Développement et expérimentation d'algorithmes de réorientation pour un robot sériel en chute libre

Mémoire

Jean-Alexandre Bettez-Bouchard

Maîtrise en génie mécanique
Maître ès sciences (M.Sc.)

Québec, Canada

Développement et expérimentation d'algorithmes de réorientation pour un robot sériel en chute libre

Mémoire

Jean-Alexandre Bettez-Bouchard

Sous la direction de:

Clément Gosselin, directeur de recherche

Résumé

Ce mémoire présente 2 types de méthodes pour effectuer la réorientation d'un robot sériel en chute libre en utilisant les mouvements internes de celui-ci. Ces mouvements sont prescrits à partir d'algorithmes de planification de trajectoire basés sur le modèle dynamique du robot. La première méthode tente de réorienter le robot en appliquant une technique d'optimisation locale fonctionnant avec une fonction potentielle décrivant l'orientation du système, et la deuxième méthode applique des fonctions sinusoïdales aux articulations pour réorienter le robot. Pour tester les performances des méthodes en simulation, on tente de réorienter le robot pour une configuration initiale et finale identiques où toutes les membrures sont alignées mais avec le robot ayant complété une rotation de 180 degrés sur lui-même. Afin de comparer les résultats obtenus avec la réalité, un prototype de robot sériel plan flottant possédant trois membrures et deux liaisons rotoïdes est construit. Les expérimentations effectuées montrent que le prototype est capable d'atteindre les réorientations prescrites si peu de perturbations extérieures sont présentes et ce, même si le contrôle de l'orientation est effectué en boucle ouverte.

Abstract

This master's thesis presents two different types of methods to reorient a free-floating serial manipulator with internal motion using path planning algorithms based on a dynamic model of the manipulator. The first method attempts to reorient the robot with a local optimisation technique using a potential function describing the global orientation of the robot, while the second method applies sinusoidal functions to the joints of the robot in order to reorient it. The proposed methods are tested with a robot that starts from a pose in which all the links are aligned and ends with the same configuration but with the robot having completed a 180 degrees rotation. To verify the simulation results against a real robot, a prototype of a planar robot with three bodies and two revolute joints is built. The experiments conducted show that the prototype is able to achieve the prescribed reorientation if almost no external torque is applied to the system, even though the control of the orientation is implemented in an open-loop mode.

Table des matières

Résumé iii

Abstract iv

Table des matières v

Liste des tableaux vii

Liste des figures viii

Remerciements xi

Introduction 1

1 Développement du modèle dynamique du système 5
1.1 Mise en contexte . . . 5
1.2 Développement des équations cinématiques et dynamiques . . . 6
1.3 Validation du modèle dynamique . . . 10
1.4 Reformulation du modèle . . . 12
1.5 Conclusion . . . 13

2 Développement de l'algorithme de réorientation 14
2.1 Algorithmes de réorientation par minimisation d'une fonction potentielle . . . 14
2.1.1 Description de l'algorithme principal . . . 15
2.1.1.1 Construction de l'algorithme . . . 15
2.1.1.2 Limitation des vitesses articulaires . . . 17
2.1.2 Algorithme en mode hors-ligne . . . 20
2.1.2.1 Construction de l'algorithme . . . 20
2.1.2.2 Simulation de réorientation pour un robot plan à 3 membrures et 2 liaisons rotoïdes sans limites articulaires . . . 23
2.1.2.3 Simulation de réorientation pour un robot plan à 3 membrures et 2 liaisons rotoïdes avec limites articulaires . . . 28
2.1.2.4 Discussion sur les résultats pour le robot à 3 membrures et 2 liaisons rotoïdes . . . 31
2.1.2.5 Simulation de réorientation en 3 dimensions pour un robot à 4 membrures et 3 liaisons rotoïdes . . . 32
2.1.2.6 Discussion sur les résultats pour le robot à 4 membrures et 3 liaisons rotoïdes . . . 36

2.1.3 Algorithme adaptatif . . . 37
2.1.3.1 Construction de l'algorithme . . . 37
2.1.3.2 Simulation de réorientation pour un robot plan à 3 membrures et 2 liaisons rotoïdes . . . 39
2.1.3.3 Discussion . . . 41
2.2 Méthode basée sur l'application de fonctions sinusoïdales aux articulations . 42
2.2.1 Simulation de trajectoires sinusoïdales pour un robot à 3 membrures et 2 liaisons rotoïdes . . . 42
2.2.2 Discussion . . . 48
2.3 Conclusion . . . 48

3 Construction d'un prototype et validation expérimentale 50
3.1 Développement du prototype . . . 50
3.1.1 Construction du prototype . . . 50
3.1.2 Considérations géométriques . . . 53
3.2 Validation expérimentale . . . 54
3.2.1 Résultats expérimentaux avec la méthode hors-ligne . . . 54
3.2.2 Discussion . . . 56
3.3 Conclusion . . . 57

Conclusion 59

Bibliographie 61

A Développement du modèle dynamique pour un robot à 3 membrures et 2 liaisons rotoïdes 63

Liste des tableaux

2.1 Valeur initiale des paramètres de simulation qui permettent de réorienter le robot dans la meilleure orientation finale. 24
2.2 Valeur initiale des paramètres de simulation qui permettent de réorienter le robot dans la meilleure orientation finale pour une réorientation avec limite de débattement angulaire au niveau des articulations. 30
2.3 Valeur initiale des paramètres de simulation qui permettent de réorienter le robot dans la meilleure orientation finale pour un robot évoluant en 3D. 34
2.4 Valeur des 4 paramètres déterminés par la fonction fmincon pour la réorientation avec la méthode de fonction sinus aux articulations. 44

3.1 Valeur des masses en kg de chacune des membrure du prototype 52

Liste des figures

1.1 Définition des vecteurs de construction du robot. 8
1.2 Définition des vecteurs barycentriques pour chacune des membrures du robot. . 9
1.3 Vitesse angulaire $\boldsymbol{\omega}_1$ de la première membrure selon l'axe Z obtenue avec le logiciel Adams et avec le modèle présenté dans cette section, programmé dans Matlab. 11
1.4 Différence de vitesse angulaire $\boldsymbol{\omega}_1$ selon l'axe Z entre les deux modélisations. . 12

2.1 Définition des vecteurs qui représentent l'orientation actuelle $\boldsymbol{s}_{ir}$ de chaque membrure i et leur orientation finale désirée $\boldsymbol{s}_{i0}$. 16
2.2 Représentation de la boîte de contrainte des vitesses articulaires pouvant être appliquées aux articulations pour un pas de l'optimisation donné. 18
2.3 Représentation de la remise à niveau du vecteur $\dot{\boldsymbol{\theta}}$ lorsque sa norme est trop grande et qu'il intersecte la boîte de contrainte. 19
2.4 Représentation de la possibilité de satisfaire (à gauche) ou non (à droite) la contrainte d'accélération quand la vitesse est près de 0. 19
2.5 Défintion des angles γ_1 et γ_2. 21
2.6 Structure de l'algorithme hors-ligne. 22
2.7 Procédure de la fonction $fmincon$ avec l'algorithme hors-ligne. 23
2.8 Orientation initiale et finale désirée pour la réorientation du robot. 23
2.9 Visualisation de la réorientation du robot sans contraintes aux articulations. . . 25
2.10 Progression de la réorientation du robot exprimée par le scalaire η. 26
2.11 Trajectoire des vitesses articulaires à appliquer aux moteurs, déterminées par l'algorithme. 26
2.12 Progression des angles d'Euler pendant la réorientation selon la convention XYZ. 27
2.13 Déplacements articulaires pendant la réorientation. 27
2.14 Valeur finale de l'angle ψ obtenue selon le décalage de temps entre l'application des courbes de vitesses aux articulations. 28
2.15 Visualisation de la réorientation avec limites articulaires. 29
2.16 Progression du paramètre η pendant la réorientation avec limites articulaires aux liaisons rotoïdes. 30
2.17 Déplacements articulaires pendant la réorientation avec limites articulaires aux liaisons. 30
2.18 Représentation d'un robot à 4 membrures et 3 liaisons rotoïdes. 33
2.19 Orientation initiale et finale désirée pour la réorientation du robot. 33
2.20 Progression de la réorientation du robot exprimée par le scalaire η. 35
2.21 Trajectoire des vitesses articulaires déterminées par l'algorithme. 35
2.22 Progression des angles d'Euler pendant la réorientation selon la convention XYZ. 36

2.23 Déplacements articulaires pendant la réorientation. 36
2.24 Structure de l'algorithme en mode adaptatif. 39
2.25 Progression du paramètre η pendant la réorientation effectuée avec la méthode adaptative. 40
2.26 Vitesses articulaires pendant la réorientation effectuée avec la méthode adaptative. 40
2.27 Comparaison des résultats de la réorientation pour la méthode hors-ligne et adaptative selon une variation dans l'orientation initiale selon l'axe Z du robot. 41
2.28 Progression du scalaire η pour une période pendant la réorientation effectuée avec la méthode de fonction sinus aux articulations. 44
2.29 Visualisation de la réorientation des mouvements sinus aux articulations. . . . 45
2.30 Progression des angles d'Euler pour une période pendant la réorientation effectuée avec la méthode de fonction sinus aux articulations. 46
2.31 Déplacements articulaires des liaisons rotoïdes pour une période pendant la réorientation effectuée avec la méthode de fonction sinus aux articulations. . . 46
2.32 Vitesses articulaires pour une période pendant la réorientation effectuée avec la méthode de fonction sinus aux articulations. 47
2.33 Valeur finale de l'angle d'Euler ψ selon la valeur des amplitudes de mouvement des articulations. 47

3.1 a) Photo du prototype et b) vue de face du modèle CAO du prototype. 51
3.2 Progression des coordonnées articulaires pendant la réorientation. 55
3.3 Progression de l'orientation de la membrure centrale du prototype 56

Pour réussir, il ne suffit pas de prévoir. Il faut aussi savoir improviser.

Isaac Asimov

Remerciements

Je voudrais d'abord remercier Clément Gosselin pour m'avoir fait confiance et m'avoir permis de travailler sur un sujet aussi intéressant. Tu as été très présent pour moi et tes judicieux conseils ont su m'éclairer lorsque les problèmes semblaient insurmontables. Merci de m'avoir transmis ta passion pour la robotique, et je te souhaite un bon succès dans les autres grands projets qui t'attendent.

Un énorme merci aux professionnels de recherche, Thierry et Simon, ainsi qu'à tous les étudiants du laboratoire qui ont su répondre à toutes mes questions et m'apporter un support technique incroyable tout au long de mes travaux.

Je voudrais remercier ma famille pour toujours m'avoir encouragé et soutenu dans les projets que j'ai entrepris. Merci à mon père Luc pour m'avoir fait découvrir et adorer la science alors que j'étais très jeune, et à ma mère Judith pour m'avoir poussé dans le derrière en me disant que je pouvais toujours faire mieux.

Je voudrais aussi remercier le monde de la natation pour ces 18 merveilleuses années. À mes entraineurs de natation, Heather, Emmanuel et Nicholas, ainsi qu'à toutes les personnes avec qui j'ai partagé des aller-retours de piscine, vous avez parsemé mon quotidien de moments que je n'oublierai jamais.

Chacun à votre manière, vous avez contribué à me faire comprendre que l'atteinte de ses objectifs n'est possible qu'au prix d'un immense don de soi, mais que tous les efforts et sacrifices sont tellement plus faciles à faire lorsqu'on est bien entouré.

Introduction

Depuis le lancement du premier satellite dans l'espace en 1957 par l'URSS, suivi plus tard par les navettes spatiales et plus récemment avec la station spatiale internationale, la quantité de systèmes mécaniques en orbite autour de la Terre n'a cessé de s'accroître. Leur utilisation est aujourd'hui monnaie courante, puisqu'ils jouent un rôle clé dans une multitude de domaines, notamment en télécommunication, en recherche, en environnement et sur le plan militaire.

Bien que toutes ces technologies aient permis de faire des avancées significatives dans une foule de champs scientifiques, celles-ci ont de manière équivalente posé une quantité énorme de problèmes à résoudre afin de permettre aux différents appareils d'effectuer leurs missions de plus en plus sophistiquées. Outre les considérations techniques et le choix des matériaux appropriés, assurer que les appareils soient en mesure d'effectuer leurs tâches en assurant une bonne position et orientation de ceux-ci est tout aussi crucial. Par exemple, prendre une photo d'une zone précise dans l'espace à partir d'un satellite peut nécessiter une précision de la vitesse de rotation du système de l'ordre du millième de degré par seconde, sous peine d'obtenir des images floues. Cela illustre ainsi toute l'importance d'arriver à contrôler adéquatement ces paramètres.

Au fil des années, différentes stratégies et technologies ont été développées afin de parvenir à contrôler la position et l'orientation des appareils dans l'espace [1]. La position de l'orbite de l'appareil par rapport à la Terre est majoritairement ajustée en utilisant des propulseurs au gaz installés à des endroits stratégiques. Par ailleurs, ces mêmes propulseurs peuvent aussi contribuer à contrôler l'orientation en créant des couples externes sur le système afin de le faire tourner sur lui-même. Une autre technologie utilisée pour modifier l'orientation est le magnéto-coupleur, qui produit des couples externes sur l'appareil qui sont proportionnels à la force du champ magnétique de la Terre [1]. Cette technologie est de manière générale utilisée pour effectuer des réorientations grossières de l'appareil et pour stopper des mouvements non désirés [2]. Ensuite, une autre méthode utilisée pour réorienter des mécanismes spatiaux est l'utilisation de roues d'inertie disposées selon les trois axes du système [3]. Le changement de l'orientation est fait en modifiant la vitesse de rotation des roues. Finalement, on retrouve l'utilisation d'actionneurs gyroscopiques produisant des couples internes sur le système. Les couples obtenus avec cette technologie peuvent être très élevés et offrent un très bon rendement

énergétique comparé aux roues d'inertie. Ces actionneurs peuvent être utilisés autant pour les gros appareils que pour les plus petits satellites qui demandent des manœuvres nécessitant de la rapidité.

Cependant, bien que les manières d'utiliser toutes les technologies mentionnées aient été étudiées en profondeur, ces dernières posent tout de même quelques problèmes. D'abord, la capacité d'utiliser des propulseurs au gaz afin d'effectuer des manœuvres de modification de trajectoire ou de réorientation est plutôt limitée, car cette technologie consomme des ressources de carburant de l'appareil, qui sont en quantité restreinte dans l'espace dû à la difficulté d'approvisionnement. Dépenser du carburant signifie ainsi réduire la durée de vie utile du système. Ensuite, bien que l'utilisation de roues d'inertie, de magnéto-coupleurs et d'actionneurs gyroscopiques soit moins coûteuse en terme de ressource, celles-ci ajoutent de la masse au système, ce qui n'est pas nécessairement désiré car la masse totale de l'appareil est un facteur limitant des voyages dans l'espace. De plus, les différentes technologies mentionnées sont parfois utilisées en redondance avec d'autres afin d'améliorer la performance des manœuvres dans l'espace. Cela offre l'avantage de permettre le maintien des opérations même lorsqu'une composante se brise ou est défectueuse, mais présente aussi le désavantage d'ajouter de la masse au système.

Une autre méthode de réorientation qui peut être utilisée et qui ne présente pas les désavantages mentionnés est de combiner les mouvements des articulations de l'appareil afin de créer des changements d'orientation. Bien sûr, cette méthode est exclusivement adaptée pour les type d'appareils qui possèdent des manipulateurs, comme les navettes et certains satellites possédant une série d'articulations, ainsi que pour des robots terrestres articulés que l'on désirerait réorienter pendant une chute ou un saut. Le problème de réorientation d'un mécanisme articulé en apesanteur ou en chute libre utilisant cette méthode est un exemple de problème de planification de trajectoire soumis à des contraintes non-holonomes. Ce problème est défini physiquement par la loi de la conservation du moment angulaire dans lequel on peut identifier deux cas : celui où le moment angulaire au début de la réorientation est nul, et celui où il ne l'est pas.

D'abord, ce type de méthode de réorientation peut fréquemment être observé dans la vie courante. Par exemple, la manière dont les chats réussissent à se retourner sur eux-mêmes lorsqu'on les laisse tomber à partir de n'importe quelle position illustre bien cela. Kane et Scher [4] ont d'ailleurs étudié ce phénomène, et ils ont conclu que les chats sont capables d'accomplir de telles prouesses grâce à la grande flexibilité de leurs membres et articulations ainsi que par la bonne combinaison de mouvement de ceux-ci.

Dans la littérature, plusieurs techniques ont d'abord été proposées pour modéliser la cinématique et la dynamique de robots en chute libre. Parmi celles-ci, on retrouve la modélisation de la dynamique avec les équations de Kane [5], la modélisation à l'aide du concept de manipu-

lateur virtuel développé par Dubowsky [6] ainsi que la modélisation à l'aide d'une jacobienne généralisée dérivée à partir des équations de la conservation du moment angulaire de Yoshida [7]. Chacune de ces méthodes possède des avantages, et choisir l'une ou l'autre dépend des applications que l'on veut viser.

Par la suite, plusieurs méthodes ont été proposées afin de satisfaire les diverses contraintes que le problème de réorientation fait intervenir. Nakamura et Mukherjee [8] ont travaillé sur une méthode, dite bidirectionnelle, qui détermine des trajectoires articulaires à appliquer par les articulations du système en utilisant une fonction de Lyapunov. Plus tard, Suzuki et Nakamura [9] ont développé une stratégie de contrôle de l'orientation basée sur un mouvement spiral effectué par l'effecteur du manipulateur d'un robot. Papadopoulos et Dubowsky [10] se sont concentrés sur la faisabilité de la réorientation pouvant être effectuée par l'effecteur d'un manipulateur en définissant son espace de travail atteignable en orientation et en introduisant le concept de singularité dynamique.

Bien que jusqu'à présent il n'a été discuté que de réorientation dans l'espace, plusieurs travaux ont aussi été réalisés pour des situation faisant intervenir des robots qui évoluent à partir de la terre ferme. Effectivement, avec l'évolution des applications en robotique, on peut s'attendre à ce que la capacité de se réorienter en chute libre due à un saut ou à une chute devienne une caractéristique importante des robots [11]. Ainsi, Chang-Siu et al. [12] ont construit un véhicule qui peut modifier son orientation pendant un saut en contrôlant une queue attachée à l'arrière comme le fait un lézard. À partir du même principe, Zhao et al. [13] ont développé un robot sauteur qui peut se réorienter dans les airs avec une queue. Plus récemment, Bingham et al. [11] ont développé un algorithme pour déterminer quelle suite de configurations adopter par un robot en chute libre pour atterrir dans une pose qui minimise l'impact avec le sol. Ensuite, Yang et al. [14] ont développé des algorithmes de contrôle pour un système sous-actionné pour lui permettre d'atterrir sur le sol avec la configuration désirée. Murthy et Keerthi [15] ont aussi construit le modèle dynamique des manœuvres aériennes d'un plongeur et établi les lois de contrôle régulant celui-ci.

Ce mémoire se propose de fournir des solutions au problème de réorientation en utilisant les mouvements internes des membrures de l'appareil pour un moment angulaire initial nul. De plus, ces solutions sont destinées à permettre la réorientation de la totalité ou de plusieurs membrures du robot, et non seulement de son effecteur. On parle donc ici de la réorientation d'un robot à partir d'une configuration initiale donnée vers une configuration finale prescrite.

Les solutions fournies ont comme objectif d'être applicables pour plusieurs types de géométrie de robot, qu'ils soient spatiaux ou terrestres. Cependant, celles-ci ont été développées et optimisées pour un robot que l'on désire retourner à la manière d'un chat tenu à l'envers par les pattes et qu'on laisse tomber pour qu'il retombe sur celles-ci. Le retournement ainsi créé correspond à une réorientation de 180 degrés du système pour les mêmes configurations

initiales et finales des membrures.

Objectifs du mémoire

Ici sont présentés les problématiques qui tenteront d'être résolues dans ce mémoire. Celles-ci peuvent être formulées en trois parties distinctes.

1. Construire un modèle dynamique d'un robot sériel en chute libre possédant un moment angulaire initial nul afin de déterminer son comportement en orientation.
2. Développer un algorithme de planification de trajectoire pour les articulations d'un robot afin que celui-ci accomplisse une rotation de 180 degrés sur lui même et termine sa course dans la même configuration qu'au début de la réorientation.
3. Construire un prototype de robot sériel plan flottant afin de vérifier expérimentalement les résultats obtenus en simulation.

Structure du mémoire

Le premier chapitre de ce mémoire construit le modèle dynamique d'un robot sériel en chute libre afin de pouvoir exprimer le comportement de celui-ci lorsque des mouvements aux moteurs des articulations sont prescrits. Ce chapitre présente aussi une manière avantageuse de reformuler le modèle afin de le rendre apte à être utilisé par les algorithmes de planification de trajectoire qui sont présentés au chapitre suivant.

Le deuxième chapitre du mémoire propose deux types de stratégies de planification de trajectoire pour effectuer le retournement du robot. La première stratégie présentée repose sur des méthodes de descente de gradient qui tentent de minimiser une fonction potentielle. Pour ce type de méthode, plusieurs types d'algorithmes sont développés afin de répondre aux différentes circonstances, soit que l'on tente de déterminer les trajectoires articulaires pendant que la chute libre se produit, ou plutôt avant qu'elle ne se produise. La deuxième stratégie consiste à prescrire des mouvements sinusoïdaux aux articulations. Les résultats obtenus en simulation sont présentés pour chacune des méthodes décrites.

Le troisième chapitre du mémoire présente la conception du prototype de robot sériel plan flottant possédant trois membrures et deux liaisons rotoïdes afin de valider expérimentalement les résultats obtenus au chapitre précédent. Les considérations techniques nécessaires à sa réalisation ainsi que les performances pour chacun des algorithmes sont aussi discutées.

Le mémoire se termine avec une brève conclusion qui résume les différentes étapes suivies afin d'assurer la réussite des objectifs fixés. Une brève discussion des résultats obtenus est aussi faite.

Chapitre 1

Développement du modèle dynamique du système

Ce chapitre présente comment est construit le modèle dynamique d'un robot sériel sans base fixe. D'abord, les diverses équations de cinématique et de dynamique représentant le système sont exposées. Celles-ci sont ensuite combinées afin d'obtenir une formulation qui permet de faire le lien entre les variables articulaires et la vitesse angulaire de la première membrure du robot. Finalement, cette formulation est légèrement modifiée afin de faciliter le calcul des mouvements nécessaires à la réorientation du robot selon les différentes stratégies de réorientation proposées au chapitre suivant.

1.1 Mise en contexte

Le problème de position et d'orientation d'un robot sériel se pose d'une manière très différente selon si sa base est fixe ou non. En effet, pour un robot sériel possédant une base fixe, la position et l'orientation de chacune des membrures peut être déterminée grâce à une chaîne cinématique partant de la base du robot et terminant à son effecteur. Cette chaîne de vecteurs, pouvant être définie à l'aide des paramètres de Denavit-Hartenberg [16], permet de construire une matrice jacobienne qui fait le lien entre les vitesses articulaires du robot et la vitesse généralisée de l'effecteur.

Cependant, il en est tout autrement pour les robots sériels en chute libre, c'est-à-dire sans base fixe, puisque l'orientation et la position de la première membrure n'est pas constante dans le temps et dépend de la position et de l'orientation de toutes les autres membrures du mécanisme de par la loi de la conservation du moment angulaire. Il est donc de mise de poser les équations qui permettent d'effectuer des liens entre ces caractéristiques.

Ce mémoire se propose d'explorer le problème d'orientation des robots sans base fixe, puisqu'à priori la position de leur centre de masse global est constante dans un référentiel qui se déplace

avec celui-ci. De plus, puisque dans le cas présent on désire caractériser l'orientation de toutes les membrures du robot, et pas seulement celle de son effecteur, il est important que les modèles développés fassent ressortir les paramètres de l'orientation de toutes les membrures du système.

1.2 Développement des équations cinématiques et dynamiques

La cinématique et la dynamique d'un robot sériel en chute libre avec n membrures et j liaisons rotoïdes ($j = n - 1$) peut être exprimée par rapport à son centre de masse (CM) [17]. En considérant que la position de ce CM est constante dans un référentiel qui se déplace avec lui (noté R_0), la position $\boldsymbol{r}_i$ ainsi que la vitesse cartésienne $\boldsymbol{v}_i$ du centre de masse (CM_i) de chaque membrure $i\,(i = 1, \ldots, n)$ peut être directement reliée à la position et vitesse de chacune des autres membrures du mécanisme, ce qui peut être écrit comme

$$\sum_{i=1}^{n} m_i \boldsymbol{r}_i = \boldsymbol{0} \tag{1.1}$$

et

$$\sum_{i=1}^{n} m_i \boldsymbol{v}_i = \boldsymbol{0} \tag{1.2}$$

où m_i est la masse de la membrure i. Ces relations découlent directement de la loi de la conservation de la quantité de mouvement. De plus, la loi de la conservation du moment angulaire s'écrit

$$\frac{d\boldsymbol{h}}{dt} = \boldsymbol{0} \tag{1.3}$$

avec

$$\boldsymbol{h} = \sum_{i=1}^{n} (\boldsymbol{I}_i \boldsymbol{\omega}_i + m_i \boldsymbol{r}_i \times \boldsymbol{v}_i) \tag{1.4}$$

où $\boldsymbol{h}$ est le moment angulaire, $\boldsymbol{I}_i$ est le tenseur d'inertie de la membrure i et $\boldsymbol{\omega}_i$ est sa vitesse angulaire. Dans ce mémoire, on suppose que le robot est initialement au repos et qu'aucun couple externe n'est appliqué sur celui-ci tout au long de sa réorientation. De plus, l'effet de chute libre sur le robot annule l'action de la gravité sur celui-ci car il existe une compensation exacte entre les deux. On peut donc poser que $\boldsymbol{h}$ vaut $\boldsymbol{0}$.

Effectuer le calcul de l'équation (1.4) nécessite que tous les vecteurs et tenseurs soient exprimés dans le même repère inertiel, dans le présent cas R_0. Pour ce faire, on introduit les matrices de rotation $\boldsymbol{Q}_i$, pour $i = 1, \ldots, n$, qui font le changement de repère du référentiel i vers $i - 1$. La matrice $\boldsymbol{Q}_1$, qui fait le lien entre la première membrure du robot et le repère inertiel, est définie en utilisant la notation des angles d'Euler ϕ, ρ et ψ suivant la convention XYZ [17]. La matrice $\boldsymbol{Q}_1$ s'écrit de la manière suivante

$$\boldsymbol{Q}_1(\phi,\rho,\psi) = \begin{bmatrix} c_\psi c_\rho & -c_\rho s_\psi & s_\rho \\ c_\phi s_\psi + c_\psi s_\phi s_\rho & c_\phi s_\psi - s_\phi s_\psi s_\rho & -c_\rho s_\phi \\ s_\phi s_\psi - c_\phi c_\psi s_\rho & c_\psi s_\phi + c_\phi s_\psi s_\rho & c_\phi c_\rho \end{bmatrix} \tag{1.5}$$

où c and s correspondent respectivement à cos et sin. La matrice $\boldsymbol{Q}_1$ peut également être définie comme le produit de trois matrices de rotations, $\boldsymbol{M}$, $\boldsymbol{N}$ et $\boldsymbol{O}$, qui expriment respectivement les rotations des angles d'Euler autour des axes XYZ. Ces matrices sont définies comme suit

$$\boldsymbol{M} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \tag{1.6}$$

$$\boldsymbol{N} = \begin{bmatrix} \cos\rho & 0 & \sin\rho \\ 0 & 1 & 0 \\ -\sin\rho & 0 & \cos\rho \end{bmatrix} \tag{1.7}$$

$$\boldsymbol{O} = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{1.8}$$

Pour ce qui est de la matrice $\boldsymbol{Q}_i$, pour $i = 2, \ldots, n$, celle-ci est définie comme une rotation d'un angle θ_{i-1} autour d'un vecteur unitaire $\underline{\boldsymbol{e}}_{i-1}$ correspondant à la direction de l'axe de la liaison rotoïde $i-1$ du robot[18], et est exprimée comme

$$\boldsymbol{Q}_i = \underline{\boldsymbol{e}}_{i-1}\underline{\boldsymbol{e}}_{i-1}^T + c_i(\mathbf{1} - \underline{\boldsymbol{e}}_{i-1}\underline{\boldsymbol{e}}_{i-1}^T) + s_i\mathbf{1} \times \underline{\boldsymbol{e}}_{i-1} \quad i = 2, \ldots, n \tag{1.9}$$

où c_i and s_i correspondent respectivement à $\cos\theta_{i-1}$ et $\sin\theta_{i-1}$ et où $\mathbf{1}$ est la matrice identité.

La représentation dans R_0 des différents vecteurs et tenseurs est obtenue en pré-multipliant par le produit des matrices $\boldsymbol{Q}_1 \ldots \boldsymbol{Q}_i$ dans le cas d'un vecteur, et en post-multipliant aussi par le produit des transposées des matrices $\boldsymbol{Q}_i \ldots \boldsymbol{Q}_1$ dans le cas d'un tenseur. Par exemple, le vecteur $\boldsymbol{e}_i$ et le tenseur $\boldsymbol{I}_i$ sont représentés dans R_0 de la manière suivante

$$\boldsymbol{e}_i = (\prod_1^i \boldsymbol{Q}_i)\, \underline{\boldsymbol{e}}_i \tag{1.10}$$

$$\boldsymbol{I}_i = (\prod_1^i \boldsymbol{Q}_i)\, \underline{\boldsymbol{I}}_i\, (\prod_i^1 \boldsymbol{Q}_i^T). \tag{1.11}$$

Dans ce mémoire, afin d'alléger les équations et par soucis de simplicité, tous les vecteurs et tenseurs exprimés dans leur référentiel local sont soulignés.

L'équation (1.4) est suffisante pour décrire l'évolution de l'orientation d'un robot en chute libre durant une manœuvre de réorientation. Cependant, pour arriver à contrôler cette réorientation, il est nécessaire d'exprimer les vecteurs $\boldsymbol{r}_i$, $\boldsymbol{v}_i$ et $\boldsymbol{\omega}_i$ en terme du vecteur de contrôle du robot qui est le vecteur de vitesse articulaire de ses liaisons $\dot{\boldsymbol{\theta}} = [\dot{\theta}_1, \ldots, \dot{\theta}_j]^T$. Pour y parvenir, il est d'abord nécessaire de définir le concept de membrure augmentée [18]. Une membrure augmentée représente une membrure dont, pour chacune des liaisons qui lui sont rattachées, est attaché un point masse équivalant à la masse de toutes les autres membrures rattachées directement et indirectement à cette liaison. Par exemple, la membrure 3 d'une chaîne sérielle comprenant 6 membrures au total comprendrait 2 points masse valant respectivement $m_1 + m_2$ et $m_4 + m_5 + m_6$ et situés respectivement à la position des 2 liaisons qui lui sont rattachées. Une telle membrure ainsi créée possède la masse totale M du robot et représente un corps rigide qui ne varie pas selon les mouvements du système ni ceux de ses liaisons. Une membrure augmentée a de cette manière une densité de masse instantanée, et possède un centre de masse qui généralement ne correspond pas avec le CM_i de sa membrure. Ce centre de masse est appelé le barycentre CM_{Bi} de la membrure i. La position du barycentre par rapport au centre de masse de la membrure peut aisément être déterminée en définissant des paramètres de construction du robot. Pour chaque membrure du robot, deux vecteurs de construction sont définis, notés $\underline{\boldsymbol{r}}_{0i}$ et $\underline{\boldsymbol{l}}_{0i}$, comme montré à la figure 1.1. Le vecteur $\underline{\boldsymbol{r}}_{0i}$ relie le CM_i à la liaison i, tandis que le vecteur $\underline{\boldsymbol{l}}_{0i}$ relie le CM_i à la liaison $i-1$.

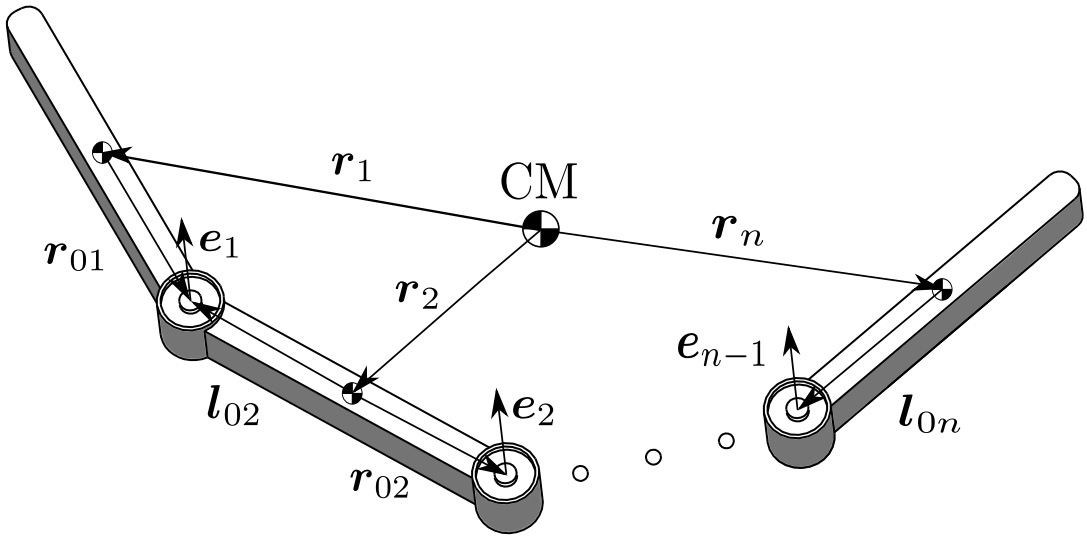


FIGURE 1.1 – Définition des vecteurs de construction du robot.

En combinant ces vecteurs avec la valeur des points masse définis pour chaque membrure, on obtient la position respective de leur barycentre par rapport à leur centre de masse [10] donnée par

$$\boldsymbol{c}_{0i} = \boldsymbol{l}_{0i}\mu_i + \boldsymbol{r}_{0i}(1 - \mu_{i+1}) \tag{1.12}$$

avec μ_i représentant la distribution de la masse dans le robot et déterminée selon la règle suivante

$$\mu_i = \begin{cases} 0 & i = 1 \\ \sum_{k=1}^{i-1} \frac{m_k}{M} & i = 2, \ldots, n \\ 1 & i = n+1 \end{cases} \tag{1.13}$$

où M est la masse totale du robot. Procéder ainsi permet de définir également d'autres vecteurs de construction, dits barycentriques, qui peuvent être exprimés par rapport au barycentre de la membrure. Ces vecteurs sont montrés à la figure 1.2 (tirée de Papadopoulos et Dubowsky [10].) et sont définis comme suit

$$\underline{\boldsymbol{c}}^*_{0i} = -\underline{\boldsymbol{c}}_{0i} \tag{1.14}$$

$$\underline{\boldsymbol{r}}^*_{0i} = \underline{\boldsymbol{r}}_{0i} - \underline{\boldsymbol{c}}_{0i} \tag{1.15}$$

$$\underline{\boldsymbol{l}}^*_{0i} = \underline{\boldsymbol{l}}_{0i} - \underline{\boldsymbol{c}}_{0i}. \tag{1.16}$$

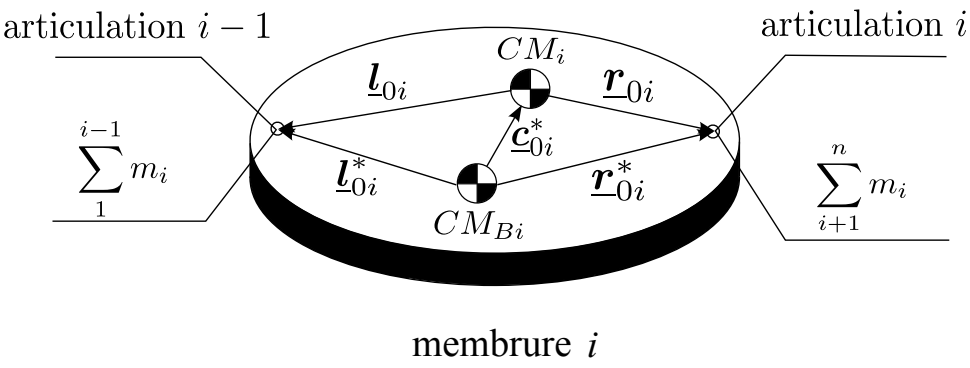


FIGURE 1.2 – Définition des vecteurs barycentriques pour chacune des membrures du robot.

Or, selon un développement mathématique qui est hors de la portée de ce mémoire, il peut être montré que la position $\boldsymbol{r}_i$ des CM_i peut être exprimée en terme d'une combinaison linéaire des vecteurs barycentriques ainsi définis [18] selon l'équation

$$\boldsymbol{r}_i = \sum_{k=1}^{n} \boldsymbol{b}_{ki} \tag{1.17}$$

avec $\boldsymbol{b}_{ki}$ défini en utilisant la règle suivante

$$\boldsymbol{b}_{ki} = \begin{cases} \boldsymbol{l}^*_{0k} & k > i \\ \boldsymbol{c}^*_{0k} & k = i \\ \boldsymbol{r}^*_{0k} & k < i \end{cases}. \tag{1.18}$$

Le vecteur $\boldsymbol{\omega}_i$ de chaque membrure exprimé dans R_0 est obtenu en additionnant les vitesses articulaires de chaque liaison exprimées dans la direction de la liaison avec la vitesse angulaire $\boldsymbol{\omega}_1$ de la première membrure selon la règle suivante

$$\boldsymbol{\omega}_i = \begin{cases} \boldsymbol{\omega}_1 & i = 1 \\ \boldsymbol{\omega}_{i-1} + \boldsymbol{e}_{i-1}\dot{\theta}_{i-1} & i = 2, \ldots, n. \end{cases} \tag{1.19}$$

De plus, en considérant que le vecteur $\boldsymbol{b}_{ki}$ de chaque membrure i est constant dans un repère qui tourne avec une vitesse $\boldsymbol{\omega}_i$, le vecteur $\boldsymbol{v}_i$ est obtenu par

$$\boldsymbol{v}_i = \sum_{k=1}^{n} \boldsymbol{\omega}_k \times \boldsymbol{b}_{ki}. \tag{1.20}$$

En substituant l'équation (1.19) dans l'équation (1.20), on obtient

$$\boldsymbol{v}_i = \boldsymbol{\omega}_1 \times \boldsymbol{r}_i + \boldsymbol{C}_i\dot{\boldsymbol{\theta}} \tag{1.21}$$

où $\boldsymbol{C}_i$ est une matrice de dimension $3 \times j$ qui dépend des paramètres de construction du robot.

Finalement, on substitue les équations (1.17), (1.19) et (1.21) dans l'équation (1.4) afin de former un système d'équation linéaire dépendant uniquement des inconnues $\boldsymbol{\omega}_1$ et $\dot{\boldsymbol{\theta}}$. En plaçant tous les termes en $\boldsymbol{\omega}_1$ d'un côté, et tous les termes en $\dot{\boldsymbol{\theta}}$ d'un autre, on obtient une formulation du modèle dynamique du robot qui exprime le comportement de la première membrure du robot en fonction des vitesses articulaires des liaisons, qui s'exprime comme suit

$$\boldsymbol{A}\boldsymbol{\omega}_1 = \boldsymbol{B}\dot{\boldsymbol{\theta}} \tag{1.22}$$

où $\boldsymbol{A}$ est une matrice de dimension 3×3 et $\boldsymbol{B}$ est une matrice de dimension $3 \times j$. Ces matrices dépendent toutes deux de la masse et de l'inertie des membrures, des positions des centres de masse sur les membrures, des paramètres géométriques du robot ainsi que de sa configuration. Un exemple détaillé de la construction de ce modèle pour un système à trois membrures et deux liaisons est présenté à l'annexe A.

1.3 Validation du modèle dynamique

Afin de s'assurer de la validité du modèle dynamique, une comparaison de celui-ci a été effectuée entre le logiciel de modélisation Adams et l'implantation du modèle présenté ci-haut dans Matlab. Pour effectuer les simulations, un modèle de robot sériel plan à 3 membrures et 2 articulations est utilisé. La géométrie de ce robot est donnée au chapitre 3. La manière utilisée afin de vérifier le modèle a été de prescrire aux deux logiciels les mêmes vitesses angulaires $\dot{\boldsymbol{\theta}}$ aux articulations du robot et de comparer par la suite la vitesse angulaire $\boldsymbol{\omega}_1$ obtenue.

Dans Matlab, le vecteur de la vitesse angulaire $\boldsymbol{\omega}_1$ est obtenu aisément en inversant la matrice $\boldsymbol{A}$ dans l'équation (1.22), à condition que celle-ci soit de rang plein [1], ce qui donne

$$\boldsymbol{\omega}_1 = \boldsymbol{A}^{-1}\boldsymbol{B}\dot{\boldsymbol{\theta}} \tag{1.23}$$

Plusieurs différents types de courbes de vitesses articulaires ont été testés, et les figures 1.3 et 1.4 montrent un cas typique de la comparaison de l'évolution de la composante en Z de la vitesse angulaire $\boldsymbol{\omega}_1$ pour les deux logiciels. En effet, puisque les deux liaisons du robot pointent tous deux dans la direction Z et que les matrices d'inerties des membrures sont fortement diagonales, seule une vitesse angulaire dans cette direction est produite.

Dans cet exemple la vitesse angulaire $\dot{\boldsymbol{\theta}}$ prescrite aux articulations est donnée par

$$\dot{\boldsymbol{\theta}} = \begin{bmatrix} 5\sin(2t) \\ 15\sin(t) \end{bmatrix}. \tag{1.24}$$

Dans ce cas, les figures 1.3 et 1.4 illustrent bien que les résultats obtenus par les deux logiciels sont très semblables.

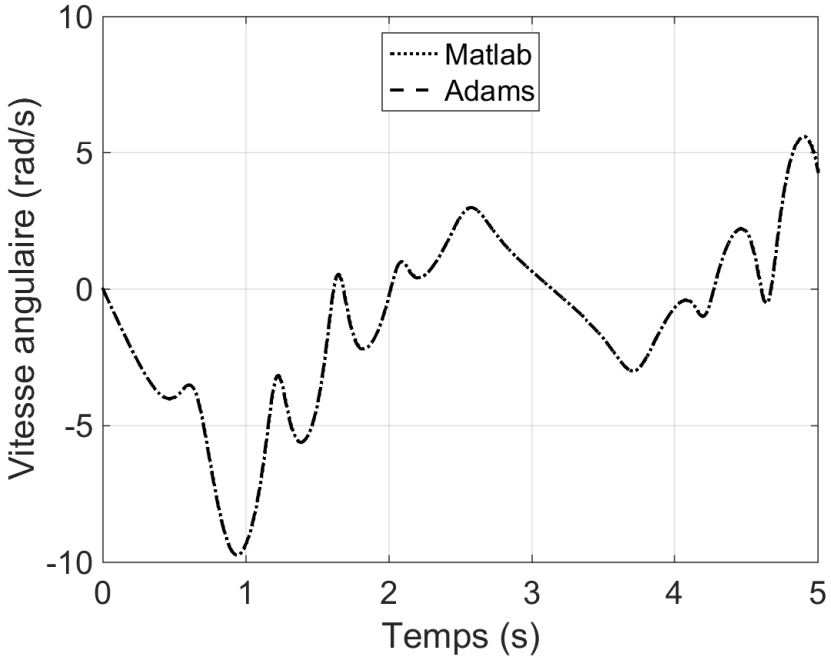


FIGURE 1.3 – Vitesse angulaire $\boldsymbol{\omega}_1$ de la première membrure selon l'axe Z obtenue avec le logiciel Adams et avec le modèle présenté dans cette section, programmé dans Matlab.

1. Par observation de l'équation (A.17), on peut voir que pour un corps physique réel, la matrice $\boldsymbol{A}$ est de rang plein.

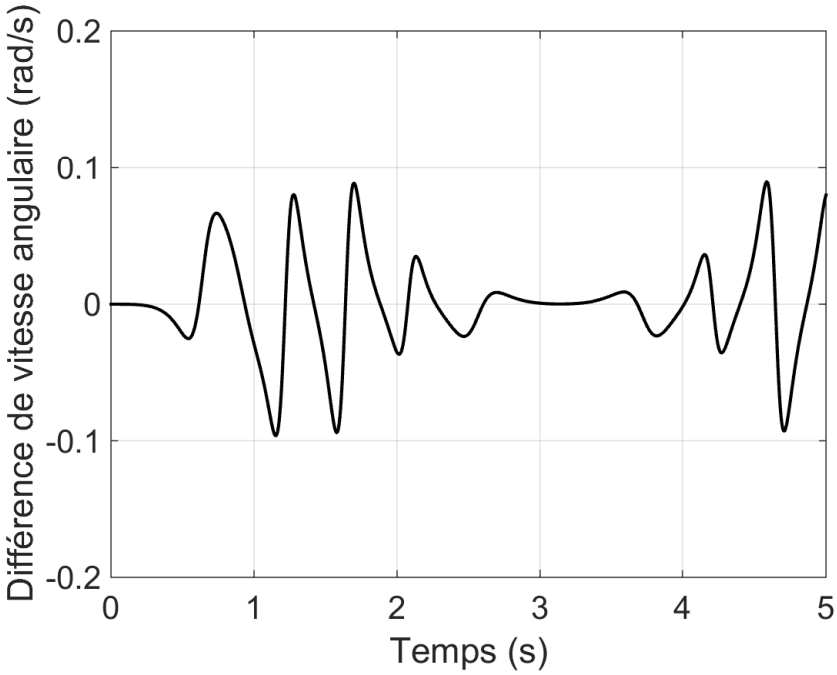


FIGURE 1.4 – Différence de vitesse angulaire $\boldsymbol{\omega}_1$ selon l'axe Z entre les deux modélisations.

1.4 Reformulation du modèle

Comme mentionné précédemment, l'équation (1.22) permet de décrire comment évolue l'orientation d'un robot en chute libre pendant une réorientation. Cette section se concentre à reformuler cette équation afin de la rendre apte à être utilisée par l'algorithme de réorientation présenté au chapitre 2.

Cette nouvelle formulation vise à regrouper le vecteur de vitesse angulaire de la première membrure $\boldsymbol{\omega}_1$ avec les vitesses articulaires des liaisons $\dot{\boldsymbol{\theta}}$ de manière à créer un seul vecteur de contrôle. Ce vecteur permettra par la suite de déterminer la vitesse angulaire de la première membrure ainsi que celle de toutes les autres grâce à l'équation (1.19), et ainsi connaître l'évolution des mouvements du robot dans le temps.

Pour y arriver, le vecteur $\boldsymbol{\omega}_1$ est d'abord exprimé en terme des dérivées temporelles des angles d'Euler, ce qui s'écrit comme suit

$$\boldsymbol{\omega}_1 = \boldsymbol{S}\dot{\boldsymbol{\beta}} \tag{1.25}$$

où le vecteur $\dot{\boldsymbol{\beta}}$ représente la dérivée par rapport au temps des angles d'Euler et où $\boldsymbol{S}$ est une matrice qui permet de faire la transformation entre les deux vecteurs. En utilisant la

convention XYZ des angles d'Euler, la matrice $\boldsymbol{S}$ s'écrit comme suit

$$\boldsymbol{S} = \begin{bmatrix} \boldsymbol{e}_x, & \boldsymbol{M}\boldsymbol{e}_y, & \boldsymbol{M}\boldsymbol{N}\boldsymbol{e}_z \end{bmatrix} = \begin{bmatrix} 1 & 0 & \sin\rho \\ 0 & \cos\phi & -\sin\phi\cos\rho \\ 0 & \sin\phi & \cos\phi\cos\rho \end{bmatrix} \tag{1.26}$$

En substituant l'équation (1.25) dans l'équation (1.22) on obtient le résultat suivant

$$\boldsymbol{A}\boldsymbol{S}\dot{\boldsymbol{\beta}} = \boldsymbol{B}\dot{\boldsymbol{\theta}}. \tag{1.27}$$

À noter que cette dernière équation peut facilement être modifiée afin de connaître l'évolution de l'orientation de la première membrure dans le temps lorsque des vitesses articulaires sont prescrites, ce qui s'écrit comme suit

$$\dot{\boldsymbol{\beta}} = (\boldsymbol{A}\boldsymbol{S})^{-1}\boldsymbol{B}\dot{\boldsymbol{\theta}}. \tag{1.28}$$

La nouvelle formulation désirée du modèle dynamique peut être produite en regroupant les différents vecteurs et matrices de l'équation 1.27 de la manière suivante

$$\boldsymbol{K}\dot{\boldsymbol{\lambda}} = \boldsymbol{0} \tag{1.29}$$

où $\boldsymbol{K}$ est une matrice de dimension $3 \times (3 + j)$ définie par

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{A}\boldsymbol{S} & -\boldsymbol{B} \end{bmatrix} \tag{1.30}$$

et où $\dot{\boldsymbol{\lambda}}$ est un vecteur de dimension $(3 + j)$ qui est défini par

$$\dot{\boldsymbol{\lambda}} = \begin{bmatrix} \dot{\boldsymbol{\beta}} \\ \dot{\boldsymbol{\theta}} \end{bmatrix}. \tag{1.31}$$

La matrice $\boldsymbol{K}$ contient toute la dynamique d'orientation du robot, et le vecteur $\dot{\boldsymbol{\lambda}}$, quant à lui, contient les variables inconnues qui décrivent la progression de l'orientation de chacune des membrures du robot.

1.5 Conclusion

Ce chapitre a permis d'énoncer une méthode systématique de construire le modèle dynamique de l'orientation d'un robot sériel en chute libre à l'aide de la loi de la conservation du moment angulaire. En prescrivant des vitesses articulaires données aux liaisons du robot, la formulation utilisée permet de connaitre l'évolution de l'orientation de chacune des membrures du mécanisme dans le temps. Une validation de ce modèle a été effectuée en comparant les résultats obtenus avec le modèle développé à ceux obtenus avec le logiciel commercial Adams. Ceci a permis d'affirmer que le modèle décrivait de manière appropriée l'évolution de l'orientation du type de robot étudié. Une modification de la formulation de ce modèle a ensuite été effectuée, et les raisons de celle-ci seront expliquées au prochain chapitre.

Chapitre 2

Développement de l'algorithme de réorientation

Ce chapitre présente différentes méthodes pour la réorientation d'un robot sériel en chute libre. Les méthodes développées sont regroupées en deux catégories distinctes. La première de celles-ci détermine des trajectoires articulaires à travers une fonction potentielle exprimant le degré de réorientation du robot, tandis que la deuxième se propose d'appliquer des fonctions de vitesse sinusoïdales aux articulations du robot.

Dans tous les cas, les méthodes sont développées pour permettre la réorientation d'un robot à partir d'une orientation initiale vers une orientation prescrite pour lesquelles la configuration du robot est la même, c'est-à-dire que les angles entre les liaisons doivent être les mêmes au début et à la fin de la réorientation.

2.1 Algorithmes de réorientation par minimisation d'une fonction potentielle

Cette section présente deux différentes stratégies de réorientation basées sur la minimisation d'une fonction potentielle exprimant le degré de réorientation du robot.

La première méthode tente de déterminer des trajectoires articulaires des articulations du robot avant que la chute se produise, pour ensuite être en mesure de les appliquer directement lorsque la chute se produit. Ensuite, la deuxième méthode constitue en quelque sorte une évolution de la première, dans le sens où celle-ci tente de réorienter le robot en ayant une connaissance préalable des types de trajectoires articulaires permettant une réorientation efficace, tout en effectuant des modifications en temps réel sur les trajectoires afin de pallier aux erreurs de contrôle et de s'adapter aux imprévus.

Bien que les deux méthodes proposées diffèrent dans la manière de réorienter le robot, celles-ci

reposent toutes sur le même algorithme principal. Cet algorithme est d'abord présenté, puis les variantes associées aux différentes stratégies sont exposées suivi des résultats obtenus en simulation pour chacune d'elles.

2.1.1 Description de l'algorithme principal

2.1.1.1 Construction de l'algorithme

L'algorithme principal de réorientation utilisé par toutes les méthodes présentées dans cette section est basé sur l'algorithme de planification de trajectoire pour des robots sériels redondants développé par Liégeois [19]. Sommairement, cet algorithme consiste à faire effectuer une tâche principale au robot, comme déplacer l'effecteur du robot d'un point à un autre, tout en minimisant une fonction potentielle représentant une tâche secondaire, par exemple minimiser l'énergie cinétique du robot. Cette méthode consiste en une technique d'optimisation locale fonctionnant par descente de gradient. De plus, cette méthode possède l'avantage d'être déterministe, c'est-à-dire qu'elle donne toujours la même solution pour la même combinaison de pose du robot et de valeur de la fonction potentielle. Cela signifie donc que le résultat final de l'algorithme dépend uniquement des valeurs initiales des paramètres du problème. L'algorithme, qui doit être calculé pour chaque pas de temps de la réorientation, est exprimé comme suit

$$\dot{\boldsymbol{\lambda}} = \boldsymbol{K}^I \boldsymbol{t} - \alpha(\mathbf{1} - \boldsymbol{K}^I \boldsymbol{K})(\frac{\partial \eta}{\partial \boldsymbol{\lambda}}) \tag{2.1}$$

où $\boldsymbol{K}^I \boldsymbol{t}$ représente la tâche principale et où $-\alpha(\mathbf{1} - \boldsymbol{K}^I \boldsymbol{K})(\frac{\partial \eta}{\partial \boldsymbol{\lambda}})$ représente la tâche secondaire avec α un scalaire positif qui est déterminé expérimentalement, $\boldsymbol{K}^I$ la pseudoinverse de la matrice $\boldsymbol{K}$, $(\mathbf{1} - \boldsymbol{K}^I \boldsymbol{K})$ un opérateur qui projète dans le noyau de la matrice $\boldsymbol{K}$ [20] et $\frac{\partial \eta}{\partial \boldsymbol{\lambda}}$ la dérivée partielle d'une fonction potentielle η par rapport aux éléments du vecteur $\boldsymbol{\lambda}$. Le vecteur $\boldsymbol{t}$ de cette équation représente en temps normal la vitesse cartésienne de l'effecteur du robot, et peut être relié aux vitesses articulaires du robot par l'intermédiaire d'une matrice Jacobienne $\boldsymbol{J}$ [16] par l'équation suivante

$$\boldsymbol{J}\dot{\boldsymbol{\theta}} = \boldsymbol{t}. \tag{2.2}$$

Or, en substituant cette équation par l'équation du modèle dynamique développé à l'équation (1.29), on obtient que le vecteur $\boldsymbol{t}$ vaut $\mathbf{0}$. On peut ainsi simplifier l'expression de l'algorithme de Liégeois et obtenir l'expression finale de l'équation qui permet de déterminer quel vecteur $\dot{\boldsymbol{\lambda}}$ appliquer à chaque pas de temps de l'algorithme afin de satisfaire l'équation (1.29) tout en minimisant une certaine fonction potentielle

$$\dot{\boldsymbol{\lambda}} = -\alpha(\mathbf{1} - \boldsymbol{K}^I \boldsymbol{K})(\frac{\partial \eta}{\partial \boldsymbol{\lambda}}). \tag{2.3}$$

Qui plus est, la formulation du modèle dynamique exprimé à l'équation (1.29) révèle une propriété intéressante de celui-ci, soit que le vecteur $\dot{\boldsymbol{\lambda}}$ est contenu dans le noyau de la matrice $\boldsymbol{K}$. Respecter cette propriété assure d'ailleurs que le moment angulaire soit conservé tout au long de la réorientation.

Dans le cas présent, on désire que la fonction potentielle représente le degré de réorientation du robot, c'est-à-dire à quel point celui-ci est près d'atteindre l'orientation désirée, ce qui est exprimée par le scalaire η. Cette fonction peut aussi être vu comme la différence entre l'orientation actuelle de chacune des membrures du robot et leur orientation désirée. Pour exprimer cette fonction, deux vecteurs unitaires sont définis pour chacune des membrures du robot, soit $\boldsymbol{s}_{ir}$ et $\boldsymbol{s}_{i0}$. Ces vecteurs sont montrés à la figure 2.1. Le vecteur $\boldsymbol{s}_{ir}$ représente l'orientation actuelle de la membrure i, tandis que le vecteur $\boldsymbol{s}_{i0}$ représente l'orientation finale désirée de cette membrure. Le vecteur $\boldsymbol{s}_{i0}$ est exprimé dans le repère R_0, tandis que le vecteur $\boldsymbol{s}_{ir}$ est défini dans le repère attaché au corps i et peut donc être écrit comme une fonction des matrices $\boldsymbol{Q}_i$. Ainsi, η peut être formulé comme

$$\eta = \frac{1}{2}\sum_{i=1}^{n} \|(\prod_{1}^{i} \boldsymbol{Q}_i)\underline{\boldsymbol{s}}_{ir} - \boldsymbol{s}_{i0}\| \qquad\qquad 0 \leq \eta \leq n \tag{2.4}$$

où $\|\cdot\|$ représente la norme euclidienne. Une valeur de 0 pour cette fonction signifie que le robot a atteint l'orientation désirée. Il est aussi important de spécifier que dans ce cas on désire orienter des vecteurs (2 degrés de liberté) selon la verticale, et que la rotation de ces vecteurs autour de cet axe n'a pas d'importance.

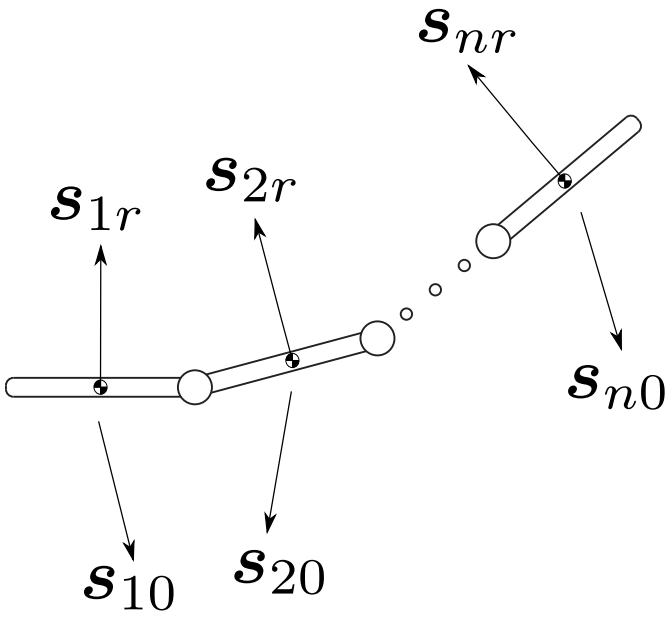


FIGURE 2.1 – Définition des vecteurs qui représentent l'orientation actuelle $\boldsymbol{s}_{ir}$ de chaque membrure i et leur orientation finale désirée $\boldsymbol{s}_{i0}$.

L'algorithme général de réorientation consiste ainsi à combiner les équations (2.3) et (2.4) pour chaque pas de temps de manière à déterminer la progression de l'orientation du robot dans le temps représentée par le vecteur $\boldsymbol{\lambda}$, ce qui se calcule par intégration

$$\boldsymbol{\lambda}_{t+1} = \dot{\boldsymbol{\lambda}}\Delta t + \boldsymbol{\lambda}_t \tag{2.5}$$

où Δt représente la longueur du pas de temps utilisé et les indices t et $t+1$ réfèrent au pas de temps de l'algorithme. On répète ainsi ces étapes jusqu'à ce qu'un minimum de la fonction potentielle soit atteint.

Cependant, procéder de cette manière ne garantit pas de parvenir à réorienter le robot dans la configuration désirée. En effet, puisque la méthode de minimisation utilisée fonctionne par descente de gradient, il est possible que l'optimisation aboutisse dans un minimum local plus ou moins rapproché du minimum global du problème. De plus, le fait que la méthode soit de type déterministe fait en sorte que l'algorithme donne toujours la même solution pour une même configuration initiale du robot. En d'autres termes, c'est donc dire qu'il existe potentiellement un nombre limité de configurations initiales du robot qui permettent de se rendre à l'orientation finale désirée, et que ces configurations ne correspondent pas nécessairement à la configuration initiale du problème. La manière de résoudre cette problématique est décrite pour chacun des algorithmes aux sections 2.1.2 et 2.1.3.

2.1.1.2 Limitation des vitesses articulaires

Comme mentionné précédemment, le vecteur $\dot{\boldsymbol{\lambda}}$ déterminé par l'équation (2.3) contient le vecteur de vitesses articulaires $\dot{\boldsymbol{\theta}}$ qui doit être appliqué au pas de temps actuel. Puisque ce vecteur est proportionnel au paramètre α qui est ajustable au besoin, on peut aussi le voir comme un ratio à respecter entre les différentes vitesses articulaires $\dot{\theta}_i$. Cependant, même si les valeurs contenues dans ce vecteur permettent de respecter la conservation du moment angulaire et de minimiser la fonction potentielle demandée, celles-ci ne respectent pas nécessairement les contraintes dynamiques au niveau des moteurs du robot. En effet, il est possible que les valeurs de vitesses articulaires calculées pour un pas de temps de la réorientation soient trop grandes, ou même impraticables comparé à la capacité des moteurs. Bien sûr, le vecteur $\dot{\boldsymbol{\lambda}}$ peut être remis à l'échelle en diminuant le paramètre α de manière à satisfaire les limites de vitesses articulaires minimales $\dot{\boldsymbol{\theta}}_{min}$ et maximales $\dot{\boldsymbol{\theta}}_{max}$. Cependant, dans certaines conditions, il n'est pas possible de procéder de cette manière pour satisfaire les contraintes d'accélération minimales $\ddot{\boldsymbol{\theta}}_{min}$ et maximales $\ddot{\boldsymbol{\theta}}_{max}$ des moteurs, en majeure partie puisque l'accélération à un pas de temps donné est fonction de la vitesse au pas de temps précédent, ce qui s'exprime comme suit

$$\ddot{\boldsymbol{\theta}}_t \simeq \frac{\dot{\boldsymbol{\theta}}_t - \dot{\boldsymbol{\theta}}_{t-1}}{\Delta t}. \tag{2.6}$$

Où Δt est le pas de temps entre deux points successifs de la trajectoire. On peut remanier la précédente équation afin de déterminer quelles sont les vitesses minimales ou maximales possibles des articulations selon les limites d'accélération des moteurs, ce qui donne

$$\dot{\boldsymbol{\theta}}_{t,min} = \ddot{\boldsymbol{\theta}}_{t,min}\Delta t + \dot{\boldsymbol{\theta}}_{t-1} \tag{2.7}$$

et

$$\dot{\boldsymbol{\theta}}_{t,max} = \ddot{\boldsymbol{\theta}}_{t,max}\Delta t + \dot{\boldsymbol{\theta}}_{t-1}. \tag{2.8}$$

Afin de déterminer si le vecteur $\dot{\boldsymbol{\lambda}}$ peut être remis à l'échelle ou non, on commence donc par calculer toutes les contraintes de vitesse et d'accélération, puis on vérifie si le vecteur $\dot{\boldsymbol{\theta}}$ peut être mis à l'échelle afin d'être contenu dans la boîte de contrainte ainsi formée. Une représentation graphique de la boîte de contrainte pour le cas d'un robot possédant deux articulations est montrée à la figure 2.2.

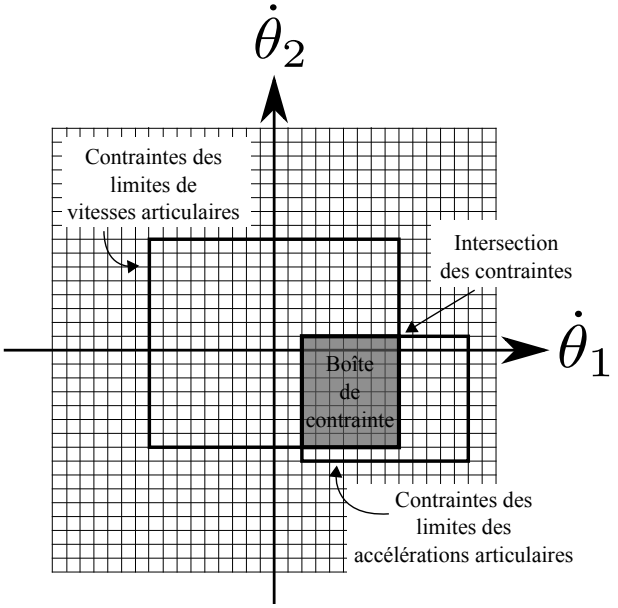


FIGURE 2.2 – Représentation de la boîte de contrainte des vitesses articulaires pouvant être appliquées aux articulations pour un pas de l'optimisation donné.

Si la droite supportant le vecteur $\dot{\boldsymbol{\theta}}$ calculé intersecte la boîte de contrainte, on ajuste le paramètre α de façon à obtenir le vecteur $\dot{\boldsymbol{\theta}}$ de module maximum qui intersecte la boîte de contrainte, puis on applique ce vecteur pour le pas de temps donné, comme montré à la figure 2.3. Le vecteur $\dot{\boldsymbol{\beta}}$ résultant pour ce pas de temps est obtenu en diminuant le vecteur $\dot{\boldsymbol{\beta}}$ initial d'un facteur égal à celui calculé pour le vecteur $\dot{\boldsymbol{\theta}}$.

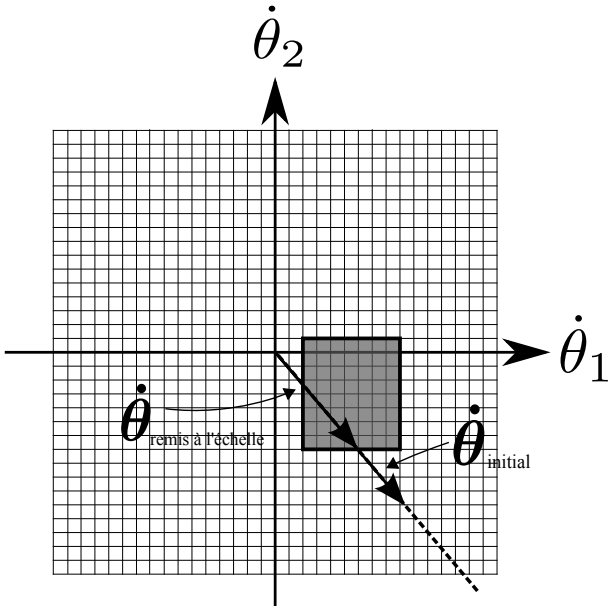


FIGURE 2.3 – Représentation de la remise à niveau du vecteur $\dot{\boldsymbol{\theta}}$ lorsque sa norme est trop grande et qu'il intersecte la boîte de contrainte.

Cependant, il se peut aussi qu'un tel réajustement ne soit pas possible dans le cas où la droite supportant le vecteur $\dot{\boldsymbol{\theta}}$ n'intersecte pas la boîte de contrainte. De plus, le fait que le paramètre α soit défini positif implique que le signe des éléments du vecteur $\dot{\boldsymbol{\lambda}}$ ne peut être modifié lors d'une remise à l'échelle, c'est-à-dire qu'une vitesse initialement calculée positive pour un pas de temps donné ne peut être diminuée à une valeur plus basse que 0, et vice versa. Sinon, la fonction objectif serait maximisée au lieu d'être minimisée. Ainsi, selon le signe des vitesses articulaires des vecteurs $\dot{\boldsymbol{\theta}}_{t-1}$ et $\dot{\boldsymbol{\theta}}_t$ et des valeurs limites possibles d'accélération, il se peut qu'il ne soit pas possible de respecter les contraintes d'accélération. La figure 2.4 illustre ce phénomène.

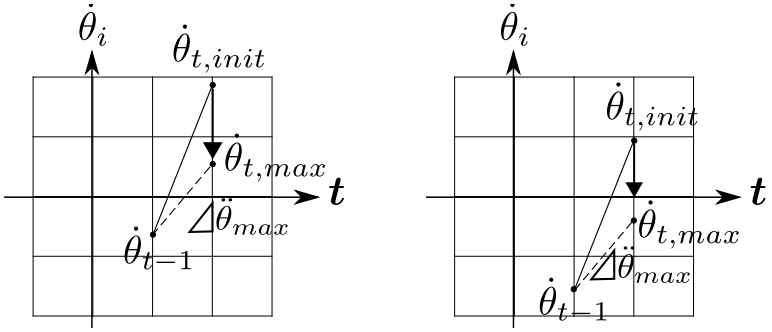


FIGURE 2.4 – Représentation de la possibilité de satisfaire (à gauche) ou non (à droite) la contrainte d'accélération quand la vitesse est près de 0.

Lorsque le vecteur $\dot{\boldsymbol{\theta}}$ ne permet pas de respecter les contraintes de vitesse et d'accélération

pour le pas de temps donné, on utilise plutôt la projection de celui-ci sur les limites minimales ou maximales de la boîte de contrainte pour déterminer le vecteur $\dot{\boldsymbol{\theta}}$ à appliquer. Procéder de cette manière permet de respecter les contraintes dynamiques du moteur et les différents essais effectués ont montré que cela favorisait aussi au mieux la réorientation. Le vecteur $\dot{\boldsymbol{\beta}}$ correspondant au nouveau vecteur $\dot{\boldsymbol{\theta}}$ pour ce pas de temps donné est déterminé en utilisant l'équation (1.28). La règle qui permet de déterminer le nouveau vecteur $\dot{\boldsymbol{\theta}}$ est la suivante

$$\dot{\theta}_k = \begin{cases} \dot{\theta}_{k,min} & \dot{\theta}_k < \dot{\theta}_{k,min} \\ \dot{\theta}_k & \dot{\theta}_{k,min} < \dot{\theta}_k < \dot{\theta}_{k,max} \\ \dot{\theta}_{k,max} & \dot{\theta}_k > \dot{\theta}_{k,max} \end{cases} \qquad k = 1, \ldots, j. \tag{2.9}$$

Bien que des contraintes en vitesse et en accélération sont suffisantes pour représenter dynamiquement les caractéristiques d'un moteur, il est à noter que des contraintes reliées à la secousse (*jerk* en anglais) pourraient être introduites afin de rendre les trajectoires articulaires plus régulières. Cependant, ajouter de telles contraintes implique aussi la possibilité de réduire la taille des boîtes de contraintes, et ainsi empêcher la possibilité que le vecteur $\dot{\boldsymbol{\lambda}}$ soit contenu dans celle-ci, ce qui n'est pas nécessairement souhaitable.

2.1.2 Algorithme en mode hors-ligne

2.1.2.1 Construction de l'algorithme

La première méthode de réorientation présentée est de type «hors-ligne», puisqu'elle consiste à calculer la trajectoire de réorientation avant que la chute libre se produise, pour ensuite pouvoir l'appliquer et être en mesure de se retourner lorsque la chute se produit.

Pour y arriver, l'algorithme de réorientation de base présenté à la section 2.1.1 est utilisé. Comme mentionné à cette section, utiliser directement cet algorithme peut faire en sorte que la procédure converge vers un minimum local, ce qui ne permettrait plus de continuer la réorientation. Pour contourner le problème, la méthode présentée ici propose de modifier légèrement la manière de formuler la fonction potentielle de l'équation (2.4) de manière à ce que sa forme varie constamment pour chaque pas de temps de la simulation et éloigne ainsi la position de ses minimum locaux. On souhaite de cette manière faire en sorte que le minimum local rencontré à la fin de la trajectoire soit le plus près possible du minimum global du problème. Pour y arriver, on définit un troisième vecteur unitaire $\boldsymbol{s}_{it}$ pour chaque membrure i que l'on substitue au vecteur $\boldsymbol{s}_{i0}$ dans l'équation (2.4) pour former l'équation suivante

$$\tilde{\eta} = \frac{1}{2} \sum_{i=1}^{n} \| (\prod_{1}^{i} \boldsymbol{Q}_i) \underline{\boldsymbol{s}}_{ir} - \boldsymbol{s}_{it} \| \qquad 0 \leq \tilde{\eta} \leq n. \tag{2.10}$$

où $\tilde{\eta}$ représente une fonction potentielle modifiée qui varie dynamiquement tout au long de la réorientation. La direction dans l'espace du vecteur $\boldsymbol{s}_{it}$ par rapport à celle du vecteur $\boldsymbol{s}_{i0}$

peut être déterminée avec les deux angles γ_1 et γ_2 montrés à la figure 2.5

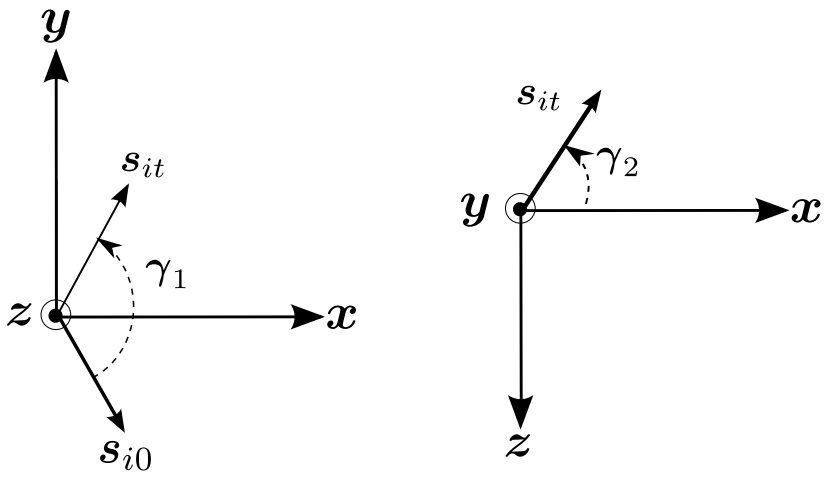


FIGURE 2.5 – Défintion des angles γ_1 et γ_2.

et s'exprime de la manière suivante

$$\boldsymbol{s}_{it} = \boldsymbol{Q}_y(\gamma_2)\boldsymbol{Q}_z(\gamma_1)\boldsymbol{s}_{i0} \tag{2.11}$$

où $\boldsymbol{Q}_y$ et $\boldsymbol{Q}_z$ représentent respectivement des matrices de rotation autour des axes y et z d'angles γ_2 et γ_1 avec

$$\begin{aligned} -\pi \leq \gamma_1 \leq \pi \\ 0 \leq \gamma_2 \leq \pi \end{aligned} \tag{2.12}$$

Ainsi, la direction initiale du vecteur $\boldsymbol{s}_{it}$ est déterminée en définissant des valeurs initiales des angles γ_1 et γ_2, identifiées γ_{10} et γ_{20}. Par ailleurs, de manière à diminuer le nombre d'inconnues à déterminer par l'algorithme qui sera présenté, il a été décidé que les angles γ_1 et γ_2 seraient les mêmes pour toutes les membrures tout au long de la réorientation. Qui plus est, afin que la valeur de ces angles ait le même effet sur chacune des membrures i dans le calcul de la fonction potentielle peu importe leur orientation initiale, on pose que tous les vecteurs $\boldsymbol{s}_{i0}$ pointent dans la même direction.

Puis, une fois l'algorithme lancé, on fait varier la valeur du vecteur $\boldsymbol{s}_{it}$ à chaque pas de temps en faisant varier la valeur de l'angle γ_1 de manière à la faire tendre vers 0, ce qui fait en sorte que $\boldsymbol{s}_{it} = \boldsymbol{s}_{i0}$. Pour ce qui est de l'angle γ_2, il a été décidé arbitrairement de le laisser constant tout au long de la réorientation. Pour arriver à faire varier le vecteur $\boldsymbol{s}_{it}$, on utilise une heuristique relative à la diminution de η entre deux pas de temps successifs de la minimisation, et celle-ci se définit selon la règle suivante

$$\gamma_1(t+1) = \begin{cases} \gamma_1(t) + \xi(\eta(t) - \eta(t+1)) & \gamma_{10} < 0 \\ \gamma_1(t) - \xi(\eta(t) - \eta(t+1)) & \gamma_{10} > 0 \end{cases} \tag{2.13}$$

avec ξ un paramètre déterminé expérimentalement. De plus, on ajoute la condition que l'angle γ_1 et ne peut prendre une valeur plus grande ou plus petite que 0 tout dépendant de si la valeur initiale de γ_{10} est respectivement négative ou positive.

Ainsi, poser toutes les équations ci-dessus permet de créer un algorithme de minimisation déterministe où seulement les valeurs initiales de 4 paramètres (γ_1, γ_2, ξ et α) ont un impact sur le degré de réorientation final pouvant être atteint par le robot. L'algorithme est présenté à la figure 2.6.

(1) Poser des valeurs pour γ_{10}, γ_{20}, ξ et α et déterminer la valeur initiale du vecteur $\boldsymbol{\lambda}$.
While $\eta_{(t+1)} < \eta_{(t)}$ (η est calculé avec la fonction de l'équation (2.4)
- (2) Calculer la matrice $\boldsymbol{K}$.
- (3) Calculer le vecteur $\dot{\boldsymbol{\lambda}}$ avec l'équation (2.3) et le $\tilde{\eta}$ de l'équation (2.10).
- (4) Procéder à la mise au niveau du vecteur $\dot{\boldsymbol{\lambda}}$ au besoin.
- (5) Déterminer la valeur de $\boldsymbol{\lambda}_{(t+1)}$: $\boldsymbol{\lambda}_{(t+1)}$=$\dot{\boldsymbol{\lambda}}$ Δt +$\boldsymbol{\lambda}_t$.
- (6) Faire progresser la valeur des angles γ_1 et γ_2 avec (2.13)
- (7) $t = t + 1$

End While
(8) Procéder à la réorientation avec le vecteur $\dot{\boldsymbol{\theta}}$ déterminé

FIGURE 2.6 – Structure de l'algorithme hors-ligne.

Cependant, il n'est pas aisé de déterminer les 4 valeurs adéquates des paramètres mentionnés plus haut, car le problème d'optimisation est non-linéaire et les combinaisons possibles de valeurs pour ces paramètres sont très nombreuses.

Pour optimiser les paramètres, on utilise la fonction $fmincon$ du logiciel Matlab, exécutée avec l'option $active-set$. Cette fonction utilise une méthode d'optimisation locale, fonctionnant par descente de gradient, qui tente de minimiser une fonction potentielle donnée en faisant varier des variables qu'on lui donne en entrée. En d'autres mots, on fournit des estimés pour les valeurs initiales des 4 paramètres mentionnés plus haut, et $fmincon$ tente de faire varier la valeur de ceux-ci de manière à minimiser la valeur finale de η obtenue avec l'algorithme décrit précédemment.

Cependant, il est important de mentionner qu'exécuter la fonction $fmincon$ ne garantit pas de déterminer du premier coup les valeurs des 4 paramètres qui permettront d'obtenir une réorientation adéquate. En effet, la fonction utilise une méthode d'optimisation locale ce qui la soumet à la possibilité d'atteindre des minimums locaux tout dépendant des valeurs des estimés initiaux qui lui sont donnés en entrée. Par conséquent, afin d'améliorer les performances de la méthode, on exécute la fonction $fmincon$ autant de fois que nécessaire avec des estimés différents jusqu'à ce qu'une valeur satisfaisante de η_{seuil} soit atteinte. La procédure pour déterminer les valeurs des 4 paramètres γ_{10}, γ_{20}, ξ et α pour la méthode hors-ligne est

présentée à la figure 2.7.

(1) Déterminer des estimés initiaux aléatoires des paramètres γ_{10}, γ_{20}, ξ et α.
(2) Démarrer la fonction *fmincon* avec les valeurs déterminées à (1).
    (3) Exécuter l'algorithme présenté à la figure 2.6.
    (4) Modifier la valeur de γ_{10}, γ_{20}, ξ et α et jusqu'à ce que η_{final} soit minimisé.
(5) si η_{final}>seuil, retourner à (1).

FIGURE 2.7 – Procédure de la fonction *fmincon* avec l'algorithme hors-ligne.

Les sections 2.1.2.2 et 2.1.2.3 montrent les résultats obtenus en simulation en utilisant la présente méthode pour la réorientation d'un modèle de robot sériel plan à 3 membrures et 2 liaisons rotoïdes possédant ou non des limites de débattement angulaire au niveau des articulations. La géométrie et le design de ce robot sont présentés au chapitre 3. La section 2.1.2.4 discute ensuite de la pertinence de la méthode à la lumière des résultats obtenus.

2.1.2.2 Simulation de réorientation pour un robot plan à 3 membrures et 2 liaisons rotoïdes sans limites articulaires

Cette section présente les résultats de simulations avec l'algorithme présenté à la figure 2.7 pour la réorientation d'un robot plan à 3 membrures et 2 liaisons rotoïdes ne possédant pas de limites de débattement angulaire au niveau des articulations.

L'objectif de la réorientation est de débuter avec toutes les membrures alignées et de terminer dans la même configuration mais en ayant effectué une rotation de 180 degrés du robot autour de l'axe Z. En d'autres mots, on désire que $\boldsymbol{\theta}_0 = \boldsymbol{\theta}_f = [0\ 0]^T$ avec $\boldsymbol{\beta}_0 = [0\ 0\ 0]^T$ et $\boldsymbol{\beta}_f = [0\ 0\ \pi]^T$. Une représentation de l'orientation du robot avant et après la réorientation est montrée à la figure 2.8.

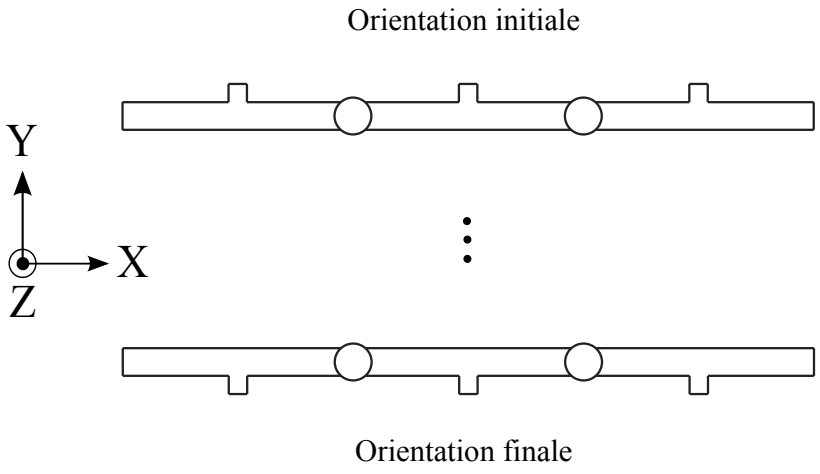


FIGURE 2.8 – Orientation initiale et finale désirée pour la réorientation du robot.

Ainsi, comme la réorientation a lieu dans le plan XY, c'est-à-dire selon l'axe Z, on pose que la direction initiale des vecteurs d'orientation $\underline{\boldsymbol{s}}_{ir}$ et $\boldsymbol{s}_{i0}$ vaut respectivement $[0\ 1\ 0]^T$ et $[0\ -1\ 0]^T$ pour $i = 1, \ldots, n$. Insérer ces valeurs dans l'équation (2.4) avec $n = 3$ donne une valeur initiale de η égale à 3. De plus, puisque la réorientation se produit autour de l'axe Z, on peut poser que l'angle γ_2 vaut 0 pour toute la durée de la réorientation.

La valeur du pas de temps de la simulation est posée arbitrairement à 0,005 seconde. Il est important de garder cette valeur assez petite, car une valeur trop grande aurait comme conséquence potentielle de créer des vecteurs $\dot{\boldsymbol{\lambda}}$ très éloignés les uns des autres entre deux pas consécutifs ce qui pourrait avoir comme effet de faire sortir les vecteur hors des limites de la boite de contrainte définie à la section 2.1.1.2.

Démarrer la fonction $fmincon$ nécessite de fournir des estimés initiaux pour les 3 paramètres γ_1, ξ et α. On définit ces estimés comme des valeurs aléatoires comprises entre des limites possibles que l'on pose pour chacun des paramètres. Ces limites sont les suivantes

$$\begin{aligned} -\pi \leq \gamma_1 \leq \pi \\ 0 \leq \xi \leq 0{,}1 \\ 0 \leq \alpha \leq 30. \end{aligned} \tag{2.14}$$

De plus, la fonction $fmincon$ est exécutée avec l'option $active-set$, car c'est celle qui a permis d'obtenir les meilleurs résultats. On utilise aussi l'option $nlcon$, car celle-ci permet d'obtenir des trajectoires de vitesses articulaires qui terminent avec une valeur de 0.

Après avoir exécuté l'algorithme, la meilleure combinaison des valeurs des paramètres ξ, γ_1 et α permet d'obtenir une valeur finale de η égale à 0,0649. Cela signifie qu'appliquer les trajectoires articulaires déterminées par l'algorithme permet de réorienter le robot d'environ 98%. Une visualisation de la progression de la réorientation du robot est présentée à la figure 2.9. Les valeurs numériques trouvées pour les 3 paramètres sont données au tableau 2.1

Tableau 2.1 – Valeur initiale des paramètres de simulation qui permettent de réorienter le robot dans la meilleure orientation finale.

γ_1	$-1{,}5925$
ξ	0,0084
α	6,6864

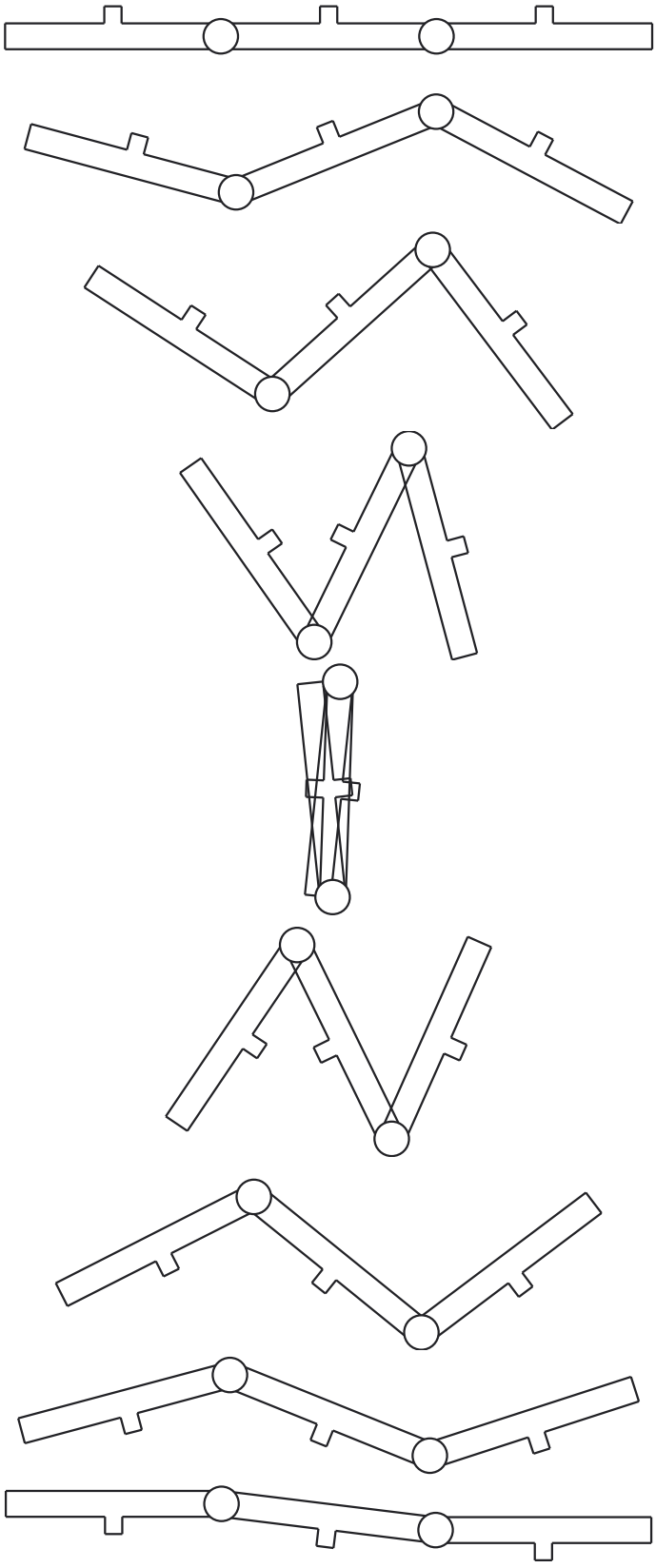


FIGURE 2.9 – Visualisation de la réorientation du robot sans contraintes aux articulations.

La progression du paramètre η le long de la trajectoire est présentée à la figure 2.10 et montre que la fonction ne rencontre pas de minimum locaux tout au long de la réorientation.

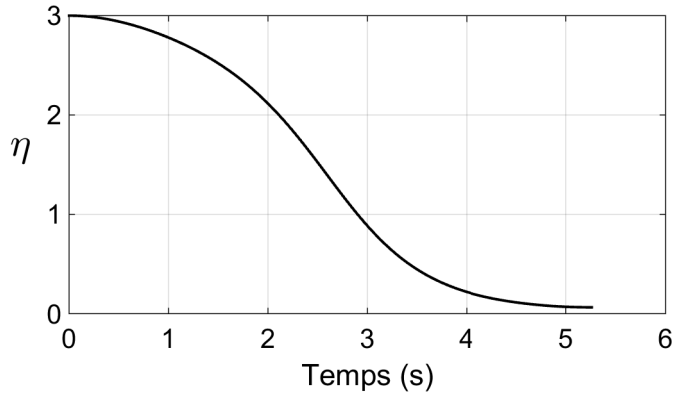


FIGURE 2.10 – Progression de la réorientation du robot exprimée par le scalaire η.

Les trajectoires de vitesses articulaires produites par l'algorithme sont obtenues directement du vecteur $\dot{\boldsymbol{\lambda}}$ calculé à chaque pas de temps de la simulation et sont présentées à la figure 2.11.

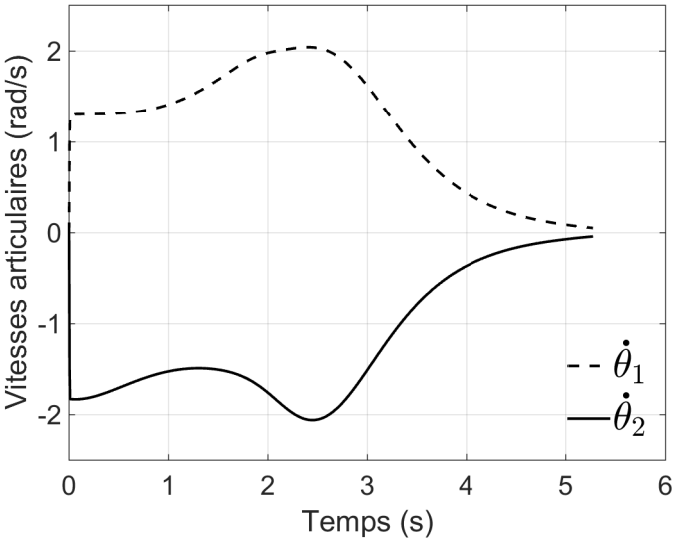


FIGURE 2.11 – Trajectoire des vitesses articulaires à appliquer aux moteurs, déterminées par l'algorithme.

Les figures 2.12 et 2.13 montrent respectivement la progression des angles d'Euler et des angles des articulations tout au long de la réorientation. La valeur du vecteur $\boldsymbol{\beta}$ à la fin de réorientation est égale à $[0{,}0034\ -0{,}0856\ -3{,}1417]^T$ et celle de $\boldsymbol{\theta}$ est de $[6{,}61685\ -6{,}61747]^T$.

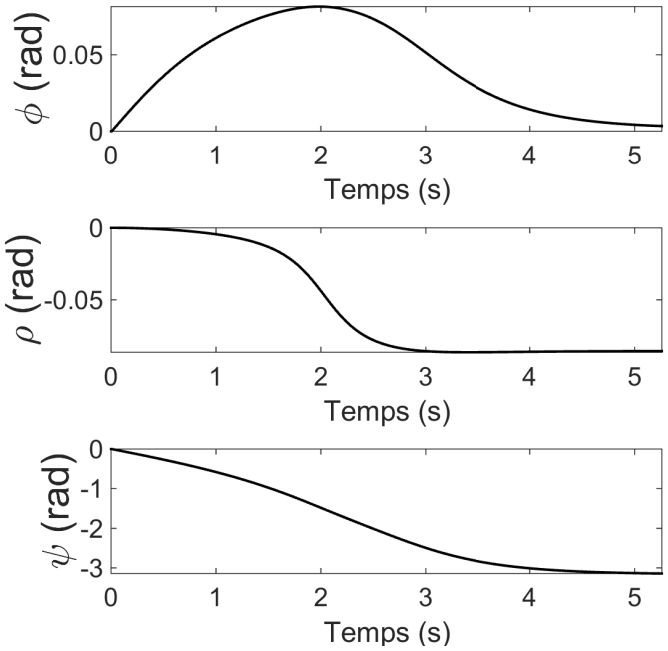


FIGURE 2.12 – Progression des angles d'Euler pendant la réorientation selon la convention XYZ.

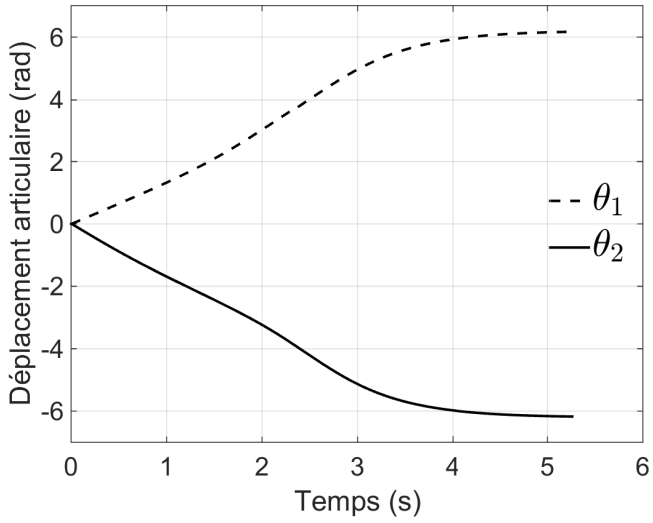


FIGURE 2.13 – Déplacements articulaires pendant la réorientation.

Suivant les résultats obtenus plus haut, d'autres simulations ont été faites afin de savoir ce qui se produirait si l'on appliquait directement au robot les courbes de vitesses présentées à la figure 2.11, mais en ajoutant un délai de plus en plus grand entre le début du mouvement de la deuxième articulation par rapport à la première. La figure 2.14 montre la valeur finale de ψ pouvant être obtenue selon le délai de temps entre l'application des vitesses.

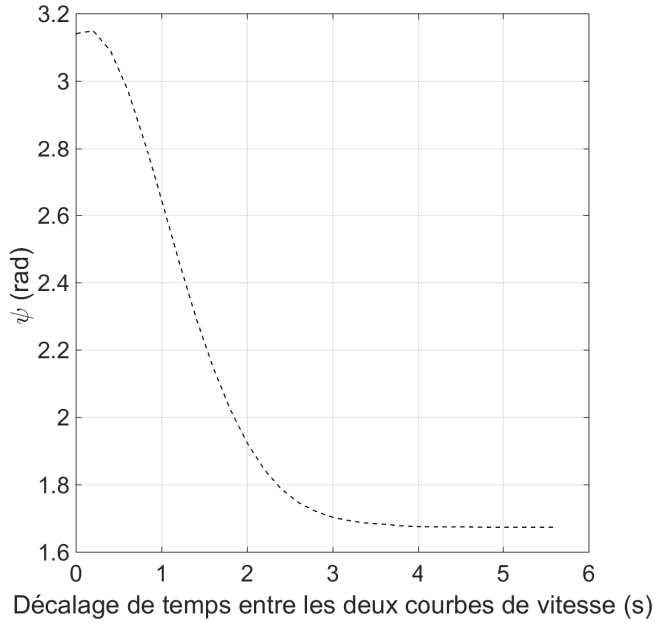


FIGURE 2.14 – Valeur finale de l'angle ψ obtenue selon le décalage de temps entre l'application des courbes de vitesses aux articulations.

Dans ce cas, la valeur de 3,1414 rad pour une décalage nul correspond aux résultats obtenus pour la réorientation avec la méthode hors-ligne présentée à la figure 2.9. La figure 2.14 montre que le résultat de la réorientation tend à se dégrader avec l'augmentation du décalage entre les courbes.

2.1.2.3 Simulation de réorientation pour un robot plan à 3 membrures et 2 liaisons rotoïdes avec limites articulaires

Cette section présente les résultats obtenus avec l'algorithme présenté à la figure 2.7 auquel on ajoute des contraintes de débattement angulaire au niveau des articulations. L'intérêt de tester une telle méthode est de voir s'il est possible de déterminer des trajectoires articulaires qui ne créent pas de collisions entre les membrures du robot.

Les conditions initiales et finales désirées pour le robot sont les mêmes qu'à la section précédente. La seule différence dans l'algorithme réside dans le fait que l'on abaisse la vitesse d'une articulation à 0 lorsque sa position atteint la limite articulaire. Dans ce cas-ci, les limites de débattement angulaire ont été fixées à $\pm\ \pi/2$ de leur valeur initiale.

Après exécution de l'algorithme, la meilleure combinaison des 3 paramètres ξ, γ_1 et α permet d'obtenir une valeur finale de η égale à 2,4082. Le tableau 2.2 montre les valeurs numériques des paramètres en question. Une visualisation de la progression de la réorientation du robot est présentée à la figure 2.15.

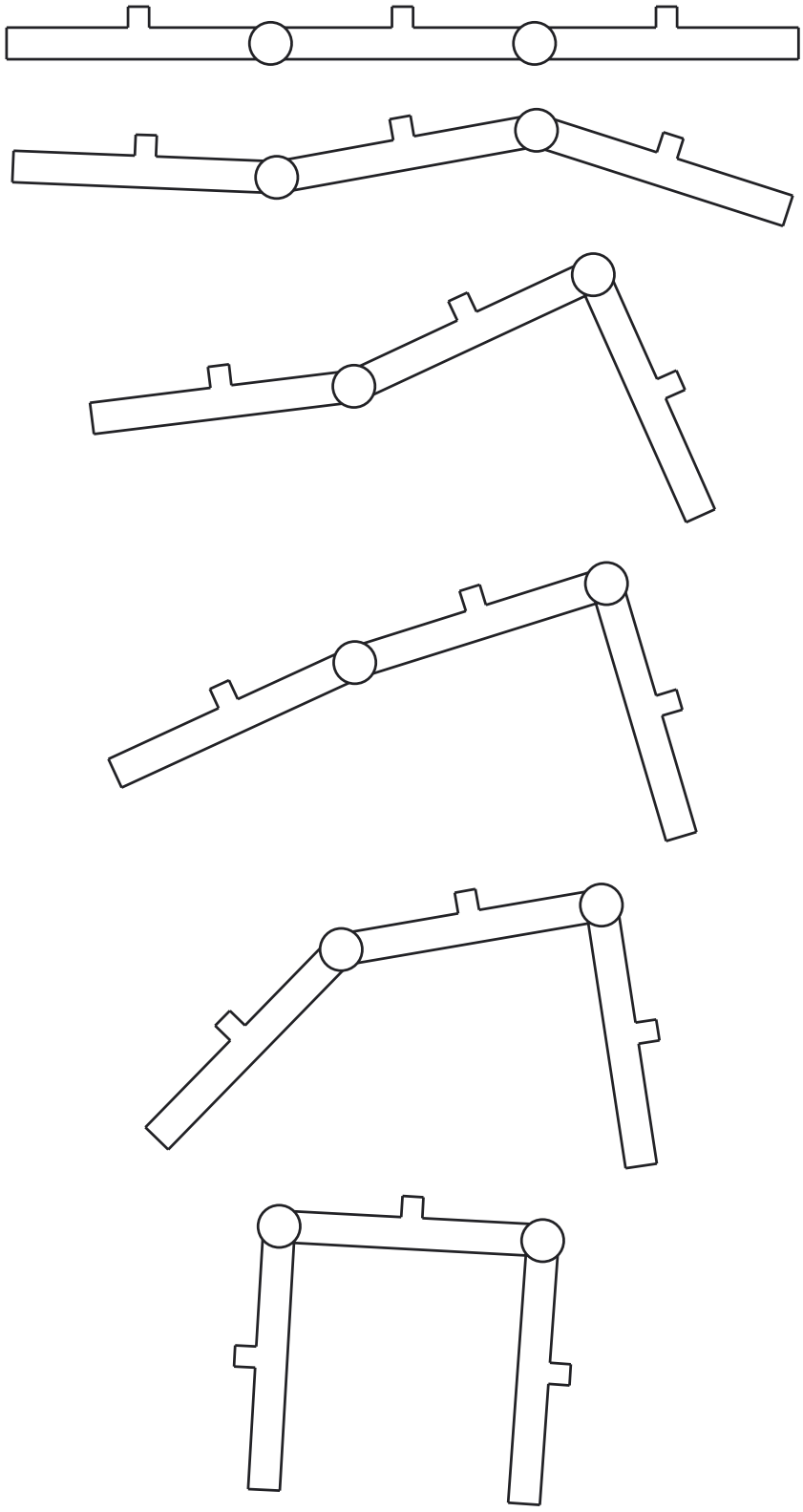


FIGURE 2.15 – Visualisation de la réorientation avec limites articulaires.

Tableau 2.2 – Valeur initiale des paramètres de simulation qui permettent de réorienter le robot dans la meilleure orientation finale pour une réorientation avec limite de débattement angulaire au niveau des articulations.

γ_1	$-2{,}8300$
ξ	0,0043
α	5,7753

La progression du paramètre η est présentée à la figure 2.16 et la progression de la position des articulations du robot est présentée à la figure 2.17. Les deux figures montrent que l'algorithme atteint un miminum local lorsque les deux articulations atteignent leur limite de débattement angulaire.

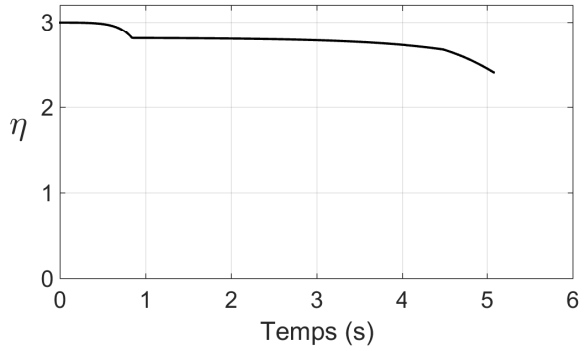


FIGURE 2.16 – Progression du paramètre η pendant la réorientation avec limites articulaires aux liaisons rotoïdes.

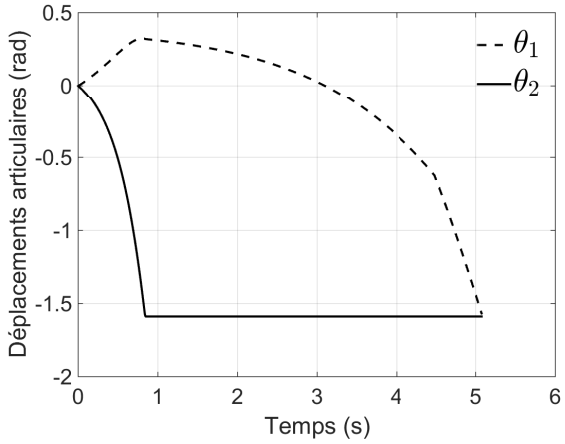


FIGURE 2.17 – Déplacements articulaires pendant la réorientation avec limites articulaires aux liaisons.

2.1.2.4 Discussion sur les résultats pour le robot à 3 membrures et 2 liaisons rotoïdes

Les résultats obtenus aux précédentes sections permettent de tirer plusieurs conclusions sur la pertinence de la méthode proposée.

D'abord, la figure 2.10 exprimant la progression de la réorientation pour la méthode sans limite de débattement angulaire montre que la combinaison de l'algorithme avec les 3 valeurs appropriées des paramètres ξ, γ_1, γ_2 et α permet d'effectuer la minimisation tout en évitant les minimums locaux de la fonction objective décrite à l'équation (2.4). De plus, la petite valeur finale obtenue pour le paramètre η montre que l'algorithme est capable de déterminer des trajectoires qui permettent de réorienter le robot très près de l'orientation désirée. Cependant, il semble que cette méthode ne soit pas adaptée, ou simplement pas assez raffinée, pour des applications où l'orientation finale du robot doit être atteinte de manière extrêmement précise, par exemple pour un satellite photographe. En contre partie, la méthode semble très intéressante pour des cas de réorientation lors d'un saut ou d'une chute où le robot possède une certaine compliance dans ses articulations permettant de le réorienter des derniers degrés manquant lors du contact avec le sol.

Ensuite, la figure 2.12 présentant la progression des angles d'Euler pour le cas plan montre que la réorientation a bien lieu dans le plan XY. Cela est dû au fait que les matrices d'inertie des membrures sont fortement diagonales, ce qui limite les tendances du robot à tourner autour d'un autre axe que celui de ses articulations. De plus, il faut mentionner que l'algorithme en tant que tel n'empêche pas une rotation selon l'angle ρ. En effet, la manière dont sont définis les objectifs finaux $\boldsymbol{s}_{i0}$, c'est-à-dire pointant tous vers la direction négative de l'axe Y, fait en sorte qu'une rotation dans le plan XZ n'a aucun poids dans le calcul de la fonction potentielle.

De plus, différents essais ont permis de conclure que l'algorithme est beaucoup plus puissant à trouver des trajectoires efficaces lorsque les vecteurs $\boldsymbol{s}_{ir}$ et $\boldsymbol{s}_{i0}$ sont initialement très éloignés l'un de l'autre. En effet, cela permet d'augmenter les possibilités de chemins à prendre par l'algorithme pour éviter les minimums locaux. Par ailleurs, il a été observé qu'il est pratiquement impossible de réorienter le robot d'une petite valeur de η avec l'algorithme actuel pour des mêmes configurations initiales et finales du robot, car cela fait obligatoirement passer l'algorithme par un minimum local. Une solution pour permettre une légère réorientation du robot serait de définir des vecteurs objectifs intermédiaires le plus éloignés possibles des vecteurs objectifs initiaux et finaux, et d'exécuter l'algorithme deux fois. Une autre possibilité serait d'utiliser les types de résultats présentés à la figure 2.14. Par exemple, exécuter deux fois une trajectoire qui permet de réorienter le robot plan d'une valeur de ψ égale à $\pi/2$ ferait en sorte que la réorientation désirée serait atteinte.

Qui plus est, un fait très intéressant peut être déduit des résultats présentés à la figure 2.14. En effet, ce n'est pas le déplacement des articulations ou les vitesses de celles-ci qui permettent

la réorientation, mais plutôt le ratio de vitesse entre eux. C'est donc dire que respecter ce ratio permet de réorienter le robot comme prévu, et qu'au contraire ne pas le faire a un impact négatif sur le degré de réorientation que peut atteindre le robot.

Un autre fait intéressant relatif au respect du ratio est que tout dépendant des limites de vitesse et d'accélération des articulations du robot, il est possible de faire varier la durée de la réorientation. Tant que le $\delta\boldsymbol{\lambda}$ est le même pour un pas de temps, il est possible de jouer sur la grandeur du vecteur $\dot{\boldsymbol{\lambda}}$ et sur la durée du pas de temps pendent lequel celui-ci est appliqué. Ainsi, si la capacité des moteurs le permet, il serait possible de diminuer ou d'allonger le temps de réorientation d'un facteur inverse aux vitesses à appliquer.

Finalement, comme vu pour les résultats de simulations avec contraintes de débattement angulaire au niveau des articulations, il semble très difficile, voire impossible de parvenir à réorienter un robot à 3 membrures dans le plan avec la méthode présentée sans dépasser ses limites articulaires et sans créer de collisions entre les membrures. Cependant, il n'est pas exclu que ça ne soit pas le cas pour des modèles de robot évoluant en 3 dimensions possédant plus de membrures et d'articulations. La section 2.1.2.5 tente d'explorer les possibilités qu'offre un tel robot.

2.1.2.5 Simulation de réorientation en 3 dimensions pour un robot à 4 membrures et 3 liaisons rotoïdes

Cette section présente les résultats obtenus en simulation avec l'algorithme présenté à la figure 2.7 pour un robot sériel à 4 membrures et 3 liaisons rotoïdes évoluant en 3 dimensions (3D) auquel on n'impose pas de limites de débattement articulaire.

La géométrie du robot utilisé pour effectuer les simulations est tirée de celle du prototype de robot plan présenté au chapitre 3. Ainsi, la membrure supplémentaire est créée en dédoublant la deuxième membrure du prototype et en l'insérant en troisième position de la série de membrure. Afin de permettre une réorientation en 3D, une liaison rotoïde axiale est insérée entre la deuxième et la troisième membrure. La première et la troisième liaison rotoïde sont de type radial. La direction de chacune des liaisons exprimée dans leur repère respectif est définie comme suit

$$\underline{\boldsymbol{e}}_1 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad \underline{\boldsymbol{e}}_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \quad \underline{\boldsymbol{e}}_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

Une représentation du robot ainsi créé est présentée à la figure 2.18

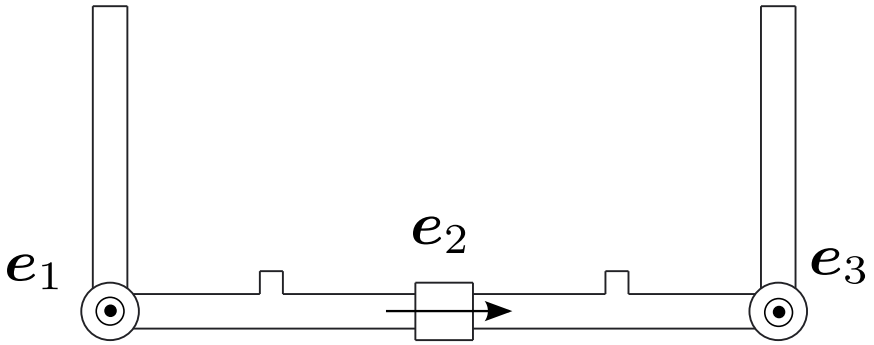


FIGURE 2.18 – Représentation d'un robot à 4 membrures et 3 liaisons rotoïdes.

Créer le robot à 4 membrures de cette manière fait en sorte que, contrairement à ce qui est mentionné au chapitre 3 au niveau de la conception du prototype, la géométrie de celui-ci n'est pas optimisée afin d'offrir les meilleurs résultats possibles avec l'algorithme. Cependant, même si ce n'est pas idéal, utiliser un tel modèle permet tout de même d'explorer les possibilités qu'offre l'algorithme avec un robot plus complexe.

L'objectif de la réorientation pour les simulations est de débuter et de terminer dans la même configuration des membrures, tout en ayant effectué une rotation de 180 degrés du robot autour de l'axe Z. Les configurations initiales et finales désirées pour la réorientation sont montrées à la figure 2.19

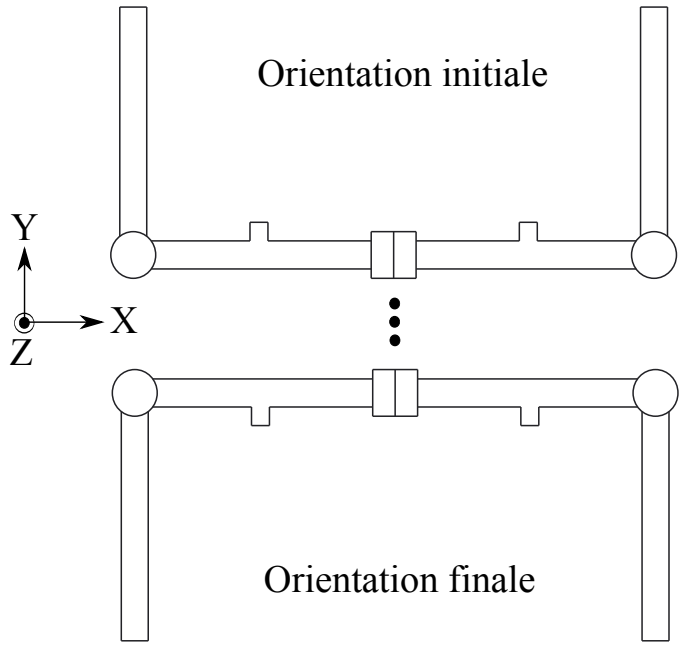


FIGURE 2.19 – Orientation initiale et finale désirée pour la réorientation du robot.

Suivant le schéma de la figure 2.19, on désire que $\boldsymbol{\theta}_0 = \boldsymbol{\theta}_f = [\frac{\pi}{2}\ 0\ \frac{\pi}{2}]^T$ avec $\boldsymbol{\beta}_0 = [0\ 0\ -\frac{\pi}{2}]^T$ et $\boldsymbol{\beta}_f = [0\ 0\ \frac{\pi}{2}]^T$. De plus, on pose que la direction initiale du vecteur $\boldsymbol{s}_{i0}$ est de $[0\ -1\ 0]^T$ pour $i = 1, \ldots, n$. Pour ce qui est du vecteur $\underline{\boldsymbol{s}}_{ir}$, pour $i = 1, \ldots, n$, celui-ci est défini comme suit

$$\underline{\boldsymbol{s}}_{1r} = \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix}, \quad \underline{\boldsymbol{s}}_{2r} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad \underline{\boldsymbol{s}}_{3r} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \quad \underline{\boldsymbol{s}}_{4r} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}.$$

Insérer ces valeurs dans l'équation (2.4) avec $n = 4$ donne une valeur initiale de η égale à 4.

Démarrer la fonction *fmincon* nécessite de fournir des estimés initiaux des 4 paramètres γ_1,γ_2, ξ et α. On définit ces estimés comme des valeurs aléatoires comprises entre des limites possibles que l'on pose pour chacun des paramètres. Ces limites sont les suivantes

$$\begin{aligned} -\pi \leq \gamma_1 \leq \pi \\ 0 \leq \gamma_2 \leq \pi \\ 0 \leq \xi \leq 1 \\ 0 \leq \alpha \leq 50. \end{aligned} \tag{2.15}$$

Le reste des paramètres de simulation sont les mêmes que ceux définis à la section 2.1.2.2.

Après avoir exécuté l'algorithme, la meilleure combinaison des valeurs des paramètres ξ, γ_1, γ_2 et α permet d'obtenir une valeur finale de η égale à 0,0805. Cela signifie qu'appliquer les trajectoires articulaires déterminées par l'algorithme permet de réorienter le robot d'environ 98%. Les valeurs numériques trouvées pour les 4 paramètres sont données au tableau 2.3

Tableau 2.3 – Valeur initiale des paramètres de simulation qui permettent de réorienter le robot dans la meilleure orientation finale pour un robot évoluant en 3D.

γ_1	−1,1941
γ_2	1,5519
ξ	0,7997
α	39,113

La progression du paramètre η le long de la trajectoire est présentée à la figure 2.20 et montre que la fonction ne rencontre pas de minimum locaux tout au long de la réorientation.

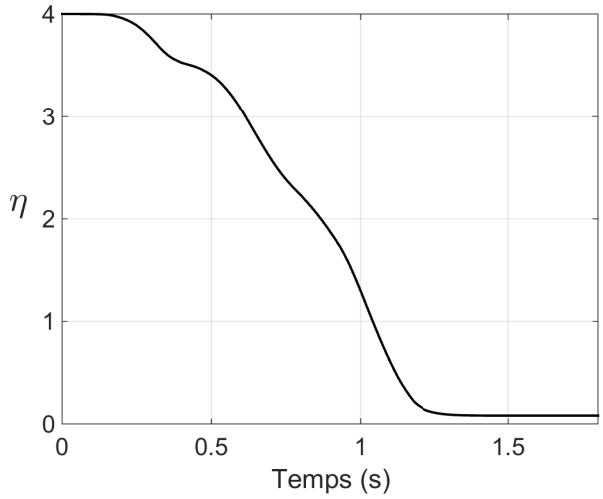


FIGURE 2.20 – Progression de la réorientation du robot exprimée par le scalaire η.

Les trajectoires de vitesses articulaires produites par l'algorithme sont obtenues directement du vecteur $\dot{\boldsymbol{\lambda}}$ calculé à chaque pas de temps de la simulation et sont présentées à la figure 2.21.

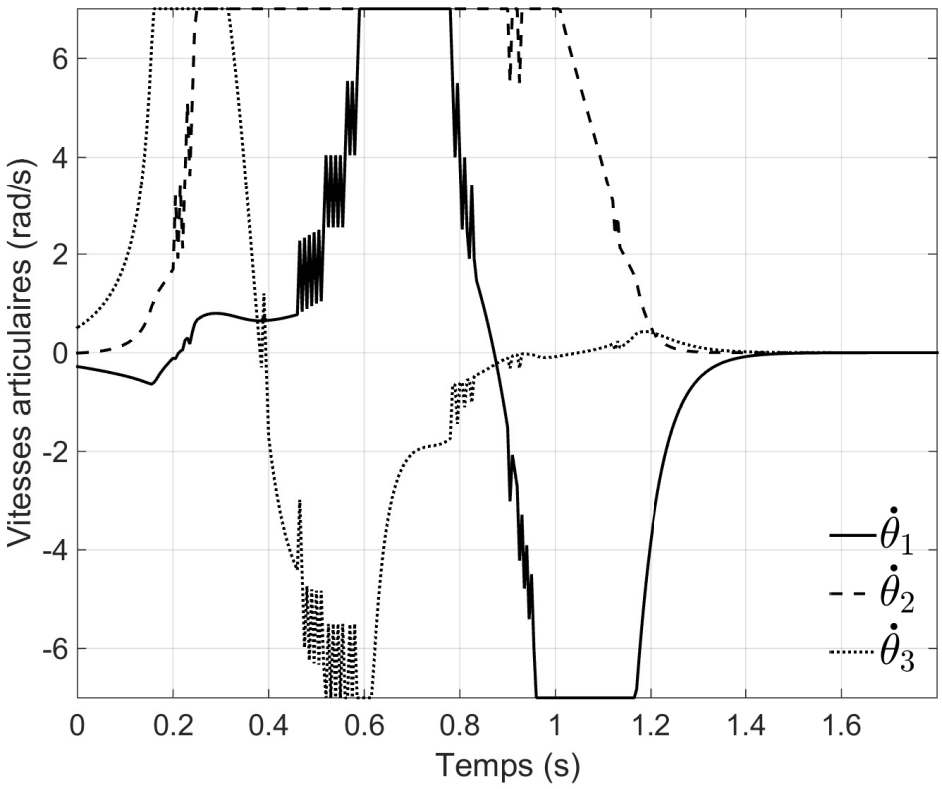


FIGURE 2.21 – Trajectoire des vitesses articulaires déterminées par l'algorithme.

Les figures 2.22 et 2.23 montrent respectivement la progression des angles d'Euler et des déplacements articulaires tout au long de la réorientation. La valeur du vecteur $\boldsymbol{\beta}$ à la fin de la réorientation est égale à $[-3{,}1533\ -0{,}0109\ -1{,}5822]^T$ et celle de $\boldsymbol{\theta}$ est de $[1{,}5182\ -0{,}0001\ 1{,}6255]^T$.

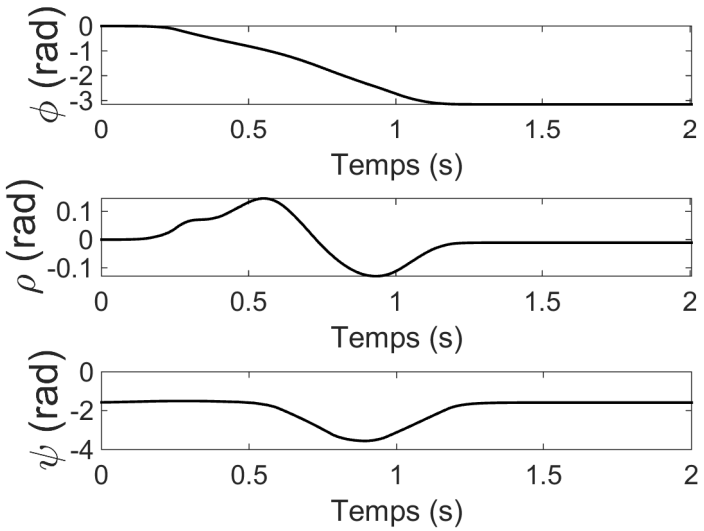


FIGURE 2.22 – Progression des angles d'Euler pendant la réorientation selon la convention XYZ.

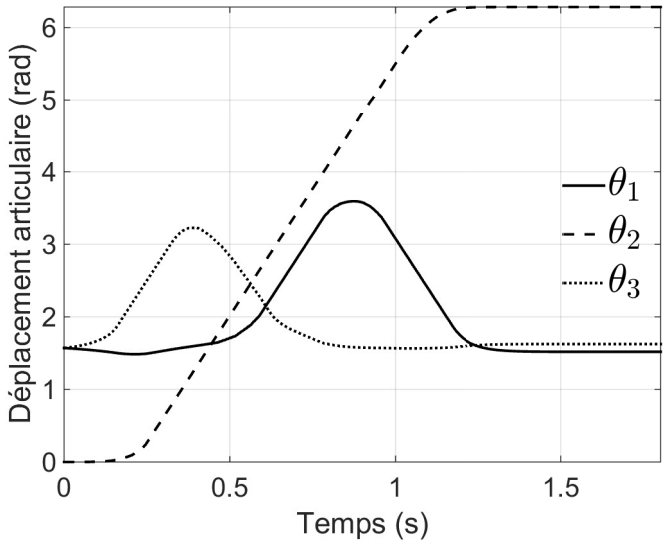


FIGURE 2.23 – Déplacements articulaires pendant la réorientation.

2.1.2.6 Discussion sur les résultats pour le robot à 4 membrures et 3 liaisons rotoïdes

Les résultats obtenus à la précédente section permettent de tirer plusieurs conclusions sur les performances de l'algorithme hors-ligne avec un modèle de robot évoluant en 3 dimensions.

D'abord, la figure 2.20 exprimant la progression de la réorientation montre que la combinaison de l'algorithme avec les 4 valeurs appropriées des paramètres ξ, γ_1, γ_2 et α permet encore une fois d'effectuer la minimisation tout en évitant les minimums locaux de la fonction objective. De plus, la petite valeur finale obtenue de η permet de croire qu'il serait possible d'obtenir de bien meilleurs résultats si la géométrie du robot était optimisée.

Ensuite, la figure 2.22 présentant la progression des angles d'Euler montre que le robot effectue sa réorientation dans l'espace, et qu'il termine sa course dans le même plan que celui d'origine. De plus, on peut observer que la valeur finale du vecteur $\boldsymbol{\beta}$ obtenue dans la simulation ($[-3{,}1533\ -0{,}0109\ -1{,}5822]^T$) est différente de celle qui était prescrite ($[0\ 0\ \frac{\pi}{2}]^T$). Cependant, ces deux vecteurs sont équivalents, dans le sens où leurs valeurs représentent les deux triplets d'angles d'Euler qui exprime l'orientation désirée de la membrure en fin de trajectoire. Qui plus est, utiliser comme orientation prescrite l'une ou l'autre des formulations n'a pas d'importance, car l'algorithme n'en tient pas compte pour déterminer les trajectoires articulaires.

On peut également observer sur la figure 2.21 la présence de sauts dans les trajectoires de vitesse pour certains pas de temps. Ce phénomène peut être expliqué par le fait que certains des éléments du vecteur $\boldsymbol{\theta}$ varient beaucoup à l'intérieur de la boîte de contrainte de limite de vitesse entre deux pas de temps de l'optimisation. De plus, la forme des courbes est essentiellement due au fait que les contraintes de secousse ($jerk$ en anglais) ne sont pas incluses dans le calcul de la boite de contrainte. Ainsi, même si les trajectoires ne sont pas esthétiques, celles-ci restent faisables car elles respectent les contraintes dynamiques des moteurs aux articulations.

Finalement, on peut constater que sans même tenter de limiter le débattement angulaire des articulations, l'algorithme a produit des trajectoires articulaires qui ne font pas effectuer de rotations complètes des liaisons radiales, et que ces trajectoires créent de légères collisions entre les membrures (la figure 2.23 montre que θ_1 et θ_3 atteignent une valeur dépassant légèrement π). De plus, il faut mentionner que les dites trajectoires ont été déterminées pour une géométrie de robot qui n'était pas nécessairement adaptée ou optimisée pour effectuer la réorientation. Ainsi, on peut supposer qu'en faisant plus d'itérations de l'algorithme, il serait possible de générer des trajectoires articulaires qui permettraient de réorienter le robot en évitant les collisions.

2.1.3 Algorithme adaptatif

2.1.3.1 Construction de l'algorithme

L'autre méthode basée sur l'algorithme principal de la section 2.1.1 est définie comme étant de type adaptatif, dans le sens où elle offre une flexibilité en adaptant les trajectoires de vitesse articulaire selon les différentes erreurs qui peuvent survenir. Globalement, la stratégie

proposée est de débuter la réorientation avec la meilleure trajectoire déterminée à la section 2.1.2 et de rajouter une part d'aléatoire sur les objectifs à atteindre en fin de trajectoire pour se rendre à l'orientation désirée.

Chacune des deux étapes de cet algorithme a ses avantages et ses fonctions précises. Le but principal d'utiliser une trajectoire déterminée par la méthode hors-ligne en début de réorientation est de garantir que le robot arrivera dans une orientation près de celle désirée sans avoir à se soucier de rencontrer de minimum locaux de la fonction. La dernière partie aléatoire de l'algorithme permet quand à elle de pallier aux différentes erreurs de contrôle qui pourraient survenir en pratique, et aussi de rajuster le tir lorsque la configuration initiale du robot n'est pas celle qui a été supposée pour calculer la trajectoire. Cela entraîne aussi la conséquence que l'algorithme doit être calculé en temps réel, et que l'utilisation d'instruments de mesure de l'orientation doit être faite.

La logique de l'algorithme va comme suit. On commence par appliquer les trajectoires articulaires définies par l'algorithme hors-ligne. Au moment où une valeur seuil de η est atteinte, l'algorithme tombe dans la deuxième phase dite aléatoire pour une durée maximum t_{lim}. Dans cette phase, les vecteurs objectifs $\boldsymbol{s}_{it}$ à utiliser dans l'équation 2.10 sont définis en combinant un vecteur unitaire aléatoire $\boldsymbol{s}_{ia}$ avec le vecteur $\boldsymbol{s}_{i0}$. Le vecteur $\boldsymbol{s}_{it}$ est ainsi défini comme suit

$$\boldsymbol{s}_{it} = \nu_i \boldsymbol{s}_{i0} + (1 - \nu_i) \frac{\boldsymbol{s}_{ia}}{\|\boldsymbol{s}_{ia}\|} \tag{2.16}$$

où ν_i est un scalaire proportionnel au degré de réorientation de chacune des membrures du robot et est défini de la manière suivante

$$\nu_i = 1 - \frac{1}{2}\|\boldsymbol{s}_{ir} - \boldsymbol{s}_{i0}\|. \tag{2.17}$$

À noter qu'il est possible que la valeur seuil de η ne soit pas atteinte à la première phase, mais qu'il n'y a pas beaucoup de chance que cela se produise. En effet, pour que cela arrive, il faudrait que l'orientation ou la configuration initiale du robot soit vraiment éloignée de celle présumée initialement, faisant en sorte que les trajectoires calculées par la méthode hors-ligne n'arrivent pas à réorienter convenablement le robot. La procédure de l'algorithme adaptatif est présentée à la figure 2.24.

(1) Prendre les valeurs de γ_1, γ_2, ξ et α déterminées par l'algorithme hors-ligne.
While $\frac{\delta\eta}{dt} < 0$ || t$\leq t_{lim}$ (η est calculé avec la fonction de l'équation (2.4))
(2) Calculer la valeur de la matrice $\boldsymbol{K}$.
(3) Calculer le vecteur $\dot{\boldsymbol{\lambda}}$ avec l'équation (2.3) et le $\tilde{\eta}$ de l'équation (2.10).
(4) Procéder à la mise au niveau du vecteur $\dot{\boldsymbol{\lambda}}$ au besoin.
(5) Déterminer la valeur de $\boldsymbol{\lambda}_{(t+1)}$: $\boldsymbol{\lambda}_{(t+1)}$=$\dot{\boldsymbol{\lambda}}\ \Delta t + \boldsymbol{\lambda}_t$.
(6) Calculer la valeur du vecteur $\boldsymbol{s}_{i(t+1)}$ avec
si $\eta > \eta_{seuil}$: équations (2.11) et (2.13)
si $\eta \leq \eta_{seuil}$: équation (2.16) et (2.17)
(7) Calculer la valeur de $\eta_{(t+1)}$ avec l'équation (2.4)
(8) $t = t + 1$
End While

FIGURE 2.24 – Structure de l'algorithme en mode adaptatif.

2.1.3.2 Simulation de réorientation pour un robot plan à 3 membrures et 2 liaisons rotoïdes

Les résultats des simulations faites pour l'algorithme adaptatif sont présentés ici. Les figures présentées montrent des résultats typiques d'une réorientation pouvant survenir en utilisant l'algorithme proposé.

La valeur de η_{seuil} utilisée pour passer à la partie aléatoire de l'algorithme est de 0,8, car celle-ci produit de bons résultats. Par ailleurs, il a été observé qu'une valeur trop basse de ce paramètre empêche de donner de la flexibilité en fin de trajectoire à l'algorithme, tandis qu'une valeur trop élevée offre la possibilité que la réorientation ne puisse avoir lieu convenablement dû au caractère aléatoire proportionnel au degré de réorientation inclus dans la définition du vecteur $\boldsymbol{s}_{it}$. De plus, le vecteur unitaire aléatoire utilisé pour calculer le vecteur $\boldsymbol{s}_{it}$ est défini dans le plan, ce qui signifie que sa composante selon l'axe Z est 0. Le pas de temps utilisé pour les simulations est de 0,005 seconde.

La figure 2.25 montre la progression du scalaire η tout au long de la réorientation. La valeur finale de η pour ce cas est de 0,0802.

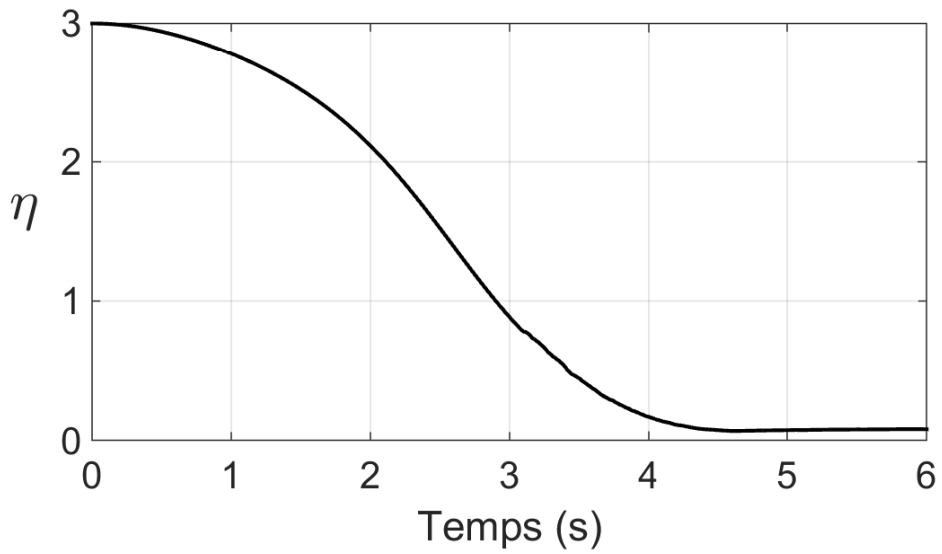


FIGURE 2.25 – Progression du paramètre η pendant la réorientation effectuée avec la méthode adaptative.

Les vitesses articulaires sont présentées à la figure 2.26.

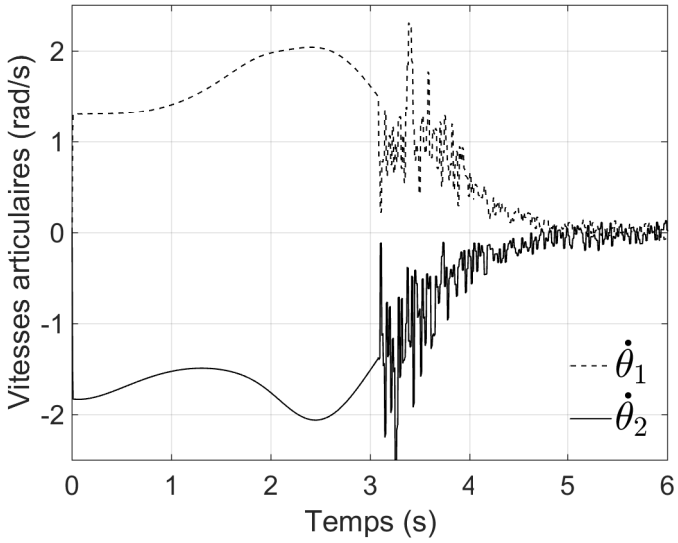


FIGURE 2.26 – Vitesses articulaires pendant la réorientation effectuée avec la méthode adaptative.

De plus, afin d'étudier la flexibilité qu'offre l'algorithme, une comparaison de celui-ci avec l'algorithme hors-ligne présenté à la section 2.1.2 est faite. Cette comparaison a été faite en exécutant les deux algorithmes en temps réel en utilisant les paramètres de simulation présentés au tableau 2.1, mais en faisant varier l'orientation initiale de l'angle ψ. La figure

2.27 montre les résultats obtenus. On peut ainsi observer que l'algorithme hors-ligne offre de meilleurs résultats pour une légère divergence de l'angle ψ, mais que l'algorithme adaptatif devient rapidement supérieur avec l'augmentation de cette divergence, notamment après une différence initiale de 0,6 rad où l'algorithme hors-ligne perd drastiquement de sa performance.

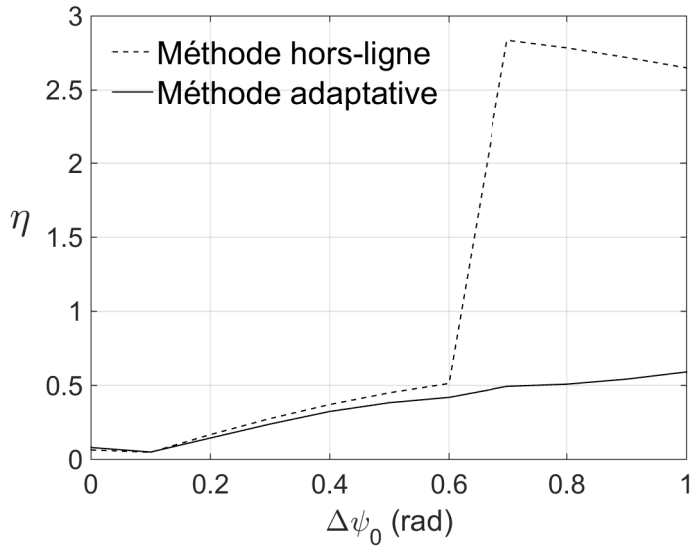


FIGURE 2.27 – Comparaison des résultats de la réorientation pour la méthode hors-ligne et adaptative selon une variation dans l'orientation initiale selon l'axe Z du robot.

2.1.3.3 Discussion

Les résultats pour la progression de la réorientation montrent que la méthode adaptative permet de réorienter le robot d'une manière quasi aussi performante que la méthode hors-ligne. De plus, la comparaison de cet algorithme effectué avec la méthode hors-ligne montre qu'il permet une plus grande capacité à rectifier le tir lorsque des erreurs de contrôle ou des divergences initiales sont présentes dans les paramètres de la simulation.

De plus, même si aucun résultat à ce sujet n'a été présenté, cet algorithme n'est pas vraiment adapté pour ce qui est de réorienter un robot avec des limites de débattement articulaire. En théorie, il serait possible d'amorcer la partie aléatoire de l'algorithme avant que les limites articulaires soient atteintes pour ensuite tenter de se rendre à l'orientation désirée en effectuant des mouvements majoritairement aléatoires tout en respectant les limites. Cependant, il a été jugé que procéder de la sorte est inefficace.

Bien sûr, exécuter cette méthode nécessite de faire des calculs en temps réel. Comme mentionné précédemment, il est important que le pas de temps entre les calculs de l'équation (2.3) soit assez petit pour respecter les contraintes dynamiques des moteurs et aussi permettre une progression fluide du gradient de la fonction potentielle. Cela correspond à dire que la fréquence de traitement de l'algorithme doit être le plus élevée possible. Divers essais ont été

réalisés en faisant varier le pas de temps utilisé en simulation, et il a été montré qu'un pas de temps inférieur à 0,1 seconde était suffisant pour permettre une réorientation du robot équivalente aux résultats présentés.

2.2 Méthode basée sur l'application de fonctions sinusoïdales aux articulations

Cette section propose d'étudier la pertinence et la faisabilité d'utiliser une stratégie de réorientation basée sur l'application de fonctions de vitesse sinusoïdales aux articulations. En faisant varier la forme des courbes de vitesse entre chaque articulation, on tente de déterminer quelle combinaison serait la plus efficace pour réorienter un robot en chute libre.

Utiliser des trajectoires sinusoïdales possède à priori quelques avantages. D'abord, on peut contrôler l'amplitude du mouvement des liaisons de manière à ne jamais dépasser leurs limites articulaires. Ensuite, appliquer de telles trajectoires pendant la durée de leur période signifie que le robot revient à sa configuration initiale. Ainsi, selon le degré de réorientation atteint pour une période, il serait possible d'effectuer la trajectoire pendant autant de périodes que nécessaire de manière à atteindre l'orientation désirée.

Afin de vérifier la faisabilité de la méthode proposée, la prochaine section effectue une simulation de génération de trajectoire pour un robot à 3 membrures et 2 liaisons rotoïdes.

2.2.1 Simulation de trajectoires sinusoïdales pour un robot à 3 membrures et 2 liaisons rotoïdes

Comme fait à la section 2.1.2, on utilise la fonction $fmincon$ pour déterminer la capacité des trajectoires sinusoïdales à réorienter le robot. Les simulations sont faites pour la durée de la plus longue période des trajectoires des articulations, au terme de laquelle on désire que la configuration du robot (l'angle θ_i des articulations entre les membrures) soit la même qu'au début. Les meilleures trajectoires sont ensuite identifiées comme celles produisant la meilleure progression du scalaire η au terme de la période.

Chacune des trajectoires articulaires relève de 3 paramètres à identifier, soit A_i l'amplitude de mouvement, ω_i la fréquence angulaire du mouvement ainsi que p_i la phase. On peut cependant poser immédiatement que dans le cas présent, la phase n'est pas une inconnue car celle-ci représente la valeur initiale de la coordonnée articulaire de chaque articulation θ_i et qu'elle vaut 0.

Ainsi, pour deux articulations du robot, on retrouve en tout 4 paramètres à déterminer créant ainsi les fonctions de coordonnées articulaires suivantes

$$\theta_1(t) = A_1 \sin(\omega_1 t) \tag{2.18}$$

$$\theta_2(t) = A_2 \sin(\omega_2 t) \tag{2.19}$$

que l'on dérive pour obtenir les trajectoires de vitesses articulaires à fournir aux articulations

$$\dot{\theta}_1(t) = A_1 \omega_1 \cos(\omega_1 t) \tag{2.20}$$

$$\dot{\theta}_2(t) = A_2 \omega_2 \cos(\omega_2 t). \tag{2.21}$$

On peut aussi établir un lien entre les fréquences angulaires, puisque l'on désire que les deux mouvements terminent en même temps. Cela signifie ainsi que le ratio entre les deux fréquences angulaires doit être un entier. Puisque la fonction $fmincon$ ne produit que des nombres décimaux, la manière de déterminer la valeur de l'inconnue s_1 représentant le ratio se fait par la règle suivante

$$s_1 = \begin{cases} \text{arrondi}(s_1) & s_{1\text{initial}} > 1 \\ (\text{arrondi}(\frac{1}{s_1}))^{-1} & 0 \leq s_{1\text{initial}} \leq 1 \end{cases} \tag{2.22}$$

où arrondi$(\cdot)$ est une fonction qui retourne la valeur entière la plus proche de son argument et avec

$$s_1 = \frac{\omega_1}{\omega_2} \tag{2.23}$$

La fonction $fmincon$, utilisée avec l'option $active-set$, est exécutée avec des estimés initiaux des paramètres à déterminer. Les valeurs limites de ces estimés sont définis comme suit

$$\begin{aligned} -2 \leq A_1 \leq 2 \\ -2 \leq A_2 \leq 2 \\ 0 \leq \omega_1 \leq 10 \\ 0 \leq s_1 \leq 2 \end{aligned} \tag{2.24}$$

Dans cette simulation, les valeurs maximales d'amplitude sont limitées de manière à ne pas dépasser les limites de vitesses articulaires des moteurs, car on désire voir l'effet de courbes sinusoïdales sur la réorientation, et non celui de sinus tronqués. Le tableau 2.4 présente les valeurs obtenues pour les 4 paramètres après exécution de la fonction $fmincon$.

Tableau 2.4 – Valeur des 4 paramètres déterminés par la fonction fmincon pour la réorientation avec la méthode de fonction sinus aux articulations.

A_1	1,9971
A_2	1,9971
ω_1	2,0433
s_1	2

La combinaison de ces paramètres donne pour un cycle de sinus, une valeur finale de η égale à 2,9889. Puisque la réorientation a lieu dans le plan XY, on peut également exprimer la réorientation en terme d'une variation de l'angle ψ correspondant à une valeur de 0,181 radian. La figure 2.28 présente la progression dans le temps de η et la figure 2.29 montre de quelle manière se déplace le robot selon les vitesses articulaires qui lui sont prescrites.

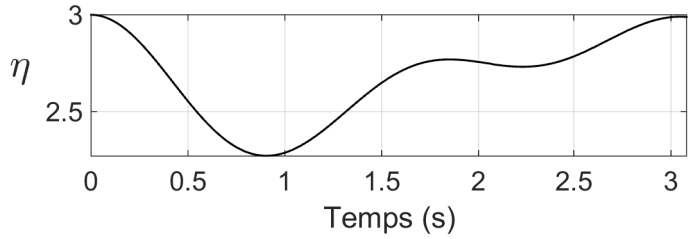


FIGURE 2.28 – Progression du scalaire η pour une période pendant la réorientation effectuée avec la méthode de fonction sinus aux articulations.

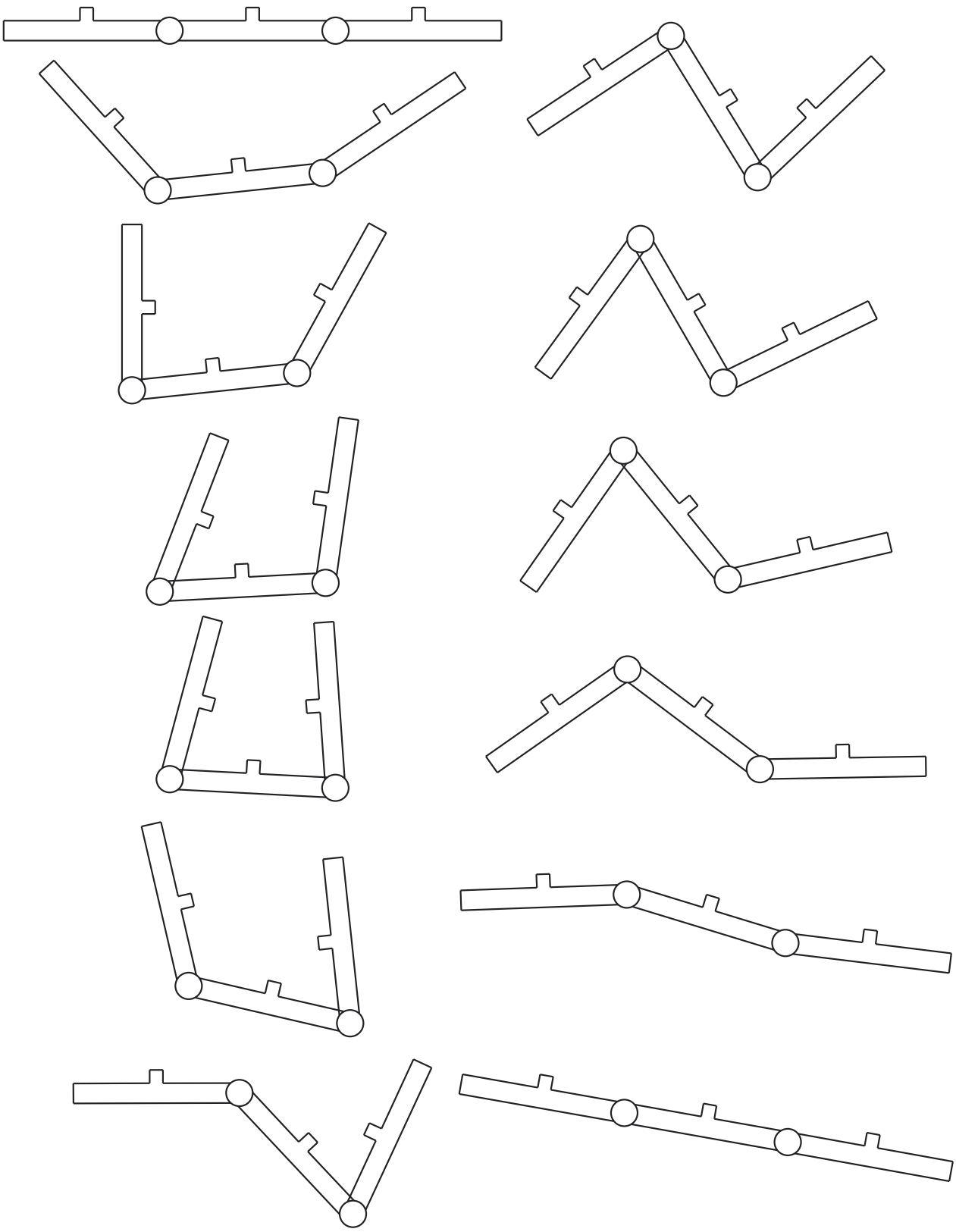


FIGURE 2.29 – Visualisation de la réorientation des mouvements sinus aux articulations.

Les figures 2.30 et 2.31 présentent respectivement la progression des éléments des vecteurs $\boldsymbol{\beta}$ et $\boldsymbol{\theta}$ pendant la réorientation et la figure 2.32 montre la progression des vitesses articulaires $\boldsymbol{\theta}_i$.

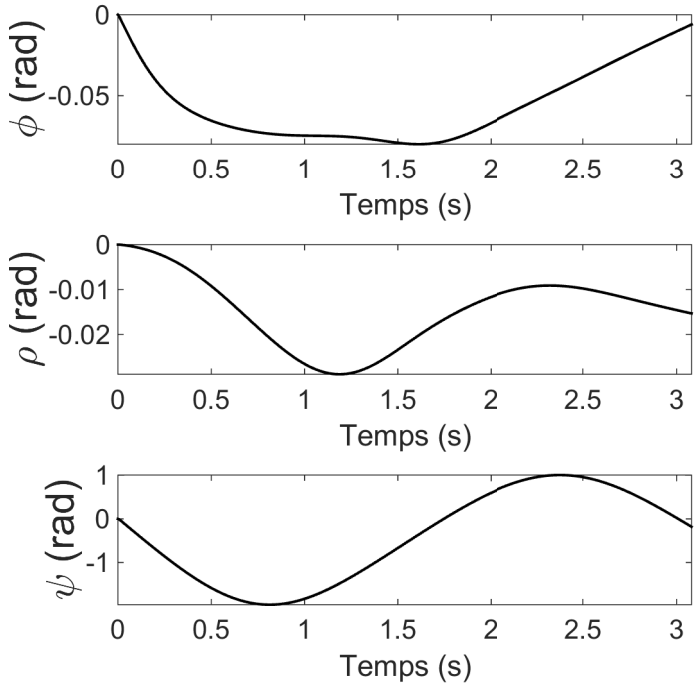


FIGURE 2.30 – Progression des angles d'Euler pour une période pendant la réorientation effectuée avec la méthode de fonction sinus aux articulations.

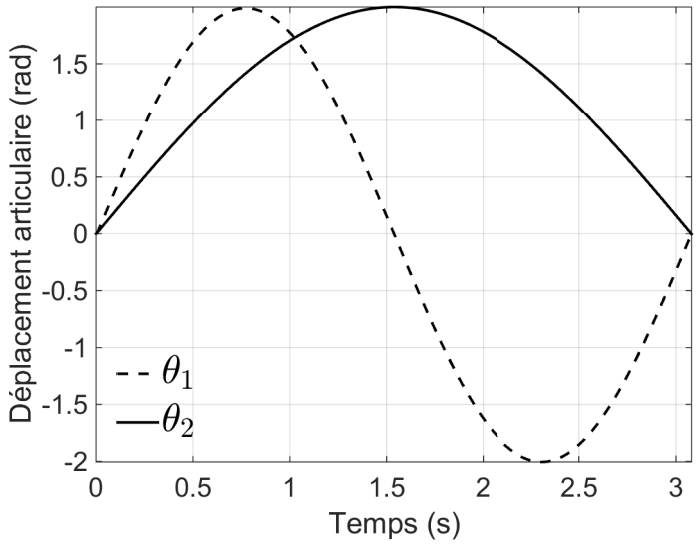


FIGURE 2.31 – Déplacements articulaires des liaisons rotoïdes pour une période pendant la réorientation effectuée avec la méthode de fonction sinus aux articulations.

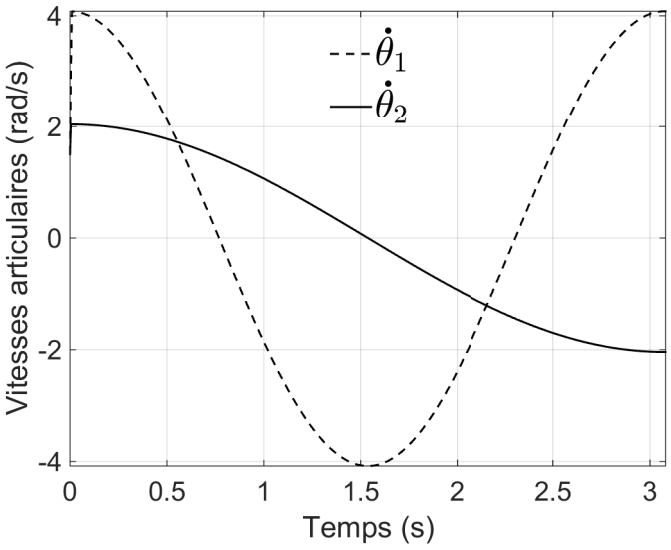


FIGURE 2.32 – Vitesses articulaires pour une période pendant la réorientation effectuée avec la méthode de fonction sinus aux articulations.

Suivant ces résultats, des essais ont été réalisés afin de connaître l'impact de la modification simultanée des amplitudes de mouvement sur l'orientation finale pouvant être atteinte. La figure 2.33 montre la variation de l'angle ψ obtenue selon l'amplitude de mouvement des articulations.

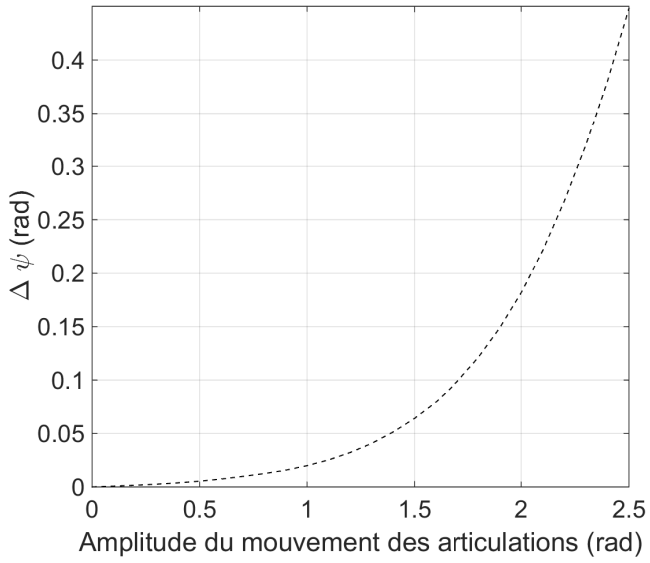


FIGURE 2.33 – Valeur finale de l'angle d'Euler ψ selon la valeur des amplitudes de mouvement des articulations.

2.2.2 Discussion

Cette section discute de la pertinence de la méthode proposée à la lumière des résultats obtenus.

D'abord, pour ce qui des des résultats obtenus avec la fonction $fmincon$, on arrive aux conclusions suivantes. Pour une amplitude de mouvement de 1,9971 radians, l'amélioration produite de η est de 0,0111, ce qui équivaut à une rotation de l'angle ψ de 0,178 radian. Cela revient à dire qu'il faudrait exécuter les trajectoires 18 fois pour atteindre l'orientation désirée, pour une durée totale d'environ 58 secondes.

Ensuite, on peut se référer à la figure 2.33 afin de connaître quels seraient les résultats si l'on ajoutait des limites de débattement articulaire aux liaisons rotoïdes. Si l'on pose que les limites sont les mêmes que celles utilisées pour la méthode hors-ligne, c'est-à-dire $\pi/2$, on obtient une différence de l'angle ψ d'environ 0,07 radian pour une période de temps. Tenter une réorientation complète avec avec des limites articulaires nécessiterait ainsi d'exécuter la trajectoire environ 45 fois, ce qui prendrait 144 secondes.

Même si appliquer de telles trajectoires ne semble pas optimal en terme de temps, cette méthode reste la seule de toutes celles présentées dans ce mémoire qui permette de réorienter un robot plan en respectant des contraintes de limites articulaires. De plus, cette méthode présente l'avantage, du moins en simulation, de permettre de très petits changements d'orientation du robot lorsque celle-ci doit être atteinte précisément. Cela fait aussi de cette méthode une solution complémentaire au deux autres présentées à la section 2.1. En effet, on pourrait utiliser ces dernières pour obtenir un réorientation générale du robot et ensuite venir peaufiner son orientation avec une période de la trajectoire sinusoïdale appropriée.

2.3 Conclusion

Ce chapitre a présenté deux types de méthode pour la réorientation d'un robot sériel en chute libre, soit une déterminant les vitesses articulaires des articulations par minimisation d'une fonction potentielle, et une autre appliquant des trajectoires de vitesse sinusoïdales aux articulations.

La première méthode a été testée avec plusieurs variantes, selon que l'on désire calculer les trajectoires avant ou pendant que la réorientation se produise. Les résultats en simulation pour le robot plan et 3D ont permis de montrer les forces des algorithmes développés. D'abord, ceux-ci réussissent à déterminer des trajectoires permettant une réorientation lorsque l'orientation initiale et finale du robot sont très éloignées l'une de l'autre. Ensuite, ils permettent une certaine flexibilité sur l'orientation initiale du robot qui n'affectent pas trop l'orientation finale pouvant être atteinte. Cependant, il a aussi été montré que les algorithmes ne permettent pas de réorienter un robot plan lorsqu'une limite de débattement articulaire est imposée.

La deuxième méthode a quant à elle permis de montrer la capacité de trajectoires sinusoïdales aux articulations à réorienter le robot. Il a été montré que même si cela prenait beaucoup de temps, la réorientation d'un robot plan avec cette méthode était possible, et qu'elle permettait aussi de fins ajustements sur l'orientation du robot. Il a aussi été mentionné que cette méthode était complémentaire à la première présentée.

Chapitre 3

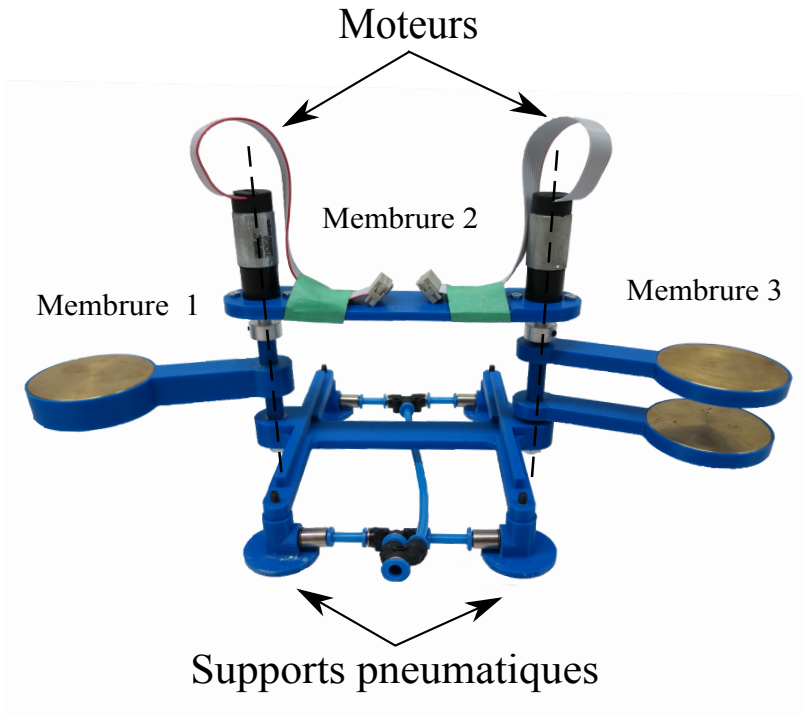
Construction d'un prototype et validation expérimentale

Ce chapitre présente dans un premier temps la manière dont est construit un prototype de robot sériel plan flottant ainsi que les diverses considérations qui ont été prises pour y arriver. Bien qu'en théorie les algorithmes développés au chapitre 2 peuvent être fonctionnels pour tous les types de géométrie de robot, il a été observé que de meilleurs résultats pouvaient être obtenus selon la manière dont était construit le robot. Des résultats expérimentaux obtenus avec les algorithmes développés sont ensuite présentés. Le but d'effectuer ces tests est d'abord de confirmer la validité du modèle dynamique présenté au chapitre 1, mais surtout d'illustrer la faisabilité de la mise en pratique des algorithmes.

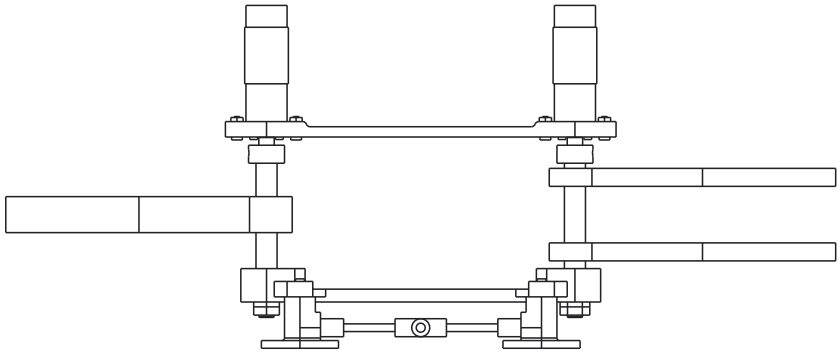
3.1 Développement du prototype

3.1.1 Construction du prototype

Afin de vérifier les résultats obtenus en simulation au chapitre 2, un prototype de robot sériel flottant a été construit. Le prototype est conçu pour évoluer dans un plan, et possède 3 membrures et 2 liaisons rotoïdes dont les axes sont parallèles entre eux et perpendiculaires au plan du mouvement, comme montré à la figure 3.1.



(a)



(b)

FIGURE 3.1 – a) Photo du prototype et b) vue de face du modèle CAO du prototype.

Afin de simuler la chute libre, le prototype est monté sur 4 pattes alimentées de manière pneumatique. Celles-ci lui permettent de flotter et de se déplacer avec très peu de friction sur une table de travail plane horizontale prévue à cet effet. Le câble d'alimentation pneumatique est connecté à la base du robot et est orienté perpendiculairement à la table de travail de manière à minimiser le couple extérieur qui pourrait être transmis au système lors de la réorientation. Pour cette même raison, le câble d'alimentation pneumatique ainsi que les fils électriques d'alimentation des moteurs des articulations sont choisis pour être les plus souples possible. De plus, les articulations du prototype sont conçues pour permettre un

débattement articulaire illimité et ainsi pouvoir effectuer n'importe quelle trajectoire produite par l'algorithme.

Afin de ne pas excéder le poids maximal que les pattes pneumatiques peuvent supporter, le prototype possède une masse totale de seulement 0,571 kg. La majorité des pièces du robot sont fabriquées par impression 3D, à l'exception des moteurs, des roulements situés aux articulations ainsi que des masses cylindriques en laiton insérées aux extrémités des membrures distales. Ces masses sont cruciales au bon retournement du robot, et les raisons de leur présence sont expliquées à la section 3.1.2.

Afin d'obtenir des résultats le plus reproductible possible entre les essais expérimentaux et les simulations, les valeurs des masses et des inerties ont été obtenues en faisant la modélisation de toutes les pièces du prototype dans le logiciel Pro/Engineer.

Les inerties du stator et du rotor de chaque moteur, obtenues à partir de leurs feuilles de spécifications, ont été attribuées respectivement à la membrure centrale et aux membrures distales du robot afin d'être calculées avec la membrure avec laquelle ils tournent. La valeur de l'inertie du rotor vue par les membrures distales, notée I_{rm}, peut être calculée par le théorème des axes parallèles [21] qui s'exprime comme suit

$$I_{rm} = r_r I_{rot} + m_{rot} d_{rot}^2 \tag{3.1}$$

où r_r est le ratio de réduction du réducteur, I_{rot} l'inertie du rotor, m_{rot} la masse du rotor et d_{rot} la distance projetée de l'axe du rotor jusqu'au centre de masse de la membrure. Ici, la valeur de r_r est de $1:24$ et celle de d_{rot} est de 0,0604 mètre. Il est à noter que procéder ainsi ne transfère que l'inertie du rotor selon l'axe Z, mais il a été jugé que cela était suffisant car la réorientation a lieu seulement selon cet axe.

La masse de chacune des membrures est donnée au tableau 3.1.

Tableau 3.1 – Valeur des masses en kg de chacune des membrure du prototype

m_1	m_2	m_3
0,193	0,184	0,194

Les inerties des membrures en kgm^2 exprimées par rapport à leur centre de masse sont

$$\boldsymbol{I}_1 = \begin{bmatrix} 1{,}1092e{-}4 & 0 & 3{,}9920e{-}5 \\ 0 & 2{,}7122e{-}4 & 0 \\ 3{,}9920e{-}5 & 0 & 3{,}1888e{-}4 \end{bmatrix} \tag{3.2}$$

$$\boldsymbol{I}_2 = \begin{bmatrix} 5{,}7901e{-}4 & 0 & 0 \\ 0 & 8{,}7646e{-}4 & 5{,}0266e{-}7 \\ 0 & 5{,}0266e{-}7 & 6{,}8869e{-}4 \end{bmatrix} \tag{3.3}$$

$$\boldsymbol{I}_3 = \begin{bmatrix} 7{,}6322e{-}5 & 0 & 3{,}7252e{-}5 \\ 0 & 2{,}3667e{-}4 & 0 \\ 3{,}7252e{-}5 & 0 & 3{,}1902e{-}4 \end{bmatrix} \tag{3.4}$$

La valeur en mètre des positions des vecteurs de construction barycentriques $\underline{\boldsymbol{l}}_{0i}$ et $\underline{\boldsymbol{r}}_{0i}$ sont les suivantes

$$\underline{\boldsymbol{r}}_{01} = \begin{bmatrix} 0{,}0604 \\ -2{,}3134e{-}5 \\ 0{,}0053 \end{bmatrix} \tag{3.5}$$

$$\underline{\boldsymbol{r}}_{02} = \begin{bmatrix} 0{,}0600 \\ -2{,}2979e{-}5 \\ -0{,}0048 \end{bmatrix} \tag{3.6}$$

$$\underline{\boldsymbol{l}}_{02} = \begin{bmatrix} -0{,}0600 \\ 2{,}2982e{-}5 \\ -0{,}0053 \end{bmatrix} \tag{3.7}$$

$$\underline{\boldsymbol{l}}_{03} = \begin{bmatrix} -0{,}0604 \\ 2{,}3134e{-}5 \\ 0{,}0049 \end{bmatrix} \tag{3.8}$$

De plus, afin de limiter les efforts radiaux exercés sur les arbres des moteurs, des valeurs limites de vitesse et d'accélération possibles ont été posées. Ces valeurs ont été utilisées pour les simulations et sont les suivantes

$$[\dot{\theta}_{min}, \dot{\theta}_{max}] = [-7, 7](rad/s) \tag{3.9}$$

$$[\ddot{\theta}_{min}, \ddot{\theta}_{max}] = [-300, 300](rad/s^2) \tag{3.10}$$

3.1.2 Considérations géométriques

Plusieurs considérations géométriques au niveau de la construction du prototype ont été prises en compte afin de lui permettre d'être réorienté convenablement par les algorithmes développés. En tout, 3 paramètres influent majoritairement sur la performance des algorithmes, soit la masse des membrures, leur inertie et la position des centres de masse par rapport aux articulations.

Les valeurs de ces paramètres sont étroitement liées entre elles, et tenter d'en modifier une a directement un impact sur les autres. C'est pourquoi une série d'essais sur la géométrie du robot ont été effectués afin d'optimiser les résultats pouvant être obtenus en simulation, principalement en modifiant la position et la masse des deux masses en laiton insérées sur les membrures distales du robot. De ces essais, certaines généralités ont été observées. D'abord, un ratio d'environ 1 pour 1 concernant la masse des membrures semble devoir être respecté. Il en est de même pour la position des centres de masse des membrures. Pour ce qui est de l'inertie, il semble qu'un ratio de 1 : 2 pour la composante en Z de la membrure distale sur la membrure du centre soit à respecter. Les valeurs des autres composantes des matrices d'inerties n'ont pas vraiment été optimisées, car tous les mouvements potentiels induits par la rotation des articulations hors du plan de réorientation sont contraints par la table de travail. De plus, puisque l'inertie des rotors est comprise dans le calcul de l'inertie des membrures distales, il est aussi très important de bien ajuster leur position par rapport au centre de masse des membrures. Ainsi, il a été calculé que l'inertie des rotors compte pour 34% de l'inertie des membrures distales, ce qui n'est pas négligeable.

Toutes ces considérations géométriques font en sorte que chacune des membrures représente environ le même fardeau à déplacer en terme de masse et d'inertie. Cela est primordial au bon fonctionnement des algorithmes, car la manière de calculer le degré de l'orientation du robot présenté à l'équation (2.4) ne prend pas en compte des variations d'importance entre les membrures. Une avenue qui n'a pas été exploitée dans ce mémoire serait de modifier la manière de calculer la valeur de la fonction potentielle, par exemple en ajoutant des poids proportionnels au fardeau pour chacune des membrures.

3.2 Validation expérimentale

Cette section présente les résultats expérimentaux obtenus avec le prototype selon les courbes de vitesse produites par l'algorithme en mode hors-ligne développé à la section 2.1.2. Des essais avec la méthode adaptative n'ont malheureusement pas pu être réalisés, car l'équipement utilisé (ordinateur et caméra) au moment des essais n'était pas assez puissant pour effectuer des calculs en temps réel nécessaires à l'application de cette méthode. De plus, l'équipement n'était plus disponible au moment où la méthode basée sur l'application de fonctions sinus aux articulations a été développée, et donc aucun essai expérimental n'a été réalisé pour cette méthode. Cependant, une projection de ce que seraient les résultats obtenus avec ces méthodes est incluse dans la discussion à la lumière des résultats obtenus pour la méthode hors-ligne.

3.2.1 Résultats expérimentaux avec la méthode hors-ligne

Les résultats de la réorientation obtenue avec la méthode hors-ligne sont présentés ici. La réorientation du prototype a été obtenue en appliquant directement les courbes calculées en

simulation aux moteurs. Pour les essais effectués, le contrôle des moteurs du robot a été fait via des contrôleurs reliés avec le logiciel *Matlab* et la fréquence de contrôle de ceux-ci est de 125 Hz. On s'assure ainsi que les articulations suivent bien la trajectoire calculée en simulation. Afin de mesurer la progression de l'orientation du robot, une caméra montée au dessus du plan de mouvement ainsi qu'à un marqueur posé sur la membrure centrale du robot ont été utilisés. Malheureusement, la fréquence de traitement des images provenant de la caméra étant très basse, il n'a pas été possible d'utiliser cette dernière pour mesurer la progression de l'orientation du robot en temps réel et ainsi obtenir une rétroaction sur l'orientation pour le calcul des vitesses dans l'algorithme.

Par ailleurs, les essais ont été effectués avant que les résultats présentés à la section 2.1.2 soient obtenus, et ont donc été réalisés avec les meilleures courbes qui étaient disponibles à ce moment. Ces courbes produisaient toutefois une valeur de réorientation finale de η égale à 0,188 correspondant à une réorientation du prototype de 94%, ce qui donne une configuration finale du robot assez près de celle désirée.

Des exemples typiques de résultats expérimentaux obtenus par l'application des trajectoires aux articulations sont montrés aux figures 3.2 et 3.3. Ces figures présentent respectivement une comparaison en simulation et en expérimentation de la progression des coordonnées articulaires ainsi que la progression de l'orientation, exprimée dans le repère R_0, de la membrure centrale du prototype.

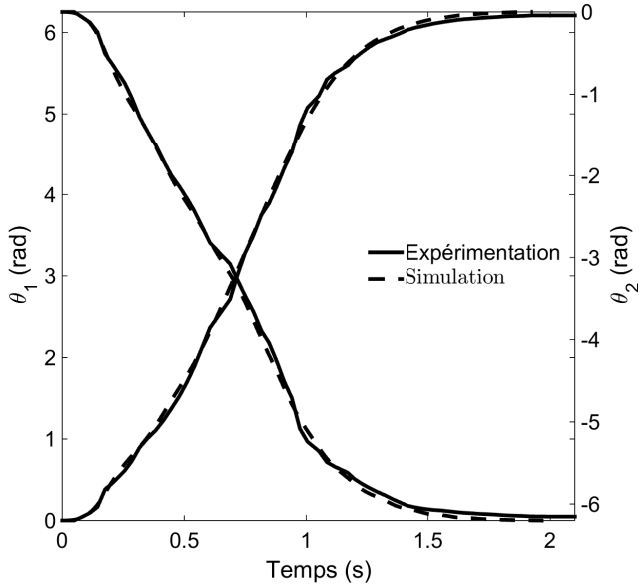


FIGURE 3.2 – Progression des coordonnées articulaires pendant la réorientation.

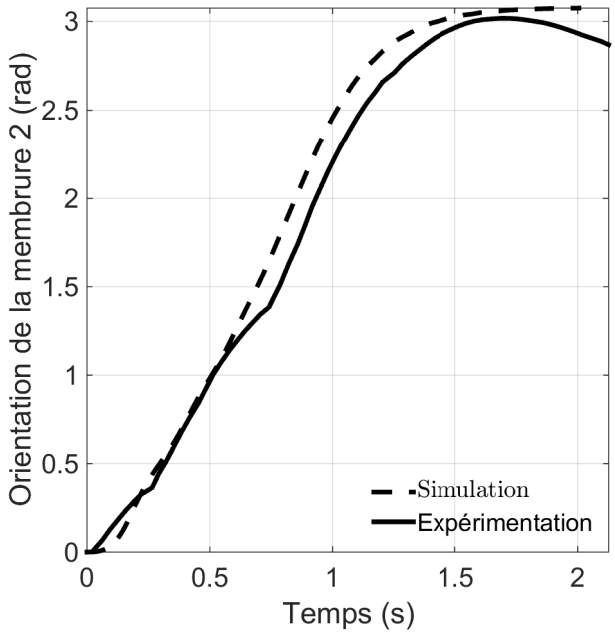


FIGURE 3.3 – Progression de l'orientation de la membrure centrale du prototype

La valeur finale de η relative à ces résultats est de 0,197, ce qui correspond à une réorientation globale de 93,4%.

Une vidéo présentant le prototype ainsi que la réorientation accomplie est disponible sur le site youtube.com [1].

3.2.2 Discussion

La figure 3.3 montre que la progression de l'orientation de la membrure centrale en expérimentation est sensiblement la même que celle en simulation pour le début de la trajectoire, mais qu'une légère divergence est observable à partir de la moitié de celle-ci. Même si les résultats ne sont pas exactement les mêmes, on peut tout de même affirmer que ceux-ci confirment la validité du modèle dynamique développé au chapitre 1, dans le sens où les divergences observées sont dues à des erreurs de contrôle et à des couples externes présents dans le système.

En effet, outre un léger hystéréris observable dans les articulations du robot ainsi qu'une possible divergence entre les inerties simulées et réelles des membrures, une des principales raisons de la divergence observable est le fait que le câble d'alimentation d'air comprimé applique un couple de rappel non-négligeable sur le système en fin de trajectoire. C'est ce qui explique la courbe descendante qui peut être observée pour la partie expérimentale sur la

1. La vidéo est disponible sur la chaîne du Laboratoire de robotique de l'université Laval et est intitulée *reorientation of a free-floating robot using internal motion.*

figure 3.3. Par ailleurs, il a été observé que lorsque les moteurs cessent de tourner, le robot tend à retourner tranquillement à son orientation initiale. Qui plus est, ce phénomène permet aussi de dire que le temps nécessaire pour effectuer la réorientation a un impact sur les résultats de celle-ci. Ainsi, on peut supposer que plus la réorientation est effectuée rapidement, plus celle-ci a le potentiel d'être bien accomplie.

Un autre facteur qui influence les performances de l'algorithme à retourner le robot en simulation est la capacité des contrôleurs des moteurs à reproduire les trajectoires de vitesses articulaires qui lui sont prescrites. Bien que de telles courbes ne peuvent être présentées en raison d'un manque de données, on peut tout de même se faire une idée de celles-ci en regardant la figure 3.2. En effet, on peut voir sur celle-ci que les courbes expérimentales de position oscillent légèrement de part et d'autre des courbes simulées, ce qui permet de dire que les vitesses expérimentales évoluent probablement de la même manière autour des vitesses simulées, et donc qu'elles ne sont pas reproduites parfaitement. Par ailleurs, il est aussi très important que le ratio de vitesse entre les courbes expérimentales soit le même qu'entre celles simulées pour produire la même réorientation. En effet, le vecteur $\dot{\boldsymbol{\lambda}}$ calculé à chaque pas de temps par l'algorithme représente un ratio à appliquer entre chacun de ses éléments pour arriver à l'orientation désirée. Par conséquent, à toutes les fois que ce ratio n'est pas respecté, le degré final de réorientation qui peut être atteint est diminué.

De plus, comme il a été mentionné, les essais expérimentaux ont été effectués sans procurer une rétroaction au système sur son orientation. Faire fonctionner l'algorithme en boucle fermée aurait ainsi la possibilité d'améliorer les résultats observés, car cela aurait comme effet de réagir activement aux erreurs de contrôle ainsi qu'aux couples externes qui ont un effet néfaste sur la réorientation. Cela permet aussi de supposer que l'algorithme adaptatif développé à la section 2.1.3 aurait potentiellement la chance d'obtenir de meilleurs résultats, d'autant plus qu'il a été montré à la figure 2.27 que cet algorithme réagit bien même lorsqu'il y a présence de divergences dans l'orientation des membrures.

Qui plus est, rien ne suggère qu'appliquer des fonctions de vitesse sinusoïdales aux articulations pour un robot plan en général ne fonctionnerait pas pour effectuer la réorientation, d'autant plus qu'il serait facile de faire plusieurs cycles supplémentaires afin de compenser pour les erreurs de contrôle réduisant le gain possible d'amélioration de l'orientation. Cependant, pour ce qui est d'utiliser cette technique avec le prototype construit, on peut aisément affirmer que les résultats obtenus seraient assez médiocres car le temps de réorientation deviendrait un facteur nuisible considérant le couple de rappel présent dans le système.

3.3 Conclusion

Ce chapitre a en premier lieu fait la présentation d'un prototype de robot plan sériel flottant qui a été utilisé pour effectuer les simulations et les essais expérimentaux. Diverses considé-

rations au niveau du design du robot ont dû être prises en compte afin de rendre possible la réorientation avec les algorithmes utilisés, et celles-ci ont été exposées.

Les résultats expérimentaux avec le prototype selon la méthode hors-ligne ont ensuite été présentés. Ceux-ci ont permis de conclure que la méthode hors-ligne était applicable si l'on était en mesure de respecter les ratios de vitesse prescrits entre les moteurs et de diminuer les couples extérieurs agissant sur le système. De plus, il a été montré qu'il était possible d'effectuer la réorientation en boucle ouverte, quoiqu'une rétroaction sur l'orientation du robot en temps réel améliorerait les résultats.

Conclusion

Ce mémoire a abordé la problématique de la réorientation de robots sériels en chute libre avec un moment angulaire initial nul. Parmi les différents types de méthodes existants pour réorienter de tels robot, il a été choisi d'explorer l'avenue de la réorientation par mouvements internes des membrures. Pour parvenir à valider la pertinence des solutions proposées, plusieurs étapes ont dû être suivies, et celles-ci ont été abordées dans les chapitres de ce mémoire.

Le chapitre 1 a d'abord permis de construire le modèle dynamique d'un robot sériel en chute libre. Un tel système est contraint à respecter la loi de la conservation du moment angulaire, et en posant les différentes relations en découlant, il a été possible d'établir un système matriciel permettant de déterminer les mouvements du robot en fonction des vitesses prescrites aux articulations. Afin d'assurer la validité de ce modèle, une comparaison de celui-ci, mis en oeuvre dans Matlab, a été effectuée avec le logiciel Adams.

Le chapitre 2 a proposé deux catégories de méthodes pour la réorientation de robot en chute libre. La première de celles-ci est basée sur l'algorithme de planification de trajectoire développée par Liégeois, et utilise une formulation adaptée du modèle dynamique ainsi qu'une fonction potentielle décrivant le degré de réorientation du robot. De cet algorithme, deux variantes ont été développées, et leurs résultats en simulation ont montré qu'elles étaient aptes à réorienter un robot plan sans limites articulaires. La deuxième catégorie de méthode présentée est basée sur l'application de fonctions sinusoïdales aux articulations du robot. Il a été montré qu'une telle méthode n'était pas vraiment optimale pour effectuer la réorientation d'un robot sériel, quoiqu'elle permettait de respecter les contraintes de débattement articulaire des liaisons rotoïdes. Néanmoins, celle-ci peut être utilisée en fin de trajectoire de manière à venir compléter les deux premières méthodes présentées.

Le chapitre 3 a présenté comment était construit un prototype de robot sériel plan flottant à 3 membrures et 2 liaisons rotoïdes. Certaines considérations géométriques au niveau du design du robot rendent possible la réorientation avec les méthodes développées au chapitre 2, et celles-ci ont été exposées. Des essais expérimentaux ont aussi été réalisés, et les résultats obtenus ont permis en premier lieu de valider le modèle dynamique développé, et en second lieu de déterminer les forces et faiblesses des algorithmes présentés.

En conclusion, ce mémoire a permis d'explorer et de décrire des approches novatrices dans le domaine de la réorientation de robot en chute libre par mouvements internes des membrures. Cependant, beaucoup de travail reste encore à accomplir afin de perfectionner les méthodes développées et de réorienter des robots dans une plus grande gamme de situations.

Notamment, il serait intéressant d'étudier les performances de l'algorithme pour des robots plus complexes possédant des membrures dont les propriétés inertielles et massiques sont très différentes entre elles. De tels robots nécessiteraient sans doute de procéder à des modifications dans la manière de définir les fonctions potentielles utilisées. Aussi, il serait de mise d'intégrer l'information provenant de plusieurs capteurs inertiels dans les algorithmes, et de développer des lois de contrôle utilisant cette information pour améliorer les résultats obtenus tant en simulation qu'en expérimentation.

Bibliographie

[1] N.M. Horri, P. Palmer, and A. Giffen. *Active Attitude Control Mechanisms*. John Wiley & Sons, Ltd, 2010.

[2] J. R. Wertz. *Spacecraft Attitude Determination and Control*. Reildel Publishing, Holland, 2002.

[3] J.S. White and Q.M. Hansen. *Study of systems using inertia wheels for precise attitude control of a satellite*, volume 691. National Aeronautics and Space Administration, 1961.

[4] T.R. Kane and M.P. Scher. A dynamical explanation of the falling cat phenomenon. *International Journal of Solids and Structures*, 5(7) :663, 1969.

[5] T.R. Kane, P.W. Likins, and D.A. Levinson. Spacecraft dynamics. *New York, McGraw-Hill Book Co, 1983, 445 p.*, 1, 1983.

[6] S. Dubowsky and E. Papadopoulos. The kinematics, dynamics, and control of free-flying and free-floating space robotic systems. *Robotics and Automation, IEEE Transactions on*, 9(5) :531–543, 1993.

[7] K. Yoshida and Y. Umetani. Control of space manipulators with generalized jacobian matrix. In *Space robotics : Dynamics and control*, pages 165–204. Springer, 1993.

[8] Y. Nakamura and R. Mukherjee. Nonholonomic path planning of space robots via a bi-directional approach. *IEEE transactions on Robotics and Automation*, volume 7(4) :500–514, August 1991.

[9] T. Suzuki and Y. Nakamura. Planning spiral motion of nonholonomic space robots. In *IEEE International Conference on Robotics and Automation*, volume 1, pages 718–725, April 1996.

[10] E. Papadopoulos and S. Dubowsky. Dynamic singularities in free-floating space manipulators. *ASME Journal of Dynamic Systems, Measurement, and Control*, Volume 115 :44–52, March 1993.

[11] J.T. Bingham, J. Lee, R.N. Haksar, J. Ueda, and C.K. Liu. Orienting in mid-air through configuration changes to achieve a rolling landing for reducing impact after a fall. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 3610–3617, Sept 2014.

[12] E. Chang-Siu, T. Libby, M. Tomizuka, and R.J. Full. A lizard-inspired active tail enables rapid maneuvers and dynamic stabilization in a terrestrial robot. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 1887–1894, Sept 2011.

[13] J. Zhao, T. Zhao, N. Xi, M.W. Mutka, and L. Xiao. Msu tailbot : Controlling aerial maneuver of a miniature-tailed jumping robot. *IEEE/ASME Transactions on Mechatronics*, volume 20 :2903–2914, 2015.

[14] E.C.-Y. Yang, P.C.-P. Chao, and C.-K. Sung. Optimal control of an under-actuated system for landing with desired postures. *Control Systems Technology, IEEE Transactions on*, volume 19(2) :248–255, 2011.

[15] N.V.R.K.N. Murthy and S.S. Keerthi. Optimal control of a somersaulting platform diver : A numerical approach. In *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*, pages 1013–1018. IEEE, 1993.

[16] M. Shahinpoor. *A robot engineering textbook*. Harper & Row Publishers, Inc., 1987.

[17] R.M. Murray, Z. Li, and S.S. Sastry. *A mathematical introduction to robotic manipulation*. CRC press, 1994.

[18] J. Wittenburg. *Dynamics of multibody systems*. Springer Science & Business Media, 2007.

[19] A. Liégeois. Automatic supervisory control of the configuration and behavior of multibody mechanisms. *IEEE Transactions on Systems, Man, and Cybernetics*, 7(12) :868–871, 1977.

[20] Y. Nakamura. *Advanced Robotics : Redundancy and Optimization*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition, 1990.

[21] J.L. Meriam and L.G. Kraige. *Engineering Mechanics Dynamics*. SI Version. John Wiley & Sons, Inc., sixth edition, 2008.

Annexe A

Développement du modèle dynamique pour un robot à 3 membrures et 2 liaisons rotoïdes

Cette annexe présente un exemple détaillé de construction du modèle dynamique d'un robot sériel en chute libre. Dans ce cas-ci, le modèle est créé pour un robot plan possédant 3 membrures et 2 liaisons rotoïdes dont les axes sont parallèles entre eux et orthogonaux au plan de réorientation, puisque c'est ce modèle de robot qui a été utilisé pour construire le prototype et effectuer la plupart des simulations.

Pour commencer, comme mentionné au chapitre 1, tous les vecteurs et tenseurs doivent être exprimés dans le même repère, ici R_0. Ce résultat est atteint en utilisant les matrices de rotations $\boldsymbol{Q}_i$ pour $i = 1, \ldots, n$. La formulation de la matrice $\boldsymbol{Q}_1$ est donnée directement à l'équation (1.5), tandis que la formulation des matrices de rotation qui font le changement de repère i à $i-1$ pour $i = 2, 3$ sont obtenus avec l'équation (1.9). En posant que le plan de réorientation est défini selon les axes X et Y, la direction des articulations peut être exprimée par un vecteur unitaire orienté dans la même direction que l'axe Z, soit

$$\underline{\boldsymbol{e}}_1 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \qquad \underline{\boldsymbol{e}}_2 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}. \tag{A.1}$$

Les matrices de rotation $\boldsymbol{Q}_2$ et $\boldsymbol{Q}_3$ sont maintenant calculées et sont représentées de la manière suivante

$$\boldsymbol{Q}_2 = \begin{bmatrix} \cos\theta_1 & \sin\theta_1 & 0 \\ -\sin\theta_1 & \cos\theta_1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \qquad \boldsymbol{Q}_3 = \begin{bmatrix} \cos\theta_2 & \sin\theta_2 & 0 \\ -\sin\theta_2 & \cos\theta_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{A.2}$$

où θ_1 et θ_2 sont respectivement les angles associés au mouvement des articulations.

Ensuite, le vecteur de position $\boldsymbol{r}_i$, de vitesse angulaire $\boldsymbol{\omega}_i$ et de vitesse cartésienne $\boldsymbol{v}_i$ des centres de masse de chaque membrure i est obtenu en utilisant la notion de barycentre qui fait intervenir les vecteurs de construction du robot $\boldsymbol{r}_{0i}$ et $\boldsymbol{l}_{0i}$. En combinant les équations (1.12) à (1.18), le vecteur $\boldsymbol{r}_i$ s'exprime de la manière suivante

$$\begin{aligned} \boldsymbol{r}_1 &= -\sum_{i=2}^{3}(\boldsymbol{r}_{0i-1} - \boldsymbol{l}_{0i})(1-\mu_i) \\ &= -\frac{(m_2+m_3)}{M}(\boldsymbol{r}_{01} - \boldsymbol{l}_{02}) - \frac{m_3}{M}(\boldsymbol{r}_{02} - \boldsymbol{l}_{03}) \qquad \text{(A.3)} \\ \boldsymbol{r}_2 &= \sum_{i=2}^{2}(\boldsymbol{r}_{0i-1} - \boldsymbol{l}_{0i})\mu_i - \sum_{i=3}^{3}(\boldsymbol{r}_{0i-1} - \boldsymbol{l}_{0i})(1-\mu_i) \\ &= \frac{m_1}{M}(\boldsymbol{r}_{01} - \boldsymbol{l}_{02}) - \frac{m_3}{M}(\boldsymbol{r}_{02} - \boldsymbol{l}_{03}) \qquad \text{(A.4)} \\ \boldsymbol{r}_3 &= \sum_{i=2}^{3}(\boldsymbol{r}_{0i-1} - \boldsymbol{l}_{0i})\mu_i \\ &= \frac{m_1}{M}(\boldsymbol{r}_{01} - \boldsymbol{l}_{02}) + \frac{(m_1+m_2)}{M}(\boldsymbol{r}_{02} - \boldsymbol{l}_{03}). \qquad \text{(A.5)} \end{aligned}$$

où m_i est la masse de chaque membrure i et M est la masse totale du robot.

Le vecteur $\boldsymbol{\omega}_i$ de chaque membrure i est obtenue de manière récursive avec l'équation (1.19) et est exprimé de la manière suivante

$$\boldsymbol{\omega}_1 = \boldsymbol{\omega}_1 \tag{A.6}$$

$$\boldsymbol{\omega}_2 = \boldsymbol{\omega}_1 + \dot{\theta_1}\boldsymbol{e}_1 \tag{A.7}$$

$$\boldsymbol{\omega}_3 = \boldsymbol{\omega}_1 + \dot{\theta_1}\boldsymbol{e}_1 + \dot{\theta_2}\boldsymbol{e}_2. \tag{A.8}$$

Le vecteur de vitesse cartésienne $\boldsymbol{v}_i$ de chaque membrure i est obtenue avec l'équation (1.20)

et est exprimé comme suit

$$\begin{aligned}
\boldsymbol{v}_1 &= \boldsymbol{\omega}_1 \times (-\frac{m_2+m_3}{M}\boldsymbol{r}_{01}) + \boldsymbol{\omega}_2 \times (\frac{m_2+m_3}{M}\boldsymbol{l}_{02} - \frac{m_3}{M}\boldsymbol{r}_{02}) + \boldsymbol{\omega}_3 \times (\frac{m_3}{M}\boldsymbol{l}_{03}) \\
&= \boldsymbol{\omega}_1 \times (-\frac{(m_2+m_3)}{M}(\boldsymbol{r}_{01} - \boldsymbol{l}_{02}) - \frac{m_3}{M}(\boldsymbol{r}_{02} - \boldsymbol{l}_{03})) + \\
&\quad \Big[\boldsymbol{e}_1 \times (\tfrac{m_2+m_3}{M}\boldsymbol{l}_{02} - \tfrac{m_3}{M}\boldsymbol{r}_{02} + \tfrac{m_3}{M}\boldsymbol{l}_{03}), \quad \boldsymbol{e}_2 \times (\tfrac{m_3}{M}\boldsymbol{l}_{03})\Big]\,\dot{\boldsymbol{\theta}} \\
&= \boldsymbol{\omega}_1 \times \boldsymbol{r}_1 + \boldsymbol{C}_1\dot{\boldsymbol{\theta}}
\end{aligned} \tag{A.9}$$

$$\begin{aligned}
\boldsymbol{v}_2 &= \boldsymbol{\omega}_1 \times (\frac{m_1}{M}\boldsymbol{r}_{01}) + \boldsymbol{\omega}_2 \times (-\frac{m_1}{M}\boldsymbol{l}_{02} - \frac{m_3}{M}\boldsymbol{r}_{02}) + \boldsymbol{\omega}_3 \times (\frac{m_3}{M}\boldsymbol{l}_{03}) \\
&= \boldsymbol{\omega}_1 \times (\frac{m_1}{M}(\boldsymbol{r}_{01} - \boldsymbol{l}_{02}) - \frac{m_3}{M}(\boldsymbol{r}_{02} - \boldsymbol{l}_{03})) + \\
&\quad \Big[\boldsymbol{e}_1 \times (-\tfrac{m_1}{M}\boldsymbol{l}_{02} - \tfrac{m_3}{M}\boldsymbol{r}_{02} + \tfrac{m_3}{M}\boldsymbol{l}_{03}), \quad \boldsymbol{e}_2 \times (\tfrac{m_3}{M}\boldsymbol{l}_{03})\Big]\,\dot{\boldsymbol{\theta}} \\
&= \boldsymbol{\omega}_1 \times \boldsymbol{r}_2 + \boldsymbol{C}_2\dot{\boldsymbol{\theta}}
\end{aligned} \tag{A.10}$$

$$\begin{aligned}
\boldsymbol{v}_3 &= \boldsymbol{\omega}_1 \times (\frac{m_1}{M}\boldsymbol{r}_{01}) + \boldsymbol{\omega}_2 \times (-\frac{m_1}{M}\boldsymbol{l}_{02} + \frac{m_1+m_2}{M}\boldsymbol{r}_{02}) + \boldsymbol{\omega}_3 \times (-\frac{m_1+m_2}{M}\boldsymbol{l}_{03}) \\
&= \boldsymbol{\omega}_1 \times (\frac{m_1}{M}(\boldsymbol{r}_{01} - \boldsymbol{l}_{02}) + \frac{(m_1+m_2)}{M}(\boldsymbol{r}_{02} - \boldsymbol{l}_{03})) + \\
&\quad \Big[\boldsymbol{e}_1 \times (-\tfrac{m_1}{M}\boldsymbol{l}_{02} + \tfrac{m_1+m_2}{M}\boldsymbol{r}_{02} - \tfrac{m_1+m_2}{M}\boldsymbol{l}_{03}), \quad \boldsymbol{e}_2 \times (-\tfrac{m_1+m_2}{M}\boldsymbol{l}_{03})\Big]\,\dot{\boldsymbol{\theta}} \\
&= \boldsymbol{\omega}_1 \times \boldsymbol{r}_3 + \boldsymbol{C}_3\dot{\boldsymbol{\theta}}
\end{aligned} \tag{A.11}$$

où les matrices $\boldsymbol{C}_i$ sont de dimension 3×2 et sont définies comme suit

$$\boldsymbol{C}_1 = \Big[\boldsymbol{e}_1 \times (\tfrac{m_2+m_3}{M}\boldsymbol{l}_{02} - \tfrac{m_3}{M}\boldsymbol{r}_{02} + \tfrac{m_3}{M}\boldsymbol{l}_{03}), \quad \boldsymbol{e}_2 \times (\tfrac{m_3}{M}\boldsymbol{l}_{03})\Big] \tag{A.12}$$

$$\boldsymbol{C}_2 = \Big[\boldsymbol{e}_1 \times (-\tfrac{m_1}{M}\boldsymbol{l}_{02} - \tfrac{m_3}{M}\boldsymbol{r}_{02} + \tfrac{m_3}{M}\boldsymbol{l}_{03}), \quad \boldsymbol{e}_2 \times (\tfrac{m_3}{M}\boldsymbol{l}_{03})\Big] \tag{A.13}$$

$$\boldsymbol{C}_3 = \Big[\boldsymbol{e}_1 \times (-\tfrac{m_1}{M}\boldsymbol{l}_{02} + \tfrac{m_1+m_2}{M}\boldsymbol{r}_{02} - \tfrac{m_1+m_2}{M}\boldsymbol{l}_{03}), \quad \boldsymbol{e}_2 \times (-\tfrac{m_1+m_2}{M}\boldsymbol{l}_{03})\Big] \tag{A.14}$$

La formulation du modèle dynamique du robot peut maintenant être déterminée en remplaçant tous les vecteurs déterminés plus haut dans l'équation de la conservation du moment angulaire (1.4). Le produit vectoriel double obtenu dans le processus peut être réarrangé grâce à l'identité suivante

$$m_i\boldsymbol{r}_i \times (\boldsymbol{\omega}_1 \times \boldsymbol{r}_i) = m_i(\boldsymbol{r}_i^T\boldsymbol{r}_i\mathbf{1} - \boldsymbol{r}_i\boldsymbol{r}_i^T)\boldsymbol{\omega}_1 \tag{A.15}$$

permettant ainsi de séparer tous les termes en $\boldsymbol{\omega}_1$ d'un côté de l'équation et tous ceux en $\dot{\boldsymbol{\theta}}$ de l'autre. Le système d'équation représentant le modèle dynamique du robot et présenté à l'équation (1.22) est ainsi défini

$$\boldsymbol{A}\boldsymbol{\omega}_1 = \boldsymbol{B}\dot{\boldsymbol{\theta}} \tag{A.16}$$

avec les matrices $\boldsymbol{A}$ et $\boldsymbol{B}$ de la forme

$$\begin{aligned}\boldsymbol{A} &= \boldsymbol{I}_1 + \boldsymbol{I}_2 + \boldsymbol{I}_3 + m_1(\boldsymbol{r}_1^T\boldsymbol{r}_1\,\mathbf{1} - \boldsymbol{r}_1\boldsymbol{r}_1^T) + \\ &\quad m_2(\boldsymbol{r}_2^T\boldsymbol{r}_2\,\mathbf{1} - \boldsymbol{r}_2\boldsymbol{r}_2^T) + m_3(\boldsymbol{r}_3^T\boldsymbol{r}_3\,\mathbf{1} - \boldsymbol{r}_3\boldsymbol{r}_3^T) \qquad \text{(A.17)}\\ \boldsymbol{B} &= -(\boldsymbol{I}_2\begin{bmatrix}\boldsymbol{e}_1, & \mathbf{0}\end{bmatrix} + \boldsymbol{I}_3\begin{bmatrix}\boldsymbol{e}_1, & \boldsymbol{e}_2\end{bmatrix} + \begin{bmatrix}m_1\boldsymbol{r}_1 \times \boldsymbol{C}_{11}, & m_1\boldsymbol{r}_1 \times \boldsymbol{C}_{12}\end{bmatrix} + \\ &\quad \begin{bmatrix}m_2\boldsymbol{r}_2 \times \boldsymbol{C}_{21}, & m_1\boldsymbol{r}_2 \times \boldsymbol{C}_{22}\end{bmatrix} + \begin{bmatrix}m_3\boldsymbol{r}_3 \times \boldsymbol{C}_{31}, & m_3\boldsymbol{r}_3 \times \boldsymbol{C}_{32}\end{bmatrix}) \qquad \text{(A.18)}\end{aligned}$$

où les vecteurs $\boldsymbol{C}_{ik}$, pour $k = 1, 2$, représentent les colonnes des matrices $\boldsymbol{C}_i$.