



Université de Bouira
Faculté des Sciences Appliquées
Département de Génie Electrique

Domaine : Sciences et Technologies

Filière : *Electronique*

Spécialité : *ESE*

Autorisation de déposer un mémoire de Master
-après soutenance et levé des réserves-
(Une autorisation/mémoire)

Je soussigné, l'enseignant : *ISSAOUNI Salim*

Encadreur de l'étudiant (nom et prénom) : 1- *HAMBLI Billal*
2- *REKHILA Amine*

Ayant soutenu le mémoire de master intitulé : *Modélisation, Commande et réalisation d'un Robot SCARA à 4DDL*

Inscrit dans la faculté des sciences et des sciences appliquées dans la spécialité citée en haut, promotion : 2024/2025.

Et après voir et consulter le mémoire dans sa forme finale (après soutenance et levé des réserves), nous autorisons les étudiants à l'imprimer et le déposer aux bibliothèques centrale et de la faculté.

Bouira le : *29/06/2025*

Signature du rapporteur

ISSAOUNI S

Signature du président de jury

ASRAO S. Z



ID: y3ozsb-200045

Rapport d'analyse de la similarité textuelle

- Nom du document: **Memoire Finale Robot Scara 4ddl.pdf**
- Soumis par: **Aggoun Ghania** **Enseignant**
- Date de soumission: **2025-06-15**

**Taux global de similarité**

- 19.1% Similarité Forte
- 0.0% Exclu manuellement

**Nombre de sources**

175 sources internet
14 sources Thèses-Algérie
0 sources dépôt privé

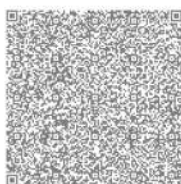
**Passages surlignés**

15602 mots
122968 caractères
3.8% de citations

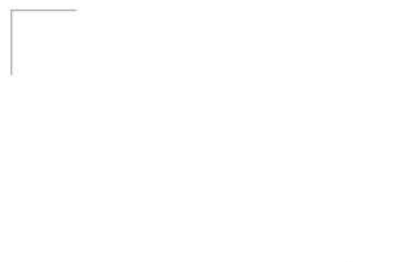
Ce document est un rapport d'analyse et de détection de similarité textuelle qui peut fournir des preuves pour l'établissement d'un rapport de plagiat. Il revient à l'examineur, l'encadrant ou bien au comité déontologique de l'université ou de l'école d'émettre un avis quant au statut de plagiat du document analysé.

© Consultez l'arrêté N° 1082 du 27 Décembre 2020 fixant les règles relatives à la prévention et la lutte contre le plagiat pour en savoir plus concernant ce qui est considéré comme étant un acte de plagiat, les procédures ainsi que les sanctions.

Signature d'intégrité



Cachet et Signature



الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur
et de la Recherche Scientifique
Université Akli Mohand Oulhadj - Bouira -
Tasdawit Akli Muḥend Ulḥağ - Tibirett -



وزارة التعليم العالي والبحث العلمي
جامعة أكلي محمد أولحاج
- البويرة -

Mémoire de Master

Présenté au

Faculté des Sciences Appliquées

Département : Génie Électrique

Domaine : Sciences et Technologies

Filière : Electronique

Spécialité : Electronique des systèmes embarqués

Réalisé par :

REKHILA Amine

Et

HAMBLI Billal

Thème

Modélisation, Commande et Réalisation du Robot SCARA à 4DDL

Soutenu le : 24/06/2025

Devant la commission composée de :

Mr : Z. Asradj	M C. B	Univ. Bouira	Président
Mr : S. ISSAOUNI	M.A. A	Univ. Bouira	Rapporteur
Mr : S. Haroun	M.C. B	Univ. Bouira	Examineur

Année Universitaire : 2024-2025

Remerciements

Je tiens à exprimer ma profonde gratitude à Monsieur ISSAOUM Salim, enseignant-chercheur au département de Génie Électrique de l'Université de Bouira, pour sa disponibilité, ses conseils avisés, son encadrement rigoureux et son soutien tout au long de la réalisation de ce mémoire.

Je remercie également l'ensemble des membres du jury pour avoir accepté d'évaluer ce travail, ainsi que les enseignants du département pour la qualité de l'enseignement dispensé au cours de cette formation.

J'exprime aussi ma reconnaissance au département de Génie Mécanique, ainsi qu'à Monsieur le Doyen HOUNASSINE Hamza de la faculté pour m'avoir accordé l'autorisation d'utiliser le laboratoire de fabrication, ce qui a permis l'impression des pièces du robot. Mes sincères remerciements vont également à l'ingénieure du laboratoire, pour sa disponibilité et son accompagnement constant jusqu'à la finalisation de cette étape importante.

Mes remerciements s'adressent également à mes camarades et amis pour leur esprit de collaboration, leurs encouragements et leur soutien moral.

Enfin, je remercie ma famille, et tout particulièrement mes parents, pour leur amour inconditionnel, leur patience, leur confiance et leurs sacrifices constants tout au long de mon parcours universitaire.

Dédicace

À mes chers parents,

Vous êtes la source de mon inspiration et de ma force. Ce travail est avant tout le fruit de vos sacrifices, de votre patience et de vos prières. Merci pour tout.

À mes frères et sœurs,

Merci pour votre soutien constant, vos encouragements et votre présence réconfortante à chaque étape de mon parcours.

À mes enseignants,

Pour leur engagement, leur passion pour la transmission du savoir, et leur rôle fondamental dans ma formation.

À mes amis et camarades,

Pour les échanges enrichissants, l'entraide, les rires et les souvenirs partagés durant ces années.

Je vous dédie ce mémoire avec toute ma reconnaissance, mon affection et mon respect.

Rekhila Amine

Dédicace

Je dédie ce modeste travail :

À mon père bien-aimé, que Dieu ait son âme en paix (رحمك الله يا أبي). Même absent, ta mémoire, tes conseils et ton amour continuent de me guider à chaque étape de ma vie. Ce travail est aussi le fruit de ce que tu m'as transmis.

Tu restes à jamais dans mon cœur.

À ma chère mère, pour son amour inconditionnel, ses prières, ses sacrifices et son soutien sans faille. Tu es la lumière de ma vie et ma source de force. À mes frères et sœurs,

Merci pour votre soutien constant, vos encouragements et votre présence réconfortante à chaque étape de mon parcours.

À mes enseignants,

Pour leur engagement, leur passion pour la transmission du savoir, et leur rôle fondamental dans ma formation.

À mes amis et camarades,

Pour les échanges enrichissants, l'entraide, les rires et les souvenirs partagés durant ces années.

Je vous dédie ce mémoire avec toute ma reconnaissance, mon affection et mon respect.

Hamblil Billal

Résumé

Ce mémoire porte sur la robotique industrielle, avec un intérêt particulier pour le robot *SCARA* à quatre degrés de liberté (*4DDL*). L'objectif principal est d'étudier, simuler et contrôler ce type de robot dans un environnement de simulation intégré, en vue d'une réalisation partielle matérielle. L'approche adoptée repose sur trois axes majeurs : la modélisation, la commande et la réalisation.

Dans un premier temps, une modélisation géométrique et dynamique du robot *SCARA* a été menée. Cela inclut la formulation des équations cinématiques et dynamiques, permettant de simuler avec précision le comportement du robot dans différentes configurations.

La seconde étape est consacrée à la commande du robot à l'aide d'un régulateur *PID*. Ce type de commande, largement utilisé pour sa simplicité et son efficacité, a été réglé de manière à garantir une bonne stabilité du système ainsi qu'une précision satisfaisante en position et en vitesse.

L'animation du robot, intégrée à la phase de simulation, a été réalisée selon trois méthodes : *MATLAB*, *Simscape Multibody* et un fichier *URDF*, permettant une visualisation dynamique adaptée à différents environnements de simulation.

Enfin, une réalisation matérielle simplifiée a été effectuée à l'aide d'une carte *Arduino*, permettant une validation expérimentale partielle des résultats obtenus en simulation.

Les résultats montrent que la commande *PID* assure une réponse stable et précise, malgré certaines limitations comme l'absence de capteurs de retour et la prise en compte limitée des perturbations. Ce travail constitue ainsi une base solide pour une future réalisation complète du robot *SCARA* et pour le développement de solutions robotiques plus avancées.

Mots-clés : Robotique industrielle, Robot *SCARA*, degrés de liberté, Modélisation cinématique, Modélisation dynamique, Commande *PID*, Animation robotique, *Simscape Multibody*, Fichier *URDF*, *Arduino*.

الملخص

يركز هذا البحث على الروبوتات الصناعية، مع اهتمام خاص بـ الروبوت *SCARA* ذو أربع درجات حرية (*DDL4*). الهدف الرئيسي هو دراسة هذا النوع من الروبوتات، ومحاكاته، والتحكم فيه ضمن بيئة محاكاة متكاملة، وذلك بهدف تحقيق جزئي مادي للنموذج. تعتمد المنهجية على ثلاث مراحل رئيسية: النمذجة، التحكم، والتحقيق الفعلي.

تتمثل المرحلة الأولى في النمذجة الهندسية والديناميكية للروبوت *SCARA*، من خلال صياغة المعادلات السينماتيكية والديناميكية التي تسمح بمحاكاة دقيقة لسلوك الروبوت في مختلف الوضعيات.

أما المرحلة الثانية، فتُخصص لـ التحكم في الروبوت باستخدام منظم *PID*، نظراً لبساطته وفعاليته، مع ضبط مناسب للمعاملات لضمان ثبات النظام ودقته في المواضع والسرعة.

تم إجراء تحريك (أنيميشين) الروبوت كجزء من مرحلة المحاكاة، باستخدام ثلاث طرق: *MATLAB*، و *Simscape*، و *Multibody*، وملف *URDF*، مما سمح بعرض مرئي ديناميكي في بيئات محاكاة مختلفة.

وفي المرحلة الأخيرة، تم تنفيذ تحقيق مادي مبسط باستخدام لوحة أردوينو، مما أتاح تحقيق تحقق تجريبي جزئي من نتائج المحاكاة.

أظهرت النتائج أن التحكم بـ *PID* يوفر استجابة مستقرة ودقيقة، على الرغم من بعض القيود، مثل غياب مستشعرات التغذية الراجعة، وضعف التعامل مع الاضطرابات الخارجية. يشكل هذا العمل أساساً متيناً لتحقيق كامل مستقبلي لهذا الروبوت، ولتطوير مشاريع روبوتية أكثر تقدماً.

الكلمات المفتاحية:

الروبوتات الصناعية، روبوت *SCARA*، أربع درجات حرية، النمذجة السينماتيكية، النمذجة الديناميكية، التحكم *PID*، *MATLAB/Simulink*، التحريك، *Simscape Multibody*، ملف *URDF*، التحقيق المادي، أردوينو، التحقق التجريبي.

Abstract

This thesis focuses on industrial robotics, with particular emphasis on the *SCARA* robot with four degrees of freedom (*4DOF*). The main objective is to study, simulate, and control this type of robot in an integrated simulation environment, with the aim of a partial hardware implementation. The approach is based on three main stages: modeling, control, and implementation.

The first stage involves the geometric and dynamic modeling of the *SCARA* robot. This includes the formulation of kinematic and dynamic equations, enabling precise simulation of the robot's behavior in various configurations.

The second stage is devoted to the control of the robot using a *PID* controller. This type of control was selected for its simplicity and efficiency, with proper tuning of the *PID* parameters to ensure system stability and accuracy in both position and speed.

The animation of the robot, integrated into the simulation phase, was carried out using three methods: *MATLAB*, *Simscape Multibody*, and a *URDF* file, enabling dynamic visualization in different simulation platforms.

Finally, a simplified hardware implementation was developed using an Arduino board, allowing for partial experimental validation of the simulated results.

The results demonstrate that the *PID* control provides a stable and accurate response, despite certain limitations such as the lack of feedback sensors and limited disturbance handling. This work provides a solid foundation for a future complete realization of the *SCARA* robot and for the development of more advanced robotic systems.

Keywords:

Industrial robotics, *SCARA* robot, *4DOF*, Kinematic modeling, Dynamic modeling, *PID* control, *MATLAB/Simulink*, Animation, *Simscape Multibody*, *URDF* file, Hardware implementation, *Arduino*, Experimental validation.

Table des matières

Remerciements.....	I
Résumé	II
Table des Matières	III
Liste des Figures	IV
Liste des Tableaux	V
Listes des Acronymes et Symboles.....	VI
Introduction Générale.....	1

Chapitre 1 : Modélisation du Robot SCARA

1.1	Introduction	2
1.2	Objectifs des deux règlements Historique de la robotique	2
1.2.1	Origines et premières inventions	2
1.2.2	Le XXe siècle : naissance de la robotique moderne	3
1.2.3	Le XXIe siècle : l'ère de l'intelligence artificielle	3
1.3	Définitions	4
1.3.1	Robotique	4
1.3.2	Robot.....	4
1.4	Constitution des robots.....	4
1.5	Notions base de la robotique	5
1.5.1	Articulation.....	5
1.5.2	Espace articulaire	5
1.5.3	Espace opérationnel	6
1.5.4	Degré de liberté (<i>DDL</i>)	6
1.6	Types des robots	6
1.6.1	En fonction de la structure.....	6
1.6.2	En fonction de leur mobilité	6

1.6.3	En fonction de leur application.....	7
1.6.4	En fonction de leur méthode de contrôle	7
1.6.5	En fonction de leur source d'énergie	7
1.7	Notations et outils mathématiques pour la modélisation des robots.....	8
1.7.1	Représentation d'un point	8
1.7.2	Représentation d'une direction	9
1.7.3	Représentation d'un plan.....	9
1.7.4	Transformations homogènes des repères	10
1.7.5	Transformation des vecteurs.....	11
1.7.6	Transformation des plans	11
1.7.7	Matrice de transformation de translation pure.....	11
1.7.8	Matrice de transformation de rotation autour des axes principaux.....	12
1.7.9	Propriétés des matrices de transformation homogène	13
1.8	Géométrie des Robots à Structure Ouverte Simple	13
1.8.1	Principes de Description.....	14
1.8.2	Définition des Repères	14
1.8.3	Paramètres de Transformation entre R_{j-1} et R_j	14
1.9	Paramètres de <i>Denavit-Hartenberg</i>	15
1.10	Modélisation géométrique en chaine ouverte simple	16
1.10.1	Modèle géométrique direct.....	16
1.10.2	Modèle géométrique inverse	16
1.10.3	Le modèle cinématique robots à structures ouvertes simples	18
1.10.4	Modèle cinématique directe	18
1.10.5	Modèle cinématique inverse.....	18
1.10.6	Modèle dynamique du robot SCARA	19

1.10.7	Modélisation du robot <i>SCARA</i>	20
1.11	Calcul de Modèle géométrique direct.....	22
1.12	Calcul de modèle géométrique inverse.....	25
1.13	Calcul de modèle cinématique directe.....	26
1.14	Calcul de Modèle cinématique inverse.....	27
1.15	Modèle dynamique du robot <i>SCARA</i>	28
1.16	Espace de travail d'un robot	29
1.16.1	Principaux facteurs influençant l'espace de travail	30
1.17	Conclusion	30

Chapitre 2 : Commande du Robot *SCARA*

2.1	Introduction	32
2.2	Commande des mouvements articulaires	32
2.3	Généralités sur la régulation.....	33
2.3.1	Définition de la régulation.....	33
2.3.2	Objectifs dans le contexte du robot <i>SCARA</i>	33
2.4	Régulation.....	34
2.5	Régulation <i>PID</i> : principes fondamentaux	34
2.5.1	Formulation mathématique et structure du <i>PID</i>	34
2.5.2	Interprétation des composantes <i>P</i> , <i>I</i> et <i>D</i>	35
2.5.3	Atouts et limitations de la commande <i>PID</i>	36
2.6	COMMANDE EN POSITION.....	36
2.6.1	Régulation de la position articulaire	36
2.6.2	Architecture du contrôle en position.....	37
2.6.3	Schéma de la commande <i>PID</i> en position.....	37
2.6.4	Sélection des capteurs de position (encodeurs)	37

2.6.5	Réglage des paramètres <i>PID</i>	38
2.6.6	Simulation et évaluation des résultats (référence en θ).....	38
2.7	COMMANDE EN VITESSE	38
2.7.1	Régulation de la vitesse articulaire	38
2.7.2	Importance de la commande en vitesse (asservissement moteur)	38
2.7.3	Schéma de la commande <i>PID</i> en vitesse.....	39
2.7.4	Simulation et analyse des vitesses articulaires	39
2.8	GÉNÉRATION DE TRAJECTOIRES	40
2.8.1	Planification de trajectoires	40
2.8.2	Trajectoire en espace articulaire	40
2.8.3	Trajectoire en espace cartésien	40
2.9	Méthode d'interpolation	41
2.9.1	Polynôme de degré 3.....	41
2.10	Loi Bang-Bang	42
2.11	Simulation complète de la commande <i>PID</i>	43
2.11.1	Simulation des trajectoires « Q_{ref}, dQ_{ref} ».....	44
2.11.2	Référence 1 (Loi <i>Bang-Bang</i>).....	44
2.11.3	Référence 1 (Polynôme de degré 3).....	46
2.11.4	Simulation des Trajectoires « X_{ref}, dX_{ref} »	48
2.12	Conclusion	59

Chapitre 3 : Animation du Robot SCARA

3.1	Introduction	60
3.2	Outils de modélisation et d'animation	60
3.2.1	MATLAB et ses outils dédiés à la robotique	60
3.2.1.2	<i>Simscape Multibody</i>	61

3.2.1.3	Robotics System Toolbox	61
3.2.2	Logiciels de <i>CAO</i>	62
3.2.2.1	<i>SolidWorks</i>	62
3.2.3	Justification du choix	62
•	Interface intuitive et puissante	62
•	Outils d'assemblage avancés	63
•	Compatibilité directe avec <i>MATLAB</i>	63
3.3	Patrie animation du robot	63
3.3.1	Modélisation des pièces du robot	63
3.3.1.1	Base.....	63
3.3.1.2	Bras 1 (premier bras articulé)	63
3.3.1.3	Bras 2 (deuxième bras articulé)	64
3.3.1.4	Axe de l'effecteur terminal.....	64
3.3.2	Exportation et Simulation du modèle mécanique	66
3.3.2.2	Export <i>URDF</i> pour <i>MATLAB</i>	69
3.4	Patrie réalisation du robot.....	74
3.4.1	Conception mécanique du robot <i>SCARA</i>	74
3.4.2	Choix des composants électroniques	74
3.4.3	Impression <i>3D</i> des pièces du robot	75
3.4.4	Visualisation des pièces après impression <i>3D</i>	77
3.4.5	Schéma électronique sous Fritzing	78
3.4.6	Assemblage du Robot	78
3.4.7	Génération du code à partir de <i>MATLAB</i>	79
3.4.8	Implémentation dans <i>Arduino</i>	80
3.5	Conclusion.....	80

Conclusion Générale 81

Références 82

Liste des figures

Figure 1.1 L'automa cavaliere di Leonardo da Vinci [3].....	3
Figure 1.2 La structure d'un bras manipulateur [2].	5
Figure 1.3 Différentes types des robots [10].	8
Figure 1.4 Représentation d'un point [11].	9
Figure 1.5 Transformation des repères [11].	10
Figure 1.6 Transformation d'un vecteur [11].	11
Figure 1.7 Transformation de translation pure [11].	12
Figure 1.8 Transformation de rotation pure autour de l'axe x [11].	13
Figure 1.9 Robot à structure ouverte simple [11]	14
Figure 1.10 Paramètres géométrique dans le cas d'une structure ouverte simple [11]	15
Figure 1.11 Robot manipulateur SCARA [11].	21
Figure 1.12 Placement des repères [13].	21
Figure 1.13 Espace du travail du robot	30
Figure 2.1 Schéma de principe de régulateur <i>PID</i> [18].....	34
Figure 2.2 Schéma de la commande <i>PID</i> position.....	37
Figure 2.3 Schéma de la commande <i>PID</i> vitesse.....	39
Figure 2.4 Loi polynomiale de de degré trois.[11]	42
Figure 2.5 Loi de Bang-Bang [11].	43
Figure 2.6 Poursuite de position avec erreur des 4 articulations référence 1	44
Figure 2.7 Poursuite de vitesse avec erreur des 4 articulations référence 1	45
Figure 2.8 Poursuite de position avec erreur des 4 articulations référence 2	46
Figure 2.9 Poursuite de vitesse avec erreur des 4 articulations référence 1	47
Figure 2.10 Suivi de Trajectoire Cartésienne et Erreurs pour les position XE , YE , ZE , $ref1$	48
Figure 2.11 Suivi de Trajectoire Cartésienne et Erreurs pour les vitesses dXE , dYE , dZE , $ref1$	49
Figure 2.12 Poursuite de position avec erreur des 4 articulations	50
Figure 2.13 Poursuite de vitesse avec errer des 4 articulations	51
Figure 2.14 Trajectoire 3D des Positions Cartésiennes pour un Robot SCARA à 4 DDL.....	52
Figure 2.15 Trajectoire 3D des Positions Cartésiennes pour un Robot SCARA à 4DDL.....	52

Figure 2.16 Suivi de Trajectoire Cartésienne et Erreurs pour les position XE, YE, ZE.....	54
Figure 2.17 Suivi de Trajectoire Cartésienne et Erreurs pour les vitesses dXE, dYE, dZE	55
Figure 2.18 Poursuite de position avec erreur des 4 articulations	56
Figure 2.19 Poursuite de vitesse avec erreur des 4 articulations	57
Figure 2.20 Trajectoire 3D des Positions Cartésiennes pour un Robot SCARA à 4DDL.....	58
Figure 2.21 Trajectoire 3D des Positions Cartésiennes pour un Robot SCARA à 4DDL.....	58
Figure 3.1 La base du robot.....	63
Figure 3.2 Le 1-er bras	64
Figure 3.3 Le 2 -ème bras.....	64
Figure 3.4 Axe de l'effecteur terminal.....	65
Figure 3.5 Effecteur terminal.....	65
Figure 3.6 Assemblage du Robot.....	66
Figure 3.7 Schéma bloc du ROBOT SCARA.....	67
Figure 3.8 Schema de commande principale.....	68
Figure 3.9 Simulation Mechanics Explorer t=0.....	68
Figure 3.10 Simulation Mechanics Explorer t=10.....	69
Figure 3.11 Visualization URDF position initial ref1.....	70
Figure 3.12 Visualization URDF position finale ref1	70
Figure 3.13 Visualization URDF position initial ref2.....	70
Figure 3.14 Visualization URDF position finale ref2.....	71
Figure 3.15 Visualisation de la position finale en vue de haut	71
Figure 3.16 Visualisation Dynamique du Contrôle SCARA au départ ref1.....	72
Figure 3.17 Visualisation Dynamique du Contrôle SCARA finale ref1	72
Figure 3.18 Visualisation Dynamique du Contrôle SCARA au départ ref1.....	73
Figure 3.19 Visualisation Dynamique du Contrôle SCARA finale ref2.....	73
Figure 3.20 la structure complète du robot.....	74
Figure 3.21 Motif impression dans Luban	76
Figure 3.22 Generation de G-Code.....	76
Figure 3.23 Les pièces après impression 3D	77
Figure 3.24 Schéma électronique sous Fritzing.....	78
Figure 3.25 Assemblage du robot	79

Liste des Tableaux

Tableau 1.1 Types d'équations rencontrés avec la méthode de Paul [12].	17
Tableau 1.2 Paramètres de DH du robot SCARA [11].	22

Listes des Acronymes et Symboles

Listes des Acronymes

CAO : Conception Assistée par Ordinateur.

DDL : Degrés De Liberté.

D-H : Denavit-Hartenberg.

dXref : Vitesse de référence (dérivée de la position de consigne).

IA : Intelligence Artificielle.

Kd : Gain dérivé.

Ki : Gain intégral.

Kp : Gain proportionnel.

MCD : Modèle Cinématique Direct.

MCI : Modèle Cinématique Inverse.

MD : Modèle Dynamique.

MGD : Modèle Géométrique Direct.

MGI : Modèle Géométrique Inverse.

PID : Proportionnel-Intégral-Dérivé.

PME : Petite et Moyenne Entreprise.

ROS : Robot Operating System.

RRRP : Rotatif-Rotatif-Rotatif-Plan (P = translation linéaire).

SCARA: Selective Compliance Assembly Robot Arm.

STL: StereoLithography ou Standard Triangle Language.

URDF : Unified Robot Description Format.

X_{ref} : Position de référence (consigne).

3D : Représentation tridimensionnelle (*CAO*, visualisation, animation).

Listes des Symboles

q_i : position (angle ou translation) de l'articulation i .

\dot{q}_i : vitesse articulaire (dérivée de q_i).

\ddot{q}_i : accélération articulaire.

θ_i : angle de rotation autour de l'axe de l'articulation i .

d_i : distance de translation selon l'axe de l'articulation i .

l_1, l_2 : longueurs des segments du bras du robot *SCARA*.

lc_1, lc_2 : distances des centres de masse par rapport aux bases des segments.

m_1, m_2 : masses des différents segments du robot.

I_1, I_2, I_3 : moments d'inertie des segments.

g : gravité (accélération gravitationnelle, en m/s^2).

X_{ref} : position de référence dans l'espace opérationnel (consigne).

\dot{X}_{ref} : vitesse de référence (dérivée de la consigne positionnelle).

e : erreur (écart entre la consigne et la valeur mesurée).

K_p : gain proportionnel du correcteur *PID*. K_i : gain intégral du correcteur *PID*.

K_d : gain dérivé du correcteur *PID*.

τ : couple appliqué sur une articulation.

$M(q)$: matrice d'inertie dépendante de la configuration q .

$C(q, \dot{q}_i)$: matrice des forces de Coriolis et centrifuges.

$J(q)$: matrice Jacobienne reliant les vitesses articulaires aux vitesses cartésiennes.

iT_j : Matrice de transformation.

Introduction Générale

Introduction Générale

La robotique industrielle joue un rôle essentiel dans l'automatisation des tâches répétitives, rapides et de haute précision. Parmi les différentes architectures existantes, le robot SCARA (Selective Compliance Assembly Robot Arm) se distingue par sa rigidité dans le plan horizontal, sa précision et sa rapidité, ce qui en fait un choix privilégié pour les opérations d'assemblage [1].

Ce mémoire porte sur l'étude complète d'un robot SCARA à quatre degrés de liberté, depuis sa modélisation jusqu'à sa réalisation virtuelle. L'objectif est de concevoir une chaîne cohérente incluant la modélisation mécanique, le développement d'une commande efficace basée sur des régulateurs PID pour le contrôle en position et en vitesse, ainsi que l'animation dynamique du robot dans un environnement de simulation unifié sous MATLAB/Simulink. L'ensemble du processus vise à aboutir à une représentation fidèle du comportement réel du robot, dans le but de valider les performances de la commande et de visualiser son fonctionnement.

La problématique centrale traitée dans ce travail est la suivante :

Comment modéliser, commander et animer de manière cohérente un robot SCARA à 4 degrés de liberté dans un environnement de simulation intégré ?

Pour y répondre, le mémoire est structuré en trois chapitres complémentaires :

Chapitre 1 : présentation des notions fondamentales des robots manipulateurs, définition des paramètres géométriques et dynamiques du robot SCARA, et modélisation mécanique.

Chapitre 2 : élaboration et mise en œuvre de la commande en position et en vitesse à l'aide de régulateurs PID, accompagnée de simulations pour évaluer la précision et la stabilité.

Chapitre 3 : animation et visualisation du robot dans MATLAB/Simulink, avec intégration d'un modèle dynamique pour une simulation réaliste.

Ce travail vise à proposer une démarche structurée, allant de l'étude théorique à la validation pratique dans un environnement numérique, pour aboutir à la réalisation virtuelle complète d'un robot SCARA fonctionnel.

Chapitre 1 : Modélisation du Robot *SCARA*

1 Chapitre 1 : Modélisation du Robot SCARA

1.1 Introduction

Les robots *SCARA* sont couramment employés dans l'industrie pour diverses applications telles que l'assemblage, la manipulation de pièces et la gravure. Leur conception mécanique offre une rigidité notable sur l'axe vertical tout en autorisant une souplesse dans le plan horizontal, ce qui en fait un choix adapté à de nombreuses tâches de précision [2].

Ce chapitre s'intéresse à l'élaboration d'un modèle mathématique d'un robot *SCARA* à quatre degrés de liberté. L'objectif est d'établir les relations cinématiques et dynamiques permettant de définir sa commande en vitesse et d'en assurer l'animation. La modélisation repose sur les paramètres physiques et géométriques du système, en s'appuyant sur les fondements de la robotique.

1.2 Historique de la robotique

1.2.1 Origines et premières inventions

Antiquité : Les Égyptiens développent des mécanismes automatisés comme l'horloge à eau.

Grèce antique : Archimède (III^e siècle av. J.-C.) met au point des systèmes de leviers et de poulies imitant les mouvements humains.

Moyen Âge : Apparition des premiers automates (ex. : homme de fer d'Alberto Magno au XIII^e siècle).

Renaissance : Léonard de Vinci conçoit l'Automa Cavaliere (1495), capable de bouger bras et jambes.

XVII^e siècle : Leibniz invente le système binaire, fondement de l'informatique.

Chapitre 01 : Modélisation du Robot SCARA

1.2.2 Le XXe siècle : naissance de la robotique moderne

- **1937** : Bill Taylor crée Gargantua, premier robot industriel.
- **1939** : Présentation de robots industriels et domestiques à la foire de New York.
- **1950** : Isaac Asimov publie Robot et introduit les trois lois de la robotique.
- **1960s** : Unimation développe le premier robot programmable, utilisé par General Motors pour l'automatisation industrielle.
- **1980s-1990s** : Apparition de l'intelligence artificielle et du Big Data, révolutionnant la robotique.

1.2.3 Le XXIe siècle : l'ère de l'intelligence artificielle

Robots dans la vie quotidienne : Aspirateurs autonomes, drones, véhicules sans conducteur, robots chirurgicaux...

Quatrième Révolution Industrielle : Transformation des industries grâce aux machines autonomes.

Robotique industrielle : Désormais accessible aux *PME*, favorisant l'innovation et la compétitivité.

Tendances actuelles : Collaboration homme-machine, flexibilité et diversification des usages de la robotique [1].

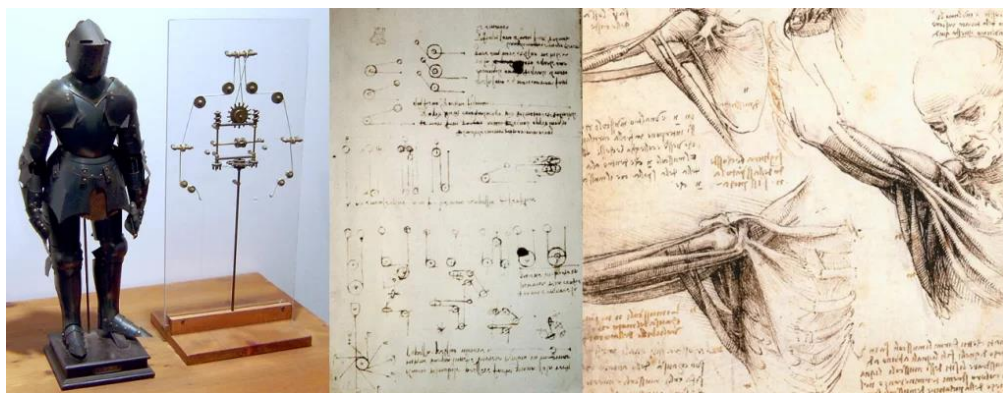


Figure 1.1 L'automa cavaliere di Leonardo da Vinci [3]

1.3 Définitions

1.3.1 Robotique

La robotique est un domaine multidisciplinaire qui englobe l'étude, la conception et la construction de robots ou de machines automatiques. Elle combine des connaissances en électronique, informatique et mécanique pour développer des systèmes capables de percevoir leur environnement, de prendre des décisions et d'agir de manière autonome [4].

1.3.2 Robot

Un robot autonome doit être capable de percevoir son environnement, de se déplacer, d'agir, de planifier et d'apprendre [5].

1.4 Constitution des robots

Un robot est constitué de plusieurs éléments essentiels qui assurent son fonctionnement et son interaction avec l'environnement :

- Structure mécanique

Comprend l'ossature (châssis, bras articulés) et les systèmes de déplacement (roues, chenilles, jambes), fabriqués à partir de matériaux adaptés (métal, plastique, composites).

- Actionneurs

Permettent le mouvement grâce à différents types de moteurs (électriques, hydrauliques ou pneumatiques), utilisés selon la puissance et la précision requises.

- Capteurs

Facilitent la perception de l'environnement. On retrouve des capteurs de position (encodeurs), de vision (caméras), de distance (ultrasons, *LiDAR*) et tactiles (capteurs de force et de pression).

- Unité de commande

Correspond au "cerveau" du robot, qui analyse les informations des capteurs et pilote les actionneurs. Elle peut être basée sur des microcontrôleurs (*Arduino*, *STM32*) ou des ordinateurs embarqués (*Raspberry Pi*, *Jetson*) avec des algorithmes d'intelligence artificielle.

Chapitre 01 : Modélisation du Robot SCARA

- Alimentation

Fournit l'énergie nécessaire au fonctionnement, via des batteries (lithium-ion, plomb-acide), une alimentation filaire ou des sources alternatives (solaire, hydrogène).

- Programmation et logiciels

Le robot est contrôlé par des programmes utilisant des langages comme Python ou C++. Il peut fonctionner avec des systèmes comme *ROS* (Robot Operating System) pour gérer ses actions et optimiser ses déplacements [6]

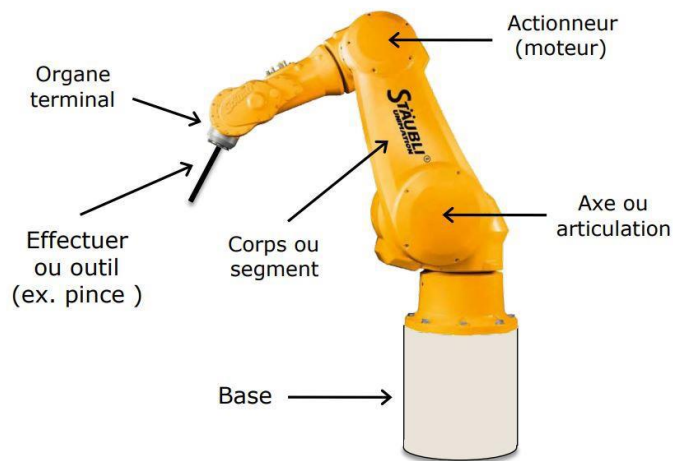


Figure 1.2 La structure d'un bras manipulateur [2].

1.5 Notions base de la robotique

1.5.1 Articulation

En robotique, une articulation est une liaison mécanique reliant deux segments du robot, permettant un mouvement relatif entre eux [7].

1.5.2 Espace articulaire

L'espace articulaire, ou espace de configuration, est l'ensemble des positions possibles des articulations d'un robot, définies par les angles ou les déplacements de chaque articulation [8].

1.5.3 Espace opérationnel

L'espace opérationnel est l'ensemble des positions et orientations que l'organe terminal (effecteur) d'un robot peut atteindre dans son environnement de travail [8].

1.5.4 Degré de liberté (DDL)

Le degré de liberté d'un mécanisme correspond au nombre de paramètres indépendants nécessaires pour définir sa position à un instant donné [8].

1.6 Types des robots

Les robots peuvent être classés selon plusieurs critères, en voici quelques types courants [9].

1.6.1 En fonction de la structure

- **Robots articulés** : Ces robots ont des articulations semblables à celles des bras humains, souvent utilisés dans l'industrie pour des tâches comme la soudure, l'assemblage, etc.
- **Robots SCARA** : Ce sont des robots à bras rigides avec un mouvement en "plan horizontal", principalement utilisés dans l'assemblage de précision.
- **Robots Delta** : Des robots à 3 bras articulés qui se déplacent dans l'espace tridimensionnel, utilisés pour des applications de *pick-and-place* rapides.
- **Robots parallèles** : Ils possèdent une structure qui relie plusieurs bras à un même point fixe, souvent utilisés pour des applications où la précision et la rigidité sont cruciales.

1.6.2 En fonction de leur mobilité

- **Robots mobiles** : Ils peuvent se déplacer, généralement en suivant un chemin prédéfini ou en se déplaçant de manière autonome (par exemple, des robots de service ou des robots de nettoyage).
- **Robots stationnaires** : Ces robots n'ont pas de capacité de déplacement mais accomplissent leurs tâches en restant dans une position fixe (comme un bras robotique).

1.6.3 En fonction de leur application

- **Robots industriels** : Utilisés dans les usines et pour des tâches comme le soudage, le montage, la peinture, la manipulation de matériaux, ...etc.
- **Robots de service** : Employés dans des environnements domestiques, médicaux ou de maintenance, comme les robots d'aspiration ou d'assistance aux personnes âgées.
- **Robots humanoïdes** : Conçus pour ressembler à l'humain, avec des bras, des jambes et un visage, utilisés pour la recherche, l'éducation ou même l'interaction avec les gens.
- **Robots de recherche** : Utilisés principalement pour explorer des environnements difficiles d'accès, comme les fonds marins ou l'espace (rovers martiens, par exemple).

1.6.4 En fonction de leur méthode de contrôle

- **Robots autonomes** : Ils sont capables de prendre des décisions et d'agir sans intervention humaine, grâce à des capteurs et des algorithmes de contrôle.
- **Robots téléopérés** : Ces robots sont contrôlés à distance par un opérateur humain, souvent utilisés dans des environnements où la présence humaine est risquée (sauvetage, déminage).

1.6.5 En fonction de leur source d'énergie

- **Robots électriques** : Alimentés par des moteurs électriques, généralement utilisés dans des applications où une grande précision est requise.
- **Robots hydrauliques** : Utilisent des systèmes hydrauliques pour effectuer des mouvements puissants, souvent dans des environnements industriels.
- **Robots pneumatiques** : Utilisent l'air comprimé pour leurs mouvements, adaptés aux tâches nécessitant une force modérée.

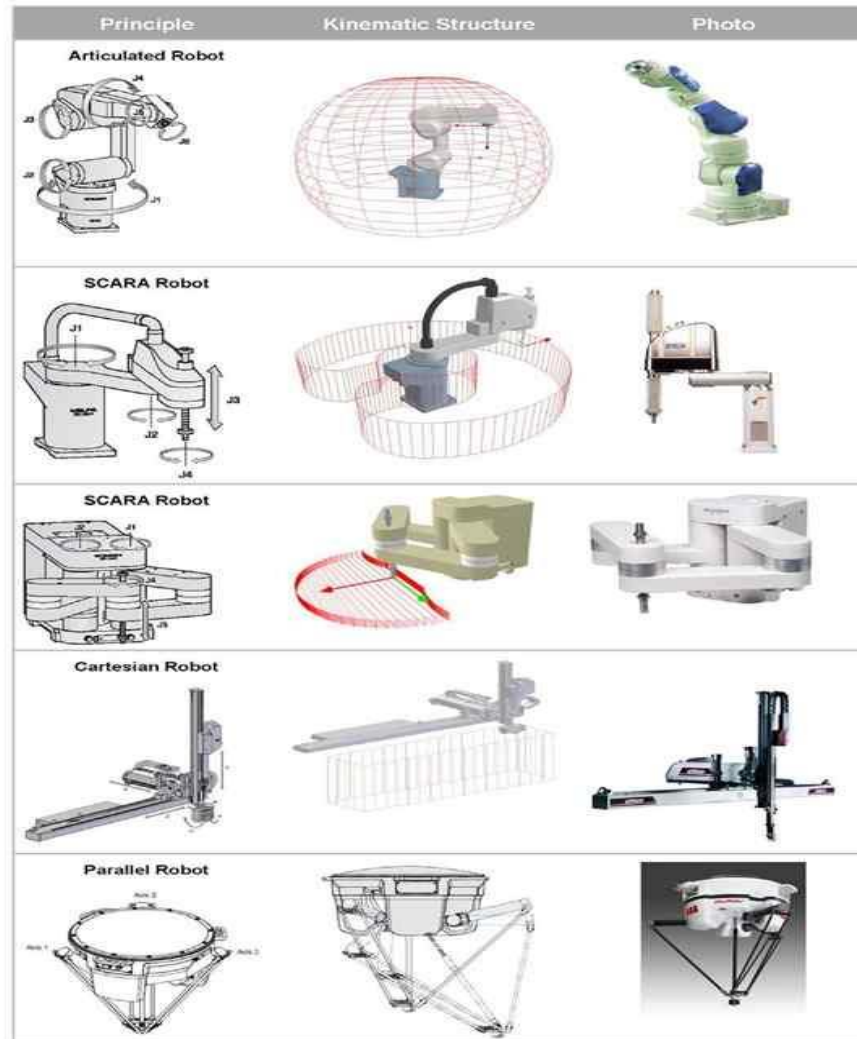


Figure 1.3 Différentes types des robots [10].

1.7 Notations et outils mathématiques pour la modélisation des robots

1.7.1 Représentation d'un point

Soit P un point défini par ses coordonnées cartésiennes (P_x, P_y, P_z) . Les coordonnées homogènes de ce point sont données par $w.p_x, w.p_y, w.p_z$ où w est un facteur d'échelle. En robotique, ce facteur est généralement fixé à 1. Ainsi, les coordonnées homogènes du point P peuvent être exprimées sous la forme du vecteur [11].

$$\mathbf{P} = \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} \quad (1.1)$$

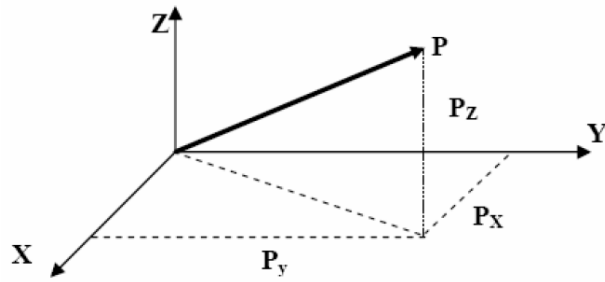


Figure 1.4 Représentation d'un point [11].

1.7.2 Représentation d'une direction

Une direction, représentée par un vecteur libre, est décrite à l'aide de quatre composantes, la dernière étant nulle, ce qui correspond à un point situé à l'infini. Lorsqu'on enregistre les coordonnées cartésiennes d'un vecteur unitaire U , elles sont notées (U_x, U_y, U_z) . En coordonnées homogènes, ce vecteur s'écrit alors [11].

$$U = \begin{bmatrix} U_x \\ U_y \\ U_z \\ 0 \end{bmatrix} \quad (1.2)$$

1.7.3 Représentation d'un plan

Un vecteur ligne Q représente le plan, avec :

$$Q = [\alpha, \beta, \gamma, \delta] \quad (1.3)$$

Où les coefficients $\alpha, \beta, \gamma, \delta$:

α Composante x du vecteur normal.

β Composante y du vecteur normal.

γ Composante z du vecteur normal.

δ Terme constant (position du plan).

Pour tout point P appartenant au plan Q le produit matriciel $Q P$ est nul :

$$Q P = [\alpha, \beta, \gamma, \delta] \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix} = 0 \quad (1.4)$$

1.7.4 Transformations homogènes des repères

Appliquons une transformation quelconque, composée d'une translation et/ou d'une rotation, au repère R_i , de manière à le déplacer vers le repère R_j . Cette transformation est décrite par la matrice ${}^i T_j$, appelée matrice de transformation homogène, qui est de dimension 4×4 et s'exprime sous la forme

$${}^i T_j = [{}^i s_j \ {}^i n_j \ {}^i a_j \ {}^i p_j] = \begin{bmatrix} s_x & n_x & a_x & p_x \\ s_y & n_y & a_y & p_y \\ s_z & n_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.5)$$

Où ${}^i s_j$, ${}^i n_j$ et ${}^i a_j$ représentent respectivement les vecteurs unitaires suivant les axes x_j , y_j et z_j du repère R_j , exprimés dans le repère R_i . De plus, ${}^i p_j$ désigne le vecteur position de l'origine du repère R_j par rapport au repère R_i .

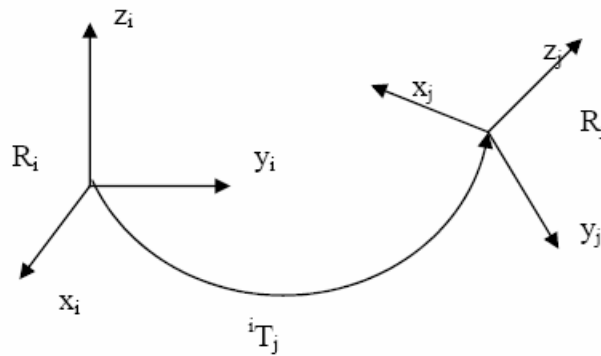


Figure 1.5 Transformation des repères [11].

La matrice ${}^i T_j$ définit le repère R_j dans le repère R_i . Par la suite on notera souvent la matrice de transformation [1.6] sous forme d'une matrice partitionnée.

$${}^i T_j = \begin{bmatrix} {}^i A_j & {}^i P_j \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} {}^i s_j & {}^i n_j & {}^i a_j & {}^i p_j \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.6)$$

1.7.5 Transformation des vecteurs

Soit un vecteur jP_1 représentant la position du point P_1 dans le repère R_j (voir Figure.....). En utilisant la définition des coordonnées homogènes, les coordonnées du point P_1 dans le repère R_i sont obtenues à l'aide de l'équation suivante

$${}^iP_1 = {}^i(O_iP_1) = {}^is_j {}^ip_{1x} + {}^jn_i {}^jp_{1y} + {}^ja_i {}^jp_{1z} + {}^ip_j = {}^iT_j {}^jp_1 \quad (1.7)$$

Ainsi, la matrice de transformation iT_j permet de convertir les coordonnées d'un point exprimé dans le repère R_j vers le repère R_i .

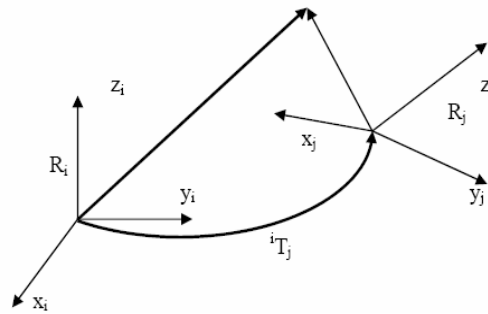


Figure 1.6 Transformation d'un vecteur [11].

1.7.6 Transformation des plans

La position relative d'un point par rapport à un plan reste invariante quelles que soient les transformations appliquées à l'ensemble constitué du point et du plan. Ainsi

$${}^jQ^jP = {}^iP^iQ = {}^iQ^iT_j^jP \quad (1.8)$$

Donc

$${}^jQ = {}^iQ^iT_j \quad (1.9)$$

1.7.7 Matrice de transformation de translation pure

Cette transformation est appelée Trans (a, b, c) ou a, b, c sont les éléments de la translation le long des axes x, y et z respectivement.

$${}^iT_j = Trans(a, b, c) = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.10)$$

Chapitre 01 : Modélisation du Robot SCARA

L'orientation étant conservée dans cette transformation, Trans (a, b, c) a pour expression (figure 1.7)

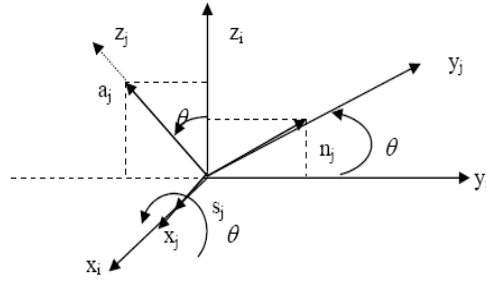


Figure 1.7 Transformation de translation pure [11].

1.7.8 Matrice de transformation de rotation autour des axes principaux

1.7.8.1 Matrice de transformation correspondant à une rotation θ autour de l'axe x

Cette rotation est notée $Rot(x, \theta)$. Les composantes des vecteurs unitaires ${}^i s_j$, ${}^i n_j$, et ${}^i a_j$, respectivement alignés avec les axes x_j , y_j et z_j du repère R_i , sont déterminées à partir de la figure.... et exprimées dans R_i .

En utilisant les abréviations $S\theta$ et $C\theta$ pour désigner respectivement le sinus et le cosinus de l'angle θ , et en notant que la transposition d'une matrice est indiquée par l'exposant T , on peut exprimer ces vecteurs sous une forme compacte.

On obtient ainsi :

$${}^i T_j = Rot(x, \theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C\theta & -S\theta & 0 \\ 0 & S\theta & C\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} Rot(y, \theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.11)$$

La matrice d'orientation de dimension (3×3) est appelée $Rot(x, \theta)$.

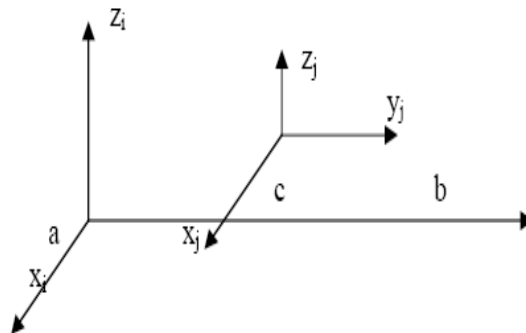


Figure 1.8 Transformation de rotation pure autour de l'axe x [11].

1.7.8.2 Matrice de transformation correspondant à une rotation θ autour de l'axe y

En utilisant un raisonnement similaire, on obtient :

$${}^i T_j = Rot(x, \theta) = \begin{bmatrix} C\theta & 0 & S\theta & 0 \\ 0 & 1 & 0 & 0 \\ -S\theta & 0 & C\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} Rot(x, \theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.12)$$

1.7.8.3 Matrice de transformation correspondant à une rotation θ autour de l'axe z

Il est vérifié que :

$${}^i T_j = Rot(z, \theta) = \begin{bmatrix} C\theta & -S\theta & 0 & 0 \\ S\theta & C\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} Rot(z, \theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.13)$$

1.7.9 Propriétés des matrices de transformation homogène

On vérifie aisément que :

$$Rot^{-1}(u, \theta) = Rot(u, -\theta) = Rot(-u, \theta) \quad (1.14)$$

$$Trans^{-1}(u, d) = Trans(u, -d) = Trans(-u, d) \quad (1.15)$$

La transformation d'un repère R_o peut être déduite de la composition des multiplications à droite de ces transformations si chaque transformation i , ($i=1, \dots, k$), est définie par rapport au repère courant R_{i-1}

$${}^0 T_k = {}^0 T_1 {}^1 T_2 {}^2 T_3 \dots {}^{k-1} T_k \quad (1.16)$$

1.8 Géométrie des Robots à Structure Ouverte Simple

Un robot à structure ouverte est composé de $n+1$ segments rigides $C_0, C_1 \dots C_n$ reliés par n articulations. Le premier corps C_0 correspond à la base, tandis que C_n porte l'effecteur final. Chaque articulation j relie un segment C_j au précédent C_{j-1} .

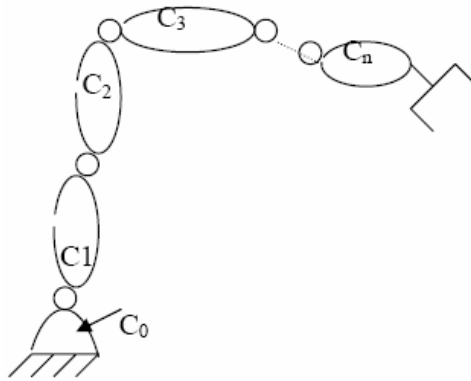


Figure 1.9 Robot à structure ouverte simple [11]

1.8.1 Principes de Description

Les segments sont considérés parfaitement rigides et connectés par des articulations idéales (sans jeu ni déformation), qui peuvent être rotoïdes (rotation) ou prismatiques (translation). Chaque segment C_j possède un repère R_j . La variable de l'articulation j est notée q_j .

1.8.2 Définition des Repères

Le repère R_j , fixé à C_j , est défini ainsi :

- L'axe z_j correspond à l'axe de l'articulation j .
- L'axe x_j est orienté selon la perpendiculaire commune aux axes z_j et z_{j+1} . En cas de parallélisme, il est choisi selon des critères de symétrie ou de simplicité.

1.8.3 Paramètres de Transformation entre R_{j-1} et R_j

La transition d'un repère à un autre repose sur quatre paramètres géométriques :

α_j : Angle entre z_{j-1} et z_j (rotation autour de x_{j-1}).

d_j : Distance entre z_{j-1} et z_j selon x_{j-1} .

θ_j : Angle entre x_{j-1} et x_j (rotation autour de z_j).

r_j : Distance entre x_{j-1} et x_j selon z_j .

Ces paramètres permettent de modéliser la cinématique des robots et d'exprimer les transformations entre les repères successifs à l'aide de matrices homogènes.

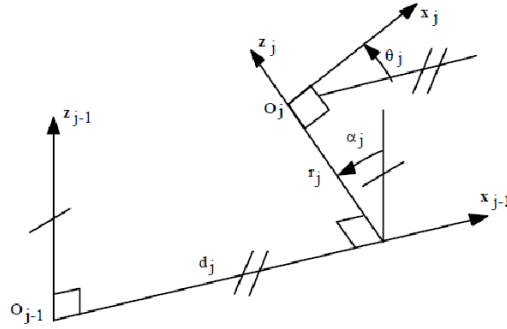


Figure 1.10 Paramètres géométrique dans le cas d'une structure ouverte simple [11]

La variable articulaire q_j associée à la $j^{ème}$ articulation est soit θ_j , soit r_j , ce qui se traduit par la relation suivante

$$q_j = \bar{\sigma}_j \theta_j + \sigma_j r_j \quad (1.17)$$

Avec :

$\sigma_j = 0$ si articulation j est rotoïde

$\sigma_j = 1$ si articulation j est prismatique

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

$$\begin{aligned} {}^{j-1}\mathbf{T}_j &= \text{Rot}(x, \alpha_j) \text{Trans}(x, d_j) \text{Rot}(z, \theta_j) \text{Trans}(z, r_j) \\ &= \begin{bmatrix} C\theta_j & -S\theta_j & 0 & d_j \\ C\alpha_j S\theta_j & C\alpha_j C\theta_j & -S\alpha_j & -r_j S\alpha_j \\ S\alpha_j S\theta_j & S\alpha_j C\theta_j & C\alpha_j & r_j C\alpha_j \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \quad (1.18)$$

Avec,

$$C\theta_j = \cos\theta_j ; S\theta_j = \sin\theta_j ; C\alpha_j = \cos\alpha_j ; S\alpha_j = \sin\alpha_j \quad (1.19)$$

La matrice ${}^{j-1}\mathbf{T}_j$ est dite : La matrice de transformation *Denavit-Hartenberg*

1.9 Paramètres de *Denavit-Hartenberg*

Denavit et *Hartenberg* ont proposé une méthode permettant d'attribuer un repère unique à chaque segment d'un robot. Cette approche systématique simplifie la transition entre les différentes articulations d'un système robotique.

Chapitre 01 : Modélisation du Robot SCARA

Dans les chaînes cinématiques ouvertes, chaque articulation possède un seul degré de liberté, garantissant un contact constant entre les surfaces adjacentes. En sélectionnant judicieusement les repères associés aux liaisons, il devient plus aisé de déterminer les matrices homogènes de *Denavit-Hartenberg*. Cela permet d'exprimer efficacement la position et l'orientation de l'effecteur par rapport à la base du robot, et inversement [12].[13].

1.10 Modélisation géométrique en chaîne ouverte simple

1.10.1 Modèle géométrique direct

Le modèle géométrique direct (*MGD*) regroupe l'ensemble des relations permettant de déterminer la position et l'orientation de l'effecteur en fonction des coordonnées articulaires du robot. Il établit ainsi le lien entre l'espace des joints et l'espace opérationnel [12].

Ce modèle peut être représenté sous forme de matrice de transformation homogène, notée 0T_n , qui exprime la position et l'orientation du repère lié à l'organe terminal par rapport au repère de base du robot.

$${}^0T_n = {}^0T_1(q_1) {}^1T_2(q_2) \dots {}^{k-1}T_n(q_n) \quad (1.20)$$

On peut également représenter le modèle géométrique direct du robot en utilisant la relation suivante $X = (q)$

Avec q comme vecteur des variables articulaires telles que

$$q = [q_1 q_2 q_3 \dots q_n] \quad (1.21)$$

Et les coordonnées opérationnelles

$$X = [x_1 x_2 x_3 \dots x_m]^T \quad (1.22)$$

1.10.2 Modèle géométrique inverse

Le *MGI* consiste à déterminer les coordonnées articulaires nécessaires pour que l'effecteur atteigne une position et une orientation données, spécifiées dans l'espace opérationnel. Cette opération, souvent appelée transformation de coordonnées ou changement de coordonnées, est essentielle en robotique pour la planification des mouvements et le contrôle des trajectoires [12].

Chapitre 01 : Modélisation du Robot SCARA

Lorsqu'une expression explicite permet de déterminer toutes les solutions possibles au modèle géométrique inverse (*MGI*). Plusieurs méthodes existent pour résoudre ce problème, notamment :

La méthode de Paul, qui traite chaque situation de manière individuelle et s'adapte à la majorité des robots industriels.

La méthode de Pieper, particulièrement efficace pour les robots à six degrés de liberté, lorsqu'ils possèdent trois articulations rotoïdes aux axes concourants ou trois articulations prismatiques alignées.

D'autres approches existent selon la configuration du robot et les contraintes spécifiques du système.

Type 1	$Xr_i = Y$
Type 2	$XS\theta_i + YC\theta_i = Z$
Type 3	$X_1 S\theta_i + Y_1 C\theta_i = Z_1$ $X_2 S\theta_i + Y_2 C\theta_i = Z_2$
Type 4	$X_1 r_j S\theta_i = Y_1$ $X_2 r_j C\theta_i = Y_2$
Type 5	$X_1 S\theta_i = Y_1 + Z_1 r_j$ $X_2 C\theta_i = Y_2 + Z_2 r_j$
Type 6	$W S\theta_i = X C\theta_i + Y S\theta_i + Z_1$ $W C\theta_i = X S\theta_i - Y C\theta_i + Z_2$
Type 7	$W_1 C\theta_i + W_2 S\theta_i = X C\theta_i + Y S\theta_i + Z_1$ $W_1 S\theta_i + W_2 C\theta_i = X S\theta_i - Y C\theta_i + Z_2$
Type 8	$X C\theta_i + Y C(\theta_1 + \theta_2) = Z_1$ $X S\theta_i + Y S(\theta_1 + \theta_2) = Z_2$

Tableau 1.1 Types d'équations rencontrés avec la méthode de Paul [12].

1.10.3 Le modèle cinématique robots à structures ouvertes simples

Le modèle cinématique établit le lien entre les vitesses articulaires et celles d'un point de référence du robot, généralement son extrémité. Il complète ou remplace le modèle géométrique en intégrant les informations sur les vitesses, facilitant ainsi la planification des trajectoires.

Son principal avantage réside dans la simplification des équations, en remplaçant les relations non linéaires du modèle géométrique par des équations linéaires, ce qui facilite le calcul des transformations inverses et l'optimisation des mouvements du robot [12].

1.10.4 Modèle cinématique directe

Le modèle cinématique direct (*MCD*) établit la relation entre les vitesses des articulations et la vitesse de l'effecteur d'un robot

Il permet de déterminer comment une variation des coordonnées articulaires influence la vitesse et l'orientation de l'organe terminal.

Mathématiquement, cette relation est exprimée à l'aide du jacobien du robot $J(q)$, une matrice qui relie les vitesses articulaires aux vitesses cartésiennes de l'effecteur. Cette approche est essentielle pour le contrôle en vitesse, la planification des trajectoires, et l'évitement d'obstacles [12].

$$\dot{X} = J\dot{q} \quad (1.23)$$

Avec

$$\dot{X} = dX \quad (1.24)$$

$$J = J(q) \quad (1.25)$$

$$\dot{q} = dq \quad (1.26)$$

1.10.5 Modèle cinématique inverse

Le modèle cinématique inverse (*MCI*) vise à déterminer les vitesses articulaires \dot{q} nécessaires pour que l'effecteur suive une vitesse opérationnelle \dot{X} donnée, en fonction d'une configuration articulaire spécifique q .

Chapitre 01 : Modélisation du Robot SCARA

De manière similaire, le modèle différentiel inverse permet de calculer l'infinitésimale des coordonnées articulaires dq à partir d'une variation infinitésimale des coordonnées opérationnelles dX .

L'obtention du modèle cinématique inverse repose sur la résolution d'un système d'équations linéaires, en inversant le modèle cinématique direct. Mathématiquement [12].

Il s'exprime sous la forme :

$$\dot{q} = J^{-1} \dot{X} \quad (1.27)$$

Où J est la matrice jacobienne reliant les vitesses articulaires aux vitesses de l'effecteur.

1.10.6 Modèle dynamique du robot SCARA

On parle communément de modèle dynamique inverse, ou plus simplement de modèle dynamique [13].

Le modèle dynamique direct, quant à lui, permet de déterminer les accélérations articulaires à partir des positions, vitesses et couples appliqués aux articulations. Il peut s'exprimer de la manière suivante :

$$\ddot{q} = \ddot{q}(q, \dot{q}, f) \quad (1.28)$$

Ce type de modélisation trouve plusieurs applications, parmi lesquelles :

- La simulation basée sur le modèle dynamique direct.
- Le choix et le dimensionnement des actionneurs.
- L'identification des paramètres inertiels et des coefficients de frottement du robot.
- L'élaboration de lois de commande via le modèle dynamique inverse.

Dans le cadre de cette étude, le système robotique pris en compte est un manipulateur rigide de type SCARA à quatre degrés de liberté. Ce bras robotisé est constitué de trois articulations rotatives (θ_1 , θ_2 et θ_3), ainsi que d'un axe de translation noté d_4 [14].

Chapitre 01 : Modélisation du Robot SCARA

La modélisation dynamique du mouvement dans l'espace articulaire de ce manipulateur peut être obtenue en appliquant la formulation d'*Euler-Lagrange*, qui permet de décrire précisément son comportement mécanique.

La dynamique du mouvement dans l'espace articulaire de ce manipulateur peut être analysée à partir de son modèle mathématique, construit à l'aide de la formulation d'Euler-Lagrange [13].

Dans ce cadre, les positions angulaires θ_1 , θ_2 et θ_3 sont assimilées aux coordonnées généralisées q_1 , q_2 et q_3 , tandis que le déplacement vertical d_4 de l'effecteur final est représenté par la coordonnée généralisée q_4 .

Ainsi, le modèle dynamique du robot, exprimé sous forme matricielle, s'écrit comme suit

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + \dot{g} = \tau \quad (1.29)$$

Dans cette équation :

$M(q) \in \mathbb{R}^n$ est la matrice d'inertie du système.

$C(q, \dot{q})\dot{q} \in \mathbb{R}^{n \times n}$ représente les effets de Coriolis et des forces centrifuges.

$B = 0$ est la matrice diagonale des coefficients de friction visqueux, considérés ici comme négligeables.

$g(q) \in \mathbb{R}$ est le vecteur des forces gravitationnelles.

$\tau \in \mathbb{R}^n$ désigne le vecteur des couples appliqués aux articulations.

1.10.7 Modélisation du robot SCARA

Le robot SCARA est un robot industriel couramment utilisé, généralement avec une configuration à quatre axes. Sa zone de travail est plus réduite que celle des robots à six axes, mais il permet des déplacements selon les axes X, Y et Z, ainsi qu'une rotation complète autour de l'axe Z. Cela crée une enveloppe de travail en forme de cylindre.

Ce type de robot est plus compact et plus rapide grâce au nombre limité d'articulations à gérer, ce qui facilite les mouvements et permet des vitesses d'exécution plus élevées. Les robots SCARA sont utilisés dans de nombreux domaines industriels, notamment pour l'assemblage, la

Chapitre 01 : Modélisation du Robot SCARA

manutention, l'impression 3D, les inspections, la palettisation ou encore la distribution de matériaux.

Le terme "*Selective Compliance Articulated Robot Arm*" fait référence à leur capacité à être flexibles dans certaines directions tout en restant rigides dans d'autres [15].

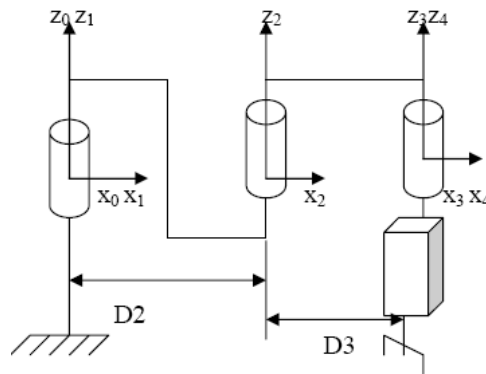


Figure 1.11 Robot manipulateur SCARA [11].

Le robot SCARA est un robot ayant quatre degrés de liberté, dont trois sont des articulations rotatives et un est prismatique, comme montré sur la figure 1.11. Il fait partie des robots à structure ouverte simple.

En suivant la méthode présentée précédemment, les repères ont été placés comme indiqué sur la figure 1.12.

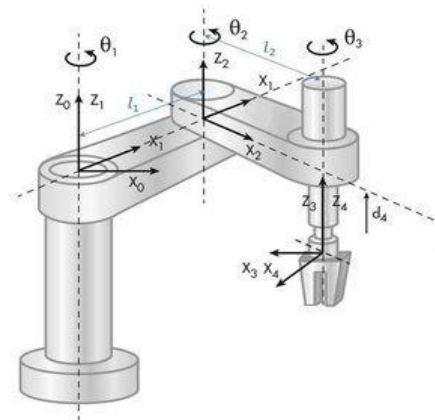


Figure 1.12 Placement des repères [13].

Chapitre 01 : Modélisation du Robot SCARA

1.11 Calcul de Modèle géométrique direct

Ce modèle permet de savoir la position exacte de l'organe terminal à partir de les angles θ . Dans notre cas notre robot *RRRP* exprime la position P en fonction de θ_1, θ_2 et θ_3 .

On retrouve les caractéristiques géométriques du robot *SCARA* dans le tableau

j	σ_j	α_j	d_j	θ_j	r_j
1	0	0	0	θ_j	0
2	0	0	D2	θ_j	0
3	0	0	D3	θ_j	0
4	1	0	0	0	r_4

Tableau 1.2 Paramètres de DH du robot *SCARA* [11].

Pour déterminer la Modélisation Géométrique Directe (*MGD*) du robot *SCARA*, on cherche à obtenir la matrice de transformation homogène 0T_4 , qui relie la base (repère 0) à l'effecteur (repère 4). Cette démarche repose sur l'utilisation de la convention de *Denavit-Hartenberg* (*D-H*). En s'appuyant sur le tableau des paramètres (Tableau 1.2) et sur la relation (1.13), la matrice globale peut être obtenue à travers le produit successif des matrices de transformation élémentaires :

$${}^0T_E = {}^0T_1 {}^1T_2 {}^2T_3 {}^3T_4 {}^4T_E \quad (1.30)$$

Avec ;

$${}^0T_1 = A1 = \begin{bmatrix} C1 & -S1 & 0 & 0 \\ S1 & C1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad {}^1T_2 = A2 = \begin{bmatrix} C2 & -S2 & 0 & D2 \\ S2 & C2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

$${}^2T_3 = A3 = \begin{bmatrix} C3 & -S3 & 0 & D3 \\ S3 & C3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad {}^3T_4 = A4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & r4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Notations :

Chapitre 01 : Modélisation du Robot SCARA

Les termes ci et si représentent respectivement le cosinus et le sinus de l'angle θ_i .

$$\sin(\theta_i) = si; \cos(\theta_i) = ci$$

$${}^2T_4 = {}^2T_3 \times {}^3T_4 = \begin{bmatrix} C3 & -S3 & 0 & D3 \\ S3 & C3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & r4 \\ 0 & 0 & 0 & 1 \end{bmatrix};$$

Alors ;

$${}^2T_4 = {}^2T_3 \times {}^3T_4 = \begin{bmatrix} C3 & -S3 & 0 & D3 \\ S3 & C3 & 0 & 0 \\ 0 & 0 & 1 & r4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^1T_4 = {}^1T_2 \times {}^2T_4 = \begin{bmatrix} C2 & -S2 & 0 & D2 \\ S2 & C2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} C3 & -S3 & 0 & D3 \\ S3 & C3 & 0 & 0 \\ 0 & 0 & 1 & r4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Alors ;

$${}^1T_4 = {}^1T_2 \times {}^2T_4 = \begin{bmatrix} C23 - S23 & -C2S3 - S2C3 & 0 & C2D3 + D2 \\ S2C3 + C2S3 & -S23 + C2C3 & 0 & S2D3 \\ 0 & 0 & 1 & r4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Après les multiplications et l'application des règles d'addition matricielle, on obtient la matrice de transformation finale.

$${}^0T_4 = {}^0T_1 \times {}^1T_4 =$$

$$\begin{bmatrix} C1 & -S1 & 0 & 0 \\ S1 & C1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} C23 - S23 & -C2S3 - S2C3 & 0 & C2D3 + D2 \\ S2C3 + C2S3 & -S23 + C2C3 & 0 & S2D3 \\ 0 & 0 & 1 & r4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$${}^0T_4 = {}^0T_1 {}^1T_2 {}^2T_3 {}^3T_4 = \begin{bmatrix} sx & nx & ax & C2D3 + D2 \\ sx & ny & ay & S2D3 \\ sz & nz & az & r4 \\ 0 & 0 & 0 & 1 \end{bmatrix};$$

La transformation homogène de l'effecteur (E) dans le repère 4 est définie par la matrice suivante

Chapitre 01 : Modélisation du Robot SCARA

$${}^4T_E = I_4 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} ; \quad L_4 \text{ est La longueur de quatrième bras ; } L_4 = 0.$$

Donc ; ${}^0T_E = {}^0T_4$

Considérons U_0 tels que :

$$U_0 = \begin{bmatrix} sx & nx & ax & px \\ sx & ny & ay & py \\ sz & nz & az & pz \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

U_0 représente la position souhaitée de l'organe terminal dans le repère R_0 , et elle est définie par $U_0 = {}^0T_4$.

$$U_0 = {}^0T_4 = \begin{bmatrix} sx & nx & ax & D3C12 + C1D2 - S12D3 \\ sx & ny & ay & D3S1C2 + S1D2 + D3C1S2 \\ sz & nz & az & r4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.31)$$

L'analyse porte principalement sur la dernière colonne de la matrice, car elle renferme, à chaque étape, les équations indépendantes nécessaires à la résolution du modèle géométrique inverse.

$$\begin{cases} px = D3C12 + C1D2 - S12D3 = D3(C12 - S12) + C1D2 \\ py = D3S1C2 + S1D2 + D3C1S2 = D3(S1C2 + C1S2) + S1D2 \\ pz = r4 \end{cases} \quad (1.32)$$

Les équations (1.32.a) et (1.32.b), mentionnées précédemment, peuvent être réécrites de la manière suivante en utilisant :

$$\sin(\theta_1 + \theta_2) = \sin(\theta_1) * \cos(\theta_2) + \cos(\theta_1) * \sin(\theta_2) \quad (1.33)$$

$$\cos(\theta_1 + \theta_2) = \cos(\theta_1) * \cos(\theta_2) - \sin(\theta_1) * \sin(\theta_2) \quad (1.34)$$

$$\begin{cases} px = D3(C12) + C1D2 \\ py = D3(S12) + S1D2 \\ pz = r4 \end{cases} \quad (1.35)$$

1.12 Calcul de modèle géométrique inverse

L'obtention du modèle géométrique inverse repose sur l'approche de Paul, décrite selon les étapes suivantes :

$$\begin{bmatrix} px \\ py \\ pz \\ 1 \end{bmatrix} = {}^0T_1 {}^1T_2 {}^2T_3 {}^3T_4 \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad (1.38)$$

Il s'agit d'une structure découplable, ce qui permet de résoudre huit types d'équations. Les équations des types (1.32.a), (1.32.b) et (1.32.c) correspondent à celles déjà présentées dans le cadre de la méthode de Paul (voir tableau 1.2).

En élevant au carré puis en additionnant les expressions (1.32.a) et (1.32.b), on obtient l'expression suivante :

$$px^2 + py^2 = [D3(C12) + C1D2]^2 + [D3(S12) + S1D2]^2$$

Alors ;

$$px^2 + py^2 = D3^2 + D2^2 + 2D2D3C2$$

Cela donne :

$$C2 = \frac{(px^2 + py^2 - D3^2 - D2^2)}{(2D2D3)}, \quad \text{car : } C1C12 + S1S12 = C2$$

$$\text{d'où : } \theta_2 = \tan^{-1} \left[\frac{S2}{C2} \right], \quad \text{avec : } 1 + \tan^2 \theta_2 = \frac{1}{C1^2}$$

$$\theta_2 = \arctg[\pm \sqrt{(1 - \frac{C2^2}{C1^2})}] \quad (1.39)$$

$$\theta_1 = \arctg(S1, C1) \quad (1.40)$$

Pour θ_1 en utilise les expressions suivantes (1.32.a) et (1.32.b)

$$px = D3C12 + C1D2 - S12D3 \quad (1.41.a)$$

$$py = D3S1C2 + S1D2 + D3C1S2 \quad (1.41.b)$$

En prenant θ_1 comme facteur commun

Chapitre 01 : Modélisation du Robot SCARA

$$px = C1[D3C2 + D2] - S1[S2D3] \quad (1.42)$$

$$py = S1[D3C2 + D2] + C1[D3S2] \quad (1.43)$$

$$\text{Avec : } K1 = D3C2 + D2 \quad , \quad K2 = D3S2 \quad , \quad C2 = \frac{(px^2 + py^2 - D3^2 - D2^2)}{(2D2D3)}$$

$$px = C1K1 - S1K2 \quad (1.44)$$

$$py = S1K1 + C1K2 \quad (1.45)$$

Pour S1 en soustrayant les équations (1.32.a) et (1.32.b), on obtient l'expression suivante :

$$py - px = S1K1 + C1K2 - C1K1 + S1K2 \quad (1.46)$$

On effectue les produits $px \cdot K2$ et $py \cdot K1$:

$$K1py - K2px = S1K1K1 + S1K2K2 \quad (1.47)$$

$$S1 = \frac{K1py - K2px}{K1^2 + K2^2} \quad (1.48)$$

La valeur de θ_3 est donnée par :

$$T = \theta_1 + \theta_2 + \theta_3 = \arctg(S123, C123) \quad (1.49)$$

1.13 Calcul de modèle cinématique directe

En dérivant le modèle géométrique direct $X=f(q)$, il est possible d'obtenir la matrice jacobienne à partir de la relation suivante : $J = \frac{dx}{dq}$

$$\begin{cases} px = D3(C12) + C1D2 \\ py = D3(S12) + S1D2 \\ pz = r4 \\ \alpha = \theta_1 + \theta_2 + \theta_3 \end{cases} \rightarrow J = \begin{bmatrix} \frac{dpx}{dq1} & \frac{dpx}{dq2} & \frac{dpx}{dq3} & \frac{dpx}{dq4} \\ \frac{dpy}{dq1} & \frac{dpy}{dq2} & \frac{dpy}{dq3} & \frac{dpy}{dq4} \\ \frac{dpz}{dq1} & \frac{dpz}{dq2} & \frac{dpz}{dq3} & \frac{dpz}{dq4} \\ \frac{d\alpha}{dq1} & \frac{d\alpha}{dq2} & \frac{d\alpha}{dq3} & \frac{d\alpha}{dq4} \end{bmatrix} \quad (1.50)$$

La matrice jacobienne peut être déduite de manière directe, ce qui conduit à l'expression suivante :

$$\rightarrow J = \begin{bmatrix} -D2S1 - D3S12 & -D3S12 & 0 & 0 \\ D2C1 + D3C12 & D3C12 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix} \quad (1.51)$$

1.14 Calcul de Modèle cinématique inverse

Le modèle cinématique inverse s'obtient en inversant le modèle cinématique directe, ce qui implique la résolution d'un système d'équations linéaires.

Il peut alors être formulé de la manière suivante :

$$\dot{q} = J^{-1} \dot{X} \quad (1.52)$$

Dans le cas où la matrice J prend la forme suivante :

$$J = \begin{bmatrix} A & 0 \\ B & C \end{bmatrix} \quad \text{Avec: } A = \begin{bmatrix} -D2S1 - D3S12 & -D3S12 \\ D2C1 + D3C12 & D3C12 \end{bmatrix} \quad (1.53)$$

$$B = \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix} \quad (1.54)$$

$$C = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \quad (1.55)$$

$$J^{-1} = \begin{bmatrix} A^{-1} & 0 \\ -C^{-1}BA^{-1} & C^{-1} \end{bmatrix} \quad (1.56)$$

$$A^{-1} = \begin{bmatrix} \frac{D3C12}{\det} & \frac{D3S12}{\det} \\ \frac{-D2C1 - D3C12}{\det} & \frac{-D2S1 - D3S12}{\det} \end{bmatrix} = \begin{bmatrix} a1 & a2 \\ a3 & a4 \end{bmatrix} \quad (1.57)$$

D'où l'inverse J-1 :

$$J^{-1} = \begin{bmatrix} a1 & a2 & 0 & 0 \\ a3 & a4 & 0 & 0 \\ -a1 - a3 & -a2 - a4 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (1.58)$$

Lorsque le déterminant la plupart de temps =0 Matlab affiche erreur alors en utilisant la pseudo-inverse

$$J^+ = J' \times \text{inv}(J \times J' + \varepsilon \times I) \quad (1.59)$$

Avec :

$$J' = \begin{bmatrix} -D2S1 - D3S12 & D2C1 + D3C12 & 0 & 1 \\ -D3S12 & D3C12 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (1.60)$$

$$I = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1.61)$$

1.15 Modèle dynamique du robot SCARA

Le robot utilisé dans le cadre de cette étude est un manipulateur rigide de type SCARA comportant quatre degrés de liberté. Comme illustré à la Figure 1.12, ce bras robotisé est constitué de trois articulations rotatives, notées θ_1, θ_2 et θ_3 , ainsi que d'un axe de translation vertical, désigné par d_4 [14].

La dynamique de ce robot dans l'espace articulaire peut être modélisée à l'aide de la formulation d'Euler-Lagrange. Pour cela, les variables généralisées ont été choisies comme suit : les angles θ_1, θ_2 et θ_3 , correspondent respectivement aux coordonnées q_1, q_2 et q_3 tandis que le déplacement linéaire d_4 est représenté par q_4 . En adoptant cette représentation. [15]

L'équation dynamique globale du système peut être exprimée sous forme matricielle comme suit :

$$M(q)\ddot{q} + C(q, \dot{q})\dot{q} + \dot{g} = \tau \quad (1.62)$$

Donc :

$$\begin{bmatrix} M_{11} & M_{12} & M_{13} & M_{14} \\ M_{21} & M_{22} & M_{23} & M_{24} \\ M_{31} & M_{32} & M_{33} & M_{34} \\ M_{41} & M_{42} & M_{43} & M_{44} \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \\ \ddot{q}_3 \\ \ddot{q}_4 \end{bmatrix} + \begin{bmatrix} C_{11} & C_{12} & C_{13} & C_{14} \\ C_{21} & C_{22} & C_{23} & C_{24} \\ C_{31} & C_{32} & C_{33} & C_{34} \\ C_{41} & C_{42} & C_{43} & C_{44} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ m_4 g \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \\ \tau_4 \end{bmatrix} \quad (1.63)$$

Où :

$$\mathbf{I1} = I1 + I2 + I3 + I4 + m1 \times lc1^2 + m2 \times l1^2 + m2 \times (lc2^2 + 2 \times l1 \times lc2 \times \cos(q(2))) + (m3 + m4) \times (l1^2 + l2^2 + 2 \times l1 \times l2 \times \cos(q(2))). \quad (1.64)$$

$$\mathbf{m12} = I2 + I3 + I4 + m2 \times (lc2^2 + l1 \times lc2 \times \cos(q(2))) + (m3 + m4) \times (l2^2 + l1 \times l2 \times \cos(q(2))). \quad (1.65)$$

Chapitre 01 : Modélisation du Robot SCARA

$$m_{13}=I_3+I_4; \quad (1.66)$$

$$m_{14}=m_{24}=m_{34}=0; \quad (1.67)$$

$$m_{21}=m_{12}; \quad (1.68)$$

$$m_{22}=I_2+I_3+I_4+m_2 \times lc_{22}+(m_3+m_4) \times l_{22}; \quad (1.69)$$

$$m_{23}=m_{13}; \quad (1.70)$$

$$m_{31}=m_{13}; \quad (1.71)$$

$$m_{32}=m_{23}; \quad (1.72)$$

$$m_{33}=m_{13}; \quad (1.73)$$

$$m_{41}=m_{14}; \quad (1.74)$$

$$m_{42}=m_{24}; \quad (1.75)$$

$$m_{43}=m_{34}; \quad (1.76)$$

$$m_{44}=m_4; \quad (1.77)$$

$$c_{11} = -m_2 \times l_1 \times lc_2 \times \sin(q(2)) \times dq(2) - (m_3 + m_4) \times l_1 \times l_2 \times \sin(q(2)) \times dq(2) \quad (1.78)$$

$$c_{12} = -m_2 \times l_1 \times lc_2 \times \sin(q(2)) \times (dq(1) + dq(2)) - (m_3 + m_4) \times l_1 \times l_2 \times \sin(q(2)) \times (dq(1) + dq(2)) \quad (1.79)$$

$$c_{21} = m_2 \times l_1 \times lc_2 \times \sin(q(2)) \times dq(1) + (m_3 + m_4) \times l_1 \times l_2 \times \sin(q(2)) \times dq(1) \quad (1.80)$$

$$c_{13} = c_{14} = c_{22} = c_{23} = c_{24} = c_{31} = c_{32} = c_{33} = c_{34} = 0 \quad (1.81)$$

$$c_{41} = c_{42} = c_{43} = c_{44} = 0 \quad (1.82)$$

1.16 Espace de travail d'un robot

L'espace de travail d'un robot manipulateur correspond à l'ensemble des positions et éventuellement des orientations que l'organe terminal (ou effecteur) peut atteindre dans l'espace. Il délimite la zone dans laquelle le robot peut opérer et dépend de sa conception mécanique ainsi que de ses paramètres de configuration [11]

Chapitre 01 : Modélisation du Robot SCARA

1.16.1 Principaux facteurs influençant l'espace de travail

- **Les limites des articulations** : Chaque articulation possède une plage de mouvement déterminée qui restreint la mobilité globale.
- **Les longueurs des segments** : La taille des bras ou des éléments du robot conditionne sa portée.
- **Le type d'articulations** : Qu'elles soient rotatives (articulées), prismatiques (linéaires), ou hybrides, elles modifient la forme de l'espace accessible.
- **La position et l'orientation de la base** : Le placement du robot dans son environnement influence la zone dans laquelle il peut évoluer.

q1 : angle de l'articulation de base (*revolut joint* 338°).

q2 : angle du premier bras (*revolut joint* $+160^\circ$ à -160°).

q3 : angle du deuxième bras (*revolut joint* 360°).

q4 : déplacement vertical de l'axe Z (*prismatic joint* « $+0.1m$ à $-0.1m$ »)

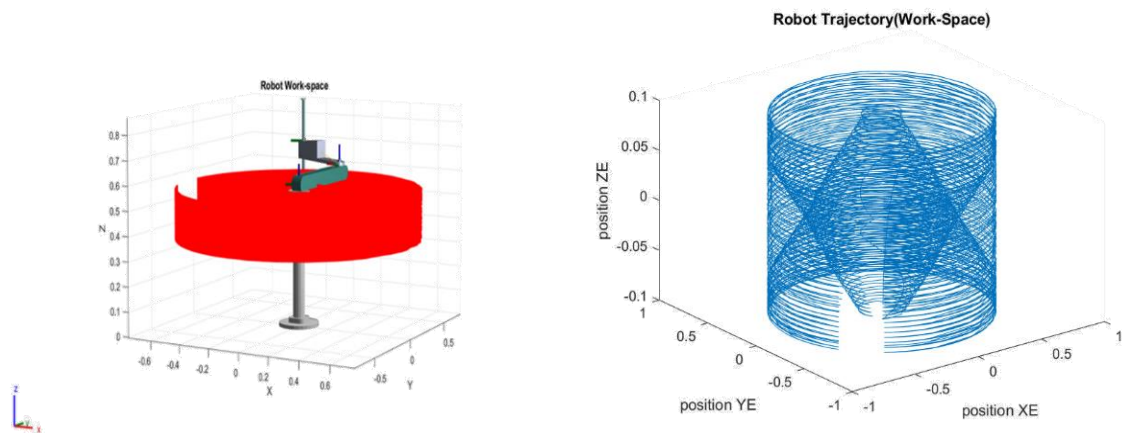


Figure 1.13 Espace du travail du robot

1.17 Conclusion

Ce premier chapitre a permis de poser les bases essentielles liées à la structure SCARA, un type de robot largement utilisé dans les applications industrielles en raison de sa précision, de sa rapidité

Chapitre 01 : Modélisation du Robot SCARA

et de sa configuration mécanique adaptée aux tâches de manipulation sur plan horizontal. Nous avons retracé brièvement l'évolution des robots jusqu'à l'émergence des architectures modernes, en mettant l'accent sur les spécificités géométriques de la structure SCARA.

Sur le plan technique, nous avons étudié la modélisation géométrique directe (*MGD*) de ce robot à l'aide de la convention de *Denavit-Hartenberg*, enrichie par les ajustements proposés par *Khalil* et *Kleinfinger*. Cette approche nous a permis d'exprimer, via des matrices de transformation homogène, la position et l'orientation de l'effecteur final par rapport à la base. La résolution de l'inverse géométrique (*MGI*) a été réalisée à l'aide de la méthode de *Paul*, outil efficace et couramment utilisé dans les systèmes robotiques industriels.

L'étude de cette architecture constitue une base solide pour les développements ultérieurs.

Chapitre 2 : Commande du Robot *SCARA*

2 Chapitre 2 : Commande du Robot SCARA

2.1 Introduction

Ce chapitre porte sur la commande d'un robot *SCARA* à quatre degrés de liberté à l'aide d'un régulateur *PID*, choisi pour sa simplicité et son efficacité en robotique industrielle. L'objectif est d'assurer un asservissement précis en position et en vitesse, garantissant stabilité, rapidité et qualité de suivi des trajectoires. La régulation en position vise à atteindre précisément les consignes angulaires, tandis que la régulation en vitesse, intégrée en boucle interne, assure des transitions fluides.

S'appuyant sur le modèle dynamique établi précédemment, la stratégie applique un *PID* sur chaque articulation. Les performances sont évaluées via des simulations numériques, analysant des critères tels que dépassement, erreur, stabilité et temps de réponse.

Le chapitre présente d'abord les principes de la régulation et du *PID*, puis détaille la commande en position, la commande en vitesse, et la génération de trajectoires par interpolation polynomiale dans les espaces articulaire et opérationnel. Enfin, les résultats de simulation sont commentés pour valider l'approche [16].

2.2 Commande des mouvements articulaires

Le contrôle des articulations est crucial pour la précision des robots manipulateurs, notamment pour les systèmes multi-axes comme le robot *SCARA*. Chaque articulation doit être pilotée selon des consignes de position et de vitesse, en intégrant les contraintes mécaniques et dynamiques du système.

Le *SCARA* à 4 degrés de liberté combine différents types de mouvements : rotations des bras horizontaux, translation verticale et rotation de l'effecteur. Cette diversité exige une stratégie de commande à la fois coordonnée et adaptée à chaque articulation.

Deux approches principales existent pour la commande des axes :

Boucle ouverte : Simple mais peu robuste aux perturbations (frottements, variations de charge).

Boucle fermée : Basée sur un retour capteur, elle permet un ajustement continu pour une meilleure précision, essentielle dans les applications industrielles.

L'enjeu consiste à concevoir un système de régulation fiable, assurant le suivi de trajectoire tout en limitant les erreurs dynamiques et en préservant la stabilité. Les contrôleurs *PID*, souvent employés pour leur robustesse et leur simplicité, seront analysés en détail dans la suite, en lien avec le modèle dynamique du robot [16].

2.3 Généralités sur la régulation

2.3.1 Définition de la régulation

La régulation industrielle vise à maintenir une grandeur physique (température, pression, vitesse, etc.) à une valeur de consigne malgré les perturbations. Elle utilise des systèmes de commande en boucle fermée, ajustant en temps réel les variables d'entrée en fonction des mesures de sortie pour garantir stabilité et performance.

Au-delà de l'industrie, ce principe s'applique aussi à la gestion des flux (transports, logistique), où l'objectif est de corriger les écarts (retards, congestions) pour optimiser le fonctionnement global [17].

2.3.2 Objectifs dans le contexte du robot SCARA

La régulation est essentielle pour garantir la précision et la stabilité des mouvements du robot SCARA. Ses principaux objectifs sont :

- **Précision :** Minimiser les écarts entre les consignes (position/vitesse) et les mouvements réels.
- **Stabilité :** Compenser les perturbations (frottements, variations de charge) et les incertitudes du modèle.
- **Performance :** Optimiser le temps de réponse, l'amortissement et éviter les dépassements.
- **Répétabilité :** Assurer des mouvements fiables pour des tâches industrielles (assemblage, dépôt).

- **Rétroaction efficace** : Exploiter les données des capteurs (encodeurs, etc.) pour une correction en temps réel [16].

2.4 Régulation

Pour la commande des robots industriels, plusieurs types de régulateurs peuvent être utilisés, tels que les régulateurs P (Proportionnel), PI (Proportionnel-Intégral), PD (Proportionnel-Dérivé), PID (Proportionnel-Intégral-Dérivé), ainsi que des techniques plus avancées comme la commande adaptative, la commande prédictive ou la commande floue. Dans ce projet, nous avons choisi d'utiliser un régulateur de type PID [16].

2.5 Régulation PID : principes fondamentaux

Le régulateur *PID* (Proportionnel-Intégral-Dérivé) est l'un des outils de contrôle les plus utilisés dans les systèmes d'asservissement en raison de sa simplicité, de sa robustesse, et de son efficacité dans un large éventail d'applications industrielles, y compris la robotique [16].

2.5.1 Formulation mathématique et structure du *PID*

Le régulateur *PID* (Proportionnel-Intégral-Dérivé) agit à partir de l'erreur entre la consigne (t) et la sortie mesurée $y(t)$, selon la loi de commande suivante :

$$u(t) = K_P e(t) + K_I \int_0^t e(\tau) d\tau + K_D \frac{de(t)}{dt} \quad (2.1)$$

Où $e(t) = r(t) - y(t)$ est l'erreur instantanée.

Les trois termes (P , I , D) introduisent respectivement une correction immédiate, une compensation du décalage permanent, et une anticipation des variations rapides de l'erreur [18].

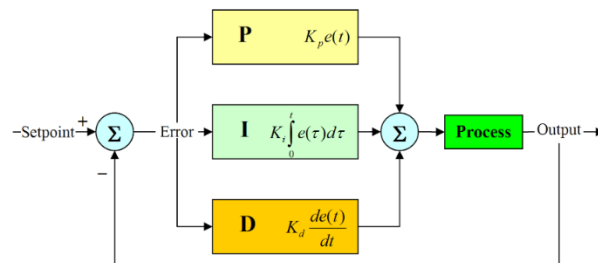


Figure 2.1 Schéma de principe de régulateur *PID* [18].

2.5.2 Interprétation des composantes P , I et D

2.5.2.1 Terme proportionnel (P)

Ce terme applique une correction proportionnelle à l'erreur actuelle. Plus l'erreur est grande, plus la commande est forte. Il influence directement la rapidité de réponse mais peut laisser une erreur résiduelle.

$$U_P(t) = K_P e(t) \quad (2.2)$$

2.5.2.2 Terme dérivé (D)

Cette action élimine l'erreur statique mais peut introduire des effets indésirables comme l'*overshoot* ou une réponse lente.

$$U_I = K_I \int_0^t e(\tau) d\tau \quad (2.3)$$

2.5.2.3 Terme intégral (I)

L'action dérivée prédit la tendance future de l'erreur en fonction de son taux de variation. Elle vise à réduire les oscillations et améliorer la stabilité du système. L'action dérivée est donnée par :

$$U_D(t) = K_D \frac{de(t)}{dt} \quad (2.4)$$

Cette composante réagit à la variation de l'erreur, en ralentissant les changements trop brusques. Elle améliore la stabilité du système mais peut amplifier les signaux parasites [19].

2.5.2.4 Commande PID complète

Le régulateur PID combine les trois actions, ce qui donne la loi de commande complète :

$$U(t) = U_P(t) + U_I(t) + U_D(t) \quad (2.5)$$

C'est-à-dire que la commande totale est une somme de l'action proportionnelle, de l'action intégrale et de l'action dérivée :

$$U(t) = K_P e(t) + K_I \int_0^t e(\tau) d\tau + K_D \frac{de(t)}{dt} \quad (2.6)$$

2.5.3 Atouts et limitations de la commande *PID*

Le *PID* est populaire en raison de sa simplicité d'implémentation et de sa robustesse dans de nombreux contextes. Il fonctionne sans qu'un modèle précis du système soit nécessaire [20].

2.5.3.1 Avantages :

- Facilité d'implémentation et d'analyse.
- Bonne efficacité pour des systèmes linéaires simples.
- Réglage intuitif dans de nombreux cas industriels.

2.5.3.2 Limites :

- Réglage empirique parfois difficile.
- Moins performant pour des systèmes fortement non linéaires.
- Sensibilité au bruit pour la partie dérivée si non filtrée.
- Application du *PID* aux manipulateurs robotiques.

Dans les robots industriels comme le SCARA, chaque articulation est généralement régulée individuellement à l'aide d'un *PID*, intégré dans une architecture en boucle fermée. Ce régulateur peut être appliqué à différents niveaux : position, vitesse ou couple, en fonction de la structure de commande utilisée.

Pour chaque axe, les paramètres *PID* doivent être ajustés en fonction de sa dynamique propre (inertie, frottements, charges variables), afin d'assurer un mouvement fluide et précis [16].

2.6 COMMANDE EN POSITION

2.6.1 Régulation de la position articulaire

La régulation de la position articulaire est une composante essentielle du contrôle des robots manipulateurs, notamment les robots SCARA, où il est crucial de suivre des consignes de position avec précision. Cela garantit que chaque articulation atteigne la position cible, permettant un mouvement coordonné et exact de l'effecteur [16].

2.6.2 Architecture du contrôle en position

Le contrôle en position repose sur une architecture à boucle fermée, où l'erreur entre la position souhaitée et la position actuelle de chaque articulation est constamment mesurée. Cette erreur dynamique est utilisée pour ajuster en temps réel les commandes envoyées aux moteurs ou aux actionneurs. En général, le régulateur *PID* est utilisé pour minimiser cette erreur en ajustant la vitesse et l'accélération des articulations jusqu'à ce que la consigne soit atteinte [21].

2.6.3 Schéma de la commande *PID* en position

Le système de contrôle repose sur plusieurs blocs fonctionnels. La position de consigne de l'effecteur dans l'espace cartésien est définie par \mathbf{X}_{ref} . Un comparateur calcule l'erreur $= \mathbf{X}_{ref} - \mathbf{X}$, qui est ensuite convertie en coordonnées articulaires q via la Modélisation Géométrique Inverse (*MGI*). Un régulateur *PID*, composé des gains K_P , K_i et K_d , ajuste la commande en agissant respectivement sur l'erreur instantanée, son intégrale (pour éliminer l'erreur statique) et sa dérivée (pour améliorer la stabilité et la rapidité). Les contributions P , I et D sont sommées pour former le couple de commande Γ , appliqué au robot. La Modélisation Géométrique Directe (*MGD*) détermine la position réelle X de l'effecteur à partir des angles articulaires q . Une boucle de rétroaction permet de corriger en permanence l'écart entre la consigne et la position réelle, assurant ainsi un contrôle précis du mouvement.

La figure ci-dessous présente le schéma fonctionnel de ce système de commande.

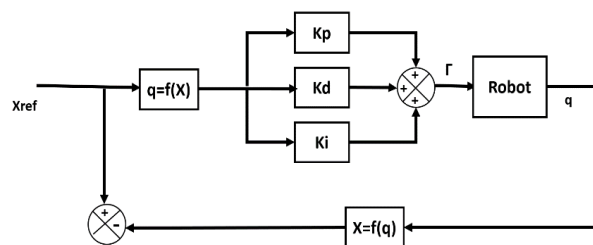


Figure 2.2 Schéma de la commande *PID* position

2.6.4 Sélection des capteurs de position (encodeurs)

La précision du contrôle en position dépend largement des capteurs de retour, comme les encodeurs rotatifs, qui mesurent l'angle exact de chaque articulation. Ces capteurs fournissent un signal précis qui est comparé à la consigne de position. Le choix du capteur est crucial : des

encodeurs à haute résolution permettent une mesure plus précise et donc un contrôle plus fin, tandis que des capteurs moins performants peuvent induire des erreurs et limiter la précision du système [16].

2.6.5 Réglage des paramètres *PID*

Les paramètres *PID* (proportionnel, intégral et dérivé) doivent être soigneusement réglés pour chaque articulation du robot. Le gain proportionnel *KP* détermine la réponse immédiate au signal d'erreur, tandis que le gain intégral *KI* est utilisé pour éliminer l'erreur résiduelle à long terme. Enfin, le gain dérivé *KD* permet de réduire les oscillations et de stabiliser la réponse du système. Le réglage optimal de ces paramètres peut être effectué de manière empirique ou à l'aide de méthodes formelles comme l'approche de *Ziegler-Nichols* [20].

2.6.6 Simulation et évaluation des résultats (référence en θ)

Les simulations permettent de tester les performances de la régulation de position avant de l'appliquer sur le robot réel. En simulant les mouvements des articulations et en observant l'évolution de l'erreur, on peut évaluer l'efficacité du régulateur *PID* et ajuster les paramètres en fonction des performances souhaitées. Les résultats de simulation, tels que les graphiques d'erreur de position en fonction du temps, montrent l'impact des réglages sur la stabilité, la rapidité de réponse et la précision de la régulation.

2.7 COMMANDE EN VITESSE

2.7.1 Régulation de la vitesse articulaire

La régulation *PID* contrôle précisément la vitesse des articulations du robot SCARA en corrigeant en temps réel les écarts entre la consigne et la mesure. Elle combine réactivité, précision et stabilité grâce à ses trois actions (*P*, *I*, *D*), tout en compensant les perturbations comme les variations de charge. Essentielle en robotique industrielle, cette technique garantit des mouvements rapides, fluides et reproductibles pour les opérations automatisées [20].

2.7.2 Importance de la commande en vitesse (asservissement moteur)

L'asservissement en vitesse est fondamental pour contrôler précisément les moteurs des robots. Il maintient la vitesse de rotation à la consigne malgré les perturbations, garantissant des mouvements fluides et synchronisés entre les articulations.

Chapitre 02 : Commande du Robot SCARA

Dans les robots multiarticulés, cette régulation améliore la stabilité globale, la précision du suivi de trajectoire et la réponse dynamique.

Intégrée dans une architecture de contrôle en cascade (position-vitesse-couple), elle renforce la robustesse du système en limitant les erreurs transitoires [16].

2.7.3 Schéma de la commande *PID* en vitesse

Le schéma de commande intègre une boucle de vitesse complémentaire à la régulation de position. La trajectoire définit une vitesse cartésienne de référence $d\mathbf{X}_{ref}$ convertie en consignes articulaires $d\mathbf{q}$ par la jacobienne inverse $d\mathbf{q} = \mathbf{J}^{-1}(\mathbf{q})d\mathbf{X}$. Ces vitesses de rotation sont comparées aux mesures des codeurs avant d'être retransformées en vitesse opérationnelle $d\mathbf{X}$ via la jacobienne directe $d\mathbf{X} = \mathbf{J}(\mathbf{q})d\mathbf{q}$.

L'action dérivée du correcteur exploite cet écart de vitesse pour améliorer la réponse dynamique. Son rôle principal consiste à amortir les oscillations mécaniques lors des mouvements rapides. La coordination entre ces deux boucles (position et vitesse) permet d'obtenir un suivi de trajectoire précis tout en garantissant la stabilité du système.

Le schéma est présenté dans la figure ci déçue :

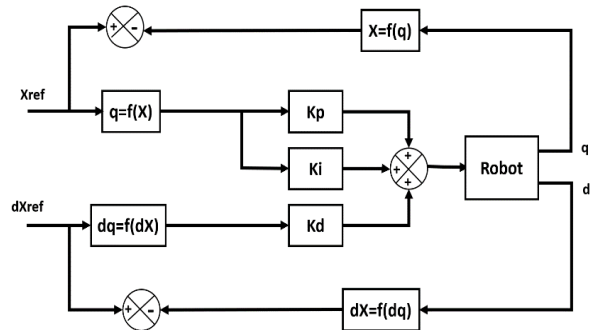


Figure 2.3 Schéma de la commande *PID* vitesse

2.7.4 Simulation et analyse des vitesses articulaires

Des simulations MATLAB/Simulink ont été réalisées pour évaluer les performances de la commande en vitesse du robot SCARA. Les résultats montrent que le régulateur *PID* permet un suivi précis et rapide des consignes de vitesse (trapézoïdales ou sinusoïdales), sans dépassement excessif, garantissant une réponse dynamique optimale lors du suivi de trajectoire.

Les résultats obtenus démontrent que :

- Les vitesses des articulations suivent fidèlement les références imposées.
- Le comportement dynamique reste stable et sans oscillations marquées.
- L'erreur de suivi reste très faible, confirmant l'efficacité de la commande.

Les courbes issues des simulations montrent clairement que les vitesses articulaires $\dot{q}_1(t)$, $\dot{q}_2(t)$, $\dot{q}_3(t)$, $\dot{q}_4(t)$ évoluent de manière cohérente avec les consignes, ce qui valide les choix faits lors de la conception du régulateur [16].

2.8 GÉNÉRATION DE TRAJECTOIRES

2.8.1 Planification de trajectoires

La planification de trajectoire constitue une étape primordiale dans la commande des robots manipulateurs. Elle vise à déterminer une suite de positions ou de configurations que le robot doit suivre pour passer d'un point initial à un point final, tout en respectant des contraintes spécifiques telles que la continuité, la limitation des vitesses ou des accélérations, et la faisabilité mécanique du mouvement [11].

2.8.2 Trajectoire en espace articulaire

La trajectoire est définie directement dans l'espace des articulations, c'est-à-dire en fonction des angles des moteurs ou des positions des articulations. Chaque articulation suit une courbe distincte, souvent déterminée par une méthode d'interpolation mathématique, telle que l'interpolation linéaire ou polynomiale.

Cette méthode est couramment choisie pour sa simplicité d'implémentation et sa compatibilité avec les régulateurs classiques, comme les contrôleurs *PID*. Toutefois, un inconvénient majeur réside dans le fait que la trajectoire de l'effecteur dans l'espace de travail peut être difficile à prévoir ou à visualiser, car elle dépend fortement de la cinématique propre au robot [16].[21].

2.8.3 Trajectoire en espace cartésien

La planification en espace cartésien consiste à établir la trajectoire de l'extrémité du robot directement dans son espace de travail, à l'aide de coordonnées spatiales telles que X , Y et Z . Cette

approche est particulièrement adaptée aux tâches nécessitant une grande précision du mouvement, comme le traçage de formes, les opérations d'assemblage ou encore le suivi de contours définis.

Pour appliquer ce type de planification à un robot, il est nécessaire de passer par la cinématique inverse, qui permet de traduire les positions cartésiennes souhaitées en positions articulaires correspondantes. Cette étape peut s'avérer complexe, notamment en présence de singularités ou de configurations multiples pour une même position cible, ce qui demande une attention particulière lors de l'implémentation [22].[23].

2.9 Méthode d'interpolation

Les modes d'interpolation polynomiale les plus couramment utilisés sont l'interpolation linéaire ainsi que l'interpolation à l'aide de polynômes de degré trois ou cinq.

2.9.1 Polynôme de degré 3

L'interpolation basée sur un polynôme de degré trois permet de générer des trajectoires lisses reliant deux configurations articulaires d'un robot. Cette technique est largement utilisée car elle garantit une évolution continue de la position et de la vitesse, tout en restant simple à mettre en œuvre [11].

La trajectoire d'une articulation est décrite par une fonction polynomiale :

$$q(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 \quad (2.7)$$

Les coefficients du polynôme sont calculés de manière à satisfaire ces conditions :

$$\begin{cases} a_0 = q^i \\ a_1 = 0 \\ a_2 = \frac{3}{t_f^2} D \\ a_3 = -\frac{2}{t_f^3} D \end{cases} \quad (2.8)$$

L'expression peut également être représentée selon d'autres formulations équivalentes :

$$r(t) = 3\left(\frac{t}{t_f}\right)^2 + 2\left(\frac{t}{t_f}\right)^3 \quad (2.9)$$

La figure 2.4 illustre l'évolution de la position, de la vitesse et de l'accélération pour l'axe j . Cette loi de mouvement garantit la continuité des vitesses, bien qu'elle ne permette pas celle des

Chapitre 02 : Commande du Robot SCARA

accélérations. En pratique, la rigidité des structures robotiques industrielles permet à la discontinuité d'accélération d'être atténuée mécaniquement. Ainsi, l'emploi d'un polynôme de degré trois reste une solution efficace et suffisante pour la majorité des applications.

Pour une articulation donnée j , la vitesse atteint son maximum lorsque $t = \frac{t_f}{2}$, avec une valeur correspondante de $|\dot{q}_{j \max}| = \frac{3|D_j|}{t_f}$, avec $|D_j| = |q_i^f - q_i^i|$ (2.10)

L'accélération atteint son maximum aux instants $t = 0$ et $t = t_f$, avec une valeur donnée par :

$$|\ddot{q}_{j \max}| = \frac{6|D_j|}{t_f^2} \quad (2.11)$$

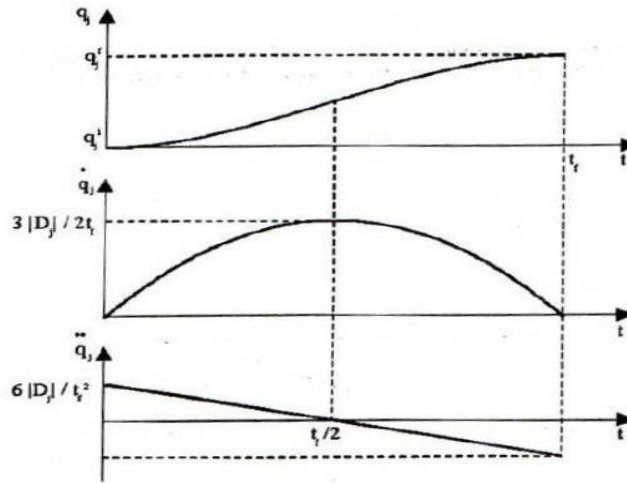


Figure 2.4 Loi polynomiale de de degré trois.[11]

2.10 Loi Bang-Bang

Le mouvement comporte une première phase où l'accélération reste constante jusqu'à $t_f/2$, suivie d'une phase de décélération également constante, comme illustré à la figure 2.5. Les vitesses initiale et finale étant nulles, la trajectoire obtenue est continue en position et en vitesse, mais présente une rupture au niveau de l'accélération [11].

La position s'exprime selon la relation suivante :

$$\begin{cases} q(t) = q^i + 2\left(\frac{t}{t_f}\right)^2 D & \text{pour } 0 \leq t \leq \frac{t_f}{2} \end{cases} \quad (2.11)$$

$$\begin{cases} q(t) = q^i + \left[-1 + 4\left(\frac{t}{t_f}\right) - 2\left(\frac{t}{t_f}\right)^2\right] D & \text{pour } \frac{t_f}{2} \leq t \leq t_f \end{cases} \quad (2.12)$$

Pour une articulation donnée j , les expressions de la vitesse maximale et de l'accélération maximale sont les suivantes :

$$|\dot{q}_{j \max}| = \frac{2|D_j|}{t_f} \quad (2.13)$$

$$|\ddot{q}_{j \max}| = \frac{4|D_j|}{t_f^2} \quad (2.14)$$

La détermination du temps minimal s'effectue ici de manière similaire à celle utilisée pour le cas polynomial :

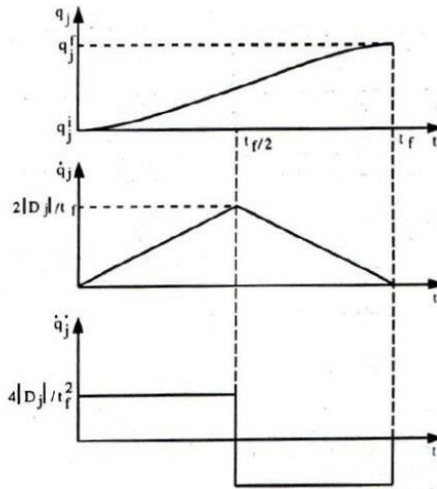


Figure 2.5 Loi de Bang-Bang [11].

2.11 Simulation complète de la commande *PID*

Cette section expose les résultats obtenus lors des simulations du robot *SCARA* équipé d'un régulateur *PID* appliqué à chaque articulation rotative. Les coefficients du *PID* ont été soigneusement réglés afin d'améliorer la dynamique du système et de réduire les erreurs de poursuite. Les performances du robot ont été examinées à partir de deux types de références appliquées aux variables de commande, avec une analyse portant à la fois sur les trajectoires visées et les écarts observés.

Chapitre 02 : Commande du Robot SCARA

Dans le cadre de la simulation, deux profils de référence ont été définis pour les variables de commande q_1, q_2, q_3 et q_4

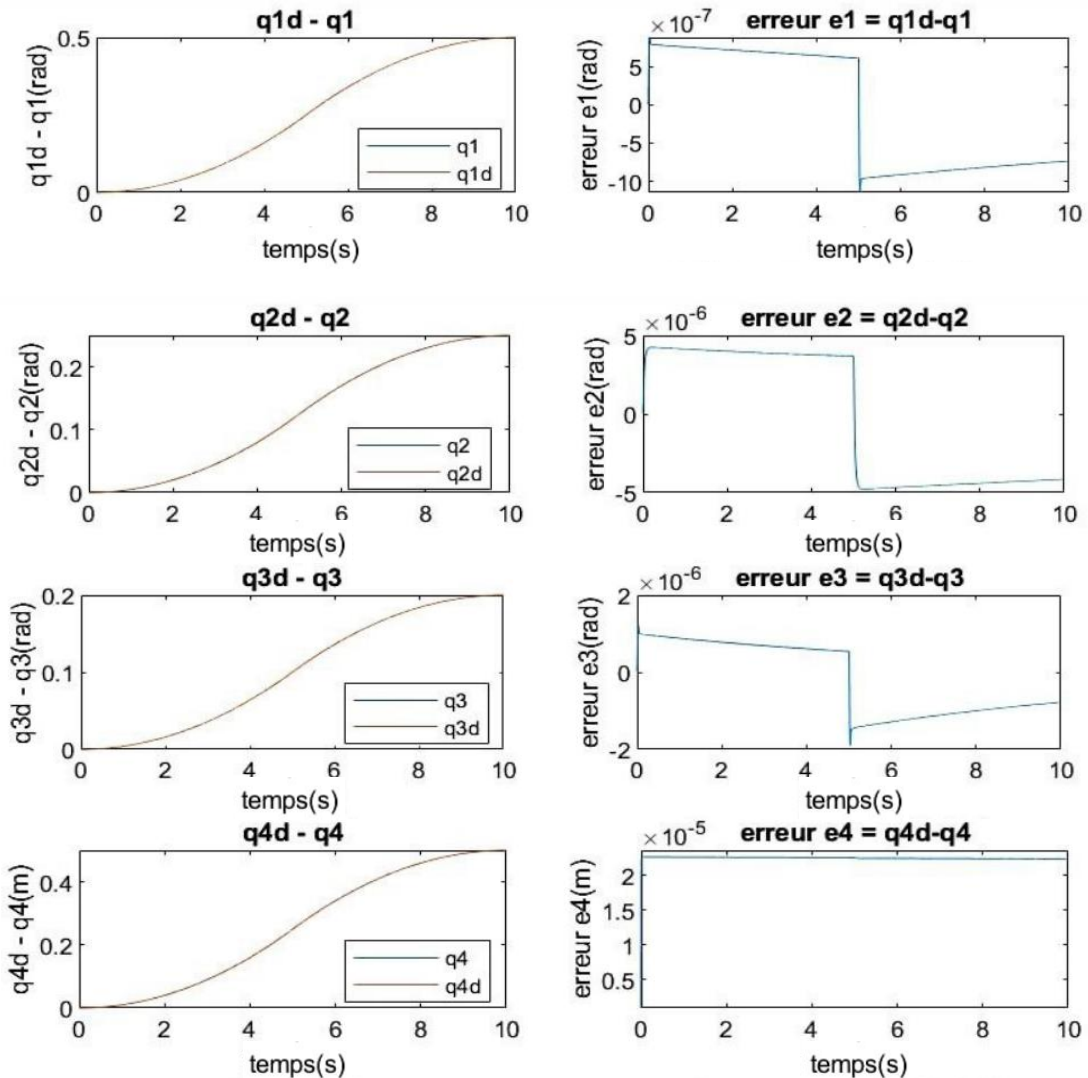
2.11.1 Simulation des trajectoires « Q_{ref}, dQ_{ref} »

2.11.2 Référence 1 (Loi *Bang-Bang*)

- Simulation des trajectoires Q_{ref}

$$\begin{cases} q(t) = q^i + 2 \left(\frac{t}{t_f} \right)^2 D & \text{pour } 0 \leq t \leq \frac{t_f}{2} \end{cases} \quad (2.15)$$

$$\begin{cases} q(t) = q^i + \left[-1 + 4 \left(\frac{t}{t_f} \right) - 2 \left(\frac{t}{t_f} \right)^2 \right] D & \text{pour } \frac{t_f}{2} \leq t \leq t_f \end{cases} \quad (2.16)$$



- Simulation des trajectoires dQ_{ref}

$$\begin{cases} q(t) = q^i + 4 \left(\frac{t}{t_f} \right) D & \text{pour } 0 \leq t \leq \frac{t_f}{2} \end{cases} \quad (2.17)$$

$$\begin{cases} q(t) = q^i + \left[4 \left(\frac{1}{t_f} \right) - 4 \left(\frac{t}{t_f} \right) \right] D & \text{pour } \frac{t_f}{2} \leq t \leq t_f \end{cases} \quad (2.18)$$

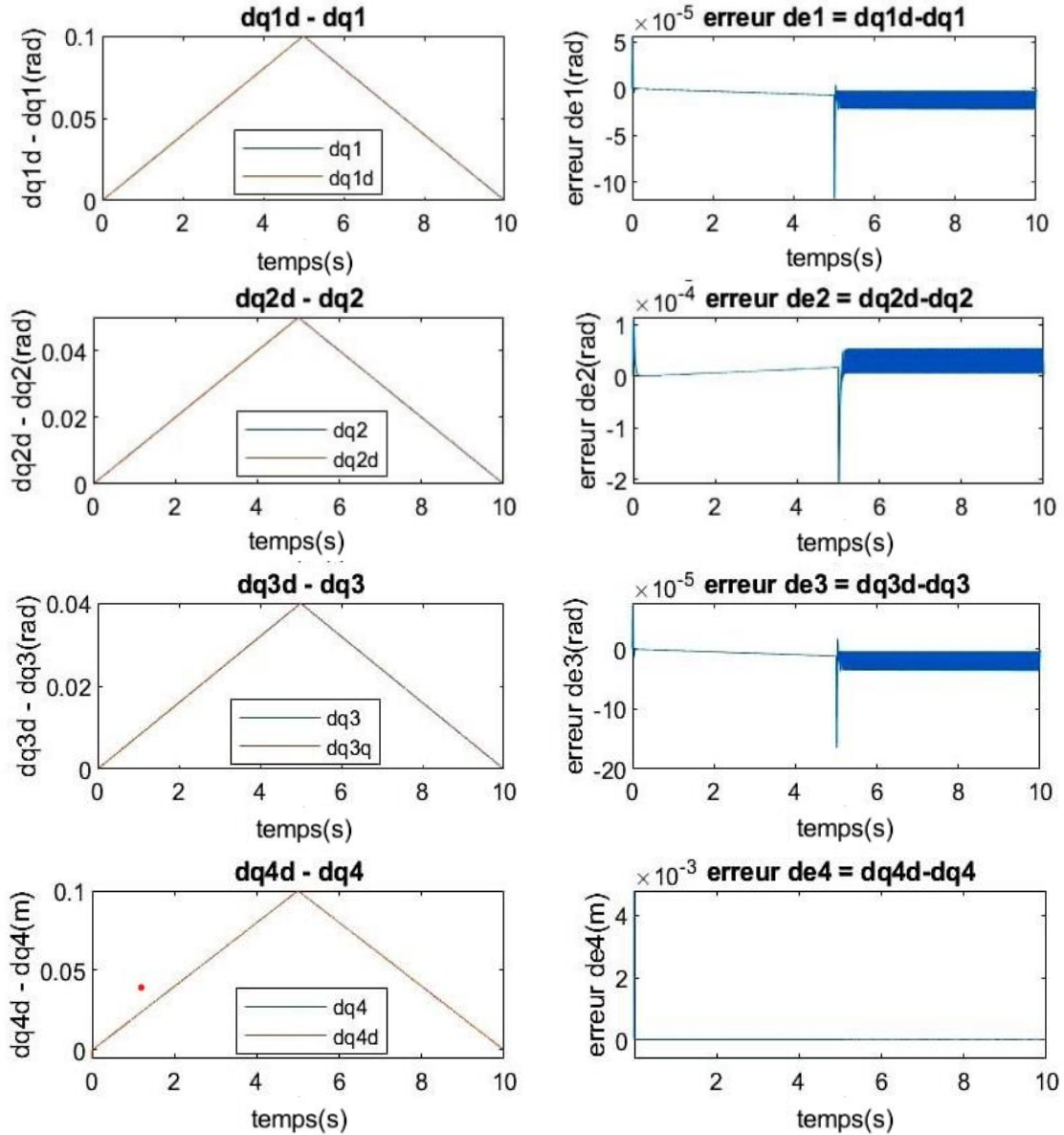


Figure 2.7 Pursuite de vitesse avec erreur des 4 articulations référence 1

Chapitre 02 : Commande du Robot SCARA

La figure 2.6 et 2.7 illustre les performances du système de commande en position et en vitesse des articulations du robot *SCARA* sur une durée de 10 secondes. Les courbes obtenues montrent un bon suivi des consignes, avec des écarts faibles et bien amortis. Ces résultats traduisent une réponse dynamique stable et une bonne précision du régulateur, capable d'assurer un comportement conforme aux trajectoires de référence.

2.11.3 Référence 1 (Polynôme de degré 3)

- Simulation des trajectoires Q_{ref}

$$r(t) = 3\left(\frac{t}{t_f}\right)^2 - 2\left(\frac{t}{t_f}\right)^3 \quad (2.19)$$

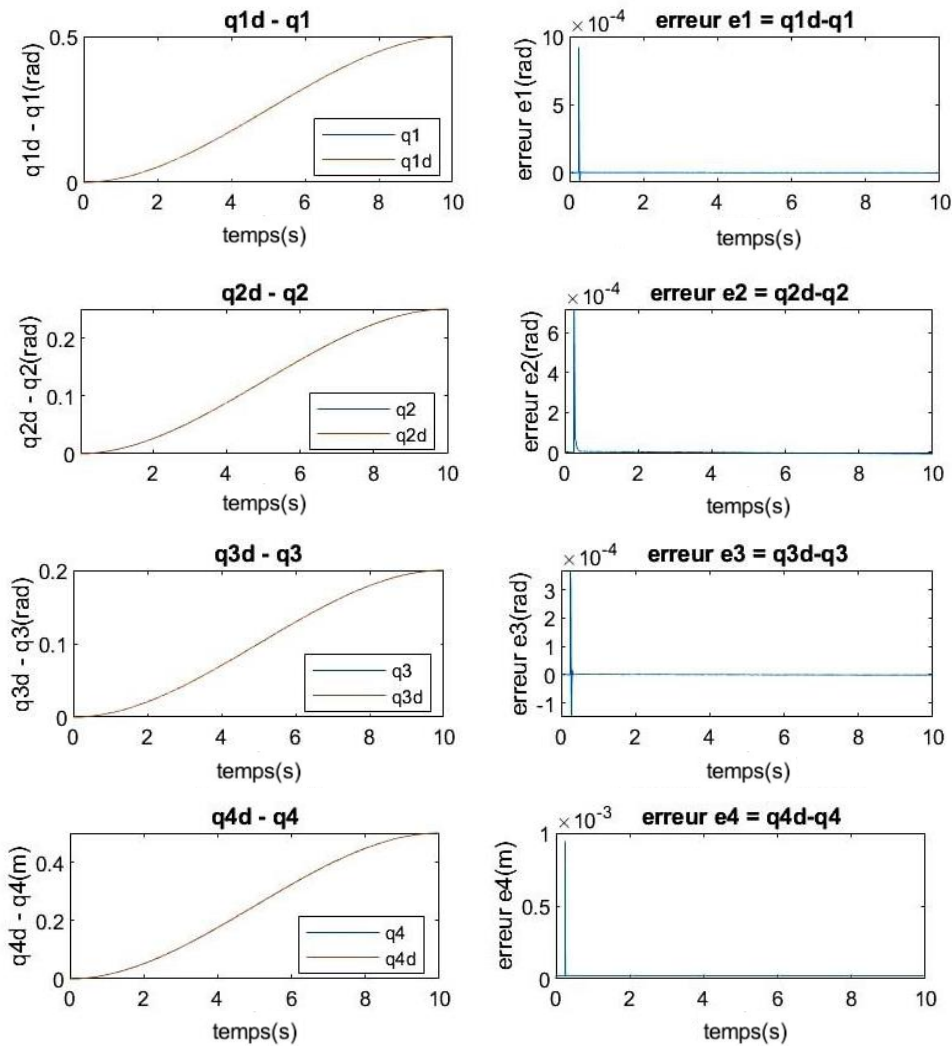


Figure 2.8 Poursuite de position avec erreur des 4 articulations référence 2

- Simulation des trajectoires Q_{ref}

$$r(t) = 6 \left(\frac{t}{t_f} \right) - 6 \left(\frac{t^2}{t_f^2} \right) \quad (2.20)$$

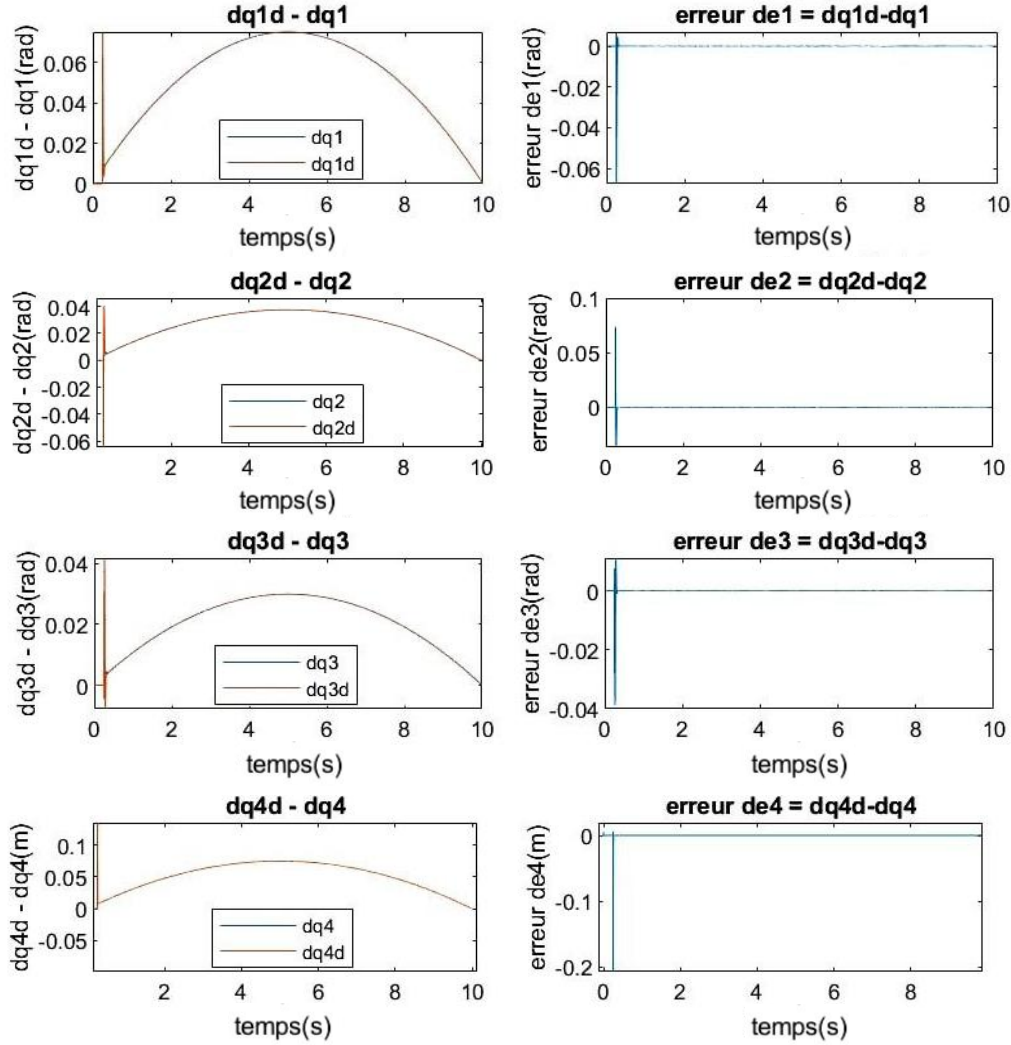


Figure 2.9 Poursuite de vitesse avec erreur des 4 articulations référence 1

Les résultats présentés dans les figures 2.9 et 2.10 mettent en évidence l'efficacité du système de commande dans le suivi des positions articulaires désirées q et des vitesses articulaires de référence \dot{q} . Les écarts observés, notés $e_q = q - q_d$ pour la position et $e_{\dot{q}} = \dot{q} - \dot{q}_d$ pour la vitesse, restent globalement faibles, avec des oscillations limitées. Cela témoigne de la performance du régulateur, probablement de type *PID*, à maintenir les articulations proches des consignes imposées tout en assurant une réponse dynamique stable.

2.11.4 Simulation des Trajectoires « X_{ref}, dX_{ref} »

2.11.4.1 Référence 1

- Simulation des trajectoires X_{ref}

$$r = 0.4 \text{ m}, \omega = 0.35 \text{ rad/s}, Az = -0.15 \text{ m}, \lambda = -0.5$$

$$\begin{cases} XE = r * \cos(\omega t) & (2.21) \\ YE = r * \sin(\omega t) & (2.22) \\ ZE = Az * e^{\lambda t} & (2.23) \end{cases}$$

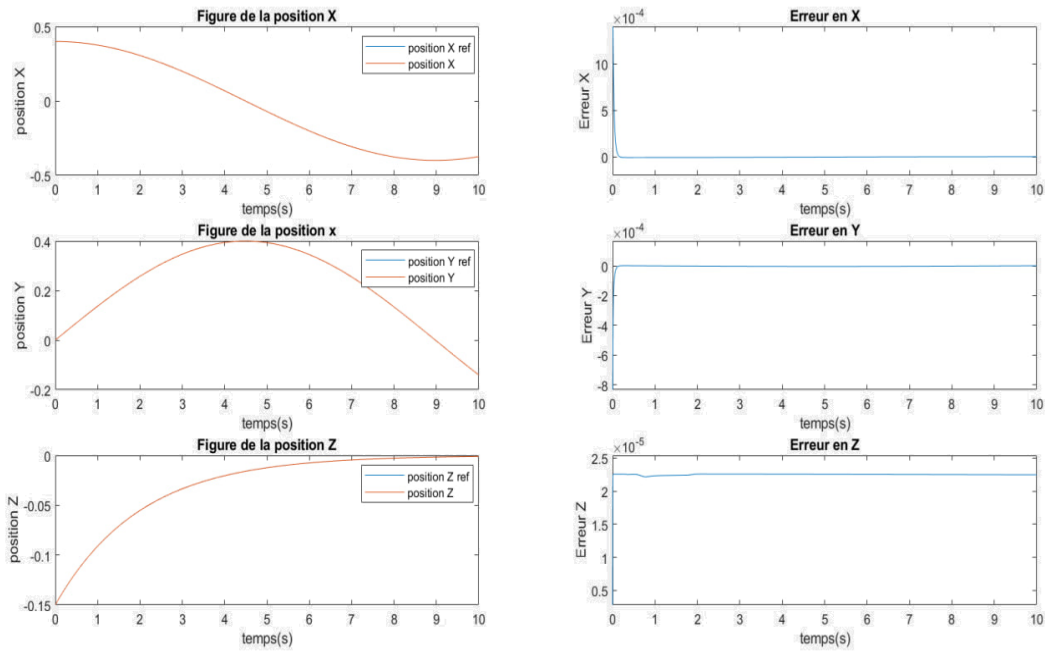


Figure 2.10 Suivi de Trajectoire Cartésienne et Erreurs pour les position $XE, YE, ZE, ref1$

La trajectoire du robot *SCARA* est suivie avec une grande précision sur les axes X, Y et Z, affichant des marges d'erreur minimales. Le régulateur *PID* corrige de manière optimale les décalages initiaux, notamment sur les axes X et Y, tandis que les variations sur l'axe Z demeurent insignifiantes, confirmant la fiabilité du système de contrôle. Cette réactivité dans la correction des écarts met en avant les performances robustes et l'efficacité du contrôleur *PID* utilisé.

Chapitre 02 : Commande du Robot SCARA

- Simulation des trajectoires dX_{ref}

$$r = 0.4 \text{ m}, \omega = 0.35 \text{ rad/s}, Az = -0.15 \text{ m}, \lambda = -0.5$$

$$\begin{cases} dXE = -r\omega * \sin(\omega t) \\ dYE = r\omega * \cos(\omega t) \\ dZE = Az\lambda * e^{\lambda t} \end{cases} \quad (2.24)$$

$$\quad (2.25)$$

$$\quad (2.26)$$

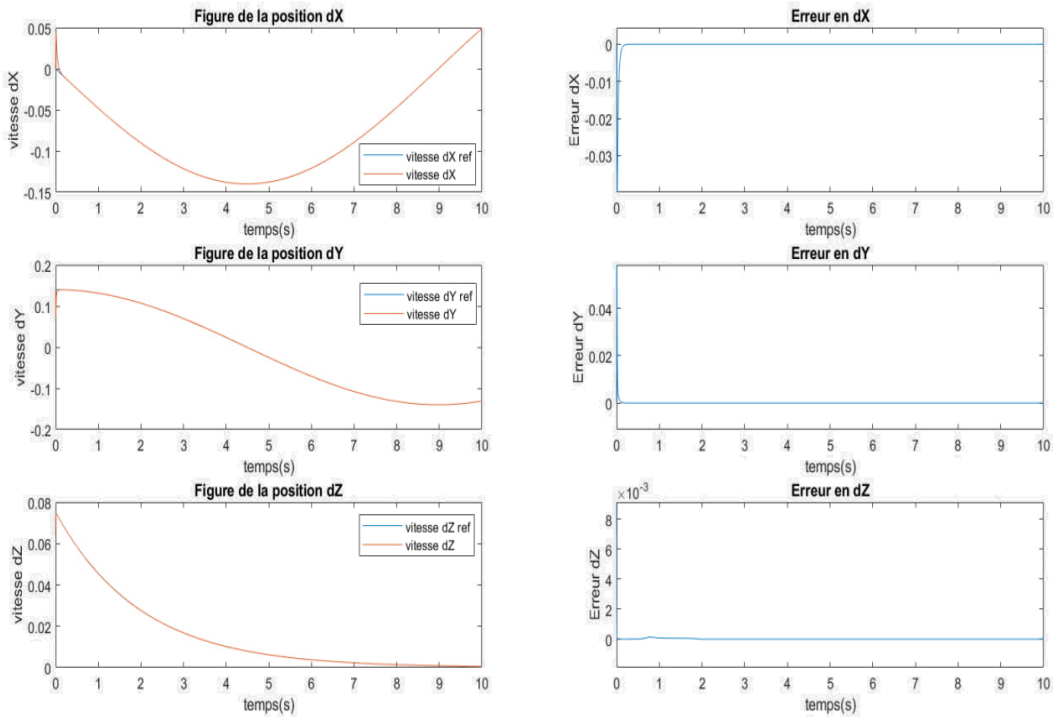


Figure 2.11 Suivi de Trajectoire Cartésienne et Erreurs pour les vitesses dXE, dYE, dZE, $ref1$

Le contrôle des vitesses cartésiennes $\dot{X}, \dot{Y}, \dot{Z}$, est également bien maîtrisé par le système de commande. Les vitesses mesurées adhèrent fidèlement aux consignes, avec des écarts dynamiques limités et une absence d'instabilités notables. Cela démontre que le régulateur *PID* assure non seulement une précision en position, mais aussi une dynamique de mouvement fluide, un atout crucial pour les applications nécessitant un suivi de trajectoire continu.

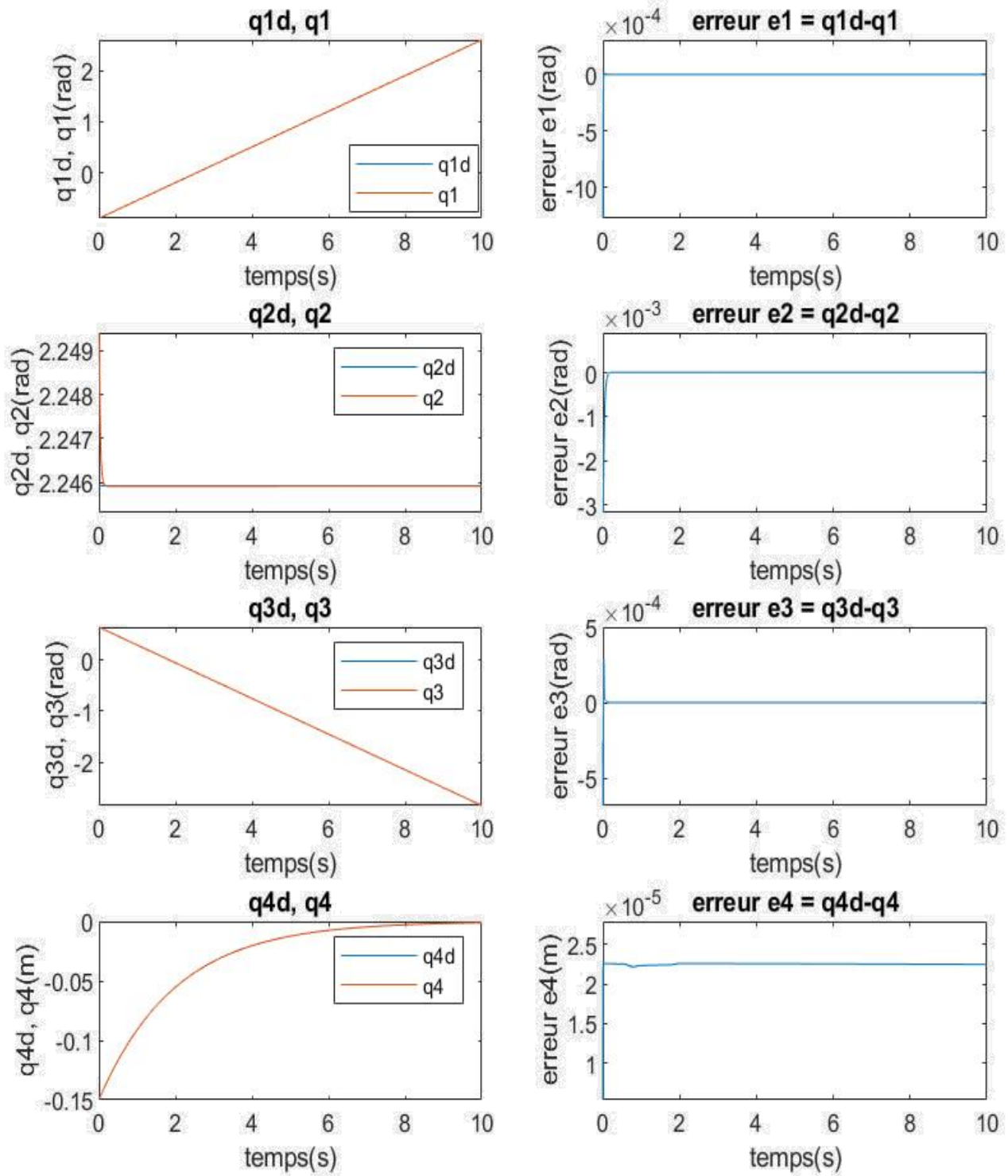


Figure 2.12 Poursuite de position avec erreur des 4 articulations

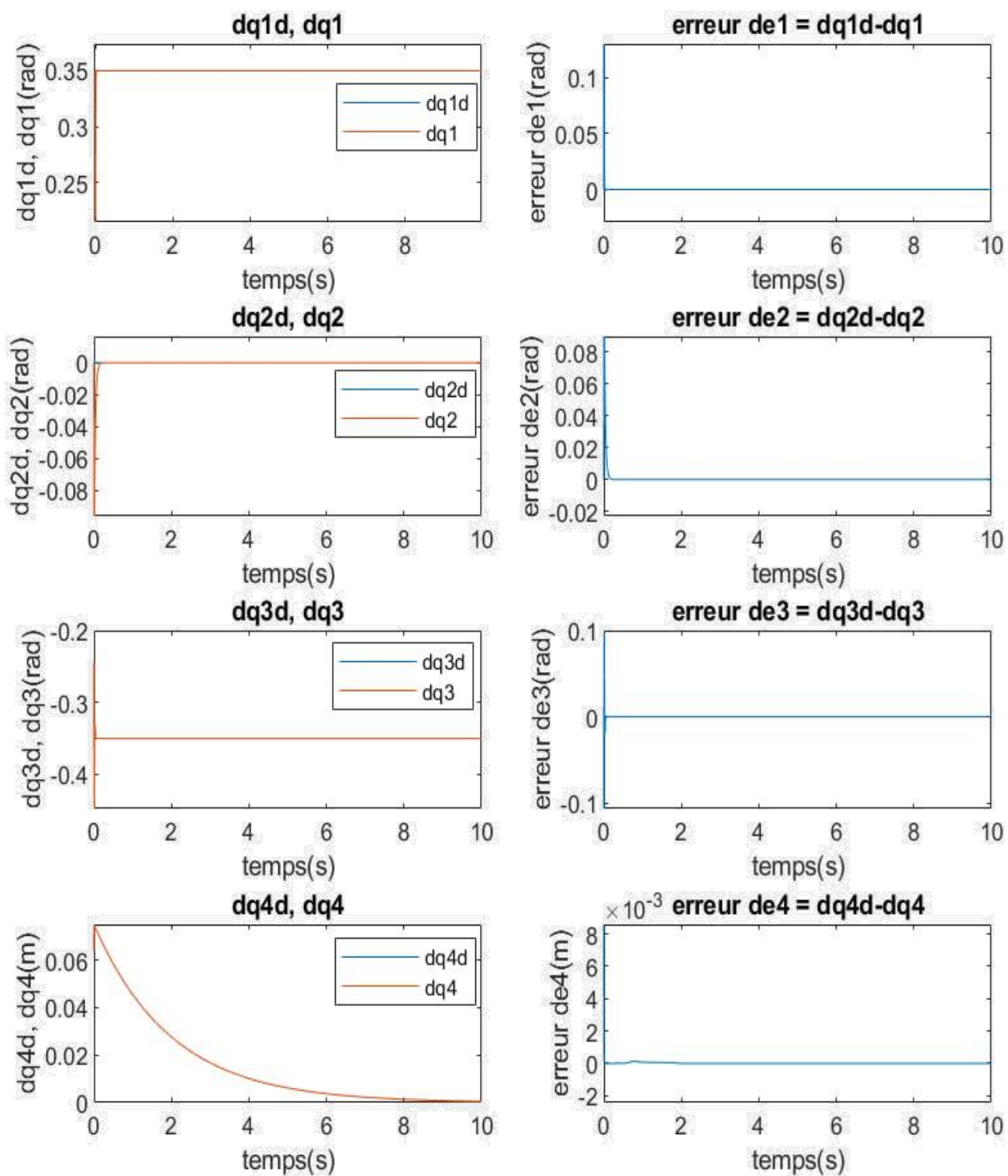


Figure 2.13 Poursuite de vitesse avec error des 4 articulations

Chapitre 02 : Commande du Robot SCARA

Les écarts observés demeurent minimes et tendent rapidement vers zéro, confirmant l'efficacité du correcteur *PID* dans la compensation des déviations. Cette convergence rapide témoigne d'une dynamique de contrôle optimale et d'un amortissement efficace des oscillations.

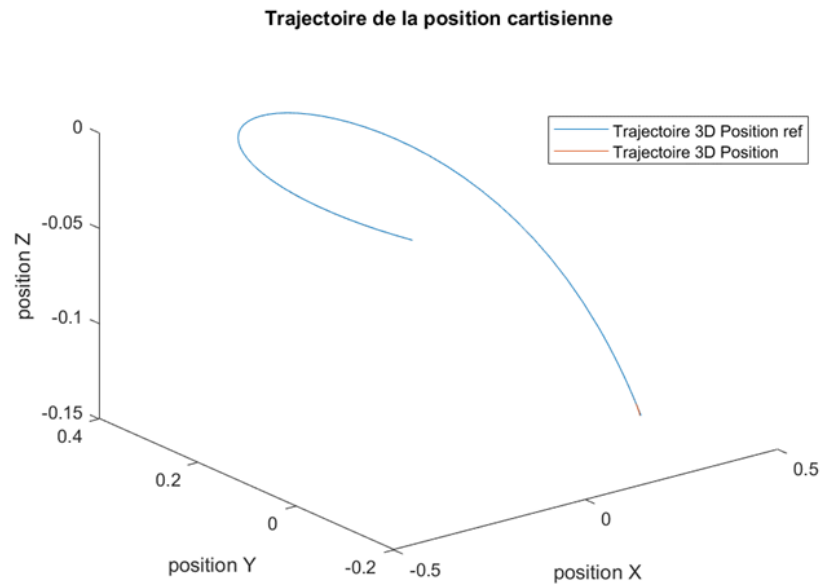


Figure 2.15 Trajectoire 3D des Positions Cartésiennes pour un Robot SCARA à 4DDL

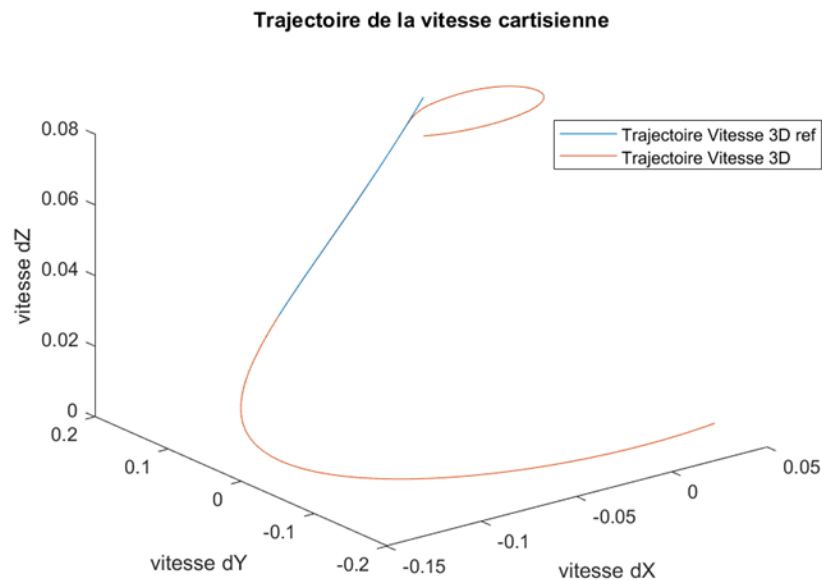


Figure 2.14 Trajectoire 3D des Positions Cartésiennes pour un Robot SCARA à 4 DDL

Chapitre 02 : Commande du Robot SCARA

Le système assure un suivi précis des consignes de position $(XE), Y(YE), Z(ZE)$ par rapport aux références $X(XE_{ref}), Y(YE_{ref}), Z(ZE_{ref})$, avec seulement quelques légères oscillations rapidement stabilisées par l'action du régulateur. Par ailleurs, les vitesses cartésiennes $(\dot{X}E_{ref}, \dot{Y}E_{ref}, \dot{Z}E_{ref})$ adhèrent fidèlement à leurs profils de référence $\dot{X}E, \dot{Y}E, \dot{Z}E$, ne présentant que de faibles déviations transitoires. Ces résultats mettent en évidence les excellentes performances du système de commande, capable d'assurer un suivi dynamique précis et réactif tant en position qu'en vitesse.

2.11.4.2 Référence 2

- **Simulation des trajectoires X_{ref}**

$$a = 0,15m, b = 0,15m, \omega = 0,8rad/s,$$

$$x0=0,3, y0=0, Z_{base}=-0,05m, Az=0,05m, \omega z=0,2rad/s, \theta r=\pi/6$$

Angle temporel pour le trèfle

$$\theta(t) = \omega t$$

Forme du trèfle à 3 pétales (coordonnées brutes avant rotation)

$$r(t) = -r\omega * \sin(3\theta) \tag{2.27}$$

$$X_{bt} = a_{rt} * \cos\theta + x0 \tag{2.28}$$

$$Y_{bt} = b_{rt} * \sin\theta + y0 \tag{2.29}$$

$$Z(t) = -Z_{base} + Az(-\sin(\omega z t)) \tag{2.30}$$

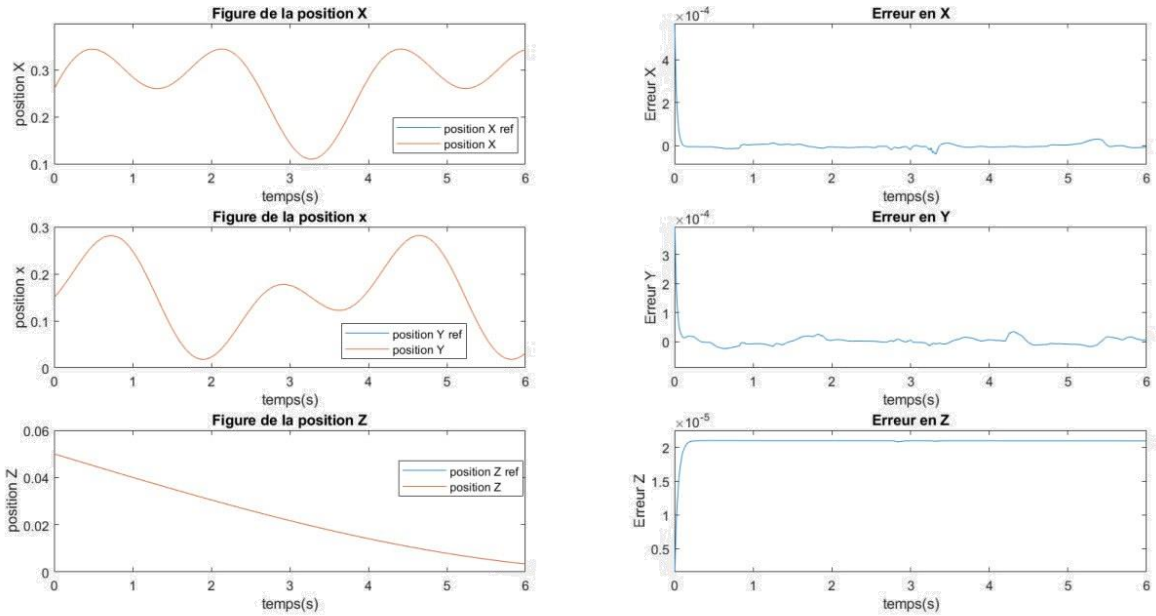


Figure 2.16 Suivi de Trajectoire Cartésienne et Erreurs pour les position XE, YE, ZE

La performance globale du suivi des positions du robot *SCARA* dans les axes X, Y et Z se révèle excellente, avec des erreurs très faibles. Le contrôleur *PID* parvient à compenser efficacement les écarts initiaux, en particulier sur les axes X et Y, tandis que les erreurs sur l'axe Z restent négligeables, traduisant une grande précision de contrôle. Cette capacité à corriger rapidement les déviations souligne la robustesse et l'efficacité du régulateur *PID* mis en œuvre.

- **Simulation des trajectoires dX_{ref}**

$$a = 0,15m, b = 0,15m, \omega = 0,8rad/s,$$

$$x0=0,3, y0=0, Z_{base}=-0,05m, Az=0,05m, \omega z=0,2rad/s, \theta r=\pi/6$$

Angle temporel pour le trèfle

$$\theta(t) = \omega t$$

Forme du trèfle à 3 pétales (coordonnées brutes avant rotation)

Chapitre 02 : Commande du Robot SCARA

$$r(t) = \sin(3\theta) \rightarrow \frac{dr}{dt} 3\omega \cos(3\theta) \quad (2.31)$$

$$dX_{bt} = a\left(\frac{dr}{dt} \cos(\theta) - r(t)\sin(\theta)\omega\right) \quad (2.32)$$

$$dY_{bt} = b\left(\frac{dr}{dt} \sin(\theta) + r(t)\cos(\theta)\omega\right) \quad (2.33)$$

$$dZ(t) = -Az\omega z * \cos(\omega z t) \quad (2.34)$$

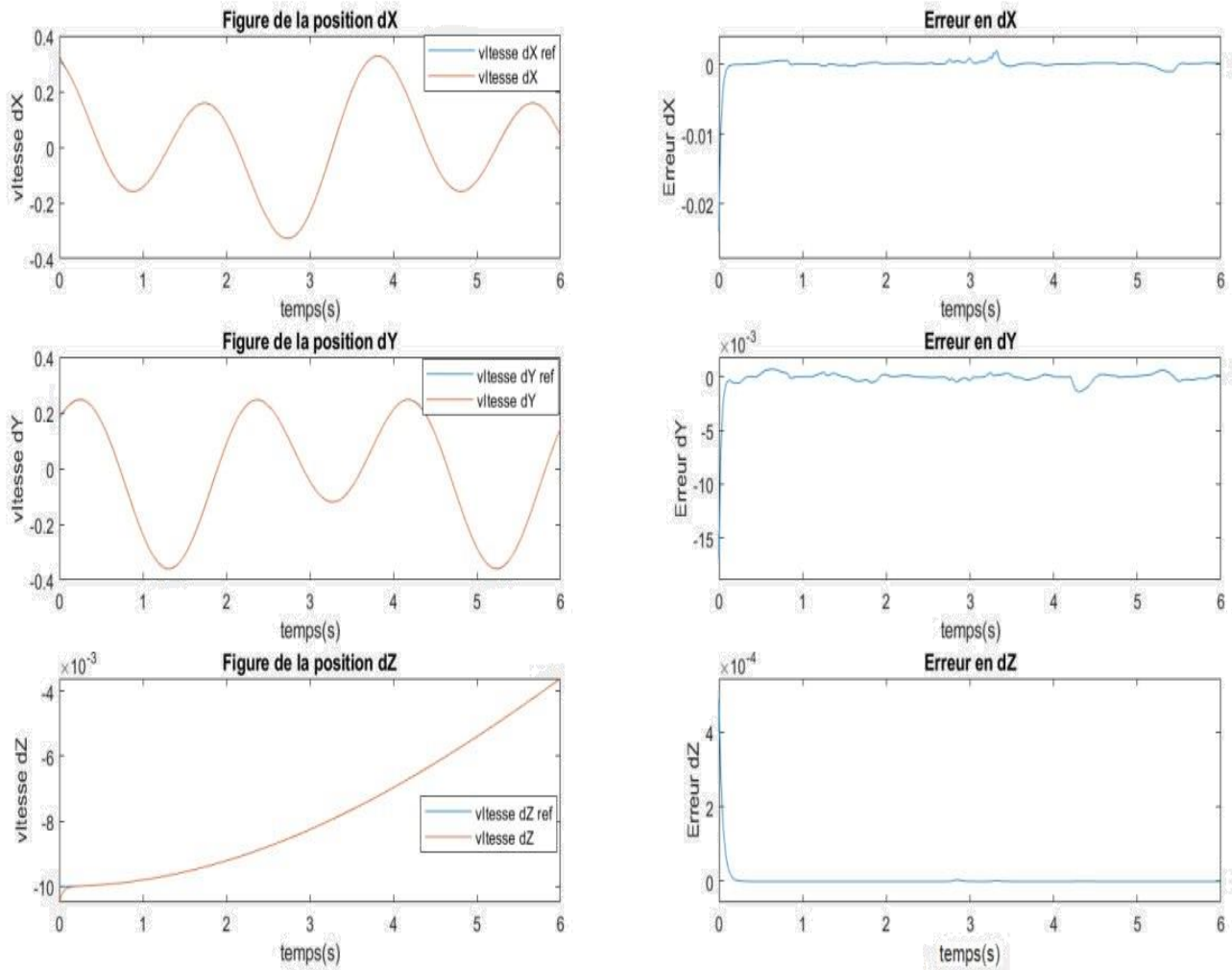


Figure 2.17 Suivi de Trajectoire Cartésienne et Erreurs pour les vitesses dXE, dYE, dZE

Chapitre 02 : Commande du Robot SCARA

Le suivi des vitesses cartésiennes $\dot{X}, \dot{Y}, \dot{Z}$, le système de commande assure également une réponse satisfaisante. Les vitesses suivent de près les profils de référence, avec des erreurs dynamiques modérées et sans instabilités majeures. Cela montre que le régulateur *PID* est capable non seulement d'assurer un bon positionnement, mais aussi de garantir une transition fluide en vitesse, ce qui est essentiel pour les applications de trajectoire continue.

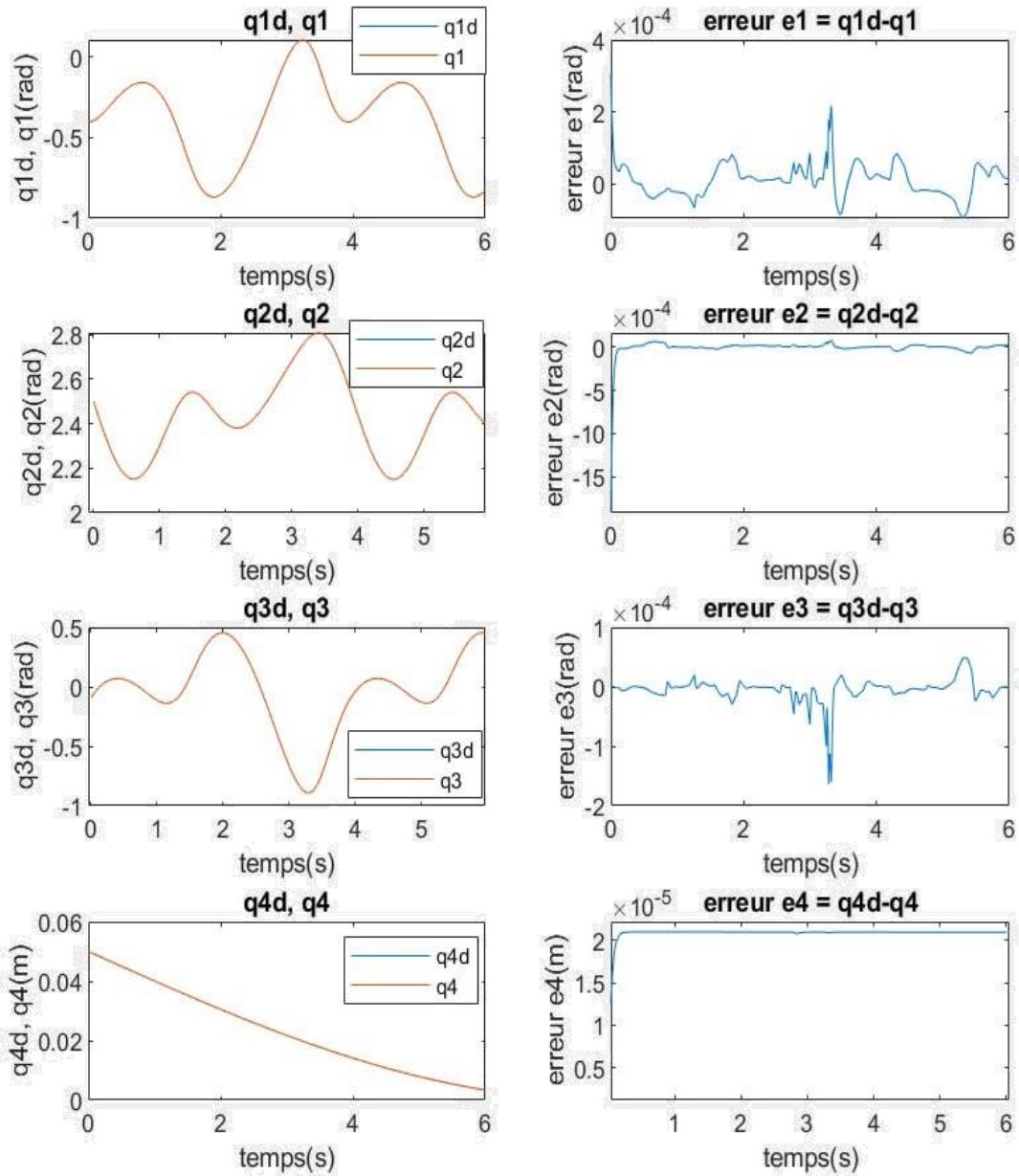


Figure 2.18 Poursuite de position avec erreur des 4 articulations

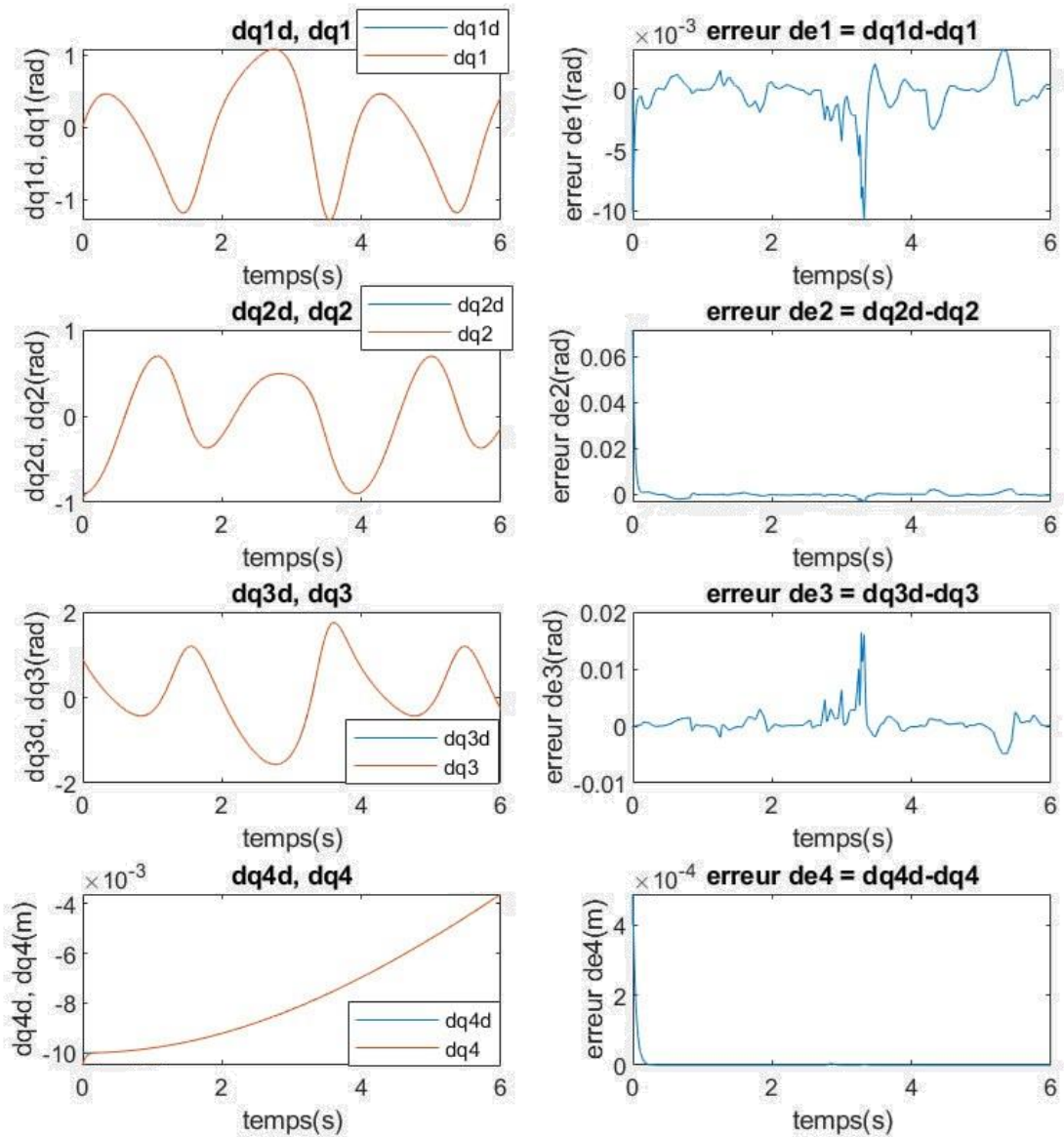


Figure 2.19 Poursuite de vitesse avec error des 4 articulations

Les erreurs restent faibles et convergent rapidement vers zéro, illustrant les bonnes performances du régulateur *PID* en termes de correction des écarts. Cette stabilité rapide reflète une bonne dynamique de régulation et une réponse bien amortie.

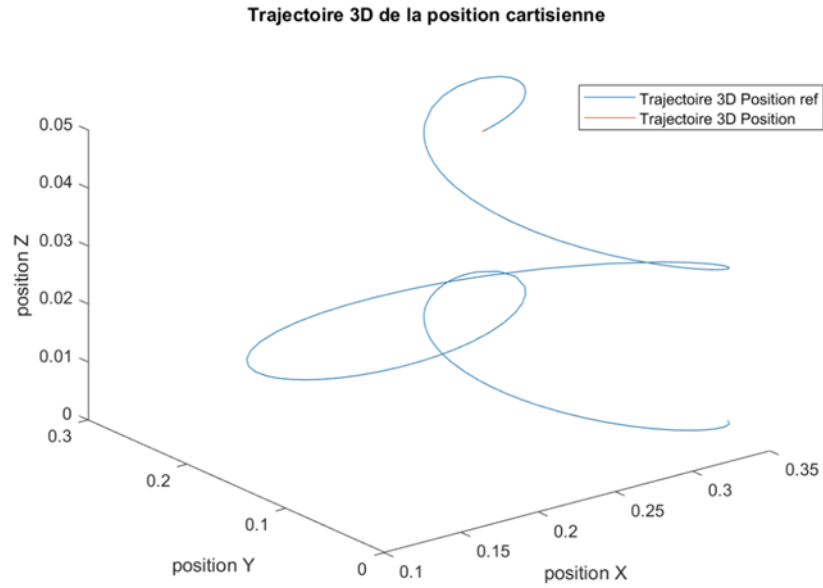


Figure 2.20 Trajectoire 3D des Positions Cartésiennes pour un Robot SCARA à 4DDL

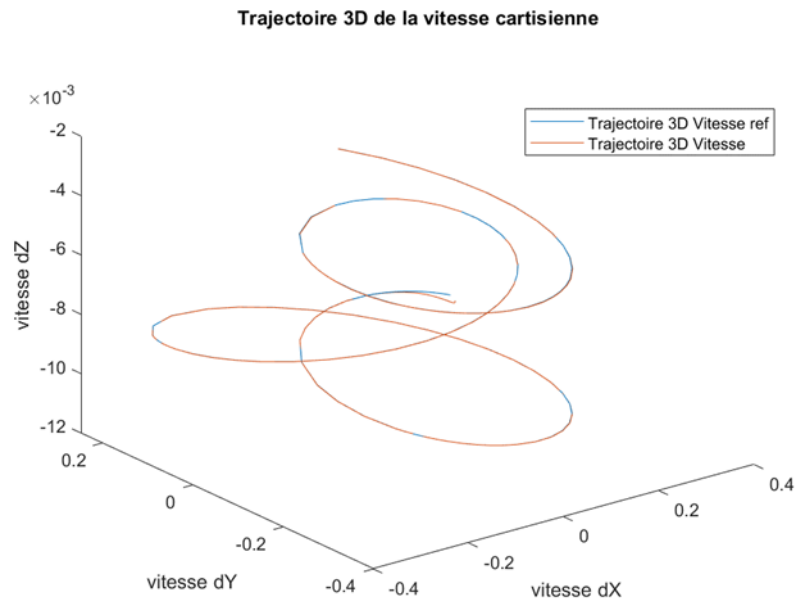


Figure 2.21 Trajectoire 3D des Positions Cartésiennes pour un Robot SCARA à 4DDL

La trajectoire de position dans les axes $X(XE), Y(YE), Z(ZE)$ suit de manière précise les références imposées, $X(XE_{ref}), Y(YE_{ref}), Z(ZE_{ref})$, malgré quelques oscillations mineures rapidement corrigées par le régulateur. De même, les trajectoires de vitesse $\dot{X}E, \dot{Y}E, \dot{Z}E$ suivent étroitement les profils de référence correspondants $(\dot{X}E_{ref}, \dot{Y}E_{ref}, \dot{Z}E_{ref})$, avec des écarts transitoires faibles, ce qui confirme la capacité du système de commande à assurer un suivi fluide et réactif aussi bien en position qu'en vitesse.

2.12 Conclusion

Ce chapitre a démontré l'efficacité d'une architecture de commande *PID* à double boucle (position et vitesse) pour le contrôle du robot *SCARA* à quatre degrés de liberté. La combinaison d'une boucle externe de position, s'appuyant sur les modélisations géométriques directe et inverse, avec une boucle interne de vitesse a permis d'obtenir un suivi de trajectoire précis et robuste. Les simulations ont validé les performances du régulateur *PID* en termes de précision, rapidité et stabilité, confirmant son adéquation aux systèmes robotiques à dynamique modérée comme le *SCARA*. Bien que présentant certaines limitations face aux fortes non-linéarités, cette approche reste particulièrement pertinente pour les applications industrielles, offrant un excellent compromis entre simplicité de mise en œuvre et performances opérationnelles. Ces résultats soulignent l'intérêt des correcteurs *PID* pour la commande de robots manipulateurs dans des contextes industriels exigeants.

Chapitre 3 : Animation et Réalisation du Robot *SCARA*

3 Chapitre 3 : Animation du Robot SCARA

3.1 Introduction

Ce chapitre présente d'abord la simulation du robot *SCARA* à partir des consignes articulaires générées par les lois de commande. Cette simulation, réalisée sous *MATLAB/Simulink*, permet d'analyser le comportement dynamique du système. Dans un second temps, la réalisation de l'animation du robot est entreprise afin de visualiser ses mouvements et valider les résultats obtenus. Pour cela, plusieurs outils sont mobilisés : les logiciels de *CAO* (*SolidWorks*, *Onshape*) pour la conception des pièces, et les environnements *MATLAB* (*Simscape Multibody*, *Robotics Toolbox*) pour la mise en mouvement. Trois approches sont réalisées et comparées : l'importation du modèle *CAO* via *Simscape Multibody*, l'utilisation du format *URDF*, et une animation personnalisée à l'aide de scripts *MATLAB* et la réalisation permet une validation visuelle efficace du fonctionnement du robot, en préparation à la simulation globale.

3.2 Outils de modélisation et d'animation

Pour visualiser avec précision le comportement du robot *SCARA*, il est indispensable de s'appuyer sur des outils adaptés à la modélisation mécanique, à la simulation dynamique et à l'animation 3D. Cette section présente les environnements logiciels utilisés pour concevoir, importer et animer le robot. Elle couvre les solutions proposées par *MATLAB/Simulink* ainsi que les logiciels de *CAO* comme *SolidWorks* et *Onshape*.

3.2.1 MATLAB et ses outils dédiés à la robotique

MATLAB constitue une plateforme complète pour la robotique, combinant calcul numérique, simulation et visualisation. Différents outils sont intégrés pour répondre aux besoins de modélisation et d'animation :

3.2.1.1 Simulink

Simulink est un environnement graphique intégré à *MATLAB*, conçu pour la modélisation, la simulation et l'analyse de systèmes dynamiques multi-domaines. Il permet de construire des modèles à l'aide de blocs fonctionnels interconnectés, ce qui facilite le développement de systèmes

Chapitre 03 : Animation du Robot SCARA

mécatroniques, y compris les robots articulés, en combinant facilement les aspects mécaniques, électroniques et de contrôle [24].

3.2.1.2 *Simscape Multibody*

(Anciennement appelé *SimMechanics*) est un environnement de modélisation physique proposé par *MathWorks*, qui permet de représenter des systèmes mécaniques complexes sous forme de corps rigides interconnectés. Il facilite l'importation directe d'assemblages 3D à partir de logiciels de CAO tels que *SolidWorks* ou *Onshape*, grâce à des outils d'interface dédiés. La simulation des mouvements est visualisée dans un environnement 3D interactif appelé *Mechanics Explorer*, qui permet d'observer les déplacements des composants en temps réel, en tenant compte des liaisons et des contraintes physiques définies dans le modèle [25].

Simscape Multibody est compatible avec les versions récentes de *MATLAB*, notamment de R2021a à R2024a.

3.2.1.3 *Robotics System Toolbox*

Robotics System Toolbox est une extension de *MATLAB* qui fournit un ensemble d'outils dédiés à la conception, à la simulation et à la visualisation de systèmes robotiques. Elle permet d'importer des modèles de robots décrits au format *URDF* (*Unified Robot Description Format*), utilisé notamment dans *ROS* (*Robot Operating System*). Grâce à ses fonctions intégrées, comme `show(robot, q)`, il est possible de visualiser les configurations articulaires du robot dans une boucle temporelle, facilitant l'animation de trajectoires définies par des vecteurs de positions articulaires [26].

3.2.1.4 Fonctions graphiques de base

Fonctions graphiques standards de *MATLAB*, telles que `plot`, `patch`, `line`, ou encore `drawnow`, offrent des outils puissants pour créer des animations personnalisées. Ces fonctions permettent de construire manuellement l'apparence du robot à chaque instant de la simulation, en actualisant dynamiquement les figures selon les positions articulaires. Cette approche assure une grande souplesse pour concevoir des visualisations spécifiques non limitées par les structures de modélisation prédéfinies [27].

3.2.2 Logiciels de CAO

La conception géométrique du robot *SCARA* a été réalisée à l'aide de deux outils de *CAO*, chacun offrant des avantages spécifiques selon le format de sortie attendu :

3.2.2.1 *SolidWorks*

SolidWorks est un logiciel de *CAO* paramétrique 3D largement adopté dans les domaines de l'ingénierie et de la fabrication. Il permet de modéliser précisément des composants mécaniques et d'effectuer des assemblages complets, comme ceux d'un robot *SCARA*. Dans ce projet, *SolidWorks* a été utilisé pour concevoir les différentes pièces du robot, telles que le bras, la colonne et la base. Grâce à l'outil *Simscape Multibody Link*, proposé par *MathWorks*, les modèles *CAO* peut être exportés directement vers *MATLAB/Simulink*, permettant ainsi de générer automatiquement un modèle physique utilisable dans l'environnement *Simscape Multibody* [28].

3.2.2.2 *Onshape*

Onshape est une solution de modélisation 3D en ligne, entièrement basée sur le cloud, qui permet de concevoir et de collaborer sur des assemblages mécaniques depuis un navigateur web. Contrairement aux logiciels de *CAO* traditionnels, *Onshape* intègre nativement une fonction d'exportation directe au format *URDF* (*Unified Robot Description Format*). Cette fonctionnalité facilite l'intégration des modèles dans la *Robotics System Toolbox* de *MATLAB* pour les simulations cinématiques, notamment lorsqu'on ne souhaite pas inclure les aspects dynamiques du système [29].

Ces outils de conception assistée par ordinateur permettent de générer un modèle 3D réaliste du robot, qui peut ensuite être utilisé pour simuler et analyser ses mouvements dans différents environnements logiciels.

3.2.3 Justification du choix

Pour la modélisation mécanique du robot *SCARA*, *SolidWorks* a été retenu comme principal outil de Conception Assistée par Ordinateur (*CAO*). Ce choix repose sur plusieurs critères déterminants :

- **Interface intuitive et puissante**, facilitant la création de pièces paramétrées avec précision.

Chapitre 03 : Animation du Robot *SCARA*

- **Outils d'assemblage avancés**, permettant de définir facilement les relations cinématiques (liaisons pivot, contraintes de coïncidence, etc.).
- **Compatibilité directe avec *MATLAB*** grâce à l'extension *Simscape Multibody Link*, qui permet d'exporter les modèles 3D vers l'environnement Simulink sans perte de géométrie.
- Large adoption dans l'industrie, assurant une continuité pédagogique et professionnelle.

Ainsi, *SolidWorks* offre une solution robuste pour modéliser un robot à plusieurs degrés de liberté et l'intégrer efficacement dans une chaîne de simulation.

3.3 Patrie animation du robot

3.3.1 Modélisation des pièces du robot

Le robot *SCARA* a été modélisé en concevant séparément ses principaux composants mécaniques :

3.3.1.1 Base

Support initial du robot, ancrée au sol ou à une table.



Figure 3.1 La base du robot

3.3.1.2 Bras 1 (premier bras articulé)

Conçu avec une liaison pivot à sa base, il effectue la première rotation autour de la colonne.

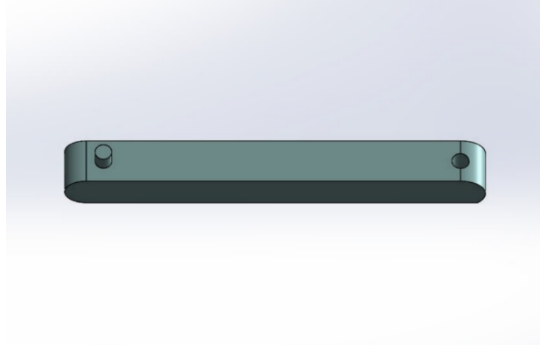


Figure 3.2 Le 1-er bras

3.3.1.3 Bras 2 (deuxième bras articulé)

Fixé à l'extrémité du bras 1, il réalise une rotation relative par rapport à ce dernier.

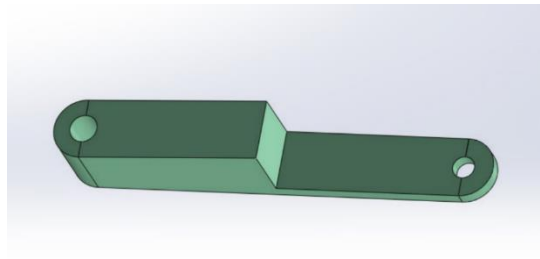


Figure 3.3 Le 2 -ème bras

3.3.1.4 Axe de l'effecteur terminal

Représenté par un cylindre vertical rotatif, situé à l'extrémité du bras 2, assure une double fonction : d'une part, il effectue une rotation autour de son axe, représentant le troisième degré de liberté du robot ; d'autre part, il agit comme un guide creux permettant à l'effecteur terminal de se déplacer verticalement à l'intérieur, ce qui correspond au quatrième degré de liberté.

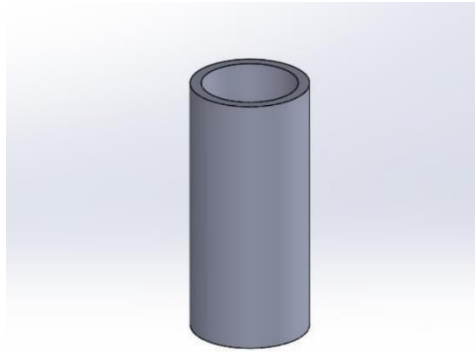


Figure 3.4 Axe de l'effecteur terminal

3.3.1.5 Effecteur terminal

Modélisé comme une pièce insérée dans le cylindre, il effectue un mouvement de translation verticale, simulant la montée/descente typique du *SCARA*.

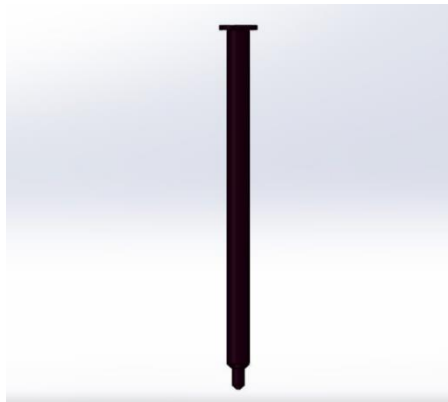


Figure 3.5 Effecteur terminal

3.3.1.6 Assemblage complet du Robot

Une fois les pièces modélisées, elles ont été **assemblées dans *SolidWorks*** en respectant la structure cinématique du robot :

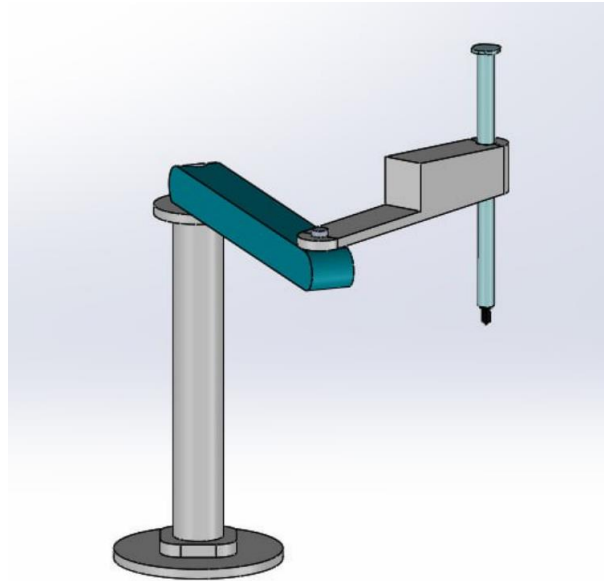


Figure 3.6 Assemblage du Robot

- Liaisons rotatives : Entre les bras et le cylindre (DOF 1 et 2 et 3).
- Liaison prismatique : Pour l'axe vertical (DOF 4).

3.3.2 Exportation et Simulation du modèle mécanique

Une fois le robot SCARA modélisé dans l'environnement de CAO, deux méthodes d'exportation ont été utilisées afin d'intégrer le modèle mécanique dans *MATLAB* pour simulation et animation : l'exportation vers *Simscape Multibody* et l'exportation au format *URDF*. Ces approches permettent d'exploiter respectivement les capacités de modélisation physique et cinématique de *MATLAB/Simulink*.

3.3.2.1 Exportation vers Simscape Multibody

Tous d'abord, Il faut lier *SolidWorks* avec Matlab par les étapes suivantes

(Les fichiers d'installation doivent être dans le même dossier)

Command Window

```
fx >> install_addon('smlink.r2024a.win64.zip')  
smlink_linksw
```


Chapitre 03 : Animation du Robot SCARA

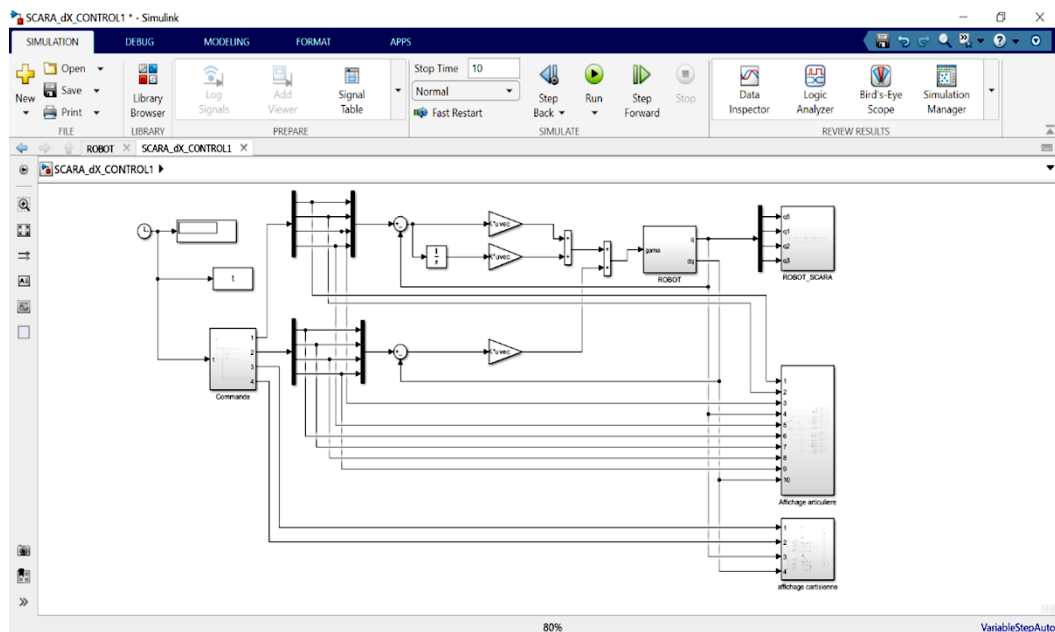


Figure 3.8 Schema de commande principale

Une fois la simulation lancée, le mouvement du robot est animé automatiquement dans *Mechanics Explorer*, permettant ainsi une visualisation en temps réel de son comportement mécanique. Comme représenté ci-dessous.

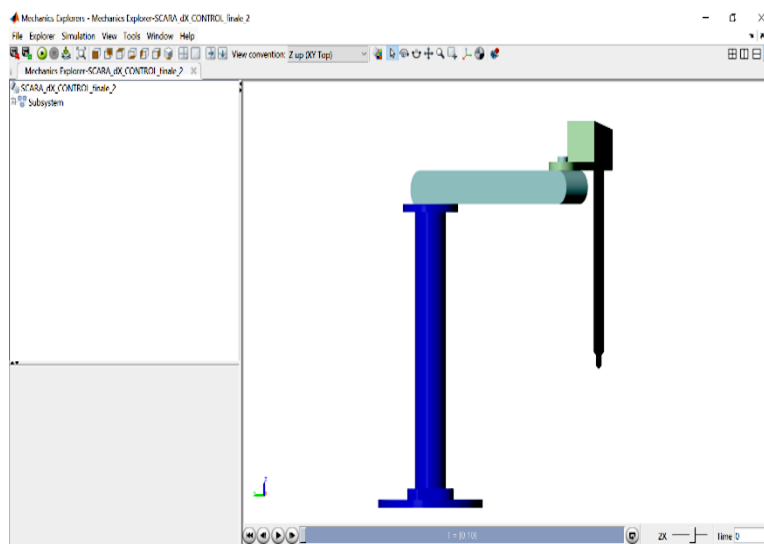


Figure 3.9 Simulation Mechanics Explorer t=0

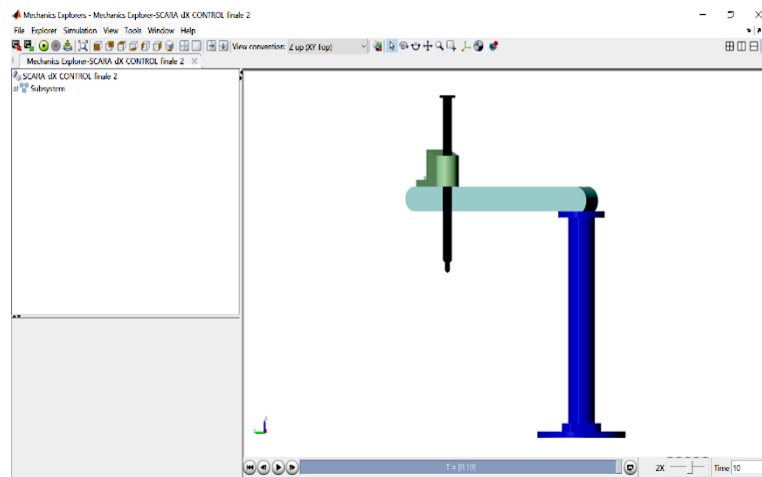


Figure 3.10 Simulation Mechanics Explorer t=10

3.3.2.2 Export *URDF* pour *MATLAB*

Pour exporter le robot au format *URDF*, dans *SolidWorks*, si l'extension adéquate est installée, on utilise l'option d'exportation *URDF* disponible via

Outil > Exporter au format *URDF*

Il est important de configurer correctement les liaisons, les axes de rotation, ainsi que les noms des pièces. Une fois l'exportation effectuée, on place le dossier *URDF* (contenant les fichiers *.urdf* et les sous-dossiers *meshes*) dans le répertoire de travail *MATLAB*, afin de pouvoir importer le robot avec la commande

Command Window

```
fx>> Robot = importrobot(' ROBOT_SCARA.urdf ')  
show(Robot)
```

Pour contrôler le robot importé au format *URDF*, on a utilisé les valeurs articulaires q obtenues à partir de la simulation *Simulink*, afin de reproduire le mouvement dans *MATLAB*. Deux trajectoires été réalisées et suivies par le robot, en mettant à jour sa configuration à chaque instant via une boucle d'animation.

Chapitre 03 : Animation du Robot SCARA

- Résultat d'animation « référence 1 » : Trajectoire simple

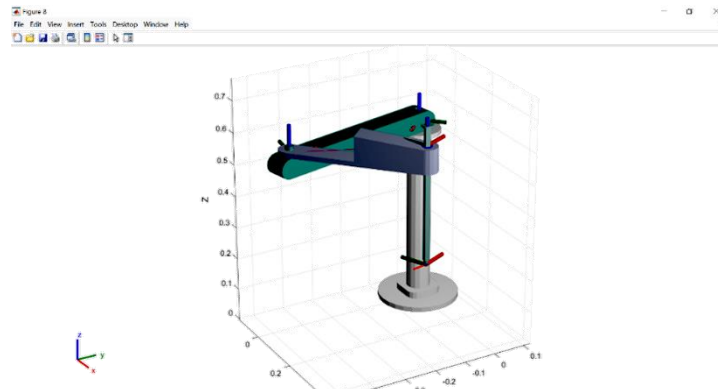


Figure 3.11 Visualization *URDF* position initial *ref1*

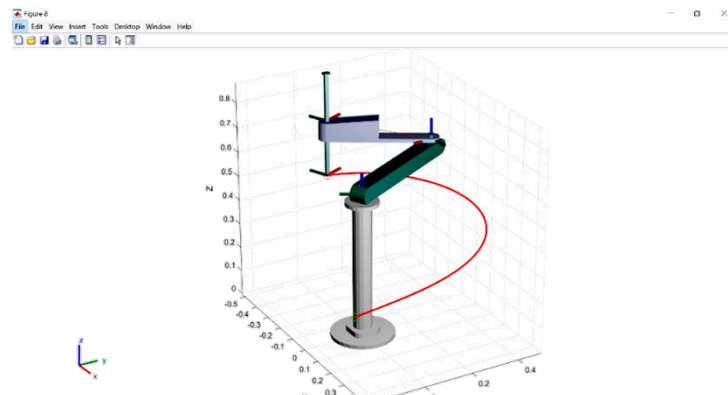


Figure 3.12 Visualization *URDF* position finale *ref1*

- Résultat d'animation « référence 2 » : Trajectoire complexe

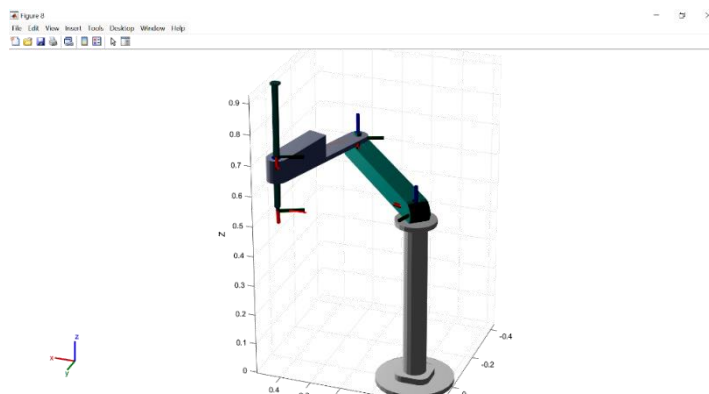


Figure 3.13 Visualization *URDF* position initial *ref2*

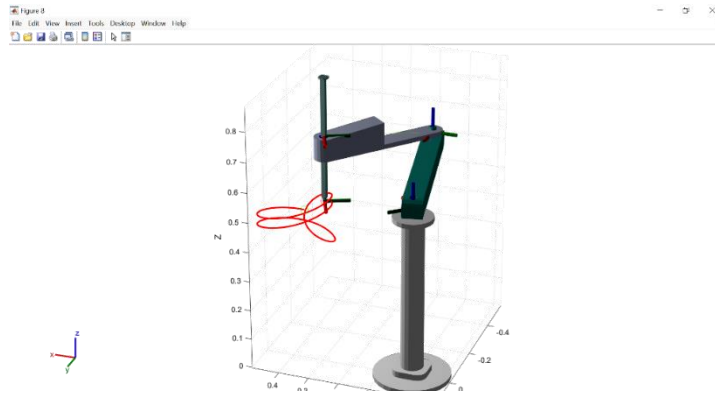


Figure 3.14 Visualization *URDF* position finale *ref2*

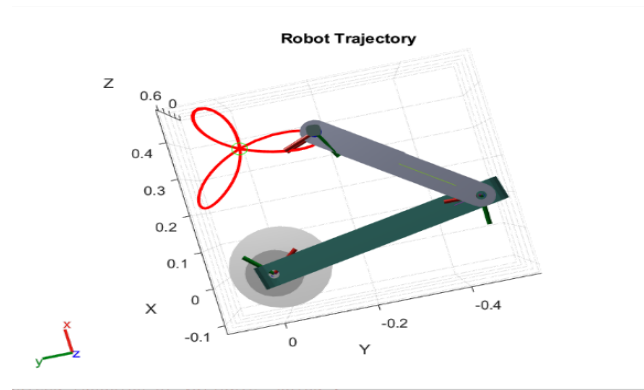


Figure 3.15 Visualisation de la position finale en vue de haut

3.3.2.3 Animation personnalisée dans MATLAB

Dans cette section, une animation personnalisée a été développée sous *MATLAB* afin de visualiser en temps réel l'évolution des positions, des erreurs et des vitesses du robot *SCARA*. La figure générée est organisée en plusieurs sous-graphes affichant respectivement :

- Les positions et erreurs selon les axes X, Y et Z.
- La trajectoire *3D* de l'effecteur (Réelle vs. Référence).
- La vitesse *3D* correspondante.

La mise à jour des courbes est effectuée dynamiquement dans une boucle temporelle, ce qui permet de suivre l'évolution du comportement du robot tout au long de sa trajectoire. Deux vues *3D* permettent également de comparer la trajectoire de position et celle de la vitesse dans l'espace.

Chapitre 03 : Animation du Robot SCARA

Les figures suivantes illustrent l'évolution dynamique des positions, des erreurs et des vitesses du robot SCARA au cours du suivi de trajectoire, générées via l'animation personnalisée sous MATLAB.

- Résultat d'animation « référence 1 » : Trajectoire simple

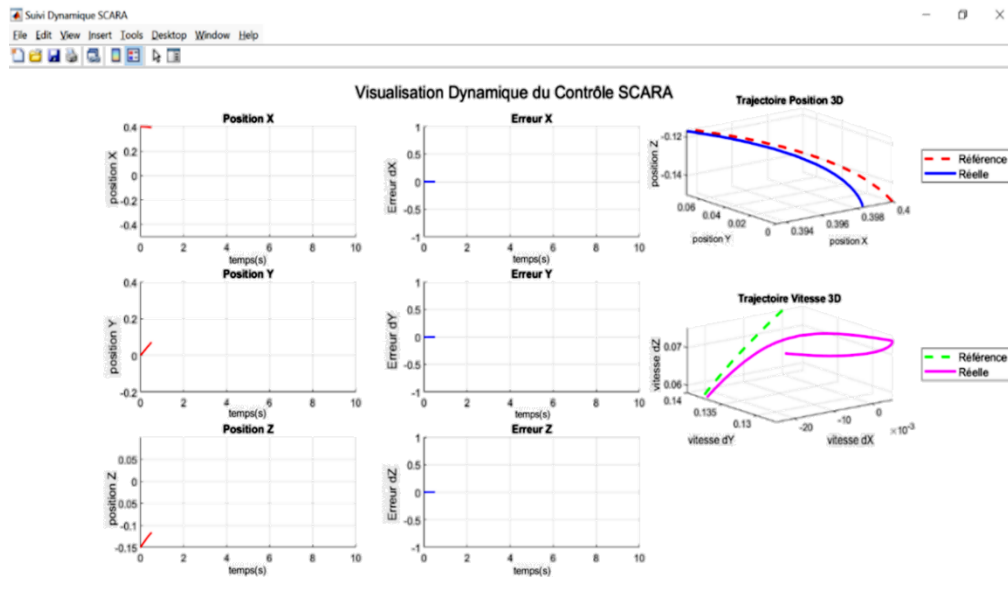


Figure 3.16 Visualisation Dynamique du Contrôle SCARA au départ *ref1*

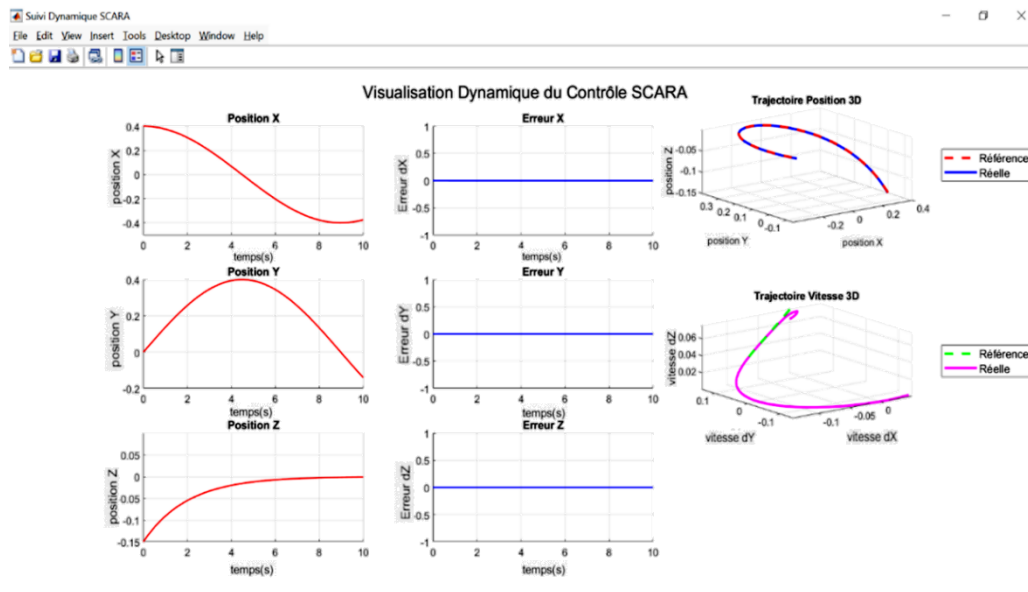


Figure 3.17 Visualisation Dynamique du Contrôle SCARA finale *ref1*

Chapitre 03 : Animation du Robot SCARA

- Résultat d'animation « référence 1 » : Trajectoire complexe

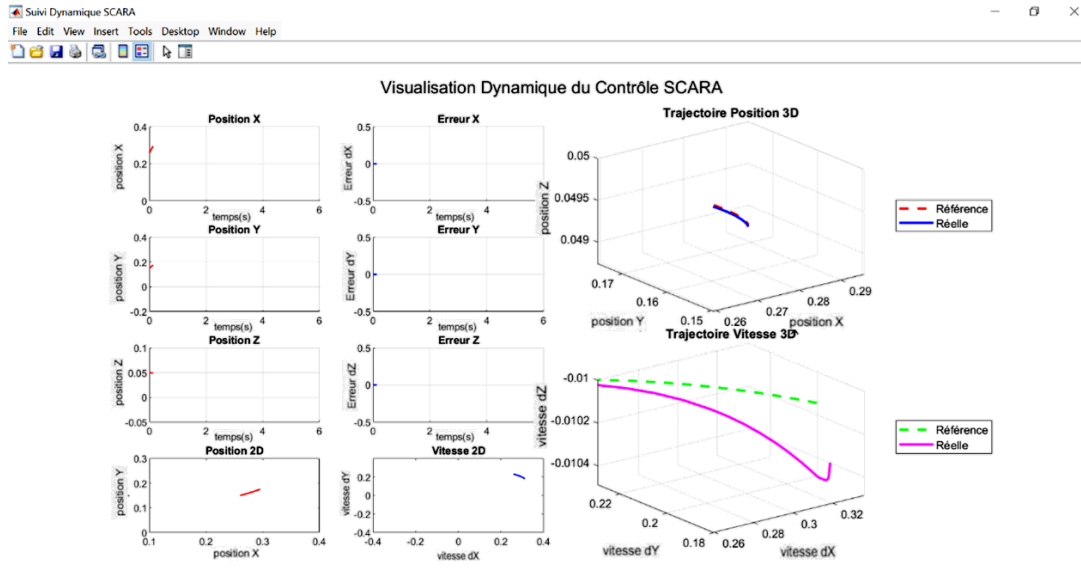


Figure 3.18 Visualisation Dynamique du Contrôle SCARA au départ *ref1*

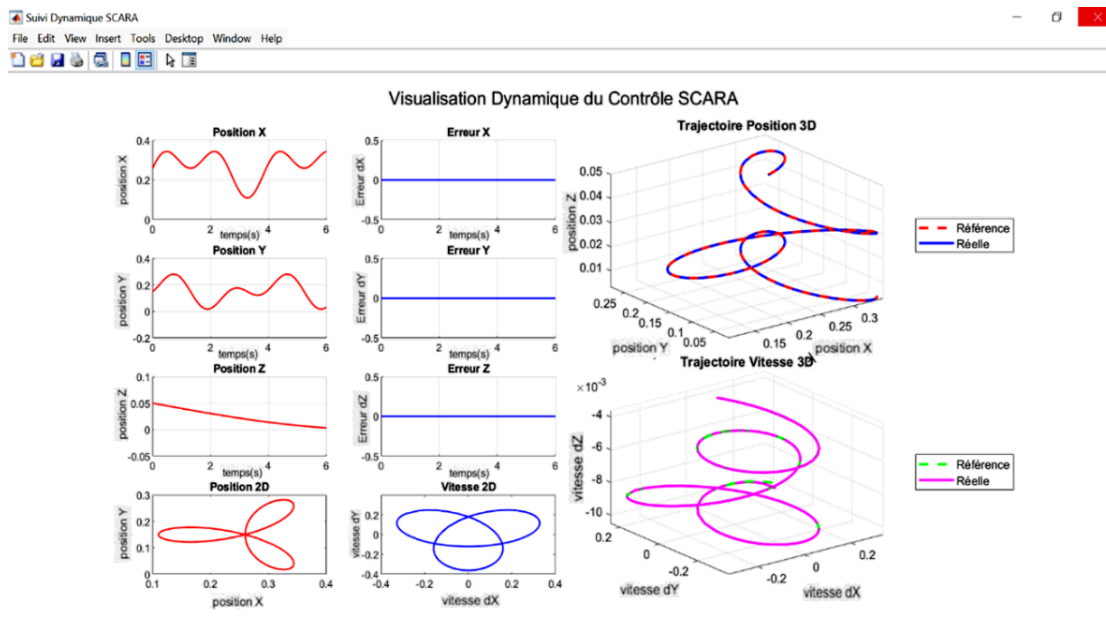


Figure 3.19 Visualisation Dynamique du Contrôle SCARA finale *ref2*

3.4 Patrie réalisation du robot

3.4.1 Conception mécanique du robot SCARA

La conception mécanique du robot SCARA a été réalisée à l'aide du logiciel de modélisation 3D *SolidWorks*. Chaque composant du robot a été modélisé individuellement afin de respecter les dimensions et les contraintes mécaniques du système. L'ensemble des pièces a ensuite été assemblé virtuellement pour former la structure complète du robot. Une fois la modélisation terminée, les fichiers ont été exportés au format *STL*, un standard compatible avec les imprimantes 3D. Cette conversion a permis de fabriquer physiquement les différentes parties du robot à l'aide d'une imprimante 3D, en vue de tests mécaniques et d'assemblage réel.

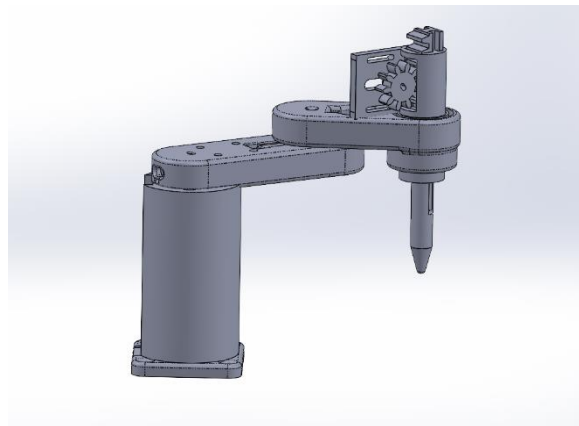


Figure 3.20 la structure complète du robot

3.4.2 Choix des composants électroniques

Pour la réalisation de l'animation physique du robot plusieurs composants électroniques ont été sélectionnés en tenant compte de leur simplicité d'utilisation, de leur coût et de leur compatibilité avec les besoins du projet.

3.4.2.1 Moteur pas-à-pas 28BYJ-48 + driver ULN2003

Le moteur 28BYJ-48 est un moteur pas-à-pas unipolaire à 5 fils, souvent utilisé dans les applications de prototypage grâce à sa simplicité et à son faible coût. Il fonctionne généralement sous 5V et offre une résolution suffisante pour des mouvements précis. Il est couplé au driver

ULN2003, qui permet d'interface les signaux de commande de l'*Arduino* avec les bobines du moteur [30].

3.4.2.2 Carte *Arduino Uno*

L'*Arduino Uno* est une carte de développement basée sur le microcontrôleur ATmega328P. Elle permet la génération des signaux numériques de commande et l'exécution de programmes embarqués. Son environnement de développement simple (*IDE Arduino*) ainsi que ses nombreuses bibliothèques facilitent son utilisation dans des projets robotiques éducatifs ou semi-industriels [31].

3.4.3 Impression 3D des pièces du robot

3.4.3.1 Préparation du modèle 3D à partir de *SolidWorks*

Les différentes pièces du robot *SCARA* ont d'abord été modélisées sous *SolidWorks*. Pour les rendre compatibles avec l'impression 3D, chaque pièce a été exportée au format *STL*, un format standard reconnu par la majorité des logiciels de tranchage. Dans notre cas, le logiciel *Luban* du fabricant *Snapmaker* a été utilisé pour générer le fichier de commande destiné à l'imprimante 3D.

Une fois le modèle chargé dans le logiciel, nous avons choisi l'imprimante adaptée, puis réglé les paramètres d'impression selon la forme de chaque pièce. Certaines pièces, à cause de leur forme compliquée ou de parties qui flottent dans l'air, ont besoin de supports pour bien s'imprimer sans défauts. Ces supports aident à maintenir la pièce pendant l'impression et assurent une bonne qualité.

Chapitre 03 : Animation du Robot SCARA

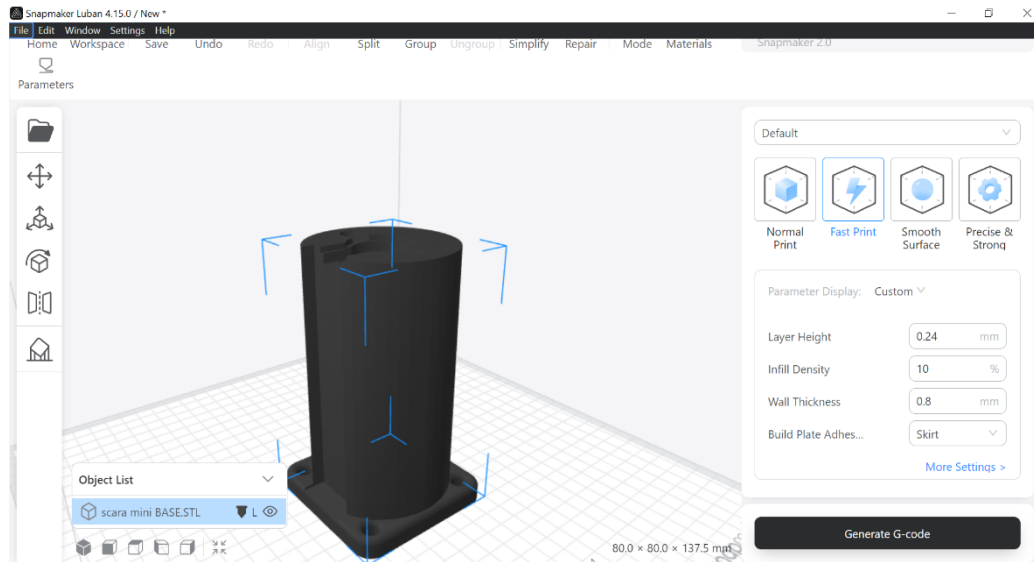


Figure 3.21 Motif impression dans Luban

3.4.3.2 Génération du G-code et impression

Après avoir correctement orienté et positionné les pièces dans l'espace d'impression virtuelle, le logiciel *Luban* a permis de générer le fichier *G-code*. Ce dernier représente les instructions interprétables directement par l'imprimante 3D pour exécuter l'impression couche par couche. Une fois le *G-code* transféré, les pièces ont été imprimées une à une en tenant compte des contraintes de dimensions, de support et de qualité de finition.

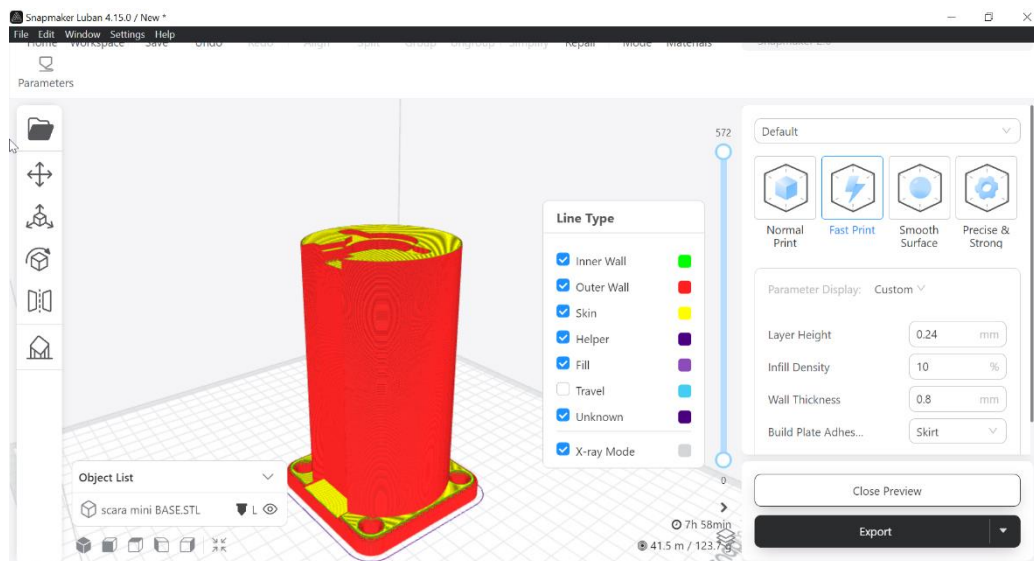


Figure 3.22 Generation de G-Code

3.4.4 Visualisation des pièces après impression 3D

Ci-dessous, on peut voir les pièces du robot *SCARA* une fois imprimées en 3D. Ces photos montrent le résultat final obtenu à partir des modèles conçus sur *SolidWorks* et préparés avec le logiciel d'impression.



Figure 3.23 Les pièces après impression 3D

3.4.5 Schéma électronique sous Fritzing

Le schéma électronique élaboré à l'aide de *Fritzing* regroupe les composants essentiels du système, tels que les drivers des moteurs pas-à-pas, l'alimentation, ainsi que les connexions entre la carte *Arduino* et les moteurs. Ce schéma offre une vue d'ensemble claire de l'architecture électrique du montage, facilitant sa vérification avant l'assemblage physique.

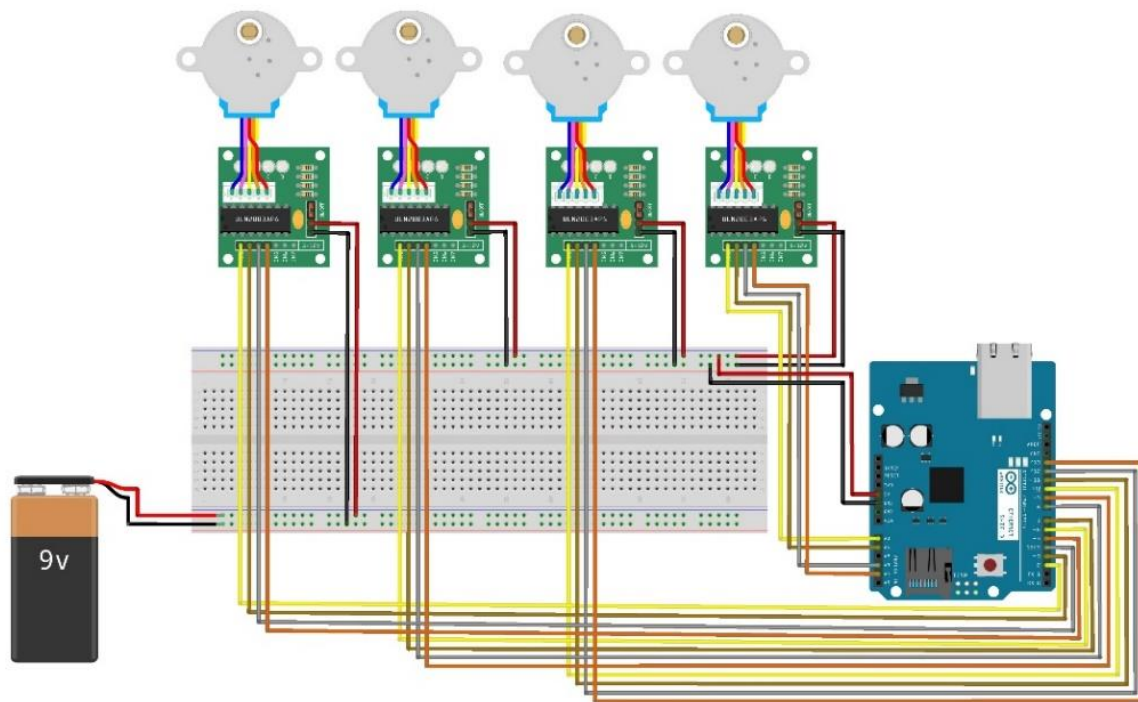


Figure 3.24 Schéma électronique sous Fritzing

3.4.6 Assemblage du Robot

Les pièces mécaniques du robot ont été imprimées puis assemblées selon les plans établis. Les composants électroniques, incluant la carte *Arduino* et les drivers moteurs, ont été câblés conformément au schéma électrique. Après cette étape, le code généré à partir de *MATLAB* a été transféré sur la carte *Arduino*. Le système ainsi mis en place a permis de contrôler le robot et de visualiser la trajectoire prévue.



Figure 3.25 Assemblage du robot

3.4.7 Génération du code à partir de MATLAB

À l'issue de la simulation sous *Simulink*, les signaux de position angulaire des quatre articulations du robot SCARA, notés q_1, q_2, q_3, q_4 , sont exportés vers l'espace de travail *MATLAB* à l'aide du bloc "*To Workspace*". Ces variables sont regroupées sous la forme d'une matrice q , où chaque ligne correspond à un instant de la trajectoire, et chaque colonne à une articulation.

Cette matrice q , de taille $(N \times 4)$ (avec N le nombre d'échantillons temporels), représente donc la trajectoire complète du robot. Elle est ensuite extraite et exploitée dans un script *MATLAB* en vue de générer le code de commande destiné à être embarqué sur la carte *Arduino*. Ce code permet de reproduire le mouvement simulé en pilotant physiquement les moteurs pas-à-pas du robot.

L'approche adoptée assure une continuité entre la simulation et l'expérimentation réelle, tout en facilitant la vérification et la validation des trajectoires avant leur exécution sur le système physique.

3.4.8 Implémentation dans *Arduino*

La matrice q_1, q_2, q_3, q_4 , provenant de *MATLAB* est convertie pour être compatible avec le langage *Arduino*, qui est basé sur le C/C++. Cette matrice est ensuite insérée dans le programme *Arduino*, permettant au robot de suivre la trajectoire simulée.

3.5 Conclusion

Ce chapitre a présenté l'animation du robot *SCARA* à travers différentes approches de modélisation et de simulation, avec pour objectif de visualiser le comportement du système à partir des résultats de commande. Après la modélisation mécanique dans des environnements de *CAO* comme *SolidWorks*, deux méthodes d'intégration ont été exploitées : l'exportation vers *Simscape Multibody* pour une simulation physique complète, et l'exportation au format *URDF* pour une animation cinématique articulée. En complément, une animation personnalisée a été développée à l'aide des fonctions graphiques de *MATLAB*, offrant une flexibilité accrue pour illustrer les trajectoires du robot. L'analyse comparative de ces trois techniques a permis de mettre en évidence les avantages et limites de chacune en termes de précision visuelle, d'adaptabilité et de facilité d'implémentation. L'animation s'est ainsi révélée être un outil fondamental pour appuyer la validation fonctionnelle et visuelle du robot *SCARA*, en reliant concrètement les résultats de simulation aux comportements attendus du système.

Conclusion Générale

Conclusion Générale

Ce mémoire traite de l'étude, de la commande et de l'animation d'un robot *SCARA* à quatre degrés de liberté, avec pour objectif de développer une approche complète intégrant modélisation, simulation et visualisation dynamique. La problématique centrale porte sur la mise en œuvre d'une commande efficace respectant les contraintes mécaniques et dynamiques du robot.

Une démarche progressive a été adoptée : modélisation mécanique et géométrique du robot, élaboration d'une commande basée sur des régulateurs *PID*, et intégration d'une animation dans *MATLAB/Simulink* pour visualiser le comportement simulé du robot. Les résultats ont montré la stabilité et la précision du système de commande, ainsi que l'intérêt pédagogique de l'animation.

Ce travail offre une compréhension structurée des étapes nécessaires à la commande d'un manipulateur robotique et souligne l'utilité de combiner modélisation, simulation et visualisation. Toutefois, certaines limites sont relevées, notamment l'absence de prise en compte des perturbations réelles et de retour capteur.

Des perspectives d'amélioration sont proposées, comme l'implémentation de commandes plus robustes, l'intégration de capteurs, la connexion à un système réel ou le développement d'une interface interactive. Ce mémoire constitue ainsi une base solide pour des applications plus avancées en robotique.

Références bibliographiques

Références bibliographiques

- [1]. HAJ SALAH, Med Ali. L'historique de la robotique [en ligne]. [Consulté le : 20 février 2025]. Disponible sur : <https://www.robotique.tech/blog/lhistorique-de-la-robotique/>
- [2] INTEL. Types et applications des robots [en ligne]. [Consulté le : 20 février 2025]. Disponible sur : <https://www.intel.fr/content/www/fr/fr/robotics/types-and-applications.html>
- [3]. CAPITANI, Martina. Il primo esemplare di robot : l'Automa Cavaliere di Leonardo da Vinci [en ligne]. [Consulté le : 1 mai 2025]. Disponible sur : <https://metropolitanmagazine.it/il-primo-esemplare-di-robot-lautoma-cavaliere-di-leonardo-da-vinci/>
- [4]. LAUMOND, Jean-Paul. Robotique : entre science, technologie et imaginaire. Paris : Odile Jacob, 2023. 284 p. ISBN 978-2-415-00604-4
- [5]. COLAS, Francis. Robotique autonome - Introduction [en ligne]. [Consulté le : 1 mai 2025]. Disponible sur : https://members.loria.fr/FColas/lectures/robotique_autonome_sir/1_introduction.pdf
- [6]. DUPONT, Jean. Les composants essentiels d'un robot [en ligne]. [Consulté le : 1 mai 2025]. Disponible sur : <https://www.robotique.tech/blog/les-composants-essentiels-d-un-robot/>
- [7]. OFFICE QUÉBÉCOIS DE LA LANGUE FRANÇAISE. Articulation de robot [cour en ligne]. [Consulté le : 1 mai 2025].
- [8] AMEDDAH, Hacene. Dynamique – Concept robotique. Université de Batna 2, 74 p. [cour en ligne].
- [9]. AMEDDAH, Hacene. Machine-outil à commande numérique. Université de Batna, 65 p. [cour en ligne].
- [10]. Plant Automation Technology. Types of Robots Based on Configuration. Consulté le 29 juin 2025, depuis : <https://www.plantautomation-technology.com/articles/types-of-robots-based-on-configuration>.
- [11] KHALIL, Wisama et DOMBRE, Étienne. Modélisation, identification et commande des robots. 2e éd. Paris : Hermès Science Publications, 2002. 374 p. ISBN 2-7462-0003-1.

Références bibliographiques

- [12]. KHALIL, Wisama et DOMBRE, Étienne. Modélisation, identification et commande des robots. 2^e éd. Paris : Hermès Science Publications, 1999. 503 p. (Traité des nouvelles technologies. Série Robotique). ISBN 2-7462-0003-1.
- [13]. KERMICHE, Salah. Modélisation et commande d'un robot par méthodes intelligentes. Thèse de doctorat en Automatique Industrielle, Université Badji Mokhtar – Annaba, 2006. 206 p.
- [14]. TORRES-DEL CARMEN, Felipe de Jesús, JARAMILLO-HERNÁNDEZ, Ricardo, DÍAZ-SÁNCHEZ, Arnaldo et NÚÑEZ-ALTAMIRANO, Diego Alfredo. "Comparison of numerical methods in code as solvers for simulation of robotic systems". *Revista de Cómputo Aplicado*, vol. 15, n° 4, pp. 1–8, décembre 2020. DOI : 10.35429/JCA.2020.15.4.1.8.
- [15]. LAVÍN-DELGADO, Jorge Enrique, CHÁVEZ-VÁZQUEZ, Samuel, GÓMEZ-AGUILAR, José Francisco, DELGADO-REYES, Gerardo et RUIZ-JAIMES, Miguel Ángel. "Fractional-order passivity-based adaptive controller for a robot manipulator type SCARA". *Fractals*, vol. 28, n° 5, article 2040008, mai 2020. DOI : 10.1142/S0218348X20400083.
- [16] Spong, M. W., Hutchinson, S., & Vidyasagar, M. (2006). *Robot Modeling and Control*. Hoboken, NJ, USA : John Wiley & Sons. ISBN 978-0-471-64990-8.
- [17]. Wikipédia. Régulation (automatique). Disponible sur : [https://fr.wikipedia.org/wiki/Régulation_\(automatique\)](https://fr.wikipedia.org/wiki/Régulation_(automatique)) (consulté en avril 2025).
- [18]. ÅSTRÖM, Karl J. et HÄGGLUND, Tore. *Advanced PID Control*. Research Triangle Park, NC: ISA – The Instrumentation, Systems, and Automation Society, 2006. 461 p. ISBN 978-1-55617-942-6
- [19]. Dorf, R. C., & Bishop, R. H. (2017). *Modern Control Systems* (13^e édition). Pearson Education. ISBN 978-0-13-440762-3.
- [20]. Ogata, K. (2010). *Modern Control Engineering* (5^e édition). Prentice Hall. ISBN 978-0-13-615673-4.
- [21]. Craig, J. J. (2005). *Introduction to Robotics: Mechanics and Control* (3^e édition). Pearson Prentice Hall. ISBN 978-0-13-148214-3.

Références bibliographiques

- [22]. Dr. AMEDDAH, HACENE. COURS Dynamique Concept robotique 74p.Université de Batna.
- [23]. Morbidi, Fabio. Robotique Industrielle. Lausanne, Suisse : Presses Polytechniques et Universitaires Romandes (PPUR), 2020. ISBN : 978-2-88915-360-2.
- [24]. MathWorks. (2024). Simulink Overview – Simulate and Model Dynamic Systems. Récupéré de <https://www.mathworks.com/products/simulink.html> (consulté en avril 2025).
- [25]. MathWorks. (2024). Simscape Multibody Documentation. Récupéré de <https://www.mathworks.com/help/physmod/sm/> (consulté en avril 2025).
- [26]. MathWorks. (2024). Robotics System Toolbox Documentation. Récupéré de <https://www.mathworks.com/help/robotics/> (consulté en avril 2025).
- [27]. MathWorks. (2024). Techniques for Visualizing Simulation Results. Récupéré de https://www.mathworks.com/help/matlab/creating_plots/animation-techniques.html (consulté en avril 2025).
- [28]. MathWorks. (2024). Simscape Multibody Link – Import CAD Assemblies into Simscape Multibody. Récupéré de <https://www.mathworks.com/help/physmod/smlink/> (consulté en avril 2025).
- [29]. Onshape. (2024). URDF Export for Robotics Integration. Récupéré de <https://www.onshape.com/en/resource-center/tech-tips/tech-tip-using-onshape-with-ros> (consulté en avril 2025).
- [30]. A. Dogan, Beginning *Arduino*, Apress, 2012.
- [31]. M. Banzi and M. Shiloh, Getting Started with *Arduino*, 3rd ed., Maker Media, 2014.