généricité. Elle peut être appliquée aussi bien à un système holonome qu'à

un système non-holonome complexe (voir paragraphe 3.3.3) car elle s'appuie uniquement sur la connaissance des champs de vecteurs qui définissent le mouvement du système.

D'un point de vue utilisateur, il y a une grande similitude avec la bande élastique présentée précédemment. En effet, la trajectoire est perçue comme une entité qui se déforme sous l'action de "forces" créées par les obstacles. Mais contrairement à la bande élastique qui utilise un certain type de trajectoire et des métriques associées (les chemin de Reeds et Shepp par exemple) notre approche travaille dans l'espace de dimension infinie des trajectoires admissibles pour le système sans avoir besoin de métriques particulières.

II
Déformation de trajectoire

2.1 Présentation du problème

Dans ce chapitre, nous décrivons la méthode de déformation de trajectoire que nous avons développée. Une trajectoire initiale est déformée par perturbation des fonctions d'entrée du système qui définissent la trajectoire. A chaque itération, la perturbation choisie fait décroître un potentiel engendré par les obstacles tout en assurant que les bornes de l'intervalle de déformation restent constantes. La discrétisation du processus de déformation fait apparaître une dérive des contraintes non-holonomes. Ce phénomène indésirable est corrigé par un mécanisme de régulation à zéro des vitesses dans les directions interdites.

2.1.1 Système non-holonome et trajectoires

Pour un système dynamique $\sum$, la configuration $\mathbf{q} = (q_1, \ldots, q_n)$ est un vecteur de dimension n dans l'espace des configurations du système noté $\mathcal{C}$.

Une trajectoire $\mathbf{\Gamma}$, pour ce système, est une courbe dans l'espace des configurations définie sur un intervalle $I = [0, S] \subset \mathbb{R}$ par :

$$\begin{array}{rccc} \mathbf{\Gamma}: & I & \rightarrow & \mathcal{C} \\ & s & \mapsto & \mathbf{\Gamma}(s) \end{array}$$

Cette trajectoire $\mathbf{\Gamma}$ est obtenue en appliquant des commandes $u_i(s) \in \mathbb{R}$, $i = 1, 2, \ldots, k$, au système $\sum$.

FIG. 2.1 – représentation générale d'un système dynamique

La relation entre la trajectoire et les commandes d'un système peut être donnée par une équation différentielle entre $\mathbf{\Gamma}'(s)$ et $\mathbf{\Gamma}(s)$ et $\mathbf{u}(s)$. Ici $\mathbf{\Gamma}'(s)$ représente la dérivée du vecteur $\mathbf{\Gamma}(s)$ par rapport au paramètre s.

Un système dynamique non-holonome de dimension n est caractérisé par k champs de vecteurs $(k < n)$ $X_1(\mathbf{q}), \ldots, X_k(\mathbf{q})$. Pour chaque configuration $\mathbf{q}$, le vecteur vitesse $\mathbf{q}'$ admissible du système est une combinaison linéaire de $X_i(\mathbf{q})$.

Ainsi, une trajectoire $\mathbf{\Gamma}$ est faisable si et seulement s'il existe des fonctions $u_1, \ldots, u_k$ à valeurs dans $\mathbb{R}$ telles que

$$\forall s \in I,\ \mathbf{\Gamma}'(s) = \sum_{i=1}^{k} u_i(s)\mathbf{X_i}(\mathbf{\Gamma}(s)) \tag{2.1}$$

2.1.2 La déformation élémentaire de trajectoire

Nous avons vu que la trajectoire d'un système non-holonome est caractérisée par ses fonctions d'entrée. Pour déformer une trajectoire, il suffit donc de perturber ces entrées. Pour cela, nous définissons une famille de fonctions d'entrée indicées par un réel τ : $u_1(s,\tau) \ldots u_k(s,\tau)$ et nous établissons une relation entre la variation de ces fonctions d'entrée et la variation de la trajectoire correspondante.

On définit ainsi un faisceau de trajectoires Φ défini par une application de $I \times [0, +\infty[$ dans l'espace des configurations $\mathcal{C}$ du système par

$$\begin{array}{rcl} \Phi : I \times [0,+\infty[& \rightarrow & \mathcal{C} \\ (s,\tau) & \mapsto & \Phi(s,\tau) \end{array}$$

Pour toute valeur de τ, $\Phi(s,\tau)$ est une trajectoire du système. On la note $\mathbf{\Gamma}_\tau$.

$$\begin{array}{rcl} \forall \tau \in [0,+\infty[,\ \mathbf{\Gamma}_\tau : I & \rightarrow & \mathcal{C} \\ s & \mapsto & \mathbf{\Gamma}_\tau(s) = \Phi(s,\tau) \end{array}$$

La trajectoire $\mathbf{\Gamma}_0$ définie par $s \mapsto \mathbf{\Gamma}_0(s) = \Phi(s,0)$ est la trajectoire initiale du système et les trajectoires $\mathbf{\Gamma}_\tau$ sont les trajectoires déformées.

De la même manière que précédemment, on considère des trajectoires faisables qui vérifient donc l'égalité (2.1) qui devient :

$$\forall (s,\tau) \in I \times [0,+\infty[,\ \mathbf{\Gamma}_\tau{}'(s) = \sum_{i=1}^{k} u_i(s,\tau)\mathbf{X_i}(\mathbf{\Gamma}_\tau(s)) \tag{2.2}$$

Ainsi, une trajectoire faisable $\mathbf{\Gamma}_\tau$ du système est parfaitement caractérisée par la configuration initiale $\mathbf{\Gamma}_\tau(0)$ et les commandes appliquées au système $u_i(s,\tau)$.

Ce qui nous intéresse ici, c'est de connaître la "déformation" subie par une configuration d'une trajectoire faisable $\mathbf{\Gamma}_\tau$ du faisceau de trajectoire Φ lorsque l'on fait varier τ à s constant.

Pour cela, dérivons l'équation (2.2) par rapport à τ :

$$\frac{\partial^2 \Phi}{\partial s \partial \tau}(s,\tau) = \sum_{i=1}^{k} \frac{\partial u_i}{\partial \tau}(s,\tau)\mathbf{X_i}(\Phi(s,\tau)) + u_i(s,\tau)\frac{\partial \mathbf{X_i}}{\partial \mathbf{q}}(\Phi(s,\tau))\frac{\partial \Phi}{\partial \tau}(s,\tau) \tag{2.3}$$

En appelant "perturbations des entrées" le vecteur :

$$\mathbf{v}(s,\tau) = \frac{\partial \mathbf{u}}{\partial \tau}(s,\tau)$$

et "direction de déformation" le vecteur (voir figure 2.2) :

$$\eta(s,\tau) = \frac{\partial \Phi}{\partial \tau}(s,\tau)$$

l'équation (2.3), qui est le système linéarisé tangent par rapport au paramètre τ du système (2.2), est un système différentiel qui relie la direction de déformation $\eta(s,\tau)$ aux perturbations d'entrée $\mathbf{v}(s,\tau)$.

On peut alors la re-écrire sous la forme :

$$\eta'(s,\tau) = A(s,\tau)\eta(s,\tau) + B(s,\tau)\mathbf{v}(s,\tau) \tag{2.4}$$

où $A(s,\tau)$ est une matrice de dimension $n \times n$.

$$\begin{aligned} A(s,\tau) &= \sum_{i=1}^{k} u_i(s,\tau)\frac{\partial \mathbf{X_i}}{\partial \mathbf{q}}(\Phi(s,\tau)) \\ &= \sum_{i=1}^{k} u_i(s,\tau) \begin{pmatrix} \frac{\partial X_i^1}{\partial \mathbf{q}_1}(\Phi(s,\tau)) & \cdots & \frac{\partial X_i^1}{\partial \mathbf{q}_n}(\Phi(s,\tau)) \\ \vdots & & \\ \frac{\partial X_i^n}{\partial \mathbf{q}_1}(\Phi(s,\tau)) & \cdots & \frac{\partial X_i^n}{\partial \mathbf{q}_n}(\Phi(s,\tau)) \end{pmatrix} \end{aligned}$$

où X_i^k est la k-ième coordonnée du champ de vecteurs $\mathbf{X_i}$,

et $B(s,\tau)$ est une matrice de dimension $n \times k$ dont les colonnes sont les champs de vecteurs du système $\sum$:

$$B(s,\tau) = (\mathbf{X_1}(\Phi(s,\tau)) \cdots \mathbf{X_k}(\Phi(s,\tau)))$$

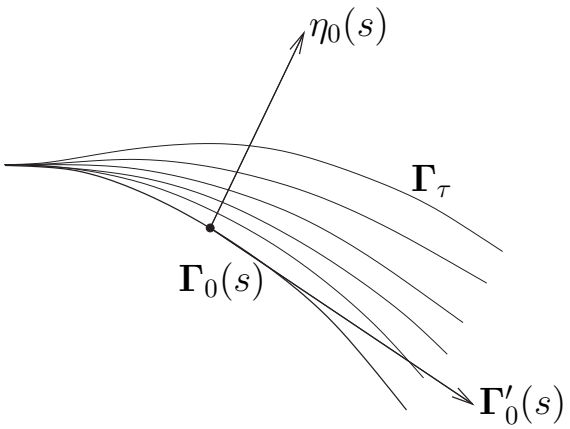

FIG. 2.2 – direction de déformation $\eta_\tau(s)$

Ainsi, connaissant la trajectoire $\mathbf{q}(s,\tau)$, les commandes $u_i(s,\tau)$, les perturbations d'entrée $\mathbf{v}(s,\tau)$ et la condition initiale $\eta_0 = \eta(0,\tau)$ on peut intégrer par rapport à s l'équation (2.4) pour connaître la direction de déformation $\eta(s,\tau)$.

$$\eta(s,\tau) = H(s,\tau)\left(\int_0^s H^{-1}(\iota,\tau)B(\iota,\tau)\mathbf{v}(\iota,\tau)d\iota + \eta_0\right) \tag{2.5}$$

où $H(s,\tau)$ est la matrice de dimension $n \times n$ qui satisfait :

$$\begin{aligned} H(0,\tau) &= I_n \\ H'(s,\tau) &= A(s,\tau)H(s,\tau) \end{aligned}$$

I_n est la matrice identité d'ordre n.

Pour alléger les notations et faciliter la compréhension on définit les fonctions d'une seule variable suivantes :

$$\mathbf{v}_\tau : s \mapsto \mathbf{v}(s,\tau)$$

et

$$\eta_\tau : s \mapsto \eta(s, \tau)$$

2.1.3 La déformation de trajectoire comme système asservi

Pour résumer le raisonnement précédent, on peut considérer la déformation de trajectoire comme la commande d'un système régi par les équations différentielles (2.2) et (2.4) asservi sur l'environnement.

La figure 2.3 schématise le système en boucle ouverte. L'état du système qui est aussi la sortie est une trajectoire $\mathbf{\Gamma}_\tau$ et l'entrée du système (la commande) est le vecteur $\mathbf{v}_\tau$. En effet, la perturbation des entrées $\mathbf{v}_\tau$ contrôle les variations de l'état du système, c'est à dire la déformation de la trajectoire courante.

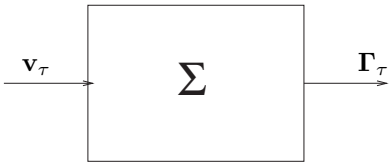

FIG. 2.3 – Le système dynamique en boucle ouverte. $\mathbf{v}_\tau \in C^\infty(I, \mathbb{R}^k)$, $\mathbf{\Gamma}_\tau \in C^\infty(I, \mathcal{C})$

La figure 2.4 schématise le système en boucle fermée.

La commande $\mathbf{v}_\tau$, calculée par le contrôleur, devra optimiser un critère relatif à la trajectoire qui restera toujours, en théorie, faisable par construction.

Bien que la méthode présentée ici puisse utiliser de nombreux critères, nous allons en utiliser un seul : l'éloignement de la trajectoire des obstacles.

Il nous reste donc à définir ce critère grâce auquel la régulation se fera et nous pourrons alors commencer la conception du contrôleur.

2.1.4 Champ de potentiel et produit scalaire

Dans notre cas, c'est le potentiel "obstacle" de la trajectoire qui devra être minimisé et c'est grâce aux obstacles perçus par le robot que les nouvelles commandes seront générées.

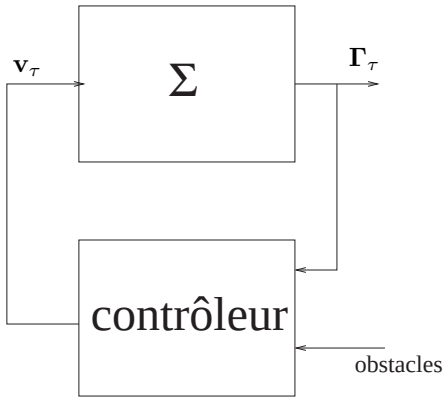

FIG. 2.4 – Le système dynamique en boucle fermée.

Le potentiel "obstacle" V d'une trajectoire $\mathbf{\Gamma}_\tau$ est défini en intégrant, le long de la trajectoire, un champ de potentiel U défini dans l'espace des configurations :

$$\begin{aligned} V : \mathbb{R} &\rightarrow \mathbb{R} \\ \tau &\mapsto V(\tau) = \int_0^S U(\mathbf{\Gamma}_\tau(s))ds \end{aligned}$$

Le potentiel U croît quand le robot s'approche des obstacles et décroît lorsqu'il s'en éloigne. Ainsi, une trajectoire qui passe près des obstacles a un potentiel élevé alors qu'une trajectoire éloignée des obstacles a un potentiel bas.

La variation du potentiel d'une trajectoire d'un faisceau de trajectoire par rapport à τ est :

$$\frac{dV}{d\tau}(\tau) = \int_0^S \frac{\partial U}{\partial \mathbf{q}}(\mathbf{\Gamma}_\tau(s))^T \eta_\tau(s) ds \tag{2.6}$$

Le contrôleur doit donc, pour faire diminuer le potentiel d'une trajectoire, déterminer $\mathbf{v}_\tau$ de façon à ce que la variation du potentiel soit négative.

Remarquons, avant de continuer, que l'espace dans lequel on cherche $\mathbf{v}_\tau$, la commande de la déformation, est un espace de dimension infinie, noté $C^\infty(I, \mathbb{R}^k)$ et l'espace des déformations est aussi un espace de dimension infinie noté $C^\infty(I, \mathbb{R}^n)$ dans lesquels le produit scalaire L_2 est défini par :

$$(f \mid g)_{L^2} = \int_0^S f(s)^T g(s) ds \tag{2.7}$$

Avec cette définition, la variation du potentiel d'une trajectoire s'écrit :

$$\frac{dV}{d\tau}(\tau) = \left(\frac{\partial U}{\partial \mathbf{q}}(\mathbf{\Gamma}_\tau(s)) \mid \eta_\tau(s) \right)_{L^2} \tag{2.8}$$

Ainsi, à norme L^2 constante, la direction de déformation qui minimise la variation du potentiel de la trajectoire est :

$$\eta_\tau(s) = -\frac{\partial U}{\partial \mathbf{q}}(\mathbf{\Gamma}_\tau(s)) \tag{2.9}$$

Malheureusement, cette direction de déformation n'est, en général, pas admissible, c'est à dire qu'elle n'est pas solution de l'équation (2.4). Si on appliquait cette déformation, les contraintes non-holonomes ne seraient plus respectées.

Une méthode pour obtenir une direction admissible serait de projeter orthogonalement cette direction sur l'espace des directions de déformation admissibles. Mais là encore, ce n'est pas possible. En effet, l'espace des directions de déformation admissibles (c'est à dire qui vérifient l'équation (2.4)) est de dimension infinie et la projection orthogonale n'existe pas nécessairement.

Pour résoudre ce problème, nous allons restreindre l'espace des perturbations d'entrée à un espace de dimension finie sur lequel la projection orthogonale est définie.

2.1.5 Résumé du problème

L'équation différentielle (2.4) nous donne une relation entre la direction de déformation η et la perturbation d'entrée $\mathbf{v}$.

$$\eta'(s,\tau) = A(s,\tau)\eta(s,\tau) + B(s,\tau)\mathbf{v}(s,\tau)$$

De plus, nous devons imposer que les configurations initiale et finale de la trajectoire déformée ne soient pas modifiées. En effet, la trajectoire déformée peut être une trajectoire entière ou une portion de trajectoire. Dans le premier cas, la configuration initiale correspond donc à la position de départ du robot et la configuration finale correspond au but du robot. On ne peut donc pas les modifier. Dans le second cas, les configurations initiale et finale appartiennent à la fois à la trajectoire entière et à la trajectoire déformée et ne doivent pas être modifiées pour assurer la continuité de la trajectoire entière.

Nous devons donc imposer :

$$\begin{aligned} \forall \tau \in [0, +\infty[, \quad & \Phi(0,\tau) = \Phi(0,0) \\ & \Phi(S,\tau) = \Phi(S,0) \end{aligned}$$

Nous verrons dans le paragraphe 2.2.4 que ces conditions se traduisent par :

$$\eta(0,\tau) = 0_{\mathbb{R}^n}$$
$$\eta(S,\tau) = 0_{\mathbb{R}^n}$$

Ce qui nous donne un système différentiel linéaire à coefficients variables avec des contraintes aux extrémités difficile à résoudre.

De plus la déformation doit faire décroître le potentiel obstacle de la trajectoire. La déformation doit assurer l'inégalité suivante :

$$\left(\frac{\partial U}{\partial \mathbf{q}}(\mathbf{\Gamma}_\tau(s)) \mid \eta_\tau(s)\right)_{L^2} < 0$$

Ce problème est excessivement difficile à résoudre. L'espace des fonctions η et $\mathbf{v}$ est un espace de dimension infinie et il n'existe aucune méthode permettant de le résoudre.

Dans la section suivante, nous proposons un algorithme qui, avec une simplification importante mais réaliste, nous permet de déterminer une déformation satisfaisant nos exigences.

2.2 Résolution : l'algorithme de déformation

2.2.1 Les étapes de la déformation

Avant d'aller plus loin dans la construction de notre contrôleur de déformation de trajectoire non-holonome, décrivons grossièrement les différentes étapes nécessaires.

Le contrôleur, présenté sur la figure 2.4, doit, à partir des obstacles perçus par le robot, calculer les commandes $\mathbf{v}$ pour éloigner la trajectoire des obstacles (en minimisant son potentiel) tout en restant faisable.

Ceci se fait en trois étapes :

1. **Calcul de la commande**

   à partir des obstacles perçus par le robot, on calcule un vecteur de commandes $\mathbf{v}(s,\tau)$ afin d'éloigner la trajectoire des obstacles,

2. **Calcul de la direction de déformation**

   on intègre l'équation (2.4) pour obtenir la direction de déformation $\eta(s,\tau)$,

3. **Application de la déformation**
   on approxime la trajectoire déformée par :

$$\Phi(s, \tau + \Delta\tau) = \Phi(s, \tau) + \Delta\tau \; \eta(s, \tau) \tag{2.10}$$

   avec les notations simplifiées $\mathbf{\Gamma}_\tau$ et η_τ cette équation s'écrit :

$$\mathbf{\Gamma}_{\tau+\Delta\tau}(s) = \mathbf{\Gamma}_\tau(s) + \Delta\tau \; \eta_\tau(s)$$

La détermination de la trajectoire déformée par une approximation au premier ordre a été préférée au calcul de la trajectoire déformée par intégration du système $\sum$ après perturbation des commandes. Les conditions aux limites peuvent ainsi être respectées exactement. Ce choix se répercute sur le respect des contraintes non-holonomes. Ce sera abordé en détail dans le paragraphe 2.2.6 "Dérive des contraintes non-holonomes".

Les paragraphes suivants décrivent la première étape "Calcul de la commande".

2.2.2 L'espace de dimension finie des perturbations

Comme indiqué dans la section 2.1.2, les perturbations d'entrée appartiennent à un espace de dimension infinie : l'espace des fonctions vectorielles de dimension k définies sur un intervalle I noté $C^\infty(I, \mathbb{R}^k)$. Ceci rend la résolution du problème difficile.

Afin de faciliter la recherche des perturbations d'entrée amenant la trajectoire à s'éloigner des obstacles, nous allons restreindre $\mathbf{v}(s, \tau)$ à un sous-espace de dimension finie engendré par un ensemble linéairement indépendant $(\mathbf{e_1}(s), \ldots, \mathbf{e_p}(s)), p > n$ de fonctions vectorielles de dimension k, défini sur l'intervalle I :

$$\begin{aligned} \mathbf{e_i} : I &\rightarrow \mathbb{R}^k \\ s &\mapsto \mathbf{e_i}(s) \end{aligned}$$

Ainsi, le vecteur des perturbations d'entrée est défini comme une combinaison linéaire des $\mathbf{e_i}$, $\lambda_i \in \mathbb{R}$:

$$\begin{aligned} \mathbf{v}_\tau : I &\rightarrow \mathbb{R}^k \\ s &\mapsto \mathbf{v}_\tau(s) = \sum_{i=1}^{p} \lambda_i(\tau)\mathbf{e_i}(s) \end{aligned} \tag{2.11}$$

On appelle $\mathbf{e_i}$ la i-ème perturbation élémentaire et $\mathbf{E_i}$ la direction de déformation qui en résulte et $\mathbf{E} = (\mathbf{E_1}|\ldots|\mathbf{E_p})$ la matrice dont les colonnes sont les vecteurs $\mathbf{E_i}$. On note $\lambda = (\lambda_1, \ldots, \lambda_p)$ le vecteur de coordonnées de $\mathbf{v}_\tau$.

$\mathbf{E_i}$ est solution du système différentiel (2.4). En prenant une condition initiale nulle on a :

$$\begin{aligned} \mathbf{E_i}(0,\tau) &= 0 && (2.12)\\ \mathbf{E_i}'(s,\tau) &= A(s,\tau)\mathbf{E_i}(s,\tau) + B(s,\tau)\mathbf{e_i}(s) && (2.13) \end{aligned}$$

La linéarité du système différentiel (2.4) implique que la déformation résultante η est une combinaison linéaire des déformations élémentaires $\mathbf{E_i}$ dûes aux perturbations élémentaires $\mathbf{e_i}$. En prenant comme condition initiale $\eta_0 = \eta(0,\tau) = 0_{\mathbb{R}^n}$ on a :

$$\eta(s,\tau) = \sum_{i=1}^{p} \lambda_i(\tau)\mathbf{E_i}(s,\tau) \qquad (2.14)$$

En effet, une solution de l'équation (2.4) s'écrit :

$$\eta(s,\tau) = H(s,\tau)\int_0^s H^{-1}(\iota,\tau)B(\iota,\tau)\mathbf{v}(\iota,\tau)d\iota \qquad (2.15)$$

où $H(s,\tau)$ est une matrice $n \times n$ dont les valeurs satisfont :

$$\begin{aligned} H(0,\tau) &= I_n\\ H'(s,\tau) &= A(s,\tau)H(s,\tau) \end{aligned}$$

I_n est la matrice identité d'ordre n.
Donc, en remplaçant $\mathbf{v}$ par son expression (2.11) dans (2.14) on obtient :

$$\begin{aligned} \eta(s,\tau) &= H(s,\tau)\int_0^s H^{-1}(\iota,\tau)B(\iota,\tau)\mathbf{v}(\iota,\tau)d\iota\\ &= H(s,\tau)\int_0^s H^{-1}(\iota,\tau)B(\iota,\tau)\sum_{i=1}^{p}\lambda_i(\tau)\mathbf{e_i}(s)d\iota\\ &= \sum_{i=1}^{p}\lambda_i(\tau)H(s,\tau)\int_0^s H^{-1}(\iota,\tau)B(\iota,\tau)\mathbf{e_i}(s)d\iota\\ &= \sum_{i=1}^{p}\lambda_i(\tau)\mathbf{E_i}(s,\tau) \end{aligned}$$

Les directions élémentaires de déformation $\mathbf{E_i}$ sont parfaitement connues (les matrices A et B ne dépendent que de la trajectoire courante $\mathbf{\Gamma}_\tau$). Le rôle du contrôleur est donc maintenant de déterminer les coefficients λ_i.

2.2.3 Champ de potentiel et déformation élémentaire

En injectant l'expression (2.14) de η en fonction des déformations élémentaires $\mathbf{E_i}$ dans l'expression de la variation du potentiel de la trajectoire (2.6) on obtient :

$$\frac{dV}{d\tau}(\tau) = \sum_{i=1}^{p} \lambda_i(\tau) \int_0^S \frac{\partial U}{\partial \mathbf{q}}(\mathbf{\Gamma}_\tau(s))^T \mathbf{E_i}(s,\tau) ds \tag{2.16}$$

Cette expression est une combinaison linéaire des variations de potentiel μ_i, définies ci-après, induites par les directions de déformation $\mathbf{E_i}$.

En posant

$$\mu_i(\tau) = \int_0^S \frac{\partial U}{\partial \mathbf{q}}(\mathbf{\Gamma}_\tau(s))^T \mathbf{E_i}(s,\tau) ds \tag{2.17}$$

l'expression (2.16) peut s'écrire alors :

$$\begin{aligned} \frac{dV}{d\tau}(\tau) &= \sum_{i=1}^{p} \lambda_i(\tau)\mu_i(\tau) \\ &= \mu^T(\tau)\lambda(\tau) \end{aligned}$$

L'objectif est de faire décroître le potentiel V de la trajectoire. On doit donc avoir $\frac{dV}{d\tau}(\tau) < 0$. Pour cela on peut choisir :

$$\lambda_i(\tau) = -\mu_i(\tau) \tag{2.18}$$

ainsi

$$\frac{dV}{d\tau}(\tau) = -\sum_{i=1}^{p} \mu_i^2(\tau) \leq 0 \tag{2.19}$$

On appelle $\lambda^{\mathbf{0}}$ cette solution.

$$\lambda^{\mathbf{0}}(\tau) = (-\mu_1(\tau), \ldots, -\mu_p(\tau)) \tag{2.20}$$

Avec ce choix, la direction de déformation

$$\eta^{\mathbf{0}} = \mathbf{E}\lambda^{\mathbf{0}}$$

fait décroître le potentiel de la trajectoire et la trajectoire reste faisable.

Rien ne prouve cependant que cette direction de déformation vérifie les conditions aux limites présentées en suivant.

2.2.4 Prise en compte des conditions aux limites

Pour que les configurations initiale et finale ne soient pas modifiées, nous devons imposer que :

$$\begin{aligned} \forall \tau \in [0, +\infty[, \quad & \Phi(0,\tau) = \Phi(0,0) \\ & \Phi(S,\tau) = \Phi(S,0) \end{aligned}$$

En utilisant l'expression (2.10) ces conditions aux limites sur Φ se traduisent par des conditions aux limites sur la direction de déformation η :

$$\forall \tau \in [0, +\infty[, \quad \eta(0,\tau) = 0_{\mathbb{R}^n} \tag{2.21}$$

$$\eta(S,\tau) = 0_{\mathbb{R}^n} \tag{2.22}$$

La première contrainte (2.21) est toujours satisfaite par le choix des conditions initiales (2.13) et l'expression de η (2.14).

$$\begin{aligned} \eta(0,\tau) &= \sum_{i=1}^{p} \lambda_i(\tau) \mathbf{E_i}(0,\tau) \\ &= 0_{\mathbb{R}^n} \end{aligned}$$

La deuxième contrainte (2.22) est une contrainte linéaire par rapport aux coefficients λ_i. Le vecteur λ doit satisfaire l'équation suivante :

$$L\lambda = 0 \tag{2.23}$$

où L est une matrice de dimension $n \times p$ dont les colonnes sont les $\mathbf{E_i}(S,\tau)$.

$$L = (\mathbf{E_1}(S,\tau), \ldots, \mathbf{E_p}(S,\tau))$$

Cette équation (2.23) définit un sous-espace vectoriel solution sur lequel nous allons projeter le vecteur λ^0. Nous appellerons $\bar{\lambda}^0 = (\bar{\lambda}^0_1, \ldots, \bar{\lambda}^0_p)$ ce vecteur.

$$\bar{\lambda}^0 = (I_p - L^+ L)\lambda^0 \tag{2.24}$$

$L^+ = L^T(LL^T)^{-1}$ est la pseudo inverse de L ($LL^+L = L$).
On peut vérifier que la direction de déformation donnée par

$$\eta(s,\tau) = \sum_{i=1}^{p} \bar{\lambda}_i^0(\tau)\mathbf{E_i}(s,\tau) \tag{2.25}$$

fait décroître le potentiel de la trajectoire.
En effet,

$$\begin{aligned} \frac{dV}{d\tau}(\tau) &= \mu^T\bar{\lambda}^0 \\ &= -\mu^T(I_p - L^+L)\mu \\ &= -\mu^T\mu + \mu^T L^+ L\mu \end{aligned}$$

et par construction de la matrice L^+, on a $\mu^T\mu \geq \mu^T L^+ L\mu$ et donc :

$$\frac{dV}{d\tau}(\tau) = -\mu^T\mu + \mu^T L^+ L\mu \leq 0$$

Ainsi, nous avons trouvé une direction de déformation admissible qui respecte les conditions aux limites, on la note $\bar{\eta}^0$:

$$\begin{aligned} \bar{\eta}^0(s,\tau) &= \sum_{i=1}^{p} \bar{\lambda}_i^0(\tau)\mathbf{E_i}(s,\tau) \\ &= \mathbf{E}\bar{\lambda}^0 \end{aligned}$$

Nous allons voir dans le paragraphe suivant que cette solution peut être améliorée.

2.2.5 Produit scalaire et base orthonormée

Énoncé du problème

Dans cet algorithme de déformation, nous utilisons une approximation au premier ordre de la trajectoire déformée (2.10). Dans cette approximation, le terme $\Delta\tau\eta(s,\tau)$ doit être petit, c'est à dire

$$\Delta\tau\|\eta(s,\tau)\|_\infty \leq \eta_{max}$$

où

$$\|\eta(s,\tau)\|_\infty = \max_{s\in I}\|\eta(s,\tau)\|$$

et η_{max} est un réel positif petit.

Le choix de $\overline{\lambda}^0$ utilisé précédemment n'est pas optimal du point de vue de la norme infinie $\|.\|_\infty$. De plus, l'application de ce choix s'est révélée peu efficace expérimentalement.

L'objectif à atteindre est de faire décroître le potentiel de la trajectoire au maximum pour $\|\eta\|_\infty$ constant.

La valeur optimale de λ devrait donc minimiser :

$$\frac{1}{\|\eta\|_\infty}\frac{dV}{d\tau} = \frac{\sum_{i=1}^{p}\mu_i\lambda_i}{\|\sum_{i=1}^{p}\lambda_i\mathbf{E_i}\|_\infty}$$

Cette valeur de λ est très difficile à déterminer.

Pour se rapprocher de la valeur optimale, on calcule λ de façon à minimiser :

$$\frac{1}{\|\eta\|_{L^2}}\frac{dV}{d\tau} = \frac{\sum_{i=1}^{p}\mu_i\lambda_i}{\|\sum_{i=1}^{p}\lambda_i\mathbf{E_i}\|_{L^2}} \tag{2.26}$$

Ce second problème est plus facile à résoudre comme nous allons le voir dans le paragraphe suivant. Sa pertinence repose sur l'hypothèse d'une corrélation forte entre la norme infinie $\|\ \|_\infty$ et la norme $\|\ \|_{L^2}$, hypothèse raisonnable étant données les fonctions que nous utilisons.

Résolution du problème

Pour trouver un vecteur λ qui minime l'expression ci-dessus, nous allons travailler dans une base orthonormée (pour le produit scalaire L^2) $\mathcal{F} = \{\mathbf{F_1}, \ldots, \mathbf{F_p}\}$ de l'espace engendré par la base $\mathcal{E} = \{\mathbf{E_1}, \ldots, \mathbf{E_p}\}$. Nous allons donc reprendre les opérations effectuées dans les chapitres 2.2.2 puis 2.2.3 et enfin 2.2.4 dans cette base orthonormée $\mathcal{F}$. Ainsi nous allons obtenir la solution optimale de notre problème.

Base orthonormée

On construit la base $\mathcal{F}$ à partir des vecteurs de la base $\mathcal{E} = \{\mathbf{E_1}, \ldots, \mathbf{E_p}\}$ en utilisant le procédé d'orthonormalisation de Gram-Schmidt. On appelle P la matrice de dimension $p \times p$ de changement de base ($\mathbf{F_i} = \sum_{j=1}^{p} P_{ji}\mathbf{E_j}$).

Expression de η

Dans cette base $\mathcal{F}$, on reprend les calculs présentés au paragraphe 2.2.2 et on exprime ainsi la direction de déformation η comme une combinaison linéaire des directions de déformation élémentaires orthonormalisées $\mathbf{F_i}$:

$$\eta(s,\tau) = \sum_{i=1}^{p} \rho_i(\tau)\mathbf{F_i}(s,\tau) \tag{2.27}$$

Variation du potentiel de la trajectoire

On reprend ici les calculs présentés dans la section 2.2.3. On injecte l'expression (2.27) de η dans l'expression (2.6) de la variation du potentiel de la trajectoire par rapport à τ.

$$\begin{aligned}\frac{dV}{d\tau}(\tau) &= \int_0^S \frac{\partial U}{\partial \mathbf{q}}(\mathbf{\Gamma}_\tau(s))^T \eta(s,\tau) ds \\ &= \sum_{i=1}^{p} \rho_i(\tau) \int_0^S \frac{\partial U}{\partial \mathbf{q}}(\mathbf{\Gamma}_\tau(s))^T \mathbf{F_i}(s,\tau) ds\end{aligned}$$

En posant

$$\begin{aligned}\mu_i^\perp(\tau) &= \int_0^S \frac{\partial U}{\partial \mathbf{q}}(\mathbf{\Gamma}_\tau(s))^T \mathbf{F_i}(s,\tau) ds \\ &= \left(\frac{\partial U}{\partial \mathbf{q}}(\mathbf{\Gamma}_\tau(s)) \mid \mathbf{F_i}\right)_{L^2}\end{aligned}$$

on a

$$\frac{dV}{d\tau}(\tau) = \sum_{i=1}^{p} \rho_i(\tau)\mu_i^\perp(\tau)$$

On reconnaît là le produit scalaire L^2 entre le vecteur $\eta(s,\tau)$ (2.27) et le vecteur $\mu^\perp(s,\tau) = \sum_{i=1}^{p} \mu_i^\perp(\tau)\mathbf{F_i}(s,\tau)$.

$$\frac{dV}{d\tau}(\tau) = \big(\eta(s,\tau) \mid \mu^\perp(s,\tau)\big)_{L^2}$$

On peut aussi remarquer que μ est la projection orthogonale de $\frac{\partial U}{\partial \mathbf{q}}(\mathbf{\Gamma}_\tau(s))$ sur le sous-espace des déformations admissibles (l'espace de η).

Ainsi pour minimiser la variation du potentiel de la trajectoire, nous devons prendre $\eta = -\mu^\perp$, soit :

$$\rho_i^\perp(\tau) = -\left(\frac{\partial U}{\partial \mathbf{q}}(\mathbf{\Gamma}_\tau(s)) \mid \mathbf{F_i}(s,\tau)\right)_{L^2} \tag{2.28}$$

Ainsi, la direction de déformation admissible qui minimise le potentiel de la trajectoire est donnée par :

$$\eta^{\perp}(s,\tau) = \sum_{i=1}^{p} \rho_i^{\perp}(\tau)\mathbf{F_i}(s,\tau) \tag{2.29}$$

Conditions aux limites exprimées dans la base $\mathcal{F}$

La direction de déformation $\eta^{\perp}$ ne satisfait pas les conditions aux limites présentées dans la section 2.2.4 et rappelées ici :

$$\forall \tau \in [0,+\infty[, \quad \eta(0,\tau) = 0_{\mathbb{R}^n} \tag{2.30}$$
$$\eta(S,\tau) = 0_{\mathbb{R}^n} \tag{2.31}$$

La première condition (2.30) est toujours satisfaite car $\mathbf{F_i}(0,\tau) = 0$.

Pour satisfaire la deuxième condition (2.31) nous devons projeter orthogonalement dans la base $\mathcal{F}$ la direction de déformation $\eta^{\perp}$ (expression 2.29) sur le sous-espace vectoriel solution de (2.31). Le vecteur de coordonnée $\bar{\rho}^{\perp}$ de cette projection, notée $\bar{\eta}^{\perp}$, dans la base $\mathcal{F}$ est alors :

$$\bar{\rho}^{\perp} = \left(I_p - L^{\perp +} L^{\perp}\right) \rho^{\perp} \tag{2.32}$$

où $L^{\perp}$ est une matrice de dimension $n \times p$ dont les colonnes sont les $\mathbf{F_i}(S,\tau)$.

$$L^{\perp} = (\mathbf{F_1}(S,\tau), \ldots, \mathbf{F_p}(S,\tau))$$

Ainsi, la direction de déformation admissible qui minimise le potentiel de la trajectoire et qui assure le respect des conditions aux limites est donnée par :

$$\bar{\eta}^{\perp}(s,\tau) = \sum_{i=1}^{p} \bar{\rho}_i^{\perp}(\tau)\mathbf{F_i}(s,\tau) \tag{2.33}$$

Expression dans la base $\mathcal{E}$

Nous pourrions nous arrêter à ce stade. En effet, nous avons obtenu une valeur optimale du vecteur de coordonnées $\bar{\rho}^{\perp}$ pour le critère (2.26). Mais pour la calculer, nous devons déterminer les vecteurs $\mathbf{F_i}$ de la base orthonormée $\mathcal{F}$ ce qui est coûteux et comme nous allons le voir inutile. Nous allons montrer dans le paragraphe suivant que l'on peut exprimer cette valeur optimale dans la base $\mathcal{E}$ et en plus en fonction de λ_0 dont l'expression (2.20) a été établie au chapitre 2.2.3.

Nous avons

$$\mathbf{F_l}(s,\tau) = \sum_{i=1}^{p} P_{il}\mathbf{E_i}(s,\tau)$$

donc

$$\begin{aligned}
\rho_i^{\perp}(\tau) &= -\left(\frac{\partial U}{\partial \mathbf{q}}(\mathbf{\Gamma}_\tau(s)) \mid \mathbf{F_i}(s,\tau)\right)_{L^2} && (2.34)\\
&= -\sum_{i=1}^{p} P_{il}\left(\frac{\partial U}{\partial \mathbf{q}}(\mathbf{\Gamma}_\tau(s)) \mid \mathbf{E_i}(s,\tau)\right)_{L^2} && (2.35)\\
&= \sum_{i=1}^{p} P_{il}\lambda_i^0(\tau) && (2.36)
\end{aligned}$$

soit

$$\rho^{\perp}(\tau) = P^T\lambda^0(\tau)$$

Ainsi l'expression (2.32) de $\bar{\rho}^{\perp}$ s'écrit :

$$\bar{\rho}^{\perp} = \left(I_p - L^{\perp +}L^{\perp}\right)P^T\lambda^0$$

De plus on sait que $L^{\perp} = LP$. Ainsi l'expression du vecteur coordonnées $\bar{\rho}^{\perp}$ devient :

$$\bar{\rho}^{\perp} = \left(I_p - (LP)^{+}LP\right)P^T\lambda^0$$

Il vient alors l'expression du vecteur coordonnées $\bar{\lambda}^{\perp}$ de $\bar{\eta}^{\perp}$ exprimé dans la base $\mathcal{E}$:

$$\begin{aligned}
\bar{\lambda}^{\perp} &= P\bar{\rho}^{\perp}\\
&= P\left(I_p - (LP)^{+}LP\right)P^T\lambda^0\\
&= \left(I_p - P(LP)^{+}L\right)PP^T\lambda^0
\end{aligned}$$

Solution du problème

La direction de déformation qui respecte les conditions aux limites et fait décroître le potentiel de la trajectoire de manière optimale du point de vue de la norme L^2 est donnée par :

$$\bar{\eta}^{\perp}(s,\tau) = \sum_{i=1}^{p} \bar{\lambda}_i^{\perp}(\tau)\mathbf{E_i}(s,\tau) \qquad (2.37)$$

2.2.6 Dérive des contraintes non-holonomes

L'approximation (2.10) que nous avons faite pour approcher la trajectoire déformée a un effet secondaire. Après quelques itérations, les contraintes non-holonomes ne sont plus satisfaites, l'équation (2.1) n'est donc plus vérifiée.

Un système étendu

La vitesse du système $\mathbf{\Gamma}'(s)$ n'est plus une combinaison linéaire des champs de vecteurs $\mathbf{X_i}$ du système. On détermine donc $n-k$ champs de vecteurs additionnels pour former une base de $\mathbb{R}^n$.

$$\forall s \in I \quad \mathbf{\Gamma}'(s) = \sum_{i=1}^{n} u_i(s)\mathbf{X_i}(\mathbf{\Gamma}(s)) \tag{2.38}$$

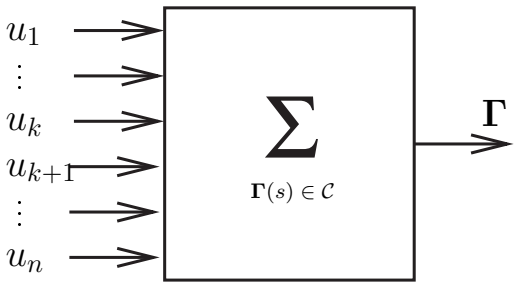

FIG. 2.5 – représentation générale d'un système dynamique étendu

Pour contrecarrer cette dérive des contraintes non-holonomes et ainsi conserver une trajectoire faisable, nous devons maintenir autour de 0 les composantes u_i pour $k+1 \leq i \leq n$.

Dans les paragraphes précédents, nous perturbions les commandes correspondant aux champs de vecteurs du système, nous allons maintenant également perturber les entrées correspondant aux champs de vecteurs $\mathbf{X_{k+1}}, \ldots, \mathbf{X_n}$.

L'équation différentielle (2.4) devient, pour notre système étendu :

$$\eta'(s,\tau) = \bar{A}(s,\tau)\eta(s,\tau) + \bar{B}(s,\tau)\bar{\mathbf{v}}(s,\tau) \tag{2.39}$$

où $\bar{A}(s,\tau)$ et $\bar{B}(s,\tau)$ sont des matrices $n \times n$:

$$\bar{A}(s,\tau) = \sum_{i=1}^{n} u_i(s,\tau)\frac{\partial \mathbf{X_i}}{\partial \mathbf{q}}(\mathbf{\Gamma}_\tau(s)) \tag{2.40}$$

et

$$\bar{B}(s,\tau) = (B(s,\tau)B^{\perp}(s,\tau)) \tag{2.41}$$

avec $B^{\perp}(s,\tau) = (\mathbf{X_{k+1}}(\mathbf{\Gamma}_\tau(s)) \dots \mathbf{X_n}(\mathbf{\Gamma}_\tau(s))$ la matrice dont les colonnes sont les champs de vecteurs additionnels.

On peut faire apparaître, de manière distincte, les commandes sur les champs de vecteurs du système initial $\mathbf{v}$ et les commandes sur les champs de vecteurs additionnels $\mathbf{v}^{\perp}$:

$$\eta'(s,\tau) = \bar{A}(s,\tau)\eta(s,\tau) + B(s,\tau)\mathbf{v}(s,\tau) + B^{\perp}(s,\tau)\mathbf{v}^{\perp}(s,\tau) \tag{2.42}$$

où $\mathbf{v}^{\perp} = (v_{k+1}(s,\tau), \dots, v_n(s,\tau))$

Correction de la dérive non-holonome

Une régulation proportionnelle suffit :

$$\forall i \in \{k+1, \dots, n\},\ v_i(s,\tau) = \frac{\partial u_i}{\partial \tau}(s,\tau) = -\alpha u_i(s,\tau)$$

où α est un réel positif.

Ainsi les commandes sur les champs de vecteurs additionnels tendent exponentiellement vers 0 lorsque τ augmente,

$$\forall s \in I,\ u_i(s,\tau) = e^{-\alpha\tau} u_i(s,0)$$

On appelle $\eta_{\mathbf{1}}$ la déformation résultante donnée par l'équation différentielle (2.42) :

$$\begin{aligned} \eta'_{\mathbf{1}}(s,\tau) &= \bar{A}(s,\tau)\eta_{\mathbf{1}}(s,\tau) + B^{\perp}(s,\tau)\mathbf{v}^{\perp}(s,\tau) \\ \eta_{\mathbf{1}}(0,\tau) &= 0 \end{aligned} \tag{2.43}$$

Déformation due aux obstacles

La direction de déformation due aux obstacles est notée η_2.

La direction de déformation calculée comme expliqué dans la section 2.2.5 est notée $\eta_2^{\perp}$.

$$\eta_2^{\perp}(s,\tau) = \sum_{i=1}^{p} \lambda_i^{\perp}(\tau)\mathbf{E_i}(s,\tau)$$

où les $\mathbf{E_i}$ sont solutions du système :

$$\begin{aligned} \mathbf{E}'_{\mathbf{i}}(s,\tau) &= \bar{A}(s,\tau)\mathbf{E}_{\mathbf{i}}(s,\tau) + B(s,\tau)\mathbf{e}_{\mathbf{i}}(s) \\ \mathbf{E}_{\mathbf{i}}(0,\tau) &= 0 \end{aligned} \tag{2.44}$$

et le vecteur $\lambda^{\perp}(\tau)$ est donné par $\lambda^{\perp} = PP^T\lambda^0$.

Conditions aux limites

Comme le système (2.42) est linéaire, la déformation obtenue par $\mathbf{v} + \mathbf{v}^{\perp}$ est égale à la somme de η_1 et η_2.

Cette déformation doit respecter les mêmes contraintes que celles exprimées dans la section 2.2.4 :

$$\begin{aligned} \eta_1(0,\tau) + \eta_2(0,\tau) &= 0 \\ \eta_1(S,\tau) + \eta_2(S,\tau) &= 0 \end{aligned}$$

La première condition est toujours satisfaite.

La deuxième est une contrainte affine par rapport au paramètre λ :

$$\begin{aligned} \eta_2(S,\tau) &= -\eta_1(S,\tau) \\ L\lambda &= -\eta_1(S,\tau) \end{aligned}$$

Le vecteur $\lambda^{\perp}$ ne satisfait pas ces conditions, on projette orthogonalement alors ce vecteur sur l'espace des solutions :

$$\bar{\lambda}^{\perp} = -P(LP)^{+}\eta_1(S,\tau) + (I_p - P(LP)^{+}L)\lambda^{\perp}$$

Ainsi, la direction de déformation

$$\bar{\eta}^{\perp}(s,\tau) = \sum_{i=1}^{p} \bar{\lambda}_i^{\perp}(\tau)\mathbf{E}_{\mathbf{i}}(s,\tau) + \eta_{\mathbf{1}}(s,\tau)$$

satisfait les conditions aux limites, ramène les composantes $v_{k+1}, \ldots, v_n$ vers 0 et éloigne la trajectoire des obstacles.

2.2.7 Résumé : l'algorithme de déformation de trajectoire non-holonome

Grâce à la restriction de l'espace des perturbations d'entrées à un sous-espace engendré par des fonctions élémentaires, nous avons pu résoudre le

problème présenté dans la section 2.1.5. Nous résumons, ici, les données nécessaires et les étapes de l'algorithme de déformation de trajectoire.

Les données nécessaires sont : la trajectoire $\mathbf{\Gamma}_\tau$, la variation du potentiel le long de la trajectoire $\frac{\partial U}{\partial \mathbf{q}}$, un jeu de fonctions élémentaires de perturbations $\mathbf{e_i}$.

À partir de ces données et en suivant les étapes résumées ci-dessous, nous obtenons une nouvelle trajectoire dont le potentiel obstacle a diminué (donc qui s'est éloigné des obstacles) et pour laquelle les contraintes non-holonomes sont quasi satisfaites et tendent à l'être de plus en plus à chaque itération.

1. **Préparation des calculs**
   calculer $\bar{A}(s,\tau)$ et $\bar{B}(s,\tau)$ pour $s \in I$,
2. **Correction des dérives non-holonomes**
   - Pour $i \in \{k+1,\dots,n\}$ calculer $u_i(s,\tau)$,
   - calculer la correction $v_i(s,\tau) = -\alpha u_i(s,\tau)$,
   - calculer la déformation résultante $\eta_1(s,\tau)$ en utilisant (2.43),
3. **Déformation élémentaire**
   - Pour $i \in \{1,\dots,p\}$ calculer les $\mathbf{E_i}$ en intégrant (2.44),
   - Pour $s \in I$, calculer $\frac{\partial U}{\partial \mathbf{q}}(\mathbf{\Gamma}_\tau(s))$,
   - Pour $i \in \{1,\dots,p\}$ calculer $\lambda_i^0 = -\int_0^S \frac{\partial U}{\partial \mathbf{q}}(\mathbf{\Gamma}_\tau(s))\mathbf{E_i}(s,\tau)$,
4. **Orthonormalisation**
   Calculer la matrice P en utilisant le procédé d'orthonormalisation de Gram-Schmidt,
5. **Conditions aux limites**
   Calculer la projection orthogonale $\bar{\lambda}^\perp$ de λ^0 sur l'espace solution : $\bar{\lambda}^\perp = -P(LP)^+\eta_{\mathbf{1}}(S,\tau) + (I_p - P(LP^+L)PP^T\lambda^0$,
6. **Calcul de la déformation**
   Calculer $\bar{\eta}^\perp(s,\tau) = \sum_{i=1}^p \bar{\lambda}_i^\perp(\tau)\mathbf{E_i}(s,\tau) + \eta_{\mathbf{1}}(s,\tau)$,
7. **Application de la déformation**
   - Si $\|\bar{\eta}^\perp\|_\infty > \eta_{max}$ alors $\Delta\tau = \frac{\eta_{max}}{\|\bar{\eta}^\perp\|_\infty}$ sinon $\Delta\tau = 1$,
   - $\Phi(s,\tau+\Delta\tau) = \Phi(s,\tau) + \Delta\tau\,\bar{\eta}^\perp(s,\tau)$.
   - $\tau \leftarrow \tau + \Delta\tau$

2.3 Exemple : le robot Hilare 2

Pour illustrer cet algorithme, nous allons faire une itération de déformation sur une trajectoire rectiligne à vitesse constante pour le robot Hilare 2 (voir figure 2.6).

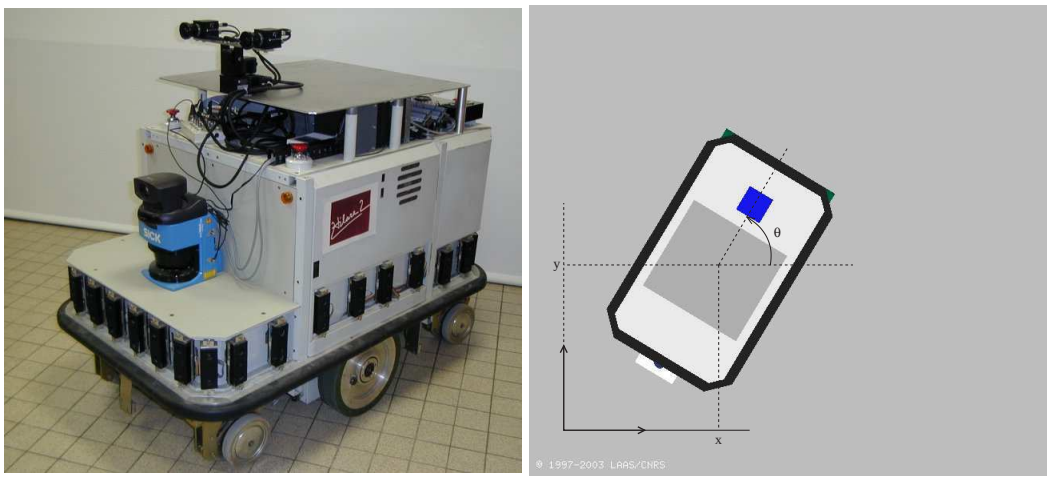

FIG. 2.6 – le robot Hilare 2 dans une configuration (x, y, θ)

La modélisation d'Hilare 2 est la suivante. L'espace des configurations $\mathcal{C}$ est de dimension 3, soit $n = 3$, $\mathbf{q} = (x, y, \theta)$. Nous avons 2 champs de vecteurs, X_1 et X_2, soit $k = 2$ et les commandes $u_1(s) = v(s)$, la vitesse linéaire, et $u_2(s) = \omega(s)$, la vitesse angulaire.

$$\mathbf{\Gamma}(s) = \begin{pmatrix} x(s) \\ y(s) \\ \theta(s) \end{pmatrix}, \forall s \in I, \mathbf{\Gamma}'(s) = v(s) \begin{pmatrix} \cos(\theta(s)) \\ \sin(\theta(s)) \\ 0 \end{pmatrix} + \omega(s) \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

où

Dans notre exemple, nous avons $v(s) = 1$ pour tout s et $\omega(s) = 0$ pour tout s. La position initiale du robot est $(0, 0, 0)$.

Les fonctions élémentaires $\mathbf{e_i}$ que nous utilisons sont des séries de Fourier tronquées à l'ordre q.

$$\begin{aligned}
\mathbf{e}_1(s) &= (1,0)^T & \mathbf{e}_2(s) &= (0,1)^T \\
\mathbf{e}_3(s) &= (\cos(\frac{2\pi s}{L}),0)^T & \mathbf{e}_4(s) &= (0,\cos(\tfrac{2\pi s}{L}))^T \\
\mathbf{e}_5(s) &= (\sin(\frac{2\pi s}{L}),0)^T & \mathbf{e}_6(s) &= (0,\sin(\tfrac{2\pi s}{L}))^T \\
&\vdots & &\vdots \\
\mathbf{e}_{4q-1}(s) &= (\cos(\frac{2q\pi s}{L}),0)^T & \mathbf{e}_{4q}(s) &= (0,\cos(\tfrac{2q\pi s}{L}))^T \\
\mathbf{e}_{4q+1}(s) &= (\sin(\frac{2q\pi s}{L}),0)^T & \mathbf{e}_{4q+2}(s) &= (0,\sin(\tfrac{2q\pi s}{L}))^T
\end{aligned}$$

Le potentiel répulsif est : $\frac{\partial U}{\partial q}(\mathbf{\Gamma}(s)) = (0, 1, 0)$, c'est à dire un potentiel constant tout le long de la trajectoire.

Les premières courbes (2.7) et (2.8) montre la déformation obtenue en appliquant la méthode jusqu'au paragraphe 2.2.3. La configuration initiale n'est pas modifiée mais la configuration finale est déplacée. La composante sur l'axe x est nulle.

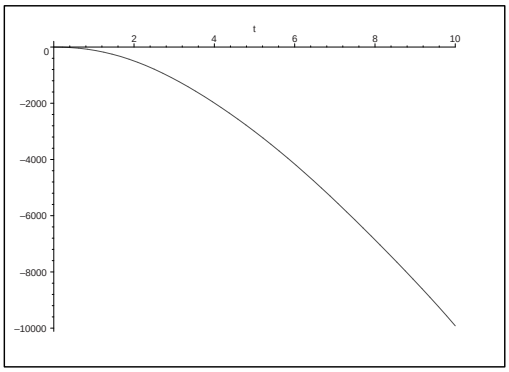

FIG. 2.7 – composante y de $\eta(s)$

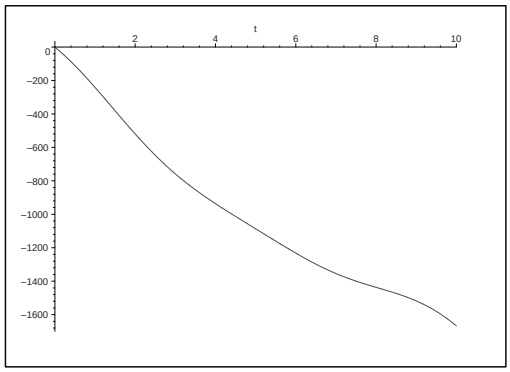

FIG. 2.8 – composante θ de $\eta(s)$

Sur les figures 2.9 et 2.10, on peut voir l'effet de la projection du vecteur η sur l'espace qui assure le respect des conditions aux limites (section 2.2.4).

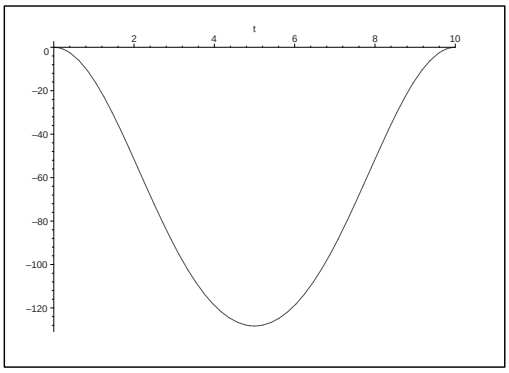

FIG. 2.9 – composante y de $\eta(s)$ calculé dans la base $\mathcal{E}$

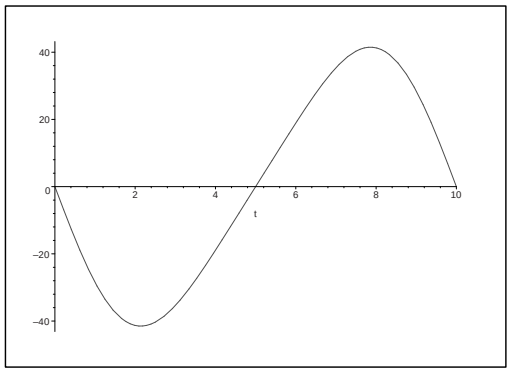

FIG. 2.10 – composante θ de $\eta(s)$ calculé dans la base $\mathcal{E}$

La section 2.2.5 propose une amélioration de la direction de déformation précédente. On peut voir la déformation correspondante sur les figures 2.11 et 2.12.

Pour constater l'amélioration de la déformation, il faut prendre en compte la normalisation effectuée à l'étape 7 de l'algorithme de déformation (section

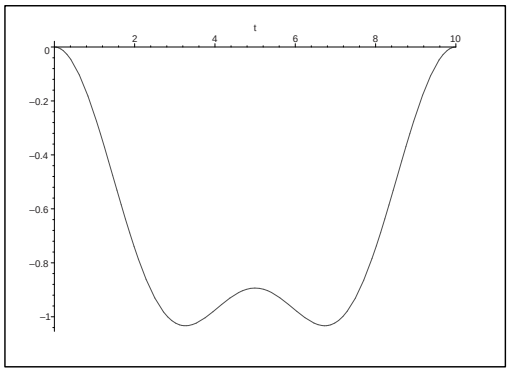

FIG. 2.11 – composante y de $\eta(s)$ calculé dans la base $\mathcal{F}$

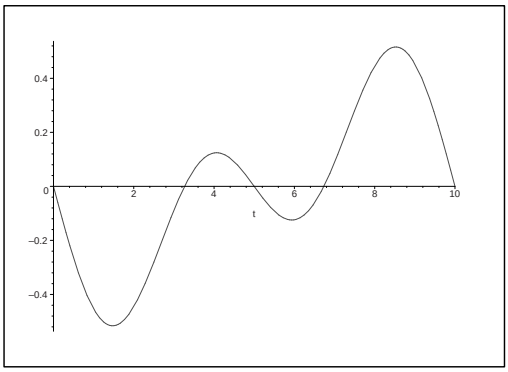

FIG. 2.12 – composante θ de $\eta(s)$ calculé dans la base $\mathcal{F}$

2.2.7). Les courbes 2.13 et 2.14 présente les normes euclidiennes des déformations utilisées pour le calcul de la norme infinie des déformations.

$$\|\eta\|_\infty = \max_{s\in I} \|\eta(s)\|$$

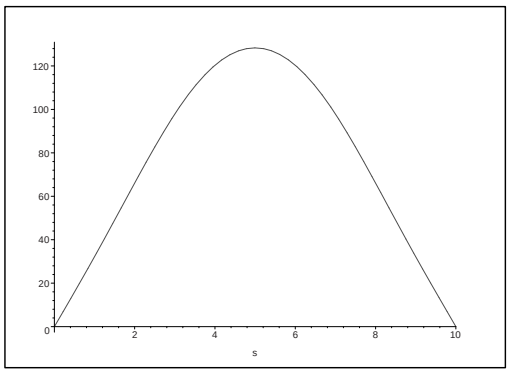

FIG. 2.13 – norme euclidienne de la déformation calculée dans la base $\mathcal{E}$

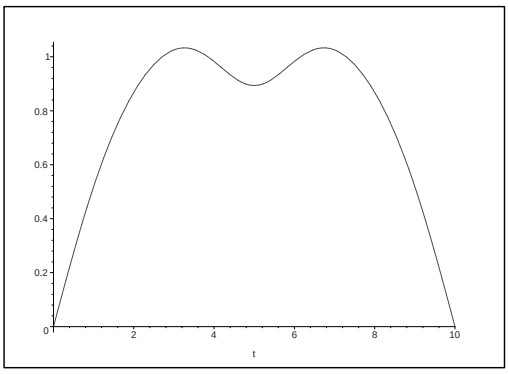

FIG. 2.14 – norme euclidienne de la déformation calculée dans la base $\mathcal{F}$.

Après normalisation des déformations, la variation du potentiel

$$\frac{dV}{d\tau}(\tau) = \int_0^S \frac{\partial U}{\partial \mathbf{q}}(\mathbf{\Gamma}_\tau(s))^T \eta(s,\tau) ds$$

est de -52 pour la déformation calculée dans la base $\mathcal{E}$ alors qu'elle est de -662 lorsque le calcul est effectué dans la base $\mathcal{F}$. La déformation obtenue par calcul dans la base orthonormée est donc bien meilleure.

Nous n'avons jamais prouvé que la variation du potentiel serait toujours meilleure avec la direction de déformation calculée dans la base orthonormée. Néanmoins, nos nombreuses expérimentations nous ont conduit à utiliser cette technique qui produit des déformations beaucoup plus proche de l'idéal.

III
Définition du potentiel d'une trajectoire

3.1 Introduction

Dans le chapitre précédent, nous avons développé et décrit la méthode de déformation de trajectoire pour des systèmes non-holonomes. Cette méthode d'optimisation dans l'espace des trajectoires cherche à minimiser un critère que nous avons appelé le potentiel de la trajectoire (section 2.1.4). Dans ce chapitre, nous allons définir ce potentiel et présenter quelques exemples.

Le potentiel V d'une trajectoire $\mathbf{\Gamma}$ est défini à partir d'une fonction de potentiel U sur $\mathcal{C}$ de la façon suivante :

$$\begin{aligned} V : C^\infty([0,S],\mathcal{C}) &\rightarrow \mathbb{R} \\ \mathbf{\Gamma} &\mapsto V(\mathbf{\Gamma}) = \int_0^S U(\mathbf{\Gamma}(s))ds \end{aligned} \tag{3.1}$$

où $U(\mathbf{\Gamma}(s))$ est le potentiel de la configuration $\mathbf{\Gamma}(s)$.

Cette fonction de potentiel U définie sur $\mathcal{C}$ à valeur dans $\mathbb{R}$ peut permettre d'optimiser un grand nombre de critères différents.

$$\begin{aligned} U : \mathcal{C} &\rightarrow \mathbb{R} \\ \mathbf{q} &\mapsto U(\mathbf{q}) \end{aligned} \tag{3.2}$$

Par exemple, si le robot ne doit pas atteindre une configuration $\mathbf{q}_D$ jugée dangereuse on exprimera le potentiel d'une configuration de telle sorte qu'il augmente quand le robot s'approche de $\mathbf{q}_D$ et diminue lorsqu'il s'en éloigne.

Dans la section suivante, nous allons voir comment définir U pour que la trajectoire s'éloigne des obstacles.

3.2 Évitement d'obstacles

Afin d'éloigner la trajectoire des obstacles, on définit le potentiel d'une configuration en fonction de la distance entre le robot et les obstacles de manière à ce qu'il augmente lorsque le robot se rapproche des obstacles et diminue lorsqu'il s'en éloigne.

Pour cela, nous définissons U_d une fonction décroissante,

$$\begin{array}{rcl} U_d : \mathbb{R}^+ & \rightarrow & \mathbb{R} \\ d & \mapsto & U_d(d) = \left\{ \begin{array}{cc} \frac{1}{d+d_0} - \frac{1}{d_1+d_0} & si\ 0 \leq d \leq d_1 \\ 0 & si\ d > d_1 \end{array} \right. \end{array} \tag{3.3}$$

qui pour une primitive obstacle P un robot R dans la configuration $\mathbf{q}$ donne le potentiel de cette configuration du robot en fonction de la distance minimum, notée d, entre P et R.

La figure (3.1) présente les courbes d'equi-potentiel pour un obstacle P circulaire ainsi que la forme d'une courbe représentative de U_d.

Avec cette définition de U_d nous avons

$$U(\mathbf{q}) = U_d(d) \tag{3.4}$$

Dans le cas d'un robot composé de plusieurs corps (voir les exemples) le potentiel U d'une configuration est défini comme la somme des potentiels U_d évalués pour chacun des corps.

De même, lorsqu'il y a plusieurs primitives obstacles, le potentiel U d'une configuration est défini comme la somme des potentiels U_d évalués pour chacune des primitives obstacles.

La distance euclidienne minimum d entre le robot R et la primitive obstacle P est la distance euclidienne entre les deux points les plus proches de l'obstacle et du robot notés respectivement P et$R$ (voir figure 3.2).

La distance d est donc définie de la manière suivante :

$$\begin{array}{rcl} d : \mathcal{W} \times \mathcal{W} & \rightarrow & \mathbb{R}^+ \\ (P, R) & \mapsto & d(P, R) = \sqrt{(p_1 - r_1)^2 + \ldots + (p_w - r_w)^2} \end{array} \tag{3.5}$$

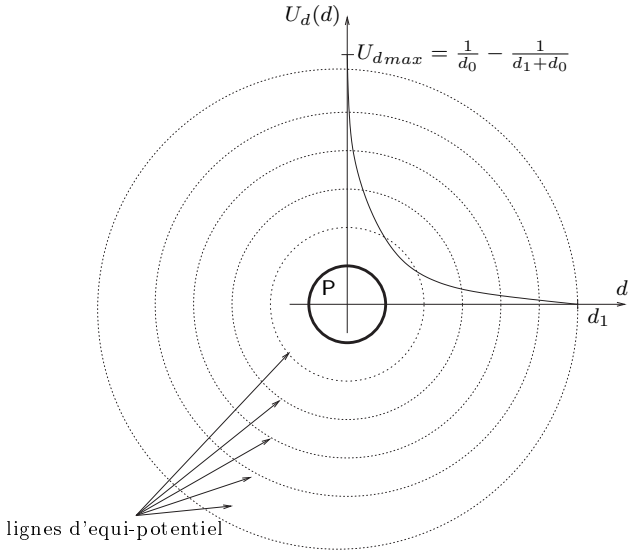

FIG. 3.1 – Sur cette figure on peut voir les courbes d'equi-potentiel pour un obstacle P circulaire ainsi que la forme de U_d.

où $(p_1, \ldots, p_w)$ et $(r_1, \ldots, r_w)$ sont respectivement les vecteurs de coordonnées des points P et R dans l'espace de travail $\mathcal{W}$. On note w la dimension de cet espace. Dans nos exemples $w = 2$ ou 3.

Avant de nous intéresser au gradient de $U(\mathbf{q})$, remarquons que les points P et R dépendent de la configuration $\mathbf{q}$ du robot (l'obstacle étant considéré fixe par rapport au référentiel global). On a donc

$$\begin{array}{rcl} P : \mathcal{C} & \rightarrow & \mathcal{W} \\ \mathbf{q} & \mapsto & P(\mathbf{q}) \end{array} \tag{3.6}$$

$$\begin{array}{rcl} R : \mathcal{C} & \rightarrow & \mathcal{W} \\ \mathbf{q} & \mapsto & R(\mathbf{q}) \end{array} \tag{3.7}$$

Nous avons maintenant défini tous les éléments nécessaires pour étudier le gradient du potentiel d'une configuration. En effet, comme nous l'avons expliqué au chapitre précédent (voir paragraphe 2.1.4) nous avons besoin du gradient du potentiel et uniquement de ce gradient.

D'après l'égalité (3.4) nous avons

$$\frac{\partial U(\mathbf{q})}{\partial \mathbf{q}} = \frac{\partial}{\partial \mathbf{q}} U_d(d(P(\mathbf{q}), R(\mathbf{q})))$$

soit

$$\begin{aligned}\frac{\partial U(\mathbf{q})}{\partial \mathbf{q}} &= U'_d(d)\frac{\partial}{\partial \mathbf{q}}d(P(\mathbf{q}),R(\mathbf{q})) \\ &= U'_d(d)\left(\frac{\partial d(P,R)}{\partial P}\frac{\partial P(\mathbf{q})}{\partial \mathbf{q}}+\frac{\partial d(P,R)}{\partial R}\frac{\partial R(\mathbf{q})}{\partial \mathbf{q}}\right) \qquad (3.8)\end{aligned}$$

Examinons les dérivées partielles $\frac{\partial d(P,R)}{\partial P}$ et $\frac{\partial d(P,R)}{\partial R}$.
D'après la définition de d (3.5) on a :

$$\frac{\partial d(P,R)}{\partial P}=\left(\frac{p_1-r_1}{d(P,R)},\ldots,\frac{p_w-r_w}{d(P,R)}\right)$$

et

$$\frac{\partial d(P,R)}{\partial R}=\left(\frac{-(p_1-r_1)}{d(P,R)},\ldots,\frac{-(p_w-r_w)}{d(P,R)}\right)$$

on remarque alors que

$$\frac{\partial d(P,R)}{\partial P}=-\frac{\partial d(P,R)}{\partial R}$$

ainsi l'expression (3.8) devient

$$\begin{aligned}\frac{\partial U(\mathbf{q})}{\partial \mathbf{q}} &= U'_d(d)\frac{\partial d(P,R)}{\partial P}\left(\frac{\partial P(\mathbf{q})}{\partial \mathbf{q}}-\frac{\partial R(\mathbf{q})}{\partial \mathbf{q}}\right) \\ &= U'_d(d)\frac{(P-R)^T}{d(P,R)}\left(\frac{\partial P(\mathbf{q})}{\partial \mathbf{q}}-\frac{\partial R(\mathbf{q})}{\partial \mathbf{q}}\right) \qquad (3.9)\end{aligned}$$

On appelle le terme $U'_d(d)\frac{(P-R)^T}{d(P,R)}$ la force dans l'espace de travail $\mathcal{W}$ et on la note $\mathbf{f}_\mathcal{W}$ (voir figure 3.2). $\mathbf{f}_\mathcal{W}$ est un vecteur dirigé de P vers R et d'intensité $|U'_d(d)|$.

$$\begin{aligned}\mathbf{f}_\mathcal{W}:\mathcal{W}\times\mathcal{W} &\rightarrow \mathcal{W} \\ (P,R) &\mapsto \mathbf{f}_\mathcal{W}(P,R)=U'_d(d)\frac{(P-R)^T}{d(P,R)}\end{aligned}$$

L'expression (3.9) devient donc

$$\frac{\partial U(\mathbf{q})}{\partial \mathbf{q}}=\mathbf{f}_\mathcal{W}(P,R)\left(\frac{\partial P(\mathbf{q})}{\partial \mathbf{q}}-\frac{\partial R(\mathbf{q})}{\partial \mathbf{q}}\right) \qquad (3.10)$$

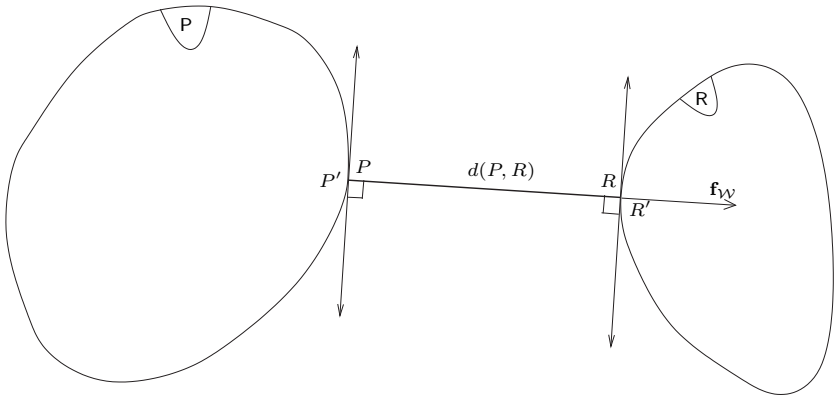

FIG. 3.2 –

Lorsque nous avons défini les points P et R nous avons simplement dit qu'ils étaient fonctions de la configuration du robot. En effet l'expression de $P(\mathbf{q})$ et de $R(\mathbf{q})$ est difficile à obtenir et les dérivées partielles aussi. Mais la cinématique nous donne un résultat très intéressant.

Reprenons l'expression (3.10) de $\frac{\partial U(\mathbf{q})}{\partial \mathbf{q}}$ et développons-la partiellement :

$$\begin{aligned}\frac{\partial U(\mathbf{q})}{\partial \mathbf{q}} &= \mathbf{f}_{\mathcal{W}}(P,R)\left(\frac{\partial P(\mathbf{q})}{\partial \mathbf{q}} - \frac{\partial R(\mathbf{q})}{\partial \mathbf{q}}\right) \\ &= \left(\mathbf{f}_{\mathcal{W}}(P,R)\frac{\partial P(\mathbf{q})}{\partial q_1}, \ldots, \mathbf{f}_{\mathcal{W}}(P,R)\frac{\partial P(\mathbf{q})}{\partial q_n}\right) \\ &\quad - \left(\mathbf{f}_{\mathcal{W}}(P,R)\frac{\partial R(\mathbf{q})}{\partial q_1}, \ldots, \mathbf{f}_{\mathcal{W}}(P,R)\frac{\partial R(\mathbf{q})}{\partial q_n}\right) \qquad (3.11)\end{aligned}$$

Examinons le terme $\mathbf{f}_{\mathcal{W}}(P,R)\frac{\partial R(\mathbf{q})}{\partial q_1}$ de l'expression (3.11). Considérons un mouvement dans lequel seule la variable de configuration q_1 évolue de telle sorte que $q_1 = t$ où t est le temps. Dans ce mouvement, la vitesse du point R est donnée par $\frac{\partial R(\mathbf{q})}{\partial q_1}$. La loi de composition des vitesses en cinématique (voir figure 3.3) nous dit alors que cette vitesse $\vec{V}_R$ est la somme de la vitesse $\vec{V}_{R/\mathsf{R}}$ du point R par rapport au corps R et de la vitesse $\vec{V}_{R'}$ du point coïncidant R' appartenant au corps R par rapport au référentiel global (appelée vitesse d'entraînement).

On a donc

$$\mathbf{f}_{\mathcal{W}}(P,R)\vec{V}_R = \mathbf{f}_{\mathcal{W}}(P,R)\left(\vec{V}_{R/\mathsf{R}} + \vec{V}_{R'}\right)$$

Or, la vitesse $\vec{V}_{R/\mathsf{R}}$ est tangente à la surface du robot R car R se déplace

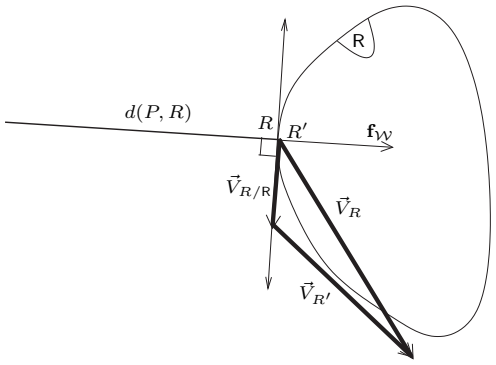

FIG. 3.3 – Cinématique du point R. La vitesse du point R par rapport au référentiel global est la somme de la vitesse du point R' appartenant au robot et de la vitesse du point R par rapport au solide R.

sur cette surface et $\mathbf{f}_\mathcal{W}$ est normale à cette surface donc le produit scalaire est nul. Alors

$$\mathbf{f}_\mathcal{W}(P,R)\vec{V}_R = \mathbf{f}_\mathcal{W}(P,R)\vec{V}_{R'}$$

Nous pouvons faire ce raisonnement pour chacun des termes de l'expression (3.11) et il en résulte que

$$\frac{\partial U(\mathbf{q})}{\partial \mathbf{q}} = \mathbf{f}_\mathcal{W}(P,R)\left(\frac{\partial P'(\mathbf{q})}{\partial \mathbf{q}} - \frac{\partial R'(\mathbf{q})}{\partial \mathbf{q}}\right) \tag{3.12}$$

où les expressions des dérivées partielles de P' et R' sont connues car elles expriment la vitesse d'un point fixe du robot en fonction de la variation de la configuration $\mathbf{q}$ du robot.

De la même manière que nous avons appelé $\mathbf{f}_\mathcal{W}$ la force dans l'espace de travail, on appellera $\frac{\partial U(\mathbf{q})}{\partial \mathbf{q}}$ la force dans l'espace des configurations, notée $\mathbf{f}_\mathcal{C}$.

Ainsi, la variation du potentiel d'un corps défini par la distance de ce corps à l'obstacle est égale à la variation du potentiel du point du corps coïncidant avec le point le plus proche de l'obstacle.

3.3 Exemples

3.3.1 Exemple 1 : le robot Hilare 2

Dans cet exemple, le robot utilisé est Hilare 2, l'espace des configurations est de dimension 3 et on note $\mathbf{q} = (q_x, q_y, q_\theta)$ (voir figure 2.6).

Les primitives obstacles utilisées sont les points fournis par le télémètre laser à balayage horizontal placé à l'avant du robot.

Dans cet exemple, illustré par la figure 3.4, nous considérons deux primitives obstacles : les points P_1 et P_2.

Comme nous l'avons dit précédemment, lorsqu'il y a plusieurs primitives obstacles, le potentiel U d'une configuration est la somme des potentiels U_d calculés pour chaque primitive obstacle P_1 et P_2.

Examinons le calcul de $\mathbf{f}_{\mathcal{C}}(\mathbf{q})$.

$$\begin{aligned}\mathbf{f}_{\mathcal{C}}(\mathbf{q}) &= \frac{\partial U(\mathbf{q})}{\partial \mathbf{q}} \\ &= \frac{\partial U_d(d_1)}{\partial \mathbf{q}} + \frac{\partial U_d(d_2)}{\partial \mathbf{q}} \\ &= \mathbf{f}_{\mathcal{C}1}(\mathbf{q}) + \mathbf{f}_{\mathcal{C}2}(\mathbf{q})\end{aligned}$$

Le calcul de $\frac{\partial U_d(d_1)}{\partial \mathbf{q}}$ et de $\frac{\partial U_d(d_2)}{\partial \mathbf{q}}$ étant identique, nous allons considérer uniquement le premier terme $\mathbf{f}_{\mathcal{C}1}(\mathbf{q})$.

Le point R_1, c'est à dire le point du coté du robot le plus proche de P_1 est la projection orthogonale de P_1 sur le coté du robot. Les coordonnées du point R_1 dans le repère du robot $(O_r, \vec{x}_r, \vec{y}_r)$ sont donc :

$$R_1 = \begin{pmatrix} R_{1x_r} \\ R_{1y_r} \end{pmatrix}_{(O_r,\vec{x}_r,\vec{y}_r)} = \begin{pmatrix} P_{1x_r} \\ \frac{l}{2} \end{pmatrix}_{(O_r,\vec{x}_r,\vec{y}_r)}$$

où P_{1x_r} est l'abscisse du point P_1 dans ce même repère.

Dans le repère global $(O, \vec{x}, \vec{y})$, les coordonnées de R_1 sont :

$$R_1 = \begin{pmatrix} q_x + R_{1x_r}\cos(q_\theta) - R_{1y_r}\sin(q_\theta) \\ q_y + R_{1x_r}\sin(q_\theta) + R_{1y_r}\cos(q_\theta) \end{pmatrix}_{(O,\vec{x},\vec{y})} \tag{3.13}$$

Nous savons d'après l'expression (3.10) que

$$\mathbf{f}_{\mathcal{C}1}(\mathbf{q}) = \mathbf{f}_{\mathcal{W}}(P_1, R_1)\frac{\partial R_1(\mathbf{q})}{\partial \mathbf{q}}$$

soit

$$\mathbf{f}_{\mathcal{C}1}(\mathbf{q}) = \begin{pmatrix} \mathrm{f}_{\mathcal{W}x} \\ \mathrm{f}_{\mathcal{W}y} \end{pmatrix}^T \begin{pmatrix} 1+\frac{\partial R_{1x_r}}{\partial q_x}\cos(q_\theta) & \frac{\partial R_{1x_r}}{\partial q_y}\cos(q_\theta) & -R_{1x_r}\sin(q_\theta) - R_{1y_r}\cos(q_\theta) \\ \frac{\partial R_{1x_r}}{\partial q_x}\sin(q_\theta) & 1+\frac{\partial R_{1y_r}}{\partial q_y}\sin(q_\theta) & R_{1x_r}\cos(q_\theta) - R_{1y_r}\sin(q_\theta) \end{pmatrix}$$

$$= \begin{pmatrix} \mathrm{f}_{\mathcal{W}x} + \frac{\partial R_{1x_r}}{\partial q_x}\left(\mathrm{f}_{\mathcal{W}x}\cos(q_\theta) + \mathrm{f}_{\mathcal{W}y}\sin(q_\theta)\right) \\ \mathrm{f}_{\mathcal{W}y} + \frac{\partial R_{1x_r}}{\partial q_y}\left(\mathrm{f}_{\mathcal{W}x}\cos(q_\theta) + \mathrm{f}_{\mathcal{W}y}\sin(q_\theta)\right) \\ \frac{\partial R_{1x_r}}{\partial q_\theta}\left(\mathrm{f}_{\mathcal{W}x}\cos(q_\theta) + \mathrm{f}_{\mathcal{W}y}\sin(q_\theta)\right) - \\ \mathrm{f}_{\mathcal{W}x}\left(R_{1x_r}\sin(q_\theta) + R_{1y_r}\cos(q_\theta)\right) + \\ \mathrm{f}_{\mathcal{W}y}\left(R_{1x_r}\cos(q_\theta) - R_{1y_r}\sin(q_\theta)\right) \end{pmatrix}^T$$

Or, la direction $\begin{pmatrix} \cos(q_\theta) \\ \sin(q_\theta) \end{pmatrix}_{(O,\vec{x},\vec{y})}$ est orthogonale à la direction $\begin{pmatrix} \mathrm{f}_{\mathcal{W}x} \\ \mathrm{f}_{\mathcal{W}y} \end{pmatrix}_{(O,\vec{x},\vec{y})}$ donc $\mathrm{f}_{\mathcal{W}x}\cos(q_\theta) + \mathrm{f}_{\mathcal{W}y}\sin(q_\theta) = 0$.

Alors

$$\mathbf{f}_{\mathcal{C}1}(\mathbf{q})^T = \begin{pmatrix} \mathrm{f}_{\mathcal{W}x} \\ \mathrm{f}_{\mathcal{W}y} \\ -\mathrm{f}_{\mathcal{W}x}\left(R_{1x_r}\sin(q_\theta) + R_{1y_r}\cos(q_\theta)\right) + \\ \mathrm{f}_{\mathcal{W}y}\left(R_{1x_r}\cos(q_\theta) - R_{1y_r}\sin(q_\theta)\right) \end{pmatrix} \quad (3.14)$$

Nous aurions pu obtenir ce résultat plus directement en utilisant l'expression (3.12) de $\mathbf{f}_{\mathcal{C}}$.

$$\frac{\partial U(\mathbf{q})}{\partial \mathbf{q}} = -\mathbf{f}_{\mathcal{W}}(P_1, R_1)\frac{\partial R_1'(\mathbf{q})}{\partial \mathbf{q}}$$

En effet, à partir des coordonnées de R_1 exprimée dans le repère global (3.13) et en considérant le point R_1 fixe sur le robot on a :

$$\frac{\partial R_1'(\mathbf{q})}{\partial \mathbf{q}} = \begin{pmatrix} 1 & 0 & R_{1x_r}\sin(q_\theta) + R_{1y_r}\cos(q_\theta) \\ 0 & 1 & R_{1x_r}\cos(q_\theta) - R_{1y_r}\sin(q_\theta) \end{pmatrix}$$

d'où le résultat (3.14).

3.3.2 Exemple 2 : le robot Hilare 2 muni d'une remorque

Précisons maintenant comment déterminer la force dans l'espace des configurations du robot Hilare 2 avec sa remorque (voir figure 3.5).

Dans ce cas, l'espace des configurations $\mathcal{C}$ est de dimension 4 et on note $\mathbf{q} = (q_x, q_y, q_\theta, q_\varphi)$ (voir figure 3.6).

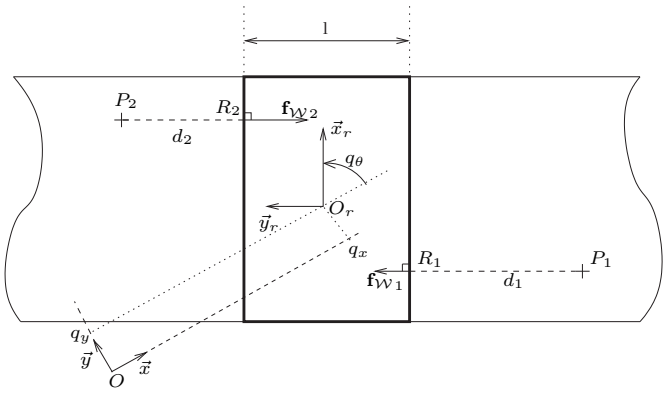

FIG. 3.4 – Le repère $(O_r, \vec{x}_r, \vec{y}_r)$ est un repère lié au robot, l'axe $\vec{x}_r$ est dirigé vers l'avant du robot. Les points R_1 et R_2 sont respectivement la projection orthogonale des points obstacles P_1 et P_2 sur les flans du robot. La largeur du robot est l.

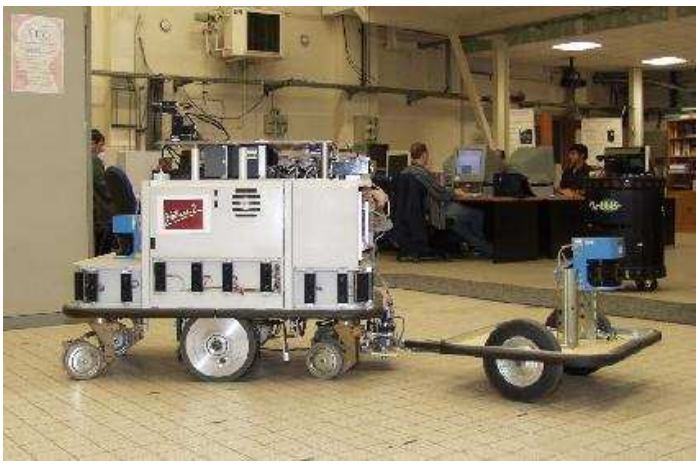

FIG. 3.5 – Le robot Hilare 2 muni de sa remorque. Sur cette figure, on peut voir les deux télémètres laser à balayage horizontal à l'avant du robot et sur la remorque. Un codeur optique absolu placé à la liaison pivot du robot et de la remorque permet de connaître l'orientation de la remorque par rapport au robot.

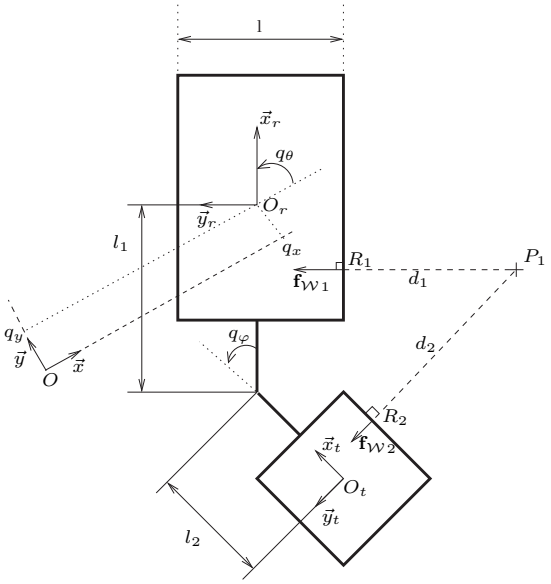

FIG. 3.6 – Définition du point R_1 et R_2 sur le robot Hilare 2 avec la remorque.

Dans le cas d'un robot multi-corps (robot+remorque) le potentiel U d'une configuration est la somme des potentiels U_d évalués pour chacun des corps. Soit $U(\mathbf{q}) = U_d(d_1) + U_d(d_2)$.

La force $\mathbf{f}_\mathcal{C}$ dans l'espace des configurations est donc

$$\begin{aligned} \mathbf{f}_\mathcal{C}(\mathbf{q}) &= -\frac{\partial U(\mathbf{q})}{\partial \mathbf{q}} \\ &= \frac{\partial}{\partial \mathbf{q}} U_d(d_1) + \frac{\partial}{\partial \mathbf{q}} U_d(d_2) \\ &= \mathbf{f}_{\mathcal{C}1}(\mathbf{q}) + \mathbf{f}_{\mathcal{C}2}(\mathbf{q}) \end{aligned}$$

Le premier terme $\mathbf{f}_{\mathcal{C}1}$ se calcule de la même manière que pour Hilare 2 sans la remorque avec une quatrième composante toujours nulle. Cette quatrième composante signifie qu'une force dans l'espace des configurations agissant sur le robot seul n'influence pas la composante q_φ.

Le deuxième terme $\mathbf{f}_{\mathcal{C}2}$ pourrait être déterminé par la méthode présentée au paragraphe 3.2 mais nous allons montrer une autre méthode qui utilise les résultats obtenus dans l'espace des configurations du robot seul.

En effet, les coordonnées du point R_2 peuvent s'écrire comme une fonction de la configuration de la remorque considérée comme un robot (sans remorque).

$$R_2 = R_2(\mathbf{q}_T(\mathbf{q}))$$

où $\mathbf{q}$ est la configuration du robot $\mathbf{q} = (q_x, q_y, q_\theta, q_\varphi)$ et $\mathbf{q}_T = (q_{Tx}, q_{Ty}, q_{T\theta})$ la configuration de la remorque dans l'espace des configurations $\mathcal{C}_R$ du robot sans la remorque. $q_{Tx} = q_x - l_1 \cos\theta - l_2 \cos(\theta + \varphi)$, $q_{Ty} = q_x - l_1 \sin\theta - l_2 \sin(\theta + \varphi)$ et $q_{T\theta} = \theta + \varphi$.

Alors

$$\frac{\partial R_2'(\mathbf{q}_T(\mathbf{q}))}{\partial \mathbf{q}} = \frac{\partial R_2'}{\partial \mathbf{q}_T} \frac{\partial \mathbf{q}_T}{\partial \mathbf{q}}$$

En utilisant l'expression (3.12) de $\mathbf{f}_\mathcal{C}(\mathbf{q})$ on obtient :

$$\mathbf{f}_{\mathcal{C}2}(\mathbf{q}) = \mathbf{f}_\mathcal{W}(P_1, R_2) \frac{\partial R_2'}{\partial \mathbf{q}_T} \frac{\partial \mathbf{q}_T}{\partial \mathbf{q}}$$

Le terme $\mathbf{f}_\mathcal{W}(P_1, R_2) \frac{\partial R_2'}{\partial \mathbf{q}_T}$ représente la force dans l'espace des configurations $\mathcal{C}_R$ du robot seul. On le calcule comme expliqué au paragraphe précédent. Le terme $\frac{\partial \mathbf{q}_T}{\partial \mathbf{q}}$ est la matrice (3×4) des dérivées partielles de la configuration de la remorque dans l'espace $\mathcal{C}_R$ par rapport aux variables de configurations du robot complet (dans l'espace $\mathcal{C}$ de dimension 4).

3.3.3 Exemple 3 : camion à remorque avec timon

Dans cet exemple (voir figure 3.7) le robot considéré est un camion tirant une remorque avec timon utilisé pour le transport des pièces de l'avion Airbus A380. Les 24 roues de la remorque s'orientent de façon à ce que leur axe de rotation passe par le centre instantané de rotation de la remorque.

Cet exemple est tiré d'un projet entre la société Airbus, le LAAS, la Direction Régionale de l'Equipement et Kineo-CAM, une start-up créée par des chercheurs du LAAS.

L'objectif de ce projet était d'étudier la faisabilité d'un itinéraire pour le transport de pièces volumineuses de l'avion Airbus A380 sur des routes secondaires. En effet, les ponts des autoroutes sont trop bas par rapport à la taille des pièces transportées.

La planification de trajectoires pour des systèmes si complexes est actuellement impossible. De plus, la classe d'homotopie des trajectoires est imposée par les routes. Il s'agissait donc, dans ce projet, d'optimiser des trajectoires initiales comportant des collisions avec les obstacles bordant la route.

Dans ce contexte, la méthode de déformation de trajectoire présentée au chapitre précédent s'applique parfaitement.

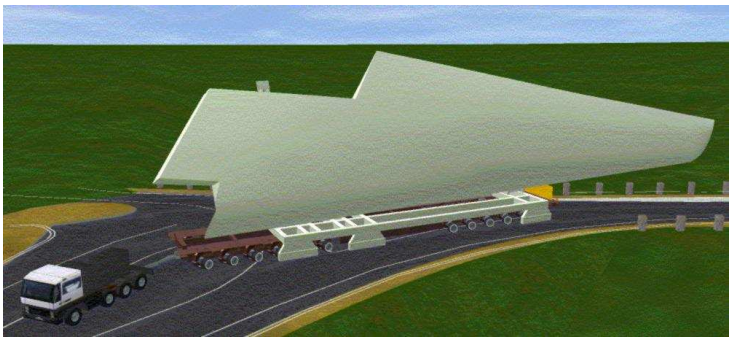

FIG. 3.7 – Un modèle du camion utilisé pour le transport des pièces de l'avion Airbus A380.

Dans cette simulation, l'information de distance disponible est la plus petite distance entre chaque corps du robot et l'environnement.

Les premières simulations ont été effectuées en définissant deux corps : la remorque et le tracteur. Le processus de déformation oscillait. En effet, pour une petite variation d'orientation de la remorque, la distance la plus petite entre la remorque et l'environnement oscillait entre la gauche et la droite du convoi et donc la direction de déformation aussi. La méthode était donc inefficace.

Pour résoudre ce problème, nous avons partagé la remorque en deux, dans le sens de la longueur et ainsi stabilisé le processus de déformation (voir figure 3.8).

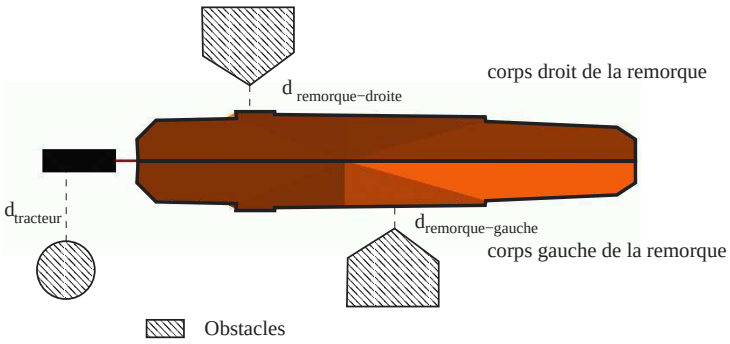

FIG. 3.8 – Séparation en deux corps de la remorque : corps droit de la remorque et corps gauche de la remorque.

Les développements informatiques résultant de ce projet ont été intégrés

dans un logiciel développé par Kineo-CAM et utilisés pour d'autres types de camions.

3.4 Utilisation de segments

Dans les chapitres précédents, nous avons montré comment calculer le potentiel "obstacle" d'une trajectoire à partir des points obstacles perçus par le robot. Dans ce chapitre nous allons voir comment calculer ce même potentiel grâce à des segments. Ces segments sont construits par un algorithme de segmentation à partir d'un ensemble de points fournis par les télémètres laser du robot.

L'utilisation de segments permet d'accélérer le temps de calcul des forces de répulsion nécessaires à la déformation de trajectoire en réduisant le nombre de primitives obstacles à traiter. En effet, dans un environnement d'intérieur contraint, le télémètre laser détecte de nombreux points appartenant à une même surface plane (bureau, mur,...). La déformation est donc plus rapide et l'exécution de la trajectoire est alors moins saccadée (voir chapitre 4).

Les points isolés, c'est à dire n'appartenant à aucun segment, sont "transformés" en segments en fonction de la distance au robot et de l'angle avec lequel il est perçu (voir figure 3.9). Ceci permet, d'une part, d'éviter le problème de cohérence entre des forces provenant de points et des forces provenant de segments et d'autre part de prendre en compte la précision angulaire des télémètres.

Les segments sont notés $[A_j, B_j]$. Un point $P_j(\lambda)$ appartenant à ce segment est tel que :

$$\overrightarrow{OP_j}(\lambda) = \overrightarrow{OA_j} + \frac{\lambda}{L}(\overrightarrow{OB_j} - \overrightarrow{OA_j})$$

où L est la longueur du segment $[A_j B_j]$, λ un réel appartenant à l'intervalle $[0, L]$ et O est l'origine du repère global.

La force $\mathbf{F}_\mathcal{C}$ dans l'espace des configurations du robot engendré par le segment $[A_j Bj]$ est définie comme l'intégrale de la force $\mathbf{f}_\mathcal{C}$ produite par le champ de potentiel d'un point $P_j(\lambda)$ le long du segment.

$$\mathbf{F}_\mathcal{C} = \int_0^L \frac{\partial U_d(d(P(\mathbf{q},\lambda), R(\mathbf{q},\lambda)))}{\partial \mathbf{q}} d\lambda \tag{3.15}$$

L'implantation, à bord du robot Hilare 2, du calcul des forces générées par les segments se fait de la manière suivante : pour chaque configuration du robot le long de la trajectoire, les segments sont découpés comme le montre la figure 3.10 puis la formule (3.15) est appliquée aux segments ainsi obtenus.

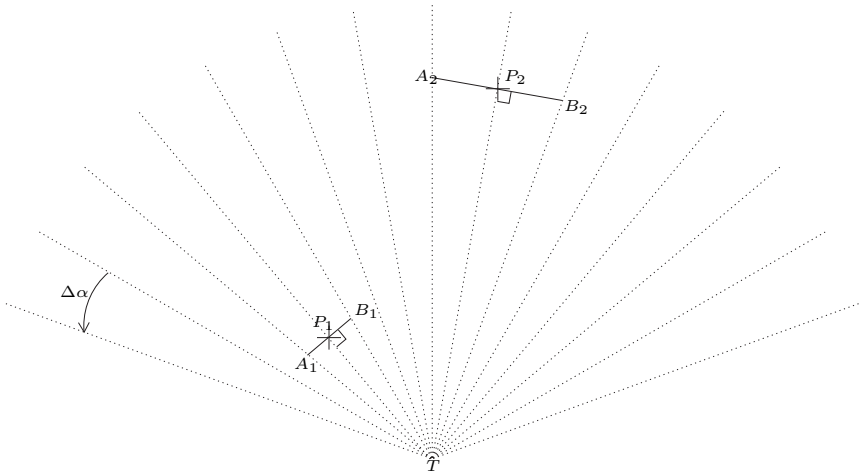

FIG. 3.9 – Transformation de points isolés P_1 et P_2 en segments. $\Delta\alpha$ est la résolution angulaire du télémètre laser situé au point T. Les lignes pointillées représentent les faisceaux laser.

Ce découpage est justifié pour deux raisons :

1. Comme l'indique l'équation (3.3), les obstacles perçus par le robot génèrent une force nulle au-delà d'une certaine distance notée d_1. Il est donc inutile de prendre en compte les segments au-delà de cette distance.
2. Seule la composante de la force orthogonale à la trajectoire est importante. La composante tangentielle induit un glissement des configurations le long de la trajectoire qui ne supprime pas les collisions détectées.

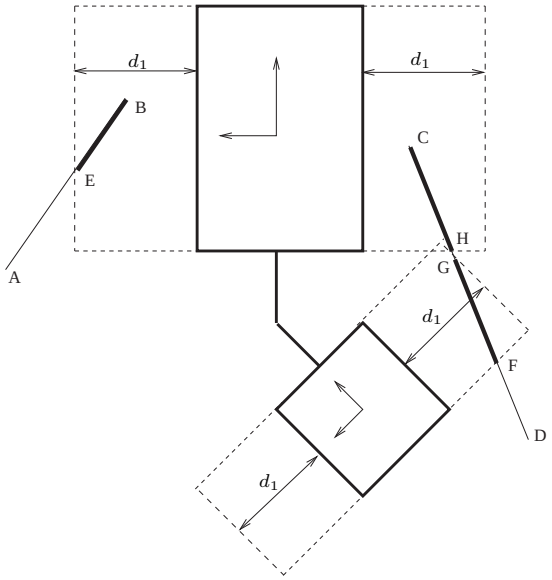

FIG. 3.10 – Sur cette figure, deux segments ont été détectés ([AB] et [CD]). Ils sont décomposés en 3 segments : [AB] en [EB], [CD] en [FG] et [HC], correspondant au découpage des 2 segments détectés dans les zones d'influences.

IV
Mise en oeuvre à bord du robot Hilare 2

Dans les deux chapitres précédents, nous avons présenté l'algorithme de déformation de trajectoire et comment les obstacles perçus par le robot génèrent les forces qui repoussent la trajectoire.

Dans ce chapitre, nous allons voir comment ces algorithmes fonctionnent à bord du robot Hilare 2.

Dans un premier temps nous présentons les difficultés posées par l'implantation sur les calculateurs du robot. Nous verrons ensuite les hypothèses nécessaires pour surmonter ces difficultés pour finalement présenter la méthode utilisée et un exemple d'exécution de trajectoire réelle.

4.1 Difficultés d'implantation à bord du robot Hilare 2

En appliquant ces algorithmes le problème qui se pose à nous est de garantir que la trajectoire du robot reste faisable. En effet, même si la méthode permet de maintenir "la dérive" des contraintes non-holonomes à un niveau acceptable proche de 0 (voir la section 2.2.6), nous devons assurer la cohérence entre l'exécution de la trajectoire et le processus de déformation.

En d'autres termes, la trajectoire ne doit pas être déformée sous les roues du robot mais toujours en aval de celui-ci.

Pour illustrer ce problème, considérons que le robot parcourt sa trajectoire et détecte une collision. Il s'arrête alors pour modifier sa trajectoire

initiale jusqu'à ce que la collision disparaisse. Si rien n'est fait pour l'éviter, le processus de déformation peut déformer la trajectoire en amont de la position courante du robot. Lorsque la trajectoire ne sera plus en collision, le robot ne sera plus sur la trajectoire. Ce qui n'est, évidemment, pas souhaitable. Nous devons donc déterminer un intervalle de trajectoire à déformer que le robot ne pourra pas franchir.

Il y a une deuxième raison de calculer cet intervalle, c'est pour diminuer le temps de calcul. En effet, plus la trajectoire à déformer est longue plus le processus est long. En particulier le calcul des forces dues aux obstacles perçus est l'opération la plus coûteuse en temps de calcul dans le processus.

La trajectoire est donc une ressource partagée entre trois tâches (figure 4.1)

1. la tâches de détection de collisions. Nous l'appellerons cCheck.
2. la tâche de déformation. Nous l'appellerons tDeform.
3. la tâche de suivi de trajectoire. Nous l'appellerons tFollow.

Les trois tâches qui utilisent la trajectoire doivent donc être synchronisées pour assurer l'exécution de la trajectoire sans collision et sans discontinuité. Elles peuvent être classées en deux catégories :

1. **une tâche temps réel périodique (40Hz)** qui assure le suivi de trajectoire. Elle échantillonne la trajectoire et transmet les configurations à la tâche d'asservissement en position.
2. **deux tâches apériodiques et de durée variable**. La tâche de détection de collisions et la tâche de déformation.

Les difficultés d'implantation purement informatiques ne seront pas abordées ici. Bien qu'importantes et très intéressantes elles trouvent des solutions dans les outils développés au LAAS depuis des années [Camargo 1991], [Fleury 1996].

La difficulté que nous devons surmonter provient des contraintes cinématiques en vitesse et en accélération du robot. Lorsqu'il suit une trajectoire, il ne peut pas s'arrêter instantanément, il lui faut un certain intervalle que nous devons déterminer. Pour cela, un certain nombre d'hypothèses sont nécessaires et sont présentées dans le chapitre suivant.

4.2 Hypothèses et contraintes

Vitesses et accélérations maximales

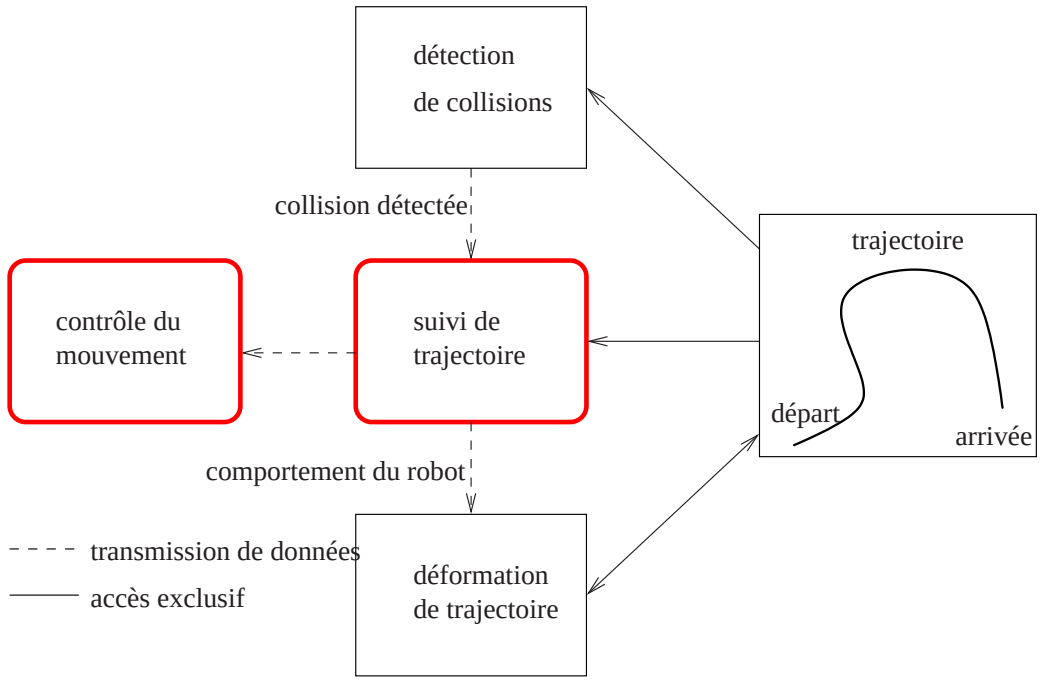

FIG. 4.1 – La trajectoire est une ressource partagée entre 3 tâches : la détection de collision, le suivi de trajectoire et la déformation de trajectoire. Les deux tâches entourées en gras sont des tâches fonctionnant en temps réel qui nécessitent une période d'exécution petite et fixe (25ms). La tâche "suivi de trajectoire" est décrite dans ce chapitre. La tâche "contrôle du mouvement" est l'asservissement en position du robot. Elle n'est pas décrite dans ce manuscrit (voir [De Luca 1998]).

Les vitesses linéaire v et angulaire ω ainsi que les accélérations linéaire $\dot{v}$ et angulaire $\dot{\omega}$ sont bornées ($\dot{}$ représente la dérivée par rapport au temps) :

$$
\begin{aligned}
-v_{max} &\leq v \leq v_{max} && (4.1)\\
-\omega_{max} &\leq \omega \leq \omega_{max} && (4.2)\\
-\dot{v}_{max} &\leq \dot{v} \leq \dot{v}_{max} && (4.3)\\
-\dot{\omega}_{max} &\leq \dot{\omega} \leq \dot{\omega}_{max} && (4.4)
\end{aligned}
$$

Contraintes sur les dérivées par rapport au temps de l'abscisse curviligne s

La trajectoire initiale est une courbe paramétrée $\mathbf{\Gamma}(s)$ dans l'espace des configurations. La tâche de suivi de trajectoire contrôle l'évolution de l'abscisse curviligne s en fonction du temps en tenant compte des contraintes cinématiques introduites ci-dessous.

Dans le cas du robot Hilare 2 avec la remorque, nous avons $\mathbf{\Gamma}(s) = (x(s), y(s), \theta(s), \varphi(s))$. Les vitesses linéaire et angulaire ainsi que les accélérations dépendent donc de s mais aussi de ses dérivées par rapport au temps $\dot{s}$ et $\ddot{s}$.

$$
\begin{aligned}
v &= \sqrt{\dot{x}^2 + \dot{y}^2} = d_v(s)\dot{s} && (4.5)\\
\omega &= \dot{\theta} = d_\omega(s)\dot{s} && (4.6)\\
\dot{v} &= d_v(s)\ddot{s} + d'_v(s)\dot{s}^2 && (4.7)\\
\dot{\omega} &= d_\omega(s)\ddot{s} + d'_\omega(s)\dot{s}^2 && (4.8)
\end{aligned}
$$

où $d_v(s) = \sqrt{x'^2(s) + y'^2(s)}$, $d_\omega(s) = \theta'(s)$ et $'$ représente la dérivée par rapport au paramètre s.

La trajectoire initiale est telle que pour $s = t$ $(0 \leq s < t)$ le robot la parcourt à vitesse nominale et les contraintes (4.1-4.4) sont satisfaites. Ce qui signifie que :

$$
\begin{aligned}
-v_{max} &\leq d_v(s) \leq v_{max} && (4.9)\\
-\omega_{max} &\leq d_\omega(s) \leq \omega_{max} && (4.10)\\
-\dot{v}_{max} &\leq d'_v(s) \leq \dot{v}_{max} && (4.11)\\
-\dot{\omega}_{max} &\leq d'_\omega(s) \leq \dot{\omega}_{max} && (4.12)
\end{aligned}
$$

En introduisant dans les inégalités (4.1-4.4) les expressions (4.5-4.8) on obtient des contraintes sur les dérivées première et seconde par rapport au

temps du paramètre s. La tâche de suivi de trajectoire ne peut donc pas arrêter le robot, c'est à dire $\dot{s} = 0$, instantanément mais doit calculer une courbe de décélération $\dot{s} = f(s)$ respectant des contraintes et donc un intervalle d'arrêt. Notons que la longueur de cet intervalle d'arrêt dépend des dérivées de $\mathbf{\Gamma}(s)$ et peut être arbitrairement grande.

Environnement statique

La méthode de déformation de trajectoire présentée dans les chapitres précédents est développée pour des environnements statiques. Un obstacle mobile peut être pris en compte si sa vitesse est compatible avec le temps de calcul des tâches de détection de collision et de déformation.

4.3 Algorithme mis en œuvre

L'algorithme est représenté sur la figure (4.2). La figure du haut est une représentation temporelle des événements et des actions qui se produisent au cours de l'exécution d'une trajectoire. La figure du bas est le profil de vitesse au cours de ce mouvement. Chacune des tâches sera expliquée précisément dans les sections suivantes. Nous donnons ici un aperçu général de l'algorithme.

La tâche de détection de collisions

cCheck utilise la trajectoire courante, l'abscisse curviligne courante s du robot et la perception courante de l'environnement pour calculer un certain nombre d'informations sur la trajectoire :

- la première configuration en collision $\mathbf{q}_{coll} = \mathbf{\Gamma}(s_{coll})$,
- la première portion de trajectoire non visible s_{hide},
- le meilleur intervalle de déformation $[s_{start}, s_{end}]$.

La tâche de suivi de trajectoire

Cette tâche périodique a deux fonctions.

La première fonction est de calculer, lorsque la tâche cCheck produit de nouvelles informations, l'abscisse curviligne noté s_{stop} où le robot peut s'arrêter. En l'absence de nouvelles informations, le robot s'arrêtera à cette abscisse.

La deuxième fonction de cette tâche est de faire progresser l'abscisse curviligne s et d'échantillonner périodiquement la trajectoire et de transmettre les configurations à la tâche d'asservissement. C'est ici qu'interviennent les contraintes de vitesse et d'accélération sur l'abscisse curviligne de façon à arrêter le robot à l'abscisse s_{stop}.

La tâche de déformation

Cette tâche déforme la trajectoire sur l'intervalle $[s_{stop}, s_{end}]$ comme expliqué dans les chapitres 2 et 3.

Déroulement de l'algorithme

Ces trois actions, (1) recherche de collision, (2) calcul de l'abscisse d'arrêt s_{stop} et (3) déformation de la trajectoire sur l'intervalle $[s_{stop}, s_{end}]$ sont répétées en boucle jusqu'à ce que le robot arrive à son but.

La figure 4.2 présente un chronogramme de l'enchaînement des étapes (1), (2) et (3) de l'algorithme ainsi que le profil de vitesse de l'abscisse curviligne correspondant.

Les sections suivantes décrivent plus en détail ces étapes.

4.3.1 cCheck : recherche des collisions

Chaque configuration le long de la trajectoire est, soit libre, soit en collision, ou alors cachée par les obstacles.

Configuration en collision Quand une configuration est à une distance inférieure à un seuil donné des obstacles elle est en collision. Le robot ne doit pas franchir une telle configuration.

Configuration cachée Si une configuration ne peut pas être entièrement vue par les capteurs du robot (voir figure 4.3) elle ne peut pas être étiquetée libre car il peut y avoir un obstacle non visible en collision et il n'est pas utile de déformer la trajectoire. Toutefois, en l'absence de nouvelles informations, le robot doit s'arrêter avant la première configuration cachée.

Configuration libre Lorsqu'une configuration n'est ni en collision ni cachée, elle est dite libre. Le robot peut la franchir sans risque.

La tâche cCheck cherche le long de la trajectoire, à partir de la configuration courante, la première configuration cachée et la première configuration en collision. Si une collision est détectée, cCheck calcule un intervalle optimal de déformation $[s_{start}, s_{end}]$ contenant s_{coll}. Les règles de calcul de cet intervalle sont empiriques et basées sur le temps de calcul du processus de déformation, la complexité cinématique du robot, etc. Par exemple l'intervalle est plus grand pour le robot Hilare 2 avec sa remorque que pour le robot Hilare 2 seul.

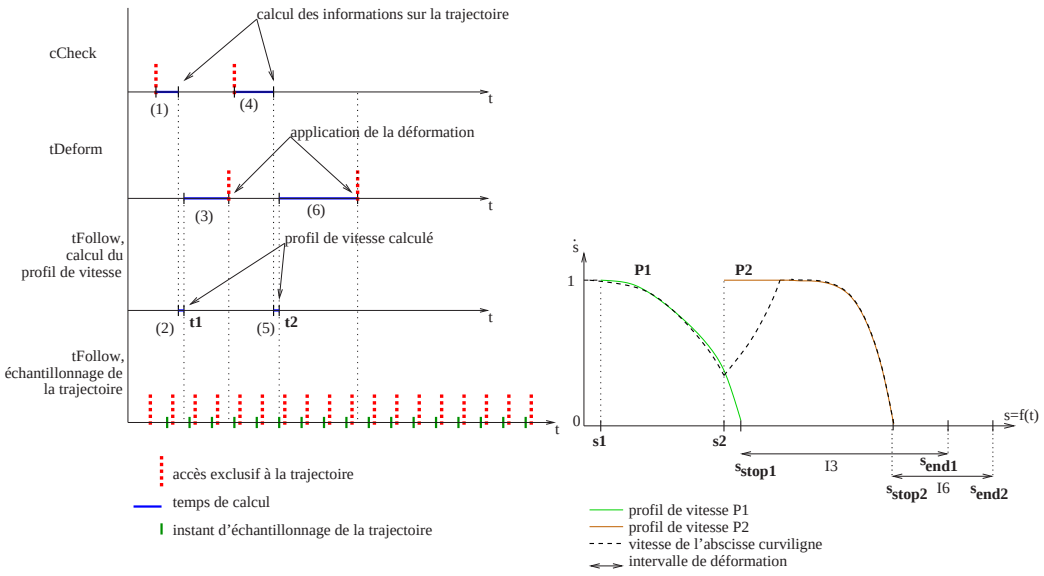

FIG. 4.2 – Chronogramme des événements de l'algorithme et profil de vitesse correspondant.

À l'étape (1) cCheck calcule un ensemble d'informations qui sont utilisées à l'étape (2) par tFollow pour déterminer une première abscisse d'arrêt s_{stop1} qui correspond au profil de vitesse $P1$. Au temps t_1 le robot qui est à l'abscisse s_1 commence à suivre ce profil de vitesse. Pendant que le robot suit ce profil, à l'étape (3), la tâche tDeform calcule la déformation correspondante sur l'intervalle noté $I_3 = [s_{stop1}, s_{end1}]$. C'est une boucle de l'algorithme.

Une autre boucle commence à l'étape (4) où cCheck produit de nouvelles informations utilisant une nouvelle perception de l'environnement. À l'étape (5) tFollow calcule un nouveau profil de vitesse $P2$ pour que le robot soit arrêté à l'abscisse s_{stop2}. Au temps t_2 correspondant à l'abscisse s_2, le robot accélère alors pour rejoindre le profil de vitesse $P2$. Pendant ce temps, à l'étape (6), la tâche tDeform déforme la trajectoire sur l'intervalle $I_6 = [s_{stop2}, s_{end2}]$. Une deuxième boucle de l'algorithme se termine.

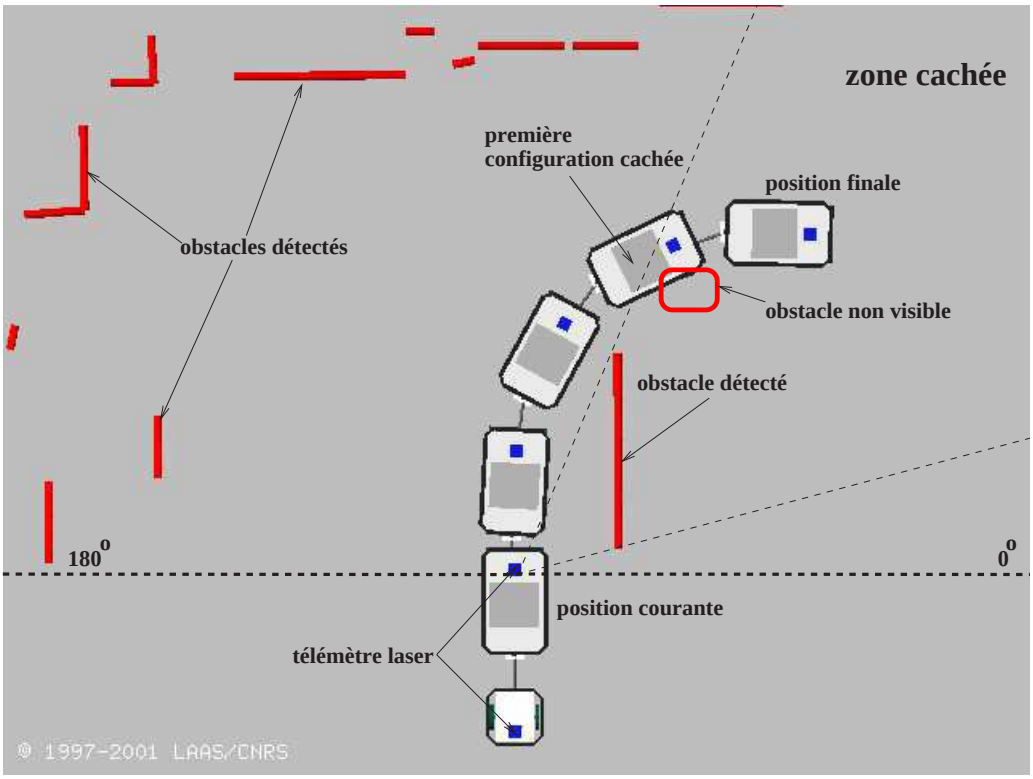

FIG. 4.3 – Une configuration cachée

4.3.2 tFollow, ou comment et quand arrêter Hilare 2

Dans cette section, nous allons présenter la technique utilisée pour faire varier la vitesse du robot le long de la trajectoire et comment nous déterminons l'intervalle nécessaire au robot pour s'arrêter. Cet intervalle nous permet de garantir que le robot s'arrêtera avant la première configuration de l'intervalle de déformation.

Problème

Comme expliqué dans la section des hypothèses (4.2) les contraintes en vitesse et en accélération du robot s'expriment en fonction de la vitesse et de l'accélération de l'abscisse curviligne s par les inégalités suivantes :

$$-v_{max} \leq d_v(s)\dot{s} \leq v_{max} \quad (4.13)$$

$$-\omega_{max} \leq d_\omega(s)\dot{s} \leq \omega_{max} \quad (4.14)$$

$$-\dot{v}_{max} \leq d_v(s)\ddot{s} + d'_v(s)\dot{s}^2 \leq \dot{v}_{max} \quad (4.15)$$

$$-\dot{\omega}_{max} \leq d_\omega(s)\ddot{s} + d'_\omega(s)\dot{s}^2 \leq \dot{\omega}_{max} \quad (4.16)$$

Les inégalités (4.13) et (4.14) sont satisfaites dès lors que $0 < \dot{s} < 1$.
Les inégalités (4.15) et (4.16) donnent des contraintes sur $\ddot{s}$ de la forme

$$\ddot{s}_{min}(s, \dot{s}) \leq \ddot{s} \leq \ddot{s}_{max}(s, \dot{s})$$

Le rôle de la tâche tFollow est donc de calculer, à chaque période, s et $\dot{s}$ de façon à satisfaire l'inégalité ci-dessus pour arrêter ou accélérer le robot.

Le problème se pose de la manière suivante : le robot étant à l'abscisse s_0 sur la trajectoire avec une vitesse de progression $\dot{s}_0$ comment calculer le profil de vitesse permettant au robot de s'arrêter à l'abscisse s_1.

Résolution analytique

Pour assurer que le robot ne dépassera pas l'abscisse s_1 et qu'il s'arrêtera en temps minimal nous devons intégrer à contre sens l'équation différentielle suivante :

$$\ddot{s} = \ddot{s}_{min}(s, \dot{s}) \quad (4.17)$$

à partir de $s = s_1$ avec $\dot{s}(s_1) = 0$.

Cette opération peut être faite dans le plan de phase $(s, \dot{s})$ [Bobrow 1985] dans lequel une courbe $s(t)$ est représentée par une relation entre s et $\dot{s}$.

Dans le plan de phase, l'équation (4.17) devient :

$$\frac{d\dot{s}}{ds} = \frac{1}{\dot{s}}\frac{d\dot{s}}{dt} = \frac{\ddot{s}_{min}(s,\dot{s})}{\dot{s}}$$

Appelons $\bar{s}(s)$ la solution, on a donc $\dot{\bar{s}}(s_1) = 0$. Si le robot parcourt la trajectoire avec une vitesse constante $\dot{s}_0$, il doit décélérer en suivant la courbe $\dot{\bar{s}}(s)$ lorsqu'il arrive à l'abscisse s_2 qui satisfait $\dot{\bar{s}}(s_2) = \dot{s}_0$ (voir figure 4.4).

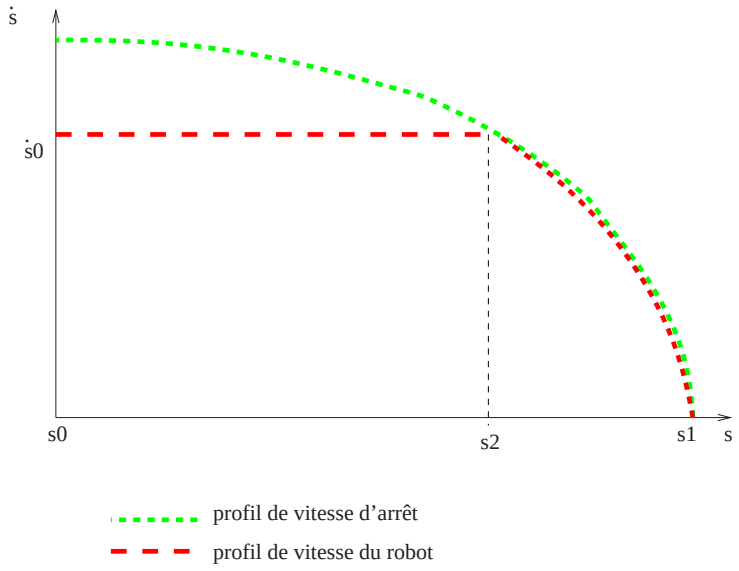

FIG. 4.4 – Courbe de décélération

Le calcul de l'intervalle le plus petit pour arrêter le robot nécessite donc d'intégrer une équation différentielle. Ce calcul peut être plus long que la période de tFollow car il dépend des conditions initiales. Nous devons donc trouver une autre méthode, en temps fini, pour avoir un majorant de cette abscisse de début de décélération.

Résolution pratique

Remarquons que si

$$0 \leq \dot{s} < 1 \tag{4.18}$$

et

$$v_{max}\,|\ddot{s}| \leq (1 - \dot{s}^2)\dot{v}_{max} \tag{4.19}$$

nous avons

$$|d_v(s)|\,|\ddot{s}| \leq (1-\dot{s}^2)\dot{v}_{max}$$

car $|d_v(s)| < v_{max}$ (voir section 4.2).
En enlevant les valeurs absolues nous avons :

$$-\dot{v}_{max} + \dot{s}^2\dot{v}_{max} \leq d_v(s)\ddot{s} \leq \dot{v}_{max} - \dot{s}^2\dot{v}_{max} \tag{4.20}$$

et comme $|d'_v(s)| < \dot{v}_{max}$ (voir section 4.2) nous pouvons écrire :

$$-\dot{v}_{max} \leq -d'_v(s) \leq \dot{v}_{max}$$

soit

$$-\dot{s}^2\dot{v}_{max} \leq -\dot{s}^2 d'_v(s) \leq \dot{s}^2\dot{v}_{max}$$

Nous pouvons donc écrire :

$$-\dot{v}_{max} - \dot{s}^2 d'_v(s) \leq -\dot{v}_{max} + \dot{s}^2\dot{v}_{max} \leq d_v(s)\ddot{s} \leq \dot{v}_{max} - \dot{s}^2\dot{v}_{max} \leq \dot{v}_{max} - \dot{s}^2 d'_v(s)$$

soit

$$-\dot{v}_{max} - \dot{s}^2 d'_v(s) \leq d_v(s)\ddot{s} \leq \dot{v}_{max} - \dot{s}^2 d'_v(s)$$
$$-\dot{v}_{max} \leq d_v(s)\ddot{s} + \dot{s}^2 d'_v(s) \leq \dot{v}_{max}$$

Cette dernière inégalité (qui est l'inégalité 4.15) nous assure que la contrainte sur l'accélération linéaire maximale du robot est satisfaite sous les conditions (4.18) et (4.19). On peut faire le même raisonnement sur l'accélération angulaire maximale et ainsi nous obtenons :

$$|\ddot{s}| \leq (1-\dot{s}^2)\min\left(\frac{\dot{v}_{max}}{v_{max}}, \frac{\dot{\omega}_{max}}{\omega_{max}}\right)$$

Dans le plan de phase cette inégalité devient

$$\left|\frac{d\dot{s}}{ds}\right| \leq A\frac{1-\dot{s}^2}{\dot{s}}$$

où $A = \min\left(\frac{\dot{v}_{max}}{v_{max}}, \frac{\dot{\omega}_{max}}{\omega_{max}}\right)$.

Il faut remarquer que les termes $d_v(s)$ et $d_\omega(s)$ ont disparu. Le résultat que nous allons obtenir est donc indépendant de la trajectoire considérée.

Considérons la borne inférieure de cette inégalité qui correspond à une décélération maximale du robot.

$$\frac{d\dot{s}}{ds} = -A\frac{1-\dot{s}^2}{\dot{s}}$$

En l'intégrant nous obtenons

$$\dot{s} = \sqrt{1-(1-\dot{s}_0^2)\,e^{2A(s-s_0)}} \tag{4.21}$$

Ainsi, si le robot parcourt la trajectoire à une vitesse $\dot{s}_0$ et qu'il doit s'arrêter à l'abscisse s_1, nous cherchons l'abscisse s_2 à laquelle le robot doit suivre la courbe de décélération.

Nous résolvons donc l'équation (4.21) avec $\dot{s} = 0.0$, $s = s_1$, $s_0 = s_2$ et $\dot{s}_0$ fixé :

$$0 = 1-(1-\dot{s}_0^2)e^{2A(s_1-s_2)}$$

soit

$$s_2 = s_1 + \frac{1}{2A}\ln(1-\dot{s}_0^2)$$

Le robot doit donc ralentir selon le profil donné par la courbe (4.21) à partir de l'abscisse s_2.

On peut aussi tirer de l'équation (4.21) l'abscisse s_1 à laquelle le robot pourra s'arrêter en décélérant au maximum avec s_0 et $\dot{s}_0$ donnés :

$$s_1 = \frac{1}{2A}\ln(1-\dot{s}_0^2) + s_0$$

Conclusion

Nous pouvons donc maintenant calculer en temps constant et très court l'abscisse de début de décélération s_2 et l'abscisse d'arrêt s_1 correspondant.

La tâche tFollow calcule alors l'abscisse s_{stop} où le robot s'arrêtera réellement de la manière suivante :

$$s_{stop} = \min(s_1, s_{hide}, s_{start}) \tag{4.22}$$

Ainsi, la tâche de déformation est assurée que le robot ne franchira pas l'abscisse s_{stop}.

4.3.3 tDeform : déformons l'intervalle de trajectoire

Cette tâche calcule la déformation de trajectoire sur l'intervalle $[s_{stop}, s_{end}]$.

Il y a trois étapes :

1. calculer les forces dans l'espace des configurations à partir des obstacles perçus (voir chapitre 3),
2. calculer la direction de déformation η (voir chapitre 2),
3. appliquer la déformation à la trajectoire courante $\Phi(s, \tau + \Delta\tau) = \Phi(s, \tau) + \Delta\tau\, \bar{\eta}^{\perp}(s, \tau)$ (voir section 2.2.7).

4.4 Expérimentation : suivi d'une trajectoire

Dans cette section nous présentons un exemple de trajectoire réellement exécutée par le robot et des courbes illustrant le déroulement de l'algorithme présenté dans ce chapitre.

La trajectoire planifiée est présentée sur la figure 4.5. Elle est d'environ 30m et comporte 2 passages étroits au niveau du passage des portes. Elles ont une largeur de $1,30m$ alors que le robot mesure $0,75m$ de large. La figure 4.7 présente une photo de Hilare 2 muni de sa remorque dans le passage étroit où ont été acquises les données présentées sur la figure 4.6.

Durant cette expérience la tâche cCheck ne recherche que l'abscisse curviligne s_{coll} de la première configuration en collision. Le seuil de tolérance est fixé à 7cm.

L'abscisse s_{start} est calculée à une distance constante en aval de l'abscisse de collision s_{coll}.

Le robot utilisé est Hilare 2 avec sa remorque (voir figure 4.7). La vitesse linéaire maximale v_{max} est de $0,12m/s$ et la vitesse angulaire maximale ω_{max} est de $0,15rad/s$. Les accélérations maximales sont $\dot{\omega}_{max} = 0,1rad/s^2$ et $\dot{v}_{max} = 0,1m/s^2$

Le télémètre laser utilisé a une précision de 15mm pour une portée de 30m et une résolution angulaire de $0,5^o$.

Les courbes de la figure 4.6 montrent l'évolution de la vitesse de l'abscisse curviligne $\dot{s}$ (noté velocityRate sur la figure) ainsi que les instants d'exécution des différentes tâches cCheck, tFollow et tDeform lorsque le robot traverse la zone marquée "zone d'acquisition des données".

Le robot exécute la trajectoire 2 fois. La première fois (courbe du haut sur la figure 4.6) s_{stop} est calculée $3.0m$ avant la première configuration en collision détectée. La deuxième fois (courbe du bas sur la figure 4.6) s_{stop} est calculée $2.0m$ avant la première configuration en collision détectée.

Sur ces courbes, nous pouvons voir les différentes étapes présentées précédemment. À l'étape (1) cCheck cherche s_{coll} et calcule s_{start}. Ensuite, à l'étape (2) tFollow détermine le profil de vitesse du robot et indique à tDeform s_{stop} qui déforme, si nécessaire, la trajectoire à l'étape (3). Ces étapes sont répétées jusqu'à ce que le robot arrive à son but.

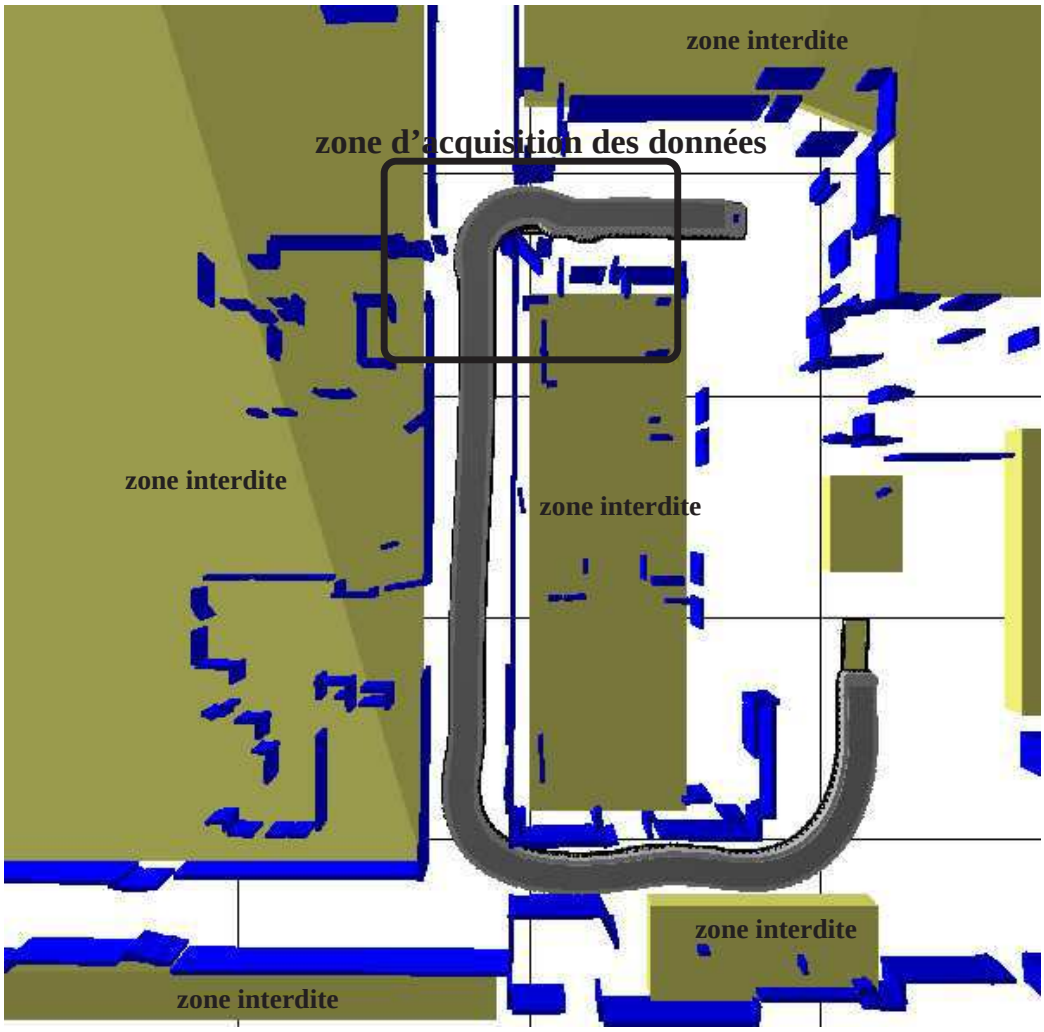

FIG. 4.5 – La trajectoire planifiée dans l'environnement modélisé. Le cadre, noté "zone d'acquisition des données", représente la zone où les données de la figure (4.6) ont été acquises.

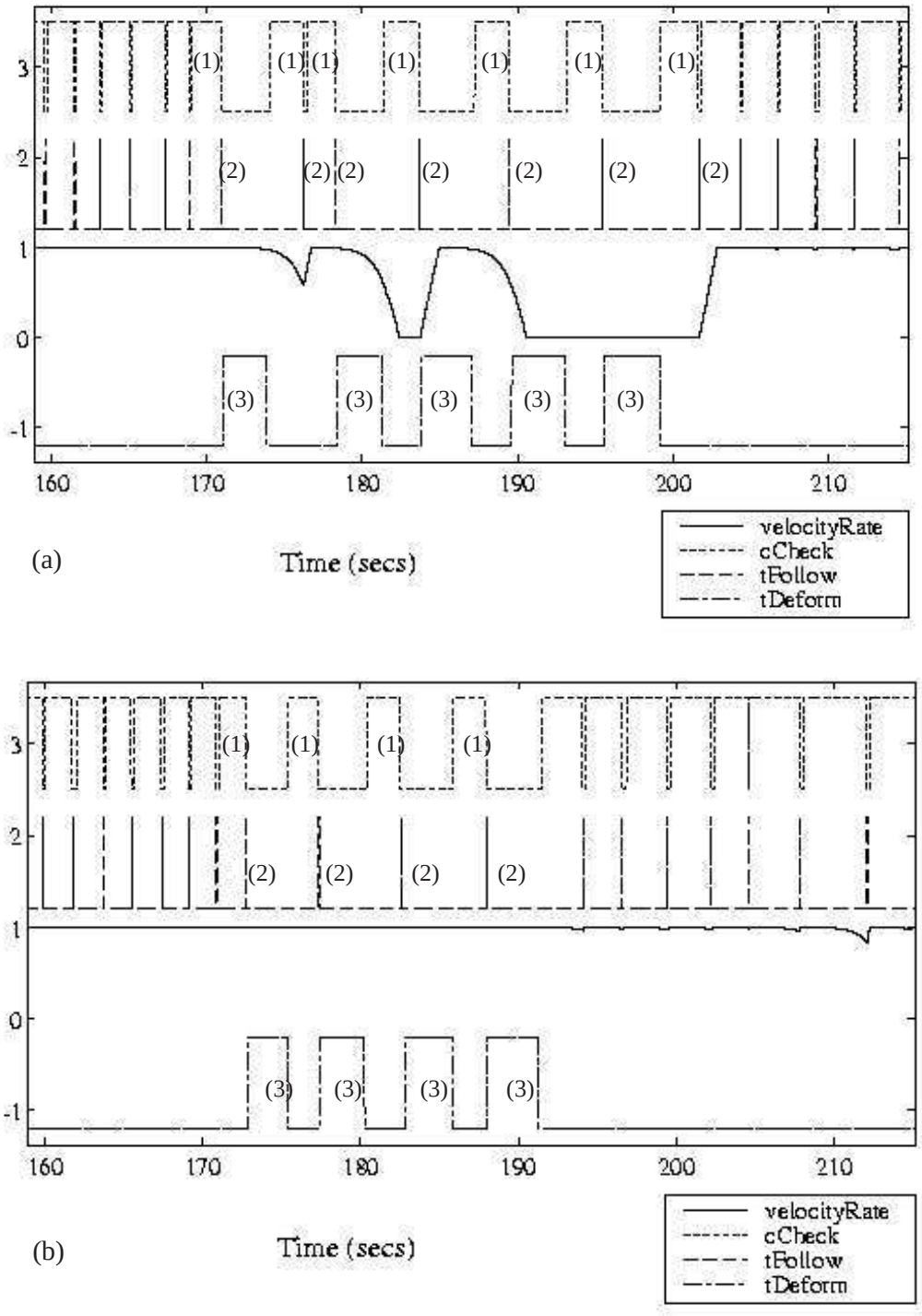

FIG. 4.6 – Chronogramme de l'exécution des tâches cCheck, tFollow et tDeform ainsi que la vitesse de l'abscisse curviligne $\dot{s}$ notée velocityRate. Les courbes du haut correspondent à s_{stop} calculé à 3m avant s_{coll}, les courbes du bas à 2m. Les données ont été acquises dans la "zone d'acquisition des données" sur la figure (4.5).

L'influence de la méthode de calcul de s_{stop} est visible sur le profil de vitesse de l'abscisse curviligne $\dot{s}$. Sur le premier tracé on constate que le robot ralentit et même s'arrête sur la trajectoire alors que sur le second tracé la vitesse de l'abscisse curviligne reste quasi constante (c'est à dire que le robot parcourt la trajectoire à vitesse nominale). Cela est dû au fait que dans le premier cas, l'intervalle de déformation est plus grand que dans le second. La déformation prend donc plus de temps et le robot arrive plus rapidement à l'abscisse s_{stop}.

FIG. 4.7 – Le robot Hilare 2 et sa remorque lors du passage de la porte dans la zone d'acquisition des données (voir figure 4.5)

4.5 Expérimentation : déformation d'une trajectoire

Dans ce paragraphe, nous présentons la déformation d'une trajectoire executée par le robot.

La figure 4.8 montre la trajectoire planifiée dans l'environnement modélisé. Le robot suit un couloir et effectue un virage à 90^o en franchissant deux portes (voir figure 4.7).

La photo de la figure 4.9 montre le couloir lors de l'exécution de la trajectoire. Il faut noter que les armoires présentes contre le mur *ne sont pas* dans le plan utilisé lors de la planification et que la trajectoire planifiée est en collision avec ces armoires.

FIG. 4.8 – La trajectoire initiale planifiée dans l'environnement modélisé. La position initiale du robot est en haut de l'image et le but est en bas.

Le robot doit donc modifier la trajectoire initiale pour pouvoir atteindre son but.

La trajectoire déformée, telle que le robot l'a executée, est présentée sur la figure 4.10. On peut voir sur cette figure que le processus de déformation a repoussé la trajectoire au loin des armoires.

La figure 4.11 montre les variables de configuration le long de la trajectoire initiale et déformée. On peut voir sur ces courbes la déformation subie par la trajectoire dans l'espace des configurations.

FIG. 4.9 – Le couloir lors de l'exécution de la trajectoire. Les armoires contre le mur de gauche ne sont pas dans le plan utilisé pour la planification de trajectoire.

FIG. 4.10 – La trajectoire finale déformée telle que le robot l'a executée.

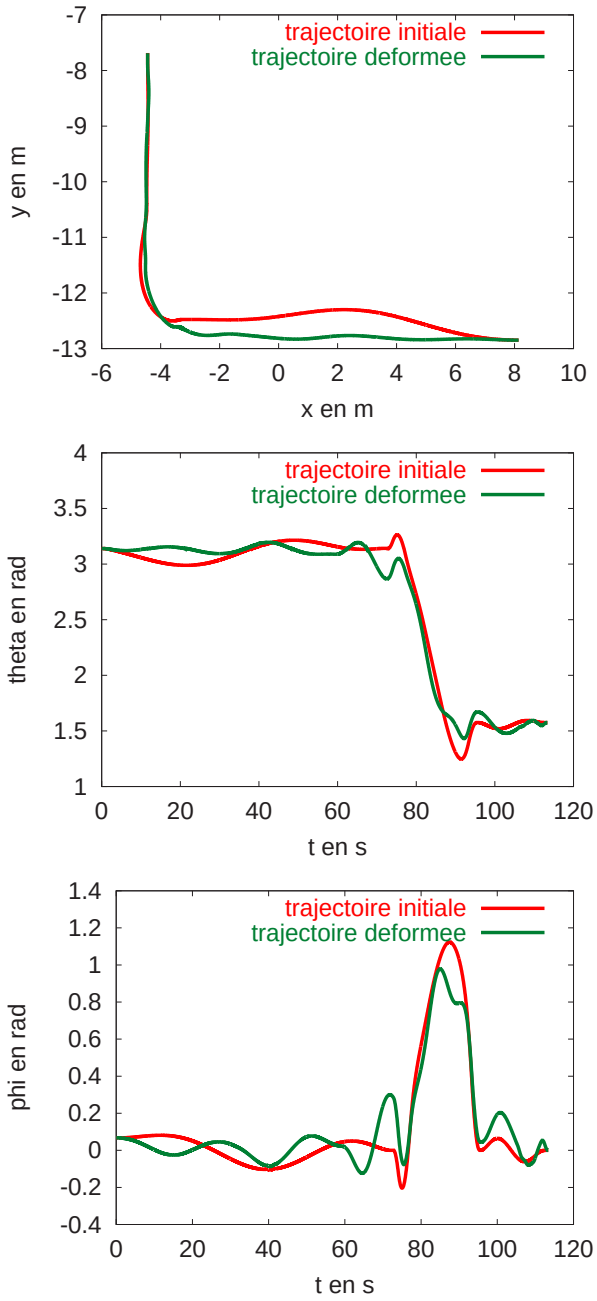

FIG. 4.11 – La trajectoire initiale et finale. Les courbes du haut montrent la position (x, y) du robot dans le plan pour la trajectoire initiale et finale, les courbes du milieu montrent l'orientation du robot et les courbes du bas montrent l'orientation de la remorque par rapport au robot.

Les courbes de la figure 4.12 montrent les commandes le long de la trajectoire initiale et déformée.

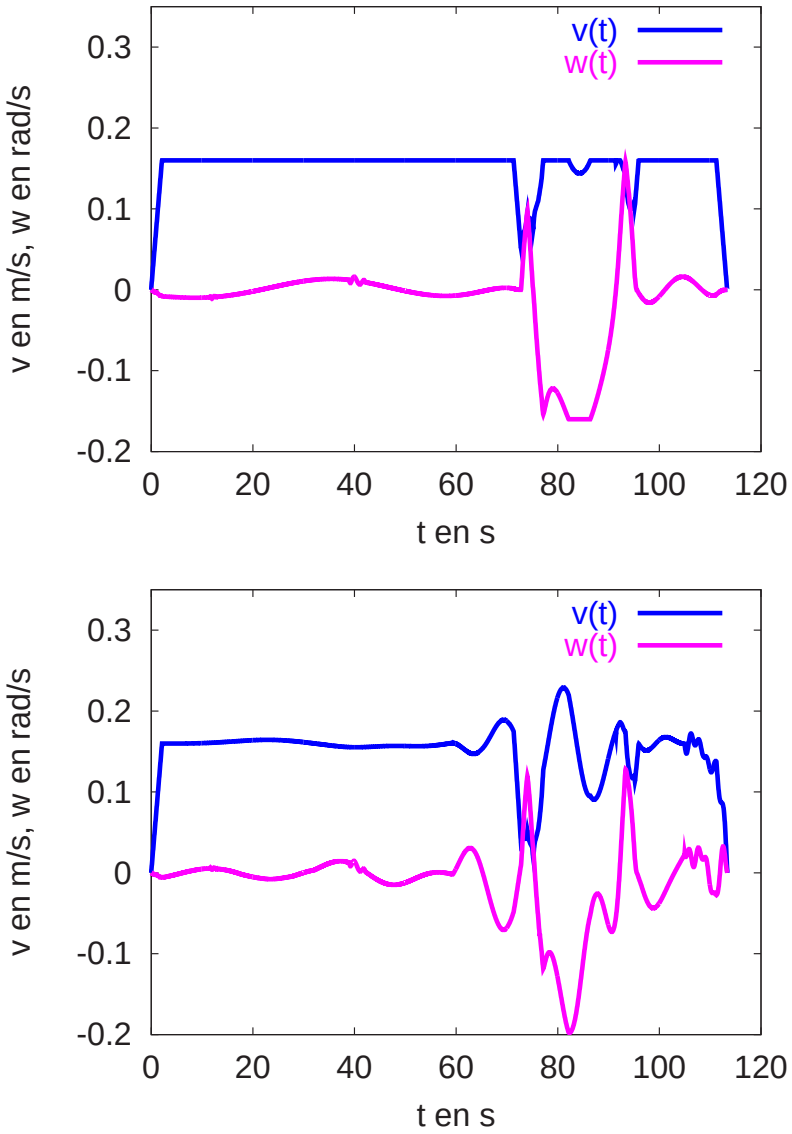

FIG. 4.12 – Commandes le long de la trajectoire initiale (en haut) et le long de la trajectoire déformée (en bas). $v(t)$ est la vitesse linéaire et $w(t)$ est la vitesse angulaire.

La figure 4.13 illustre la régulation autour de 0 des commandes le long des champs de vecteurs interdits.

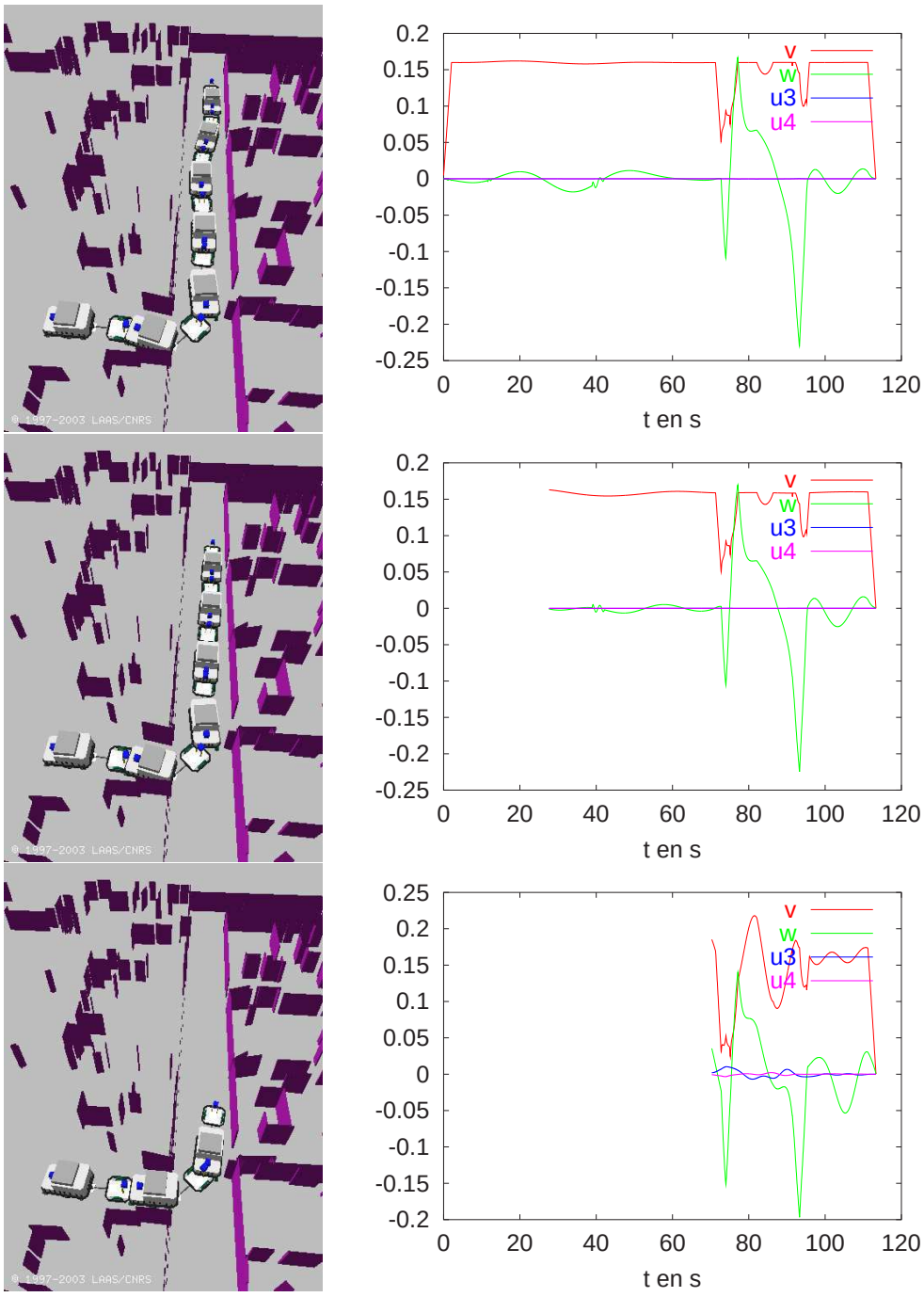

FIG. 4.13 – Régulation autour de 0 des commandes le long des champs de vecteurs interdits. Même en fin de trajectoire (courbes du bas), après 18 itérations de déformation, les commandes, notées u_3 et u_4 le long des champs de vecteurs interdits restent proches de 0 (courbes du bas).

Conclusion

Bilan

Déformation de trajectoire

Nous avons développé une méthode permettant de déformer une trajectoire d'un système non-holonome sous l'action de forces virtuelles créées par les obstacles perçus. La trajectoire et les forces s'expriment dans l'espace des configurations du système et la méthode s'appuie uniquement sur la loi d'évolution de la configuration du système en fonction des commandes appliquées. La trajectoire déformée respecte les contraintes non-holonomes du système dans la mesure où les mouvements interdits introduits par les approximations faites sont maintenus à un niveau acceptable.

Définition du potentiel d'une trajectoire

Nous avons présenté la définition du potentiel d'une trajectoire. Ce potentiel définit sur l'espace des configurations du robot permet de déterminer les forces virtuelles utilisées par la méthode de déformation de trajectoire. Les méthodes de calcul du potentiel à partir de points ou de segments perçus par les capteurs proximétriques du robot et pour différents types de robots sont présentées.

Mise en œuvre à bord du robot Hilare 2

La mise en œuvre à bord d'un robot d'expérimentation n'est pas immédiate. Nous avons développé une architecture logicielle composée de différentes tâches permettant la déformation de la trajectoire du robot pendant

son exécution. La détermination d'une borne supérieure de l'intervalle de trajectoire nécessaire pour arrêter le robot en respectant les capacités d'accélération du robot est utilisée par les différentes tâches pour garantir la progression du robot le long de sa trajectoire.

Perspectives

Généricité de la méthode de déformation

La méthode de déformation a été initialement créée pour permettre à un robot mobile de suivre effectivement une trajectoire planifiée dans un monde virtuel. Sa conception générique lui procure un potentiel d'utilisation beaucoup plus vaste. Elle a déjà été utilisée pour l'optimisation de trajectoire pour des systèmes très complexes (voir section 3.3.3). Elle commence à être utilisée pour résoudre des problèmes de planification de mouvements et certainement d'autres applications sont à venir.

Détecter les échecs du suivi de trajectoire

La méthode d'exécution réactive de trajectoire développée dans ce manuscrit se base sur des perceptions locales de l'environnement du robot. Elle peut donc être mise en échec dans certaines situations comme par exemple lorsque la trajectoire planifiée traverse une porte fermée modélisée ouverte dans l'environnement de planification. De telles situations ne sont actuellement pas détectées. Le développement de critères de pertinence de la déformation (par exemple la longueur de la trajectoire déformée par rapport à la trajectoire initiale ou une analyse de la situation à un niveau sémantique plus élevé) permettrait de détecter ces situations et de prendre les décisions adéquates comme la planification d'une nouvelle trajectoire.

Utiliser d'autres sources de données

Le chapitre 3 présente l'utilisation de deux sources de données pour le calcul du potentiel d'une trajectoire : des points et des segments issus d'un télémètre laser à balayage horizontal. L'utilisation de stéréo-vision devrait être envisagée car elle apporterait à notre approche du suivi réactif de trajectoire des données proximétriques plus riches (volumes en 3 dimensions) et nous pourrions ainsi éviter, par exemple, des angles de bureau saillants... Le passage à la stéréo-vision nécessite une optimisation préalable du calcul des interactions entre l'environnement et la trajectoire.

Bibliographie

[Bemporad 1996] A. Bemporad, A. De Luca & G. Oriolo. *Local incremental planning for a car-like robot navigation among obstacles.* IEEE International Conference on Robotics and Automation, pages 1205–1211. Mineapolis, USA, 1996.

[Bobrow 1985] J.E. Bobrow, S. Dubowsky & J.S. Gibson. *Time-Optimal Control of Robotic Manipulators along Specified Paths.* International Journal of Robotics Research, vol. 4, pages 3–17, 1985.

[Bonnafous 2001] D. Bonnafous, S. Lacroix & T. Simeon. *Motion generation for a rover on rough terrains.* IEEE International Conference on Intelligent Robots and Systems, pages 784–789. Maui, Hawaii, USA, novembre 2001.

[Brock 1999] Olivier Brock & Oussama Khatib. *High-Speed Navigation Using the Global Dynamic Window Approach.* IEEE International Conference on Robotics and Automation, pages 341–346. Detroit, Michigan, USA, 1999.

[Camargo 1991] R. Ferraz De Camargo. *Architecture matérielle et logicielle pour le contrôle d'exécution d'un robot mobile autonome.* Thèse de Doctorat, Université Paul Sabatier, Toulouse, France, Juillet 1991.

[De Luca 1998] A. De Luca, G. Oriolo & C. Samson. Robot motion planning and control, chapitre 4, Feedback Control of a Nonholonomic Car-like Robot, pages 171–253. J.P. Laumond Editor, Springer, NY, 1998.

[Divelbiss 1997] A. Divelbiss & J. Wen. *A Path Space Approach to Nonholonomic Motion Planning in the Presence of Obstacles.* IEEE Transactions on Robotics and Automation, vol. 13, n° 3, pages 443–451, June 1997.

[Fiorini 1998] P. Fiorini & Z. Shiller. *Motion Planning in Dynamic Environments Using Velocity Obstacles.* International Journal of Robotics Research, vol. 17, n° 7, pages 760–772, 1998.

[Fleury 1996] S. Fleury. *Architecture de contrôle distribuée pour robots mobiles autonome : principes, conception et application.* Thèse de Doctorat, Université Paul Sabatier, Toulouse, France, Juillet 1996.

[Fox 1997] Dieter Fox, Wolfram Burgard & Sebastian Thrun. *The Dynamic Window Approach to Collision Avoidance.* IEEE Robotics and Automation Magazine, vol. 4, n° 1, pages 23–33, 1997.

[Hayet 2003] Jean-Bernard Hayet. *Contribution à la navigation d'un robot mobile sur amers visuels texturés dans un environnement structuré.* Thèse de Doctorat, Université Paul Sabatier, Toulouse, France, 2003.

[Holmberg 2000] Robert Holmberg & Oussama Khatib. *Development and Control of a Holonomic Mobile Robot for Mobile Manipulation Tasks.* International Journal of Robotics Research, vol. 19, n° 11, pages 1066–1074, november 2000.

[Khatib 1996] Maher Khatib. *Contrôle du mouvement d'un robot mobile par retour sensoriel.* Thèse de Doctorat, Université Paul Sabatier de Toulouse, 1996.

[Khatib 1997] M. Khatib, H. Jaouni, R. Chatila & J.P. Laumond. *Dynamic Path Modification for Car-Like Nonholonomic Mobile Robots.* IEEE International Conference on Robotics and Automation, pages 2920–2925. Albuquerque, USA, 1997.

[Koenig 2002] Sven Koenig & Maxim Likhachev. *Improved Fast Replanning for Robot Navigation in Unknown Terrain.* IEEE International Conference on Robotics and Automation, pages 968–975. Washington D.C., USA, 2002.

[Large 2000] F. Large, S. Sekhavat, C.Laugier & E. Gauthier. *Towards Robust Sensor-Based Maneuvers for a Car-Like Vehicle.* IEEE International Conference on Robotics and Automation, pages 3765–3770. San Francisco, CA, USA, 2000.

[Large 2003] Frédéric Large. *Navigation Autonome d'un robot mobile en environnement dynamique et incertain.* Thèse de Doctorat, Université de Savoie, 2003.

[Latombe 1991] Jean-Claude Latombe. Robot motion planning, pages 320–323. Kluwer Academic Publishers, 1991.

[Laugier 1999] C. Laugier, T. Fraichard, Ph. Garnier, I.E. Paromtchik & A.Scheuer. *Sensor-based control architecture for a car-like vehicle.* Autonomous Robots, vol. 6, n° 2, pages 165–185, April 1999.

[Mínguez 2000] J. Mínguez & L. Montano. *Nearness Diagram Navigation (ND) : A New Real Time Collision Avoidance Approach for Holonomic and no Holonomic Mobile Robots.* IEEE International Conference on Intelligent Robots and Systems, pages 2094–2100. Kagawa University, Takamatsu, Japan, 2000.

[Mínguez 2002a] J. Mínguez, L. Montano & J. Santos-Victor. *Reactive Navigation for Non-holonomic Robots using the Ego Kinematic Space.* IEEE International Conference on Robotics and Automation, pages 3074–3080. Washington D.C., USA, 2002.

[Mínguez 2002b] J. Mínguez & Luis Montano. *Robot navigation in very complex, dense, and cluttered indoor/outdoor environnement.* IFAC World Congress on Automatic Control. Barcelona, Spain, July 2002.

[Nilsson 1971] Nils J. Nilsson. Problem-solving methods in artificial intelligence, pages 57–59. McGraw-Hill, 1971.

[Podsdkowski 1998] L. Podsdkowski. *Path Planner for nonholonomic mobile robot with fast replanning procedure.* IEEE International Conference on Robotics and Automation, vol. 2, pages 3588–3593. Leuven, Belgium, 1998.

[Podsdkowski 2001] L. Podsdkowski, J. Nowakowski, M. Idzikowski & I. Vizvary. *A new solution for path planning in partially known or unknown environment for nonholonomic mobile.* Robotics and Autonomous Systems, vol. 34, n° 2, pages 145–152, 2001.

[Quinlan 1993] S. Quinlan & O. Khatib. *Elastic Bands : Connecting Path Planning and Control.* IEEE International Conference on Robotics and Automation, vol. 2, pages 802–807. Atlanta, USA, 1993.

[Quinlan 1994] Sean Quinlan. *Real-Time Modication of CollisionFree Paths*. Thèse de Doctorat, Stanford University, 1994.

[Reeds 1990] J.A Reeds & R.A. Shepp. *Optimal path for a car that goes both forward and backwards*. Pacific Journal of Mathematics, vol. 145, pages 367–393, 1990.

[Scheuer 1997] A. Scheuer & Th. Fraichard. *Continuous-curvature path planning for car-like vehicles*. IEEE International Conference on Intelligent Robots and Systems, pages 997–1003. Grenoble, France, 1997.

[Simeon 2001] T. Simeon, J.-P. Laumond & F. Lamiraux. *Move3D : a generic platform for path planning*. International Symposium on Assembly and Task Planning, pages 25–30. Fukuoka, Japon, May 2001.

[Stentz 1994] A. Stentz. *Optimal and Efficient Path Planning for Partially-Known Environments*. IEEE International Conference on Robotics and Automation, San Diego, California USA, pages 3310–3317, May 1994.

[Stentz 1995] A. Stentz. *The Focussed D* Algorithm for Real-Time Replanning*. International Joint Conference on Artificial Intelligence, pages 1652–1659. Montréal, Québec, Canada, August 1995.

[Vendittelli 1999] M. Vendittelli, J.P. Laumond & C. Nissoux. *Obstacle distance for car-like robots*. IEEE Transaction on Robotics and Automation, vol. 15, n° 4, 1999.

[Xu 2003] F. Xu, H. Van Brussel, M. Nuttin & R. Moreas. *Concepts for dynamic obstacle avoidance and their extended application in underground navigation*. Robotics and Autonomous Systems, vol. 42, pages 1–15, 2003.