République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Ecole Nationale Polytechnique
Département de Génie Mécanique
Laboratoire de recherche Génie Mécanique et Développement

Ecole Doctorale « Mécanique et Ingénierie des Systèmes »
ENP–Univ. Boumerdes–Univ. Medea–Univ. Chlef

Mémoire de Magister en Génie Mécanique

Option : Construction Mécanique

Présenté par :

LAMMARI Halima
Ingénieur d'Etat en Génie Mécanique de l'USTHB

Intitulé

Etude d'une configuration générale d'un robot à bras manipulateur. Application au bras de mesure 3D

Soutenu le 29/09/2012 devant le jury composé de :

Président :	NOUR Abdelkader	Professeur	UMBB
Rapporteur :	BOUAZIZ Mohamed	Professeur	ENP
Examinateurs :	RECKAK Saïd	Professeur	ENP
	HADDAD Moussa	MC-A	EMP

ENP 2012

DEDICACE

Je dédie mon travail :

A ma chère maman ;
A mes sœurs AMINA et SABRINA ;
A l'homme de ma vie, mon mari AMINE ;
Au père de mon âme mon grand-père ;
A mes tantes et mes oncles et surtout ma tante MIMI ;
A mes amies et spécialement WAFA ;
A mes enseignants et mes collègues ;
A chaque personne qui m'aime.

Remerciements

Je tiens tout d'abord à remercier mon DIEU « ALLAH Le Tout Puissant» de m'avoir donné le courage ainsi que la force pour mener à bout ce modeste travail.

J'exprime ma profonde gratitude à monsieur Mohamed BOUAZIZ, professeur à l'Ecole Nationale Polytechnique, d'Alger, pour avoir assumé la responsabilité de m'encadrer, m'orienter et de me conseiller tout au long de la réalisation de ce travail, ainsi pour la confiance qu'il m'a accordé et sa patience avec moi.

Je tiens à remercier très vivement monsieur Moussa HADDAD, professeur à l'Ecole Militaire Polytechnique, d'Alger, de son aide qu'il m'a procurée pendant toute la durée de ce mémoire, ainsi que pour son grand encouragement.

Je tiens à présenter mes respectueux et vifs remerciements aux membres du jury, pour l'honneur qu'ils m'ont fait en acceptant d'être rapporteurs de mon mémoire et l'amabilité d'évaluer mon travail, en assistant à ma soutenance.

J'adresse aussi mes remerciements particulièrement à monsieur Abdelkrim BOUAFIA, professeur à l'USTHB, d'Alger, la première personne qui m'a ouvert les portes de la robotique. Et monsieur NEDARE pour son aide et son encouragement.

Je tiens à exprimer mes grands remerciements à monsieur HADJ RABAH, monsieur GHAFARE , madame KHEZNADJI, demoiselles KHANFARE et toutes les personnes de faculté de génie mécanique à l'USTHB, de leurs soutiens et leurs confiances de mes capacités

Je remercie également tous ceux qui ont contribué de près ou de loin à l'aboutissement de ce travail. Je m'excuse à l' avance à ceux que j'avais oubliés.

Merci.

Table des matières

Chapitre III MODELISATION CINEMATIQUE

Chapitre IV MODELISATION DYNAMIQUE

Chapitre V GENERATION DE MOUVEMENT

Table des figures

Liste des tableaux

Liste des abréviations

AFNOR : L'association Française de Normalisation

SMA : Structure Mécanique Articulée

P : Liaison prismatique

R : Liaison rotoïde

OT : Organe Terminal

DDL : Degrés De Libertés

DDM : Degrés De Mobilité

DDT : Degrés De libertés d'une Tâche

SCARA: Selective Compliance Arm for Robotic Assembly

JIRA: Japan Industrial Robot Association

RIA: Robot Institute of America

AFRI : Association Française de Robotique Industrielle

ISO 9946 : Organisation International de Normalisation concernant les caractéristiques des robots manipulateurs industrielles

RTL : Roulis -Tangage -Lacet

MGD : Model Géométrique Direct

MGI : Model Géométrique Inverse

MCD : Model Cinématique Direct

MCI: Model Cinématique Inverse

CAO : Conception Assisté par Ordinateur

CFAO : Conception Fabrication Assisté par Ordinateur

D-H: Denavit-Hertenberg

PTP: Point To Point

$S\theta_i$: Sin (θ_i)

$C\theta_i$: Cos (θ_i)

Introduction générale

INTRODUCTION GENERALE

Le monde industriel connait aujourd'hui une grande concurrence, ce qui ne permet plus de perte de temps, matériel et humaine. Cette concurrence exige de chercher de nouveaux moyens de fabrication pour des produits présentant un rapport qualité/prix compétitif. Mais le développement de produits est un processus lent et s'effectue avec des risques, car il nécessite plusieurs étapes de production.

Dans cette immense compétition sans fin et progressive, l'entreprise doit être équipée de systèmes performants pour une bonne gestion de ses ressources humaines et matérielles, et exploiter pleinement les nouvelles technologies qui permettent à la fois d'augmenter les bénéfices et la productivité, et de facilité les tâches humaines.

Pour atteindre ces objectifs, la robotique peut être utilisée. En effet, elle est actuellement intégrée dans plusieurs domaines de production, car elle permet des tâches automatisées avec une bonne précision, de réduire ou même de remplacer l'intervention de l'homme dans certaines activités.

On trouve de nombreuses structures de robots : robots à structure série, arborescente ou fermée, etc…

Les premiers robots sont des robots à structure série simple, semblables aux bras humains. Ces types de robots connaissent un succès dans l'industrie, en particulier dans le domaine de la construction automobile et aéronautique. Ils peuvent réaliser rapidement des tâches répétitives. On les emploie aussi dans les chaînes de fabrication et de montage, et également dans des environnements difficilement supportables par l'homme (conditions extrêmes de température ou de pression, radioactivité élevée, espace exigu, dangereux, etc.).

En général, les robots peuvent être à plusieurs degrés de liberté (rotations et translations). Six degrés de liberté (6 ddl) sont suffisants pour positionner et orienter l'organe terminal (effecteur), au-delà, la structure est dite redondante.

L'étude d'un robot repose essentiellement sur une modélisation mathématique comportant la géométrie, la cinématique et la dynamique, respectant les objectifs et les contraintes de la tâche et les performances recherchées. Evidemment, il est plus simple de faire des simulations de fonctionnement du robot (trajectoire, mouvement, stabilité) car elles sont plus faciles à établir, plus rapides, plus commodes, moins chères à employer, et permettent à l'utilisateur d'exécuter des expériences sans risque d'endommager le robot.

Dans la littérature, on trouve de nombreux travaux sur le calcul de modélisation des robots. Nous citons les plus pertinents en rapport avec notre sujet de recherche : [khal99], [Domb01], [Lall94], [Chal10], [Rena84], [Hadd11], [Boim01], etc. Dans l'ensemble de ces ouvrages, on trouve différentes méthodes générales pour la modélisation géométrique, cinématique et dynamique, ainsi la génération du mouvement. Nous nous en sommes inspirés pour étudier une configuration générale d'un robot série à 6 ddl en vue d'un calcul automatisé. Cette étude offre la possibilité d'être exploitée en mesure 3D pour un robot à 5 ddl.

L'intérêt de ce travail est de permettre de minimiser l'intervention des opérateurs dans la gestion des diverses étapes de calculs des modèles mathématique pour la commande de mouvement du robot, et de réduire le temps de calcul.

Le travail s'articule autour de six chapitres :

Le premier chapitre – ***terminologie et définitions générales*** - s'intéresse à faire le tour sur quelques généralités et définitions de base utilisées en robotique.

Le deuxième - ***modélisation géométrique des robots*** – expose les outils mathématiques utilisés pour le calcul des modèles géométrique direct et inverse d'un robot série. Ainsi, une procédure est indiquée montrant comment faire le choix des repères, définir les paramètres (convention de Denavit-Hertenberg Modifiée), et les problèmes dus à l'inversion du modèle géométrique direct.

Au troisième chapitre - ***modélisation cinématique*** - on établit les relations entre les vitesses et les accélérations entre les coordonnées opérationnelles et articulaires en utilisant le jacobien à la modélisation cinématique directe du premier et second ordre, et son inverse pour la modélisation cinématique inverse du premier et second ordre. Le jacobien inverse est résolu par l'algorithme de Greville.

Le quatrième chapitre - ***modélisation dynamique*** - présente le modèle dynamique inverse, ou tout simplement le modèle dynamique, pour un manipulateur à structure ouverte simple en employant les formalismes de Lagrange et de Newton-Euler. Pour des raisons de commodité, la programmation est faite avec le deuxième formalisme.

Le cinquième chapitre – ***Génération de mouvement*** – traite les différentes techniques (interpolations linéaires, de degré 3, 5, lois de mouvement Bang-Bang, trapèze) permettant de générer une trajectoire quelconque entre deux points, soit dans l'espace opérationnelle, soit dans l'espace articulaire.

Le sixième et dernier chapitre – ***Application au bras de mesure 3D –*** est consacré à une synthèse des chapitres précédents et se matérialise par l'étude d'un robot manipulateur à chaîne ouverte simple à cinq degrés de liberté en rotations, pour les mesures des pièces ayant une forme géométrique complexe.

CHAPITRE I

Terminologie et définitions Générales

Chapitre I

TERMINOLOGIE ET DEFINITIONS GENERALES

I .1. INTRODUCTION

Larousse définit un robot comme étant un appareil automatique capable de manipuler des objets ou d'exécuter des opérations selon un programme fixe ou modifiable.

L'Association Française de Normalisation (AFNOR) [Khal99], [Boim01] définit un robot comme étant un système mécanique de type manipulateur commandé en position, reprogrammable, polyvalent, à plusieurs degrés de liberté, capable de manipuler des matériaux, des pièces, des outils et des dispositifs spécialisés, au cours de mouvements variables et programmés pour l'exécution d'une variété de tâches. Il a souvent l'apparence d'un ou de plusieurs bras se terminant par un poignet. Son unité de commande utilise, notamment, un dispositif de mémoire et éventuellement de perception et d'adaptation à l'environnement et aux circonstances. Ces machines polyvalentes sont généralement étudiées pour effectuer la même fonction de façon cyclique et peuvent être adaptées à d'autres fonctions sans modification permanente du matériel.

Dans ce premier chapitre, nous commençons par présenter quelques définitions de base qui sont nécessaires à la compréhension des notions de robotique.

Figure I.1 : Robot industriel [Ref1]

I.2. STRUCTURE GENERALE D'UN ROBOT MANIPULATEUR [Rena84]

Un robot manipulateur est l'ensemble formé par :

- Une structure mécanique qui supporte l'organe terminal à situer ;

- Des actionneurs qui servent à agir sur la structure mécanique ;
- Les capteurs divers nécessaires à la commande ;
- Le système de commande qui pilote les actionneurs du robot ;
- Un système décisionnel qui assure la fonction de raisonnement et élabore le mouvement du robot ;
- Un système de communication qui gère les messages transmis entre le système décisionnel et l'opérateur ;

La figure I.2 illustre la composition de la structure générale d'un robot manipulateur.

Dans notre travail, on s'intéressera à la structure mécanique du robot (partie opérationnelle).

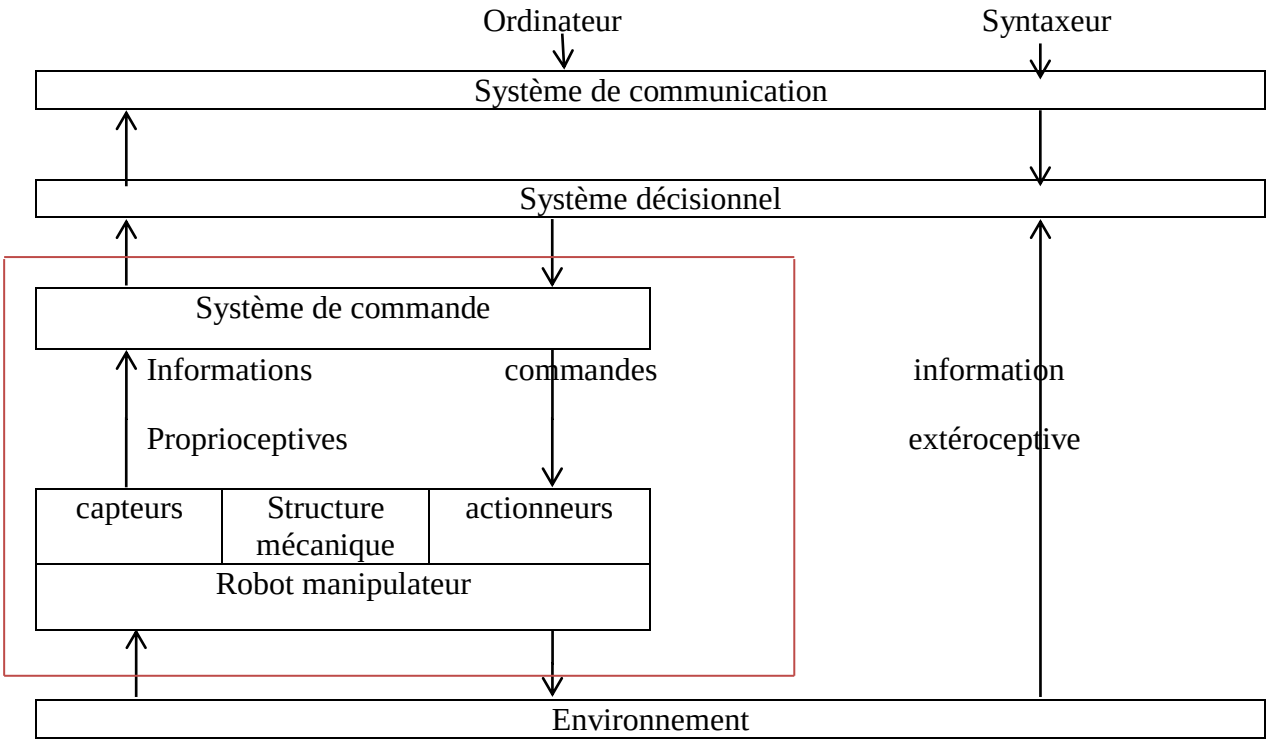


Figure I.2 : Structure générale d'un robot manipulateur.

I.3. STRUCTURE MECANIQUE ARTICULEE (SMA)

Un robot manipulateur est constitué généralement par deux sous-ensembles distincts : un (ou plusieurs) organe terminal qui est le dispositif destiné à manipuler des objets, et une structure mécanique articulée (SMA), constituée d'un ensemble de solides reliés entre eux, généralement, les uns à la suite des autres par des liaisons usuelles ou articulations, donc chaque solide est mobile par rapport au précédent.

Une structure mécanique articulée peut être représentée par une architecture composée de plusieurs chaînes de corps rigides assemblés par des liaisons appelées articulations, il existe différentes topologies de la chaîne cinématique (figures I.3 à I.7) [Khal99].

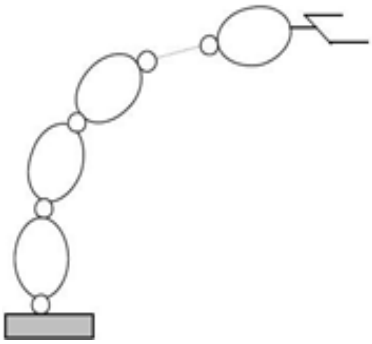


Figure I.3: Structure ouverte simple.

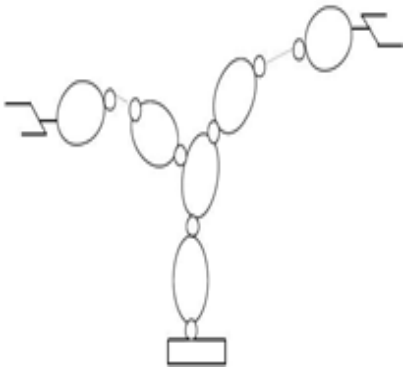


Figure I.4: Structure arborescente.

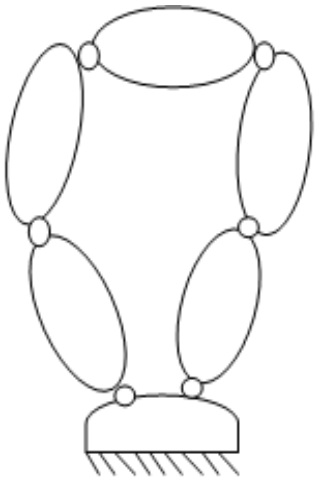


Figure I.5 : Structure fermée simple.

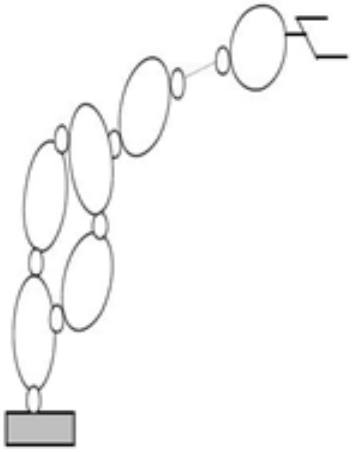


Figure I.6: Structure fermée.

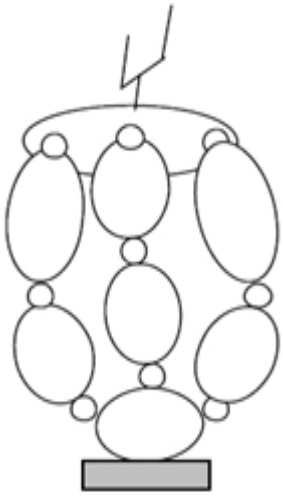


Figure I.7 : Structure Parallèle.

Comme on le constate, les structures de robots manipulateurs étant nombreuses, on se restreint dans ce travail aux robots manipulateur à structure ouverte simple (figure I.3).

I.4. LES LIAISONS USUELLES (ARTICULATIONS) [Khal99][Boim01][Rena84]

Une liaison cinématique entre deux solides est caractérisée par les degrés de liberté qu'elle autorise et permet une mobilité relative entre ceux-ci.

A un degré de liberté correspond la possibilité d'un mouvement de rotation ou de translation entre deux solides. En robotique, on rencontre fréquemment les articulations de types rotoïde (dite aussi pivot) et prismatique (ou glissière) (Figures I.8 et I.9).

A- Articulation rotoïde (pivot)

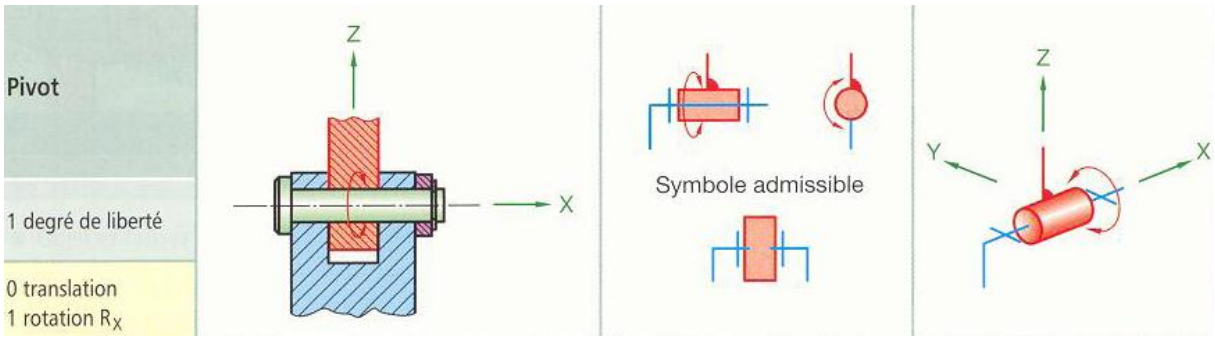


Figure I.8 : Liaison pivot. [Chev04]

B- Articulation prismatique (glissière)

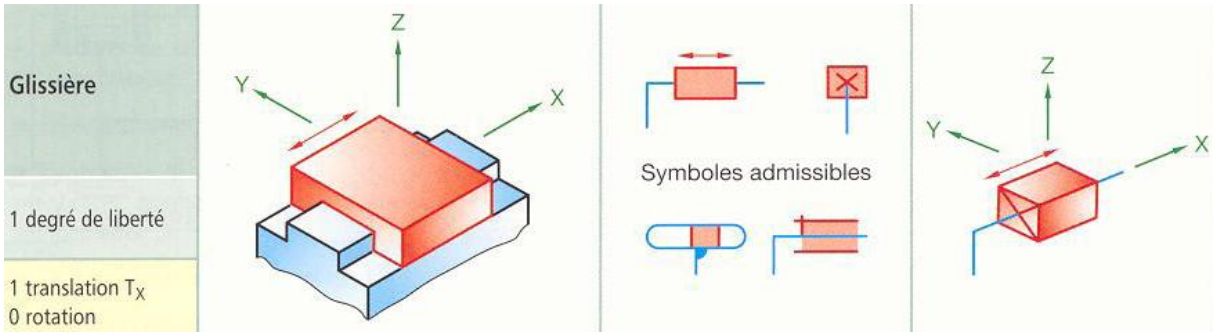


Figure I.9 : Articulation glissière. [Chev04]

A noter qu'il est possible de créer des articulations ayant une mobilité **m>1** en combinant les articulations simples. Par exemple, une liaison rotule **(m=3 rotations)** est obtenue avec trois liaisons rotoïdes d'axes concourants.

I.5. MOBILITE D'UNE STRUCTURE MECANIQUE ARTICULEE (S.M.A)

Le positionnement d'un solide dans l'espace nécessite 6 paramètres ou mouvements indépendants. Trois paramètres indépendants définissent la position (porteur) et les trois autres déterminent l'orientation du solide (poignet). Toutefois, ce même positionnement (localisation et orientation) peut résulter de l'association d'une multitude de mouvements réels, chacun d'eux pouvant être une translation ou une rotation. Dans le cas d'un système mécanique articulé, les degrés de mobilité **(DDM)** sont liés aux nombres d'axes ou de liaisons motorisées**.** Les degrés de liberté **(DDL)** du robot sont associés au nombre de déplacements indépendants (vis-à-vis du repère fixe) que peut subir l'organe terminal. L'inégalité suivante est toujours vérifiée : **DDL ≤ DDM.**

I.6. DEGRE DE LIBERTE D'UNE TÂCHE [Rena84]

Le degré de liberté d'une tâche **(DLT)** est égal au nombre de paramètres indépendants qui permettent de fixer toutes les situations que l'organe terminal doit atteindre.

I.7. CHOIX DU NOMBRE DE DEGRES DE LIBERTE D'UN ROBOT [Khal99]

Pour manipuler un solide, un robot peut disposer de 6 degrés de liberté. Toutefois, si ce solide présente une symétrie de révolution, cinq degrés liberté suffisent car il n'est pas nécessaire de spécifier la rotation autour de l'axe de révolution. De même, pour situer un corps dans un plan, il suffit qu'il y ait trois degrés de liberté: deux fixent les coordonnées d'un point du corps dans le plan et le troisième son orientation dans ce plan. A partir de ces constatations, on déduit que :

- les caractéristiques des solides manipulés par le robot, donc la classe de tâches à réaliser, permettent de déterminer le nombre de degrés de liberté dont il doit disposer.

- une condition nécessaire mais non suffisante pour qu'il y ait compatibilité entre le robot et la tâche réside dans le fait que le nombre de degrés de liberté de l'organe terminal du robot soit supérieur ou égale à celui de la tâche **(DDL≥DLT)**, le mécanisme peut alors placer l'organe terminal dans la situation désirée.

I.8. ELEMENTS CONSTITUANT UN ROBOT

La structure mécanique d'un robot se distingue par quatre ensembles :

- Le véhicule ;
- Le porteur ;
- Le poignet ;
- L'organe terminal**.**

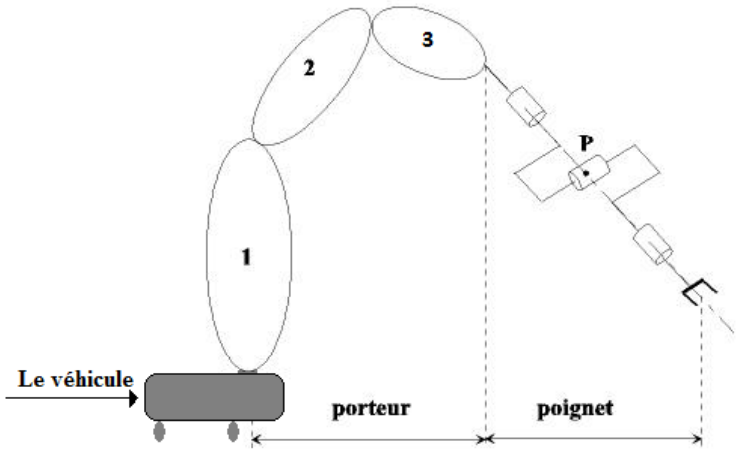


Figure I.10 : Les éléments constituants un robot.

I.8.1. Le Véhicule

Il assure le transport de la structure mécanique vers la zone d'action. Ce sous-ensemble est inexistant sur les robots industriels à poste fixe. Néanmoins, certains robots possèdent un véhicule à six ddl (satellite, sous-marin), deux ddl (robot mobile terrestre) ou un degré de liberté (pont roulant).

I.8.2. Le Porteur

Il est constitué des degrés de liberté 1, 2, 3 à partir du bâti. Il effectue des mouvements de grande amplitude et d'approche rapide. Il présente des segments massifs pour soutenir les parties en amont de l'architecture et permet de fixer la position d'un point de l'extrémité de la structure mécanique articulée dans l'espace. L'extrémité du troisième segment permet de balayer un certain volume par déplacement de ces segments. On rencontre plusieurs types de porteurs. On les classe en 5 familles (voir §9.1).

I.8.3. Le poignet

Le poignet fixé à l'extrémité du porteur, qui est destiné à l'orientation de l'organe terminal (pince, outil, effecteur). Il est caractérisé par des dimensions beaucoup plus petites et une masse plus faible que celle du porteur.

❖ Classification des poignets [khal99]

Il existe plusieurs types de poignets allant de un à trois axes concourants et non concourants. Le tableau I.1 illustre les différents types de poignets.

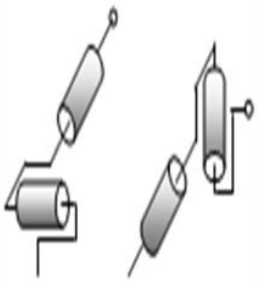
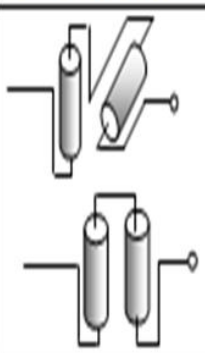
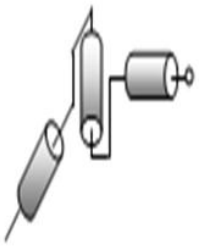
Poignet à un axe	Poignet à 2 axes concourants	Poignet à 2 axes non concourants	Poignet à 3 axes concourants	Poignet à 3 axes non concourants
				

Tableau I.1 : Différents types de poignets.

- La dernière rotation est toujours axiale.
- Le plus souvent, le poignet est à 3 axes rotoïdes concourants.
- Si les axes du poignet sont concourants, la position de centre du poignet est indépendante de l'orientation de l'effecteur.
- Les DDL excédents sont assurés par le poignet. Ce dernier est caractérisé par des dimensions beaucoup plus petites et une plus faible masse**.**

I.8.4. L'organe terminal

Les termes organe terminal, préhenseur, outil ou effecteur sont des synonymes pour nommer le dispositif d'interaction fixé à l'extrémité mobile de la structure mécanique.

Lorsqu'on parle de l'organe terminal, ils nous semblent tout dispositif destiné à manipuler des objets (dispositifs de serrage, dispositifs magnétiques, à dépression,…), ou à les transformer (outils, torche de soudage, pistolet de peinture, …). Le robot peut contenir plusieurs bras chacun d'eux portent un organe terminal.

I.9. CLASSIFICATION DES ROBOTS

Les trois classifications les plus connues sont celles de la *JIRA*, de la *RIA* et de l'*AFRI* (voir liste des abréviations). D'après ces trois classifications, les robots manipulateurs peuvent être classés selon :

- Leur morphologie ;
- Leur type d'asservissement;
- La structure de la chaine cinématique (ouverte ou fermée) ;
- Le nombre de degrés de libertés ;
- Le niveau d'intelligence ;
- Le type de programmation (programmation par langage ou par apprentissage) ;

Nous allons présenter quelque classification :

I.9.1. Classifications selon la morphologie :

C'est la classification des porteurs ou structures de base. Pour les structures ouvertes simple, on dénombre trente-six (36) morphologies possibles à 3ddl. Parmi ces architectures, douze (12) seulement sont mathématiquement différentes et non redondantes. Dans ces 12 architectures, il est possible de les classer en 5 familles : cartésien, cylindrique, sphérique, articulé et scara.

- **Le porteur Cartésien : (PPP)**

Sa structure possède trois degrés de libertés en translations (figure I.11). C'est la plus ancienne : elle est semblable au mouvement d'une machine-outil traditionnelle comme la fraiseuse ou la rectifieuse. Son volume de travail est de type parallélépipédique et est limité par la structure du robot. Le mode de génération de mouvement est plus facile et rapide car c'est une trajectoire rectiligne.

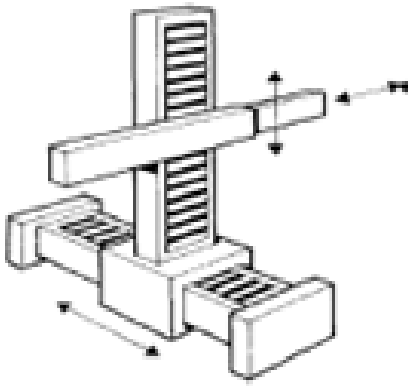


Figure I.11 : Porteur cartésien

Cette structure relativement peu utilisée, sauf dans quelques application comme : soudage par point ; palettisation; assemblage (serrage,…).

- **Le porteur Cylindrique : (RPP ou PRP**)

Il associe une rotation et deux translations, son volume de travail est un cylindre creux (figure I.12). Le mouvement du robot fonctionne avec des à-coups (lorsque la 1^{ere} et la dernière articulation travaillent simultanément) donc le robot perd de la précision.

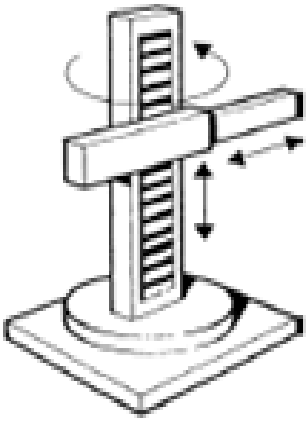


Figure I.12 : Porteur cylindrique

Cette structure présente l'inconvénient d'offrir un volume de travail faible devant un encombrement total important. On la trouve dans l'industrie des machines, des véhicules,…..

- **Le porteur Sphérique : (RRP)**

Le porteur sphérique est constitué d'un bâti rotatif, d'une colonne verticale qui supporte un axe de rotation horizontal autour duquel peut tourner un bras à déplacement télescopique (figure I.13). Le volume de travail a la forme d'une coquille sphérique et n'est pas limité à la structure de base.

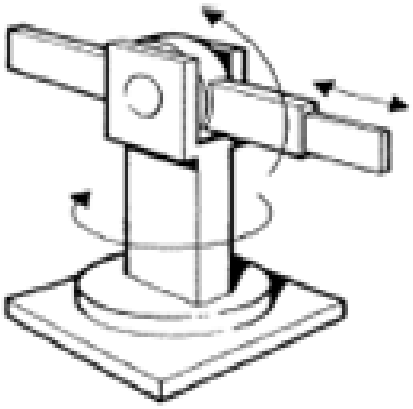


Figure I.13 : Porteur sphérique

Certains porteurs présentent une certaine flexibilité. L'axe horizontal leur permet d'atteindre des objets au sol, ce qui les privilèges dans le transport des charges, la manutention, les machines-outils, le domaine médicale,…

➢ **Le porteur Anthropomorphe-articulé- : (RRR)**

Il est muni de trois rotations, généralement une à axe vertical et deux à axes horizontaux et parallèles, pas forcément que les axes des deux premières articulations soient concourants. Les robots ayant la structure anthropomorphes sont les plus utilisés, du fait, qu'ils peuvent se programmer facilement pour différents types de tâches et disposent d'un volume de travail (sphère pleine) conséquent par rapport aux autres porteurs à dimensions égales. On les trouve dans l'assemblage, palettisation, Soudage (point par point, en arc)…

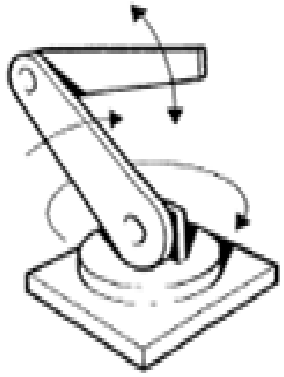


Figure I.14 : Porteur articulé

➢ **Le porteur Torique-scara- : (RPR)**

La structure scara à des axes de rotation parallèles. Elle est l'une des plus utilisées et très répandue, en particulier, pour des tâches de manutention ou d'assemblage très fréquentes dans l'industrie. Généralement, on lui associe un poignet à un degré de liberté en rotation. Le volume de travail dépend du dimensionnement de la structure, soit cylindre plein ou creux.

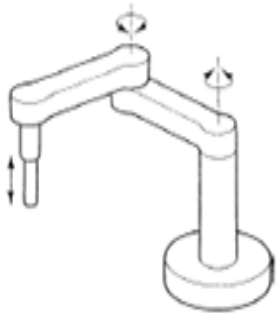


Figure I.15 : Porteur scara

I.9.2. Classification en fonction de l'asservissement :

Les robots peuvent être asservis en **boucle ouverte** ou en **boucle fermée**.

Dans le premier cas, l'exécution d'un mouvement n'a aucun effet sur les commandes qui l'ont provoqué, c-à-d aucune correction n'est faite sur le déplacement et la vitesse de l'organe terminal. La grandeur d'entrée de la boucle est la tension d'alimentation et la position désirée, la grandeur de la sortie est le déplacement de l'organe terminal.

Figure I.16 : Asservissement en boucle ouverte.

Concernant l'asservissement du robot en boucle fermée, on doit disposer d'un comparateur qui mesure l'écart entre la valeur de consigne et la valeur réelle, et délivre un signal correctif à un dispositif de commande qui modifie la valeur de sortie en conséquence.

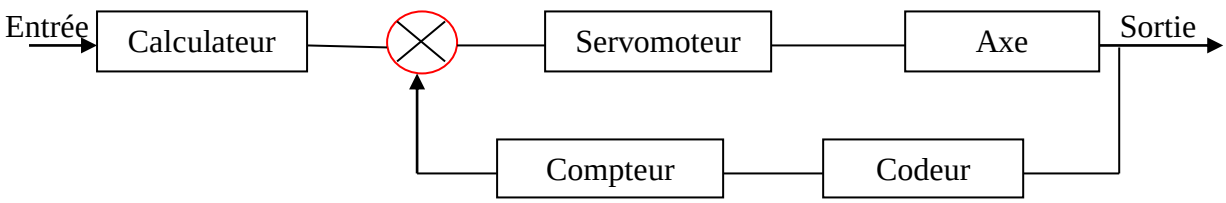


Figure I.17 : Asservissement en boucle fermé.

I.10. CARACTERISTIQUES D'UN ROBOT [Khal99] [Boim01]

Les principales caractéristiques données par la norme ISO 9946 sont :

1. Espace de S.M.A (volume de travail) :

Est défini comme l'espace physique engendré par un point de l'organe terminal lorsque la configuration du robot évolue. Il s'exprime en unités volumiques, mais la forme de son enveloppe (qui peut-être compliquée puisque formée par la combinaison des mouvements de plusieurs articulations) est aussi importante.

On définit aussi deux types d'espace relatif au robot :

a) Espace articulaire :

C'est celui dans lequel est représentée la situation de tous ses corps. On utilise des variables articulaires pour chaque articulation.

Dans une structure de type chaîne ouverte (simple ou arborescente) les variables articulaires sont indépendantes.

b) Espace opérationnel :

L'espace opérationnel est celui dans lequel est représentée la situation de l'effecteur, ou il peut être défini par le nombre de paramètres indépendants nécessaires pour décrire la situation de l'organe terminal dans l'espace.

2. Différentes charges :

- Charge maximale : c'est la charge que peut porter le robot sans dégrader la répétabilité et les performances dynamiques. Elle est directement dépendante des actionneurs.

- Charge maximale transportable (de quelques kilos à quelques tonnes) : charge à déterminer dans les conditions les plus défavorables (en élongation maximale).

3. Répétabilité

Ce paramètre caractérise la capacité d'un robot à retourner vers un point donné (position, orientation). La répétabilité correspond à l'erreur maximum de positionnement sur un point prédéfini dans le cas de trajectoires répétitives. En général, la répétabilité est de l'ordre de 0,1 mm.

4. Exactitude

L'exactitude est l'erreur de position entre une situation réel, défini par une position et une orientation dans l'espace cartésien, et le point atteint (commandé).

5. Fiabilité

Comme tout autre système, un robot peut subir des pannes. La fiabilité sera définie par un taux de pannes qui peut être exprimé soit par la fraction du temps durant lequel le robot ne remplit pas la fonction qui lui est assignée, soit par un MTBF (Mean Time Between Failures).

6. Vitesse et accélération

C'est évidemment, elles sont les caractéristiques fondamentales pour les robots industriels, puisqu'elles déterminent les temps nécessaires à l'exécution d'une tâche.

- **Vitesse** :
  - les robots sont caractérisés par la vitesse maximale de translation ou de rotation de chaque axe
  - Les constructeurs donnent souvent une vitesse de translation maximale de l'organe terminal.
- **Accélération :**
  - Dépend fortement de l'inertie donc de la position du robot.

7. Architecture du S.M.A :

Le choix est guidé par la tâche à réaliser (quelle est la rigidité de la structure ?).

8. Masse du robot.

9. Coût du robot.

10. Maintenance.

D'autres caractéristiques doivent être prises en compte : techniques (énergie, commande, programmation ...) et commerciales (coût, maintenance...).

I.11. REDONDANCE ET SINGULARITE D'UN MANIPULATEUR [Khal99] [Domb01]

I.11.1. Redondance :

Un robot est redondant lorsque le nombre de DDL est inférieur aux nombres de DDM (**redondance structurelle)**, ou lorsque le nombre de degrés de liberté de l'organe terminal est inférieur au nombre de degrés de libertés de l'espace articulaire (**redondance vis-à-vis de la tâche).** Cette propriété permet d'augmenter le volume du domaine accessible et de préserver les capacités de déplacements de l'organe terminal en présence d'obstacle. Les combinaisons d'articulations dans le champ ouvert simple donne une structure redondante :

- Plus de six articulations.
- Plus de trois articulations rotoïdes d'axes concourants.
- Plus de trois articulations rotoïdes d'axes parallèles (figure I.13.b).
- Plus de trois articulations prismatiques.
- Deux axes d'articulations prismatiques parallèles.
- Deux axes d'articulations rotoïdes confondues (figure I.13.a).

(NDDL) d'organe terminal > (NDDL) tâche

NB : (NDDL) = Nombre De Degré de Liberté.

Exemple d'une redondance vis-à-vis de la tâche :

- Meulage
- Soudage par arc ou par points
- Collage
- Assemblage électronique sur des cartes

I.11.2. Singularité :

Pour tous les robots, qu'ils soient redondants ou non, il se peut que dans certaines configurations dites singulières, le nombre de degrés de liberté de l'organe terminal soit inférieur au nombre de degrés de mobilité. Ce cas se présente par exemple lorsque :

- Deux axes d'articulations prismatiques se retrouvent parallèles.
- Deux axes d'articulations rotoïdes se trouvent confondues.

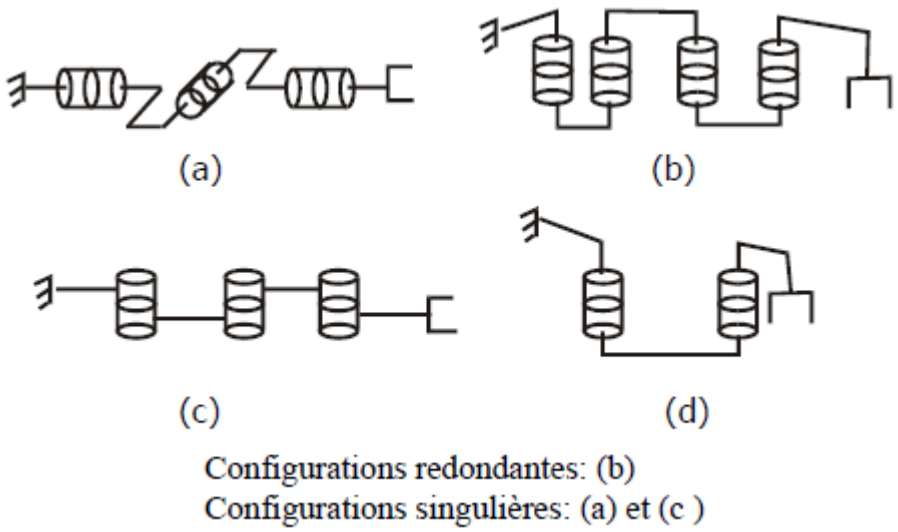


Figure I.18 : Configurations du rdondonce et du singularité.

I.12. CONCLUSION

Dans ce chapitre, nous avons présenté les différentes caractéristiques des robots manipulateurs, leurs différentes morphologies et leurs constituants.

Nous avons aussi rappelé certaines notions de base (degré de liberté, degré de mobilité, singularité, redondance, etc.…), qui sont nécessaires, pour l'étude et la modélisation des robots manipulateurs.

CHAPITRE II

Modélisation géométrique des robots

Chapitre II

MODELISATION GEOMETRIQUE DES ROBOTS

II.1. INTRODUCTION

Pour commander le robot par processeur et calculateur, il est nécessaire de posséder son modèle. Connaissant la configuration du robot, le modèle géométrique direct (M.G.D) permet de déterminer sa situation alors que le modèle géométrique inverse (M.G.I) permet de déterminer sa configuration à partir de sa situation.

La modélisation d'un robot manipulateur fait appel à des notions mathématiques précises. Le but de ce chapitre est de fournir un ensemble de définitions mathématiques précises pour l'étude des mécanismes poly-articulés. Ainsi, on se propose par la suite d'étudier la modélisation géométrique directe et inverse.

II.2. COORDONNES HOMOGENES [Khal99][Boim01][Bors11][Tech03]

➢ Les coordonnées homogènes d'un point P sont représentées par un vecteur de dimension 4x1 :

$$P = \begin{bmatrix} w.p_x \\ w.p_y \\ w.p_z \\ w \end{bmatrix} \quad \text{(II.1)}$$

Avec **w** est un facteur d'échelle, égal à **1** en robotique.

La 4ème coordonnée est introduite pour permettre d'effectuer le produit des matrices 4x4 avec ce vecteur. De ce fait, on tient compte des translations dans les matrices de transformation.

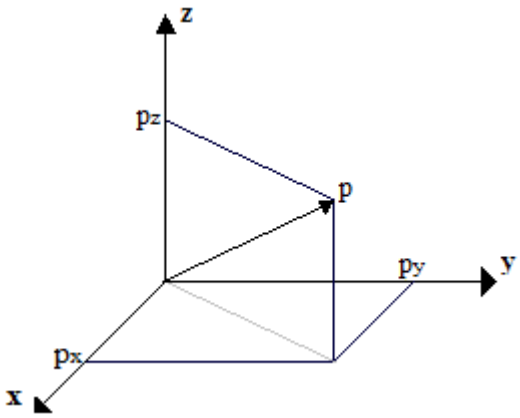


Figure II.1 : Représentation d'un point.

➢ Les coordonnées cartésiennes d'un vecteur unitaire U sont : U_X, U_Y, U_Z, par compte la représentation homogène se fait par quatre composantes, mais la quatrième est toujours **nulle :**

$$U = \begin{bmatrix} u_x \\ u_y \\ u_z \\ 0 \end{bmatrix} \tag{II.2}$$

➢ On donne l'équation du plan $\propto . x + \beta . y + \gamma . z + \delta = 0$. Elle est représentée par un vecteur ligne Q telle que :

$$Q = [\alpha \quad \beta \quad \gamma \quad \delta] \tag{II.3}$$

Pour tout point **P** appartenant au plan **Q**, le produit matriciel **Q.P** est **nul.**

$$Q.P = [\alpha\ \beta\ \gamma\ \delta].\begin{bmatrix} Px \\ Py \\ Pz \\ 1 \end{bmatrix} = 0 \tag{II.4}$$

Soit $\vec{n}(\alpha, \beta, \gamma)$ le vecteur unitaire normal au plan.

II.3. TRANSFORMATIONS HOMOGENES [Khal99][Boim01][Bors11][Tech03]

II.3.1. Transformation des repères :

La transformation rigide ou homogène résulte en général de la combinaison d'une translation P du repère $\mathbf{R_i}$ puis de la rotation de ce repère vers le repère $\mathbf{R_j}$.

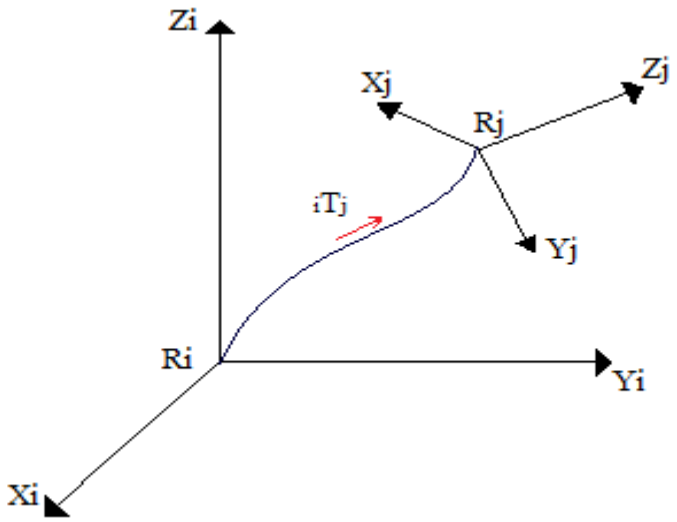


Figure II.2 : Transformation entre 2 repères.

Cette transformation est définie par la matrice ${}^{i}_{j}T$ appelée matrice de transformation homogènes, de dimension (4x4), telle que :

$$ {}^{i}_{j}T = \begin{bmatrix} Sx & Nx & Ax & Px \\ Sy & Ny & Ay & Py \\ Sz & Nz & Az & Pz \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \text{(II.5)} $$

Elle peut être décomposée en deux matrices de transformation, l'une représente une translation pure, et l'autre une rotation pure.

$$ {}^{i}_{j}T = \begin{bmatrix} A & P \\ 0\ 0\ 0 & 1 \end{bmatrix} = \begin{bmatrix} I_3 & P \\ 0\ 0\ 0 & 1 \end{bmatrix} . \begin{bmatrix} A & 0 \\ 0\ 0\ 0 & 1 \end{bmatrix} \qquad \text{(II.6)} $$

$$ A = \begin{bmatrix} S_X & N_X & A_X \\ S_Y & N_Y & A_Y \\ S_Z & N_Z & A_Z \end{bmatrix} \qquad \text{et} \qquad P = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix} $$

Où les colonnes de la matrice A sont les vecteurs normés du repère $\mathbf{R_j}$ exprimés dans le repère $\mathbf{R_i}$.

On définit $\mathbf{{}^{i}_{j}P = [Px\ \ Py\ \ Pz\ \ 1]^t}$ le vecteur exprimant l'origine du repère $\mathbf{R_j}$ dans le repère $\mathbf{R_i}$.

II.3.2. Matrice de transformation de translation pure :

Soit les composantes a, b, c désignent la translation le long des axes X, Y, et Z respectivement. La matrice de transformation de translation pure est donnée par la forme :

$$ {}^{i}_{j}T = \text{Trans}(a,b,c) = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \text{(II.7)} $$

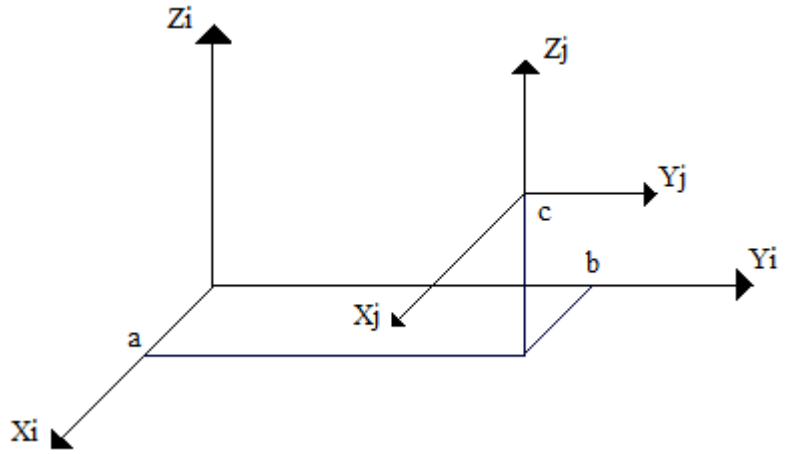


Figure II.3: Transformation de translation pure.

II.3.3. Matrice de transformation de rotation autour des axes principaux

On considère trois cas particuliers de la rotation d'un repère R_i vers un repère R_j,

a) Rotation θ autour de l'axe X :

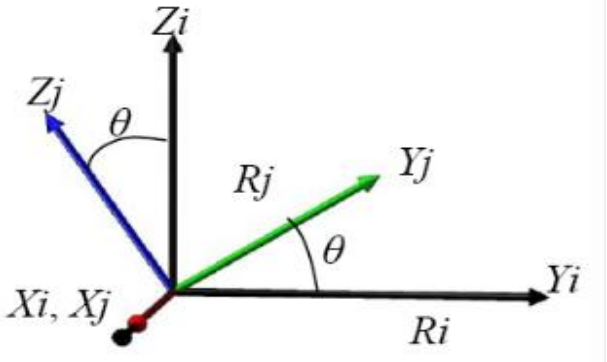


Figure II.4 : Rotation autour de X.

Rot(X, θ) désigne la rotation d'angle θ autour de X. C'est une matrice d'orientation de dimension (3x3). Les éléments de cette matrice sont appelés cosinus directeurs car ils représentent les projections des trois vecteur de la base R_j exprimés dans R_i.

$$ {}^{i}_{j}T = \text{Rot}\,(X,\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\theta & -S\theta & 0 \\ 0 & S\theta & C\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} & \text{rot}(X,\theta) & & 0 \\ & & & 0 \\ & & & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(II.8)} $$

Avec Sθ et Cθ Les **sinus** et **cosinus** de θ respectivement.

b) Rotation θ autour de l'axe Y :

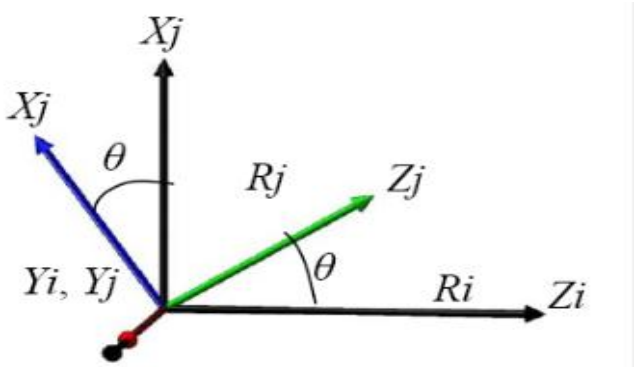


Figure II.5 : Rotation autour de Y.

$$ {}^{i}_{j}T = \text{Rot}(Y,\theta) = \begin{bmatrix} C\theta & 0 & S\theta & 0 \\ 0 & 1 & 0 & 0 \\ -S\theta & 0 & C\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} & \text{rot}(Y,\theta) & & 0 \\ & & & 0 \\ & & & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(II.9)} $$

c) Rotation θ autour de l'axe Z :

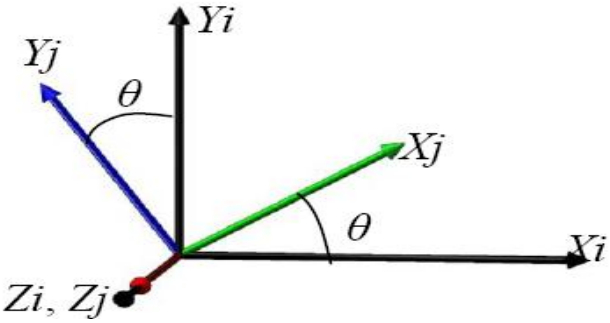


Figure II.6 : Rotation autour de Z.

$$ {}^{i}_{j}T = Rot(Z,\theta) = \begin{bmatrix} C\theta & -S\theta & 0 & 0 \\ S\theta & C\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} & \text{rot}(Z,\theta) & & 0 \\ & & & 0 \\ & & & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(II.10)} $$

II.4. METHODES DE DESCRIPTION DES ORIENTATIONS [Khal99] [Bayl08]

Pour définir la situation de l'organe terminal, on utilise généralement les coordonnés cartésiennes pour le positionner, par contre, son orientation est donnée par les matrices cosinus directeur. Plusieurs descriptions sont utilisées :

-les angles d'Euler pour les robots CINCINNATI-T3 ou les robots PUMA,
-les angles de Roulis –Tangage-Lacet (RTL) pour les robots ACMA.

II.4.1 Angles d'Euler

Les angles d'Euler classiques permettent de décrire l'orientation d'un solide par trois rotations successives Ψ, θ et Φ (Figure II.7). Les angles d'Euler sont définis de la façon suivante :

- Ψ (précession) : autour de Z_0, avec $0 \leq \Psi \leq 360°$;
- θ (nutation) : autour de X_Ψ, avec $0 \leq \theta \leq 180°$;
- φ (rotation propre) : autour de Z_θ , avec $0 \leq \varphi \leq 360°$.

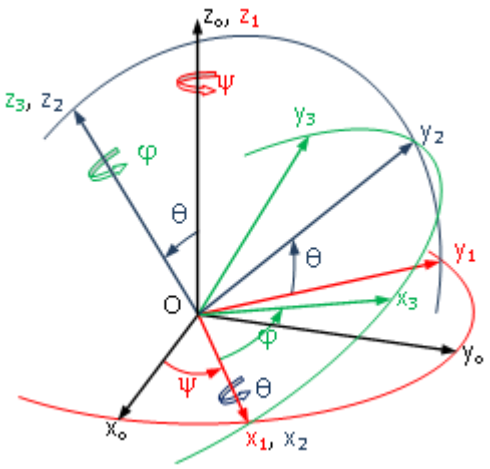


Figure II.7 : Rotations successives par les angles d'Euler.

Chaque nouvelle rotation étant effectuée par rapport à un repère ayant tourné :

$$ {}^{0}_{n}A = \mathrm{Rot}(Z,\psi).\mathrm{Rot}(X,\theta).\mathrm{Rot}(Z,\varphi) \qquad (II.11) $$

$$ = \begin{bmatrix} C\psi C\varphi - S\psi C\theta S\varphi & -C\psi S\varphi - S\psi C\theta C\varphi & S\psi S\theta \\ S\psi C\varphi + C\psi C\theta S\varphi & -S\psi S\varphi + C\psi C\theta C\varphi & -C\psi S\theta \\ S\theta S\varphi & S\theta C\varphi & C\theta \end{bmatrix} $$

❖ **Problème inverse**

La transformation inverse permet d'établir les angles d'Euler à partir des cosinus directeurs. En multipliant à gauche les deux membres de la relation (II.11) par Rot (Z,-Ψ), on obtient :

$$\mathrm{Rot}(Z,-\psi).\,{}^{0}_{n}\mathrm{A} = \mathrm{Rot}(\mathrm{X},\theta).\,\mathrm{Rot}(\mathrm{Z},\varphi) \qquad \text{(II.12)}$$

Par identification entre les membres de cette équation, on trouve :

$$\begin{cases} \Psi=\mathrm{atan2}\ (-Ax,\ Ay), \\ \Psi'=\mathrm{atan2}\ (Ax,\ -Ay)=\Psi+180°, \end{cases} \qquad \text{(II.13)}$$

Ψ étant connu on peut calculer θ et φ :

$$\begin{cases} \theta =\mathrm{atan2}\ (Ax.\sin\Psi - Ay.\cos\Psi,\ Az), \\ \varphi=\mathrm{atan2}\ (-Nx.\cos\Psi - Ny.\sin\Psi,\ Sx.\cos\Psi+Sy.\sin\Psi), \end{cases} \qquad \text{(II.14)}$$

Remarque : *atan2 désigne la fonction arctg. C'est une fonction qu'on définit en robotique et utilisée dans Matlab.*

II.4.2. Angles de cardan ou RTL (Roulis-Tangage-Lacet)

Les rotations successives, conformément à la figure II.8, sont Rot(X, γ), Rot (Y, β) et Rot (Z, α). Les angles γ, β et α sont respectivement désigné sous les noms d'angles roulis, tangage et lacet. Chaque nouvelle rotation étant effectuée par rapport à un axe du repère fixe R_0 :

$${}^{0}_{n}\mathrm{A} = \mathrm{Rot}(\mathrm{Z},\alpha)\mathrm{Rot}(\mathrm{Y},\beta)\mathrm{Rot}(\mathrm{X},\gamma) \qquad \text{(II.15)}$$

$$= \begin{bmatrix} C\alpha C\beta & C\alpha S\beta S\gamma - S\alpha C\gamma & C\alpha S\beta C\gamma + S\alpha S\gamma \\ S\alpha C\beta & S\alpha S\beta S\gamma + C\alpha C\gamma & S\alpha S\beta C\gamma - C\alpha S\gamma \\ -S\beta & C\beta S\gamma & C\beta C\gamma \end{bmatrix}$$

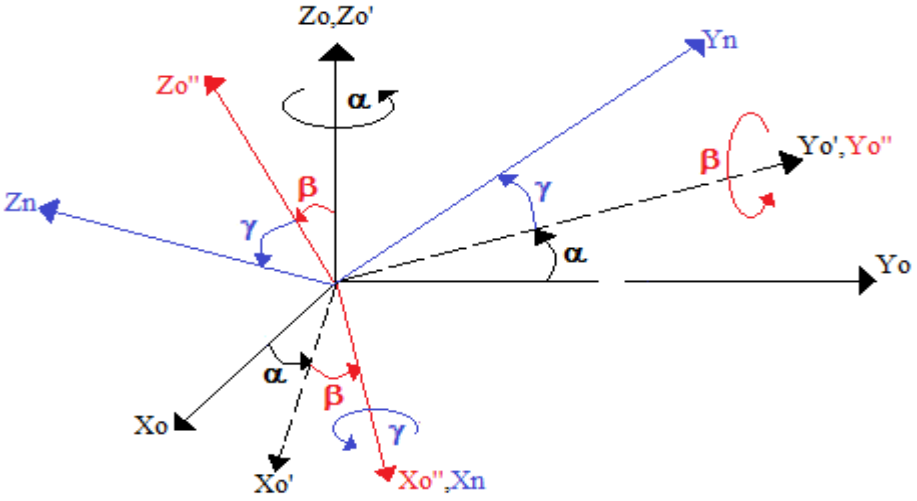


Figure II.8 : Angles de cardan.

❖ **Problème inverse**

On cherche à exprimer les angles de roulis, tangage et lacet à partir des cosinus directeurs. On résonne de la même manière que dans le paragraphe présidant. On obtient les solutions suivantes :

$$\begin{cases} \alpha=\text{atan2 (Sy, Sx)}, \\ \alpha'=\text{atan2 (-Sy,-Sx)}= \alpha+180°, \end{cases} \quad \text{(II.16)}$$

Ainsi, on a : $\begin{cases} \beta=\text{atan2 (-Sz, Sx.cos}\alpha+\text{Sy.sin}\alpha), \\ \gamma=\text{atan2 (Ax.sin}\alpha-\text{Ay.cos}\alpha, \text{-Nx.sin}\alpha+\text{Ny.cos}\alpha), \end{cases}$ (II.17)

II.5. MODELE GEOMETRIQUE DIRECT

Le modèle géométrique direct (MGD) relié entre l'espace articulaire q= [q1 q2 ...q_n], associé aux différentes liaisons du robot, avec l'espace opérationnel X=[x1 x2……x_m], dans lequel est définie la situation de l'organe terminal OT [Khal99], [Lall94], [Hadd11] [Bors11], [Nait06]. Cette relation s'exprime à l'aide de l'équation suivante:

$$X=f(q) \quad \text{(II.18)}$$

Le MGD peut être représenté par la matrice de transformation homogène T_n^0 :

$$T_n^0 = T_1^0(q1).T_2^1(q2) \dots T_n^{n-1}(qn) \quad \text{(II.19)}$$

II.5.1. Paramètres de Denavit-Hertenberg modifies

Le calcule de MGD utilise généralement les matrices de transformation homogène. On associe un repère à chaque solide du robot, en commençant par le socle. La situation de l'organe terminal par rapport au socle correspond au produit des matrices de transformation homogène des différents repères associés au solide du robot. Notons que l'écriture des matrices de transformation homogène n'est pas unique (il existe une infinité de façon de lier un repère à un solide).

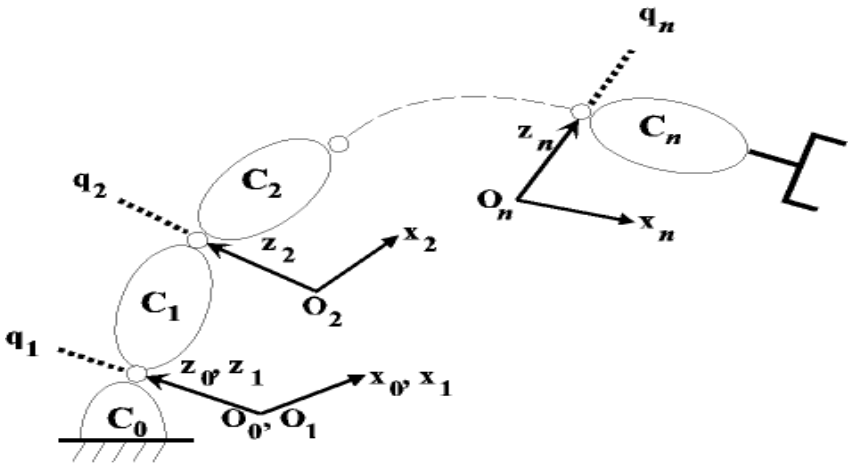


Figure II.9 : Représentation des corps du robot [Boim01].

Les paramètres de **Denavit-Hertenberg modifiés** permettent de disposer d'un paramétrage des liaisons tel que les matrices de passage aient toutes la même forme littérale, ce qui facilite les calculs. Cette convention s'applique lorsque le robot correspond à une chaîne simple ouverte et que ses articulations sont rotoïdes, ou prismatiques (ce qui est le cas en général). Les corps constituant le robot sont supposés parfaitement rigides et connectés par des articulations idéales (pas de jeu mécanique, pas d'élasticité) [khal99], [Boim01], [Ghao09] [Domb01].

Le repère R_j lié au corps C_j est défini de sorte que :

- L'axe Z_j est porté par l'articulation j.
- L'axe X_{j-1} est porté par la perpendiculaire commune aux axes Z_{j-1} et Z_j.
- l'axe X_{j-1} coupe l'axe Z_j.

Si les axes Z_j et Z_{j+1} sont parallèles ou colinéaires, le choix de X_j n'est pas unique. En général des considérations de symétrie ou de simplicité permettent un choix rationnel.

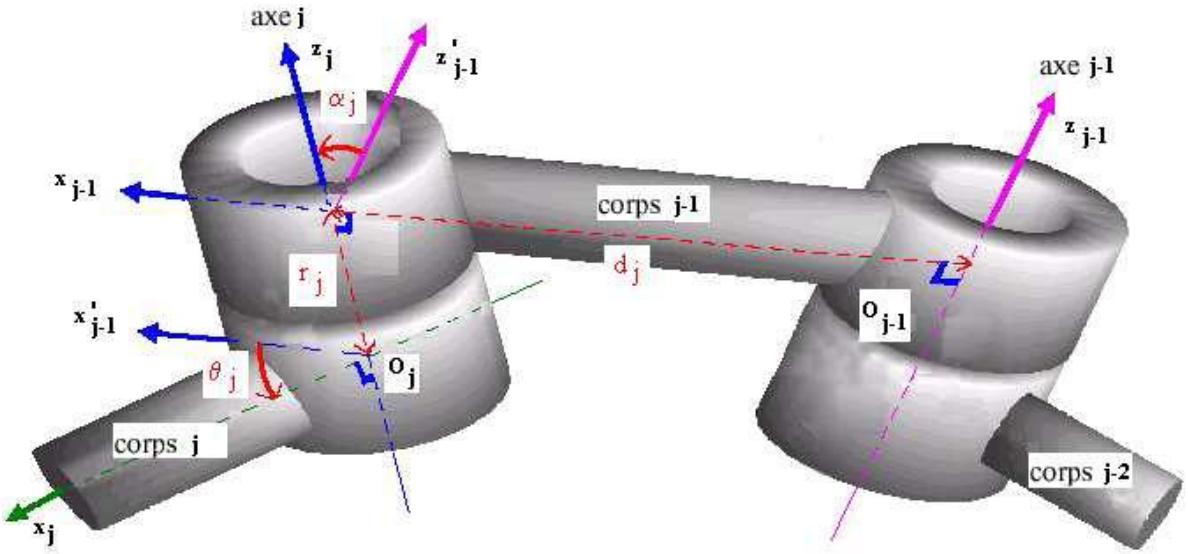


Figure II.10 : Paramètres de D-H modifiés [Boim01].

Le passage du repère R_{j-1} au repère R_j s'exprime en fonction des quatre paramètres géométriques suivants :

- $\boldsymbol{\alpha_j}$: c'est l'angle entre Z_{j-1}et Z_j obtenue en vissant Z_{j-1}vers Z_j autour de X_{j-1} ;
- $\mathbf{d_j}$: c'est la distance entre Z_{j-1}et Z_j mesurée sur l'axe X_{j-1} ;
- $\boldsymbol{\theta_j}$: c'est l'angle entre X_{j-1}et X_j obtenue par rotation de X_{j-1} vers $X_i$autour deZ_j, il est variable pour une articulation rotoïde et constant pour une articulation glissière.
- $\mathbf{r_j}$: distance entre X_{j-1}et X_j le long de l'axe Z_j. Il est variable pour une articulation glissière et constant pour une articulation rotoïde.

La variable articulaire q_j associée à la $j^{iéme}$ articulation est définie soit par θ_j soit par r_j selon le type de cette articulation (**R** ou **P**), ce qui se traduit par la relation :

$$q_j = \sigma_j \theta_j + \sigma_j r_j \tag{II.20}$$

$$\text{Avec : } \sigma_j = \begin{cases} 0 & \text{si rotoide} \\ 1 & \text{si prismatique} \end{cases}$$

La matrice de transformation exprimant le repère R_j dans le repère R_{j-1} est donnée par la relation :

$${}^{j-1}_{j}T = \text{Rot}(X_{j-1}, \alpha_j).\text{Trans}(X_{j-1}, d_j).\text{Rot}(Z_j, \theta_j).\text{Trans}(Z_j, r_j) \tag{II.21}$$

$${}^{j-1}_{j}T = \begin{bmatrix} C\theta_j & -S\theta_j & 0 & d_j \\ S\theta_j C\alpha_j & C\theta_j C\alpha_j & -S\alpha_j & -r_j S\alpha_j \\ S\theta_j S\alpha_j & C\theta_j S\alpha_j & C\alpha_j & r_j C\alpha_j \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{II.22}$$

Remarques [khal99][Boim01][Ghao09][Domb01]

- Pour le choix du repère R_0, le choix le plus simple consiste à prendre Z_0 confondu avec Z_1; ce choix n'est pas obligatoire, il rend les paramètres α_1 et d_1 nuls.
- De même, en définissant l'axe X_n du repère R_n comme étant colinéaire à X_{n-1} lorsque q_n=0,
- Lorsque Z_j est parallèle à Z_{j+1}, on place X_j de telle sorte que r_j ou r_{j+1} soit nul.
- Pour une articulation prismatique, l'axe Z_j est parallèle à l'axe de l'articulation mais la position de l'axe dans l'espace peut être quelconque. On le place de telle sorte que d_j ou d_{j+1} soit nul.
- Cette méthode de description fixe la configuration zéro « géométrique » du robot telle que q=0. Cette configuration ne correspond pas forcément à la configuration « codeur » .On peut passer de l'une à l'autre en procédant au changement de variable suivant : **$q=q_c+q_0$,**
  Où q_0 représente le décalage introduit sur les variables codeurs qc pour obtenir le vecteur des variables articulaires q.
- Lorsqu'une chaîne cinématique comporte un ou plusieurs axes parallèles consécutifs, on peut se ramener à une matrice de transformation faisant apparaître la somme des variables articulaires.

II.6. MODELE GEOMETRIQUE INVERSE

Le modèle géométrique inverse (MGI) permet d'exprimer les variables articulaires *q* du bras manipulateur en fonction des coordonnées opérationnelles *X* exigées pour l'exécution d'une tâche donnée [Rena84], [Khal99], [Hadd11], [Bayl08]. Il est l'ensemble des relations inverses à celles du modèle géométrique direct :

$$q = F^{-1}(X) \tag{II.23}$$

Il n'existe pas une méthode analytique générale qui permet de résoudre le MGI. Cependant, un certain nombre de méthodes, adaptées à des classes de cinématiques particulières, souvent citées en bibliographie, permettent de traiter le problème :

- La méthode de Peiper.
- La méthode de Paul.
- La méthode de Raghavan et Roth.
- Les méthodes numériques**.**

Sachant qu'en fonction du nombre de degré de liberté du robot dans l'espace opérationnel certaines positions ne sont pas possibles.

La résolution de l'équation peut conduire à plusieurs cas :

a) Absence de solution lorsque la situation désirée est en dehors de l'espace de travail du bras manipulateur.

b) Solution en nombre fini lorsque toutes les solutions peuvent être calculées sans ambiguïté (8 solutions dans le cas des bras manipulateurs à six degrés de liberté possédant six liaisons rotoïdes dont trois sont à axes concourants).

c) Infinité de solutions lorsque :
- Le robot est redondant vis-à-vis de la tâche.
- Le robot se trouve dans certaines configurations singulières.

N.B : Dans ce travail, on retient la méthode analytique de Paul et la méthode numérique. Cette dernière fait appel à la notion de la matrice jacobienne.

II.6.1. Calcul du modèle géométrique inverse par la méthode de Paul

Elle traite séparément chaque cas particulier et convient pour la plupart des bras manipulateurs industriels. C'est une méthode heuristique qui n'admet pas une procédure déterministe [khal99] [Domb01].

Soit U_0 la situation désirée telle que :

$$U_0 = \begin{bmatrix} S_x & N_x & A_x & P_x \\ S_y & N_y & A_y & P_y \\ S_z & N_z & A_z & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(II.24)}$$

On cherche à résoudre le système d'équations suivant :

$$U_0 = {}^0T_1(q_1)\ {}^1T_2(q_2)\ldots..\ {}^{n-1}T_n(q_n) \quad \text{(II.25)}$$

La méthode de Paul basée sur la pré-multiplier les deux membres de l'équation (II.25), par les matrices ${}^jT_{j-1}$ pour j variant de 1 à n-1, à titre d'exemple, pour un robot à six degrés de liberté, l'équation (II.25) devient :

$$U_0 = {}^0T_1.\ {}^1T_2.\ {}^2T_3.\ {}^3T_4.\ {}^4T_5.\ {}^5T_6 \quad \text{(II.26)}$$

Tel que le terme de droite a été déjà calculé avec le modèle géométrique directe (MGD).

- En multipliant les deux termes de l'expression (II.26) par ${}^1T_0$, on obtient:

$${}^1T_0\ U_0 = {}^1T_2.\ {}^2T_3.\ {}^3T_4.\ {}^4T_5.\ {}^5T_6 \quad \text{(II.27)}$$

- Par identification terme à terme des deux membres de l'équation (II.27), on se ramène à un système d'une ou de deux équations les plus simples possibles. Fonction de q_1 uniquement, dont la structure appartient à un type particulier parmi une dizaine possible (tableau II.1).
- En multipliant l'expression (II.27) par ${}^2T_1$ pour extraire q_2. En refaisant la même chose pour le calcul des q_j, on obtient le système d'équation :

$$\begin{aligned} {}^2T_1\ U_1 &= {}^2T_3\,{}^3T_4\,{}^4T_5\,{}^5T_6 \\ {}^3T_2\ U_2 &= {}^3T_4\,{}^4T_5\,{}^5T_6 \\ {}^4T_3\ U_3 &= {}^4T_5\,{}^5T_6 \\ {}^5T_4\ U_4 &= {}^5T_6 \end{aligned} \quad \text{(II.28)}$$

Avec $U_{j+1} = {}^{j+1}T_6 = {}^{j+1}T_j U_j$ pour j=0,…4

L'utilisation de la méthode de Paul sur un grand nombre de robots industriels a permis de constater les principaux types d'équations rencontrés (tableau II.1).

Type 1	Xri=Y
Type 2	XSθi +YCθi=Z
Type 3	X1 Sθi +Y1 Cθi=Z1 X2 Sθi +Y2 Cθi=Z2
Type 4	X1 rj Sθi =Y1 X2 rj Cθi =Y2
Type 5	X1 Sθi =Y1 +Z1 rj X2 Cθi =Y2 +Z2 rj
Type 6	W Sθj =X Cθi+ Y Sθi +Z1 W Cθj =X Sθi- Y Cθi +Z2
Type 7	W1Cθj + W2 Sθj = X Cθi +Y Sθi +Z1 W1Sθj - W2 Cθj = X Sθi -Y Cθi +Z2
Type 8	X Cθi +Y C (θi + θj) =Z1 X Cθi +Y S(θi + θj) =Z2

Tableau II.1 : Types d'équations rencontrés avec la méthode de Paul [Khal99]

Avec
r_i : variable articulaire prismatique i.
$S\theta_i$, $C\theta_i$: sinus et cosinus de la variable θi de l'articulation rotoïde i.

II.6.2. Inversion numérique

C'est une adaptation de la méthode classique de Newton-Raphson de résolution d'un système d'équations non-linéaires. Il existe deux méthodes d'inversion numérique par le jacobien inverse et par la linéarisation de la matrice de transformation homogènes. Nous retenons la première méthode que l'on trouve dans les références [Rena84] et [Lall94.] et qui est plus facile à programmer.

Le MGD peut s'écrire sous la forme [Lall94] :

$$F_i(q)=f_i(q_1,\ldots,q_n)-x_i=0 \quad \text{avec } i=1 \text{ à } n \qquad \text{(II.29)}$$

On se donne des $q_i^{(0)}$ initiaux, voisins des solutions q_i cherchée. Par développement en série de Taylor au voisinage des $q_i^{(0)}$, il vient, en se limitant au premier ordre :

$$F_i(q) = F_i\left(q^{(0)}\right) + \sum_{j=1}^{n} \frac{\partial F_i(q^0)}{\partial q_j} \delta q_j^{(0)} \quad \text{(II.30)}$$

Traduisant que $F_i(q)=0$, la relation (II.30) s'écrit sous forme matricielle :

$$J\,(q^{(0)})\,\delta q^{(0)} = -F(q^{(0)}) \quad \text{(II.31)}$$

Ou $J(q^{(0)})$ est le Jacobien -défini dans le chapitre suivant- calculé au point $q^{(0)}$, supposant que $\det J(q^{(0)}) \neq 0$, on peut calculer $\delta q^{(0)}$ par :

$$\delta q^{(0)} = -J^{-1}(q^{(0)})\,F\,(q^{(0)}) \quad \text{(II.32)}$$

Connaissant l'accroissement $\delta q^{(0)}$, on obtient un nouveau point $q^{(1)}$:

$$q^{(1)} = q^{(0)} + \delta q^{(0)} \quad \text{(II.33)}$$

Ce nouveau point est pris comme nouvelle initialisation de (II.31) ou $q^{(1)}$ remplace $q^{(0)}$. On obtiendra $\delta q^{(1)}$ par (II.32) puis $q^{(2)}$ par (II.33). Finalement la formule de récurrence est :

$$q^{(i+1)} = q^{(i)} + J^{-1}(q^{(i)})F(q^{(i)}) \quad \text{(II.34)}$$

Et pour i suffisamment grand égal à N tel que $\left\|q^{(N-1)} - q^{(N)}\right\| \leq \varepsilon$ donné, on obtient q à la précision ε prés.

Les inconvénients de cette méthode est :

- Il faut fixer une estimation initiale $q^{(0)}$ qui ne soit pas trop éloignée de la solution exacte correspondant à la position désirée de l'organe terminal.
- La résolution numérique ne permet la connaissance que d'une seule solution.
- Le temps de calcul est important.

II.7. CONCLUSION

Dans ce chapitre, on résume les outils mathématiques mis en œuvre pour la modélisation des robots. On décrit l'orientation d'un solide dans l'espace avec les méthodes d'Euler et RTL qui sont les plus fréquemment utilisées en robotique.

On effectue les calculs du Modèle Géométrique Direct (MGD) à structure ouverte simple en utilisant la convention de **Denavit-Hertenberg (D-H) modifiée.** Ainsi, l'inversion de ce modèle donne le Modèle Géométrique Inverse (MGI) que l'on peut obtenir par la méthode analytique de Paul ou par la méthode numérique.

Pour la simulation et la commande du mouvement du robot, la modélisation géométrique est insuffisante car il faut avoir des données sur la vitesse et l'accélération des articulations. Ceci montre l'intérêt de la modélisation cinématique.

CHAPITRE III

Modélisation cinématique

Chapitre III

MODELISATION CINEMATIQUE

III.1. INTRODUCTION

Lorsque la contrainte de temps intervient, il est nécessaire d'établir des relations entre les vitesses articulaires et les vitesses linéaires et angulaires de l'organe terminal. Ces relations sont dites modélisation cinématique de premier ordre. Par contre, les relations entre les accélérations opérationnelles et articulaires du bras manipulateur sont appelées modélisation cinématique de deuxième ordre [Hadd11], [Khal99].

Dans ce qui suit, on traite les modèles cinématiques directs et inverses du 1^{er} et $2^{ème}$ ordre.

III.2. MODELE CINEMATIQUE DIRECT

Soit $\boldsymbol{X}$ le vecteur de coordonnées opérationnelles de dimension (m, 1), et $\boldsymbol{q}$ le vecteur des variables articulaires de dimension (n, 1). Le modèle cinématique direct établit la relation entre les vitesses des coordonnées opérationnelles en fonction des vitesses articulaires [Khal99][Domb01]. On a :

$$\dot{\boldsymbol{X}} = \mathrm{J(q)} * \dot{\boldsymbol{q}} \qquad \text{(III.1)}$$

Avec $\dot{\boldsymbol{X}} = \left[\boldsymbol{V_E}, \boldsymbol{w_E}\right]^T$, $\dot{\boldsymbol{q}} = \left[\dot{q}_1 \quad \dot{q}_2 \quad \dots \quad \dot{q}_n\right]^T$

Où J ($\boldsymbol{q}$) est la matrice jacobienne de dimension (m, n) obtenue par dérivation de $\boldsymbol{X}$ par rapport à $\boldsymbol{q}$, soit $\dfrac{\overrightarrow{\partial \mathbf{X}}}{\overrightarrow{\partial \mathbf{q}}}$.

L'intérêt de la matrice jacobienne est multiple [Bors11] :

- elle intervient dans le calcul du modèle différentiel qui donne les variations dX des variables opérationnelles en fonction des variations des variables articulaires dq.
- Elle facilite le calcul des singularités et de dimension de l'espace opérationnel accessible au robot.
- Elle donne une relation liant les efforts exercés par l'organe effecteur et les forces et couples exercés aux articulations.

III.2.1. Calcul de la jacobienne

Il existe plusieurs façons de calcul de la matrice Jacobienne :

A. Par dérivation du modèle géométrique direct

Le calcul de la matrice jacobienne peut se faire en drivant le MGD : $\mathbf{X} = \mathbf{F}(\mathbf{q})$

$$\text{Avec} \quad \boldsymbol{F}(\boldsymbol{q}) = \begin{bmatrix} F_1(\boldsymbol{q}) & F_2(\boldsymbol{q}) & \cdots & F_m(\boldsymbol{q}) \end{bmatrix}$$

$$\Rightarrow \quad \frac{d\boldsymbol{X}}{dt} = \begin{bmatrix} \dfrac{\partial F_1(\boldsymbol{q})}{\partial q_1}\cdot\dfrac{dq_1}{dt} + \dfrac{\partial F_1(\boldsymbol{q})}{\partial q_2}\cdot\dfrac{dq_2}{dt} + \cdots + \dfrac{\partial F_1(\boldsymbol{q})}{\partial q_n}\cdot\dfrac{dq_n}{dt}; & \cdots\cdots \\ \dfrac{\partial F_m(\boldsymbol{q})}{\partial q_1}\cdot\dfrac{dq_1}{dt} + \dfrac{\partial F_m(\boldsymbol{q})}{\partial q_2}\cdot\dfrac{dq_2}{dt} + \cdots + \dfrac{\partial F_m(\boldsymbol{q})}{\partial q_n}\cdot\dfrac{dq_n}{dt} \end{bmatrix}$$

En mettant ce système sous forme matricielle et en l'identifiant avec la relation (III.1), on obtient l'expression suivante :

$$\frac{d\boldsymbol{X}}{dt} = \dot{\boldsymbol{X}} = \begin{bmatrix} \dfrac{\partial F_1(\boldsymbol{q})}{\partial q_1} & \dfrac{\partial F_1(\boldsymbol{q})}{\partial q_2} & \cdots & \dfrac{\partial F_1(\boldsymbol{q})}{\partial q_n} \\ \vdots & \vdots & & \vdots \\ \dfrac{\partial F_m(\boldsymbol{q})}{\partial q_1} & \dfrac{\partial F_m(\boldsymbol{q})}{\partial q_2} & \cdots & \dfrac{\partial F_m(\boldsymbol{q})}{\partial q_n} \end{bmatrix} \cdot \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \vdots \\ \vdots \\ \dot{q}_n \end{bmatrix} \qquad \text{(III.2)}$$

$$\Rightarrow \quad \dot{\boldsymbol{X}} = \left[\widetilde{\boldsymbol{J}}_{\boldsymbol{n}}\right] \cdot [\dot{\boldsymbol{q}}]$$

$$\text{Alors} \qquad J_{ij} = \frac{\partial F_i}{\partial q_j} \qquad \begin{vmatrix} i = 1 \cdots m \\ j = 1 \cdots n \end{vmatrix}$$

Cette méthode est facile à mettre en œuvre pour des robots à deux ou trois degrés de liberté. Cependant, pour des robots ayant plus de trois degrés de liberté, il est plus pratique d'utiliser la méthode de composition des vitesses [khal99] [Bors11].

B. Par les lois de composition des vitesses [Hadd11]

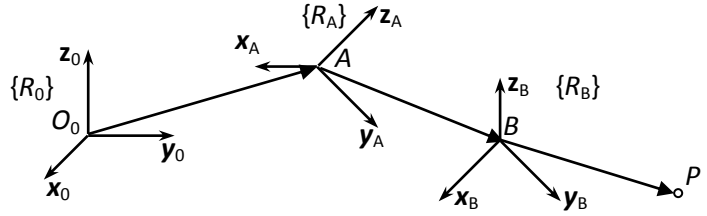
Dans cette méthode, le calcul de la matrice jacobienne utilise la récurrence basée sur le théorème de composition des vitesses. Elle est fondée sur la relation entre les vecteurs des vitesses de translation $\overrightarrow{\boldsymbol{V_n}}$ et de rotation $\overrightarrow{\mathbf{w_n}}$ de l'organe terminal :

$$\dot{\boldsymbol{X}} = \begin{bmatrix} \boldsymbol{V_n} \\ \boldsymbol{w_n} \end{bmatrix} = \left[\tilde{\boldsymbol{J}}_n\right]\left[\dot{\boldsymbol{q}}\ \right] \qquad \text{(III.3)}$$

On exprime en général $\overrightarrow{\boldsymbol{V_n}}$et $\overrightarrow{\mathbf{W_n}}$ soit dans le repère $\{R_n\}$, ou dans le repère $\{R_0\}$. La matrice Jacobienne correspondante est notée ${}^{n}\left[\tilde{\boldsymbol{J}}_n\right]$ ou ${}^{0}\left[\tilde{\boldsymbol{J}}_n\right]$ respectivement.

➢ **Théorème de composition des vitesses**

Etant donnés trois repères $\{R_A\}$, $\{R_B\}$ et $\{R_0\}$ avec ($\{R_0\}$ un repère fixe, et un point P dans l'espace.



La position de P dans $\{R_0\}$ est : $\overrightarrow{O_0P} = \overrightarrow{OA} + \overrightarrow{AB} + \overrightarrow{BP}$.

La vitesse absolue du point P est : $\overrightarrow{V_P} = \frac{d^0\overrightarrow{O_0P}}{dt} = \frac{d^0\overrightarrow{O_0A}}{dt} + \frac{d^0\overrightarrow{AB}}{dt} + \frac{d^0\overrightarrow{BP}}{dt}$.

Soit :

$\overrightarrow{\Omega_{R_A/R_0}}$: La vitesse de rotation instantanée du repère $\{R_A\}$ relativement au repère $\{R_0\}$.

$\overrightarrow{\Omega_{R_B/R_A}}$: La vitesse de rotation instantanée du repère $\{R_B\}$ relativement au repère $\{R_A\}$.

En appliquant la loi de composition des vitesses :

$\overrightarrow{w_A} = \overrightarrow{\Omega_{RA/R0}}$. $\qquad \frac{d^0\overrightarrow{AB}}{dt} = \frac{d^A\overrightarrow{AB}}{dt} + \overrightarrow{w_A} \wedge \overrightarrow{AB}.$

$\overrightarrow{w_B} = \overrightarrow{\Omega_{R_B/R_A}} + \overrightarrow{\Omega_{R_A/R_0}}$. $\qquad \frac{d^0\overrightarrow{AB}}{dt} = \frac{d^B\overrightarrow{AB}}{dt} + \overrightarrow{w_B} \wedge \overrightarrow{AB}.$

Où : $\overrightarrow{w_A}$et $\overrightarrow{w_B}$ sont les vitesses de rotation instantanée des repères $\{R_A\}$ et $\{R_B\}$ relativement au repère $\{R_0\}$.

On s'intéresse uniquement aux origines des repères :

Repère $\{R_0\}$:

$$\overrightarrow{w_0} = \overrightarrow{\Omega_{R_0/R_0}} = \vec{0}. \qquad \overrightarrow{V_0} = \frac{d^0\overrightarrow{O_0O_0}}{dt} = \vec{0}.$$

Repère $\{R_A\}$:

$$\overrightarrow{w_A} = \overrightarrow{\Omega_{R_A/R_0}} \ . \qquad \overrightarrow{V_A} = \frac{d^0\overrightarrow{O_0A}}{dt}.$$

Repère $\{R_B\}$:

$$\begin{cases} \overrightarrow{w_B} = \overrightarrow{\Omega_{R_B/R_A}} + \overrightarrow{\Omega_{R_A/R_0}} = \overrightarrow{w_A} + \overrightarrow{\Omega_{R_B/R_A}} \\ \overrightarrow{V_B} = \frac{d^0\overrightarrow{O_0A}}{dt} + \frac{d^A\overrightarrow{AB}}{dt} + \overrightarrow{w_A} \wedge \overrightarrow{AB} = \overrightarrow{V_A} + \frac{d^A\overrightarrow{AB}}{dt} + \overrightarrow{w_A} \wedge \overrightarrow{AB} \end{cases} \tag{III.4}$$

Pour un bras manipulateur à chaîne ouverte simple, le mouvement relatif du repère $\{R_j\}$ par rapport au repère $\{R_{j-1}\}$ peut être caractérisé, selon les règles de la méthode de description de Denavit-Hertenberg, par le torseur cinématique calculé au point de réduction O_{j-1}, donné par :

$$\left\{\zeta_{C_j}\right\}_{/O_{j-1}} = \begin{Bmatrix} \boldsymbol{\Omega}_{j/j-1} \\ {}^{j-1}\dot{\boldsymbol{P}}_j \end{Bmatrix} = \begin{Bmatrix} \overline{\sigma}_j * \dot{q}_j * \mathbf{z}_j \\ \sigma_j * \dot{q}_j * \mathbf{z}_j \end{Bmatrix} \tag{III.5}$$

Où :

$\boldsymbol{\Omega}_{j/j-1}$ est la vitesse de rotation instantanée du repère $\{R_j\}$ relativement au repère $\{R_{j-1}\}$.

${}^{j-1}\dot{\boldsymbol{P}}_j$ est la vitesse de translation de l'origine O_j du repère $\{R_j\}$ relativement au repère $\{R_{j-1}\}$.

$$q_j = \overline{\sigma}_j \cdot \theta_j + \sigma_j \cdot r_j$$

En appliquant les relations (III.4), on en déduit le torseur cinématique du repère $\{R_j\}$ calculé au point de réduction O_0.

$$\begin{cases} \boldsymbol{w}_j = \boldsymbol{w}_{j-1} + \overline{\sigma}_j * \dot{q}_j * \mathbf{z}_j \\ \boldsymbol{V}_j = \boldsymbol{V}_{j-1} + \boldsymbol{w}_{j-1} \wedge {}^{j-1}\boldsymbol{P}_j + \sigma_j * \dot{q}_j * \mathbf{z}_j \end{cases} \tag{III.6}$$

avec $w_0 = \vec{0}$ et $V_0 = \vec{0}$ (La base du bras manipulateur fixe).

On a donc le torseur cinématique de chaque corps du bras manipulateur :

Pour le corps C_0 : $w_0 = 0$ $\quad V_0 = 0$

Pour le corps C_1 : $\boldsymbol{w}_1 = \boldsymbol{w}_0 + \overline{\sigma}_1.\dot{q}_1.\mathbf{z}_1$ $\quad \boldsymbol{V}_1 = \boldsymbol{V}_0 + \boldsymbol{w}_0 \wedge {}^0\boldsymbol{P}_1 + \sigma_1.\dot{q}_1.\mathbf{z}_1$

$\vdots$

Pour le corps C_j : $\boldsymbol{w}_j = \boldsymbol{w}_{j-1} + \overline{\sigma}_j * \dot{q}_j * \mathbf{z}_j$ $\quad \boldsymbol{V}_j = \boldsymbol{V}_{j-1} + \boldsymbol{w}_{j-1} \wedge {}^{j-1}\boldsymbol{P}_j + \sigma_j * \dot{q}_j * \mathbf{z}_j$

$\vdots$

Pour le corps C_n : $\boldsymbol{w}_n = \boldsymbol{w}_{n-1} + \overline{\sigma}_n * \dot{q}_n * \mathbf{z}_n$ $\quad \boldsymbol{V}_n = \boldsymbol{V}_{n-1} + \boldsymbol{w}_{n-1} \wedge {}^{n-1}\boldsymbol{P}_n + \sigma_n * \dot{q}_n * \mathbf{z}_n$

En fait la somme terme à terme, on obtient :

$$\begin{aligned} \cancel{w_0} &= 0 \\ \cancel{\boldsymbol{w}_1} &= \cancel{\boldsymbol{w}_0} + \overline{\sigma}_1.\dot{q}_1.\mathbf{z}_1 \\ &\vdots \\ \cancel{\boldsymbol{w}_j} &= \cancel{\boldsymbol{w}_{j-1}} + \overline{\sigma}_j.\dot{q}_j.\mathbf{z}_j \\ &\vdots \\ \boldsymbol{w}_n &= \cancel{\boldsymbol{w}_{n-1}} + \overline{\sigma}_n.\dot{q}_n.\mathbf{z}_n \end{aligned}$$

$$\boldsymbol{w}_n = \sum_{j=1}^{n} \overline{\sigma}_j.\dot{q}_j.\mathbf{z}_j \qquad \text{(III.7)}$$

$$\begin{aligned} \cancel{V_0} &= 0 \\ \cancel{\boldsymbol{V}_1} &= \cancel{\boldsymbol{V}_0} + \boldsymbol{w}_0 \wedge {}^0\boldsymbol{P}_1 + \sigma_1.\dot{q}_1.\mathbf{z}_1 \\ &\vdots \\ \cancel{\boldsymbol{V}_j} &= \cancel{\boldsymbol{V}_{j-1}} + \boldsymbol{w}_{j-1} \wedge {}^{j-1}\boldsymbol{P}_j + \sigma_j.\dot{q}_j.\mathbf{z}_j \\ &\vdots \\ \boldsymbol{V}_n &= \cancel{\boldsymbol{V}_{n-1}} + \boldsymbol{w}_{n-1} \wedge {}^{n-1}\boldsymbol{P}_n + \sigma_n.\dot{q}_n.\mathbf{z}_n \end{aligned}$$

$$\boldsymbol{V}_n = \sum_{j=0}^{n-1} \boldsymbol{w}_j \wedge {}^j\boldsymbol{P}_{j+1} + \sum_{j=1}^{n} \sigma_j.\dot{q}_j.\mathbf{z}_j$$

Réarrangement de l'expression de $\mathbf{V}_n$ de telle sorte à isoler les $\dot{q}_j$:

$$\boldsymbol{V}_n = \sum_{j=0}^{n-1} \boldsymbol{w}_j \wedge {}^j\boldsymbol{P}_{j+1} + \sum_{j=1}^{n} \sigma_j.\dot{q}_j.\mathbf{z}_j$$

$$= \overline{\sigma}_1.\dot{q}_1.\mathbf{z}_1 \wedge {}^1\boldsymbol{P}_2 + (\overline{\sigma}_1.\dot{q}_1.\mathbf{z}_1 + \overline{\sigma}_2.\dot{q}_2.\mathbf{z}_2) \wedge {}^2\boldsymbol{P}_3 + \cdots + (\overline{\sigma}_1.\dot{q}_1.\mathbf{z}_1 + \cdots + \overline{\sigma}_{n-1}.\dot{q}_{n-1}.\mathbf{z}_{n-1}) \wedge {}^{n-1}\boldsymbol{P}_n + \sum_{j=1}^{n} \sigma_j.\dot{q}_j.\mathbf{z}_j$$

$$= \overline{\sigma}_1.\dot{q}_1.\mathbf{z}_1 \wedge ({}^1\boldsymbol{P}_2 + {}^2\boldsymbol{P}_3 + \cdots + {}^{n-1}\boldsymbol{P}_n) + \cdots + \overline{\sigma}_{n-1}.\dot{q}_{n-1}.\mathbf{z}_{n-1} \wedge {}^{n-1}\boldsymbol{P}_n + \sum_{j=1}^{n} \sigma_j.\dot{q}_j.\mathbf{z}_j$$

Soit ${}^i\boldsymbol{P}_n = {}^i\boldsymbol{P}_{i+1} + {}^{i+1}\boldsymbol{P}_{i+2} + \cdots + {}^{n-1}\boldsymbol{P}_n = \overrightarrow{O_i O_n}$

$$\Rightarrow \boldsymbol{V}_n = \sum_{j=1}^{n-1} \overline{\sigma}_j.\dot{q}_j.\mathbf{z}_j \wedge {}^j\boldsymbol{P}_n + \sum_{j=1}^{n} \sigma_j.\dot{q}_j.\mathbf{z}_j \ ;$$

${}^n\mathrm{P}_n = \overrightarrow{O_n O_n} = 0 \quad \Rightarrow \quad \boldsymbol{V}_n = \sum_{j=1}^{n} \overline{\sigma}_j.\dot{q}_j.\mathbf{z}_j \wedge {}^j\boldsymbol{P}_n + \sigma_j.\dot{q}_j.\mathbf{z}_j \qquad \text{(III.8)}$

Finalement : $\begin{pmatrix} \boldsymbol{V}_n \\ \boldsymbol{w}_n \end{pmatrix} = \begin{pmatrix} \sum_{j=1}^{n} \overline{\sigma}_j.\dot{q}_j.\mathbf{z}_j \wedge {}^j\boldsymbol{P}_n + \sigma_j.\dot{q}_j.\mathbf{z}_j \\ \sum_{j=1}^{n} \overline{\sigma}_j.\dot{q}_j.\mathbf{z}_j \end{pmatrix} \Rightarrow \begin{pmatrix} \boldsymbol{V}_n \\ \boldsymbol{w}_n \end{pmatrix} = \sum_{j=1}^{n} \begin{pmatrix} \overline{\sigma}_j.\mathbf{z}_j \wedge {}^j\boldsymbol{P}_n + \sigma_j.\mathbf{z}_j \\ \overline{\sigma}_j.\mathbf{z}_j \end{pmatrix} .\dot{q}_j$

L'en identifient avec la relation : $\begin{bmatrix} \boldsymbol{V_n} \\ \boldsymbol{w_n} \end{bmatrix} = \left[\tilde{\boldsymbol{J}}_n\right]\left[\dot{\boldsymbol{q}}\ \right]$

$$\Rightarrow \qquad \left[\tilde{\boldsymbol{J}}_n\right] = \begin{bmatrix} \bar{\sigma}_1.\mathbf{z}_1 \wedge {}^1\boldsymbol{P}_n + \sigma_1.\mathbf{z}_1 & \bar{\sigma}_2.\mathbf{z}_2 \wedge {}^2\boldsymbol{P}_n + \sigma_2.\mathbf{z}_2 & \cdots & \bar{\sigma}_n.\mathbf{z}_n \wedge {}^n\boldsymbol{P}_n + \sigma_n.\mathbf{z}_n \\ \bar{\sigma}_1.\mathbf{z}_1 & \bar{\sigma}_2.\mathbf{z}_2 & \cdots & \bar{\sigma}_n.\mathbf{z}_n \end{bmatrix}$$

$\left[\tilde{\boldsymbol{J}}_n\right]$: est appelé Jacobien vectoriel.

Projetons par exemple $\left[\tilde{\boldsymbol{J}}_n\right]$ dans le repère {R $_n$}, la j^{éme} colonne de ${}^n\left[\tilde{\boldsymbol{J}}_n\right]$ devient :

$$ {}^n\boldsymbol{J}_j = \begin{bmatrix} \bar{\sigma}_j.\ {}^n\tilde{\boldsymbol{R}}_j \left[\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \wedge {}^j\boldsymbol{P}_n\right] + \sigma_j.\ {}^n\tilde{\boldsymbol{R}}_j.\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \\ \bar{\sigma}_j.\ {}^n\tilde{\boldsymbol{R}}_j.\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \end{bmatrix} \qquad \text{(III.9)}$$

La matrice Jacobienne scalaire ${}^n\left[\tilde{\boldsymbol{J}}_n\right]$ s'écrit donc sous la forme :

$${}^n\left[\tilde{\boldsymbol{J}}_n\right] = \begin{bmatrix} {}^n\tilde{\boldsymbol{R}}_1\left[\bar{\sigma}_1.\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \wedge {}^1\boldsymbol{P}_n + \sigma_1.\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}\right] & {}^n\tilde{\boldsymbol{R}}_2\left[\bar{\sigma}_2.\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \wedge {}^2\boldsymbol{P}_n + \sigma_2.\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}\right] & \cdots & {}^n\tilde{\boldsymbol{R}}_n\left[\bar{\sigma}_n.\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \wedge {}^n\boldsymbol{P}_n + \sigma_n.\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}\right] \\ \bar{\sigma}_1.\ {}^n\tilde{\boldsymbol{R}}_1.\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} & \bar{\sigma}_2.\ {}^n\tilde{\boldsymbol{R}}_2.\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} & \cdots & {}^n\tilde{\boldsymbol{R}}_n\left[\bar{\sigma}_n.\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \wedge {}^n\boldsymbol{P}_n + \sigma_n.\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}\right] \end{bmatrix}$$

Pour trouver une projection dans un autre repère, le repère {R_k} par exemple, il suffit d'utiliser la relation suivante :

$${}^k\left[\tilde{\boldsymbol{J}}_n\right] = \begin{bmatrix} {}^k\tilde{\boldsymbol{R}}_n & 0 \\ 0 & {}^k\tilde{\boldsymbol{R}}_n \end{bmatrix}.{}^n\left[\tilde{\boldsymbol{J}}_n\right]$$

***Remarque** :*

On peut calculer la vitesse de n'importe quel point P de l'organe terminal par la même relation, il suffit de remplacer O_n par P.

III.2.2. Expression de la vitesse d'un point P du corps C_i

En appliquant le théorème de composition des vitesses, on peut écrire :

$$\mathrm{V}^{0}_{P\in C_i} = \mathrm{V}^{i}_{P\in C_i} + \mathrm{V}_i + \mathrm{w}_i \wedge \overrightarrow{O_iP}$$

Le corps C_i du bras manipulateur est un corps rigide $\Rightarrow \mathrm{V}^{i}_{P\in C_i} = 0$

$$\Rightarrow \boldsymbol{V}^{0}_{P\in C_i} = \boldsymbol{V}_i + \boldsymbol{w}_i \wedge \overrightarrow{O_iP} \qquad \text{Avec} \qquad \begin{cases} \boldsymbol{V_i} = \sum_{j=1}^{i} \overline{\sigma}_j . \dot{q}_j . \mathbf{z}_j \wedge {}^{j}\boldsymbol{P_i} + \sigma_j . \dot{q}_j . \mathbf{z}_j \\ \boldsymbol{w_i} = \sum_{j=1}^{i} \overline{\sigma}_j . \dot{q}_j . \mathbf{z}_j \end{cases}$$

$$\Rightarrow \boldsymbol{V}^{0}_{\boldsymbol{P\in C_i}} = \sum_{j=1}^{i} \overline{\sigma}_j . \dot{q}_j . \mathbf{z}_j \wedge \overrightarrow{O_jP} + \sigma_j . \dot{q}_j . \mathbf{z}_j \qquad \text{(III.10)}$$

III.2.3. Etude des singularités

L'analyse des configurations singulières peut être réalisée en s'appuyant sur le MCD. Si on note : r = rang [J ($\boldsymbol{q}$)].

Avec $r \leq$ min (DDM, DDN)

- Si DDM = DDN, on a : $r \leq$DDM
- Si DDM $\leq$ DDN (cas d'un robot redondant), on a aussi $r \leq$ DDM.

Lorsque r <DDM, il devient impossible d'engendrer une vitesse et donc un mouvement le long ou autour de certaines directions. Le robot possède une configuration singulière d'ordre égale à DDM-r.

III.3. MODELE CINEMATIQUE INVERSE

Le modèle cinématique inverse (**MCI**) présente un grand intérêt en phase d'exploitation surtout quand le bras manipulateur travaille en cours de déplacement avec des contraintes de vitesses, de l'effecteur relativement à la pièce (opérations de soudage, de découpage,…..). L'autre intérêt est qu'il se substitue au modèle géométrique inverse et permet de calculer les incréments de déplacements articulaires à réaliser sur les différentes articulations, sur la base de la configuration présente, pour matérialiser un déplacement élémentaire dans l'espace opérationnel [Hadd11].

Le modèle cinématique inverse du premier ordre permet de déterminer, dans le voisinage d'une configuration $\boldsymbol{q}$, les vitesses articulaires $\dot{\boldsymbol{q}}$ qui assurent une vitesse opérationnelle $\dot{\mathrm{X}}$ imposée.

Dans le cas régulier et si la matrice jacobienne est carrée, on a :

$$\dot{\boldsymbol{q}} = \boldsymbol{J}^{-1}(\boldsymbol{q}) * \dot{\boldsymbol{X}} \qquad \text{(III.11)}$$

Lorsque la matrice jacobienne n'est pas carrée, on utilise la pseudo inverse pour calculer le vecteur $\dot{\boldsymbol{q}}$. On a alors :

$$\dot{\boldsymbol{q}} = \tilde{\boldsymbol{J}}^{+}(\boldsymbol{q}).\dot{\boldsymbol{X}} \qquad \text{(III.12)}$$

L'idée de cette méthode est que, même si le vecteur des vitesses opérationnelles $\dot{\boldsymbol{X}}$ n'est pas compatible au voisinage des singularités (système (III.11) peut ne pas admettre de solution), la pseudo- inverse permet quand même de calculer les vitesses articulaires $\dot{\boldsymbol{q}}$ qui minimisent la norme $\left\| \dot{\boldsymbol{X}} - \tilde{\boldsymbol{J}}(\boldsymbol{q}).\dot{\boldsymbol{q}} \right\|^2$. Ces vitesses, bien que n'étant pas exactement les mêmes que les vitesses $\dot{\boldsymbol{X}}$ désirées, peuvent cependant être acceptables jusqu'à ce que la singularité soit franchie.

Il existe plusieurs techniques de mise en œuvre du MCI :

- Méthode s'appuyant sur une solution analytique, qui procède par dérivation des modèles géométriques inverses. Cette méthode exige que le robot soit résoluble, c'est à dire qu'il soit possible de calculer toutes les configurations permettant d'atteindre une situation donnée. Ainsi, elle nécessite de traiter séparément tous les cas singuliers. Elle conduit à un temps de calcul réduit.

- Méthode numérique plus générale consistent à inverser le modèle cinématique direct, fondée sur l'utilisation du pseudo inverse (équation (III.12)) qui permet de traiter tous les cas (réguliers, singuliers et redondants). Elle nécessite un temps de calcul plus important que la méthode analytique. Cette méthode est retenue en raison de son caractère général.

III.3.1. Pseudo- inverse de la matrice jacobienne

On appelle pseudo-inverse, ou inverse de Moore-Penrose de la matrice jacobienne $\tilde{\boldsymbol{J}}(\boldsymbol{q})$, la matrice généralisée $\tilde{\boldsymbol{J}}^{+}(\boldsymbol{q})$ de dimension (m*n), telle que [Rena84] :

$$\begin{cases} \left(\tilde{\boldsymbol{J}} * \tilde{\boldsymbol{J}}^{+}\right)^T = \tilde{\boldsymbol{J}} * \tilde{\boldsymbol{J}}^{+} \\ \left(\tilde{\boldsymbol{J}}^{+} * \tilde{\boldsymbol{J}}\right)^T = \tilde{\boldsymbol{J}}^{+} * \tilde{\boldsymbol{J}} \\ \tilde{\boldsymbol{J}} * \tilde{\boldsymbol{J}}^{+} * \tilde{\boldsymbol{J}} = \tilde{\boldsymbol{J}} \\ \tilde{\boldsymbol{J}}^{+} * \tilde{\boldsymbol{J}} * \tilde{\boldsymbol{J}}^{+} = \boldsymbol{J}^{+} \end{cases}$$

Il existe des algorithmes spécifiques qui permettent d'obtenir la matrice pseudo-inverse, en particulier l'algorithme de Greville.

III.3.2. Algorithme de Greville [Hadd11]:

Il s'agit d'un algorithme itératif, fondé sur les propretés du pseudo-inverse d'une matrice partitionnée.

Appelons J_k la k^ième colonne de $\tilde{J}(q)$ et $\left[\tilde{J}_{k-1}\right]$ la sous matrice constituée par (k-1) premières colonnes de $\tilde{J}(q)$ .on peut écrire donc :

$$\left[\tilde{J}_k\right] = \left[\left[\tilde{J}_{k-1}\right] : J_k\right]$$

La pseudo- inverse de $\left[\tilde{J}_k\right]$ notée $\left[\tilde{J}_k\right]^+$ s'exprime par : $\left[\tilde{J}_k\right]^+ = \begin{pmatrix} \left[\tilde{J}_{k-1}\right]^+ - d_k * b_k^T \\ b_k^T \end{pmatrix}$

Avec $d_k = \left[\tilde{J}_{k-1}\right]^+ * J_k$; $c_k = J_k - \left[\tilde{J}_{k-1}\right] * d_k$

Si $c_k \neq 0$ $b_k^T = c_k^+ = \dfrac{c_k^T}{c_k^T * c_k}$.

Si $c_k = 0$ $b_k^T = \dfrac{d_k * \left[\tilde{J}_{k-1}\right]^+}{1 + d_k^T * d_k}$.

L'initialisation des itérations dépend de la 1^ère colonne de $\tilde{J}(q)$:

Si $J_1 \neq 0$ *alors* $\left[\tilde{J}_1\right]^+ = \dfrac{J_1^T}{J_1^T * J_1}$

Si $J_1 = 0$ *alors* $\left[\tilde{J}_1\right]^+ = [0]^T$

Cet algorithme est indépendant du rang et des dimensions de la matrice jacobienne $\tilde{J}(q)$ Il permet de plus, la détection en ligne des singularités de $\tilde{J}(q)$ à chaque itération par le calcul de la trace de $\tilde{J}^+ * \tilde{J}$ qui est égal au rang de la matrice $\tilde{J}$ et de la matrice $\tilde{J}^+$.

$$r_{max} = Trace\left[\tilde{J}^+ * \tilde{J}\right] \qquad \text{(III.13)}$$

r_{max} : rang maximal de la matrice jacobienne, appelé aussi nombre de degrés de liberté opérationnel du bras manipulateur.

III.4. MODELE CINEMATIQUE DIRECT DU SECOND ORDRE

Le modèle cinématique direct du second ordre permet de calculer l'accélération des coordonnées opérationnelles $\ddot{\boldsymbol{X}}$ de l'organe terminal en fonction des positions $\boldsymbol{q}$, vitesses $\dot{\boldsymbol{q}}$ et accélérations articulaires $\ddot{\boldsymbol{q}}$. Dans le cas d'un mécanisme à chaîne ouverte simple, on obtient par dérivation du l'équation (III.1) par rapport au temps [Domb01][Hadd11]:

$$\ddot{\boldsymbol{X}} = \boldsymbol{J}(\boldsymbol{q}) * \ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) * \dot{\boldsymbol{q}} \tag{III.14}$$

Avec $\ddot{\boldsymbol{X}} = \left[\dot{\boldsymbol{V}}_{\boldsymbol{E}}, \dot{\boldsymbol{w}}_{\boldsymbol{E}}\right]^T \quad , \quad \ddot{\boldsymbol{q}} = \left[\ddot{q}_1 \quad \ddot{q}_2 \quad \dots \quad \ddot{q}_n\right]^T$

$$\dot{\boldsymbol{J}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \frac{d}{dt}\boldsymbol{J}(\boldsymbol{q})$$

Les méthodes qui permettent de construire le modèle cinématique direct du second ordre peuvent être regroupées en deux classes :

1. celles qui procèdent par dérivation du modèle cinématique direct.

2. celles, plus générales, qui consistent à mettre en œuvre une formulation récurrente sur la base des lois de composition des vitesses et des accélérations.

Les formules de composition des vitesses et des accélérations, pour $j = 1 \dots n$

$$\begin{cases} \boldsymbol{w}_j = \boldsymbol{w}_{j-1} + \overline{\sigma}_j * \dot{q}_j * \mathbf{z}_j \\ \dot{\boldsymbol{w}}_j = \dot{\boldsymbol{w}}_{j-1} + \overline{\sigma}_j * \left(\ddot{q}_j * \mathbf{z}_j + \dot{q}_j * \boldsymbol{w}_{j-1} \Lambda \mathbf{z}_j\right) \\ \dot{\boldsymbol{V}}_j = \dot{\boldsymbol{V}}_{j-1} + \dot{\boldsymbol{w}}_{j-1} \Lambda \boldsymbol{P}_j^{j-1} + \boldsymbol{w}_{j-1} \Lambda (\boldsymbol{w}_{j-1} \Lambda \boldsymbol{P}_j^{j-1}) + \overline{\sigma}_j \left(\ddot{q}_j * \mathrm{z}_j + 2 * \dot{q}_j * \mathrm{w}_{j-1} \Lambda \mathrm{z}_j\right) \end{cases} \tag{III.15}$$

- On initialise la récurrence par :

  $\boldsymbol{w}_0 = \mathbf{0}$, $\dot{\boldsymbol{w}}_0 = \mathbf{0}$, $\dot{\boldsymbol{V}}_0 = \mathbf{0}$ (La base du bras manipulateur fixe)

III.5. MODELE CINEMATIQUE INVERSE DU SECOND ORDRE

En phase d'exploitation, on utilise le modèle cinématique inverse du second ordre quand le bras manipulateur travaille en cours de trajectoire avec des contraintes de vitesse et accélération, en déplacement ou en orientation, de l'effecteur relativement à la pièce. On utilise aussi ce modèle pour minimiser les couples des actionneurs dans le cas d'un bras manipulateur redondant. En phase de la conception, on utilise ce modèle pour déterminer les

valeurs limites des accélérations articulaires $(\ddot{\boldsymbol{q}})_{Max}$ en fonction des données du cahier des charges prévisionnel.

Le modèle cinématique inverse du second ordre, traite le problème inverse du précédent. Il permet de calculer le vecteur des accélérations articulaires $\ddot{\boldsymbol{q}}$ en fonction des accélérations opérationnelles $\ddot{\boldsymbol{X}}$, pour des coordonnées et des vitesses généralisées désirées. Il s'écrit :

$$\ddot{\boldsymbol{q}} = \boldsymbol{J}^{-1}(\boldsymbol{q}) * \left[\ddot{\boldsymbol{X}} - \dot{\boldsymbol{J}}(\boldsymbol{q},\dot{\boldsymbol{q}}) * \dot{\boldsymbol{q}}\right] \qquad \text{(III.16)}$$

Pour calculer le vecteur $\dot{\boldsymbol{J}}(\boldsymbol{q},\dot{\boldsymbol{q}}) * \dot{\boldsymbol{q}}$ il suffit de prendre $\ddot{\boldsymbol{q}} = \boldsymbol{0}$ dans la procédure d'évaluation des accélérations linéaire et angulaire du repère effecteur [Hadd11].

III.6. CONCLUSION

Nous avons présenté dans ce chapitre la modélisation cinématique direct du premier ordre et deuxième ordre, ainsi, l'inversion de ces modèles. On a fait appel à la notion de pseudo- inverse, dans le cas où la matrice jacobienne n'est pas carrée à l'aide de l'algorithme de Greville. Des programmes sous Matlab sont établis pour calculer ces différents modèles.

CHAPITRE IV

Modélisation dynamique

Chapitre IV

MODELISATION DYNAMIQUE

IV.1. INTRODUCTION

Nous présentons dans ce chapitre le modèle dynamique inverse ou tout simplement le modèle dynamique d'un manipulateur à structure ouverte simple, qui est défini par les relations entre les forces généralisées (forces et/ou couples) appliquées aux articulations et les paramètres cinématiques (coordonnées, vitesses et accélérations articulaires) du bras manipulateur [Khal99][Bors11]

On représente le modèle dynamique par :

$$\Gamma=f\ (q,\dot{q},\ddot{q},\ Fext) \qquad (IV.1)$$

Avec ; Γ : Vecteur des couples/forces des actionneurs, selon que l'articulation est rotoïde ou prismatique.

F_{ext}: effort extérieur (forces et couples) à exercer par l'organe terminal.

Cette expression peut être obtenue selon les formalismes et les procédures utilisées. Les plus adoptés pour le calcul de modèle dynamique inverse sont :

- Le formalisme de Lagrange.
- Le formalisme de Newton-Euler.

IV.2. FORMALISME DE LAGRANGE [Domb01][Bors11]

La méthode présentée n'est pas celle qui donne le modèle le plus performant du point de vue nombre d'opérations, elle est utilisée surtout pour construire le modèle dynamique direct. On utilise généralement ce modèle pour la simulation des mécanismes poly-articulés libres.

Nous considérerons un robot idéal sans frottement, sans élasticité et ne subissant ou n'exerçant aucun effort extérieur sur l'organe terminal.

Le formalisme de Lagrange décrit les équations du mouvement par :

$$\Gamma_i = \frac{\partial}{\partial t}\left[\frac{\partial L}{\partial \dot{q}_i}\right] + \frac{\partial L}{\partial q_i}, \qquad \text{i=1… n.} \qquad \text{(IV.2)}$$

Où :

Γ_i: Couple moteur de la i éme articulation,

L : Lagrangien du système égale à E_C -E_P,

E_C: Energie cinétique totale,

E_P: Energie potentielle totale,

n: Nombre total d'articulations.

L'énergie cinétique du système est une forme quadratique des vitesses articulaires :

$$E_c = \frac{1}{2}.\dot{q}^t.A(q).\dot{q} \qquad \text{(IV.3)}$$

Où **A(q)** est la matrice (n×n) d'inertie du robot, qui est symétrique et définie positive.

Le couple **Γ** peut s'écrire à partir des équations (IV.2) et (IV.3), avec l'énergie potentielle étant fonction des variables articulaires $\boldsymbol{q}$, sous la forme matricielle :

$$\Gamma = [\tilde{A}].\ddot{q} + [\tilde{B}].\dot{q}\dot{q} + [\tilde{C}].\dot{q}^2 + Q + \mathbf{F}_{\text{ext}} \qquad \text{(IV.4)}$$

Avec $\widetilde{\mathbf{A}}$: Matrice d'inertie du bras manipulateur.

$\widetilde{\mathbf{B}}$**:** Matrice des termes de Coriolis.

$\widetilde{\boldsymbol{C}}$: Matrice des termes centrifuges.

Q : Vecteur des forces de gravité.

Fext: Vecteur des forces extérieures.

Les expressions des éléments des matrices $\widetilde{\mathbf{B}}$ et $\widetilde{\boldsymbol{C}}$ sont données par les relations suivantes :

$$B_{i,jk} = \frac{\partial A_{ij}}{\partial q_k} + \frac{\partial A_{ik}}{\partial q_j} - \frac{\partial A_{ik}}{\partial q_i}.$$

$$C_{ij} = \frac{\partial A_{ij}}{\partial q_j} - \frac{1}{2}\frac{\partial A_{jj}}{\partial q_i}.$$

IV.3. LE FORMALISME DE NEWTON-EULER

Cette méthode permet d'obtenir directement le modèle dynamique inverse. Elle présente un intérêt pratique indéniable aussi bien pendant la conception que lors de l'exploitation. En effet ce formalisme porte sur le calcul des torseurs complets de liaisons et permet donc le dimensionnement de la structure et des actionneurs. Le caractère itératif de ce formalisme réduit énormément le temps de calcul relativement au formalisme de Lagrange et permet l'exploitation des résultats, par une commande évoluée (commande en couple), pour améliorer la précision dynamique. [Hadd11]

IV3.1. Présentation théorique du formalisme

Considérons l'ensemble des (n+1) corps rigides d'une chaîne cinématique ouverte et dans lequel chaque articulation A_j n'admet qu'un seul mouvement de rotation ou de translation.

Chaque corps C_j de la chaîne cinématique est en équilibre sous l'action :

- Des efforts de liaison dus au corps en aval (le corps C_{j+1}) représenté par un torseur, réduit au point O_{j+1}, de résultante ($-\boldsymbol{RL}_{j+1}$) et de moment résultant ($-\boldsymbol{ML}_{j+1}$).
- Des efforts de liaison dus au corps en amont (le corps C_{j-1}) représenté par un torseur, réduit au point O_j, de résultante ($\boldsymbol{RL}_j$) et de moment résultant ($\boldsymbol{ML}_j$).
- Des efforts d'inertie dus au corps C_j lui-même, représenté par un torseur, réduit au point G_j centre de gravité du corps C_j, de résultante ($\boldsymbol{F}_j$) et de moment résultant ($\boldsymbol{N}_j$).
- Des efforts de gravité représentée par une force ($m.\vec{g}$) au point G_j.

La condition d'équilibre du corps C_j s'exprime par les deux relations ces après :

- Formule de NEWTON :

$$RL_j + M_j * g - RL_{j+1} - F_j = \vec{0} \qquad \text{(IV.5)}$$

- Formule d'EULER:

$$ML_j + \overrightarrow{O_J G_J} \wedge (M * g) - ML_{j+1} - N_j - \overrightarrow{O_J G_J} \wedge F_j - \overrightarrow{O_J O_{J+1}} \wedge RL_{j+1} = \vec{0} \qquad \text{(IV.6)}$$

L'évaluation des efforts d'inertie dus au corps C_j lui-même passe par une étude cinématique générale qui permet d'exprimer la position, la vitesse et l'accélération de n'importe quel point P du bras manipulateur par rapport au repère de base $\{R_0\}$.

IV.3.2. Expressions mises en œuvre en calcul automatique [Khal99][Hadd11][Ghao09]

Ce formalisme est fondée sur double récurrence : une récurrence avant, de la base du robot vers l'effecteur, qui calcule successivement les vitesses et accélérations des corps, puis leur torseur dynamique et une récurrence arrière, de l'effecteur vers la base, qui permet le calcul des couples des actionneurs en exprimant pour chaque corps le bilan des efforts.

Il faut projeter dans un même repère les vecteurs et tenseurs qui apparaissent dans une même équation. Les équations de la récurrence avant, s'écrivent pour j =1…n. Les formules de composition des vitesses et des accélérations sont :

$$\begin{cases} w_{j-1}^{j} = A_{j-1}^{j} * w_{j-1}^{j-1} \\ w_{j}^{j} = w_{j-1}^{j} + \bar{\sigma}_j * \dot{q}_j * z_j \\ \dot{w}_{j}^{j} = A_{j-1}^{j} * \dot{w}_{j-1}^{j-1} + \bar{\sigma}_j * (\ddot{q}_j * z_j + \dot{q}_j * w_{j-1}^{j} \wedge z_j) \\ \dot{V}_j^{j} = A_{j-1}^{j} * [\dot{V}_{j-1}^{j-1} + \dot{w}_{j-1}^{j-1} \wedge {}^{j-1}\mathrm{P_j} + w_{j-1}^{j-1} \wedge (w_{j-1}^{j-1} \wedge {}^{j-1}\mathrm{P_j})] + \sigma_j (\ddot{q}_j * z_j + 2 * \dot{q}_j * w_{j-1}^{j} \wedge z_j) \\ \dot{V}_{Gj} = \dot{V}_j + \dot{w}_j^{J} \wedge S_j + w_j^{j} \wedge (w_j^{j} \wedge S_j) \end{cases} \quad \text{(IV.7)}$$

Le torseur dynamique du corps C_j est :

$$\begin{cases} F_j^{j} = M_j * \dot{V}_{Gj} \\ N_j^{j} = \widetilde{\mathrm{I}_{GJ}} * \dot{w}_j{}^{j} + w_j^{j} \wedge (\widetilde{\mathrm{I}_{GJ}} * w_j^{j}) \end{cases} \quad \text{(IV.8)}$$

On initialise la récurrence par :

$w_0 = 0$, $\dot{w}_j = 0$ et $\dot{V}_0 = -g$ (La base du bras manipulateur fixe)

Avec : $\widetilde{\mathrm{I}_{GJ}}$ est le tenseur d'inertie du corps C_j, par rapport au repère (G_j, X_j, Y_j, Z_j).

M_j la masse du corps Cj ; Sj = $\overrightarrow{O_J G_J}$.

Pour la récurrence arrière, lorsque $j = n...1$:

$$\begin{cases} RL_j^{j} = F_j^{j} + RL_{j+1}^{j} \\ RL_j^{j-1} = A_j^{j-1} * RL_j^{j} \\ ML_j^{j} = N_j^{j} + A_{j+1}^{j} * ML_{j+1}^{j+1} + P_{j+1}^{j} \wedge RL_{j+1}^{j} + S_j \wedge F_j^{j} \end{cases} \quad \text{(IV.9)}$$

Récurrence initialisée par les efforts notés RL_{j+1} et ML_{j+1}, respectivement F_{ext} et ML_{ext} qu'exerce l'organe terminal sur l'environnement.

On obtient les couples des actionneurs Γ_j en projetant, selon la nature de la liaison A_j, le vecteur RL_j ou ML_j sur l'axe de mouvement. On ajoute les résistances complémentaires issues des actionneurs eux-mêmes (frottements et inerties des actionneurs).

$$\Gamma_j = \left(\sigma_j * RL_j^j + \bar{\sigma}_j * ML_j^j\right) * z_j + Fs_j * Sgn\left(\dot{q}_j\right) + Fv_j * \dot{q}_j + Ia_j * \ddot{q}_j \qquad \text{(IV.10)}$$

IV.4. CONCLUSION

Ce chapitre présente la modélisation dynamique inverse permettant de calculer la puissance de robot.

Plusieurs formalismes peuvent être utilisés pour obtenir le modèle dynamique. Dans ce travail, on parle de formalismes de Lagrange et de Newton-Euler. Dans la programmation, seule le deuxième formalisme est utilisé en raison de ses avantages consistant à la possibilité d'établir le calcul en ligne avec un temps réduit.

CHAPITRE V

Génération de mouvement

Chapitre V

GENERATION DE MOUVEMENT

V.1. INTRODUCTION

A partir de la tâche du robot, on définit un chemin qu'il doit suivre. Une trajectoire ou un chemin est une suite de points, correspondant aux situations successives de l'organe terminal (espace opérationnel) ou de configurations articulaires. Ces points peuvent être :

- programmés par apprentissage.
- donnés en ligne par un capteur extéroceptif.
- issue d'une base de données d'un système de CAO…

La génération de mouvement désigne la fonction de calcul des consignes articulaires du robot destinées à réaliser une tâche interprétée sous forme de position successives de l'outil du robot et de contraintes cinématiques ou dynamiques, de telle façon que chacune de ces articulations commence et termine le mouvement en même temps.

Cette fonction est centrale au sein du contrôleur du robot, dans le sens où elle fait la transition entre le niveau informatique d'interprétation- exécution du programme de la tâche - et le niveau de commande proprement dit, comme l'illustre le schéma simplifié de contrôleur de robot industriel de la figure V.1.

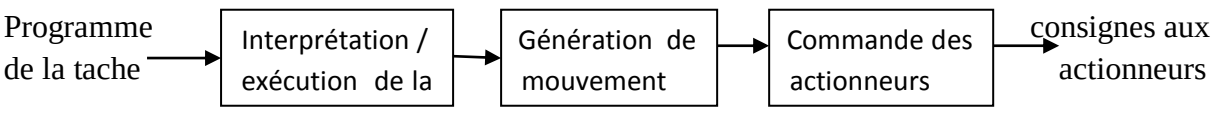


Figure V.1 : Placement de la fonction génération de mouvement au sein du contrôleur de robot industriel.

Une trajectoire est le support d'un mouvement. La planification de mouvements vise à trouver des trajectoires géométriques et des lois de commande à appliquer aux actionneurs le long de trajectoires pour atteindre une position donnée. Plusieurs classes de mouvements peuvent être distinguées suivant la tâche :

- Mouvement entre deux points avec trajectoire libre;
- Mouvements entre deux points avec points intermédiaires spécifiés en particulier pour éviter les obstacles, et trajectoire libre entre les points intermédiaires ;
- Mouvement entre deux points avec trajectoire contrainte ; par exemple trajectoire rectiligne ;
- Mouvement entre deux points avec points intermédiaires, et trajectoire contrainte entre points intermédiaires.

Dans les deux premiers cas, la génération de trajectoires peut être réalisée directement dans l'espace articulaire. Dans les deux derniers cas, comme la trajectoire est décrite dans l'espace opérationnel, il est préférable de raisonner dans cet espace. [khal99]

Nous verrons dans ce chapitre, le fondement mathématique de la génération de mouvements est le calcul polynomial qui construit l'équation de mouvement à partir de contraintes spatiales et temporelles. Mais avant d'aborder les calculs mathématiques, nous allons comparer entre la génération de mouvement dans l'espace articulaire et dans l'espace opérationnel.

V.2. GENERATION DE MOUVEMENT DANS L'ESPACE ARTICULAIRE ET DANS L'ESPACE OPERATIONNEL (commande, avantages et inconvénients)

Le choix de l'espace de génération de mouvement dépend fortement de l'application considérée. Chacune des approches à ses propres limites, inhérentes au fait que les contraintes sont exprimées soit dans l'espace articulaire (butées, vitesse et couple maximum), soit dans l'espace opérationnel (précision, prise en compte des obstacles). Les deux figures V.2 et V.3 respectivement donne le schéma des deux méthodes articulaire et opérationnel.

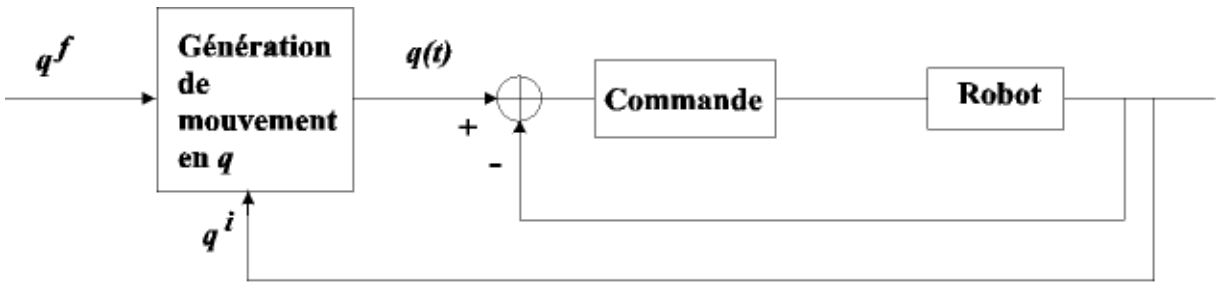


Figure V.2 : Génération de mouvement dans l'espace articulaire [Boim01]

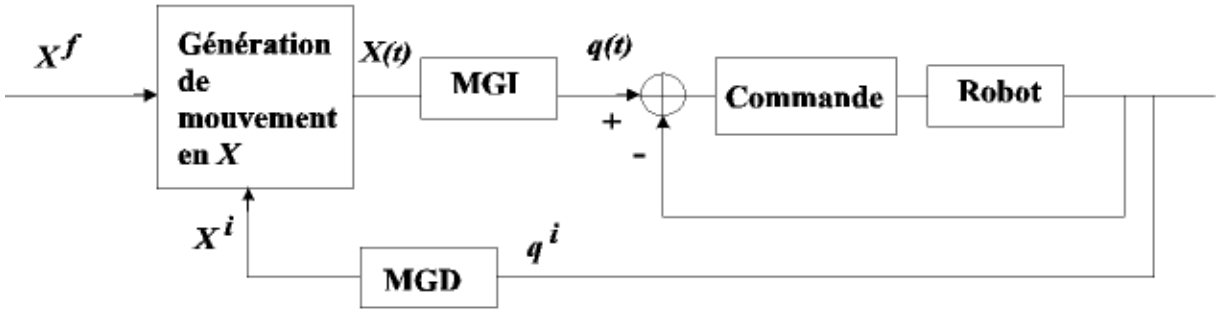


Figure V.3 : Génération de mouvement dans l'espace opérationnel [Boim01]

V.2.1. Génération de mouvement dans l'espace articulaire [Khal99]

Dans le cas de la génération des mouvements dans l'espace articulaire, les consignes sont calculées et synchronisées l'ensemble des articulations sur la plus lente (appelée articulation maître) afin que tous les axes passent en même temps sur les points imposés.

avantages	inconvénients
• Elle nécessite moins de calcul en ligne car il n'y a pas d'appel au modèle géométrique direct et inverse. • Le mouvement peut être effectué sans passage par les configurations singulières • Les contraintes de vitesse et de couples maximaux sont déduites directement des limites physiques des actionneurs.	• La géométrie de la trajectoire de l'organe terminal est imprévisible : risque de collisions lorsque le robot évolue dans un environnement très encombré.

Remarque

Ce type de mouvement est par conséquent approprié pour réaliser des déplacements rapides dans un espace dégagé.

V.2.2. Génération de mouvement dans l'espace opérationnel [Khal99]

La génération des mouvements dans l'espace opérationnel implique une trajectoire en position cartésienne avec une orientation de l'effecteur par rapport à un repère de base du robot.

avantages	inconvénients
• Maîtrise de la trajectoire	• Transformation de coordonnées de chaque point de la trajectoire • Possibilité de mise en échec quand la trajectoire passe par un point singulier • Possibilité de mise en échec si les points de la trajectoire ne sont pas dans le volume accessible du robot ou chaque fois que la trajectoire nécessite un reconfiguration du robot • Les limites en vitesse et en couple du robot varient en fonction de la configuration. On impose en général ces limites en termes de performances moyennes valables quelle que soit la configuration. On travaille donc en deçà des capacités réelles du robot.

V.3. GENERATION DE MOUVEMENT ENTRE DEUX POINTS DANS L'ESPACE ARTICULAIRE [Hadd11]

On considère un robot à n degrés de liberté. Soit $\boldsymbol{q^i}$ et $\boldsymbol{q^f}$ les vecteurs de coordonnées articulaires correspondant aux configurations initiale et finale. On désigne respectivement par ***Kv*** et ***Ka*** les vecteurs des vitesses et accélérations maximales.

Le mouvement entre $\boldsymbol{q^i}$ et $\boldsymbol{q^f}$ étant fonction du temps ***t***, est décrit par l'équation suivante :

$$q(t) = q^i + r(t).D \qquad 0 \le t \le t_f \tag{V.1}$$

$$\dot{q} = \dot{r}(t).D \qquad \text{Avec } D = q^f - q^i$$

Les valeurs aux limites de la fonction d'interpolation r(t) sont données par :

$$\begin{cases} r(0) = 0. \\ r(t_f) = 1. \end{cases}$$

L'expression (V.1) s'écrit aussi :

$$q(t) = q^f(t) - [1 - r(t)].D \tag{V.2}$$

Plusieurs fonctions permettent de satisfaire le passage de $\boldsymbol{q^i}$ correspondant à *t = 0* à $\boldsymbol{q^f}$ correspondant à *t=tf.*

V.3.1. Interpolation polynomiale

A- Interpolation linéaire

Il s'agit de l'interpolation la plus simple, le mouvement de chaque articulation est décrit par une équation linéaire en temps. L'équation du mouvement s'écrit :

$$\begin{cases} r = \frac{t}{t_f} \\ q(t) = q^i + \frac{t}{t_f}.D \end{cases} \quad \text{(V.3)}$$

Cette loi de mouvement est continue en position mais discontinue en vitesse. L'utilisation d'une telle loi de mouvement est inacceptable sur les robots réels à cause des à-coups qu'elle provoque.

Le temps minimum t_f est égal : $MAX(D_j/Kv_j)$.

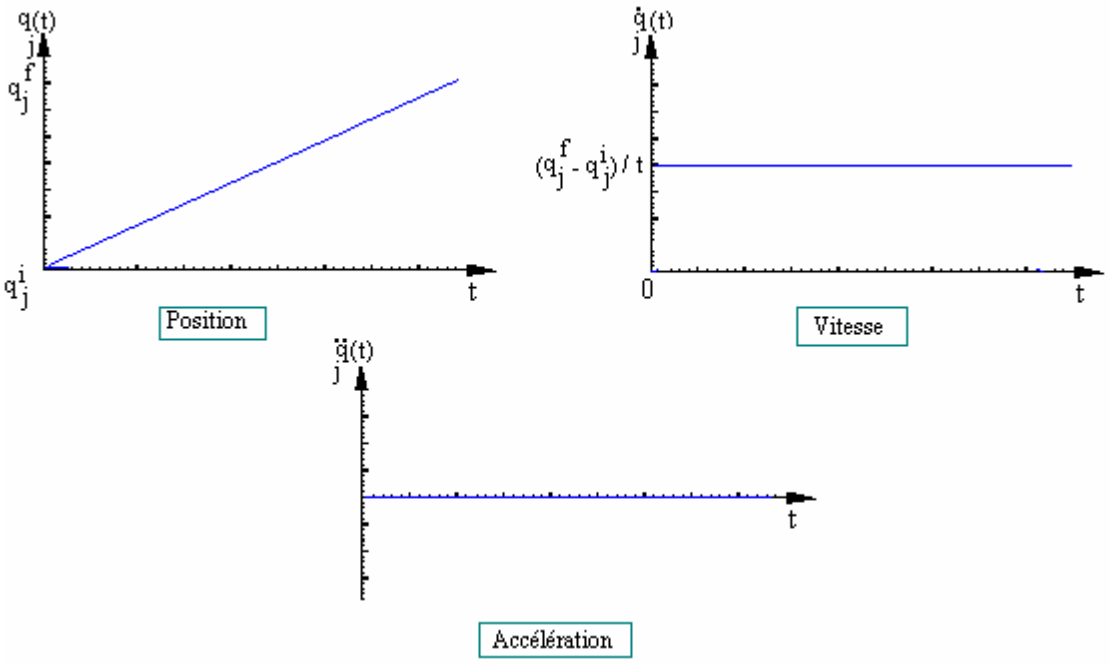


Figure V.4 : Interpolation linéaire sur une articulation A_j donnée.

B- Polynôme de degré trois

Si l'on impose une vitesse nulle aux points de départ et d'arrivée, on ajoute deux contraintes aux deux contraintes de position. Le degré minimal du polynôme qui satisfait ces quatre contraintes est de degré trois et a pour forme générale :

$$q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \quad \text{(V.4)}$$

Avec les conditions aux limites :

$$\begin{cases} a_0 = q^i \\ a_1 = 0 \\ a_2 = \frac{3}{{t_f}^2} D \\ a_3 = -\frac{2}{{t_f}^3} D \end{cases}$$

Ce qui conduit à la fonction d'interpolation suivante :

$$r(t) = 3\left(\frac{t}{t_f}\right)^2 - 2\left(\frac{t}{t_f}\right)^3 \qquad \text{(V.5)}$$

Pour une articulation quelconque A_j, la vitesse est maximum lorsque t=t_f/2. Elle a pour valeur :

$$|\dot{q}_{j\,max}| = \frac{3|D_j|}{2t_f} ;$$

Et l'accélération est maximale à t = 0 et à $t = t_f$ est égale à :

$$|\ddot{q}_{j\,max}| = \frac{6|D_j|}{{t_f}^2} ;$$

Le temps minimum t_f est donné par : $t_f = \max(t_{f1}, \dots, t_{fn})$.

Avec $$t_{fj} = max\left[\frac{3|D_j|}{2k_{vj}}, \sqrt{\frac{6|D_j|}{k_{aj}}}\right].$$

Cette loi de mouvement assure la continuité des vitesses mais pas celle des accélérations. En pratique, les robots industriels sont suffisamment rigides pour que cette discontinuité soit filtrée par la mécanique.

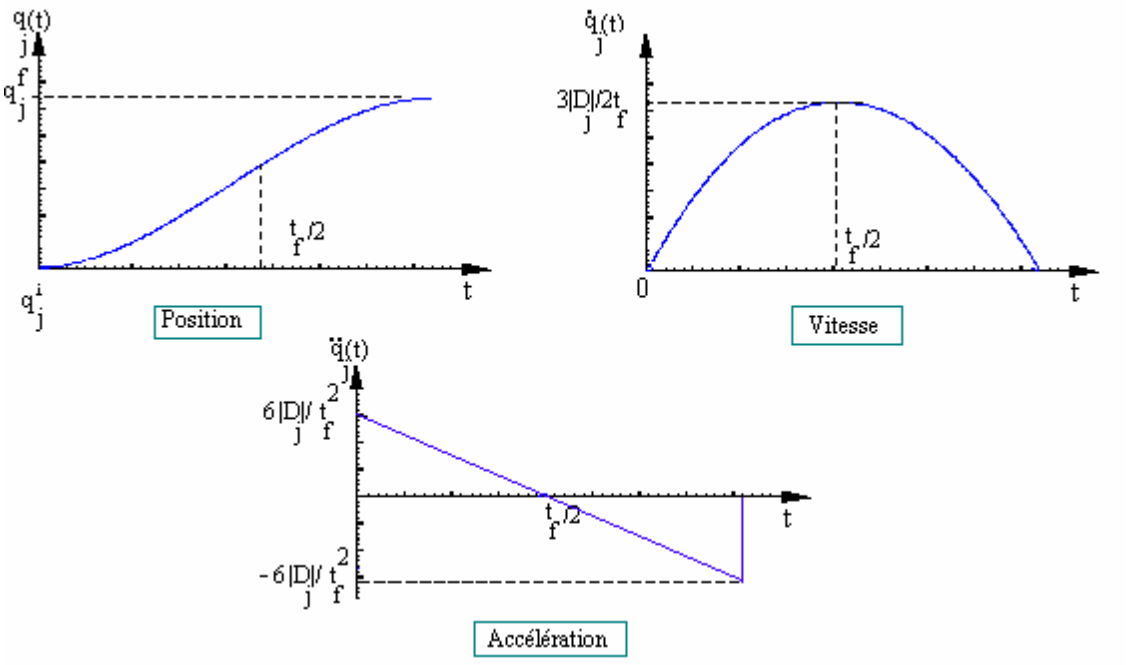


Figure V.5 : Loi polynômiale de degré trois.

C- Polynôme de degré cinq

Nous avons vu que l'interpolation de degré trois assure la continuité en vitesse mais non en accélération. Si l'on recherche en plus la continuité des accélérations, il faut satisfaire six contraintes et le polynôme d'interpolation doit être de degré cinq. Avec les contraintes supplémentaires:

$$\begin{cases} \ddot{q}(0) = 0. \\ \ddot{q}(t_f) = 0. \end{cases}$$

Alors la fonction d'interpolation est :

$$r(t) = 10\left(\frac{t}{t_f}\right)^3 - 15\left(\frac{t}{t_f}\right)^4 + 6\left(\frac{t}{t_f}\right)^5. \qquad \text{(V.6)}$$

La vitesse et accélération maximales ont pour expression :

$$\left|\dot{q}_{j\,max}\right| = \frac{15|D_j|}{8t_f} \quad \text{et} \quad \left|\ddot{q}_{j\,max}\right| = \frac{10|D_j|}{\sqrt{3}{t_f}^2}$$

Le temps minimum est donné par :

$$t_{fj} = max\left[\frac{15|D_j|}{8k_{vj}}, \sqrt{\frac{10|D_j|}{\sqrt{3}k_{aj}}}\right].$$

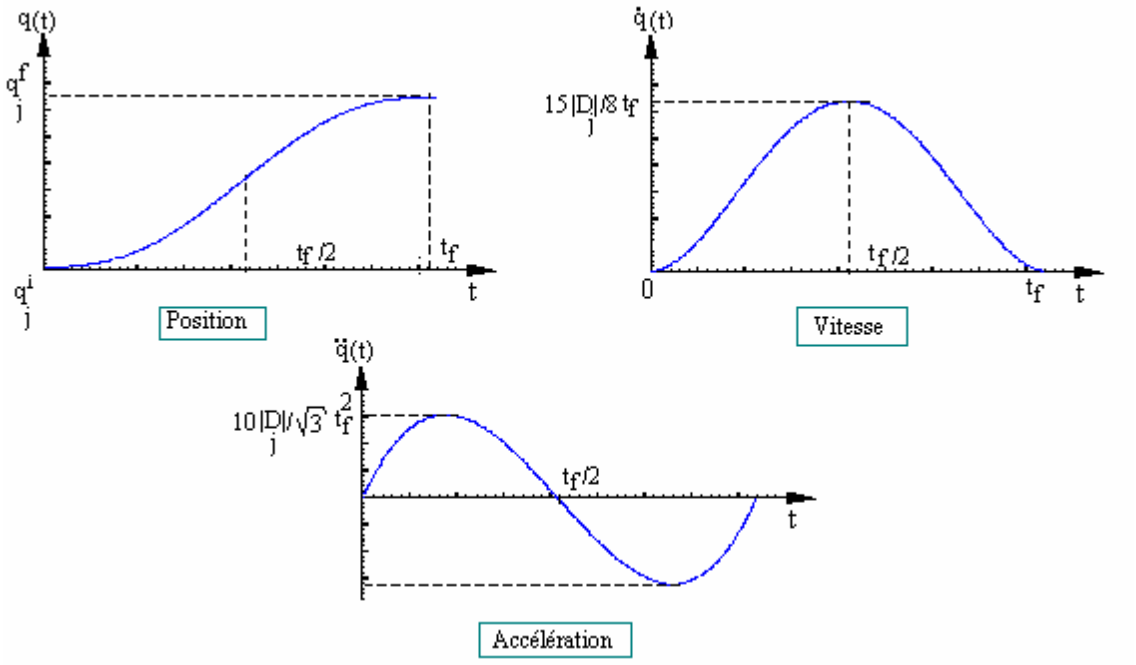


Figure V.6 : Loi polynômiale de degré cinq.

V.3.2. Loi Bang-Bang

Le mouvement est représenté par une phase d'accélération constante jusqu'à t=t_f/2 puis par une phase de décélération constante. Les vitesses de départ et d'arrivée sont nulles. Le mouvement est donc continu en position et en vitesse, discontinu en accélération.

La position est donnée par :

$$\begin{cases} q(t) = q^i + 2\left(\frac{t}{t_f}\right)^2 D & \text{Pour } 0 \le t \le t_f/2 \\ q(t) = q^i + \left[-1 + 4\left(\frac{t}{t_f}\right) - 2\left(\frac{t}{t_f}\right)^2\right] D & \text{Pour } t_f/2 \le t \le t_f \end{cases} \quad \text{(V.7)}$$

Pour une articulation j donnée, les vitesses et accélérations maximales ont pour expression :

$$\left|\dot{q}_{j\,max}\right| = \frac{2|D_j|}{t_f} \;; \qquad \left|\ddot{q}_{j\,max}\right| = \frac{4|D_j|}{{t_f}^2}.$$

Le calcul du temps minimum se fait comme dans le cas polynômiale avec :

$$t_{fj} = max\left[\frac{2|D_j|}{k_{vj}}, 2\sqrt{\frac{|D_j|}{k_{aj}}}\right]$$

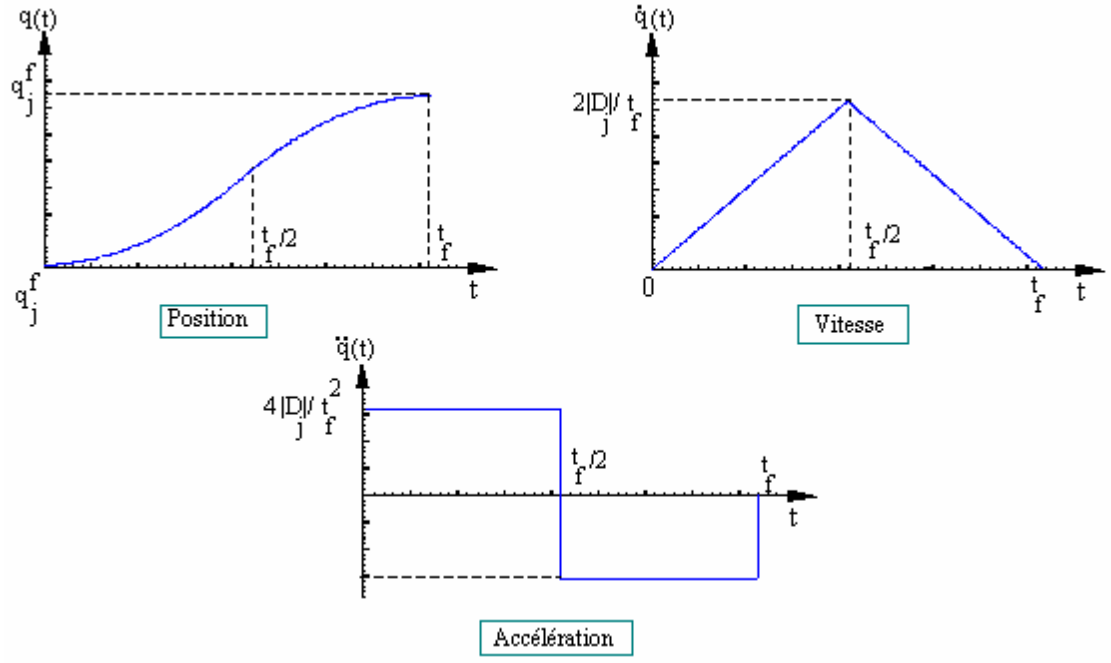


Figure V.7 : Loi Bang –Bang.

Avec la loi Bang-Bang, le temps minimal est assuré en saturant soit la vitesse soit l'accélération. Lorsque la vitesse est saturée, le fait de rajouter un palier de vitesse permet de saturer aussi l'accélération et de diminuer le temps de parcours. C'est la description de la loi trapèze.

V.3.3. Loi trapèze (loi Bang–Bang avec palier de vitesse)

Ce type de loi est le plus couramment implanté sur les contrôleurs de robots industriels. La loi trapèze est la loi optimale en temps parmi celles qui assurent la continuité en vitesse.

Le mouvement d'une articulation *Aj* donnée est représenté par les relations :

$$q(i) = \begin{cases} q^i + D_j.\frac{t^2}{2t_a(t_f - t_a)} & pour\ 0 \le t \le t_a \\ q^i + D_j.\frac{2t - t_a}{2(t_f - t_a)} & pour\ t_a \le t \le (t_f - t_a) \\ q^i + D_j.\left[1 - \frac{(t_f - t)^2}{2t_a(t_f - t_a)}\right] & pour\ (t_f - t_a) \le t \le t_f \end{cases} \quad (V.8)$$

Avec *Ta* et *Tf* sont, respectivement, le temps de la phase d'accélération (décélération) et le temps final minimum du déplacement. Ces deux paramètres sont calculés, en fonction des vitesses ($\dot{q}_j^{max}$) et accélérations ($\ddot{q}_j^{max}$) maximales des actionneurs, de la manière suivante :

$$Mrv = \max_{j=1\ldots n}\left[\frac{D_j}{\dot{q}_j^{max}}\right] \quad ; \quad Mra = \max_{j=1\ldots n}\left[\frac{D_j}{\ddot{q}_j^{max}}\right]$$

Si $\sqrt{Mra} \ge Mrv : t_f = 2\sqrt{Mra}$ et $t_a = \frac{t_f}{2}$,

Sinon $t_f = \frac{Mrv^2 + Mra}{Mrv}$ et $t_a = \frac{Mra}{Mrv}$

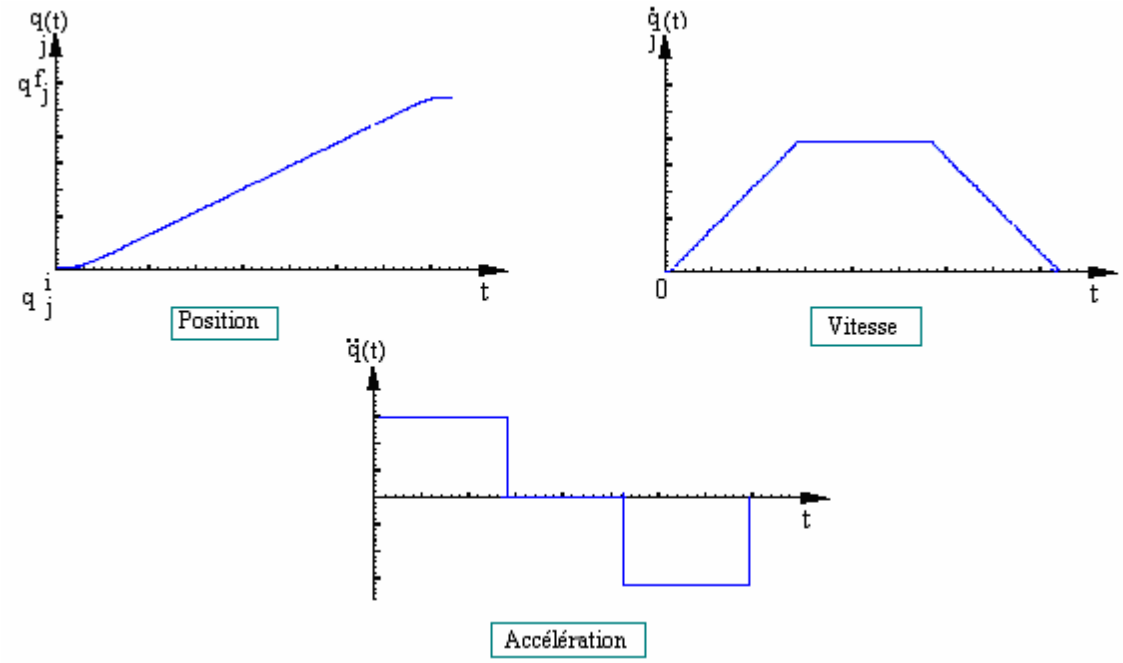


Figure V.8 : Evolution de la position, vitesse et accélération de l'articulation Aj avec une loi trapèze.

V.4. GENERATION DE MOUVEMENT ENTRE DEUX POINTS DANS L'ESPACE OPERATIONNEL [Khal99]

Soit ${}^{0}T^{i}_{E}$ et ${}^{0}T^{f}_{E}$ les matrices homogènes décrivant respectivement les situations initiale et finale désirées, notées comme suit :

$$T_E^i = \begin{bmatrix} & A^i & & P^i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad , \quad T_E^f = \begin{bmatrix} & A^f & & P^f \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{V.9}$$

On recherche une trajectoire rectiligne du point outil. On décompose le mouvement en un mouvement de translation en ligne droite entre les origines de ${}^{0}T^{i}_{E}$ et ${}^{0}T^{f}_{E}$, et en un mouvement de rotation α autour d'un axe ${}^{E}\mathbf{U}$ de l'organe terminal permettant d'aligner A^i et A^f. Les deux mouvements se terminent en même temps. La distance D à parcourir pour le mouvement de translation est telle que :

$$D = \|P^f - P^i\| = \sqrt{\left(P_x^f - P_x^i\right)^2 + \left(P_y^f - P_y^i\right)^2 + \left(P_z^f - P_z^i\right)^2} \tag{V.10}$$

Le calcul de **U** et de α se fait à partir de la relation :

$$A^i rot(\boldsymbol{U}, \alpha) = A^f \tag{V.11}$$

Où, rappelons-le, **rot** (**U**, α) désigne la matrice (3 × 3) de rotation correspondant à une rotation d'un angle α autour d'un vecteur **U**. on en tire que :

$$\mathbf{rot}(\mathbf{U}, \alpha) = \left[\mathrm{A^i}\right]^{\mathrm{T}} \mathrm{A^f} = \begin{bmatrix} \mathrm{s^{iT}} \\ \mathrm{n^{iT}} \\ \mathrm{a^{iT}} \end{bmatrix} \begin{bmatrix} s^f & n^f & a^f \end{bmatrix} = \begin{bmatrix} s^i.s^f & s^i.n^f & s^i.a^f \\ n^i.s^f & n^i.n^f & n^i.a^f \\ a^i.s^f & a^i.n^f & a^i.a^f \end{bmatrix} \tag{V.12}$$

On déduit que :

$$\begin{cases} C\alpha = \frac{1}{2}\left[s^i.s^f + n^i.n^f + a^i.a^f - 1\right] \\ S\alpha = \frac{1}{2}\sqrt{(a^i.n^f - n^i.a^f)^2 + (s^i.a^f - a^i.s^f)^2 + (n^i.s^f - s^i.n^f)^2} \\ \alpha = atan2\,(S\alpha, C\alpha) \\ U = \frac{1}{2S\alpha}\begin{bmatrix} a^i.n^f - n^i.a^f \\ s^i.a^f - a^i.s^f \\ n^i.s^f - s^i.n^f \end{bmatrix} \end{cases} \tag{V.13}$$

L'évolution de l'orientation de l'outil par rapport au vecteur ***U*** sera régie par la matrice suivante :

$$ {}^{0}_{E}T(t) = \begin{bmatrix} A(t) & P(t) \\ 0 \quad 0 & 0 \quad 1 \end{bmatrix}. $$

Avec :

$$ \begin{cases} P(t) = P^{i} + r(t)\left(P^{f} - P^{i}\right). \\ A(t) = A^{i} rot(U, r(t)\alpha). \end{cases} $$

V.5. GENERATION DE MOUVEMENT AVEC POINT INTERMEDIAIRE

Ce type de trajectoire est contraint par des configurations intermédiaires imposées ; la trajectoire est contrainte à passer par des points intermédiaires. Cela peut être le cas si la tâche du manipulateur est définie de manière à éviter des obstacles et/ou pour des opérations de «pick and place» simples. Dans ce dernier cas, il suffira de spécifier une position intermédiaire sur la normale à la surface de pose qui passe par le point de prise : en imposant à la pince de passer par cette position intermédiaire, on obtient un mouvement admissible pour le départ et pour la phase d'approche nécessaire à la dépose de l'objet manipulé [Khal99].

V.6. CONCLUSION

Le problème de la génération des mouvements des robots industriels est complexe, il s'agit fondamentalement d'un problème d'optimisation qui vise à minimiser la durée du mouvement en tenant compte à la fois des contraintes géométriques et dynamiques.

Dans ce chapitre on a présenté les techniques de génération de mouvement couramment utilisées en robotique : les différents modes d'interpolation, la loi Bang-Bang avec ou sans palier de vitesse qui est implantée sur la plupart des contrôleurs industriels. Pour chacune, on a donné l'expression du temps minimum, temps à partir duquel on peut réaliser la coordination articulaire.

La génération de mouvement est une phase importante, elle permet à l'opérateur de programmer la trajectoire selon la tâche qu'il veut accomplir.

CHAPITE VI

Application au bras de mesure 3D

Chapitre VI

APPLICATION AU BRAS DE MESURE 3D

VI. 1. INTRODUCTION

Dans le présent chapitre, nous appliquons les outils précédemment développés pour une simulation représentative d'une situation de mesure tridimensionnelle (3D).

La démarche exposée dans les précédents chapitres ainsi que les outils développés sont mis en œuvre sur cette tâche en passant par :

- le choix de l'application (numérisation des surfaces gauches);
- la représentation de la tâche par un ensemble de points de passage;
- le calcul de MGI, MCD du 1^{er} et $2^{ème}$ ordre;
- le calcul de la puissance par le modèle dynamique;
- l'extraction des résultats sous forme des courbes;
- la simulation des mouvements du robot.

VI. 2. DESCRIPTION DE LA TACHE

On cherche à palper une surface d'un solide pour faire la comparaison par rapport au modèle mathématique ou par rapport à un fichier de points donnés. A titre d'exemple, nous prenons comme surface un plan d'une pièce pour sa simplicité dans la détermination de la normale en chaque point à palper. La figure VI.1 illustre cette surface (surface en gris).

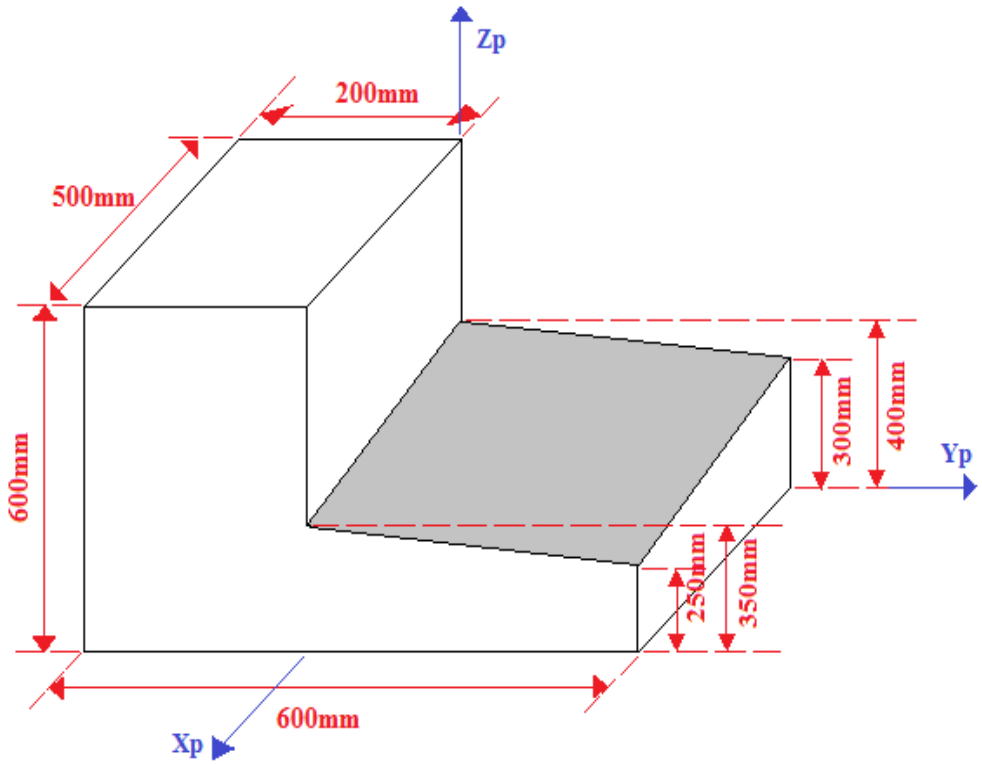


Figure VI.1 : Piece à palper.

VI.3. CHOIX DU ROBOT

Le choix du robot dépend de la tâche à réaliser. Il doit avoir un nombre de degré de liberté (ddl) égal ou supérieur à celui de la tâche. Dans le cas présent, la tâche étant un mesurage en 3D, son ddl est de 3. On choisit un robot à 5ddl rotoïdes pour augmenter ses possiblités d'accès aux points de surafces gauches ou intérieurs. Les trois premières articulations représentent le porteur type articulé, les deux autres sont le poinget à 2 axes concourant. Le robot appartient à la classe des robots à structure ouverte simple.

VI.4. DESCRIPTION DE LA TRAJECTOIRE

La tâche est répétitive pour chaque opération de mesure. Le relevé des points s'effectue par la méthode point à point (coordonnées et normales connues).

Le robot effectue des allers et retours entre deux points. Le mouvement commence par le point P'1 du plan parallèle au plan palpé qui sont distants de 50 mm. La première trajectoire est entre P'1 et P1 suivant la normale du plan. La vitesse en chacun de ces points est nulle. Lorsque le palpeur atteint le point P1, il prend ses coordonnées et revient à sa position initiale P'1. Ensuite, il passe à un autre point P'2 pour palper P2, et ainsi de suite pour les autres points restants. Dans notre application, nous avons pris 9 points (figureVI.2**).**

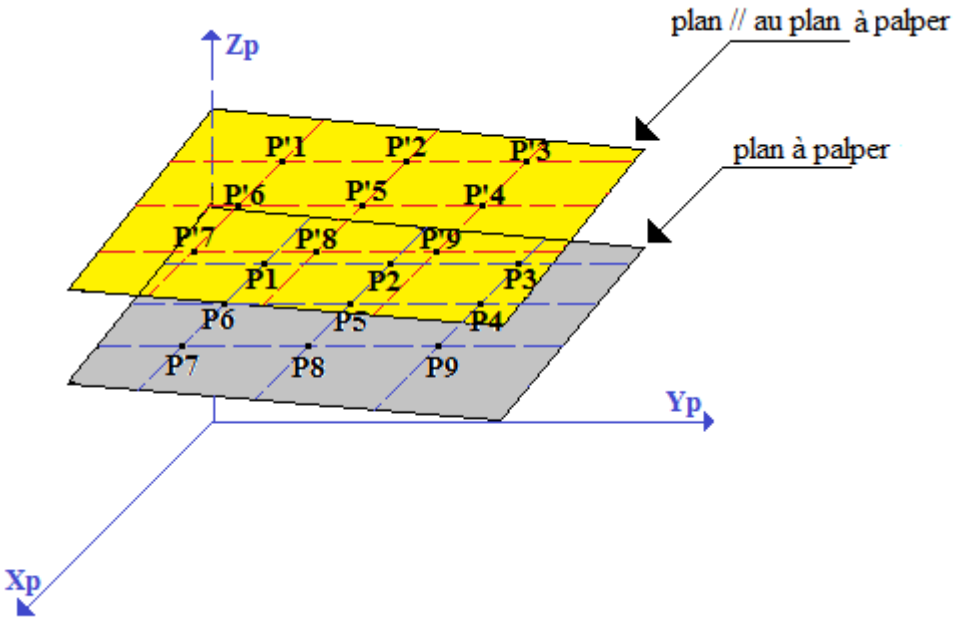


Figure VI.2 : Représentation des 9 points sur deux plans.

VI. 5. MODELISATION

À chacun des corps C_i (i=1...5), on associe un repère R_i (Oi, Xi, Yi, Zi) défini suivant le formalisme de Denavit-Hartenberg Modifié (Figure VI.3 **)**. Le repère universel est appelé R_0. On note q1, q2, … q5, les coordonnées articulaires, où qi exprime l'angle de rotation du corps C_i par rapport à C_{i-1} autour de l'axe Z_i. Compte tenu de la convention et du sens de rotation retenu pour chacune des articulations, on obtient les paramètres de Denavit-Hartenberg Modifié qui sont représentés dans le Tableau VI.1.

VI.5.1. Shéma cinématique

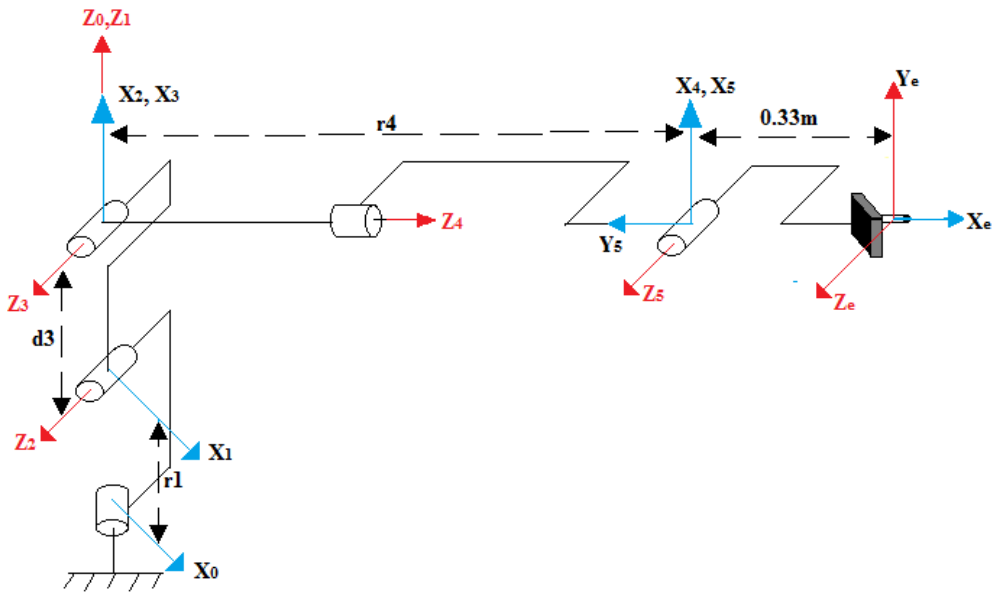


Figure VI.3 : Shéma cinématique du robot.

- **Tableau VI.1 :** Paramètres D-H-M.

Artiq	σ	$\overline{\sigma}$	α_i	d_i	r_i	q_i
1	0	1	0	0	r_1	q_1
2	0	1	$\pi/2$	0	0	q_2
3	0	1	0	d_3	0	q_3
4	0	1	$\pi/2$	0	r_4	q_4
5	0	1	$-\pi/2$	0	0	q_5

- **Tableau VI.2 :** Dimensionnement du robot.

Artiq	α_i(°)	$d_i(mm)$	$r_i(mm)$	q_i(min)(°)	q_i(max)(°)
1	0	0	409.82	-180	+180
2	90	0	0	-90	+90
3	0	1000	0	-110	+110
4	90	0	1513.9	-350	+350
5	-90	0	0	-105	+105

VI.5.2. Paramètres mécaniques du robot

- **Tableau VI.3 :** Vitesses et accélérations max.

Artiq	Vitesse des actionneurs K_V [rd/s]	Accélération des actionneurs K_A [rd/s^2]
1	1.75	28
2	1.75	28
3	1.75	28
4	1	25
5	1	25

- **Tableau VI.4 :** Masse des corps [Kg].

Artiq	1	2	3	4	5
Masse [Kg]	20	17.4	4.8	0.82	0.34

- **Tableau VI.5 :** Centre de gravité [m].

Artiq	1	2	3	4	5
X	0	0.068	0	0	0
Y	0	-0.006	-0.07	0	0
Z	-0.15	-0.2275	-0.014	-0.019	0

➢ **Tableau VI.6 :** Tenseur d'inertie [m^2].

Artiq	I_{XX}	I_{YY}	I_{ZZ}	I_{XY}	I_{XZ}	I_{YZ}
1	0	0	0.35	0	0	0
2	1.031	1.505	0.620	0.00709	0.269	-0.023
3	0.090	0.013	0.109	0	0	-0.0047
4	0.00209	0.00209	0.0013	0	0	0
5	0.0003	0.0003	0.00004	0	0	0

VI.5.3. Matrices de transformation

$$T_1^0 = \begin{bmatrix} Cq_1 & -Sq_1 & 0 & 0 \\ Sq_1 & Cq_1 & 0 & 0 \\ 0 & 0 & 1 & r_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad ; \quad T_2^1 = \begin{bmatrix} Cq_2 & -Sq_2 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ Sq_2 & Cq_2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_3^2 = \begin{bmatrix} Cq_3 & -Sq_3 & 0 & d_3 \\ Sq_3 & Cq_3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad ; \quad T_4^3 = \begin{bmatrix} Cq_4 & -Sq_4 & 0 & 0 \\ 0 & 0 & -1 & -r_4 \\ Sq_4 & Cq_4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_5^4 = \begin{bmatrix} Cq_5 & -Sq_5 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -Sq_5 & -Cq_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

VI.5.4. Calcul du MGD

$$T_5^0 = T_1^0 * T_2^1 * T_3^2 * T_4^3 * T_5^4$$

$$T_5^0 = \begin{bmatrix} Cq_1.[Cq_4.Cq_5.C(q_2+q_3) - Sq_5.S(q_2+q_3)] + Sq_1.\ Sq_4.Cq_5 \\ Sq_1.[Cq_4.Cq_5.C(q_2+q_3) - Sq_5.S(q_2+q_3)] - Cq_1.\ Sq_4.Cq_5 \\ Cq_4.Cq_5.S(q_2+q_3) + Sq_5.C(q_2+q_3) \\ 0 \end{bmatrix}$$

$$\begin{matrix} Cq_1.[-Cq_4.Sq_5.C(q_2+q_3) - Cq_5.S(q_2+q_3)] - Sq_1.\ Sq_4.Sq_5 \\ Sq_1.[-Cq_4.Sq_5.C(q_2+q_3) - Cq_5.S(q_2+q_3)] + Cq_1.\ Sq_4.Sq_5 \\ -Cq_4.Sq_5.S(q_2+q_3) + Cq_5.C(q_2+q_3) \\ 0 \end{matrix}$$

$$
\begin{array}{|c|}
\hline
-Cq_1.Sq_4.C(q_2+q_3)+Sq_1.\ Cq_4 \\
-Sq_1.Sq_4.C(q_2+q_3)-Cq_1.\ Cq_4 \\
-Sq_4.S(q_2+q_3) \\
0 \\
\hline
Cq_1[r_4.S(q_2+q_3)+d_3.Cq_2] \\
Sq_1[r_4.S(q_2+q_3)+d_3.Cq_2] \\
-r_4.C(q_2+q_3)+d_3.Sq_2+r_1 \\
1 \\
\hline
\end{array}
$$

VI.5.5. Calcul du MGI

La situation désirée est : $U_0 = \begin{bmatrix} S_x & N_x & A_x & P_x \\ S_y & N_y & A_y & P_y \\ S_z & N_z & A_z & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = T_5^0$

Pour positionner le robot, on cherche à déterminer les coordonnées articulaires $q_1,\ q_2\ et\ q_3$ à partir des points $P = \begin{cases} P_x \\ P_y \\ P_z \end{cases}$ que l'effecteur (palpeur) doit atteindre.

On déduit le système d'équations :

$$
\begin{cases}
Cq_1[r_4.S(q_2+q_3)+d_3.Cq_2] = P_x & \text{(VI.1)} \\
Sq_1[r_4.S(q_2+q_3)+d_3.Cq_2] = P_y & \text{(VI.2)} \\
-r_4.C(q_2+q_3)+d_3.Sq_2+r_1 = P_z & \text{(VI.3)}
\end{cases}
$$

On prend les deux premières équations :

$$
\begin{cases}
Cq_1[r_4.S(q_2+q_3)+d_3.Cq_2] = P_x \\
Sq_1[r_4.S(q_2+q_3)+d_3.Cq_2] = P_y
\end{cases}
$$

$\Rightarrow \dfrac{Sq_1}{Cq_1} = \dfrac{P_y}{P_x} = \text{tg } q_1 \qquad \Rightarrow \qquad q_1 = \text{Arctg}\left(\dfrac{P_y}{P_x}\right)$

Pour chercher q_2, en multipliant les équations (VI.1) et (VI.2) respectivement par Cq_1, Sq_1 :

$$\begin{cases} r_4.{Cq_1}^2.S(q_2+q_3)+d_3.{Cq_1}^2.Cq_2=P_x.Cq_1 \\ r_4.{Sq_1}^2.S(q_2+q_3)+d_3.{Sq_1}^2.Cq_2=P_y.Sq_1 \end{cases}$$

La somme de ces deux équations sonne :

$$r_4.(\cancel{{Cq_1}^2+{Sq_1}^2}^{1}).S(q_2+q_3)+d_3.(\cancel{{Cq_1}^2+{Sq_1}^2}^{1}).Cq_2=P_x.Cq_1+P_y.Sq_1$$

Apres simplification de cette équation et à l'aide de l'équation (VI.3) :

$$\begin{cases} r_4.S(q_2+q_3)+d_3.Cq_2=P_x.Cq_1+P_y.Sq_1=A \\ -r_4.C(q_2+q_3)+d_3.Sq_2=P_z-r_1=B \end{cases}$$

$$\begin{cases} r_4.S(q_2+q_3)=A-d_3.Cq_2 \\ r_4.C(q_2+q_3)=-B+d_3.Sq_2 \end{cases} \qquad \text{(VI.4)}$$

On enlève au carré les deux équations :

$$\begin{cases} {r_4}^2.S(q_2+q_3)^2=(A-d_3.Cq_2)^2 \\ {r_4}^2.C(q_2+q_3)^2=(-B+d_3.Sq_2)^2 \end{cases}$$

On somme entre ces deux dernières équations, on obtient :

$${r_4}^2.[C(q_2+q_3)^2+\cancel{S(q_2+q_3)^2}^{1}]=(A-d_3.Cq_2)^2+(-B+d_3.Sq_2)^2$$

Apres simplification, on trouve :

$$2.d_3.(A.Cq_2+B.Sq_2)={A^2}+{B^2}+{d_3}^2-{r_4}^2$$

On note : $$A.Cq_2+B.Sq_2=C \qquad \text{(VI.5)}$$

En utilisant le changement de variable suivant :

$$Cq_2=\frac{1-\left(tg\frac{q_2}{2}\right)^2}{1+\left(tg\frac{q_2}{2}\right)^2} \qquad \text{et} \qquad Sq_2=\frac{2.tg\frac{q_2}{2}}{1+\left(tg\frac{q_2}{2}\right)^2}$$

Supposons que : $tg\frac{q_2}{2}=\alpha \quad \Rightarrow \quad \frac{q_2}{2} = tg(\alpha)^{-1}$

On remplace le changement de variable dans l'équation (VI.5), on obtient :

$$A.(1-\alpha^2)+B.(2.\alpha)=C(1+\alpha^2)$$

Apres simplification, on trouve une équation d'ordre 2 avec l''inconnue α :

$$(A+C).\alpha^2-2.B.\alpha-(A-C)=0$$

Pour cherche les solutions de cette équation, on doit calculer le delta :

$$\Delta= 4.(A^2+B^2-C^2) \qquad \text{et} \qquad \sqrt{\Delta}= 2.\sqrt{A^2+B^2-C^2}$$

Les deux solutions sont :

$$\alpha_1 = \frac{-B+\sqrt{A^2+B^2-C^2}}{A+C} \qquad \text{et} \qquad \alpha_2 = \frac{B+\sqrt{A^2+B^2-C^2}}{A+C}$$

Alors $q_2 = 2.tg(\alpha_1)^{-1}$ ou bien $q_2 = 2.tg(\alpha_2)^{-1}$

On déduit maintenant q_3, à partir de système (VI.4) :

$$\begin{cases} S(q_2+q_3)=(A-d_3.Cq_2)/r_4 \\ C(q_2+q_3)=(-B+d_3.Sq_2)/r_4 \end{cases}$$

Donc

$$q_3 = Atang2\big(S(q_2+q_3), C(q_2+q_3)\big)-q_2$$

Après positionnement de l'organe terminal, on calcul son orientation par q_4 et q_5 . La surface palpée étant un plan, et selon sa disposition (figure VI.1), son équation sécrit sous la forme :

$$0.10x+0.25y+z=0.40.$$

La normale de ce plan est $\vec{n}=\begin{cases} n_1=0.09651. \\ n_2=0.24127. \\ n_3=0.96510. \end{cases}$

D'après le shéma cinématique du robot (figure VI.3) , il faut que l'axe $\overrightarrow{X_e}=-\vec{n}$. On remarque que $\overrightarrow{X_e}=-\overrightarrow{Y_5}$, donc on conclue que $\overrightarrow{Y_5^0}=\vec{n}$.

La matrice d'orientation de l'organe terminal par rapport au repère fixe est sous la forme suivante :$T_5^0=\left[\overrightarrow{X_5^0} \quad \overrightarrow{Y_5^0} \quad \overrightarrow{Z_5^0}\right]$, donc pour déduire q_4 et q_5 on prend la deuxième

colonne de la matrice d'orientation égale à la normale du plan, on obtient le système d'équations suivant :

$$\begin{cases} Cq_1.[-Cq_4.Sq_5.C(q_2+q_3)-Cq_5.S(q_2+q_3)]-Sq_1.\ Sq_4.Sq_5=n_1 & \text{(VI.6)} \\ Sq_1.[-Cq_4.Sq_5.C(q_2+q_3)-Cq_5.S(q_2+q_3)]+Cq_1.\ Sq_4.Sq_5=n_2 & \text{(VI.7)} \\ -Cq_4.Sq_5.S(q_2+q_3)+Cq_5.C(q_2+q_3)=n_3 & \text{(VI.8)} \end{cases}$$

En multipliant l'équation (VI.6) par Sq_1 et la relation (VI.7) par Cq_1 :

$$\begin{cases} Sq_1.[Cq_1.[Cq_4.Sq_5.C(q_2+q_3)+Cq_5.S(q_2+q_3)]+Sq_1.\ Sq_4.Sq_5]=-n_1.Sq_1 \\ Cq_1.[Sq_1.[-Cq_4.Sq_5.C(q_2+q_3)-Cq_5.S(q_2+q_3)]+Cq_1.\ Sq_4.Sq_5]=n_2.Cq_1 \end{cases}$$

$$\begin{cases} Sq_1.Cq_1.[Cq_4.Sq_5.C(q_2+q_3)+Cq_5.S(q_2+q_3)]+{Sq_1}^2.Sq_4.Sq_5=-n_1.Sq_1 \\ Cq_1.Sq_1.[-Cq_4.Sq_5.C(q_2+q_3)-Cq_5.S(q_2+q_3)]+{Cq_1}^2.\ Sq_4.Sq_5=n_2.Cq_1 \end{cases}$$

$${Sq_1}^2.Sq_4.Sq_5+{Cq_1}^2.\ Sq_4.Sq_5=n_2.Cq_1-n_1.Sq_1$$

$$({Cq_1}^2+{Sq_1}^2).Sq_4.Sq_5=n_2.Cq_1-n_1.Sq_1$$

$$Sq_5=\frac{n_2.Cq_1-n_1.Sq_1}{Sq_4} \qquad \text{(VI.9)}$$

En multipliant l'équation (VI.6) par Cq_1 et la relation (VI.7) par Sq_1, on obtient :

$$\begin{cases} Cq_1.[Cq_1.[-Cq_4.Sq_5.C(q_2+q_3)-Cq_5.S(q_2+q_3)]-Sq_1.\ Sq_4.Sq_5]=n_1.Cq_1 \\ Sq_1.[Sq_1.[-Cq_4.Sq_5.C(q_2+q_3)-Cq_5.S(q_2+q_3)]+Cq_1.\ Sq_4.Sq_5]=n_2.Sq_1 \end{cases}$$

$$\begin{cases} {Cq_1}^2.[-Cq_4.Sq_5.C(q_2+q_3)-Cq_5.S(q_2+q_3)]-Cq_1.Sq_1.Sq_4.Sq_5=n_1.Cq_1 \\ {Sq_1}^2.[-Cq_4.Sq_5.C(q_2+q_3)-Cq_5.S(q_2+q_3)]+Sq_1.Cq_1.Sq_4.Sq_5=n_2.Sq_1 \end{cases}$$

$$({Cq_1}^2+{Sq_1}^2)[Cq_4.Sq_5.C(q_2+q_3)+Cq_5.S(q_2+q_3)]=-n_1.Cq_1-n_2.Sq_1$$

On note que :

$$Cq_4.Sq_5.C(q_2+q_3)+Cq_5.S(q_2+q_3)=-n_1.Cq_1-n_2.Sq_1=E \qquad \text{(VI.10)}$$

En multipliant l'équation (VI.8) par $S(q_2+q_3)$ et l'équation (VI.10) par $C(q_2+q_3)$, on trouve le système d'équation :

$$\begin{cases} -Cq_4.Sq_5.S(q_2+q_3)^2 + Cq_5.C(q_2+q_3).S(q_2+q_3) = n_3.S(q_2+q_3) \\ -Cq_4.Sq_5.C(q_2+q_3)^2 - Cq_5.S(q_2+q_3).C(q_2+q_3) = -E.C(q_2+q_3) \end{cases}$$

On somme ces deux équations, après simplification on trouve :

$$Cq_4.Sq_5 = E.C(q_2+q_3) - n_3.S(q_2+q_3) \qquad \text{(VI.11)}$$

On remplace la valeur de Sq_5 par l'équation (VI.9) :

$$Cq_4.\frac{n_2.Cq_1 - n_1.Sq_1}{Sq_4} = E.C(q_2+q_3) - n_3.S(q_2+q_3)$$

Alors $$\frac{Sq_4}{Cq_4} = \frac{n_2.Cq_1 - n_1.Sq_1}{E.C(q_2+q_3) - n_3.S(q_2+q_3)} = tg(q_4)$$

On peut déterminer q_4 par l'inverse de la fonction tang (tg^{-1}). Il nous reste à déterminer q_5 par l'équation (VI.8) :

$$Cq_4.Sq_5.S(q_2+q_3) = -n_3 + Cq_5.C(q_2+q_3)$$

$$Cq_4 = \frac{-n_3 + Cq_5.C(q_2+q_3)}{Sq_5.S(q_2+q_3)}$$

On remplace ce résultat dans l'équation (VI.10) :

$$\frac{-n_3 + Cq_5.C(q_2+q_3)}{Sq_5.S(q_2+q_3)}.Sq_5.C(q_2+q_3) + Cq_5.S(q_2+q_3) = E$$

Apres simplification, on obtient :

$$Cq_5 = E.S(q_2+q_3) + n_3$$

L'équation (VI.9) donne :

$$Sq_5 = \frac{n_2.Cq_1 - n_1.Sq_1}{Sq_4}$$

Finalement $$q_5 = Atang2(Sq_5, Cq_5)$$

VI.5.6. Génération du mouvement

La génération de trajectoire permet de définir les points intermédiaires que l'organe terminal doit suivre afin qu'il puisse arriver à exécuter la tâche.

Nous avons exposé dans le chapitre précédent deux méthodes de génération de mouvement (génération de mouvements articulaire et opérationnelle) prenant en compte des points intermédiaires. Ces points permettent de définir une trajectoire évitant les collisions entre le robot et les objets de l'environnement.

Dans notre exemple, nous allons faire l'étude de la trajectoire entre deux points, en utilisant la loi de trapèze dans l'espace articulaire pour passer d'un point à un autre situé sur le plan parallèle au plan de la pièce. Cette loi est optimale en temps et assure la continuité des vitesses [Khal99].

Pour palper ou pour passer entre deux points situés sur la normale commune aux deux plans parallèles (aller-retour), on utilise la technique de l'interpolation polynomiale de degrés cinq dans l'espace opérationnelle car elle assure la continuité en vitesses et accélérations et elle vérifie le déplacement de palpeur suivant la normale.

VI.6. EXTRACTION DES POINTS

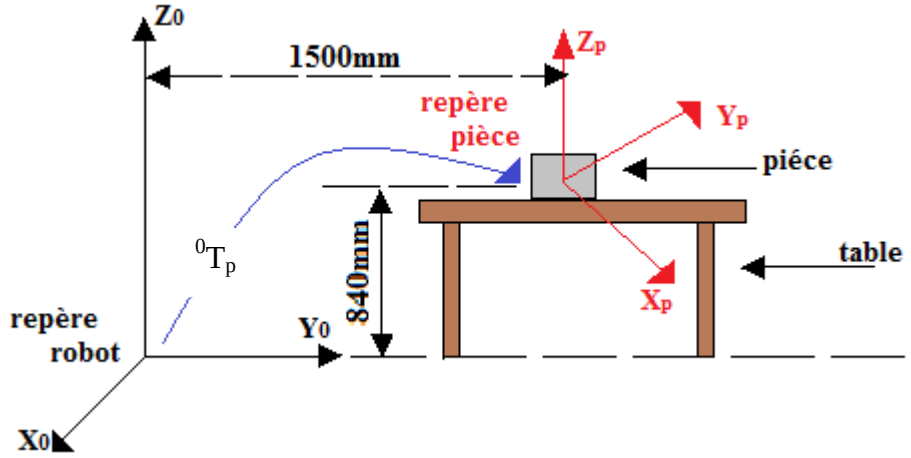


Figure VI.4 : Situation de la piéce par rapport au repère robot.

La pièce est positionnée sur la table d'une maniérée quelconque. On associe deux repères, l'un sur la base du robot et l'autre sur la pièce. En générale, le repère pièce fait trois rotations successives autour de l'axe fixe dans l'ordre X, Y, Z. La matrice de passage du repère robot au repère pièce ${}^0T_p$ s'écrit sous la forme [Boua09] :

$$T_p^0 = Rot(Z,\varphi).Rot(Y,\gamma).Rot(X,\alpha)$$

$$T_p^0 = \begin{bmatrix} C\varphi & -S\varphi & 0 \\ S\varphi & C\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix}.\begin{bmatrix} C\gamma & 0 & S\gamma \\ 0 & 1 & 0 \\ -S\gamma & 0 & C\gamma \end{bmatrix}.\begin{bmatrix} 1 & 0 & 0 \\ 0 & C\alpha & -S\alpha \\ 0 & S\alpha & C\alpha \end{bmatrix}$$

$$T_p^0 = \begin{bmatrix} C\varphi.C\gamma & C\varphi.S\gamma.S\alpha - S\varphi.C\alpha & C\varphi.S\gamma.C\alpha + S\varphi.S\alpha \\ S\varphi.C\gamma & S\varphi.S\gamma.S\alpha + C\varphi.C\alpha & S\varphi.S\gamma.C\alpha - C\varphi.S\alpha \\ -S\gamma & C\gamma.S\alpha & C\gamma.C\alpha \end{bmatrix}$$

Dans notre cas, on a choisi les axes Z des deux repères parallèles. Donc, les angles de rotation selon X et Y sont nuls ($\gamma = \alpha = 0$). On prend l'angle φ suivant Z égal à 30°. Il vient :

$$T_p^0 = \begin{bmatrix} C\varphi & -S\varphi & 0 \\ S\varphi & C\varphi & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

La distance entre repère robot et repère pièce (translation) par rapport au repère robot est (figure VI.4) : $O_0O_{P/R0} = \begin{cases} 0 \\ 1500 \\ 840 \end{cases}$ (mm).

Les coordonnées du point P dans le repère robot sont données par la relation :

$O_0P_{/R0} = O_0O_{P/R0} + O_PP_{/R0}$ avec $O_PP_{/R0} = T_0^p.O_PP_{/Rp}$

$O_PP_{/Rp}$: Coordonnées des points (9 points sur chaque plan) sur la pièce par rapport au repère pièce.

$O_PP_{/Rp}$: Coordonnées des points sur la pièce par rapport au repère robot.

Les tableaux suivants récapitulent les points par rapport aux repère-pièce (tableaux VI.7) et repère-robot (tableaux VI.8). Les coordonnées de ces points sont données dans l'espace opérationnel (X).

Points Pi du plan à palper			
Pts	X(m)	Y(m)	Z(m)
1	0.050	0.050	0.3825
2	0.050	0.200	0.3450
3	0.050	0.350	0.3075
4	0.250	0.350	0.2875
5	0.250	0.200	0.3250
6	0.250	0.050	0.3625
7	0.450	0.050	0.3425
8	0.450	0.200	0.3050
9	0.450	0.350	0.2675

Points P'i du plan // au plan palpé			
Pts	X(m)	Y(m)	Z(m)
1	0.050	0.050	0.4325
2	0.050	0.200	0.3950
3	0.050	0.350	0.3575
4	0.250	0.350	0.3375
5	0.250	0.200	0.3750
6	0.250	0.050	0.4125
7	0.450	0.050	0.3925
8	0.450	0.200	0.3550
9	0.450	0.350	0.3175

Tableaux VI.7 : Coordonnées des points sur le plan à palpé et le plan parallèle au plan palpé par rapport au repère pièce.

Points Pi du plan à palper			
Pts	X(m)	Y(m)	Z(m)
1	0.3983	1.5183	1.2225
2	0.4733	1.6482	1.1850
3	0.5483	1.7781	1.1475
4	0.7215	1.6781	1.1275
5	0.6465	1.5482	1.1650
6	0.5715	1.4183	1.2025
7	0.7447	1.3183	1.1825
8	0.8197	1.4482	1.1450
9	0.8947	1.5781	1.1075

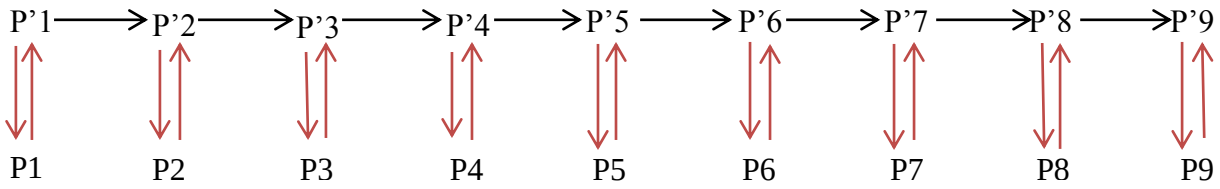
Points P'i du plan // au plan palpé			
Pts	X(m)	Y(m)	Z(m)
1	0.3983	1.5183	1.2725
2	0.4733	1.6482	12350
3	0.5483	1.7781	1.1975
4	0.7215	1.6781	1.1775
5	0.6465	1.5482	1.2150
6	0.5715	1.4183	1.2525
7	0.7447	1.3183	1.2325
8	0.8197	1.4482	1.1950
9	0.8947	1.5781	1.1575

Tableaux VI.8 : Coordonnées des points sur le plan à palpé et le plan parallèle au plan palpé par rapport au repère robot.

VI.7. ETAPES DE PLANIFICATION DE TRAJECTOIRE

VI.7.1. Lois de mouvement

- Entre les points P'1, P'2, …….. P'9 : Génération du mouvement dans l'espace articulaire avec la loi de trapèze.
- Entre P'1 et P1, P'2 et P2, ……, P'9 et P9 : Génération du mouvement dans l'espace opérationnel avec interpolation degré 5.



VI.7.2. Génération du mouvement dans l'espace articulaire

On utilise la loi de trapèze (loi Bang–Bang avec palier de vitesse). Les équations de mouvement sont données dans le chapitre V : équation (V.8). On désigne respectivement par ***Kv*** et ***Ka*** les vecteurs des vitesses et accélérations maximales (tableau VI.3).

Pour passer aux points de l'espace opérationnel (X) (tableaux VI.8) à l'espace articulaire (q), il faut utiliser le modèle géométrique inverse (MGI). Ensuite, on effectue la génération du mouvement pour déterminer les points intermédiaires entre q^i et q^f, la vitesse ($\dot{q}$) et l'accélération ($\ddot{q}$) articulaires.

On utilise par la suite le modèle cinématique direct (MCD) du premier ordre pour calculer la vitesse opérationnelle ($\dot{X}$) à partir de($\dot{q}$). L'accélération opérationnelle ($\ddot{X}$) est calculée par le modèle cinématique direct du deuxième ordre à partir de ($\ddot{q}$). Pour vérifier les

calculs, on procède de la manière inverse, c'est-à-dire, on applique le MCI du premier et deuxième ordre.

VI.7.3. Génération du mouvement dans l'espace opérationnel

Le passage d'un point P'i au point Pi (aller) et inversement (retour), s'effectue suivant une loi polynomiale de degré 5. Les relations de mouvement sont données dans le chapitre V : équation (V.6). A la durée de ce mouvement, on ajoute le temps de stabilité de palpeur pour la saisie des coordonnées des points palpés.

Les coordonnées des points sont déjà dans l'espace opérationnel (X) (tableaux VI.8), donc la génération du mouvement donne directement la position (X) les vitesses ($\dot{X}$) et l'accélération ($\ddot{X}$) dans l'espace opérationnel.

On utilise le MGI pour déterminer les positions articulaires (q). La MCI du premier et deuxième ordre permet de calculer les vitesses articulaires ($\dot{q}$) et les accélérations articulaires ($\ddot{q}$). Pour vérifier les calculs, on applique le MCD.

VI.7.4. Modélisation dynamique

Les paramètres de position, vitesse et accélération articulaires et opérationnelles du mouvement complet étant déterminés, on procède ensuite à la modélisation dynamique où l'on calcule le couple pour chaque articulation et la puissance développées durant l'exécution de la tâche.

VI.8. RESULTATS DES PROGRAMMES

VI.8.1. Trajectoire dans l'espace opérationnel

La figure VI.5 illustre la trajectoire du palpeur en 3D dans l'espace opérationnel. Le point P'1 est le point de départ de déplacement du palpeur.

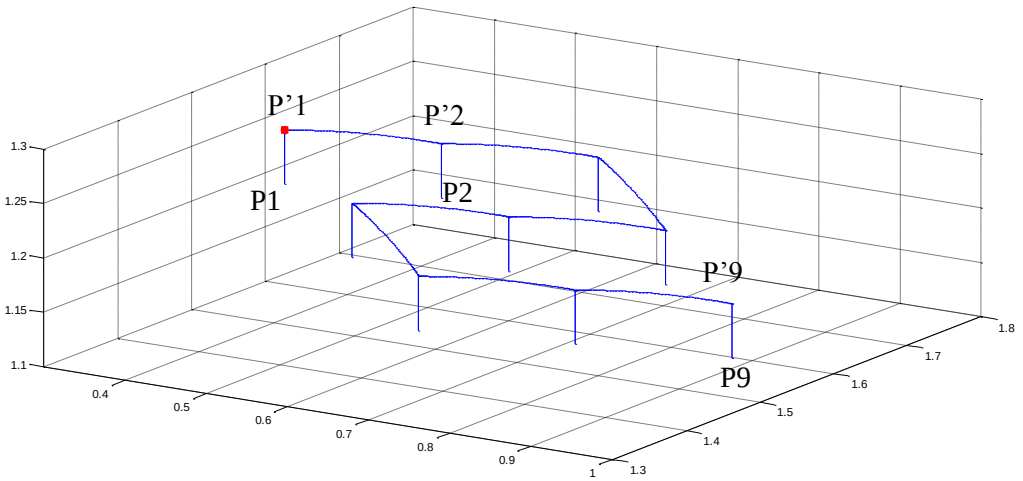


Figure VI.5 : trajectoire de palpeur dans l'espace opérationnel.

VI.8.2. Position, vitesse et accélération articulaires

Les figures VI.6, VI.9, VI.12, VI.15, VI.18 présentent les positions (coordonnées articulaires) ou encore les configurations atteintes par chaque articulation (q1 à q5) le long d'exécution de la tâche.

Les graphes VI.7, VI.10, VI.13, VI.16, VI.19 donnent la dérivé première des positions, c-à-d, la variation des vitesses pour les articulations du robot.

Les autres courbes illustrent la deuxième dérivée des positions (accélération). Le temps de palpage des neuf points est d'environ 3s.

Les figures VI.18, VI.19 et VI.20 illustrent la variation de position, de vitesse et d'accélération de la cinquième articulation par rapport au bâti (repère R_0). L'accélération articulaire prend une valeur max (k_{a5} =25rad/s^2) pour le passage entre deux points situés dans des plans différents (aller-retour) avec la loi d'interpolation de degré 5.

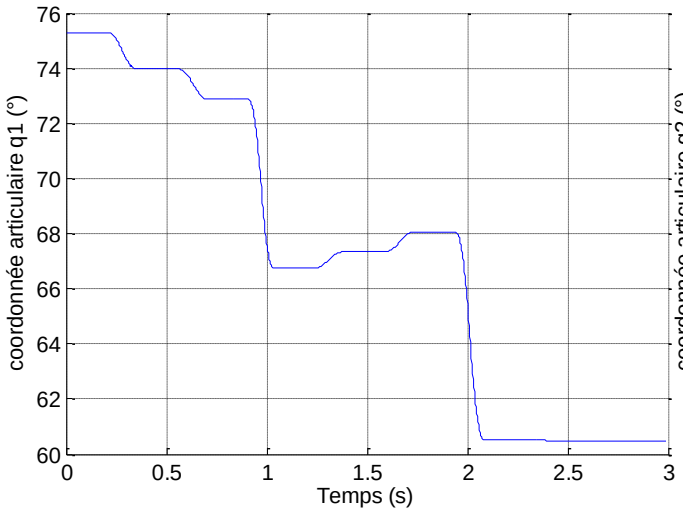


Figure VI.6 : Position (**q1**)

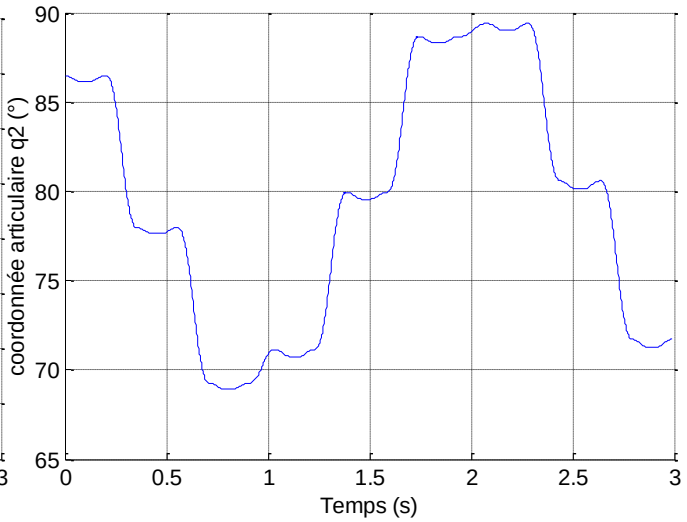


Figure VI.9 : Position (**q2**)

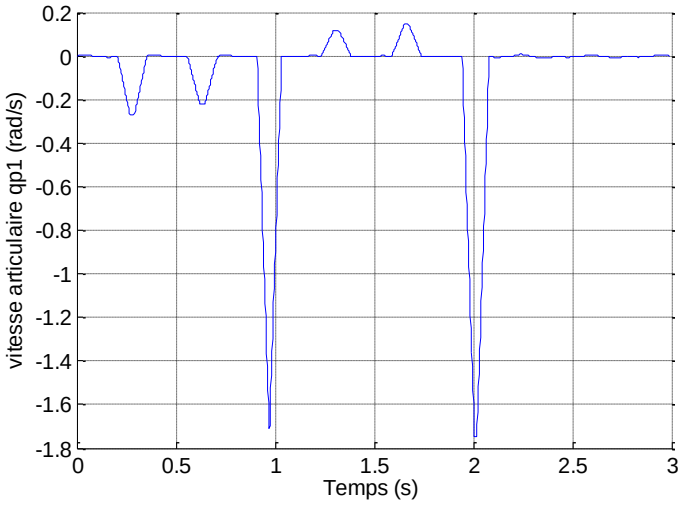


Figure VI.7 : Vitesse ($\dot{q}1$)

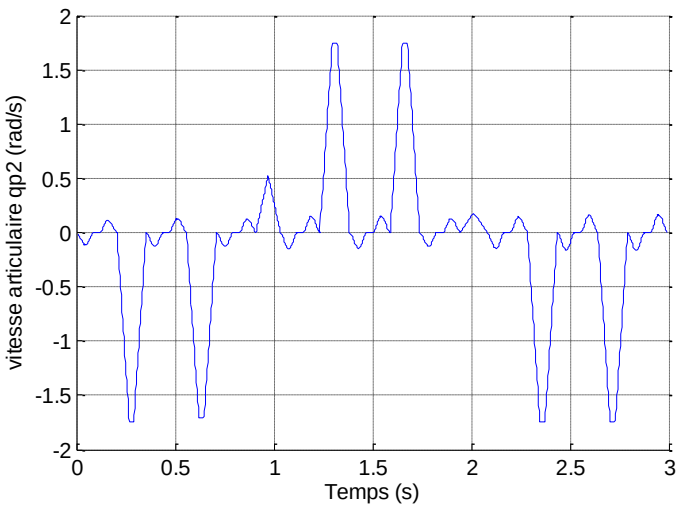


Figure VI.10 : Vitesse ($\dot{q}2$)

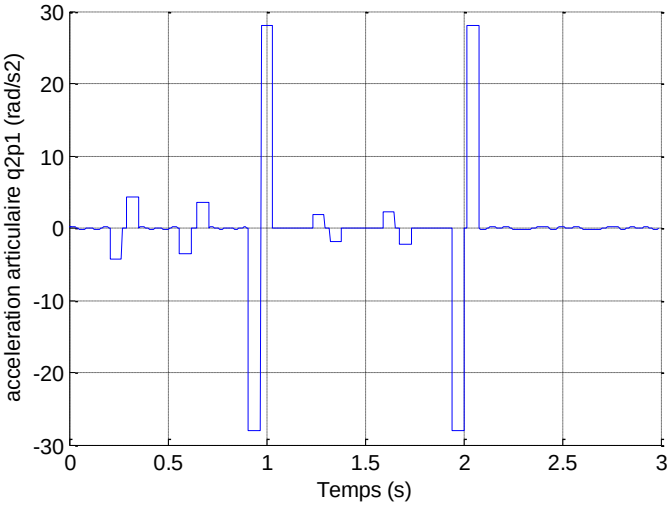


Figure VI.8 : Accélération ($\ddot{q}1$)

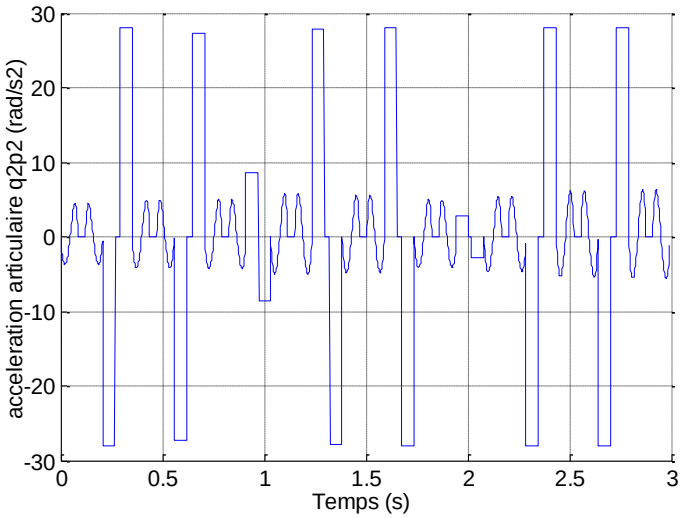


Figure VI.11 : Accélération ($\ddot{q}2$)

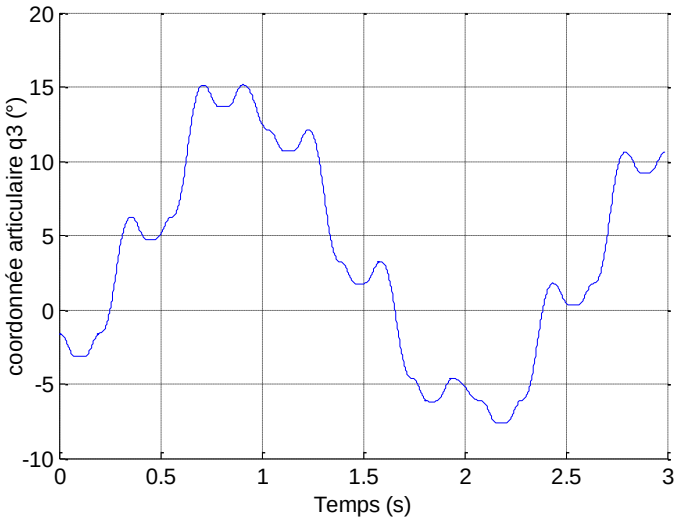


Figure VI.12 : Position (**q3**)

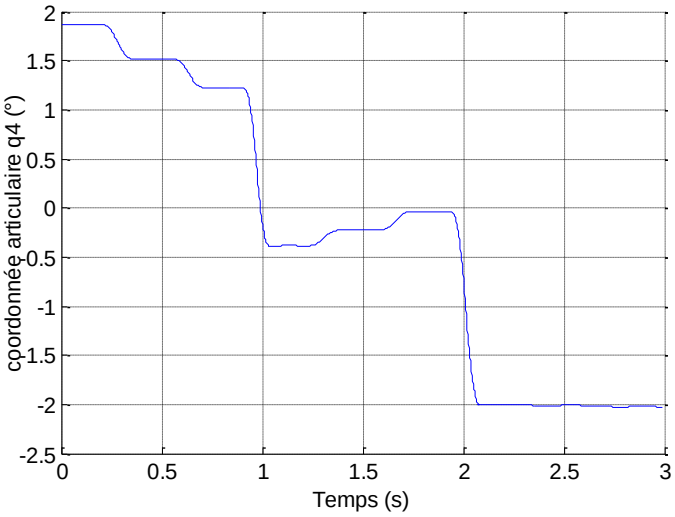


Figure VI.15 : Position (**q4**)

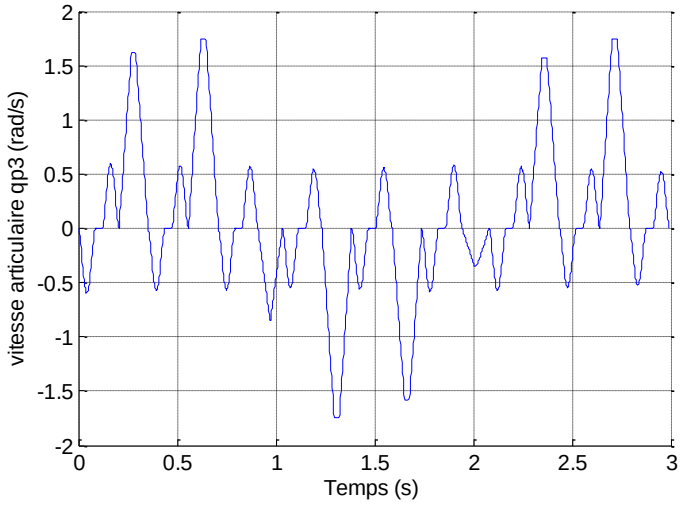


Figure VI.13 : Vitesse ($\dot{q}3$)

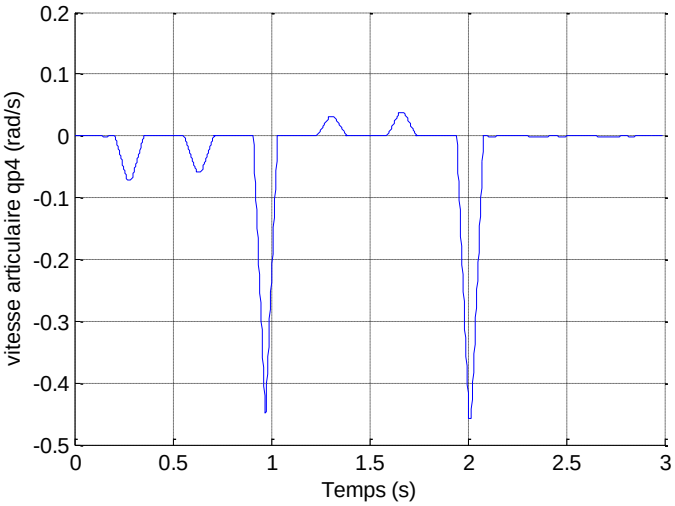


Figure VI.16 : Vitesse ($\dot{q}4$)

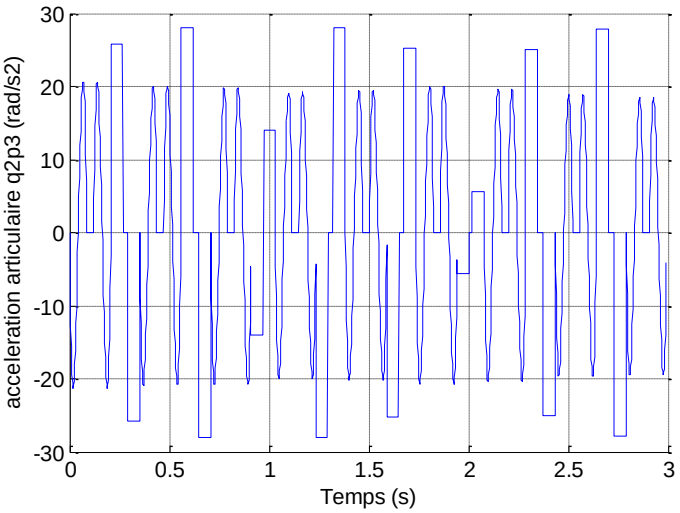


Figure VI.14 : Accélération ($\ddot{q}3$)

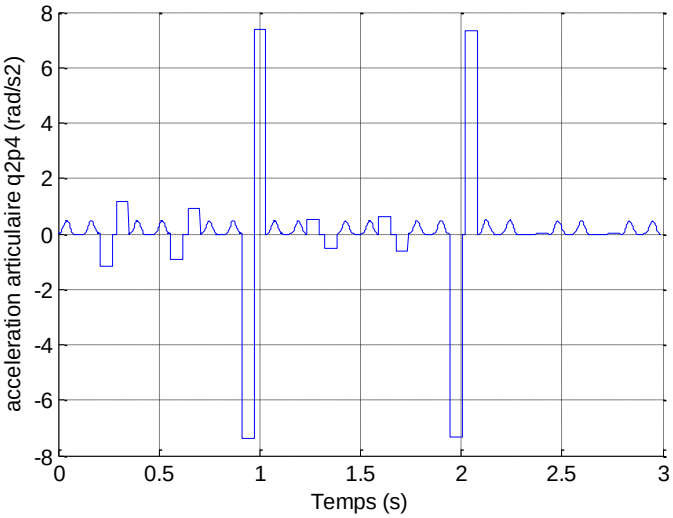


Figure VI.17 : Accélération ($\ddot{q}4$)

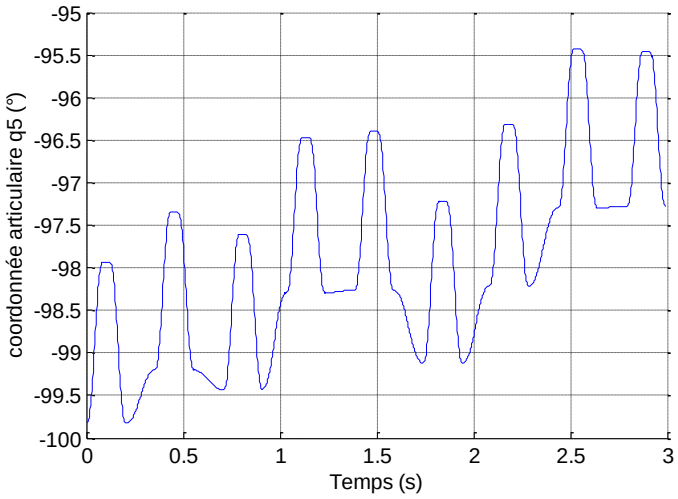


Figure VI.18 : Position (**q5**)

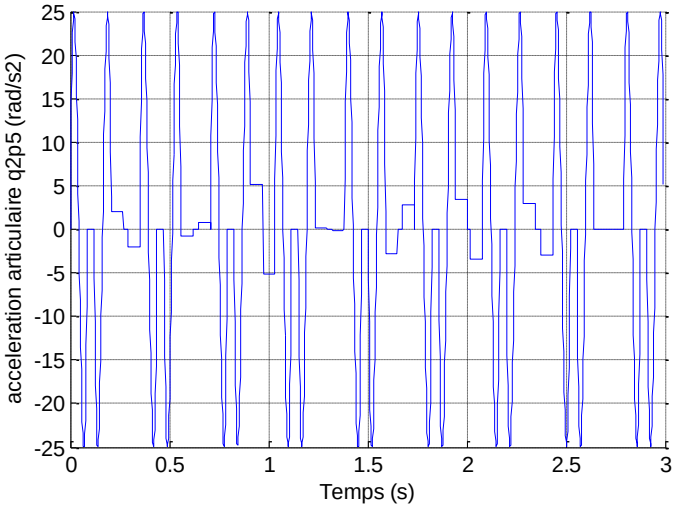


Figure VI.20 : Accélération ($\ddot{q}5$)

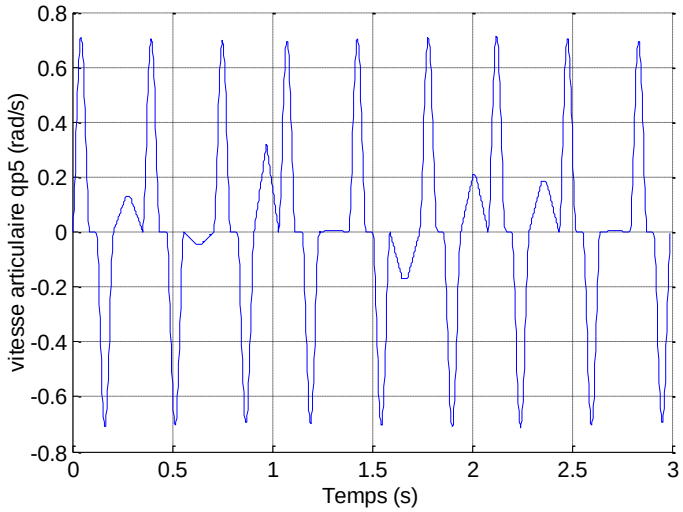


Figure VI.19 : Vitesse ($\dot{q}5$)

VI.8.3. Couples et puissance

Par définition, la puissance est le produit de la vitesse angulaire et du couple moteur. La puissance du robot est la somme des puissances des articulations. Les figures VI.21 à VI.35 représentent respectivement les graphes des vitesses, couples et le leurs produit (puissance) pour chaque articulation (5 articulations). La figure VI.36 donne la puissance globale du robot.

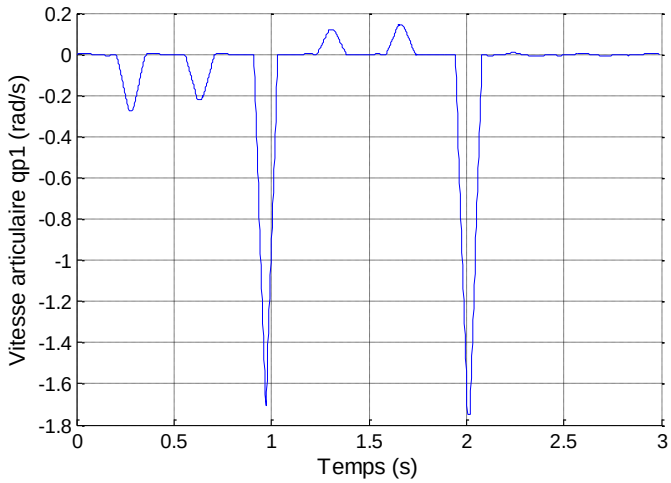


Figure VI.21 : Vitesse ($\dot{q}1$)

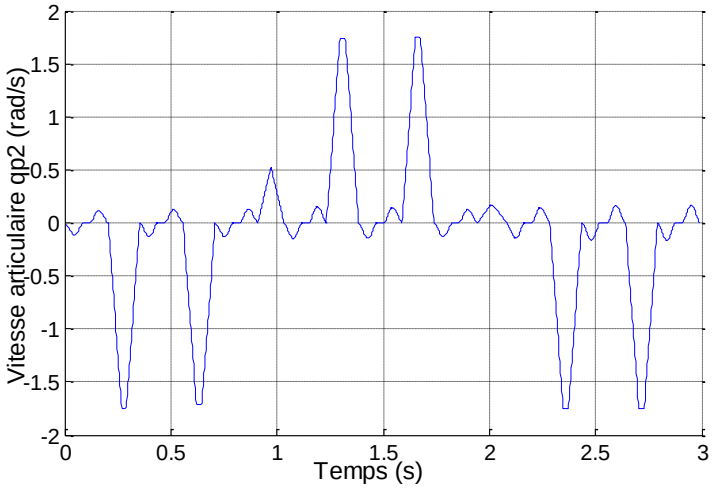


Figure VI.24 : Vitesse ($\dot{q}2$)

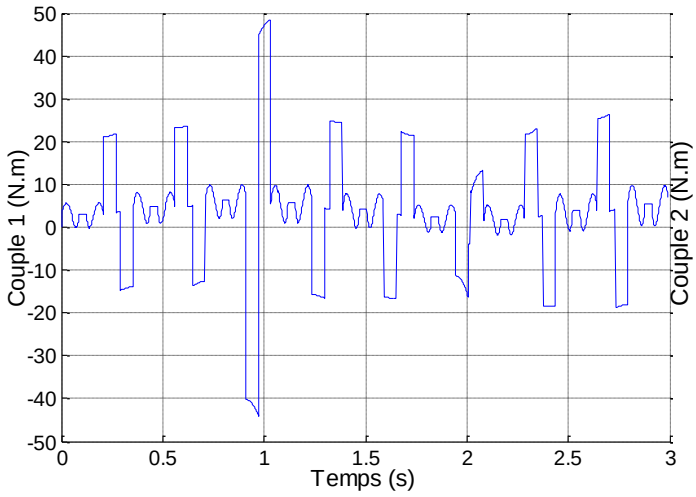


Figure VI.22 : Couple 1

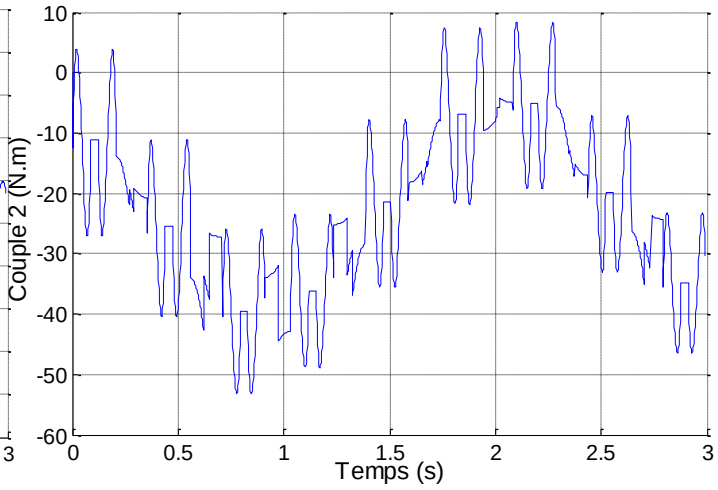


Figure VI.25 : Couple 2

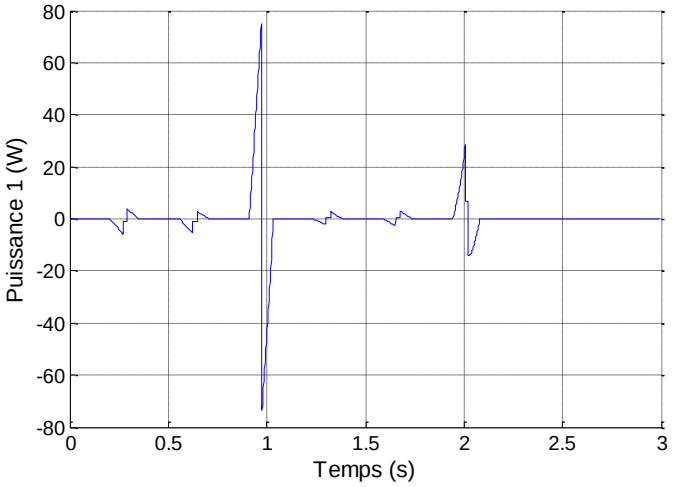


Figure VI.23 : Puissance 1

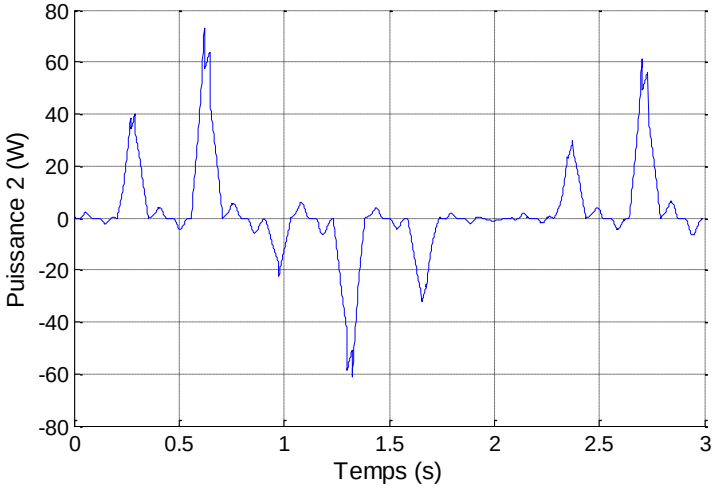


Figure VI.26 : Puissance 2

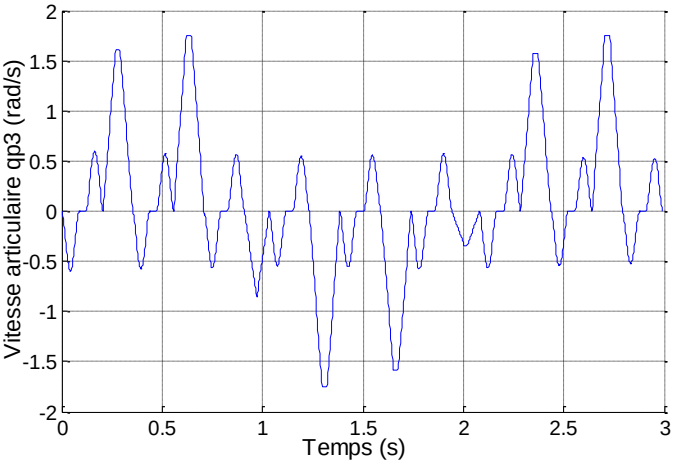


Figure VI.27 : Vitesse ($\dot{q}3$)

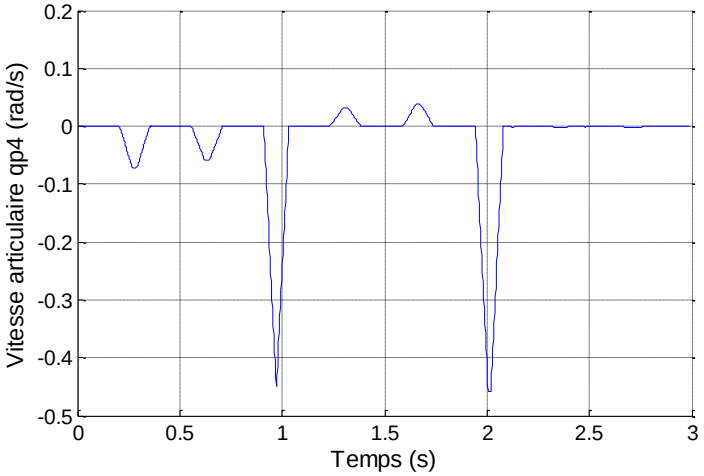


Figure VI.30 : Vitesse ($\dot{q}4$)

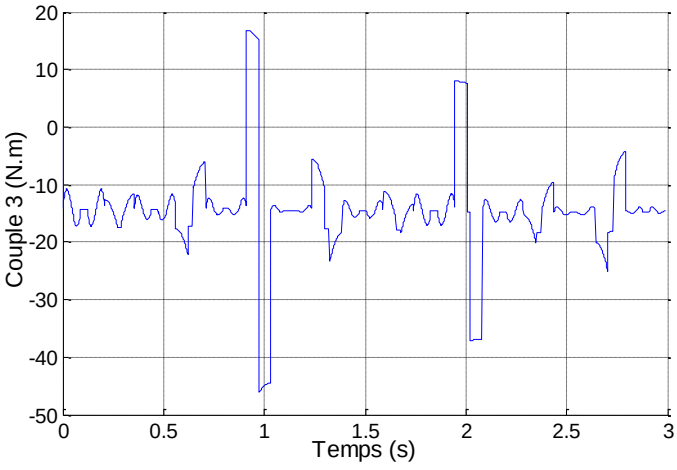


Figure VI.28 : Couple 3

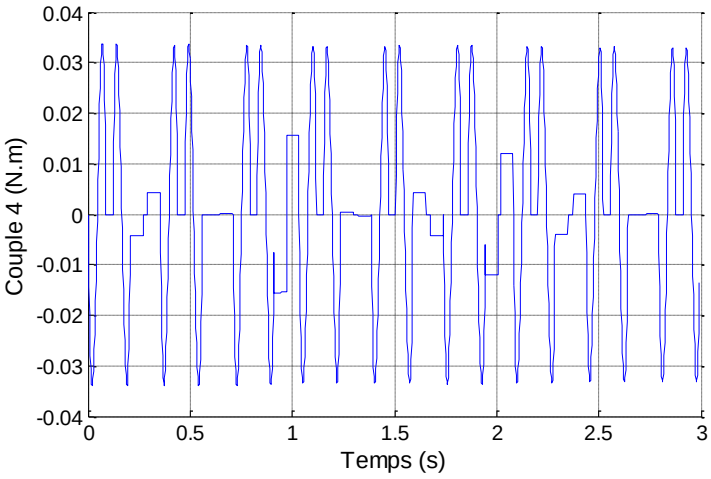


Figure VI.31 : Couple 4

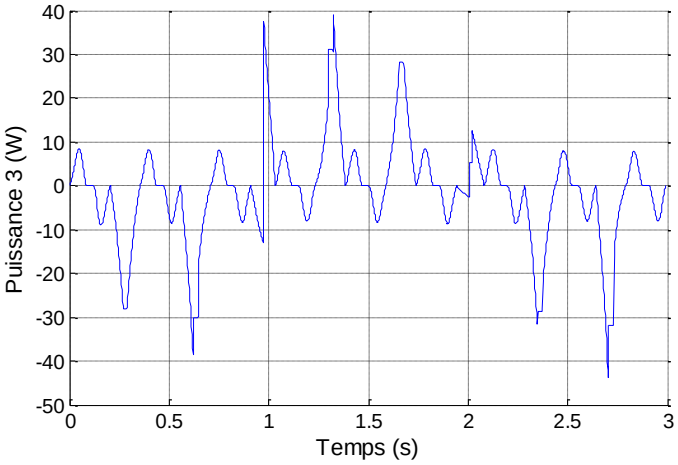


Figure VI.29 : Puissance 3

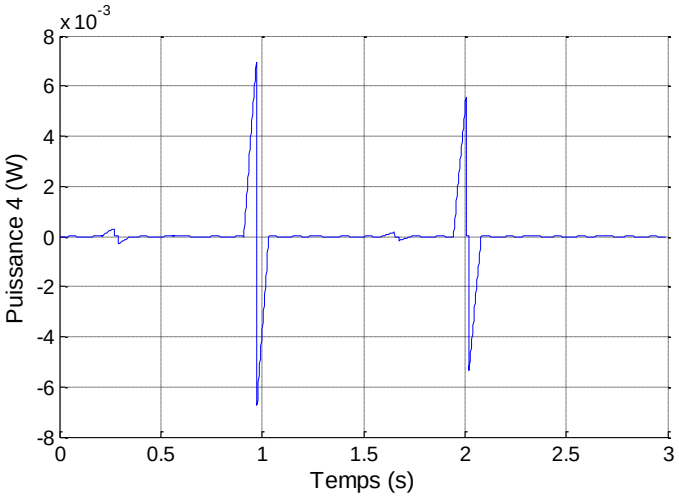


Figure VI.32 : Puissance 4

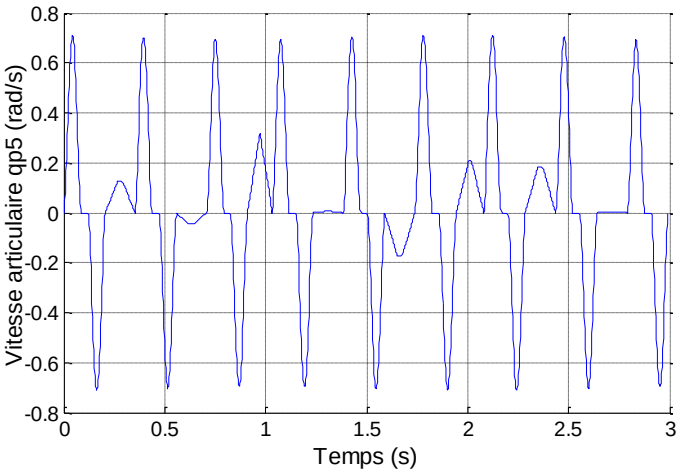


Figure VI.33 : Vitesse ($\dot{q}5$)

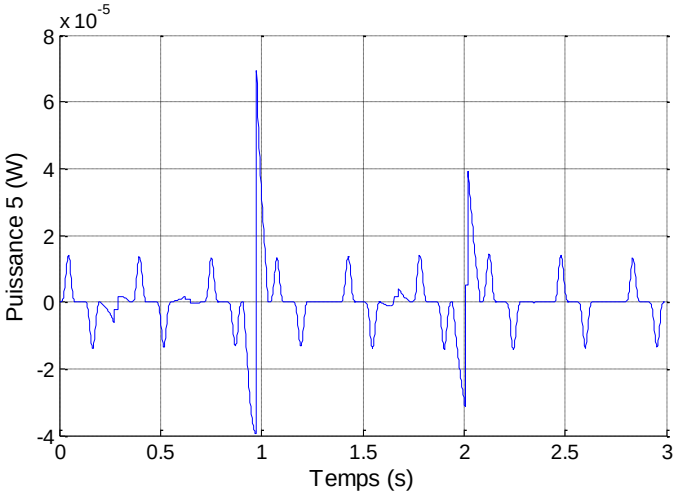


Figure VI.35 : Puissance 5

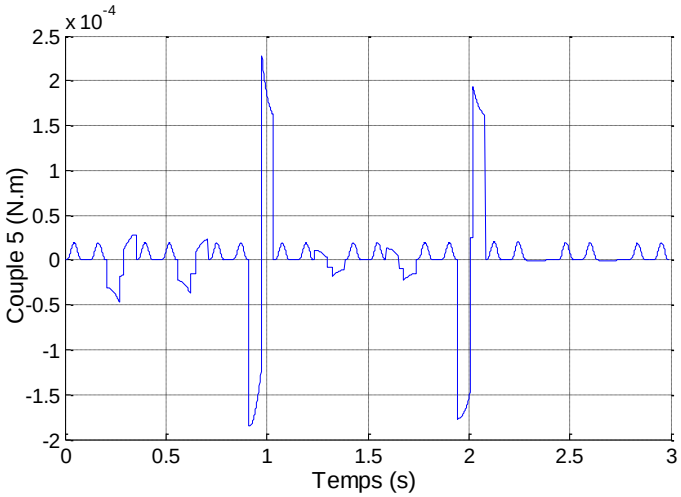


Figure VI.34 : Couple 5

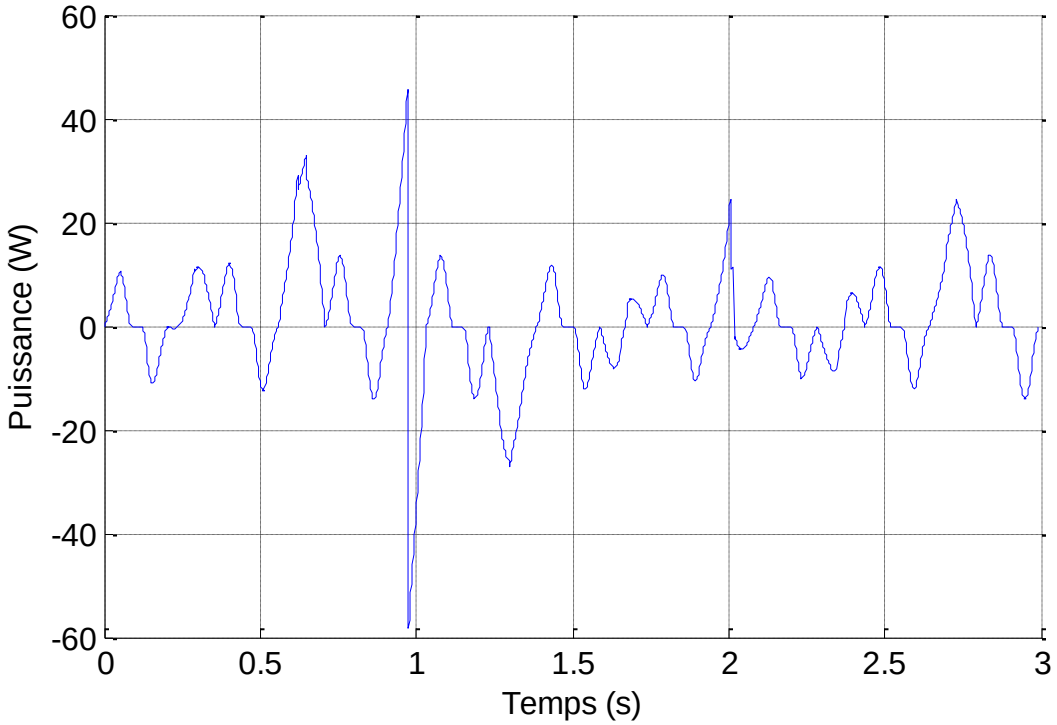


Figure VI.36 : Puissance globale du robot.

Remarques

Pour s'assurer de la validité de nos calculs à l'aide de ces programmes, nous avons comparé les résultats de nos calculs pour un exemple quelconque avec le programme établi par le laboratoire de robotique de l'EMP pour la génération de mouvement dans l'espace articulaire (voir annexe A). Nos calculs donnent des résultats pratiquement identiques à ceux de ce laboratoire; l'écart relatif est négligeable.

VI.9. SIMULATION

La simulation permet de concevoir et de tester les robots dans une variété d'environnements dans un temps court et à très faible coût. Une bonne simulation peut parfois aider facilement à résoudre certains problèmes de fonctionnement du système. La figure suivante illustre la simulation de mesure.

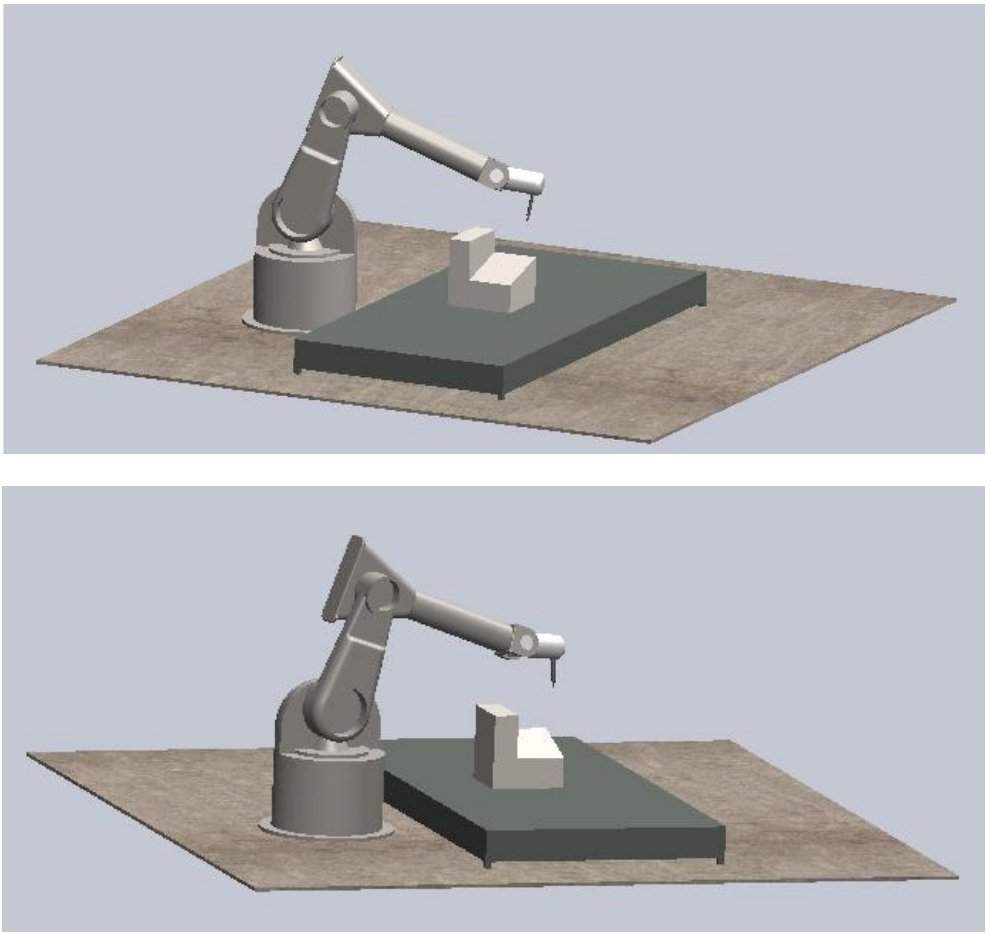


Figure VI.37 : Simulation de robot.

VI.10. CONCLUSION

Nous avons présenté dans ce chapitre une partie d'une étude d'un robot ayant cinq articulations appliqué à la mesure 3D. Les résultats sont obtenus à l'aide de programmes pour modéliser et générer la trajectoire dans le cadre de l'application étudiée.

Conclusion générale

CONCLUSION GENERALE

Le travail présenté dans ce mémoire est l'étude d'une configuration générale d'un robot à structure ouverte simple (robot série) à 6 degré de liberté en vue d'un calcul automatisé. Il consiste à proposer une méthode générale permettant une programmation des modèles mathématiques géométriques, cinématique et dynamique, et la génération de la trajectoire du mouvement. Ceci offre la possibilité de minimiser l'intervention des opérateurs dans les diverses étapes de planification de trajectoire.

En premier lieu, un rappel de terminologie fondamentale de robotique est donnée (définition de la robotique, structures articulées, articulations, degré de liberté, degré de mobilité, types de poignets, types de porteurs, etc…) nécessaire à la compréhension des notions de robotique.

En second lieu, on aborde les outils de calculs tels les transformations homogènes et les descriptions des orientations (angle d'Euler et de Cardan) qui aident à la modélisation géométrique, cinématique et dynamique.

La modélisation géométrique est traitée en considérant la convention de Denavit-Hertenberg Modifiée. Le modèle géométrique inverse est résolu par la méthode numérique car il est plus compliqué à le généraliser du fait qu'il n'existe pas de méthode analytique générale et aussi de sa forme non linéaire.

Pour planifier la trajectoire du robot, il est nécessaire d'établir la modélisation cinématique directe ou inverse. Celle-ci est établie en exprimant les relations de cinématique du premier et deuxième ordre et en employant la matrice jacobienne. Dans le cas où cette matrice n'est pas carrée, la modélisation cinématique inverse est plus difficile à réaliser. On fait alors appel à la notion de pseudo- inverse à l'aide de l'algorithme de Greville.

La commande des robots nécessite la modélisation dynamique. Elle permet de calculer la puissance (ou le couple) du robot à l'aide de deux formalismes, celui de Lagrange et de Newton-Euler. Le deuxième formalisme est plus utilisé en programmation à cause de son caractère itératif qui réduit énormément le temps de calcul relativement au premier.

La génération de mouvement est une phase importante. Elle décrit la trajectoire selon la tâche à accomplir dans l'espace articulaire ou opérationnel. Dans notre travail, nous avons fait les calculs pour les techniques de génération de mouvement couramment utilisées en robotique : modes d'interpolation (linéaire, degrés 3 et 5), loi Bang-Bang avec ou sans palier de vitesse (dans le premier cas, la loi est dite trapèze). Le mouvement du robot est simulé à l'aide d'un système CAO pour voir la trajectoire parcourue par l'organe terminal dans un milieu virtuel.

Des programmes sous Matlab sont établis pour calculer les différentes modélisations (géométrique, cinématique et dynamique) ainsi que la génération de mouvement.

La dernière étape est une application des programmes établis pour la mesure 3D à l'aide d'un bras manipulateur. Cependant, il est à noter que pour la modélisation géométrique inverse, la méthode analytique qui est adoptée. La tâche assignée est un palpage d'un plan suivant sa normale ; les points à palper étant définis par rapport à la base fixe du robot. Le nombre de degrés de liberté retenu du robot est de 5. Le schéma cinématique associé est établi en respectant le principe de Denavit-Hartenberg Modifié. La trajectoire du palpeur est décomposée en une loi trapèze dans l'espace articulaire pour passer d'un point à un autre situé sur le plan parallèle au plan de la pièce, et en une loi d'interpolation polynomiale de degrés 5 dans l'espace opérationnelle pour passer entre deux points situés sur la normale commune aux deux plans parallèles (aller-retour). La simulation est faite à l'aide du logiciel Solide Works.

En perspectives, le travail reste à compléter par de nombreux points tels que la commande, l'étude de stabilité, l'application du robot étudié aux mesures de cylindre, sphère, etc.. et aussi aux surfaces gauches. Il serait aussi intéressant d'aborder l'automatisation de calculs pour les structures fermées.

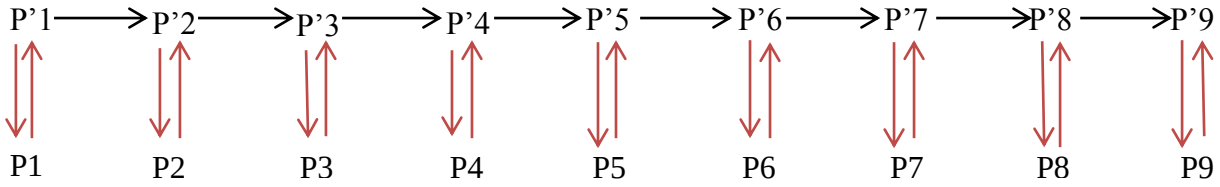
Annexes

DETAILS DE L'APPLICATION ET VALIDATION

I. Détails de l'application au bras de mesure 3D

I.1. Etapes de planification de trajectoire

- Entre les points P'1, P'2, …….. P'9 : Génération du mouvement dans l'espace articulaire avec la loi de trapèze.
- Entre P'1 et P1, P'2 et P2, ……, P'9 et P9 : Génération du mouvement dans l'espace opérationnel avec interpolation degré 5.



I.2. Génération du mouvement dans l'espace articulaire

P'1 ⟶ P'2 ⟶ P'3 ⟶ … ⟶ P'9

I.2.1. Modèle Géométrique Inverse :

Le modèle géométrique inverse (MGI) permet le passage entre les points dans l'espace opérationnel (X) (tableaux 1) à l'espace articulaire (q) (tableaux 2).

Points P'i du plan // au plan palpé			
Pts	X(m)	Y(m)	Z(m)
1	0.3983	1.5183	1.2725
2	0.4733	1.6482	12350
3	0.5483	1.7781	1.1975
4	0.7215	1.6781	1.1775
5	0.6465	1.5482	1.2150
6	0.5715	1.4183	1.2525
7	0.7447	1.3183	1.2325
8	0.8197	1.4482	1.1950
9	0.8947	1.5781	1.1575

Tableau 1 : Coordonnées des points sur le plan parallèle au plan palpé par rapport au repère robot dans l'espace opérationnel.

Points P'_i, i :	q1	q2	q3	q4	q5
1	7.5300622e+001	8.6454360e+001	-1.5858521e+000	1.8695438e+000	-9.9825592e+001
2	7.3977965e+001	7.7956864e+001	6.2499197e+000	1.5198251e+000	-9.9203436e+001
3	7.2862224e+001	6.9265693e+001	1.5141096e+001	1.2278874e+000	-9.9429901e+001
4	6.6734614e+001	7.1136464e+001	1.2087946e+001	-3.8460870e-001	-9.8293011e+001
5	6.7335344e+001	7.9947839e+001	3.2438710e+000	-2.2678370e-001	-9.8263511e+001
6	6.8052965e+001	8.8660045e+001	-4.6144154e+000	-3.8341430e-002	-9.9118641e+001
7	6.0537795e+001	8.9391589e+001	-6.1158172e+000	-2.0068593e+000	-9.8215790e+001
8	6.0489319e+001	8.0581360e+001	1.7760261e+000	-2.0150777e+000	-9.7296751e+001
9	6.0448860e+001	7.1717099e+001	1.0621187e+001	-2.0255005e+000	-9.7276259e+001

Tableau 2 : Résultats du Modèle Géométrique Inverse (Coordonnées articulaires).

I.2.2. Génération du mouvement dans l'espace articulaire avec la loi de trapèze (loi Bang–Bang avec palier de vitesse)

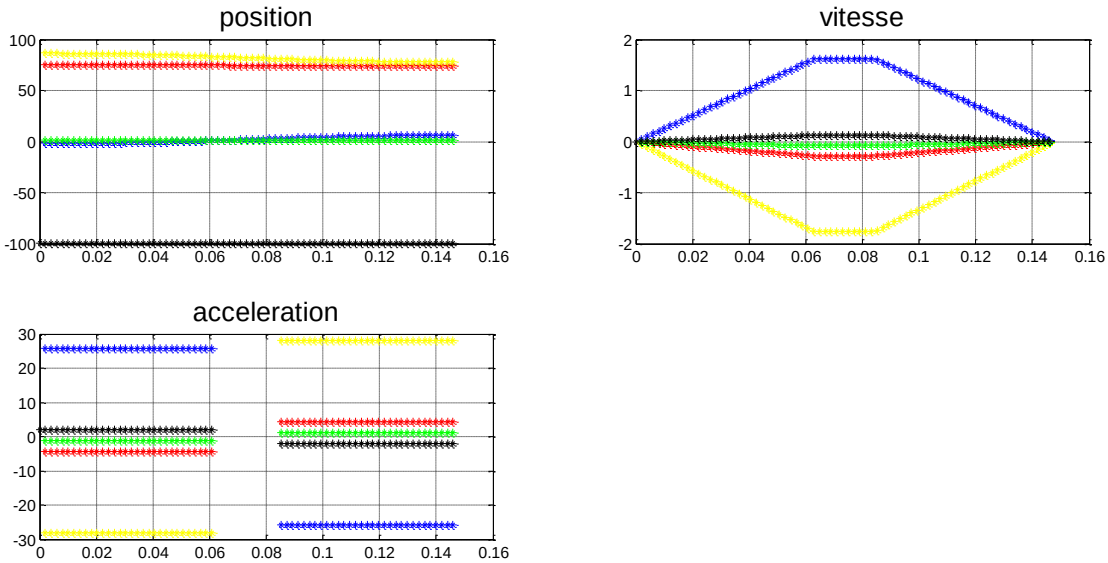


Figure 1 : Loi de trapèze entre P'1 et P'2.

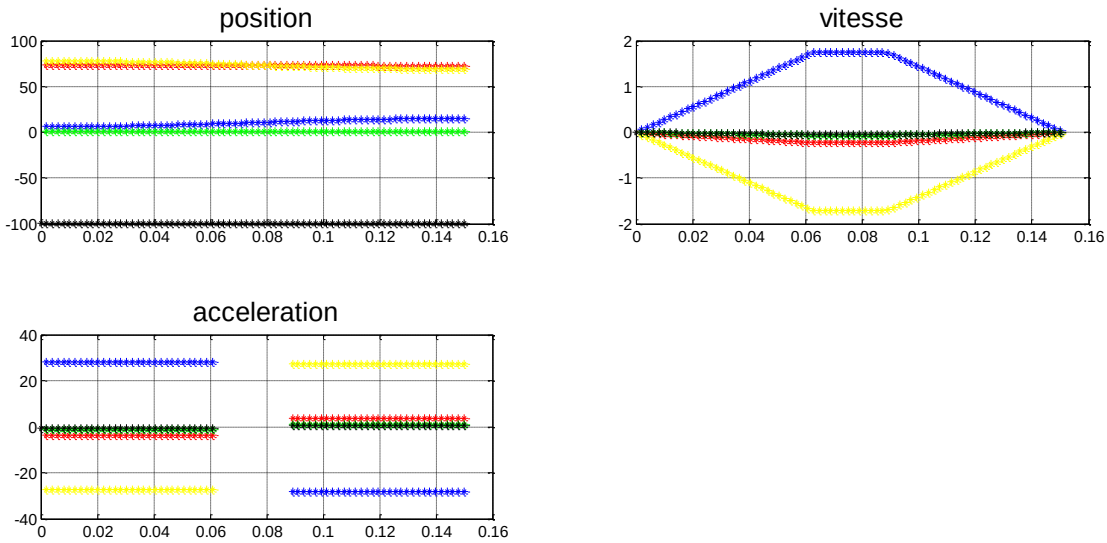


Figure 2 : Loi de trapèze entre P'2 et P'3

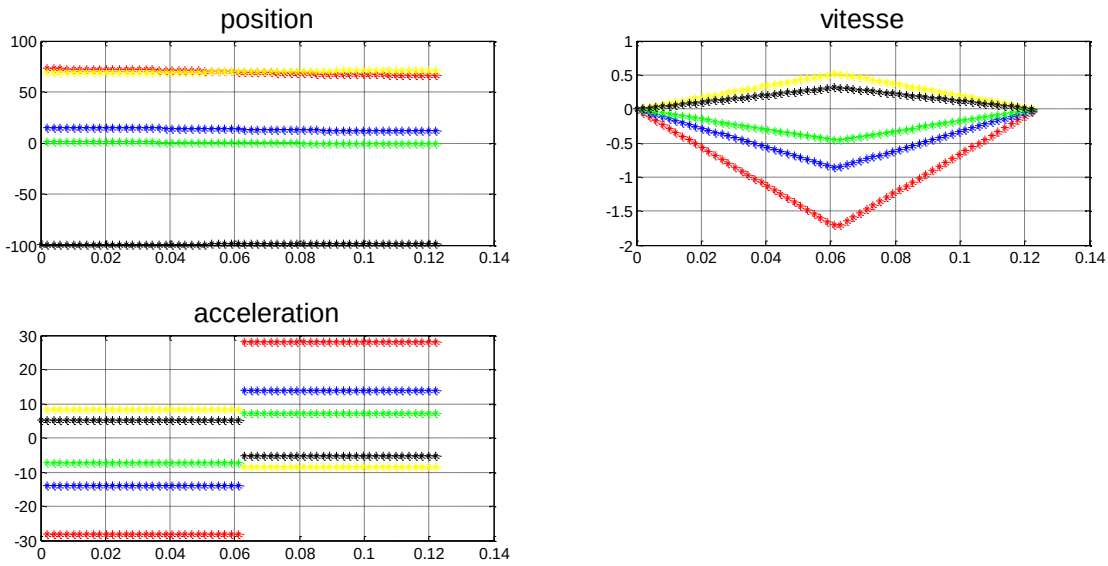


Figure 3 : Loi de trapèze entre P'3 et P'4

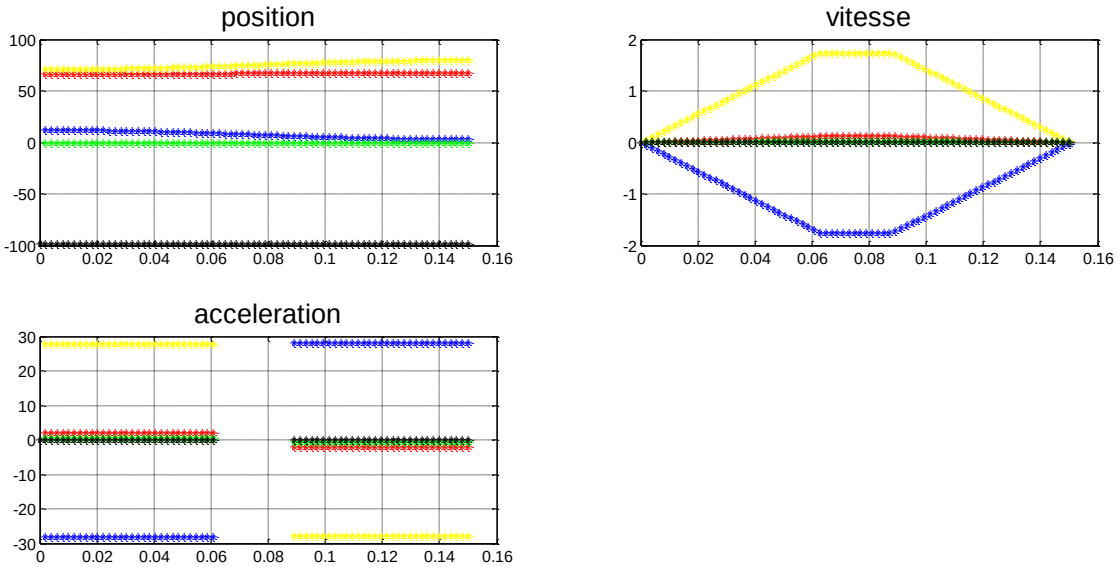


Figure 4 : Loi de trapèze entre P'4 et P'5

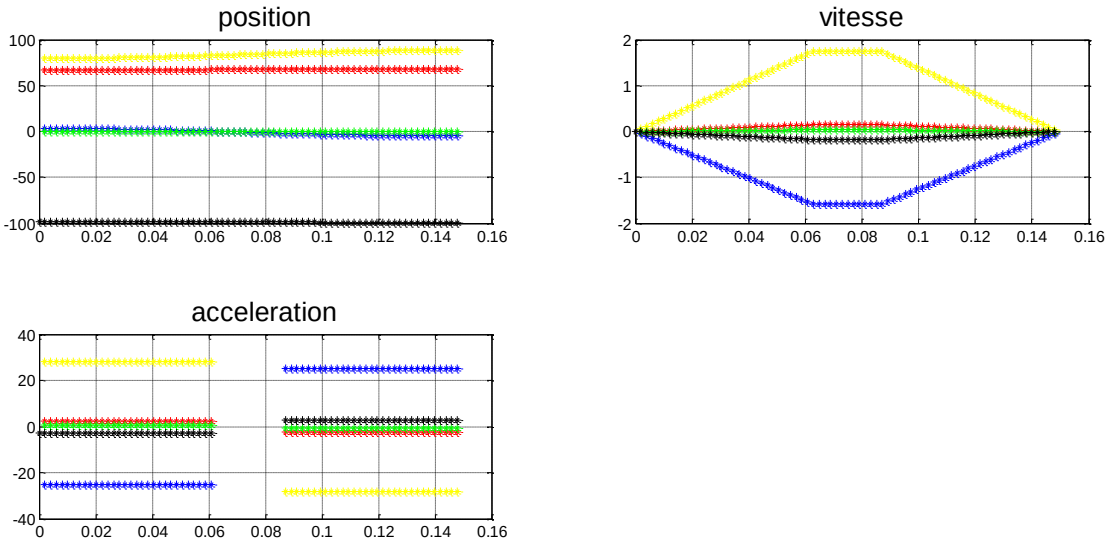


Figure 5 : Loi de trapèze entre P'5 et P'6

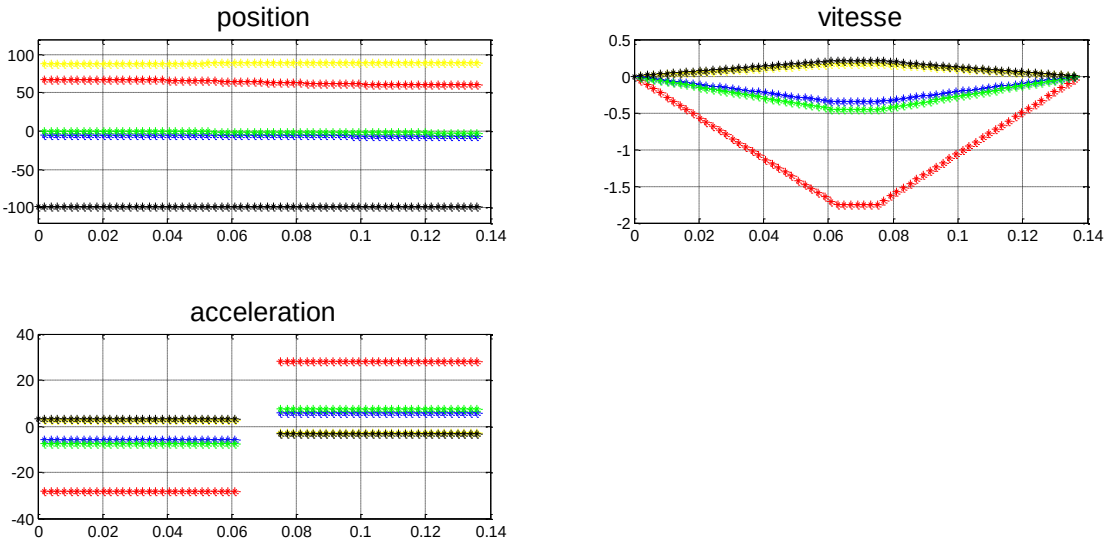


Figure 6 : Loi de trapèze entre P'6 et P'7

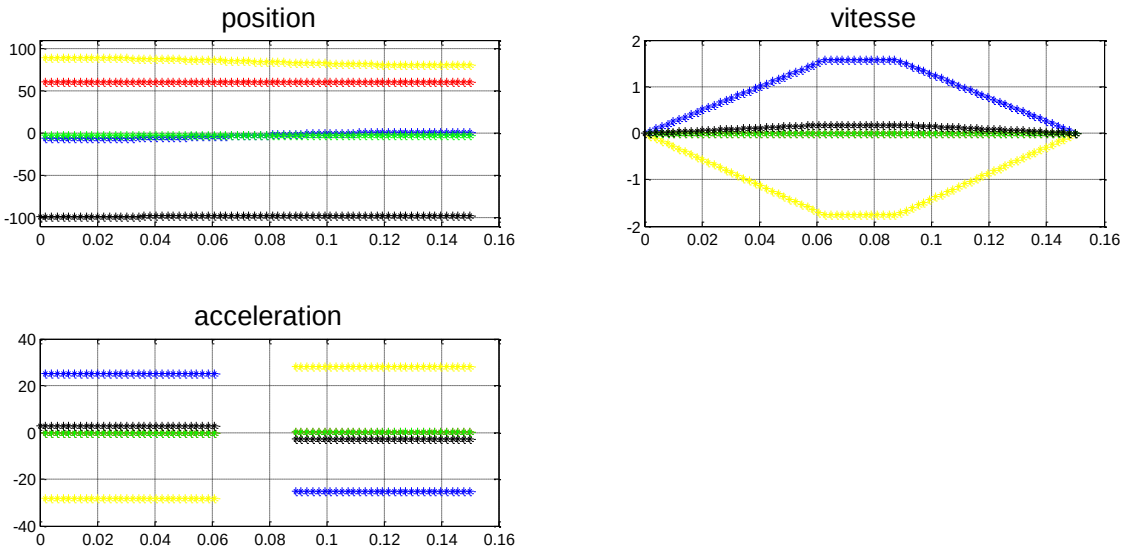


Figure 7 : Loi de trapèze entre P'7 et P'8

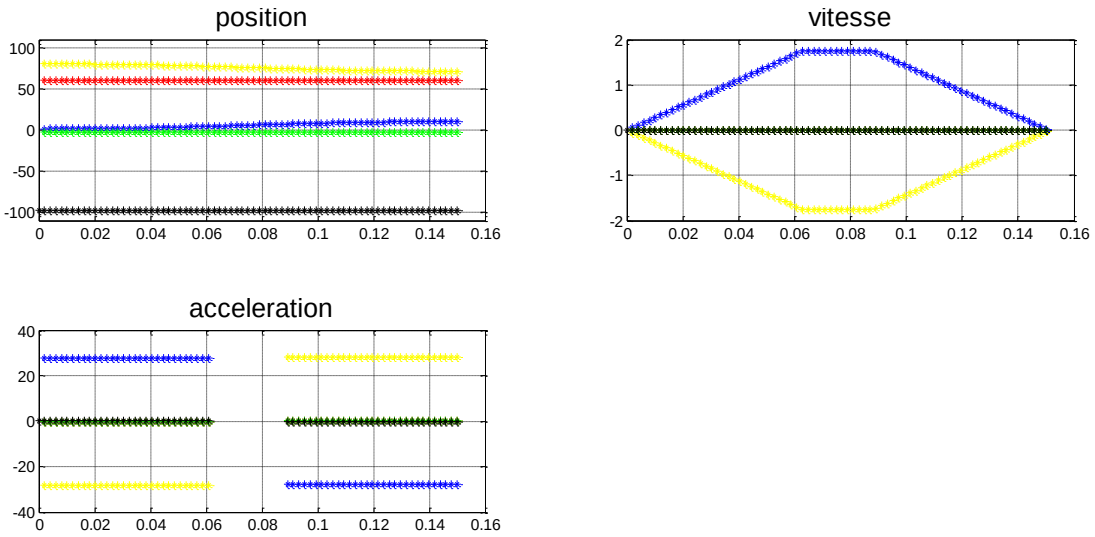


Figure 8 : Loi de trapèze entre P'8 et P'9

Les graphes de (1) à (8) présentent le passage entre deux points avec la loi du mouvement de trapèze, la couleur **rouge** correspond à la configuration de la première articulation **q1**, le **jaune** à la deuxième articulation **q2**, le **bleu** à **q3**, le **vert** à **q4 et le noir à q5.**

On remarque que pour chaque passage, une articulation arrive à la saturation de vitesse avant les autres articulations, donc elle aura la valeur max en vitesse $\mathbf{k_v}$ ou en accélération $\mathbf{k_a}$**.**

Dans la figure 7, on voie que les caractéristiques cinématiques de l'articulation q1 (couleur rouge) n'apparaissent pas dans les graphes du fait que la variation de sa position est pratiquement constante. Il en est de même sur la figure 8 pour cette même articulation et aussi pour q4 (vert).

La figure 3 représente le passage entre le point P'3 et P'4, la distance entre ces deux points est très faible. Le temps d'exécution de ce passage est donc faible. L'accélération atteint la valeur max, donc la loi de mouvement va se transformer en loi de Bang-Bang sans palier de vitesse.

I.2.3. Vérification des Modèles Cinématique

I.2.3.1. Modèle Cinématique du 1er ordre Direct et Inverse (MC.D.I. 1er ord)

Le modèle cinématique direct (MCD) du premier ordre permet de calculer la vitesse opérationnelle ($\dot{X}$) à partir de la vitesse articulaire($\dot{q}$) obtenue par la génération du mouvement (dans notre exemple, loi de trapèze). On procède de la manière inverse pour déterminer le modèle inverse (MCI).

$$\dot{q} \xrightarrow{\text{MCD}} \dot{X} \xrightarrow{\text{MCI}} \dot{q}$$

Le MCI ne donnant pas exactement $\dot{q}$ mais une valeur proche que nous désignons par $\dot{q}'$. On calcule la différence entre ces deux vitesses pour vérifier les résultats de calculs des deux programmes. Les figures ci-dessous donnent, à titre d'exemple, les graphes d'erreurs des vitesses articulaires pour les trois premières étapes. On obtient une erreur relative n'excédant pas $18*10^{-5}$%.

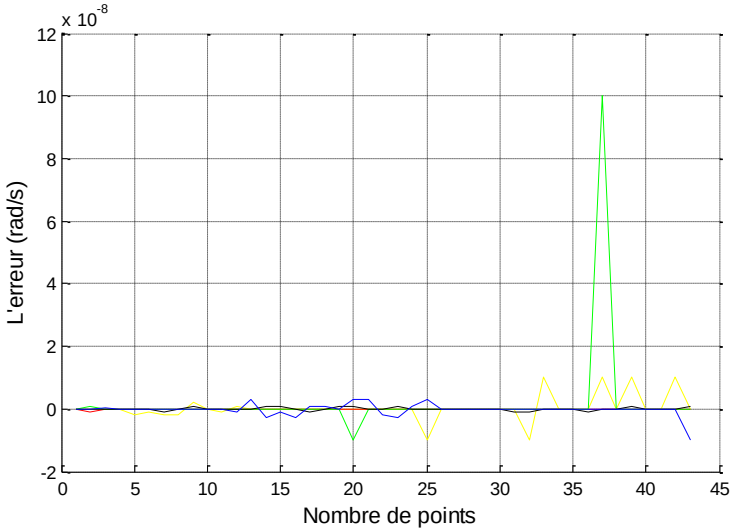


Figure 9 : Erreurs entre $\dot{q}$ et $\dot{q}'$ durant le passage entre P'1 et P'2

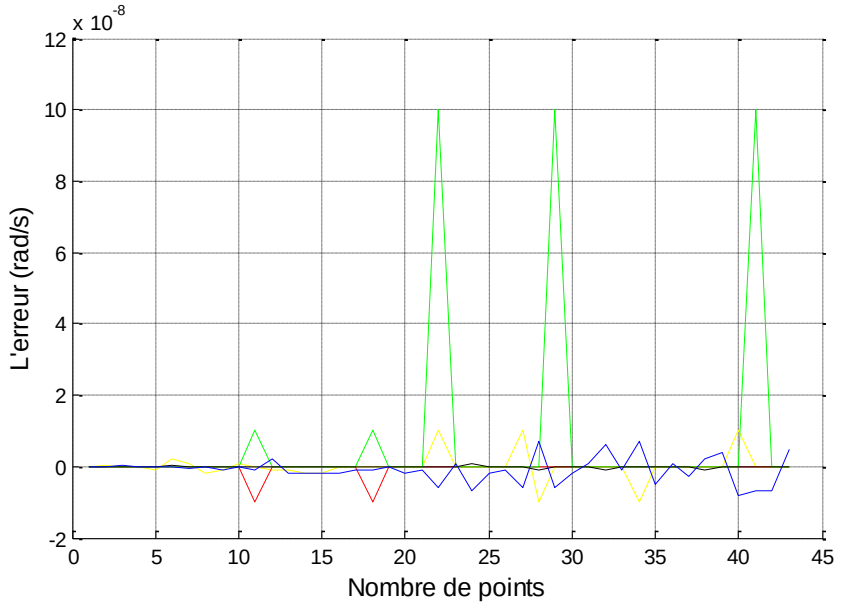


Figure 10 : Erreurs entre $\dot{q}$ et $\dot{q}'$ durant le passage entre P'2 et P'3

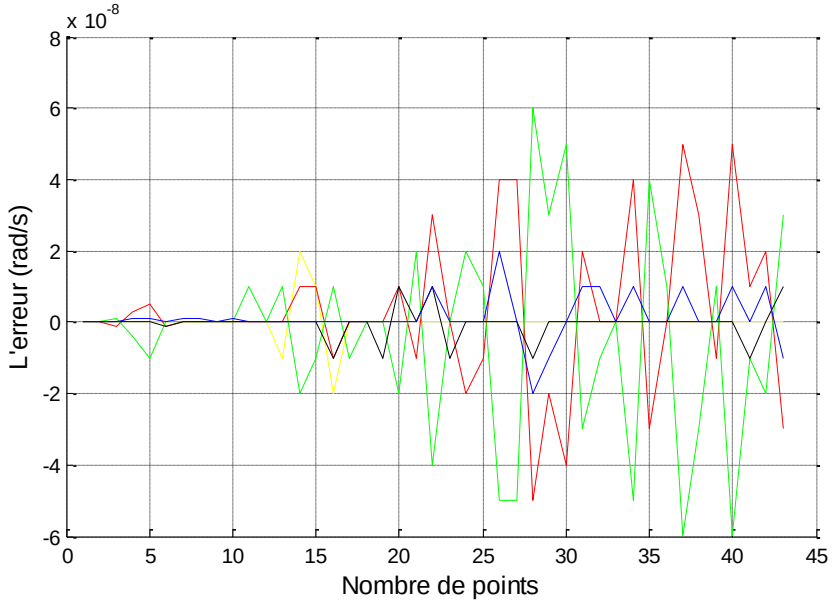


Figure 11 : Erreurs entre $\dot{q}$ et $\dot{q}'$ durant le passage entre P'3 et P'4

I.2.3.2. Modèle Cinématique du 2er ordre Direct et Inverse (MC.D.I. 2er ord)

On a refait la même chose qu'avant, mais avec l'accélération articulaire :

$$\ddot{q} \xrightarrow{\text{MCD}} \ddot{X} \xrightarrow{\text{MCI}} \ddot{q}'$$

Les figures suivantes donnent les erreurs en accélération ($\ddot{q} - \ddot{q}'$) de calculs. On obtient une erreur relative n'excédant pas $1.5*10^{-4}$%.

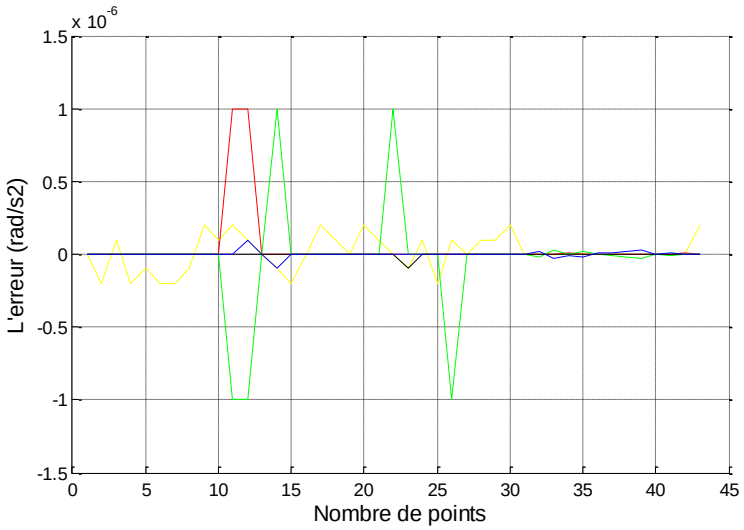


Figure 12 : Erreurs entre q̈ et q̈′ durant le passage entre P'1 et P'2

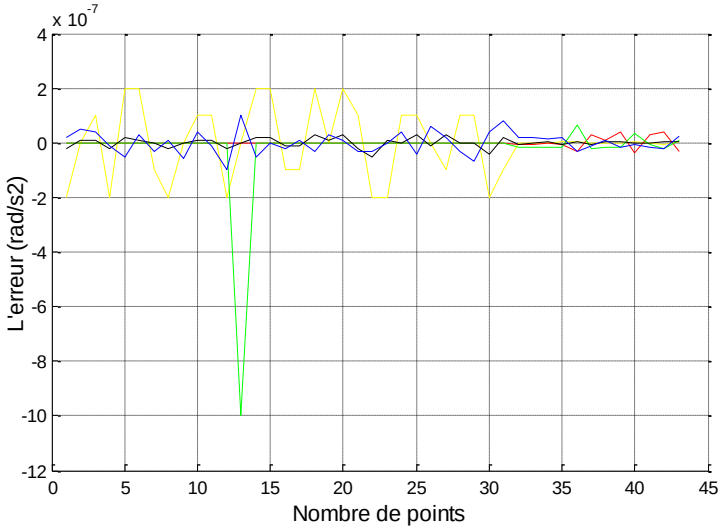


Figure 13 : Erreurs entre q̈ et q̈′ durant le passage entre P'2 et P'3

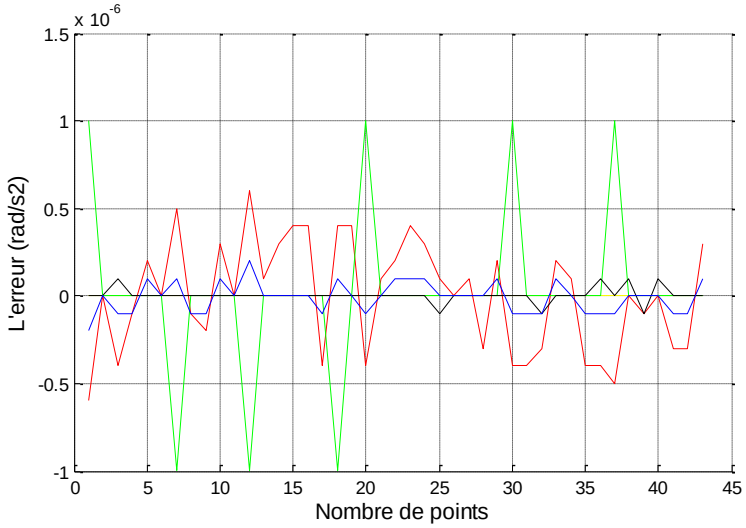
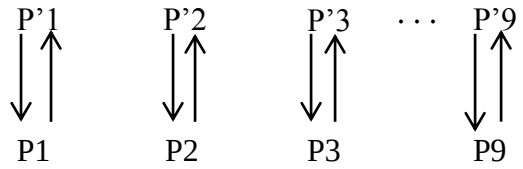


Figure 14 : Erreurs entre q̈ et q̈′ durant le passage entre P'3 et P'4

I.3. Génération du mouvement dans l'espace opérationnel



Les coordonnées des points sont déjà dans l'espace opérationnel (X) sont données dans le tableau 3 suivant :

Points Pi du plan à palper			
Pts	X(m)	Y(m)	Z(m)
1	0.3983	1.5183	1.2225
2	0.4733	1.6482	1.1850
3	0.5483	1.7781	1.1475
4	0.7215	1.6781	1.1275
5	0.6465	1.5482	1.1650
6	0.5715	1.4183	1.2025
7	0.7447	1.3183	1.1825
8	0.8197	1.4482	1.1450
9	0.8947	1.5781	1.1075

Points P'i du plan // au plan palpé			
Pts	X(m)	Y(m)	Z(m)
1	0.3983	1.5183	1.2725
2	0.4733	1.6482	12350
3	0.5483	1.7781	1.1975
4	0.7215	1.6781	1.1775
5	0.6465	1.5482	1.2150
6	0.5715	1.4183	1.2525
7	0.7447	1.3183	1.2325
8	0.8197	1.4482	1.1950
9	0.8947	1.5781	1.1575

Tableau 3 : Coordonnées des points sur le plan à palpé et le plan parallèle au plan palpé par rapport au repère robot dans l'espace opérationnel.

On remarque que la position change de valeur selon l'axe de Z seulement, c-à-d, la dérivée première (vitesse) et deuxième (accélération) existe.

I.3.1. Génération du mouvement dans l'espace opérationnel avec l'interpolation degré 5

Les graphes présentés ci-dessous donnent la variation de position, vitesse et accélération selon la loi d'interpolation de degré 5 suivant l'axe Z. La génération du mouvement donne directement la position (X) les vitesses ($\dot{X}$) et l'accélération ($\ddot{X}$) dans l'espace opérationnel.

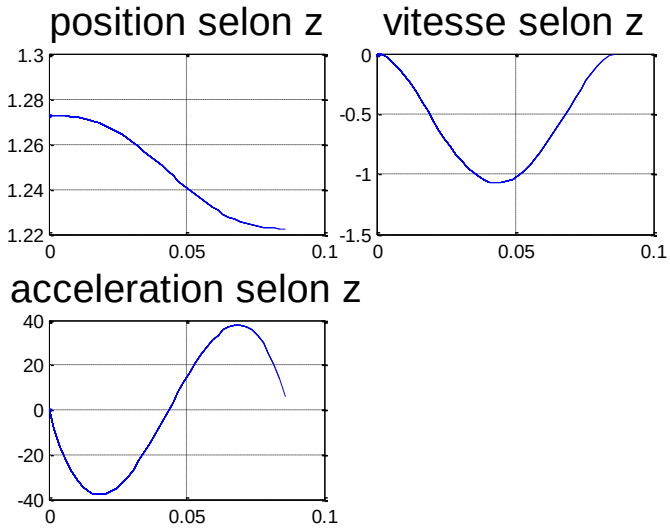


Figure 15 : Loi d'interpolation degré 5 entre P'1 et P1

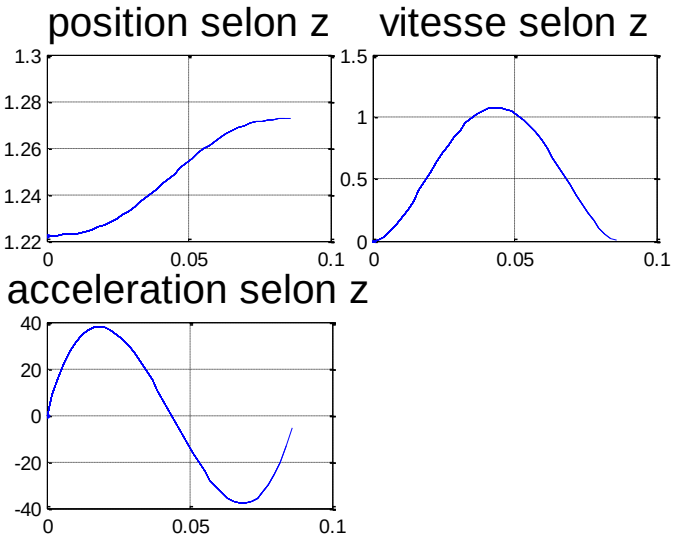


Figure 16 : Loi d'interpolation degré 5 entre P1 et P'1

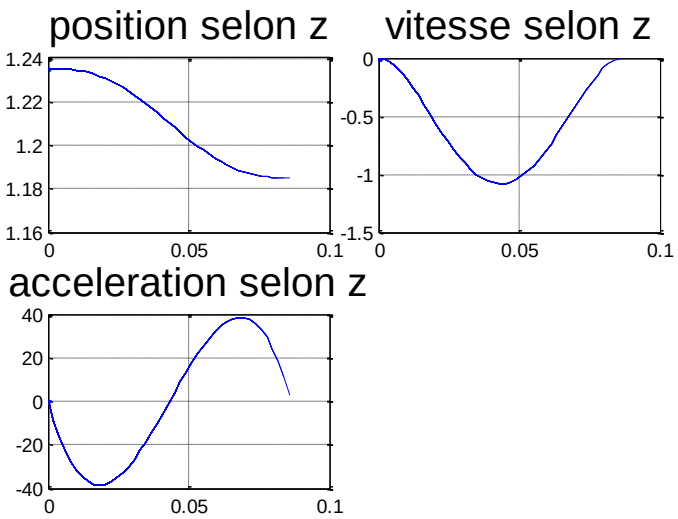


Figure 17 : Loi d'interpolation degré 5 entre P'2 et P2

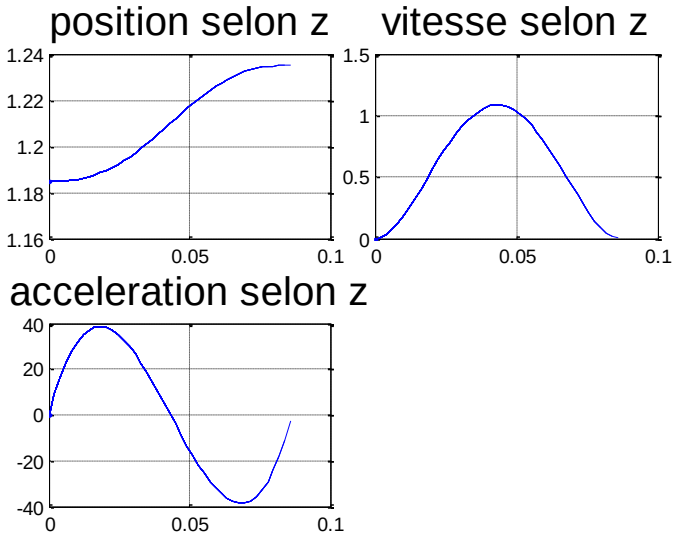


Figure 18 : Loi d'interpolation degré 5 entre P2 et P'2

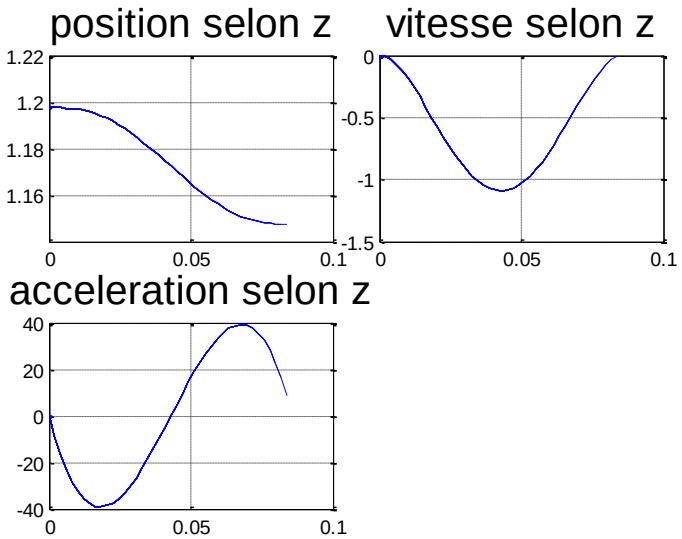


Figure 19 : Loi d'interpolation degré 5 entre P'3 et P3

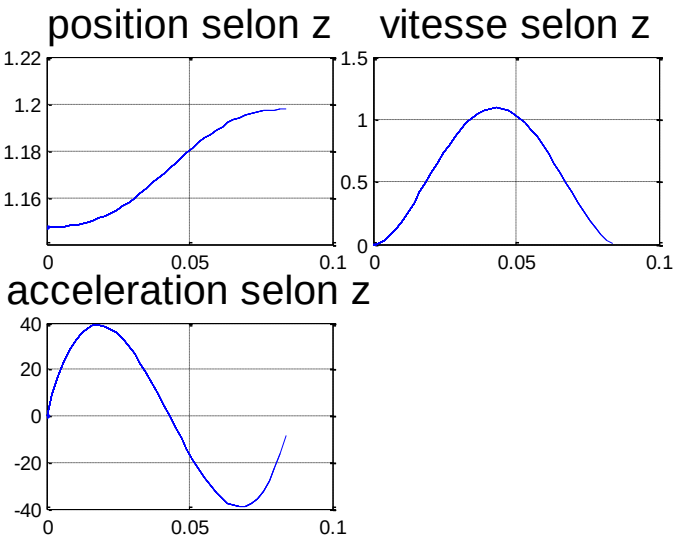


Figure 20 : Loi d'interpolation degré 5 entre P3 et P'3

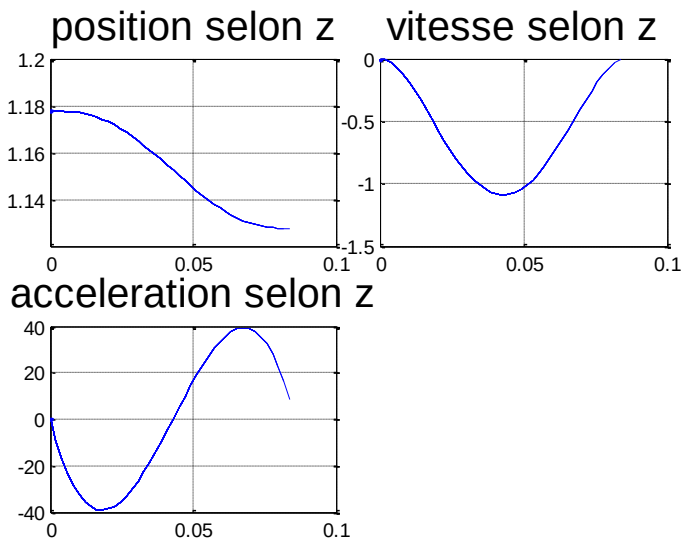


Figure 21 : Loi d'interpolation degré 5 entre P'4 et P4

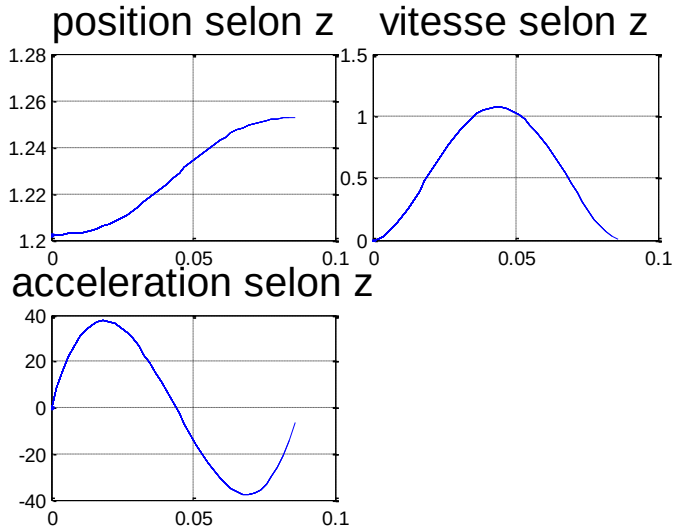


Figure 22 : Loi d'interpolation degré 5 entre P4 et P'4

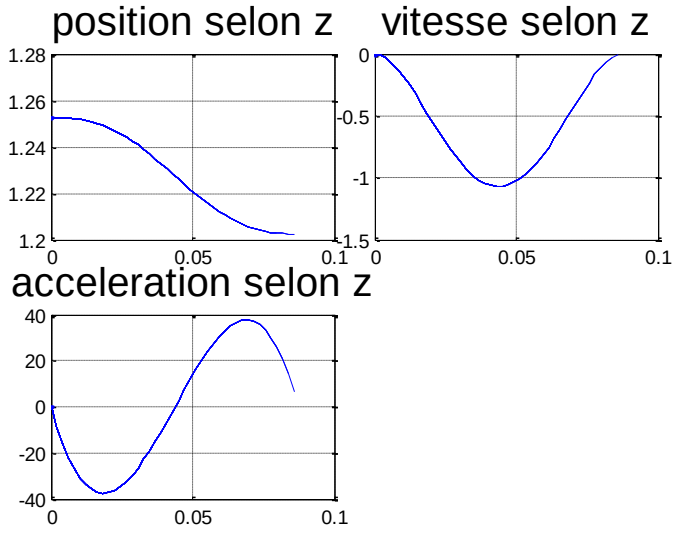


Figure 23 : Loi d'interpolation degré 5 entre P'5 et P5

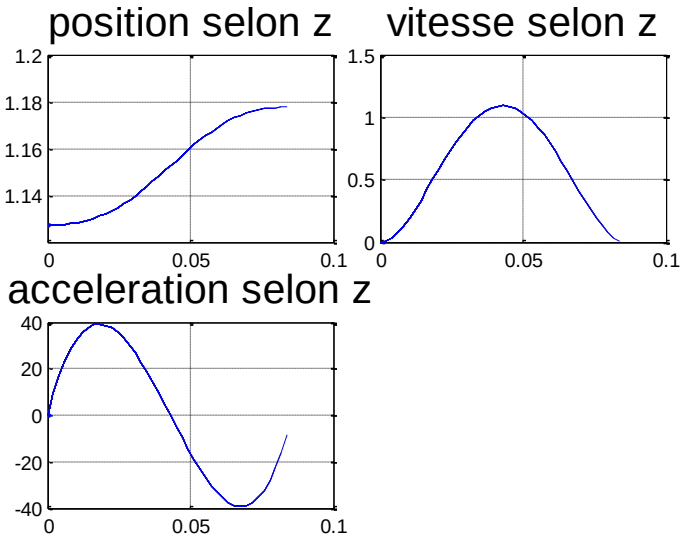


Figure 24 : Loi d'interpolation degré 5 entre P5 et P'5

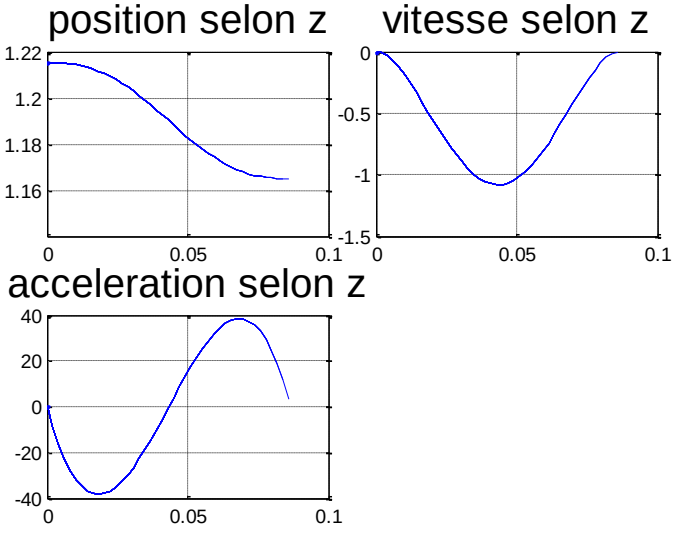


Figure 25 : Loi d'interpolation degré 5 P'6 et P6

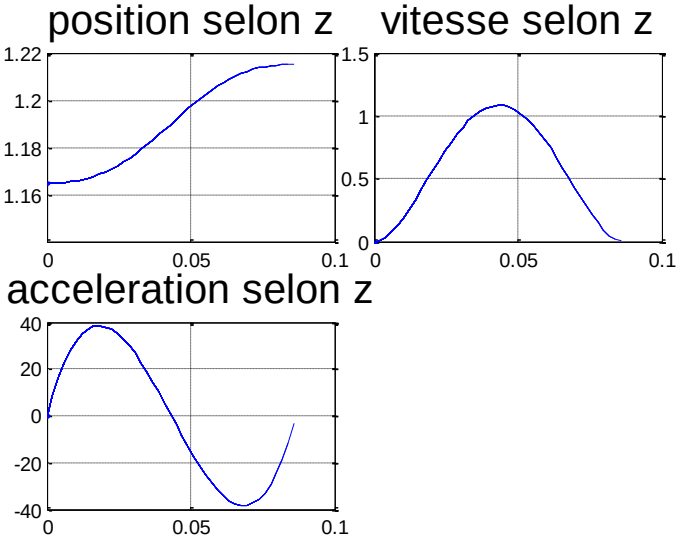


Figure 26 : Loi d'interpolation degré 5 P6 et P'6

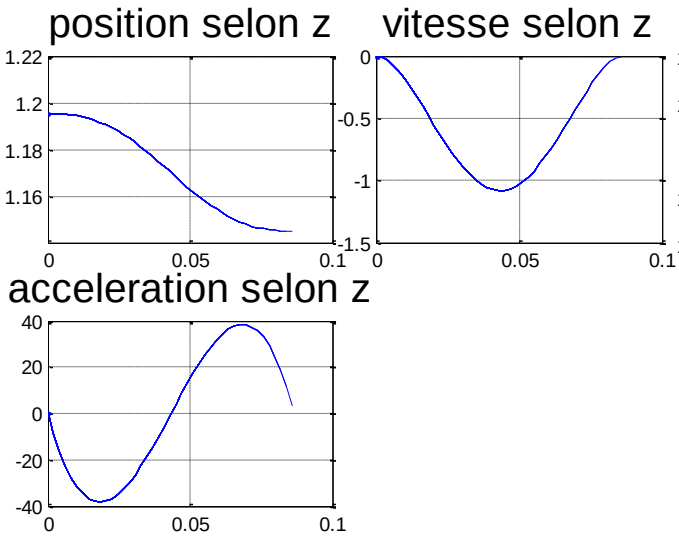


Figure 27 : Loi d'interpolation degré 5 P'7 et P7

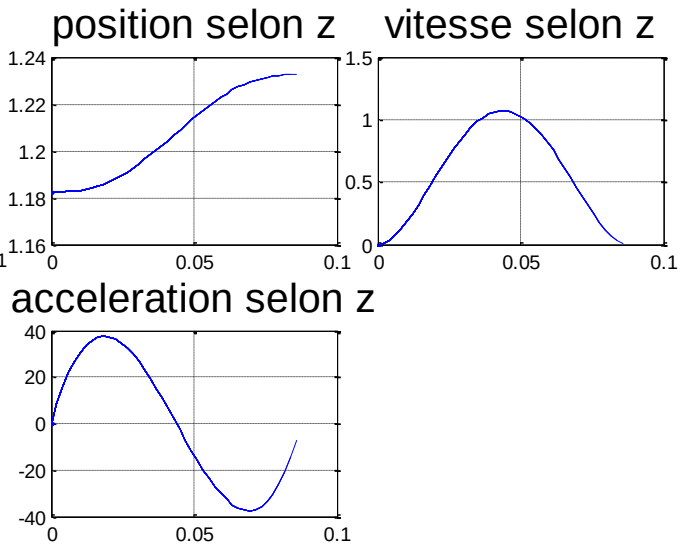


Figure 28 : Loi d'interpolation degré 5 P7 et P'7

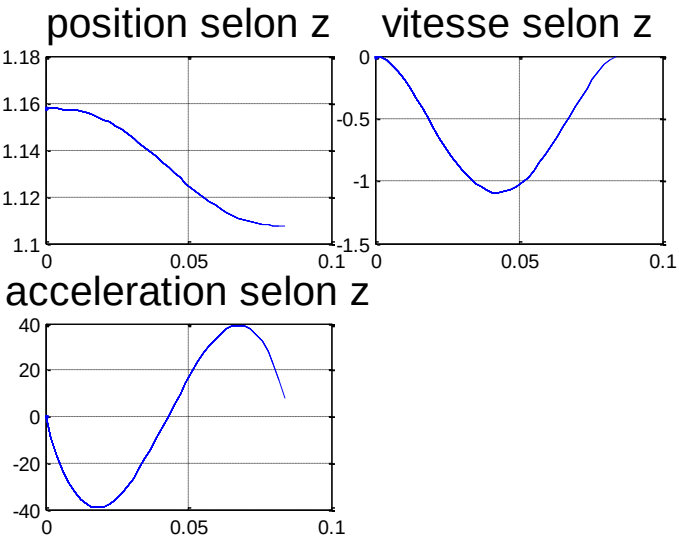


Figure 29 : Loi d'interpolation degré 5 P'8 et P8

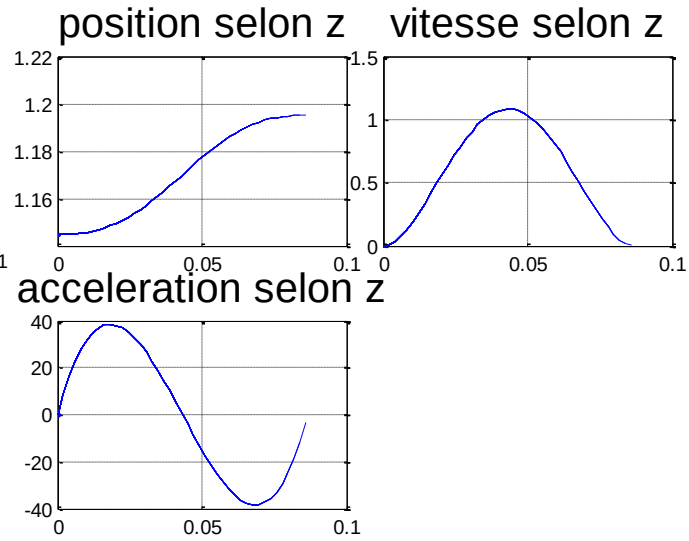


Figure 30 : Loi d'interpolation degré 5 P8 à P'8

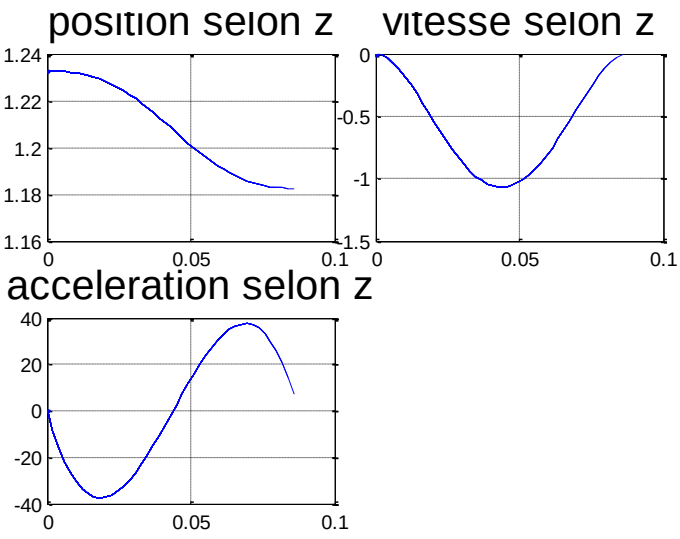


Figure 31 : Loi d'interpolation degré 5 P'9 à P9

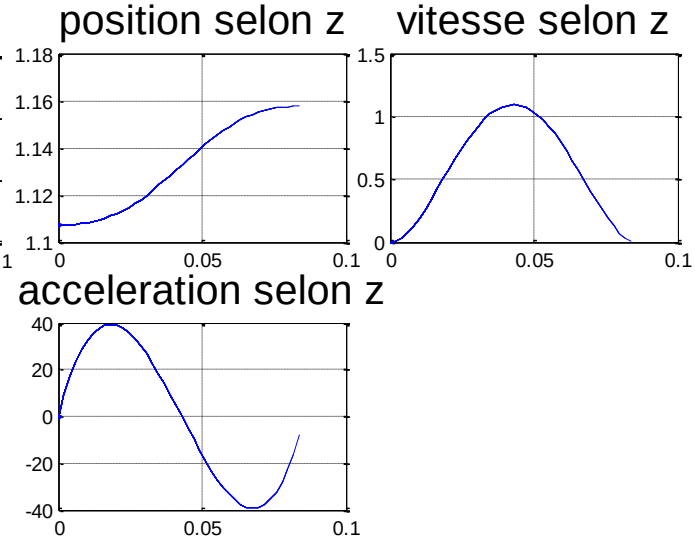


Figure 32 : Loi d'interpolation degré 5 P9 à P'9

I.3.2. Vérification des Modèles Cinématique

I.3.2.1. Modèle Cinématique du 1er ordre Direct et Inverse (MC.D.I. 1er ord)

On utilise le MGI pour déterminer les positions articulaires (q). Le MCI du premier permet de calculer les vitesses articulaires($\dot{q}$). Pour vérifier les calculs, on applique le MCD :

$$\dot{X} \xrightarrow{\text{MCI}} \dot{q} \xrightarrow{\text{MCD}} \dot{X}'$$

Comme avant, on calcule la différence entre les vitesses articulaires pour les trois premières étapes (on garde la même distribution de couleur pour les articulations). Les figures suivantes illustres les erreurs ($\dot{X} - \dot{X}'$) de calculs. On obtient une erreur relative n'excédant pas 0.03%.

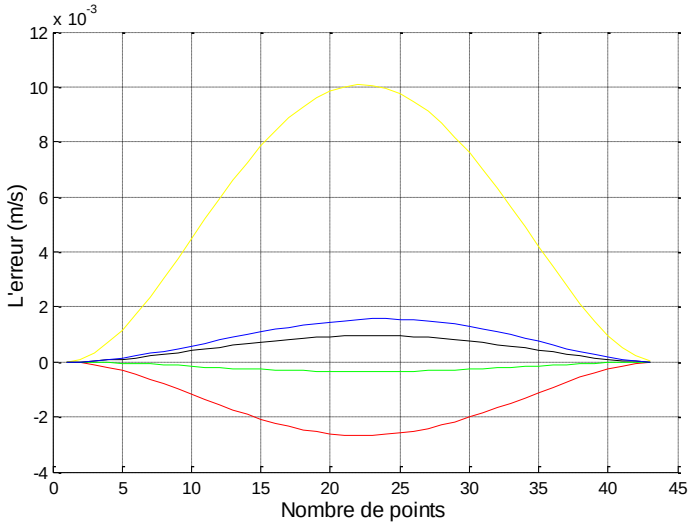


Figure 33 : Erreurs entre $\dot{X}$ et $\dot{X}'$ durant le passage entre P'1 et P'1

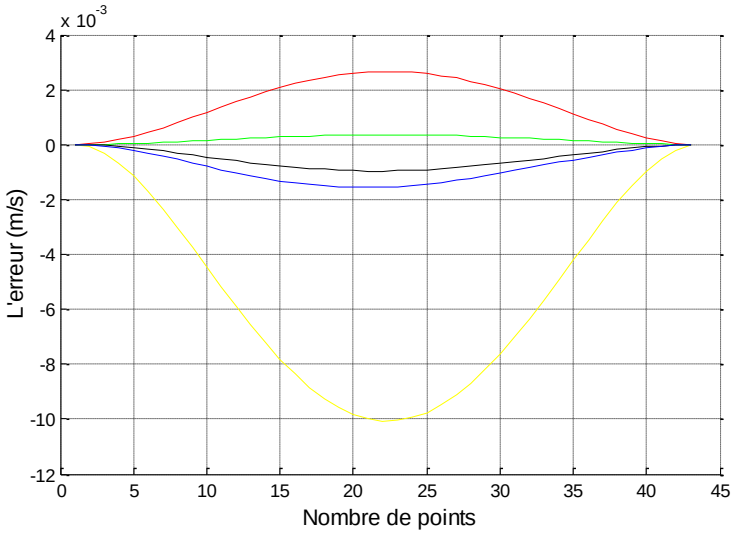


Figure 34 : Erreurs entre $\dot{X}$ et $\dot{X}'$ durant le passage entre P1 et P'1

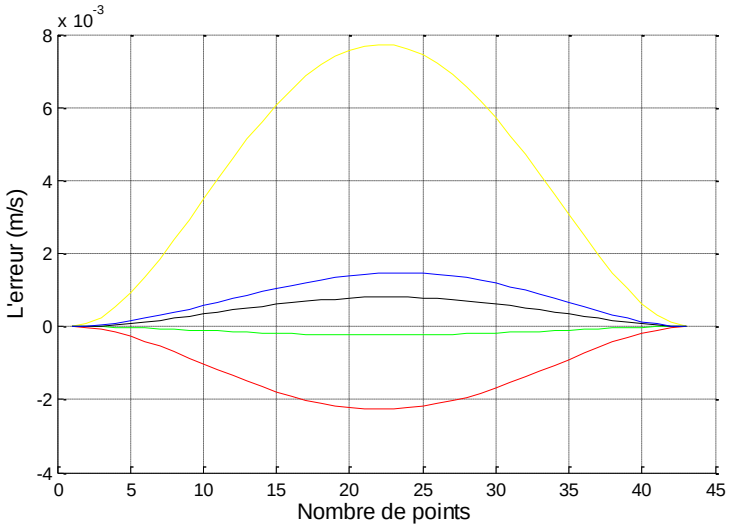


Figure 35 : Erreurs entre $\dot{\boldsymbol{X}}$ et $\dot{\boldsymbol{X}}'$ durant le passage entre P'2 et P2

I.3.2.2. Modèle Cinématique du 2er ordre Direct et Inverse (MC.D.I. 2er ord)

On procède ici aux calculs des accélérations articulaires de la même manière que précédemment. Les calculs montrent que l'erreur relative n'excédant pas 0.1%.

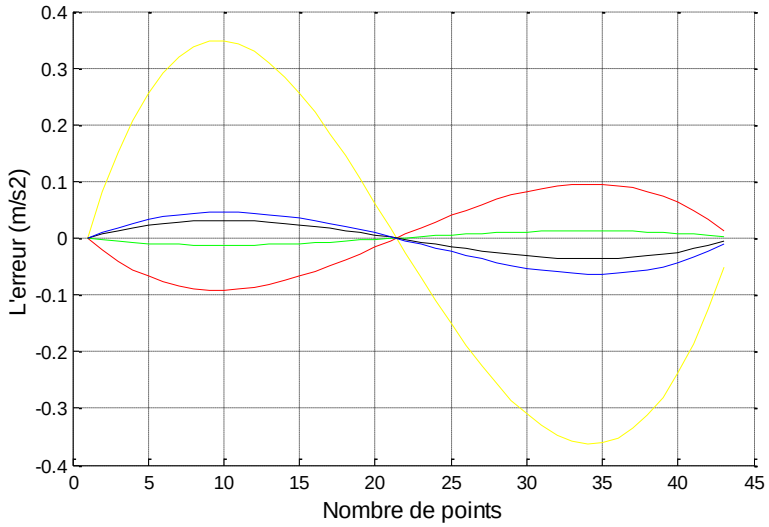


Figure 36 : Erreurs entre $\ddot{\boldsymbol{X}}$ et $\ddot{\boldsymbol{X}}'$ durant le passage entre P'1 et P1

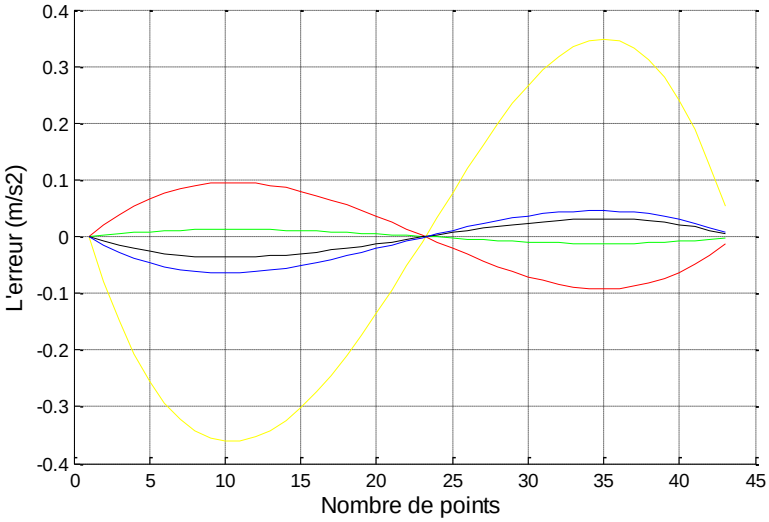


Figure 37 : Erreurs entre $\ddot{X}$ et $\ddot{X}'$ durant le passage entre P1 et P'1

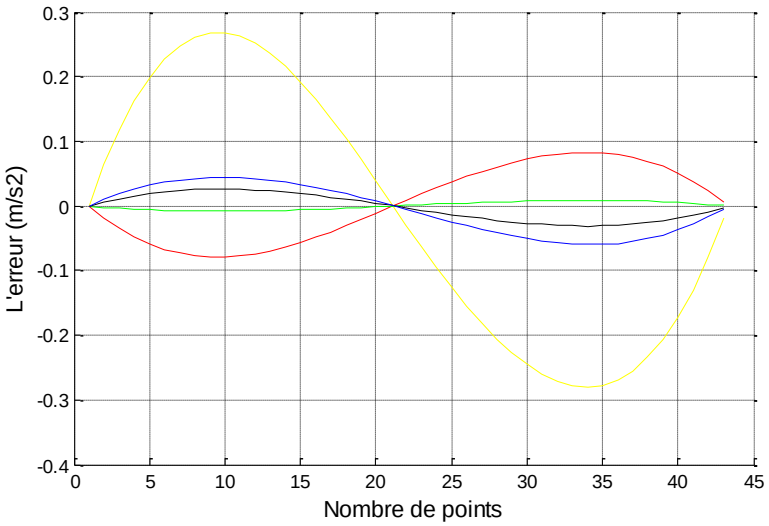


Figure 38 : Erreurs entre $\ddot{X}$ et $\ddot{X}'$ durant le passage entre P'2 et P2

I.4. Conclusion sur les programmes de modélisation

D'après les résultats obtenus, on conclue que la modélisation inverse valide le modèle direct, et le cas contraire est aussi valable. L'erreur des résultats de ces programmes est très faible, elle est due généralement à cause de processus itératif de ces programmes.

II. Validation du programme de génération de mouvement

La comparaison est faite entre notre programme sous MATLAB et celui du laboratoire de robotique de L'EMP (Ecole Militaire Polytechnique) en prenant les mêmes entrées pour la génération de mouvement dans l'espace articulaire.

Les entrées sont :

```
n=5 ;
qinitial=[0   0   0   0    0];
qfinal=[90    210    -90    180   -45];
KV=[3   3   3   2.5   2.5];
KA=[5   5   5   3    3];
Tinitial=0;
```

II.1. Interpolation linéaire :

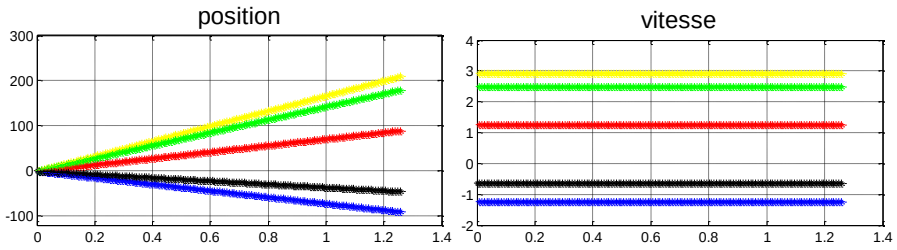
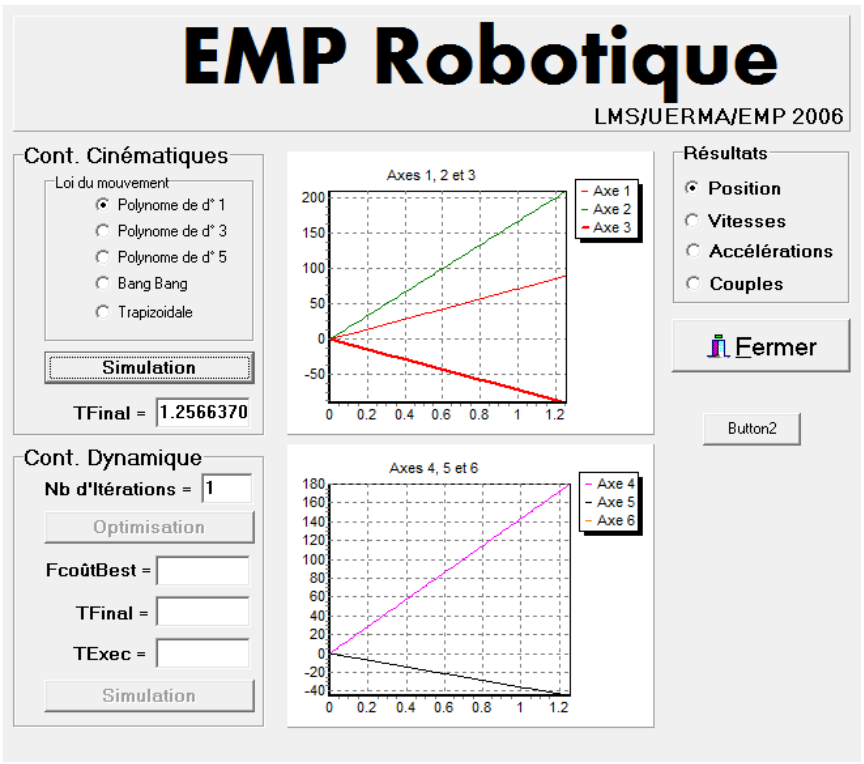


Figure (a1)



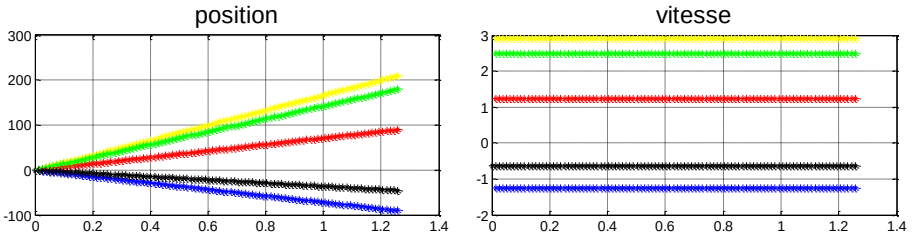


Figure (b1)

II.2. Interpolation degré 3 :

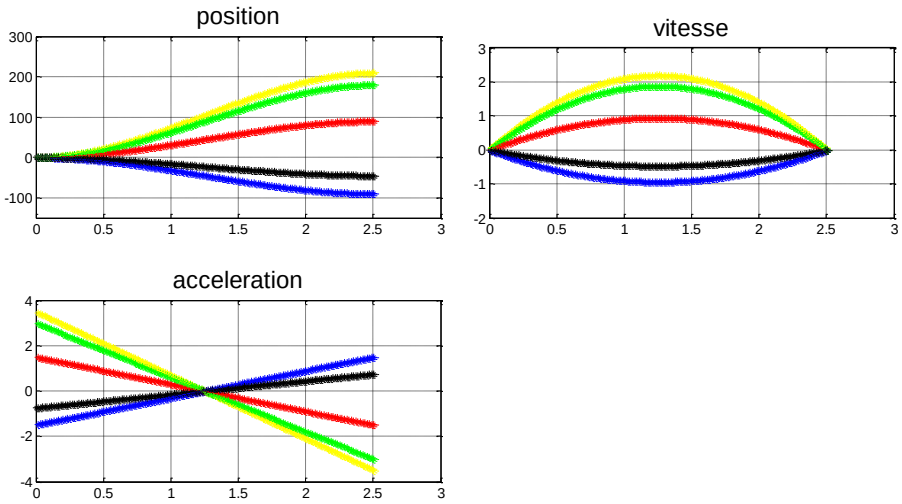


Figure (a2)

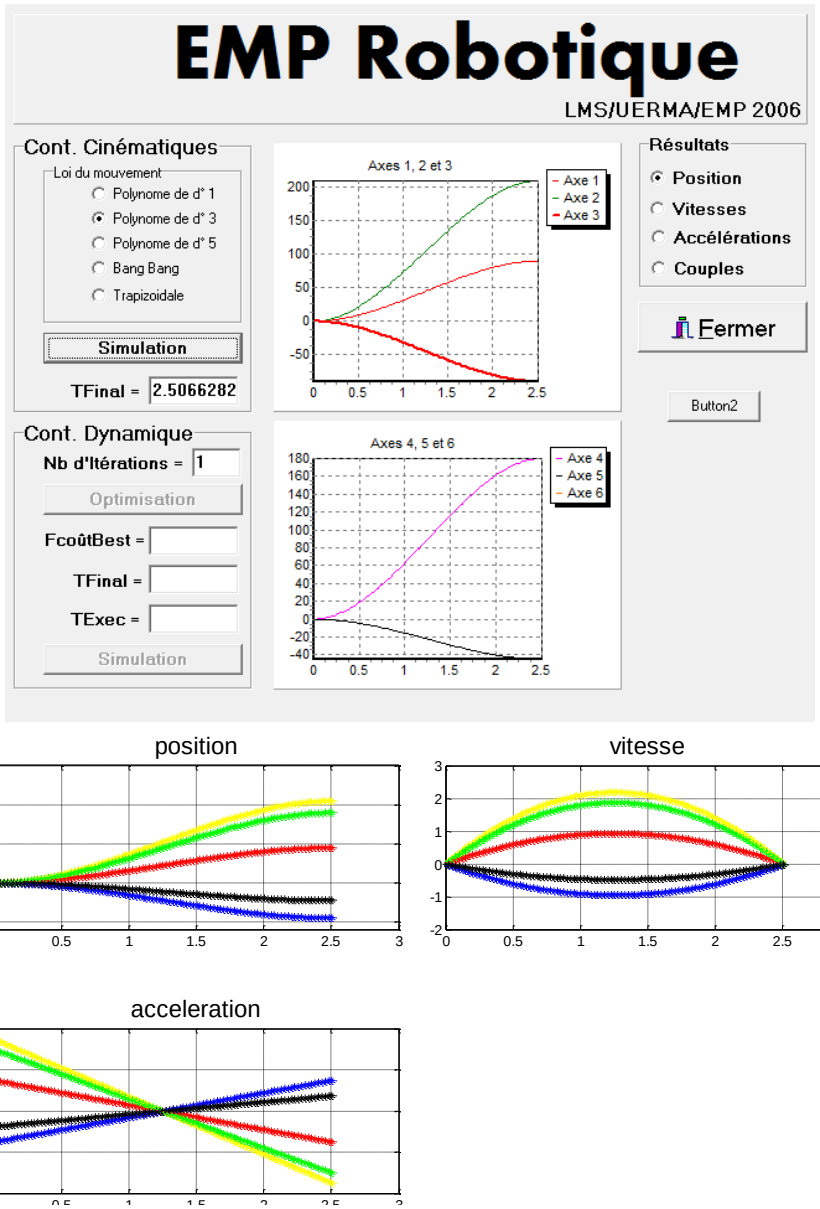


Figure (b2)

II.3. Interpolation degré 5 :

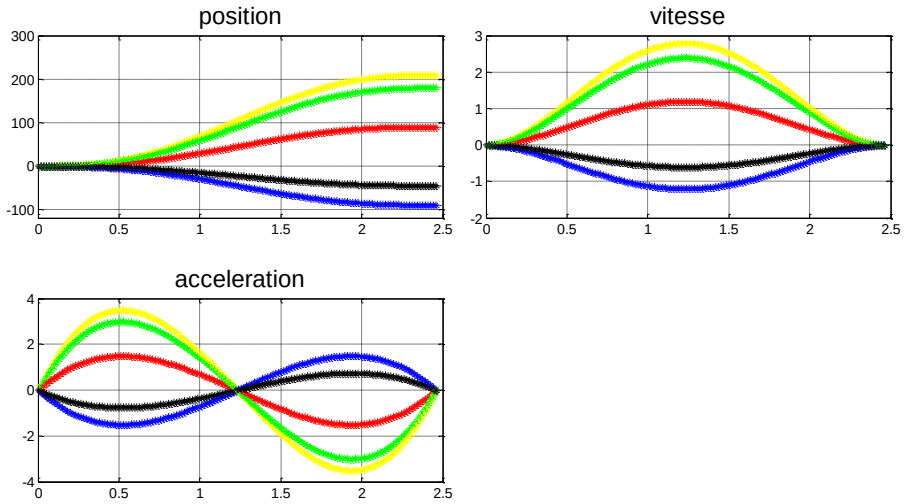


Figure (a3)

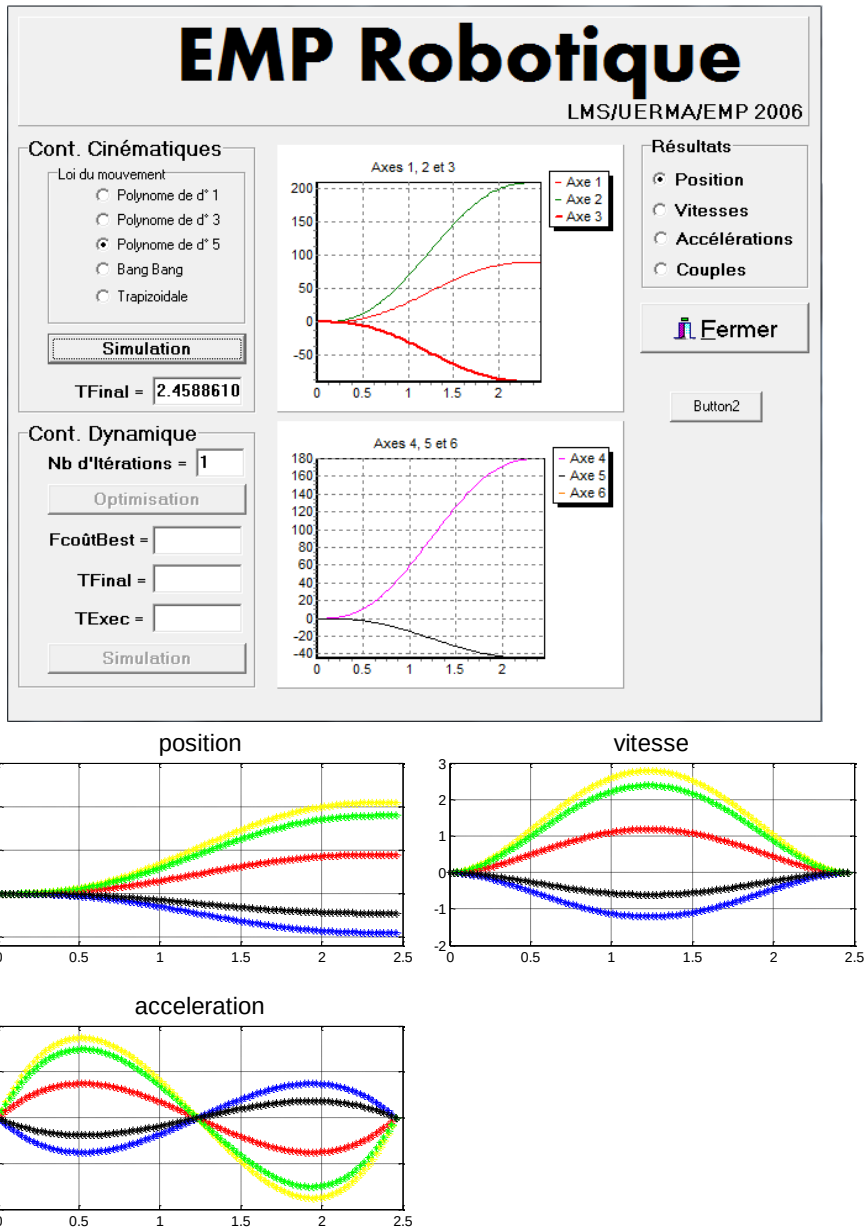


Figure (b3)

II.4. Loi Bang-Bang :

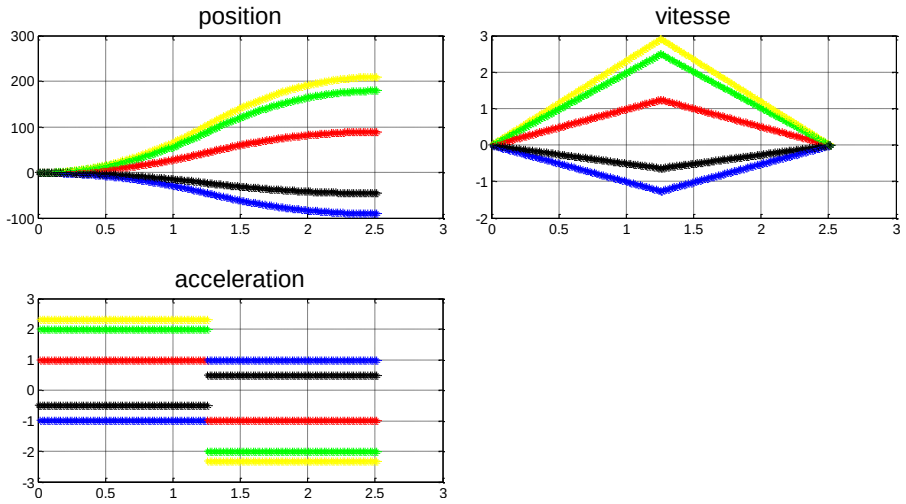


Figure (a4)

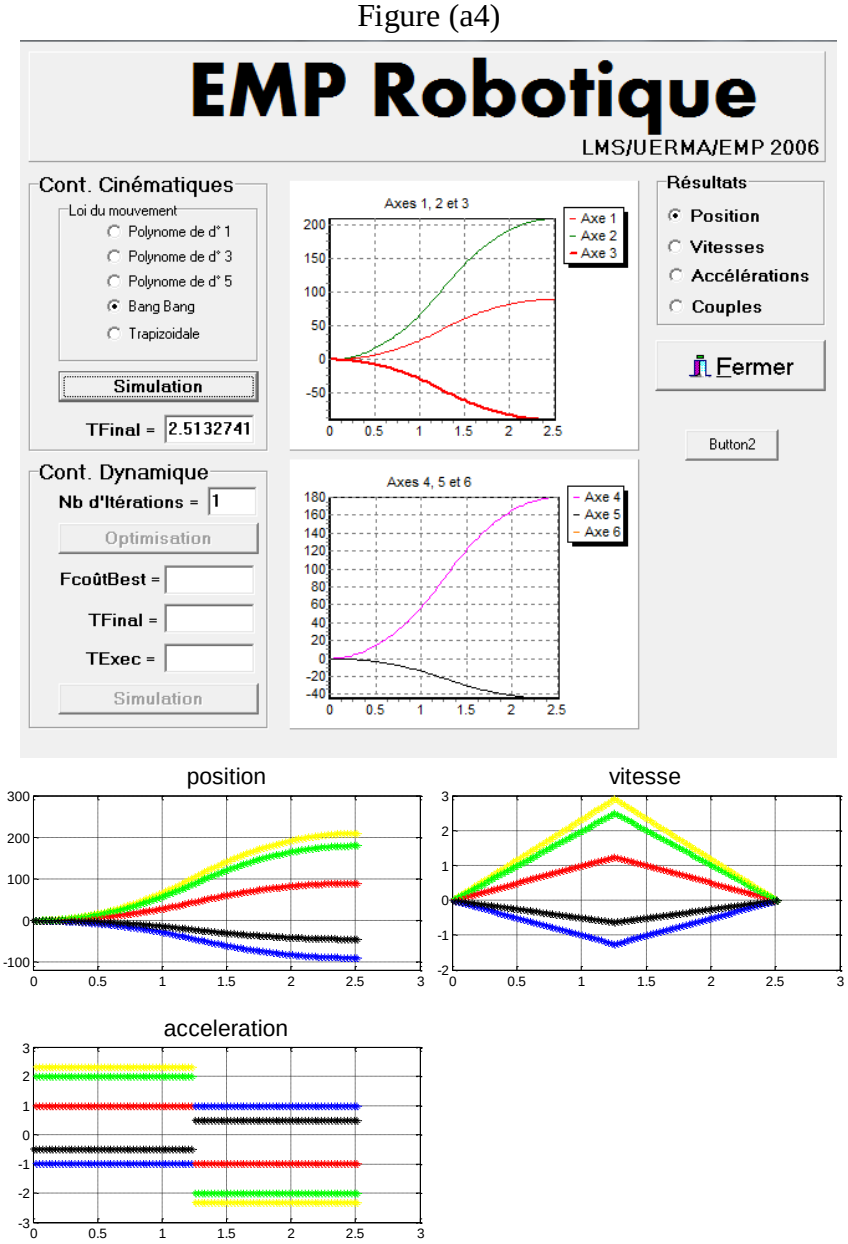


Figure (b4)

II.5. Loi Trapèze :

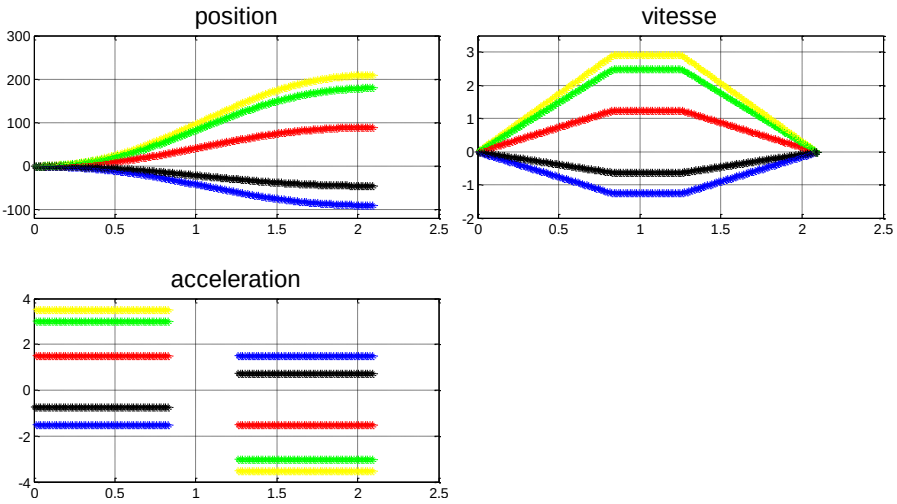


Figure (a5)

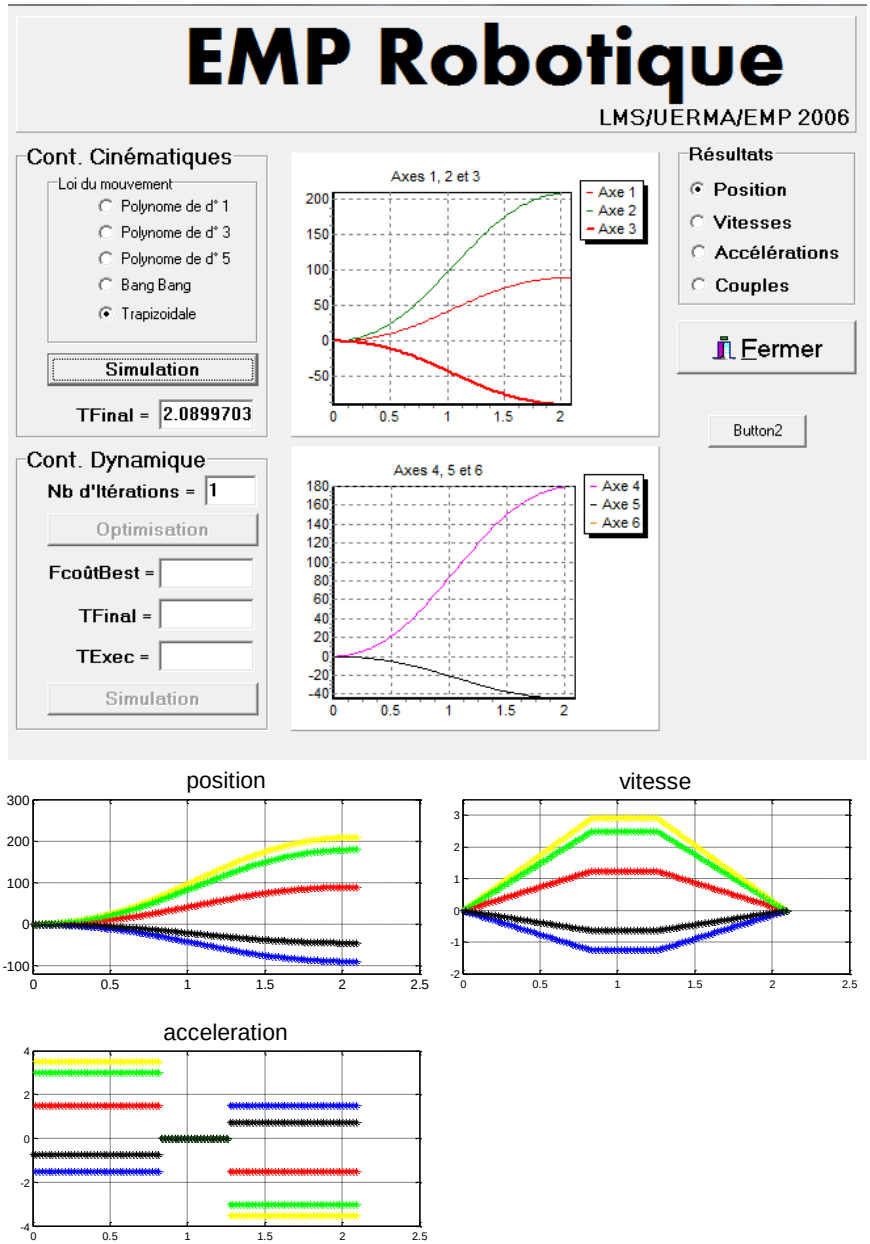


Figure (b5)

Les figure (a1) à (a5) représentent les résultats par le programme proposé, et de (b1) jusqu'à (b5) les graphes obtenus par le laboratoire de robotique de l'EMP.

Le tableau suivant fait la comparaison entre les valeurs du temps final de déplacement pour chaque loi de mouvement.

Lois de mouvement	Temps final (s)	
	Programme proposé	Programme EMP
Interpolation linéaire	1.2566	1.2566
Interpolation degré 3	2.5066	2.5066
Interpolation degré 5	2.4589	2.4588
Loi de Bang-Bang	2.5133	2.5132
Loi de Trapèze	2.0900	2.0899

Tableau 4 : Comparaison entre les valeurs du temps final

II.6. Conclusion sur les programmes de génération de mouvement

Les deux programmes donnent les mêmes résultats, ce qui valide nos calculs. L'avantage de notre programme consiste à son utilisation plus facile; il suffit juste de suivre les instructions affichées sur l'écran. Il est plus général car il englobe l'espace opérationnel et articulaire.

MODELISATIONS GEOMETRIQUES, CINEMATIQUES, DYNAMIQUES ET DE TRAJECTOIRE

Programme Modèle Géométrique Direct (MGD) :

```
clc
clear all
clear global

disp('données une valeur nbr des articulations')
disp(' ')
n=input('n=')

disp(' ')
disp('données une valeur du pas')
disp(' ')
pas=input('pas=')

disp(' ')
fprintf(' on fait entrer les parametres D-H-M format dun tableau pour
chaque parametre')
disp(' ')
fprintf('si segmat=1 larticulation est prismatique ')
disp(' ')
fprintf('et si segmat=0 donc larticulation est rotoide')
 disp(' ')
for i=1:n
fprintf('données une valeur segmat(i) soit 0 ou 1')
disp('')
segmat(i)=input(['segmat(',num2str(i),')='])
end
for i=1:n
fprintf('données une valeur alpha(i) est langle entre les z')
disp('')
alpha(i)=input(['alpha(',num2str(i),')='])
end
for i=1:n
fprintf('données une valeur d(i) est la distance entre les z')
disp('')
d(i)=input(['d(',num2str(i),')='])
end
for i=1:n
 fprintf('données une valeur r(i) est la distance entre les x')
 disp('')
 rmin(i)=input(['rmin(',num2str(i),')='])
 fprintf('données une valeur r(i) est la distance entre les x')
 disp('')
 rmax(i)=input(['rmax(',num2str(i),')='])
end

for i=1:n
fprintf('données une valeur alpha(i) est langle entre les x')
disp('')
qmin(i)=input(['qmin(',num2str(i),')='])
fprintf('données une valeur d(i) est la distance entre les x')
disp('')
qmax(i)=input(['qmax(',num2str(i),')='])
```

```
end
for i=1:4
fprintf('données coordonnées du lorgane terminal')
disp('')
OT(i)=input(['ot(',num2str(i),')='])
end
ot=OT'

for i=1:n
if segmat(i)==0
r(i)=rmin(i):rmax(i);
theta(i,:)=qmin(i):(qmax(i)-qmin(i))/pas:qmax(i);
m=length(theta(i,:));
for j=1:m
C=cosd(theta(i,j));
S=sind(theta(i,j));
    T(1,1,i,j)=C;
    T(1,2,i,j)=-S;
    T(1,3,i,j)=0 ;
    T(1,4,i,j)=d(i);
    T(2,1,i,j)=S*cosd(alpha(i));
    T(2,2,i,j)=C*cosd(alpha(i));
    T(2,3,i,j)=-sind(alpha(i));
    T(2,4,i,j)=-r(i)*sind(alpha(i));
    T(3,1,i,j)=S*sind(alpha(i));
    T(3,2,i,j)=C*sind(alpha(i));
    T(3,3,i,j)=cosd(alpha(i));
    T(3,4,i,j)=r(i)*cosd(alpha(i));
    T(4,1,i,j)=0;
    T(4,2,i,j)=0;
    T(4,3,i,j)=0;
    T(4,4,i,j)=1 ;
end

else
theta(i)=qmin(i):qmax(i);
R(i,:)=rmin(i):(rmax(i)-rmin(i))/pas:rmax(i);
m=length(R(i,:)),
for j=1:m
B=R(i,j);
    T(1,1,i,j)=cosd(theta(i));
    T(1,2,i,j)=-sind(theta(i));
    T(1,3,i,j)=0 ;
    T(1,4,i,j)=d(i);
    T(2,1,i,j)=sind(theta(i))*cosd(alpha(i));
    T(2,2,i,j)=cosd(theta(i))*cosd(alpha(i));
    T(2,3,i,j)=-sind(alpha(i));
    T(2,4,i,j)=-B*sind(alpha(i));
    T(3,1,i,j)=sind(theta(i))*sind(alpha(i));
    T(3,2,i,j)=cosd(theta(i))*sind(alpha(i));
    T(3,3,i,j)=cosd(alpha(i));
    T(3,4,i,j)=B*cosd(alpha(i));
    T(4,1,i,j)=0;
    T(4,2,i,j)=0;
    T(4,3,i,j)=0;
    T(4,4,i,j)=1 ;
    end
   end
end
```

```
if n==1
h=1;
for j=1:m
TT=eye(4);
for i=1:n
TT=TT*T(:,:,i,j);
end
xx(h)=TT(1,:)*ot;
yy(h)=TT(2,:)*ot;
zz(h)=TT(3,:)*ot;
h=h+1;
end
hold on
grid on
plot3(xx,yy,zz,'.')
end

if n==2
h=1;
k=1;
for ii=1:m
for j=1:m
TT=eye(3);
for i=1:n
if i==n
TT=TT*T(:,:,i,j);
else i==n-1
TT=TT*T(:,:,i,k);
end
end
xx(h)=TT(1,:)*ot;
yy(h)=TT(2,:)*ot;
zz(h)=TT(3,:)*ot;
h=h+1;
end
k=k+1;
end
hold on
grid on
plot3(xx,yy,zz,'.')
end

if n==3
l=1;
h=1;
for jj=1:m
k=1;
for ii=1:m
for j=1:m
TT=eye(4);
for i=1:n
if i==n
TT=TT*T(:,:,i,j);
elseif i==n-1
TT=TT*T(:,:,i,k);
else
TT=TT*T(:,:, i,j);
end
end
xx(h)=TT(1,:)*ot;
yy(h)=TT(2,:)*ot;
```

```
zz(h)=TT(3,:)*ot;
h=h+1;
end
k=k+1;
end
l=l+1;
end
hold on
grid on
plot3(xx,yy,zz,'.')
end

if n==4
h=1;
f=1;
for kk=1:m
l=1;
for jj=1:m
k=1;
for ii=1:m
for j=1:m
TT=eye(4);
for i=1:n
if i==n
TT=TT*T(:,:,i,j);
elseif i==n-1
TT=TT*T(:,:,i,k);
elseif i==n-2
TT=TT*T(:,:,i,l);
else
TT=TT*T(:,:,i,f);
end
end
xx(h)=TT(1,:)*ot;
yy(h)=TT(2,:)*ot;
zz(h)=TT(3,:)*ot;
h=h+1;
end
k=k+1;
end
l=l+1;
end
f=f+1;
end
hold on
grid on
plot3(xx,yy,zz,'.')
end

if n==5
h=1;
c=1;
for hh=1:m
f=1;
for kk=1:m
l=1;
for jj=1:m
k=1;
for ii=1:m
for j=1:m
TT=eye(4);
```

```
for i=1:n
if i==n
TT=TT*T(:,:,i,j);
elseif i==n-1
TT=TT*T(:,:,i,k);
elseif i==n-2
TT=TT*T(:,:,i,l);
else if i==n-3
TT=TT*T(:,:,i,f);
else
TT=TT*T(:,:,i,c);
end
end
end
xx(h)=TT(1,:)*ot;
yy(h)=TT(2,:)*ot;
zz(h)=TT(3,:)*ot;
h=h+1;
end
k=k+1;
end
l=l+1;
end
f=f+1;
end
c=c+1;
end
hold on
grid on
plot3(xx,yy,zz,'.')
end
if n==6
h=1;
a=1;
for ll=1:m
c=1;
for hh=1:m
f=1;
for kk=1:m
l=1;
for jj=1:m
k=1;
for ii=1:m
for j=1:m
TT=eye(4);
for i=1:n
if i==n
TT=TT*T(:,:,i,j);
elseif i==n-1
TT=TT*T(:,:,i,k);
elseif i==n-2
TT=TT*T(:,:,i,l);
else if i==n-3
TT=TT*T(:,:,i,f);
else if i==n-4
TT=TT*T(:,:,i,d);
else
TT=TT*T(:,:,i,a);
end
end
end
```

```
end
xx(h)=TT(1,:)*ot;
yy(h)=TT(2,:)*ot;
zz(h)=TT(3,:)*ot;
h=h+1;
end
k=k+1;
end
l=l+1;
end
f=f+1;
end
c=c+1;
end
a=a+1;
end
hold on
grid on
plot3(xx,yy,zz,'.')
end
```

Programme Modèle Géométrique Inverse (MGI) :

```
clc
clear all
clear global
disp(' ')
disp('données une valeur nbr des articulations')
disp(' ')
n=input('n=')
disp(' ')
disp(' ')
disp('données une valeur dd mobilité')
disp(' ')
m=input('m=')
disp(' ')
% E:l'erreur
for i=1:n
    E(i)=1e-4;
end

fprintf('on fait entrer les parametres D-H-M format dun tableau pour chaque
parametre')
disp(' ')
fprintf('si segmat=1 larticulation est prismatique ')
disp(' ')
fprintf('et si segmat=0 donc larticulation est rotoide')
disp(' ')
for i=1:n
    fprintf('données une valeur segmat(i) soit 0 ou 1')
    disp('')
    segmat(i)=input(['segmat(',num2str(i),')='])
end
for i=1:n
    fprintf('données une valeur alpha(i) est langle entre les z')
    disp('')
    alpha(i)=input(['alpha(',num2str(i),')='])
end
for i=1:n
```

```
    fprintf('données une valeur d(i) est la distance entre les z')
    disp('')
    d(i)=input(['d(',num2str(i),')='])
end
for i=1:n
    fprintf('données une valeur r0(i) est la distance entre les x')
    disp('')
    r(i)=input(['r(',num2str(i),')='])
end
for i=1:n
    fprintf('données une valeur q0(i) est langle entre les z')
    disp('')
    theta(i)=input(['theta(',num2str(i),')='])
end
for i=1:n
    if segmat(i)==0
        q(i)=theta(i);
    else
        q(i)=r(i);
    end
end

for i=1:n
    fprintf('données une valeur qp(i)')
    disp('')
    qp(i)=input(['qp(',num2str(i),')='])
end
disp(' ')
disp('entrée le choix des angles dorientation ')
disp(' ')
fprintf('si choix=1 orientation par les angles dEULER ZXZ ')
disp(' ')
fprintf('si choix=2 orientation par les angles ROULIS-TANGAGE-LACET ZYX ')
disp(' ')
choix=input('choix=')
disp(' ')
disp('données une valeur X')
disp(' ')
X(1)=input(['X='])
disp(' ')
disp('données une valeur Y')
disp(' ')
X(2)=input(['Y='])
disp(' ')
disp('données une valeur Z')
disp(' ')
X(3)=input(['Z='])
disp(' ')
disp('données les angles dorientation en degré ')
psi=input(['psi='])
disp(' ')
beta=input(['beta='])
disp(' ')
fi=input(['fi='])
ot=[0;0;0;1];

for i=1:n
    segmabar(i)=~segmat(i);
end
```

```
for i=1:n
    Zz(1,i)=0;
    Zz(2,i)=0;
    Zz(3,i)=1;
end
dq=ones(n,1);
deltaq=zeros(n,1);
h=0;
%%=========================================================================
% la valeur de dq doit etre supérieur de l'erreur
%%%=========================================================================
while(abs(dq(1))>E(1))|(abs(dq(2))>E(2))|(abs(dq(3))>E(3))|(abs(dq(4))>E(4)
)|(abs(dq(5))>E(5))|(abs(dq(6))>E(6))
    h=h+1;%pour calculer nombre d'opération
    q=deltaq'+q;
%=========================================================================
%%%%%%%%%%%%%%%%calcul matrice de transfere
%=========================================================================
for i=1:n
if segmat(i)==0

            T(1,1,i)=cosd(q(i));
            T(1,2,i)=-sind(q(i));
            T(1,3,i)=0 ;
            T(1,4,i)=d(i);
            T(2,1,i)=sind(q(i))*cosd(alpha(i));
            T(2,2,i)=cosd(q(i))*cosd(alpha(i));
            T(2,3,i)=-sind(alpha(i));
            T(2,4,i)=-r(i)*sind(alpha(i));
            T(3,1,i)=sind(q(i))*sin(alpha(i));
            T(3,2,i)=cosd(q(i))*sin(alpha(i));
            T(3,3,i)=cosd(alpha(i));
            T(3,4,i)=r(i)*cosd(alpha(i));
            T(4,1,i)=0;
            T(4,2,i)=0;
            T(4,3,i)=0;
            T(4,4,i)=1;

        else
            T(1,1,i)=cosd(theta(i));
            T(1,2,i)=-sind(theta(i));
            T(1,3,i)=0 ;
            T(1,4,i)=d(i);
            T(2,1,i)=sind(theta(i))*cosd(alpha(i));
            T(2,2,i)=cosd(theta(i))*cosd(alpha(i));
            T(2,3,i)=-sind(alpha(i));
            T(2,4,i)=-q(i)*sind(alpha(i));
            T(3,1,i)=sind(theta(i))*sind(alpha(i));
            T(3,2,i)=cosd(theta(i))*sind(alpha(i));
            T(3,3,i)=cosd(alpha(i));
            T(3,4,i)=q(i)*cosd(alpha(i));
            T(4,1,i)=0;
            T(4,2,i)=0;
            T(4,3,i)=0;
            T(4,4,i)=1;
        end
    end
%=========================================================================
%================le produit ; MGD ========================================
```

```
 TT=eye(4);
 for i=1:n
 TT=TT*T(:,:,i);
 end
 %=========================================================================
 %============== la différence entre la position désiré et commandé
    F(1:3)=TT(1:3,4)'-X(1:3);
 %=========================================================================
 %%%%%    on calcul l'orientation
 %=========================================================================
  RTL=TT(1:3,1:3);

 % l'orientation est donnée par les angles d'Euler ou de R-T-L

if choix==1
%=========================================================================
    % les angles d'EULER avec la séquence ZXZ
%=========================================================================
    psi2=atan2(-RTL(1,3),RTL(2,3));
    pssi2=(psi2*180)/pi;

    beta2=atan2(sin(psi2)*RTL(1,3)-cos(psi2)*RTL(2,3),RTL(3,3));
    betta2=(beta2*180)/pi;

    fi2=atan2(-cos(psi2)*RTL(1,2)-
sin(psi2)*RTL(2,2),cos(psi2)*RTL(1,1)+sin(psi2)*RTL(2,1));
    ffi2=(fi2*180)/pi;

%=========================================================================
else (choix==2)
%=========================================================================
    %les angles de ROULIS-TANGAGE-LACET avec la séquence ZYX
%=========================================================================
    psi2=atan2(RTL(2,1),RTL(1,1));
    pssi2=(psi2*180)/pi;

    beta2=atan2(-RTL(3,1),cos(psi2)*RTL(1,1)+sin(psi2)*RTL(2,1));
    betta2=(beta2*180)/pi;

    fi2=atan2(sin(psi2)*RTL(1,3)-cos(psi2)*RTL(2,3),-
sin(psi2)*RTL(1,2)+cos(psi2)*RTL(2,2);
    ffi2=(fi2*180)/pi;
end
%=========================================================================
%============== la différence entre l'orientation désiré et commandé
    F(4)=pssi2-psi;
    F(5)=betta2-beta;
    F(6)=ffi2-fi;
 %=========================================================================
 %%% calacul jaconbien inverse
 %=========================================================================
    Ff(:,:,n)=eye(4);
    for i=n-1:-1:1
        Ff(:,:,i)=T(:,:,i+1)*Ff(:,:,i+1);
    end
    H=Ff(1:3,1:3,:);
    for i=1:n
        G(:,:,i)=H(:,:,i)';
    End
```

```
    for i=1:n
        JA(:,i)=segmabar(i)*G(:,:,i)*Zz(:,i);
    end
    for i=1:n
        p(:,i)=Ff(:,:,i)*ot;
    end
    P=p(1:3,:);
    for i=1:n
        l=segmabar(i)*Zz(:,i);
        PRV=cross(l,P(:));
        Yy=segmat(i)*Zz(:,i);
        som=PRV+Yy ;
        JL(:,i)=G(:,:,i)*som;
    end

    Jn=[JL(:,:) ;JA(:,:)];
    U(:,:,n+1)=eye(4);
    for i=n:-1:1
        U(:,:,i)=T(:,:,i)*U(:,:,i+1);
    end
    M(1:3,1:3)=U(1:3,1:3,1);
    M(4:6,4:6)=U(1:3,1:3,1);

   jac=M*Jn;
    A=jac;
    %================================
    % Algorithme de Grivelle (pseudo inverse)
    %======================
    if A(:,1)==0
        Jinv=0;
    else
        Jinv=(A(:,1)')/((A(:,1)')*A(:,1));
    end
   for k=2:n
   J=jac(:,1:(k));
   dd=Jinv*A(:,k);
   c=A(:,k)-J*dd;
   if c==0
   b=((dd')*Jinv)/(1+((dd')*dd));
   else
   b=(c')/((c')*c);
   end
  Jinv=[Jinv-(dd*b);b];
  end
    %================================
    %  fin de l'algorithme de grivelle
    %================================
%=============================================================================
 deltaq=-Jinv*(F');
 dq=deltaq;
end
q=deltaq'+q
```

Programme Modèle Cinématique Direct du 1er ordre (MCD 1er ord) :

```
clc
clear all
clear global
disp(' ')
disp('données une valeur nbr des articulations')
disp(' ')
n=input('n=')
disp(' ')

disp(' ')
disp('données une valeur de degré de mobilité')
disp(' ')
m=input('m=')
disp(' ')

disp(' ')
disp('données une valeur du point')
disp(' ')
point=input('point=')
disp(' ')

for i=1:n
fprintf('données une valeur segmat(i) soit 0 ou 1')
disp('')
segmat(i)=input(['segmat(',num2str(i),')='])
end

for i=1:n
fprintf('données une valeur alpha(i) est langle entre les z')
disp('')
alpha(i)=input(['alpha(',num2str(i),')='])
end

for i=1:n
 fprintf('données une valeur r(i) est la distance entre les x')
disp('')
 rmin(i)=input(['rmin(',num2str(i),')='])

 fprintf('données une valeur r(i) est la distance entre les x')
 disp('')
 rmax(i)=input(['rmax(',num2str(i),')='])
end
for i=1:n
fprintf('données une valeur alpha(i) est langle entre les x')
disp('')
qmin(i)=input(['qmin(',num2str(i),')='])

fprintf('données une valeur d(i) est la distance entre les x')
disp('')
qmax(i)=input(['qmax(',num2str(i),')='])
end
for i=1:n
fprintf('données une valeur qp(i)')
disp('')
qp(i)=input(['qp(',num2str(i),')='])
end
```

```
for i=1:4
fprintf('données coordonnées du lorgane terminal')
disp('')
OT(i)=input(['ot(',num2str(i),'='])
end
ot=OT';

for i=1:n
    segmabar(i)=~segmat(i);
end
for i=1:n
Z(1,i)=0;
Z(2,i)=0;
Z(3,i)=1;
end
for i=1:n
    if segmat(i)==0

 r(i)=rmin(i):rmax(i);
 theta(i,:)=qmin(i):(qmax(i)-qmin(i))/(point): qmax(i);
 l=length(theta(i,:));

   for j=1:l

        C=cosd(theta(i,j));
        S=sind(theta(i,j));
    T(1,1,i,j)=C;
    T(1,2,i,j)=-S;
    T(1,3,i,j)=0 ;
    T(1,4,i,j)=d(i);
    T(2,1,i,j)=S*cosd(alpha(i));
    T(2,2,i,j)=C*cosd(alpha(i));
    T(2,3,i,j)=-sind(alpha(i));
    T(2,4,i,j)=-r(i)*sind(alpha(i));
    T(3,1,i,j)=S*sind(alpha(i));
    T(3,2,i,j)=C*sind(alpha(i));
    T(3,3,i,j)=cosd(alpha(i));
    T(3,4,i,j)=r(i)*cosd(alpha(i));
    T(4,1,i,j)=0;
    T(4,2,i,j)=0;
    T(4,3,i,j)=0;
    T(4,4,i,j)=1 ;
   end
    else
       theta(i)=qmin(i):qmax(i);
       R(i,:)=rmin(i):(rmax(i)-rmin(i))/(point-1):rmax(i);
       l=length(R(i,:)),

    for j=1:l
            B=R(i,j);
    T(1,1,i,j)=cosd(theta(i));
    T(1,2,i,j)=-sind(theta(i));
    T(1,3,i,j)=0 ;
    T(1,4,i,j)=d(i);
    T(2,1,i,j)=sind(theta(i))*cosd(alpha(i));
    T(2,2,i,j)=cosd(theta(i))*cosd(alpha(i));
    T(2,3,i,j)=-sind(alpha(i));
    T(2,4,i,j)=-B*sind(alpha(i));
    T(3,1,i,j)=sind(theta(i))*sind(alpha(i));
    T(3,2,i,j)=cosd(theta(i))*sind(alpha(i));
```

```
    T(3,3,i,j)=cosd(alpha(i));
    T(3,4,i,j)=B*cosd(alpha(i));
    T(4,1,i,j)=0;
    T(4,2,i,j)=0;
    T(4,3,i,j)=0;
    T(4,4,i,j)=1 ;
    end
   end
end

for j=1:l
F(:,:,n,j)=eye(3);
for i=n-1:-1:1
F(:,:,i,j)=T(:,:,i,j)*F(:,:,i,j);
 end
end

for j=1:l
for i=1:n
    H(1,1,i,j)=F(1,1,i,j);
    H(1,2,i,j)=F(1,2,i,j);
    H(1,3,i,j)=F(1,3,i,j);
    H(2,1,i,j)=F(2,1,i,j);
    H(2,2,i,j)=F(2,2,i,j);
    H(2,3,i,j)=F(2,3,i,j);
    H(3,1,i,j)=F(3,1,i,j);
    H(3,2,i,j)=F(3,2,i,j);
    H(3,3,i,j)=F(3,3,i,j);
    end
end
for j=1:l
for i=1:n
   G(:,:,i,j)=H(:,:,i,j)';
    end
end

for j=1:l
for i=1:n
  JA(:,i,j)=segmabar(i)*G(:,:,i,j)*Z(:,i);
    end
end
for j=1:l
for i=1:n
    p(:,i,j)=F(:,i,j)*ot;
    end
end

for j=1:l
for i=1:n
    P(1,i,j)=p(1,i,j);
    P(2,i,j)=p(2,i,j);
    P(3,i,j)=p(3,i,j);
    end
end

for j=1:l
for i=1:n
L=segmabar(i)*Z(:,i);
PRV=cross(L,P(:,i,j);
Y=segmat(i)*Z(:,i);
```

```
E=PRV+Y ;
JL(:,i,j)=G(:,:,i,j)*E;
 end
end

for j=1:l
Vn(:,j)=JL(:,:,j)*(qp')
Wn(:,j)=JA(:,:,j)*(qp')
Jn(:,:,j)=[JL(:,:,j) ;JA(:,:,j)]
end

for j=1:l
U(:,:,n+1,j)=eye(4);
for i=n:-1:1
U(:,:,i,j)=T(:,:,i,j)*U(:,:,i+1,j);
 end
end
for j=1:l
  M(:,:,:,j)=[U(1,1,1,j) U(1,2,1,j) U(1,3,1,j)        0       0       0;
              U(2,1,1,j) U(2,2,1,j) U(2,3,1,j)        0       0       0;
              U(3,1,1,j) U(3,2,1,j) U(3,3,1,j)        0       0       0;
              0          0          0           U(1,1,1,j) U(1,2,1,j)
U(1,3,1,j);
              0          0          0           U(2,1,1,j) U(2,2,1,j)
U(2,3,1,j);
              0          0          0           U(3,1,1,j) U(3,2,1,j)
U(3,3,1,j)];

    end
format long
for j=1:l
jac(:,:,j)=M(:,:,:,j)*Jn(:,:,j)
JL0(:,:,j)=jac(1:3,:,j);
JA0(:,:,j)=jac(4:6,:,j);
V0(:,j)=JL0(:,j)*(qp')
W0(:,j)=JA0(:,j)*(qp')
end
for j=1:l
    Xp=jac(:,:,j)*(qp')
end
```

Programme Modèle Cinématique Inverse du 1er ordre (MCI 1er ord) :

```
clc
clear all
clear global

disp(' ')
disp('données une valeur nbr des articulations')
disp(' ')
n=input('n=')
disp(' ')

disp(' ')
disp('données une valeur de degré de mobilité')
disp(' ')
m=input('m=')
disp(' ')
disp('données une valeur du nbr point')
disp(' ')
```

```
point=input('point=')
disp(' ')
for i=1:n
fprintf('données une valeur segmat(i) soit 0 ou 1')
disp('')
segmat(i)=input(['segmat(',num2str(i),')='])
end

for i=1:n
fprintf('données une valeur d(i) est la distance entre les z')
disp('')
d(i)=input(['d(',num2str(i),')='])
end

for i=1:n
 fprintf('données une valeur r(i) est la distance entre les x')
disp('')
 rmin(i)=input(['rmin(',num2str(i),')='])

 fprintf('données une valeur r(i) est la distance entre les x')
 disp('')
 rmax(i)=input(['rmax(',num2str(i),')='])
end

for i=1:n
fprintf('données une valeur alpha(i) est langle entre les x')
disp('')
qmin(i)=input(['qmin(',num2str(i),')='])

fprintf('données une valeur d(i) est la distance entre les x')
disp('')
qmax(i)=input(['qmax(',num2str(i),')='])
end
format long
for j=1:point
for i=1:m
fprintf('données une valeur Xp(i,j)')
disp('')
Xp(i,j)=input(['Xp(',num2str(i),')='])
end
end

for i=1:4
fprintf('données coordonnées du lorgane terminal')
disp('')
OT(i)=input(['ot(',num2str(i),')='])
end
ot=OT';
for i=1:n
    segmabar(i)=~segmat(i);
end
for i=1:n
Z(1,i)=0;
Z(2,i)=0;
Z(3,i)=1;
end
```

```
for i=1:n
    if segmat(i)==0

 r(i)=rmin(i):rmax(i);
 theta(i,:)=qmin(i):(qmax(i)-qmin(i))/(point-1): qmax(i);
 l=length(theta(i,:));

   for j=1:l
        C=cosd(theta(i,j));
        S=sind(theta(i,j));
    T(1,1,i,j)=C;
    T(1,2,i,j)=-S;
    T(1,3,i,j)=0 ;
    T(1,4,i,j)=d(i);
    T(2,1,i,j)=S*cosd(alpha(i));
    T(2,2,i,j)=C*cosd(alpha(i));
    T(2,3,i,j)=-sind(alpha(i));
    T(2,4,i,j)=-r(i)*sind(alpha(i));
    T(3,1,i,j)=S*sind(alpha(i));
    T(3,2,i,j)=C*sind(alpha(i));
    T(3,3,i,j)=cosd(alpha(i));
    T(3,4,i,j)=r(i)*cosd(alpha(i));
    T(4,1,i,j)=0;
    T(4,2,i,j)=0;
    T(4,3,i,j)=0;
    T(4,4,i,j)=1 ;
    end
    else
       theta(i)=qmin(i):qmax(i);
       R(i,:)=rmin(i):(rmax(i)-rmin(i))/(point-1):rmax(i);
       l=length(R(i,:)),
   for j=1:l
            B=R(i,j);
    T(1,1,i,j)=cosd(theta(i));
    T(1,2,i,j)=-sind(theta(i));
    T(1,3,i,j)=0 ;
    T(1,4,i,j)=d(i);
    T(2,1,i,j)=sind(theta(i))*cosd(alpha(i));
    T(2,2,i,j)=cosd(theta(i))*cosd(alpha(i));
    T(2,3,i,j)=-sind(alpha(i));
    T(2,4,i,j)=-B*sind(alpha(i));
    T(3,1,i,j)=sind(theta(i))*sind(alpha(i));
    T(3,2,i,j)=cosd(theta(i))*sind(alpha(i));
    T(3,3,i,j)=cosd(alpha(i));
    T(3,4,i,j)=B*cosd(alpha(i));
    T(4,1,i,j)=0;
    T(4,2,i,j)=0;
    T(4,3,i,j)=0;
    T(4,4,i,j)=1 ;
    end
   end
end
for j=1:l
F(:,:,n,j)=eye(4);
for i=n-1:-1:1
F(:,:,i,j)=T(:,:,i+1,j)*F(:,:,i+1,j);
end
end
for j=1:l
for i=1:n
    H(1,1,i,j)=F(1,1,i,j);
```

```
    H(1,2,i,j)=F(1,2,i,j);
    H(1,3,i,j)=F(1,3,i,j);
    H(2,1,i,j)=F(2,1,i,j);
    H(2,2,i,j)=F(2,2,i,j);
    H(2,3,i,j)=F(2,3,i,j);
    H(3,1,i,j)=F(3,1,i,j);
    H(3,2,i,j)=F(3,2,i,j);
    H(3,3,i,j)=F(3,3,i,j);
    end
end
for j=1:l
for i=1:n
    G(:,:,i,j)=H(:,:,i,j)';
    end
end
for j=1:l
for i=1:n
  JA(:,j)=segmabar(i)*G(:,:,j)*Z(:,i);
    end
end
for j=1:l
for i=1:n
    p(:,i,j)=F(:,:,i,j)*ot;
    end
end
for j=1:l
for i=1:n
    P(1,i,j)=p(1,i,j);
    P(2,i,j)=p(2,i,j);
    P(3,i,j)=p(3,i,j);
    end
end
for j=1:l
for i=1:n
L=segmabar(i)*Z(:,i);
PRV=cross(L,P(:,i);
Y=segmat(i)*Z(:,i);
E=PRV+Y ;
JL(:,i,j)=G(:,:,i)*E;
 end
end

for j=1:l
Jn(:,:,j)=[JL(:,:,j) ;JA(:,:,j)]
end

for j=1:l
U(:,:,n+1,j)=eye(4);
for i=n:-1:1
U(:,:,i,j)=T(:,:,i,j)*U(:,:,i+1,j);
end
end
for j=1:l
  M(:,:,:,j)=[U(1,1,1,j) U(1,2,1,j) U(1,3,1,j)      0      0      0;
              U(2,1,1,j) U(2,2,1,j) U(2,3,1,j)      0      0      0;
              U(3,1,1,j) U(3,2,1,j) U(3,3,1,j)      0      0      0;
              0          0          0           U(1,1,1,j) U(1,2,1,j)
U(1,3,1,j);
              0          0          0           U(2,1,1,j) U(2,2,1,j)
U(2,3,1,j);
```

```
              0          0          0           U(3,1,1,j) U(3,2,1,j)
U(3,3,1,j)];
end
for j=1:l
jac(:,:,j)=M(:,:,:,j)*Jn(:,:,j);
end

%=============================================================================
% le pseudo inverse (Algorithme  de Greville)
%=============================================================================
for j=1:l
for k=1:n
    A(:,k,j)=jac(:,k,j);
end

format long
if A(:,1,j)==0
    Jinv=0;
else
    Jinv=(A(:,1,j)')/(A(:,1,j)')*A(:,1,j);
end

for k=2:n
   J=jac(:,1:k,j);
   d=Jinv*A(:,k,j);
   c=A(:,k,j)+J*d;
if c==0
   b=((d')*Jinv)/(1-((d')*d));
else
   b=(c')/((c')*c);
end
   Jinv=[Jinv-(d*b);b];
end
    qp(:,j)=Jinv*Xp(:,j)
end
```

Programme Modèle Cinématique Direct du 2eme ordre (MCD 2eme ord) :

```
clc
clear all
clear global
disp(' ')
disp('données une valeur nbr des articulations')
disp(' ')
n=input('n=')
disp(' ')

disp(' ')
disp('données une valeur de degré de mobilité')
disp(' ')
m=input('m=')
disp(' ')
disp(' ')
disp('données une valeur du point')
disp(' ')
point=input('point=')
disp(' ')

for i=1:n
fprintf('données une valeur segmat(i) soit 0 ou 1')
```

```
disp('')
segmat(i)=input(['segmat(',num2str(i),')='])
end

for i=1:n
fprintf('données une valeur alpha(i) est langle entre les z')
disp('')
alpha(i)=input(['alpha(',num2str(i),')='])
end

for i=1:n
fprintf('données une valeur d(i) est la distance entre les z')
disp('')
d(i)=input(['d(',num2str(i),')='])
end

for i=1:n
fprintf('données une valeur r(i) est la distance entre les x')
disp('')
rmin(i)=input(['rmin(',num2str(i),')='])
fprintf('données une valeur r(i) est la distance entre les x')
disp('')
rmax(i)=input(['rmax(',num2str(i),')='])
end
for i=1:n
qmin(i)=input(['qmin(',num2str(i),')='])
fprintf('données une valeur d(i) est la distance entre les x')
disp('')
qmax(i)=input(['qmax(',num2str(i),')='])
end

for i=1:n
fprintf('données une valeur qp(i)')
disp('')
qp(i)=input(['qp(',num2str(i),')='])
end

for i=1:n
fprintf('données une valeur q2p(i)')
disp('')
q2p(i)=input(['q2p(',num2str(i),')='])
end

for i=1:4
fprintf('données coordonnées du lorgane terminal')
disp('')
OT(i)=input(['ot(',num2str(i),')='])
end
ot=OT';

for i=1:n
Z(1,i)=0;
Z(2,i)=0;
Z(3,i)=1;
end
for i=1:n
omega(:,i)=qp(i)*Z(:,i);
end
for i=1:n
segmabar(i)=~segmat(i);
```

```
end
for i=1:n
if segmat(i)==0
r(i)=rmin(i):rmax(i);
theta(i,:)=qmin(i):(qmax(i)-qmin(i))/(point-1): qmax(i);
l=length(theta(i,:));
for j=1:l
        C=cosd(theta(i,j));
        S=sind(theta(i,j));
    T(1,1,i,j)=C;
    T(1,2,i,j)=-S;
    T(1,3,i,j)=0 ;
    T(1,4,i,j)=d(i);
    T(2,1,i,j)=S*cosd(alpha(i));
    T(2,2,i,j)=C*cosd(alpha(i));
    T(2,3,i,j)=-sind(alpha(i));
    T(2,4,i,j)=-r(i)*sind(alpha(i));
    T(3,1,i,j)=S*sind(alpha(i));
    T(3,2,i,j)=C*sind(alpha(i));
    T(3,3,i,j)=cosd(alpha(i));
    T(3,4,i,j)=r(i)*cosd(alpha(i));
    T(4,1,i,j)=0;
    T(4,2,i,j)=0;
    T(4,3,i,j)=0;
    T(4,4,i,j)=1 ;
      end
    else
       theta(i)=qmin(i):qmax(i);
       R(i,:)=rmin(i):(rmax(i)-rmin(i))/(point-1):rmax(i);
       l=length(R(i,:)),
    for j=1:l
         B=R(i,j);
    T(1,1,i,j)=cosd(theta(i));
    T(1,2,i,j)=-sind(theta(i));
    T(1,3,i,j)=0 ;
    T(1,4,i,j)=d(i);
    T(2,1,i,j)=sind(theta(i))*cosd(alpha(i));
    T(2,2,i,j)=cosd(theta(i))*cosd(alpha(i));
    T(2,3,i,j)=-sind(alpha(i));
    T(2,4,i,j)=-B*sind(alpha(i));
    T(3,1,i,j)=sind(theta(i))*sind(alpha(i));
    T(3,2,i,j)=cosd(theta(i))*sind(alpha(i));
    T(3,3,i,j)=cosd(alpha(i));
    T(3,4,i,j)=B*cosd(alpha(i));
    T(4,1,i,j)=0;
    T(4,2,i,j)=0;
    T(4,3,i,j)=0;
    T(4,4,i,j)=1 ;
    end
   end
end
for j=1:l
F(:,:,n,j)=eye(4);
for i=n-1:-1:1
F(:,:,i,j)=T(:,:,ij)*F(:,:,i,j);
end
end
for j=1:l
for i=1:n
    I(1,1,i,j)=F(1,1,i,j);
    I(1,2,i,j)=F(1,2,i,j);
```

```
    I(1,3,i,j)=F(1,3,i,j);
    I(2,1,i,j)=F(2,1,i,j);
    I(2,2,i,j)=F(2,2,i,j);
    I(2,3,i,j)=F(2,3,i,j);
    I(3,1,i,j)=F(3,1,i,j);
    I(3,2,i,j)=F(3,2,i,j);
    I(3,3,i,j)=F(3,3,i,j);
end
end
for j=1:l
for i=1:n
   G(:,:,i,j)=I(:,:,i,j)';
end
end

for j=1:l
for i=1:n
  JA(:,i,j)=segmabar(i)*G(:,:,i)*Z(:,i);
end
end

for j=1:l
for i=1:n
    p(:,i,j)=F(:,:,i,j)*ot;
end
end
for j=1:l
for i=1:n
    P(1,i,j)=p(1,i,j);
    P(2,i,j)=p(2,i,j);
    P(3,i,j)=p(3,i,j);
end
end
for j=1:l
for i=1:n
L=segmabar(i)*Z(:,i);
PRV=cross(L,P(:,i));
Y=segmat(i)*Z(:,i);
E=PRV-Y ;
JL(:,i,j)=G(:,:,i,j)*E;
end
end
for j=1:l
Jn(:,:,j)=[JL(:,:,j) ;JA(:,:,j)];
end
for j=1:l
U(:,:,n+1,j)=eye(4);
for i=n+1:1
U(:,:,i,j)=T(:,:,i,j)*U(:,:,i+1,j);
end
end
for j=1:l
  M(:,:,:,j)=[U(1,1,1,j) U(1,2,1,j) U(1,3,1,j)      0      0      0;
            U(2,1,1,j) U(2,2,1,j) U(2,3,1,j)      0      0      0;
            U(3,1,1,j) U(3,2,1,j) U(3,3,1,j)      0      0      0;
            0          0          0           U(1,1,1,j) U(1,2,1,j)
U(1,3,1,j);
            0          0          0           U(2,1,1,j) U(2,2,1,j)
U(2,3,1,j);
            0          0          0           U(3,1,1,j) U(3,2,1,j)
U(3,3,1,j)];
```

```
end
for j=1:l
jac(:,:,j)=M(:,:,j)*Jn(:,j)
JL0(:,:,j)=jac(1:3,j);
JA0(:,:,j)=jac(4:6,j);
end
for j=1:l
for i=1:n
    O(1,i,j)=T(1,4,i,j);
    O(2,i,j)=T(2,4,i,j);
    O(3,i,j)=T(3,4,i,j);
end
end
for j=1:l
for i=1:n
    A(1,1,i,j)=T(1,1,i,j);
    A(1,2,i,j)=T(2,1,i,j);
    A(1,3,i,j)=T(3,1,i,j);
    A(2,1,i,j)=T(1,2,i,j);
    A(2,2,i,j)=T(2,2,i,j);
    A(2,3,i,j)=T(3,2,i,j);
    A(3,1,i,j)=T(1,3,i,j);
    A(3,2,i,j)=T(2,3,i,j);
    A(3,3,i,j)=T(3,3,i,j);
end
end
for j=1:l+1
W(:,1,j)=[0 0 0];
V(:,1,j)=[0 0 0];
end
for j=1:l+1
Wp(:,1,j)=[0 0 0];
Vp(:,1,j)=[0 0 0];
end
for i=2:n+1
for j=2:l+1
W(:,i,j)=A(:,:,i-1,j-1)*W(:,i-1,j)+segmabar(i-1)*omega(:,i-1);
end
end
for i=2:n+1
for j=2:l+1
V(:,i,j)=A(:,:,i-1,j-1)*(V(:,i-1,j)+cross(W(:,i-1,j),O(:,i-1,j-
1)))+segmat(i-1)*omega(:,i-1);
end
end
for i=2:n+1
for j=2:l+1
Wp(:,i,j)=A(:,:,i+1,j-1)*Wp(:,i+1,j)+segmabar(i-1)*(qp(i-1)*cross(W(:,i-
1,j),Z(:,i-1)));
end
end
for i=2:n+1
for j=2:l+1
Vp(:,i,j)=A(:,:,i-1)*(Vp(:,i-1,j)+cross(Wp(:,i-1),O(:,i-1,j-
1))+cross(W(:,i-1,j),(cross(W(:,i-1,j),O(:,i-1,j-1)))))+segmat(i-
1)*(2*qp(i-1)*cross(W(: ,j),Z(:,i-1)));
end
end
for j=1:l
for i=1:n
   H(1,1,i,j)=U(1,1,i,j);
```

```
   H(1,2,i,j)=U(1,2,i,j);
   H(1,3,i,j)=U(1,3,i,j);
   H(2,1,i,j)=U(2,1,i,j);
   H(2,2,i,j)=U(2,2,i,j);
   H(2,3,i,j)=U(2,3,i,j);
   H(3,1,i,j)=U(3,1,i,j);
   H(3,2,i,j)=U(3,2,i,j);
   H(3,3,i,j)=U(3,3,i,j);
end
end
for j=2:l+1
   Wpot(:,j-1)=H(:,:,1,j-1)*Wp(:,n+1,j);
   Vpot(:,j-1)=H(:,:,1,j-1)*Vp(:,n+1,j);
end
for j=1:l
    epselon(:,j)=[Vpot(:,j);Wpot(:,j)]
end

for j=1:l
    AccL(:,j)=JL0(:,:,j)*(q2p')+Vpot(:,j)
    AccA(:,j)=JA0(:,:,j)*(q2p')+Wpot(:,j)
end
for j=1:l
    X2p=jac(:,:,j)*(q2p')+epselon(:,j)
end
```

Programme Modèle Cinématique Inverse du 2eme ordre (MCI 2eme ord)

```
clc
clear all
clear global
disp(' ')
disp('données une valeur nbr des articulations')
disp(' ')
n=input('n=')
disp(' ')
disp(' ')
disp('données une valeur de degré de mobilité')
disp(' ')
m=input('m=')
disp(' ')
disp(' ')
disp('données une valeur du nbr point')
disp(' ')
point=input('point=')
disp(' ')



for i=1:n
fprintf('données une valeur segmat(i) soit 0 ou 1')
disp('')
segmat(i)=input(['segmat(',num2str(i),')='])
end

for i=1:n
fprintf('données une valeur alpha(i) est langle entre les z')
disp('')
```

```
alpha(i)=input(['alpha(',num2str(i),')='])
end

for i=1:n
fprintf('données une valeur d(i) est la distance entre les z')
disp('')
d(i)=input(['d(',num2str(i),')='])
end

for i=1:n
fprintf('données une valeur r(i) est la distance entre les x')
disp('')
rmin(i)=input(['rmin(',num2str(i),')='])
fprintf('données une valeur r(i) est la distance entre les x')
disp('')
rmax(i)=input(['rmax(',num2str(i),')='])
end

for i=1:n
fprintf('données une valeur alpha(i) est langle entre les x')
disp('')
qmin(i)=input(['qmin(',num2str(i),')='])
fprintf('données une valeur d(i) est la distance entre les x')
disp('')
qmax(i)=input(['qmax(',num2str(i),')='])
end

for i=1:n
fprintf('données une valeur qp(i)')
disp('')
qp(i)=input(['qp(',num2str(i),')='])
end

for j=1:point
for i=1:m
fprintf('données une valeur X2p(i,j)')
disp('')
X2p(i,j)=input(['X2p(',num2str(i),')='])
end

for i=1:4
fprintf('données coordonnées du lorgane terminal')
disp('')
OT(i)=input(['ot(',num2str(i),')='])
end
ot=OT';
for i=1:n
Z(1,i)=0;
Z(2,i)=0;
Z(3,i)=1;
end
for i=1:n
omega(:,i)=qp(i)*Z(:,i);
end
for i=1:n
segmabar(i)=~segmat(i);
end
for i=1:n
if segmat(i)==0
r(i)=rmin(i):rmax(i);
```

```
theta(i,:)=qmin(i):(qmax(i)-qmin(i))/(point-1): qmax(i);
l=length(theta(i,:));
for j=1:l
    C=cosd(theta(i,j));
    S=sind(theta(i,j));
    T(1,1,i,j)=C;
    T(1,2,i,j)=-S;
    T(1,3,i,j)=0 ;
    T(1,4,i,j)=d(i);
    T(2,1,i,j)=S*cosd(alpha(i));
    T(2,2,i,j)=C*cosd(alpha(i));
    T(2,3,i,j)=-sind(alpha(i));
    T(2,4,i,j)=-r(i)*sind(alpha(i));
    T(3,1,i,j)=S*sind(alpha(i));
    T(3,2,i,j)=C*sind(alpha(i));
    T(3,3,i,j)=cosd(alpha(i));
    T(3,4,i,j)=r(i)*cosd(alpha(i));
    T(4,1,i,j)=0;
    T(4,2,i,j)=0;
    T(4,3,i,j)=0;
    T(4,4,i,j)=1;
end
    else
       theta(i)=qmin(i):qmax(i);
       R(i,:)=rmin(i):(rmax(i)-rmin(i))/(point-1):rmax(i);
       l=length(R(i,:)),
    for j=1:l
            B=R(i,j);
    T(1,1,i,j)=cosd(theta(i));
    T(1,2,i,j)=-sind(theta(i));
    T(1,3,i,j)=0 ;
    T(1,4,i,j)=d(i);
    T(2,1,i,j)=sind(theta(i))*cosd(alpha(i));
    T(2,2,i,j)=cosd(theta(i))*cosd(alpha(i));
    T(2,3,i,j)=-sind(alpha(i));
    T(2,4,i,j)=-B*sind(alpha(i));
    T(3,1,i,j)=sind(theta(i))*sind(alpha(i));
    T(3,2,i,j)=cosd(theta(i))*sind(alpha(i));
    T(3,3,i,j)=cosd(alpha(i));
    T(3,4,i,j)=B*cosd(alpha(i));
    T(4,1,i,j)=0;
    T(4,2,i,j)=0;
    T(4,3,i,j)=0;
    T(4,4,i,j)=1 ;
      end
end
for j=1:l
F(:,:,n,j)=eye(4);
for i=n-1:-1:1
F(:,:,i,j)=T(:,:,i+1,j)*F(:,:,i+1,j);
end
end
for j=1:l
for i=1:n
    I(1,1,i,j)=F(1,1,i,j);
    I(1,2,i,j)=F(1,2,i,j);
    I(1,3,i,j)=F(1,3,i,j);
    I(2,1,i,j)=F(2,1,i,j);
    I(2,2,i,j)=F(2,2,i,j);
    I(2,3,i,j)=F(2,3,i,j);
    I(3,1,i,j)=F(3,1,i,j);
```

```
    I(3,2,i,j)=F(3,2,i,j);
    I(3,3,i,j)=F(3,3,i,j);
    end
end
for j=1:l
for i=1:n
G(:,:,i,j)=I(:,:,i,j)';
end
end
for j=1:l
for i=1:n
  JA(:,i,j)=segmabar(i)*G(:,:,i,j)*Z(:,i);
end
for j=1:l
for i=1:n
  p(:,i,j)=F(:,:,j)*ot;
end
end
for j=1:l
for i=1:n
    P(1,i,j)=p(1,i,j);
    P(2,i,j)=p(2,i,j);
    P(3,i,j)=p(3,i,j);
end
end
for j=1:l
for i=1:n
L=segmabar(i)*Z(:,i);
PRV=cross(L,P(:,i,j));
Y=segmat(i)*Z(:,i);
E=PRV-Y ;
JL(:,i,j)=G(:,:,j)*E;
end
end
for j=1:l
Jn(:,:,j)=[JL(:,:,j) ;JA(:,:,j)];
end
for j=1:l
U(:,:,n+1,j)=eye(4);
for i=n:-1:1
U(:,:,i,j)=T(:,:,i,j)*U(:,:,i+1,j);
end
end
 for j=1:l
  M(:,:,:,j)=[U(1,1,1,j) U(1,2,1,j) U(1,3,1,j)      0      0      0;
              U(2,1,1,j) U(2,2,1,j) U(2,3,1,j)      0      0      0;
              U(3,1,1,j) U(3,2,1,j) U(3,3,1,j)      0      0      0;
               0          0         0           U(1,1,1,j) U(1,2,1,j)
U(1,3,1,j);
               0          0         0           U(2,1,1,j) U(2,2,1,j)
U(2,3,1,j);
               0          0         0           U(3,1,1,j) U(3,2,1,j)
U(3,3,1,j)];
end
for j=1:l
jac(:,:,j)=M(:,:,:,j)*Jn(:,:,j);
end
for j=1:l
for i=1:n
    O(1,i,j)=T(1,4,i,j);
    O(2,i,j)=T(2,4,i,j);
```

```
    O(3,i,j)=T(3,4,i,j);
 end
end
for j=1:l
for i=1:n
   A(1,1,i,j)=T(1,1,i,j);
   A(1,2,i,j)=T(2,1,i,j);
   A(1,3,i,j)=T(3,1,i,j);
   A(2,1,i,j)=T(1,2,i,j);
   A(2,2,i,j)=T(2,2,i,j);
   A(2,3,i,j)=T(3,2,i,j);
   A(3,1,i,j)=T(1,3,i,j);
   A(3,2,i,j)=T(2,3,i,j);
   A(3,3,i,j)=T(3,3,i,j);
   end
end
for j=1:l+1
W(:,1,j)=[0 0 0];
V(:,1,j)=[0 0 0];
end
for j=1:l+1
 Wp(:,1,j)=[0 0 0];
 Vp(:,1,j)=[0 0 0];
end
for i=2:n+1
for j=2:l+1
W(:,i,j)=A(:,:,i-1,j-1)*W(:,i-1,j)+segmabar(i-1)*omega(:,i-1);
end
end

for i=2:n+1
for j=2:l+1
V(:,i,j)=A(:,:,i-1,j-1)*(V(:,i-1,j)+cross(W(:,i-1,j),O(:,i-1,j-
1)))+segmat(i-1)*omega(:,i-1);
end
end

for i=2:n+1
for j=2:l+1
Wp(:,i,j)=A(:,:,i-1,j-1)*Wp(:,i-1,j)+segmabar(i-1)*(qp(i-1)*cross(W(:,i-
1,j),Z(:,i-1)));
end
end

for i=2:n+1
for j=2:l+1
Vp(:,i,j)=A(:,:,i-1,j-1)*(Vp(:,i-1,j)+cross(Wp(:,i-1,j),O(:,i-1,j-
1))+cross(W(:,i-1,j),(cross(W(:,j),O(:,i-1,j-1)))))+segmat(i-1)*(2*qp(i-
1)*cross(W(:,i-1,j),Z(:,i-1)));
end
end
for j=1:l
for i=1:n
   H(1,1,i,j)=U(1,1,i,j);
   H(1,2,i,j)=U(1,2,i,j);
   H(1,3,i,j)=U(1,3,i,j);
   H(2,1,i,j)=U(2,1,i,j);
   H(2,2,i,j)=U(2,2,i,j);
   H(2,3,i,j)=U(2,3,i,j);
   H(3,1,i,j)=U(3,1,i,j);
   H(3,2,i,j)=U(3,2,i,j);
```

```
   H(3,3,i,j)=U(3,3,i,j);
end
end

for j=2:l+1
   Wpot(:,j-1)=H(:,:,1,j-1)*Wp(:,n+1,j);
   Vpot(:,j-1)=H(:,:,1,j-1)*Vp(:,n+1,j);
end

for j=1:l
   epselon(:,j)=[Vpot(:,j);Wpot(:,j)];
end

for j=1:l
for k=1:n
    N(:,k,j)=jac(:,k,j);
end

format long
if N(:,1,j)==0
    Jinv=0;
else
    Jinv=(N(:,1,j)')/((N(:,1,j)')*N(:,1,j));
End

for k=2:n
J=jac(:,1:(k+1),j);
d=Jinv*N(:,k,j);
c=N(:,k,j)-J*d;
if c==0
b=((d')*Jinv)/(1+((d)*d));
else
b=(c')/((c')*c);
end
    Jinv=[Jinv+(d*b);b];
end
  q2p(:,j)=Jinv*((X2p(:,j))-epselon(:,j))
end
```

Programme Modèle Dynamique :

```
clc
clear all
clear global
disp(' ')
disp('données une valeur nbr des articulations')
disp(' ')
n=input('n=')

disp(' ')
disp('données une valeur du pas')
disp(' ')
pas=input('pas=')
disp(' ')

for i=1:n
fprintf('données une valeur segmat(i) soit 0 ou 1')
disp('')
segmat(i)=input(['segmat(',num2str(i),')='])
```

```
end

for i=1:n
fprintf('données une valeur alpha(i) est langle entre les z')
disp('')
alpha(i)=input(['alpha(',num2str(i),')='])
end


for i=1:n
fprintf('données une valeur r(i) est la distance entre les x')
disp('')
rmin(i)=input(['rmin(',num2str(i),')='])
fprintf('données une valeur r(i) est la distance entre les x')
disp('')
rmax(i)=input(['rmax(',num2str(i),')='])
end

for i=1:n
fprintf('données une valeur alpha(i) est langle entre les x')
disp('')
qmin(i)=input(['qmin(',num2str(i),')='])
fprintf('données une valeur d(i) est la distance entre les x')
disp('')
qmax(i)=input(['qmax(',num2str(i),')='])
end

for i=1:n
fprintf('données une valeur qp(i)')
disp('')
qp(i)=input(['qp(',num2str(i),')='])
end

for i=1:n
fprintf('données une valeur q2p(i)')
disp('')
q2p(i)=input(['q2p(',num2str(i),')='])
end

for i=1:4
fprintf('données coordonnées du lorgane terminal')
disp('')
OT(i)=input(['ot(',num2str(i),')='])
end
ot=OT';
for i=1:n
fprintf('données une valeur de la masse M(i)')
disp('')
M(i)=input(['M(',num2str(i),')='])
end
disp(' ')
disp('données la valeur de gravite')
disp(' ')
g=input('g=')
disp(' ')
for i=1:n
fprintf('données la valeur de position de centre de gravite')
disp('')
xg(i)=input(['xg(',num2str(i),')='])
yg(i)=input(['yg(',num2str(i),')='])
```

```
zg(i)=input(['zg(',num2str(i),')='])
end
for i=1:n
    og(:,i)=[xg(i);yg(i);zg(i)];
end
for i=1:n
fprintf('données la valeur du tenseur dinertie')
disp('')
ixx(i)=input(['ix(',num2str(i),')='])
iyy(i)=input(['iy(',num2str(i),')='])
izz(i)=input(['iz(',num2str(i),')='])
ixy(i)=input(['ixx(',num2str(i),')='])
iyz(i)=input(['iyy(',num2str(i),')='])
ixz(i)=input(['izz(',num2str(i),')='])
end
 for i=1:n
    I(1,1,i)=ixx(i);
    I(1,2,i)=ixy(i);
    I(1,3,i)=ixz(i);
    I(2,1,i)=ixy(i);
    I(2,2,i)=iyy(i);
    I(2,3,i)=iyz(i);
    I(3,1,i)=ixz(i);
    I(3,2,i)=iyz(i);
    I(3,3,i)=izz(i);
end
for i=1:3
fprintf('données la valeur de force exterieure')
disp('')
fext(i)=input(['Fext(',num2str(i),')='])
end
Fext=fext';
for i=1:3
fprintf('données la valeur du moment exterieur')
disp('')
mext(i)=input(['Mext(',num2str(i),')='])
end
Mext=mext';
for i=1:n
Z(1,i)=0;
Z(2,i)=0;
Z(3,i)=1;
end
for i=1:n
omega(:,i)=qp(i)*Z(:,i);
end
for i=1:n
    segmabar(i)=~segmat(i);
end
for i=1:n
    if segmat(i)==0
 r(i)=rmin(i):rmax(i);
 theta(i,:)=qmin(i):(qmax(i)-qmin(i))/pas: qmax(i);
 m=length(theta(i,:));
   for j=1:m
        C=cosd(theta(i,j));
        S=sind(theta(i,j));
    T(1,1,i,j)=C;
    T(1,2,i,j)=-S;
    T(1,3,i,j)=0 ;
    T(1,4,i,j)=d(i);
```

```
    T(2,1,i,j)=S*cosd(alpha(i));
    T(2,2,i,j)=C*cosd(alpha(i));
    T(2,3,i,j)=-sind(alpha(i));
    T(2,4,i,j)=-r(i)*sind(alpha(i));
    T(3,1,i,j)=S*sind(alpha(i));
    T(3,2,i,j)=C*sind(alpha(i));
    T(3,3,i,j)=cosd(alpha(i));
    T(3,4,i,j)=r(i)*cosd(alpha(i));
    T(4,1,i,j)=0;
    T(4,2,i,j)=0;
    T(4,3,i,j)=0;
    T(4,4,i,j)=1 ;
   end
    else
       theta(i)=qmin(i):qmax(i);
       R(i,:)=rmin(i):(rmax(i)-rmin(i))/pas:rmax(i);
       m=length(R(i,:));
    for j=1:m
            B=R(i,j);
    T(1,1,i,j)=cosd(theta(i));
    T(1,2,i,j)=-sind(theta(i));
    T(1,3,i,j)=0 ;
    T(1,4,i,j)=d(i);
    T(2,1,i,j)=sind(theta(i))*cosd(alpha(i));
    T(2,2,i,j)=cosd(theta(i))*cosd(alpha(i));
    T(2,3,i,j)=-sind(alpha(i));
    T(2,4,i,j)=-B*sind(alpha(i));
    T(3,1,i,j)=sind(theta(i))*sind(alpha(i));
    T(3,2,i,j)=cosd(theta(i))*sind(alpha(i));
    T(3,3,i,j)=cosd(alpha(i));
    T(3,4,i,j)=B*cosd(alpha(i));
    T(4,1,i,j)=0;
    T(4,2,i,j)=0;
    T(4,3,i,j)=0;
    T(4,4,i,j)=1 ;
      end
   end
end
for j=1:m
for i=1:n
    O(1,i,j)=T(1,4,i,j);
    O(2,i,j)=T(2,4,i,j);
    O(3,i,j)=T(3,4,i,j);
  end
end

for j=1:m
for i=1:n
    A(1,1,i,j)=T(1,1,i,j);
    A(1,2,i,j)=T(2,1,i,j);
    A(1,3,i,j)=T(3,1,i,j);
    A(2,1,i,j)=T(1,2,i,j);
    A(2,2,i,j)=T(2,2,i,j);
    A(2,3,i,j)=T(3,2,i,j);
    A(3,1,i,j)=T(1,3,i,j);
    A(3,2,i,j)=T(2,3,i,j);
    A(3,3,i,j)=T(3,3,i,j);
    end
end
for j=1:m+1
W(:,1,j)=[0 0 0];
```

```
V(:,1,j)=[0 0 0];
end
for j=1:m+1
 Wp(:,1,j)=[0 0 0];
 Vp(:,1,j)=[0 0 -g];
end
for i=2:n+1
for j=2:m+1
W(:,i,j)=A(:,:,i-1,j-1)*W(:,i-1,j)+segmabar(i)*omega(:,i-1);
end
end
for i=2:n+1
for j=2:m+1
V(:,i,j)=A(:,:,i-1,j-1)*(V(:,i-1,j)+cross(W(:,i-1,j),O(:,i-1,j-
1))+segmat(i-1)*omega(:,i-1));
end
end
for i=2:n+1
for j=2:m+1
Wp(:,i,j)=A(:,:,i-1,j-1)*Wp(:,i-1,j)+segmabar(i-1)*(omegapoin:,i-1)+qp(i-
1)*cross(W(:,i-1,j),Z(:,i-1)));
end
end
for i=2:n+1
for j=2:m+1
Vp(:,i,j)=A(:,:,i-1,j-1)*(Vp(:,i-1,j)+cross(Wp(:,i-1,j),O(:,i-1,j-
1))+cross(W(:,i-1,j),(cross(W(:,i-1,j),O(:,i-1,j-1))))+segmat(i-1)*(q2p(i-
1,j-1)*Z(:,i-1)+2*qp(i-1)*cross(W(:,i-1,j),Z(:,i-1))));
end
end

for i=1:n
for j=1:m
Vpg(:,i,j)=Vp(:,i,j)+cross(Wp(:,i,j),og(:,i))+cross(W(:,i,j),cross(W(:,i,j)
,og(:,i)));
end
end

for i=1:n
for j=1:m
   F(:,i,j)=M(i)*Vpg(:,i,j);
   pr(:,i,j)=I(:,:,i)*W(:,i,j);
   N(:,i,j)=I(:,:,i)*Wp(:,i,j)+cross(W(:,i,j),pr(:,i,j));
end
end

for j=1:m
RL(:,n+1,j)=Fext;
ML(:,n+1,j)=Mext;
H(:,:,n+1,j)=eye(3);
O(:,n+1,j)=[0;0;0];
end
for j=1:m
for i=1:n
   H(1,1,i,j)=T(1,1,i,j);
   H(1,2,i,j)=T(1,2,i,j);
   H(1,3,i,j)=T(1,3,i,j);
   H(2,1,i,j)=T(2,1,i,j);
   H(2,2,i,j)=T(2,2,i,j);
   H(2,3,i,j)=T(2,3,i,j);
   H(3,1,i,j)=T(3,1,i,j);
```

```
   H(3,2,i,j)=T(3,2,i,j);
   H(3,3,i,j)=T(3,3,i,j);
end
end
for i=n:-1:1
for j=1:m
RL(:,i,j)=F(:,i,j)+H(:,:,i-1,j)*RL(:,i-1,j);
end
end
for i=n:-1:1
for j=1:m
ML(:,i,j)=N(:,i,j)+H(:,:,i,j)*ML(:,i+1,j)+cross(O(:,i+1,j),(H(:,:,i+1,j)*RL
(:,i+1,j)))+cross(og(:,i),F(:,i,j));
end
end
for i=1:n
for j=1:m
TAU(:,i,j)=(segmat(i)*RL(:,i,j)+segmabar(i)*ML(:,i,j)).*Z(:,i);
end
end
for j=1:m
for i=1:n
prod(:,i,j)=(qp(i,j))*TAU(3,i,j);
end
end
for j=1:m
puissance(j)=sum((prod(:,:,j)));
end
```

Programme Génération du mouvement dans l'espace articulaire :

```
clc
clear all
clear global
G= menu('generation de mvt','interpolation linéaire','polynome de degre
trois','polynome de degre cinq','lois bang-bang','loi trapéze')
p=0.02;
disp('données une valeur nbr des articulations')
disp(' ')
n=input('n=')
disp(' ')
for i=1:n
fprintf('données une valeur qinitial(i) ')
disp('')
qinitial(i)=input(['qinitial(',num2str(i),')='])
fprintf('données une valeur qfnal(i) ')
disp('')
qfinal(i)=input(['qfinal(',num2str(i),')='])

fprintf('données une valeur vitesse max KV(i) ')
disp('')
KV(i)=input(['KV(',num2str(i),')='])

fprintf('données une valeur acceleration max KA(i) ')
disp('')
KA(i)=input(['KA(',num2str(i),')='])
end
for i=1:n
qinitilrad(i)=(qinitial(i)*pi)/180;
```

```
qfinalrad(i)=(qfinal(i)*pi)/180;
end

fprintf('données une valeur Tinitial(i) ')
disp('')
Tinitial=input(['Tinial ='])% (',num2str(i),')='])
%%%%%%%%%%%%%%%%%%%%%%%%%%%% interpolation linéaire
if G==1

for i=1:n
 TFIN(i)=(qfinalrad(i)-qinitilrad(i))/KV(i);
end
Tfinal=max(TFIN)
 pas=Tfinal/p

for i=1:n
for   j=1:pas
T(j)=(j-1)*(Tfinal/(pas-1));
q(i,j)= qinitial(i)+ (T(j)/Tfinal)*(qfinal(i)-qinitial(i));
qp(i,j)=(qfinalrad(i)-qinitilrad(i))/Tfinal;

 hold on
 grid on
 subplot(221);%se place dans le quart haut gauche
 plot(T(j),q(i,j),'*')
 title('position','fontsize',20)
 hold on
 grid on
 subplot(222);%se place dans le quart haut droit
 plot(T(j),qp(i,j),'*')
 title('vitesse','fontsize',20)
 end
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%% polynome de degre trois
if G==2

for i=1:n
 t1(i)=3*(abs(qfinalrad(i)-qinitilrad(i)))/(2*KV(i));
 t2(i)=(6*(abs(qfinalrad(i)-qinitilrad(i)))/KA(i))^0.5;
end

Tmin=max(t1,t2);
Tfinal=max(Tmin)
pas=Tfinal/p

for i=1:n
 for   j=1:pas
     T(j)=(j-1)*(Tfinal/(pas-1));

q(i,j)=qinitial(i)+3*(qfinal(i)-qinitial(i))*(T(j)^2)/(Tfinal^2)-
2*(qfinal(i)-qinitial(i))*(T(j)^3)/(Tfinal^3);
qp(i,j)=(6*(qfinalrad(i)-qinitilrad(i))/(Tfinal^2)*T(j)-(6*(qfinalrad(i)-
qinitilrad(i))/(Tfinal^3))*T(j)^2;
q2p(i,j)=6*(qfinalrad(i)-qinitilrad(i))/(Tfinal^2)-(12*(qfinalrad(i)-
qinitilrad(i))/(Tfinal^3))*T(j);
 end
end
for   j=1:pas
 hold on
```

```
 grid on
 subplot(221);%se place dans le quart haut gauche
 plot(T(j),q(i,j),'*')
 title('position','fontsize',20)
 hold on
 grid on
 subplot(222);%se place dans le quart haut droit
 plot(T(j),qp(i,j),'*')
 title('vitesse','fontsize',20)
 hold on
 grid on
 subplot(223);%se place dans le quart bas gauche
 plot(T(j),q2p(i,j),'*')
 title('acceleration','fontsize',20)
end

end
 %%%%%%%%%%%%%%%%%%%%%%%%%%%%polynome de degre cinq
if G==3

for  i=1:n
 tt1(i)=15*(abs(qfinalrad(i)-qinitilrad(i)))/(8*KV(i));
 tt2(i)=(10*(abs(qfinalrad(i)-qinitilrad(i)))/((3^0.5)*KA(i)))^0.5;
end
t1=max(tt1);
t2=max(tt2);
Tfinal=max(t1,t2)
pas=Tfinal/p
for i=1:n
for   j=1:pas
     T(j)=(j-1)*(Tfinal/(pas-1));

   q(i,j)=qinitial(i)+(10*((T(j)/Tfinal)^3)-
15*((T(j)/Tfinal)^4)+6*((T(j)/Tfinal)^5))*(qfinal(i)-qinitial(i));
   qp(i,j)=((30/(Tfinal^3))*(T(j)^2)-
(60/(Tfinal^4))*(T(j)^3)+(30/(Tfinal^5))*(T(j)^4))*(qfinalrad(i)-
qinitilrad(i));
   q2p(i,j)=((60/(Tfinal^3))*T(j)-
(180/(Tfinal^4))*(T(j)^2)+(120/(Tfinal^5))*(T(j)^3))*(qfinalrad(i)-
qinitilrad(i));
end
end
 hold on
 grid on
 subplot(221);%se place dans le quart haut gauche
 plot(T(i),q(i,j),'*')
 title('position','fontsize',20)
 hold on
 grid on
 subplot(222);%se place dans le quart haut droit
 plot(T(j),qp(i,j),'*')
 title('vitesse','fontsize',20)
 hold on
 grid on
 subplot(223);%se place dans le quart bas gauche
 plot(T(j),q2p(i,j),'*')
 title('acceleration','fontsize',20)
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%Lois bang-bang
if G==4
```

```
for i=1:n
 tt1(i)=(2*(abs(qfinalrad(i)-qinitilrad(i))))/(KV(i));
 tt2(i)=2*(((abs(qfinalrad(i)-qinitilrad(i)))/(KA(i)))^0.5);
end
t1=max(tt1);
t2=max(tt2);
Tmin(i)=max(t1,t2);
Tfinal=max(t1,t2);
T2=Tfinal/2;
PAS=pas/2;
for i=1:n
for   j=1:PAS
     T(j)=(j-1)*(T2/(PAS-1));

q(i,j)=qinitial(i)+2*((T(j)/Tfinal)^2)*(qfinal(i)-qinitial(i));
    qp(i,j)=(4/(Tfinal^2))*T(j)*(qfinalrad(i)-qinitilrad(i));
    q2p(i,j)=(4/(Tfinal^2))*(qfinalrad(i)-qinitilrad(i));
 end
end
for   j=1:PAS
 hold on
 grid on
 subplot(221);%se place dans le quart haut gauche
 plot(T(j),q(i,j),'*')
 title('position','fontsize',20)
 hold on
 grid on
 subplot(222);%se place dans le quart haut droit
 plot(T(j),qp(i,j),'*')
 title('vitesse','fontsize',20)
 hold on
 grid on
 subplot(223);%se place dans le quart bas gauche
 plot(T(j),q2p(i,j),'*')
 title('acceleration','fontsize',20)
 end
 for   j=1:PAS
         T(j)=T2+(j-1)*((Tfinal-T2)/(PAS-1));

   q(i,j)=qinitial(i)+(-1+4*(T(j)/Tfinal)-2*((T(j)/Tfinal)^2))*(qfinal(i)-
qinitial(i));
   qp(i,j)=((4/Tfinal)-(4/(Tfinal)^2)*T(j))*(qfinalrad(i)-qinitilrad(i));
   q2p(i,j)=(-4/((Tfinal)^2))*(qfinalrad(i)-qinitilrad(i));
 end
  end

 for   j=1:PAS
 hold on
 grid on
 subplot(221);%se place dans le quart haut gauche
 plot(T(j),q(i,j),'*')
 title('position','fontsize',20)
  hold on
 grid on
 subplot(222);%se place dans le quart haut droit
 plot(T(j),qp(i,j),'*')
 title('vitesse','fontsize',20)
  hold on
 grid on
 subplot(223);%se place dans le quart bas gauche
```

```
 plot(T(j),q2p(i,j),'*')
 title('acceleration','fontsize',20)
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%loi trapéze
if G==5

for i=1:n
Drad(i)=(qfinalrad(i)-qinitilrad(i));
M1(i)=abs(Drad(i))/KV(i);
M2(i)=abs(Drad(i))/KA(i);
D(i)=(qfinal(i)-qinitial(i));
end

MRV=max(M1)
MRA=max(M2);
test=(MRA^0.5)
if (test>=MRV)
     Tfinal=2*test
     Ta=Tfinal/2;
 else
     Tfinal=((MRV^2)+MRA)/MRV
     Ta=MRA/MRV;
 end
 pas=Tfinal/p
for j=1:pas
          T(j)=(i-1)*(Tfinal/(pas);
 if   (T(j)>= Tinitial)&(T(j)<=Ta);
    for i=1:n
     q(i,j)=qinitial(i)+D(i)*((T(j)^2)/(2*Ta*(Tfinal-Ta)));
     qp(i,j)=(Drad(i)*T(j))/(Ta*(Tfinal-Ta));
     q2p(i,j)= Drad(i)/(Ta*(Tfinal-Ta));
    end
 hold on
 grid on
 subplot(221);%se place dans le quart haut gauche
 title('position','fontsize',20)
 plot(T(j),q(1,j),'r*')
 plot(T(j),q(2,j),'y*')
 plot(T(j),q(3,j),'b*')
 plot(T(j),q(4,j),'g*')
 plot(T(j),q(5,j),'k*')
 hold on
 grid on
 subplot(222);%se place dans le quart haut droit
 title('vitesse','fontsize',20)
 plot(T(j),qp(1,j),'r*')
 plot(T(j),qp(2,j),'y*')
 plot(T(j),qp(3,j),'b*')
 plot(T(j),qp(4,j),'g*')
 plot(T(j),qp(5,j),'k*')
 hold on
 grid on
 subplot(223);%se place dans le quart bas gauche
 title('acceleration','fontsize',20)
 plot(T(j),q2p(1,j),'r*')
 plot(T(j),q2p(2,j),'y*')
 plot(T(j),q2p(3,j),'b*')
 plot(T(j),q2p(4,j),'g*')
 plot(T(j),q2p(5,j),'k*')
 end
```

```
        TPRIM=Tfinal-Ta;
if   (T(i)>= Ta)&(T(i)<=TPRIM);

   for i=1:n
    q(i,j)=qinitial(i)+D(i)*((2*T(j)-Ta)/(2*(Tfinal-Ta)));
    qp(i,j)=Drad(i)/(Tfinal-Ta);
    q2p(i,j)=0;
   end
hold on
grid on
subplot(221);%se place dans le quart haut gauche
plot(T(j),q(1,j),'r*')
plot(T(j),q(2,j),'y*')
plot(T(j),q(3,j),'b*')
plot(T(j),q(4,j),'g*')
plot(T(j),q(5,j),'k*')
title('position','fontsize',20)
hold on
grid on
subplot(222);%se place dans le quart haut droit
title('vitesse','fontsize',20)
plot(T(j),qp(1,j),'r*')
plot(T(j),qp(2,j),'y*')
plot(T(j),qp(3,j),'b*')
plot(T(j),qp(4,j),'g*')
plot(T(j),qp(5,j),'k*')
end
 if    (T(j)>= TPRIM)&(T(j)<=Tfinal);
   for i=1:n
    q(i,j)= qinitial(i)+D(i)*(1-(((Tfinal-T(j))^2)/(2*Ta*(Tfinal-Ta))));
    qp(i,j)=-Drad(i)*((-2*Tfinal+2*T(j))/(2*Ta*(Tfinal-Ta)));
    q2p(i,j)=-Drad(i)/(Ta*(Tfinal-Ta));
 end
hold on
grid on
subplot(221);%se place dans le quart haut gauche
title('position','fontsize',20)
plot(T(j),q(1,j),'r*')
plot(T(j),q(2,j),'y*')
plot(T(j),q(3,j),'b*')
plot(T(j),q(4,j),'g*')
plot(T(j),q(5,j),'k*')
hold on
grid on
subplot(222);%se place dans le quart haut droit
title('vitesse','fontsize',20)
plot(T(j),qp(1,j),'r*')
plot(T(j),qp(2,j),'y*')
plot(T(j),qp(3,j),'b*')
plot(T(j),qp(4,j),'g*')
plot(T(j),qp(5,j),'k*')
hold on
grid on
subplot(223);%se place dans le quart bas gauche
title('acceleration','fontsize',20)
 plot(T(j),q2p(1,j),'r*')
plot(T(j),q2p(2,j),'y*')
plot(T(j),q2p(3,j),'b*')
plot(T(j),q2p(4,j),'g*')
plot(T(j),q2p(5,j),'k*')
end
```

```
 end
end
```

Programme Génération du mouvement dans l'espace opérationnel :

```
clc
clear all
clear global
G= menu('generation de mvt','interpolation linéaire','polynome de degre
trois','polynome de degre cinq','lois bang-bang','loi trapéze')

n=100;
fprintf('données une valeur pinitial ')
disp('')
pxinitial=input(['pxinitial='])
pyinitial=input(['pyinitial='])
pzinitial=input(['pzinitial='])
fprintf('données une valeur pfinal ')
disp('')
pxfinal=input(['pxfinal='])
pyfinal=input(['pyfinal='])
pzfinal=input(['pzfinal='])

fprintf('données une valeur Tinitial ')
disp('')
Tinitial=input(['Tinial ='])

fprintf('données une valeur vitesse max KV ')
disp('')
KV=input(['KV'])
fprintf('données une valeur acceleration max KA ')
disp('')
KA=input(['KA'])

D(1)=pxfinal-pxinitial;
D(2)=pyfinal-pyinitial;
D(3)=pzfinal-pzinitial;

%%%%%%%%%%%%%%%%%%%%%%%%%%%% interpolation linéaire %%%%%%%%%%%%%%%%%%%
if G==1

for i=1:3
 TFIN(i)=(abs(D(i)))/KV;
end
Tfinal=max(TFIN)

 for i=1:n
          T(i)=(i-1)*(Tfinal/(n-1));
px(:,i)= pxinitial+ (T(i)/Tfinal)*(pxfinal-pxinitial);
py(:,i)= pyinitial+ (T(i)/Tfinal)*(pyfinal-pyinitial);
pz(:,i)= pzinitial+ (T(i)/Tfinal)*(pzfinal-pzinitial);

pxp(:,i)=D(i)/Tfinal;
pyp(:,i)=D(i)/Tfinal;
pzp(:,i)=D(i)/Tfinal;

hold on
grid on
subplot(221);%se place dans le quart haut gauche
```

```
plot(T,px,'*')
title('position selon X','fontsize',20)
hold on
grid on
subplot(222);%se place dans le quart haut droit
plot(T,pxp,'*')
title('vitesse ','fontsize',20)
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%% polynome de degre trois %%%%%%%%%%%%%%%%%%%%
if G==2

for i=1:3
tt1(i)=3*abs(D(i))/(2*KV);
tt2(i)=(6*abs(D(i))/KA)^0.5;
end
t1=max(tt1);
t2=max(tt2);
Tfinal=max(t1,t2)
for i=1:n
          T(i)=(i-1)*(Tfinal/(n-1));

px(:,i)=pxinitial+3*(pxfinal-pxinitial)*(T(i)^2)/(Tfinal^2)-2*(pxfinal-
pxinitial)*(T(i)^3)/(Tfinal^3);
py(:,i)=pyinitial+3*(pyfinal-pyinitial)*(T(i)^2)/(Tfinal^2)-2*(pyfinal-
pyinitial)*(T(i)^2)/(Tfinal^2);
pz(:,i)=pzinitial+3*(pzfinal-pzinitial)*(T(i)^2)/(Tfinal^2)-2*(pzfinal-
pzinitial)*(T(i)^3)/(Tfinal^3);

pxp(:,i)=(6*(pxfinal-pxinitial)/(Tfinal^2))*T(i)-(6*(pxfinal-
pxinitial)/(Tfinal^3))*T(i)^2;
pyp(:,i)=(6*(pyfinal-pyinitial)/(Tfinal^2))*T(i)-(6*(pyfinal-
pyinitial)/(Tfinal^3))*T(i)^2;
pzp(:,i)=(6*(pzfinal-pzinitial)/(Tfinal^2))*T(i)-(6*(pzfinal-
pzinitial)/(Tfinal^3))*T(i)^2;


px2p(:,i)=6*(pxfinal-pxinitial)/(Tfinal^2)-(12*(pxfinal-
pxinitial)/(Tfinal^3))*T(i);
py2p(:,i)=6*(pyfinal-pyinitial)/(Tfinal^2)-(12*(pyfinal-
pyinitial)/(Tfinal^3))*T(i);
pz2p(:,i)=6*(pzfinal-pzinitial)/(Tfinal^2)-(12*(pzfinal-
pzinitial)/(Tfinal^3))*T(i);

hold on
grid on
subplot(221);%se place dans le quart haut gauche
plot(T,px,'*')
title('position selon X','fontsize',20)
hold on
grid on
subplot(222);%se place dans le quart haut droit
plot(T,pxp,'*')
title('vitesse selon X','fontsize',20)
hold on
grid on
subplot(223);%se place dans le quart bas gauche
plot(T,px2p,'*')
title('acceleration selon X','fontsize',20)
end
```

```
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%% polynome de degre cinq %%%%%%%%%%%%%%%%%%%%%%%%
if G==3

for i=1:3
 tt1(i)=15*abs(D(i))/(8*KV);
 tt2(i)=((10*abs(D(i)))/((3^0.5)*KA))^0.5;
end
t1=max(tt1);
t2=max(tt2);
 Tfinal=max(t1,t2)
for i=1:n
      T(i)=(i-1)*(Tfinal/(n-1));

   px(:,i)=pxinitial+(10*((T(i)/Tfinal)^3)-
15*((T(i)/Tfinal)^4)+6*((T(i)/Tfinal)^5))*(pxfinal-pxinitial);
   py(:,i)=pyinitial+(10*((T(i)/Tfinal)^3)-
15*((T(i)/Tfinal)^4)+6*((T(i)/Tfinal)^5))*(pyfinal-pyinitial);
   pz(:,i)=pzinitial+(10*((T(i)/Tfinal)^3)-
15*((T(i)/Tfinal)^4)+6*((T(i)/Tfinal)^5))*(pzfinal-pzinitial);

   pxp(:,i)=((30/(Tfinal^3))*(T(i)^2)-
(60/(Tfinal^4))*(T(i)^3)+(30/(Tfinal^5))*(T(i)^4))*(pxfinal-pxinitial);
   pyp(:,i)=((30/(Tfinal^3))*(T(i)^2)-
(60/(Tfinal^4))*(T(i)^3)+(30/(Tfinal^5))*(T(i)^4))*(pyfinal-pyinitial);
   pzp(:,i)=((30/(Tfinal^3))*(T(i)^2)-
(60/(Tfinal^4))*(T(i)^3)+(30/(Tfinal^5))*(T(i)^4))*(pzfinal-pzinitial);

   px2p(:,i)=((60/(Tfinal^3))*T(i)-
(180/(Tfinal^4))*(T(i)^2)+(120/(Tfinal^5))*(T(i)^3))*(pxfinal-pxinitial) ;
   py2p(:,i)=((60/(Tfinal^3))*T(i)-
(180/(Tfinal^4))*(T(i)^2)+(120/(Tfinal^5))*(T(i)^3))*(pyfinal-pyinitial) ;
   pz2p(:,i)=((60/(Tfinal^3))*T(i)-
(180/(Tfinal^4))*(T(i)^2)+(120/(Tfinal^5))*(T(i)^3))*(pzfinal-pzinitial) ;
hold on
grid on
subplot(221);%se place dans le quart haut gauche
plot(T,px,'*')
title('position selon X','fontsize',20)
hold on
grid on
subplot(222);%se place dans le quart haut droit
plot(T,pxp,'*')
title('vitesse selon X','fontsize',20)
hold on
grid on
subplot(223);%se place dans le quart bas gauche
plot(T,px2p,'*')
title('acceleration selon X','fontsize',20)
end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% lois bang-bang %%%%%%%%%%%%%%%%%%%%%%%%%%
if G==4

for i=1:3
tt1(i)=(2*abs(D(i)))/KV;
tt2(i)=2*((abs(D(i))/KA)^0.5);
end
t1=max(tt1);
```

```
t2=max(tt2);
Tfinal=max(t1,t2)
for i=1:n
     T(i)=(i-1)*(Tfinal/(n-1));
if   (T(i)>= Tinitial)&(T(i)<=Tfinal/2);

    px(:,i)=pxinitial+2*((T(i)/Tfinal)^2)*(pxfinal-pxinitial);
    py(:,i)=pyinitial+2*((T(i)/Tfinal)^2)*(pyfinal-pyinitial);
    pz(:,i)=pzinitial+2*((T(i)/Tfinal)^2)*(pzfinal-pzinitial);

    pxp(:,i)=(4/(Tfinal^2))*T(i)*(pxfinal-pxinitial);
    pyp(:,i)=(4/(Tfinal^2))*T(i)*(pyfinal-pyinitial);
    pzp(:,i)=(4/(Tfinal^2))*T(i)*(pzfinal-pzinitial);

    px2p(:,i)=(4/(Tfinal^2))*(pxfinal-pxinitial);
    py2p(:,i)=(4/(Tfinal^2))*(pyfinal-pyinitial);
    pz2p(:,i)=(4/(Tfinal^2))*(pzfinal-pzinitial);
end
if   (T(i)>=Tfinal/2)&(T(i)<=Tfinal);

   px(:,i)=pxinitial+(-1+4*(T(i)/Tfinal)-2*((T(i)/Tfinal)^2))*(pxfinal-
pxinitial);
   py(:,i)=pyinitial+(-1+4*(T(i)/Tfinal)-2*((T(i)/Tfinal)^2))*(pyfinal-
pyinitial);
   pz(:,i)=pzinitial+(-1+4*(T(i)/Tfinal)-2*((T(i)/Tfinal)^2))*(pzfinal-
pzinitial);

   pxp(:,i)=((4/Tfinal)-(4/(Tfinal)^2)*T(i))*(pxfinal-pxinitial);
   pyp(:,i)=((4/Tfinal)-(4/(Tfinal)^2)*T(i))*(pyfinal-pyinitial);
   pzp(:,i)=((4/Tfinal)-(4/(Tfinal)^2)*T(i))*(pzfinal-pzinitial);

   px2p(:,i)=(-4/(Tfinal^2))*(pxfinal-pxinitial);
   py2p(:,i)=(-4/(Tfinal^2))*(pyfinal-pyinitial);
   pz2p(:,i)=(-4/(Tfinal^2))*(pzfinal-pzinitial);
end

hold on
grid on
subplot(221);%se place dans le quart haut gauche
plot(T,px,'*')
title('position selon X','fontsize',20)
hold on
grid on
subplot(222);%se place dans le quart haut droit
plot(T,pxp,'*')
title('vitesse selon X','fontsize',20)
hold on
grid on
subplot(223);%se place dans le quart bas gauche
plot(T,px2p,'*')
title('acceleration selon X','fontsize',20)
end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% loi trapéze %%%%%%%%%%%%%%%%%%%%%%%%%
 if G==5

for i=1:3
M1(i)=abs(D(i))/KV;
M2(i)=abs(D(i))/KA;
```

```
end
 MRV=max(M1);
 MRA=max(M2);
 test=(MRA^0.5);

if (test>=MRV)
     Tfinal=2*test
     Ta=Tfinal/2
else
     Tfinal=((MRV^2)+MRA)/MRV
     Ta=MRA/MRV
end
 for i=1:n
      T(i)=(i-1)*(Tfinal/(n-1));

if   (T(i)>= Tinitial)&(T(i)<=Ta);

     px(:,i)=pxinitial+D(1)*((T(i)^2)/(2*Ta*(Tfinal-Ta)));
     py(:,i)=pyinitial+D(2)*((T(i)^2)/(2*Ta*(Tfinal-Ta)));
     pz(:,i)=pzinitial+D(3)*((T(i)^2)/(2*Ta*(Tfinal-Ta)));

     pxp(:,i)=(D(1)*T(i))/(Ta*(Tfinal-Ta));
     pyp(:,i)=(D(2)*T(i))/(Ta*(Tfinal-Ta));
     pzp(:,i)=(D(3)*T(i))/(Ta*(Tfinal-Ta));

     px2p(:,i)= D(1)/(Ta*(Tfinal-Ta));
     py2p(:,i)= D(2)/(Ta*(Tfinal-Ta));
     pz2p(:,i)= D(3)/(Ta*(Tfinal-Ta));

 hold on
 grid on
 subplot(221);%se place dans le quart haut gauche
 plot(T,px,'*')
 title('position selon X','fontsize',20)
 hold on
 grid on
 subplot(222);%se place dans le quart haut droit
 plot(T,pxp,'*')
 title('vitesse selon X','fontsize',20)
 hold on
 grid on
 subplot(223);%se place dans le quart bas gauche
 plot(T,px2p,'*')
 title('acceleration selon X','fontsize',20)
end
         TPRIM=Tfinal-Ta;

if   (T(i)>= Ta)&(T(i)<=TPRIM);

     px(:,i)=pxinitial+D(1)*((2*T(i)-Ta)/(2*(Tfinal-Ta)));
     py(:,i)=pyinitial+D(2)*((2*T(i)-Ta)/(2*(Tfinal-Ta)));
     pz(:,i)=pzinitial+D(3)*((2*T(i)-Ta)/(2*(Tfinal-Ta)));

     pxp(:,i)=D(1)/(Tfinal-Ta);
     pyp(:,i)=D(2)/(Tfinal-Ta);
     pzp(:,i)=D(3)/(Tfinal-Ta);

hold on
grid on
subplot(221);%se place dans le quart haut gauche
```

```
plot(T,px,'*')
title('position selon X','fontsize',20)
hold on
grid on
subplot(222);%se place dans le quart haut droit
plot(T,pxp,'*')
title('vitesse selon X','fontsize',20)
end

 if (T(i)>= TPRIM)&(T(i)<=Tfinal);

     px(:,i)= pxinitial+D(1)*(1-(((Tfinal-T(i))^2)/(2*Ta*(Tfinal-Ta))));
     py(:,i)= pyinitial+D(2)*(1-(((Tfinal-T(i))^2)/(2*Ta*(Tfinal-Ta))));
     pz(:,i)= pzinitial+D(3)*(1-(((Tfinal-T(i))^2)/(2*Ta*(Tfinal-Ta))));

     pxp(:,i)=-D(1)*((-2*Tfinal+2*T(i))/(2*Ta*(Tfinal-Ta)));
     pyp(:,i)=-D(2)*((-2*Tfinal+2*T(i))/(2*Ta*(Tfinal-Ta)));
     pzp(:,i)=-D(3)*((-2*Tfinal+2*T(i))/(2*Ta*(Tfinal-Ta)));

     px2p(:,i)=-D(1)/(Ta*(Tfinal-Ta));
     py2p(:,i)=-D(2)/(Ta*(Tfinal-Ta));
     pz2p(:,i)=-D(3)/(Ta*(Tfinal-Ta));

hold on
grid on
subplot(221);%se place dans le quart haut gauche
plot(T,px,'*')
title('position selon X','fontsize',20)

hold on
grid on
subplot(222);%se place dans le quart haut droit
plot(T,pxp,'*')
title('vitesse selon X','fontsize',20)
hold on
grid on
subplot(223);%se place dans le quart bas gauche
plot(T,px2p,'*')
title('acceleration selon X','fontsize',20)
end
end
end
```

Programme Modèle Géométrique Inverse dans l'application :

```
clc
clear all
poinduplan=[Px;   Py;   Pz]
poinparalduplan=[Px;   Py;  Pzeng]

coordonoperationel=poinduplan';
save coordonoperationel.txt coordonoperationel -ascii

coordonoperationeldeng=poinparalduplan';
save coordonoperationeldeng.txt coordonoperationeldeng -ascii

n=[0.0965609   0.241402   0.965091];
r1=0.40982;
d3=1.000;
r4=1.5139;
d=[0  0  1.000  0  0];
r=[0.40982  0  0  1.5139  0];
alpha=[0  90 0  90  -90];
%=========================================================================
%calcule le positionnement et orientation du robot
%=========================================================================
for i=1:9
    q1(i)=atan(Py(i)/Px(i));
    q11(i)=(q1(i)*180)/pi;

    qq3(i)=asin((((Px(i)^2)+(Py(i)^2))+((Pz(i)-r1)^2)-
(((r4)^2)+((d3)^2)))/(2*r4*d3));
    q33(i)=(qq3(i)*180)/pi;

    q2(i)=(atan((r4*cos(qq3(i)))/(r4*sin(qq3(i))+d3)))+(asin((Pz(i)-
r1)/((((r4*sin(qq3(i))+d3)^2)+((r4*cos(qq3(i)))^2))^0.5)));
    q22(i)=(q2(i)*180)/pi;

   q5(i)=-acos((n(3)*cos(q2(i)+qq3(i)))-
(n(1)*cos(q1(i))+n(2)*sin(q1(i)))*(sin(q2(i)+qq3(i))));
    q55(i)=(q5(i)*180)/pi;

    qq4(i)=asin((n(2)*cos(q1(i))-n(1)*sin(q1(i)))/sin(q5(i)));
    q44(i)=(qq4(i)*180)/pi;
 end
 for i=1:9
    Q(:,i)=[q11(i)   q22(i)   q33(i)   q44(i)   q55(i)]
    Qrad(:,i)=[q1(i)  q2(i)   qq3(i)   qq4(i)   q5(i)];
 end
 %=========================================================================
 % Modèle geométrique direct
 %=========================================================================
  for j=1:9
             for i=1:5

            T(1,1,i,j)=cosd(Q(i,j));
            T(1,2,i,j)=-sind(Q(i,j));
            T(1,3,i,j)=0 ;
            T(1,4,i,j)=d(i);
            T(2,1,i,j)=sind(Q(i,j))*cosd(alpha(i));
            T(2,2,i,j)=cosd(Q(i,j))*cosd(alpha(i));
            T(2,3,i,j)=-sind(alpha(i));
```

```
            T(2,4,i,j)=-r(i)*sind(alpha(i));
            T(3,1,i,j)=sind(Q(i,j))*sind(alpha(i));
            T(3,2,i,j)=cosd(Q(i,j))*sind(alpha(i));
            T(3,3,i,j)=cosd(alpha(i));
            T(3,4,i,j)=r(i)*cosd(alpha(i));
            T(4,1,i,j)=0;
            T(4,2,i,j)=0;
            T(4,3,i,j)=0;
            T(4,4,i,j)=1;
             end
 end
for j=1:9
TT(:,:,1,j)=eye(4);
for i=2:6
TT(:,:,i,j)=TT(:,:,i-1,j)*T(:,:,i-1,j);
end
end
coordonarticulaire=Q;
save coordonarticulaire.txt coordonarticulaire -ascii
coordonarticulairerad=Qrad;
save coordonarticulairerad.txt coordonarticulairerad -ascii
%=========================================================================
%les points d'engagement
%=========================================================================
for i=1:9
    q1(i)=atan(Py(i)/Px(i));
    qq11(i)=(q1(i)*180)/pi;

    qq3(i)=asin((((Px(i)^2)+(Py(i)^2))+((Pzeng(i)-r1)^2)-
(((r4)^2)+((d3)^2)))/(2*r4*d3));
    qq33(i)=(qq3(i)*180)/pi;

    q2(i)=(atan((r4*cos(qq3(i)))/(r4*sin(qq3(i))+d3)))+(asin((Pzeng(i)-
r1)/((((r4*sin(qq3(i))+d3)^2)+((r4*cos(qq3(i)))^2))^0.5)));
    qq22(i)=(q2(i)*180)/pi;

   q5(i)=-acos((n(3)*cos(q2(i)+qq3(i)))-
(n(1)*cos(q1(i))+n(2)*sin(q1(i)))*(sin(q2(i)+qq3(i))));
    qq55(i)=(q5(i)*180)/pi;

    qq4(i)=asin((n(2)*cos(q1(i))-n(1)*sin(q1(i)))/sin(q5(i)));
    qq44(i)=(qq4(i)*180)/pi;
end
 for i=1:9
      Qeng(:,i)=[qq11(i)   qq22(i)   qq33(i)   qq44(i)   qq55(i)]
      Qradeng(:,i)=[q1(i)   q2(i)     qq3(i)    qq4(i)    q5(i)]
 end
 %=========================================================================
 % Modèle geométrique direct
 %=========================================================================
  for j=1:9
             for i=1:5

            h(1,1,i,j)=cosd(Qeng(i,j));
            h(1,2,i,j)=-sind(Qeng(i,j));
            h(1,3,i,j)=0 ;
            h(1,4,i,j)=d(i);
            h(2,1,i,j)=sind(Qeng(i,j))*cosd(alpha(i));
            h(2,2,i,j)=cosd(Qeng(i,j))*cosd(alpha(i));
            h(2,3,i,j)=-sind(alpha(i));
```

```
            h(2,4,i,j)=-r(i)*sind(alpha(i));
            h(3,1,i,j)=sind(Qeng(i,j))*sind(alpha(i));
            h(3,2,i,j)=cosd(Qeng(i,j))*sind(alpha(i));
            h(3,3,i,j)=cosd(alpha(i));
            h(3,4,i,j)=r(i)*cosd(alpha(i));
            h(4,1,i,j)=0;
            h(4,2,i,j)=0;
            h(4,3,i,j)=0;
            h(4,4,i,j)=1;
             end
end
for j=1:9
hh(:,:,1,j)=eye(4);
for i=2:6
hh(:,:,i,j)=hh(:,:,i-1,j)*h(:,:,i-1,j);
end
end
 coordonarticulairedeng=Qeng;
save coordonarticulairedeng.txt coordonarticulairedeng -ascii
  coordonarticulairedengrad=Qradeng;
save coordonarticulairedengrad.txt coordonarticulairedengrad -ascii
```

Bibliographie

[Khal99] : W. Khalil et E. Dombr.
« *Modélisation identification et commande des robots* » 2ème édition, Hermès Science Publique, Paris1999, France.

[Rena84] : B. Gorla et M. Renaud
« *Modèles des robots manipulateurs : application à leur commande* » , Cepadues-édition imprimerie du sud, Toulouse1984, France.

[Domb01] : Etienne Dombre
« *Analyse et modélisation des robots manipulateurs* », édition Lavoisier 2001.

[Lall94] : J-P Lallemand et S. Zeghloul
« *Robotique aspects fondamentaux, Modélisation mécanique, CAO robotique, Commande* », édition Masson. Paris, Milan, Barcelone 1993-1994, France.

[Chev04] : A. Chevalier
« *Guide du dessinateur industriel* », Hachette technique, France 2004.

[Chal10] : Chales Bop
« *Robotique, Traité de robotique tome1, les architectures, conception et modélisation* », édition Ellipses 2010, France.

[Hadd11]: A. Yousnadj et M. Haddad
«*Modélisation et commande des robots pour 1er année magister* », Ecole Militaire Polytechnique d'Alger 2010-2011.

[Boim01] : Jean-louis Boimand
« *Robotique* » cour à l'ISTIA, université Angers, France 2001.

[Tech03] : Mekhilef M., Yannou B
Conception Intégrée Assistée par Ordinateur. Technique de l'ingénieur, BM5006, 1998, France.

[Bors11] : Mme S. Borsali
« *Modélisation des robots, Master automatique S2* » soutient de cour à l'université Abou-Bekr Belkaid, faculté des sciences de l'ingénieur département d'automatique Tlemcen 2011.

[Bayl08] : Bernard Bayle
« *Robotique* » cour de master ingénierie et technologie à l'université Louis Pasteur de Strasbourg 2007-2008, France.

[Boua09] : A. Bouafia
« *Méthodes de description des orientations* » soutient de coure pour les ingénieurs à l'université des sciences et de technologie Houari-Boumediene 2009.

[Ghao09] : Ghaoui Driss
« *Modélisation et simulation dynamique des systèmes multi corps. Application à la simulation du comportement dynamique d'un hélicoptère* », Ecole Militaire Polytechnique d'Alger 2009.

[Ait-A93] : Mourad Ait-Ahmed
« *Contribution à la modélisation géométrique et dynamique des robots parallèles* » Thèse du doctorat à laboratoire d'automatique et d'analyse des systèmes du CNRS, l'université Paul Sabatier. Toulouse1993, France.
[Nait06] : Khiar Nait Chabane
« *Exploitation de la redondance pour la commande coordonnée d'un manipulateur mobile d'assistance aux personnes handicapées* » Thèse du doctorat à l'université d'EVRY- Val d'Essonne 2006, France.
[Ref1] : Site Web ; Zoneindustrie.com

الملخص :

يتضمن عملنا دراسة عامة لروبوتات صناعية ثابتة بذراع ذات هيكل مفتوح بستة 6 درجات من الحرية, يسمح بحساب آلي لمختلف الدراسات الهندسية والحركية و الديناميكية اللازمة في دراسة ما قبل مرحلة تصنيع الروبوت. قمنا بالوصف الهندسي المباشر باستعمال قواعد "دينا فيت وهاتن برغ المعدل". اما بالنسبة للنموذج العكسي, فاتبعنا الحل الاوتوماتيكي. استخدمنا مفهوم "جاكوبيا" في استخراج المعادلات الحركية المباشرة ذات الدرجة الاولى و الثانية, و مفهوم بسدو-عكسي في الدراسة الحركية العكسية بمنهج "قريفال". الدراسة الديناميكية تمت بطريقة "نيوتن-اولير", و بعدها عرضنا المعادلات الممثلة للحركة التي تعتبر مرحلة مهمة في تحديد مسار الروبوت. البرامج المقترحة لكل هذه الدراسات المباشرة و العكسية تم توظيفها ضمن دراسة روبوت ذو 5 درجات حرية في مهمة قياس3ابعاد, كما اننا جسدنا تحركات الروبوت أثناء قيامه بالمهمة بالاستخدام العالم الافتراضي عن طريق البرنامج "سوليد-ووركس".

الكلمات المفتاح : علم الالية(الروبوتيك) , روبوت صناعية , مناور ذراع الروبوت , نموذجة الروبوت , توليد المسار.

Résumé :

Notre travail consiste à étudier une configuration générale d'un robot manipulateur à structure ouverte simple à 6ddl. Il permet un calcul automatisé des différentes modélisations géométrique, cinématique et dynamique nécessaires à un avant-projet de réalisation du robot. On a adopté pour la modélisation géométrique directe la convention de Denavit-Hertenberg Modifiée. Pour le problème inverse, est utilisée la résolution numérique. On a fait appel à la notion du jacobien pour développer le modèle cinématique direct du 1^{er} et $2^{ème}$ ordre, et au pseudo-inverse par l'algorithme de Greville pour l'inversion de ces modèles. Le modèle dynamique a été traité à l'aide du formalisme de Newton-Euler. L'étude de trajectoire qui est une étape importante pour générer le mouvement du robot est également présentée. Les programmes établis sont exploités pour une application de mesure 3D à l'aide d'un robot a 5ddl avec simulation sous SolidWorks.

Mots clés : Robotique, Robot industriel, Bras manipulateur, Modélisation des robots, Génération du mouvement.

Summary:

Our work consists in studying a general configuration of a robot manipulator to simple structure open to 6ddl. It allows an automated calculation of various modeling geometrical, kinematic and dynamic necessary to a preliminary draft of realization of the robot. One adopted for direct geometrical modeling the convention of Denavit-Hertenberg Modified. For the opposite problem, is used the numerical resolution. One called upon the notion of the jacobien to develop the direct kinematic model of 1st and 2nd order and with pseudo-opposite by the algorithm of Greville for the inversion of these models. The dynamic model was treated using the formalism of Newton-Euler. The study of trajectory which is a big step to generate the movement of the robot is also presented. The established programs are exploited for an application of measure 3D using a robot has 5ddl with simulation under SolidWorks.

Key words: Robotics, Industrial robot, Arm manipulator, Modeling of the robots, Generation of the movement.