الجـــــــــــــــــــــــمهـوريـــــــة الجـــــــــــــــزائـريـة الديمـــــــــقـراطيـة
الـشـــــــــــعبـيـة

RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE ET POPULAIRE

وزارة التـــــــــــــــــــــــــــــــعـليـم العـــــــــالي والبـــــــــــحث
العـــــــلـمـــــي

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

جامعة محمد البشير الإبراهيمي – برج بوعريريج

Université de Mohamed El-Bachir El-Ibrahimi - Bordj Bou Arreridj

Faculté des Sciences et de la Technologie

Département Eléctromécanique

MÉMOIRE

Présenté pour l'obtention du **diplôme** de **MASTER**

En : Automatique

Spécialité : Automatique et informatique industrielle

Par : - FATMI djouhaina

Sujet

Commande et développement d'un bras manipulateur

Soutenu publiquement, le 30 / 06 /2025 , devant le jury composé de :

Mme.ZEHAR Djamila	MCB	**Univ-BBA**	Président
Mr.BENSIDHOUM Tarek	MCA	**Univ-BBA**	Examinateur
Mr.ZAOUI Fares	MCA	**Univ-BBA**	Encadrant

Année Universitaire 2024/2025

Remerciments

Tout d'abord AL HAMDU LI ALLAH merci à Dieu de m'avoir donné la force pour Terminer ce travail

Le travail présenté dans ce mémoire a été réalisé au Département, de Sciences et Technologie, Spécialité : Automatique université mohammed bachir al ibrahimi

Je remercie monsieur **Dr. ZAOUI.Fares** et **Dr.KHANFER.Riyad** pour son esprit scientifique et compréhensif, qui a consacré son temps à mon travail et m'a beaucoup aidé avec ses idées, ses conseils et surtout ses critiques objectives.

Je tiens également à remercier **Dr. ZEHAR Djamila** pour avoir accepté de présider le jury, et à **Dr. BENSIDHOUM Tarek** d'avoir pris de leurs temps pour examiner ce travail.

Merci à tous les autres chercheurs qui ont mis leurs outils, articles et travaux accessibles à travers l'Internet.

Je veux également remercier ma famille et mes amies pour le soutien moral. Enfin, je remercie tout particulièrement **mes parents**, pour leur soutien inconditionnel tout au long de ces longues années d'études

Dédicace

Je édié ce travail avec toute ma gratitude et mon affection.

À ***mes chers parents****, pour leur amour inconditionnel, leur soutien constant et leurs sacrifices sans limites.*

À ***mes deux sœurs adorées****, pour leur tendresse, leurs encouragements et leur présence réconfortante*

À ***toutes mes précieuses amies****, pour les moments partagés, le soutien et la motivation.*

Et à ***tous ceux que j'aime****, de près ou de loin, qui occupent une place spéciale dans mon cœur.*

Je veux vous dire que je vous aime profondément, et je prie ***Allah*** *de vous protéger et de vous accorder santé et bonheur.*

Résumé

Dans ce projet, nous avons conçu, modélisé et testé un bras robotique de type industriel à 3 degrés de liberté. Le travail a porté sur l'élaboration des modèles géométriques et cinématiques directs et inverses, en vue de piloter le robot avec précision. En suivant un cahier des charges défini, nous avons sélectionné les moteurs, les réducteurs, et réalisé un programme de commande embarqué basé sur Arduino. Des simulations et des calculs analytiques ont permis de valider les trajectoires, tandis qu'un prototype physique a été utilisé pour les tests. Ce projet constitue une base solide pour des améliorations futures comme le contrôle sans fil ou l'intégration de capteurs.

ملخص

في هذا المشروع، قمنا بتصميم ونمذجة واختبار ذراع روبوتي صناعي ب3 درجات حرية. شمل العمل تطوير النماذج الهندسية والحركية المباشرة والعكسية بهدف التحكم الدقيق في حركة الروبوت. وبالاعتماد على دفتر شروط محدد، تم اختيار المحركات ونواقل الحركة المناسبة، وإنشاء برنامج تحكم يعتمد على منصة أردوينو. وقد تم التحقق من صحة النتائج باستخدام المحاكاة والحسابات التحليلية، كما تم بناء نموذج أولي حقيقي لإجراء التجارب العملية. يشكل هذا العمل قاعدة صلبة للتطوير المستقبلي مثل التحكم اللاسلكي أو دمج الحساسات

Abstract

This project focused on the design, modeling, and testing of an industrial-type robotic arm with 3 degrees of freedom. The study involved developing both forward and inverse geometric and kinematic models to achieve precise control. Based on a defined set of specifications, we selected suitable motors, gear reducers, and implemented an embedded control program using Arduino. Analytical calculations and simulations validated the trajectories, and a physical prototype was built for testing. This work provides a strong foundation for future improvements such as wireless control or sensor integration.

Table de matière

Remerciments II
Dédicace III
Résumé IV
Table de matière V
Liste des figures VII
Liste des tableaux IX
I.Introduction générale 1
Chapitre I : Généralité sur les robots 3
1. Introduction 3
2. L'histoire de la robotique 3
3. Défénition du robot 4
3.1 Robots mobiles 5
3.2 Les robots manipulateurs 7
3.3 Caractéristiques du bras robotique 12
3.4 Architecture du robot manipulateur 13
3.5 Moteur pas à pas 15
4. Conclusion 20
Chapitre II : Étude du robot BCN3D Moveo et modélisation géométrique 3
1. Introduction 23
2. Présentation du robot BCN3D Moveo 23
2.1. Architecture mécanique et composants 23
3. L'impression 3D 24
4. Description mécanique et éléctronique des parties constitutives du bras 24
4.1. Les défférentes pièces utiliser dans notre projet 24
4.2. Les moteurs 25
4.3. L'arduino 29
5. La modélisation 31
5.1. Espace de travail 31
5.2. Modélisation des robots 31
5.2.1. Les modèles géométriques direct et inverse 32
5.2.2. Les modèles cinématiques direct et inverse 32
5.3. Modèles géométrique directe 33

5.3.1.Notation standard de Denavit-Hartenberg (DH) 36
5.4. Modèles géometrique inverse 38
5.4.1. La méthode géométrique classique 38
5.4.2. La méthode géométrique analytique(méthode de richard.paul) : 38
6. La modélisation du notre robot (RRR) 40
6.1. Le modèle géometrique direct (MGD) de BCN3D (modifier) 40
6.2. Le modèle géometrique inverse (MGI) 43
7. Simulation MATLAB : 46
8. Conclusion 48
Chapitre III : Conception et réalisation du travail 20
1.Introduction 50
2. Les outils utiliser pour la fabrication des pièces de robot 50
2.1.Imprimante Creality 3-Ender PRO 50
2.2. Filament PLA High speed 51
2.3.Creality slicer 51
2.3.1.Installation et configuration du slicer 51
3. Assemblage mécanique et intégration électronique 52
4. Commande électronique et câblage 57
4.1. Software Arduino IDE 57
4.2. Test de fonctionnement 58
4.3. Programme principal de manipulation : déplacement de A vers B : 62
5. Difficultés techniques rencontrées et solutions apportées 67
6. Conclusion : 69
Conclusion générale 71
Annex 75

Liste des figures

Chapitre I : Généralités sur les robots

Figure I. 1 : Robot a roues……..... **5**

Figure I. 2 : Robot sous marins……..... **5**

Figure I. 3 : Robot volants…….. **6**

Figure I. 4 : Robot humanoïde…… **6**

Figure I.5 : Manipulateur série…….. **8**

Figure I.6 : Manipulateur parallèle……. **8**

Figure I.7: Robots cylindriques …… **8**

Figure I.8 : Robot rectilignes.……..……. **9**

Figure I.9 : Robot sphérique……. **10**

Figure I.10 : Robot SCARA.……. **10**

Figure II.11 : Robot articulés……. **11**

Figure I.12 : Architecture d'un bras robot……..... **13**

Figure I.13 : Présentation s'une articulation prismatiques …….... **14**

Figure 1.14 : Présentation s'une articulation rotoïdes …….. **14**

Figure I.15 : Défférente types de moteur pas à pas …..……... **16**

Figure I.16 : Principe de fonctionnement du moteur pas à pas en pas complet avec alimentation biphasée.…….. **17**

Figure I.17: Principe de fonctionnement du moteur en mode demi-pas avec une alimentation biphasée..... **17**

Chapitre II : Étude du robot BCN3D Moveo et modélisation géométrique

Figure II.1 :La strcture complet de bras robotique BCN3D ……..... **24**

Figure II.2 : Les différentes parties du robot BCN3D imprimer en 3D……..…. **24**

Figure II.3: Vue physique du driver DRV8825……..... **28**

Figure II .4 : vue physique du CNC sheild …… **29**

Figure II.5 :Espace de travail d'un bras manipulateur.…… **31**

Figure II.6 : Modéèle géométrique direct et inverse …..……..... **32**

Figure II.7 : Représentation d'un point ……..…….. **34**

Figure II.8 : transformation des repères ……...…… **35**

Figure II.9: Représentation des repères et Paramètres de la liaison standard Denavi Hartenberg **37**

Figure II.10 : Représentation des repère associer a chaque articulation ……..… **41**

Figure II.11 : Représentation géométrique du robot RRR dans le plantridimensionnel ……..… **43**

Figure II.12 : La projection du robot sur le plan x-y.……..... **44**

Figure II.13 : Configuration initiale du bras …….. **46**

Figure II.14 : La simulation du modèle géométrique direct **……..…….. 48**
Figure II .15 : La simulation du modèle géométrique inverse**……..…… 48**

Chapitre III: Conception et réalisation du travail

Figure III.1 : Imprimante Creality 3-Ender PRO utiliser pour fabriquer les pièces du robot**…… 50**
Figure III.2 : Filament PLA high speed utiliser**……..…….. 51**
Figure III .3 : Interfce de logiciel slicer adapté pour l'imprémante ender-3pro**……. 52**
Figure III.4 : Assemblage de la base rotative**……. 54**
Figure III.5 : Fixation de l'articulation 1 sur la base **……. 55**
Figure III.6 : Montage de la dernière articulation **……. 56**
Figure III.7 : Vue d'ensemble après l'assemblage des articulations**…….. 56**
Figure III.8 : Vue finale d'assemblage mécanique et eléctronique du robot **……..……. 57**
Figure III.9: Logiciel de programmation Arduino IDE**……...…….. 58**
Figure III.10 : Déclaration des broches et des paramètres **…….……..…. 59**
Figure III.11 : Configuration dynamique des moteur pas à pas dans le code arduino **…..…..…… 60**
Figure III .12 : Teste individuelle de chaque articulation du bras **…….. 61**
Figure III.13 :L'ensemble des fonctions utiliser dans la programme de test**…….. 61**
Figure III.14 : Code MATLAB pour calcul les angles $\theta_1\theta_2\theta_3$**…….. 64**
Figure III.15 : La configuration matériel du robot **……..……. 64**
Figure III.16 : La partie de commande dans le code**……...…… 65**
Figure III.17 : l'interface de commande dans le code**…… 65**
Figure III.18 : La partie qui représente le mouvement vers les point disérer**…….. 66**
Figure III .19 : La représentation des fonctions **……..…….. 67**
Figure III.20 : Déformation des pièces imprimé **…….. 67**
Figure III.21 : Erreur thermistor brûlé suite à une surchauffe**……..…… 68**
Fidure III.22 : Casse mécanique au niveau des articulations**……..... 69**

Liste des tableaux

Chapitre I :Généralité sur les robots

Tableau I.1 : Caractéristiques du moteur pas à pas à aimant permanent…….. **19**
Tableau I.2 : Caractéristiques du moteur pas à pas à réluctance variable (VR) …….. **19**
Tableau I.3 : Caractéristiques du moteur pas à pas hybride…….. **20**
Tableau I.4 : Caractéristiques du moteur pas à pas unipolaires **20**
Tableau I.5 : Caractéristiques du moteur pas à pas bipolaires…….. **20**

Chapitre II : Étude du robot BCN3D Moveo et modélisation géométrique

Tableaux II.1 : Les piéces mécanique utiliser dans notre robot (BCN3D modifier). …….. **25**
Tableau II.2 : Les moteurs utilisés et leurs spécifications..... **26**
Tableau II. 3 : Récapitulatif des caractéristiques du DRV8825……..... **28**
Tableau II.4 : Les caractéristique des carte arduino les plus connu…….. **30**
Tableaux II.5 : Les paramètres DH du prototype…….. **41**

Chapitre III : Conception et réalisation du travail

Tableau III.1 : Les élément utiliser pour rassembler la base. …….. **53**
Tableau III.2 : Les élément utiliser pour rassembler l'épaule . ……..... **54**
Tableau III.3 : Les élément utiliser pour rassembler le coude. ……..... **55**

Introduction générale

I.Introduction générale

Depuis plusieurs années, la robotique ne cesse d'évoluer et de prendre une place de plus en plus importante dans notre quotidien, que ce soit dans l'industrie, les hôpitaux, ou même nos foyers. Ce domaine fascinant rassemble plusieurs disciplines comme la mécanique, l'électronique et l'informatique, et il a pour objectif de concevoir des machines capables d'interagir avec leur environnement et de réaliser des tâches de manière autonome.

Dans ce contexte, nous avons choisi de travailler sur un bras robotique dans le cadre de notre projet de fin d'études. Pourquoi ce choix ? Parce que le bras manipulateur est un exemple concret et riche pour appliquer nos connaissances théoriques en les mettant en œuvre dans une réalisation réelle. C'est aussi un excellent moyen de comprendre les défis que pose la commande de mouvements précis à travers la modélisation et la programmation.

Notre projet porte sur la conception et la mise en œuvre d'un bras robotique open source, basé sur le modèle BCN3D Moveo. Le but est simple mais ambitieux : permettre au robot de déplacer un objet d'un point A vers un point B, en utilisant la cinématique géométrique inverse. Pour structurer notre travail, nous avons organisé cette mémoire en trois grandes parties :

Dans le premier chapitre, nous faisons un tour d'horizon du monde de la robotique : ses différentes catégories, ses domaines d'application, et les composants qu'on retrouve généralement dans un robot.

Le deuxième chapitre est plus ciblé on parle du bras robotique BCN3D Moveo, en décrivant ses pièces, son architecture, et les méthodes de modélisation utilisées, notamment la cinématique directe et inverse.

Enfin, dans le troisième chapitre, on passe à la réalisation pratique. C'est là qu'on détaille les étapes de fabrication, le choix des composants, le montage du robot, et la programmation qui lui permet d'exécuter ses mouvements selon les calculs de position.

Ce projet nous a permis de relier la théorie à la pratique, de comprendre les limites d'un prototype réel, et de réfléchir à des pistes d'amélioration pour aller encore plus loin.

Chapitre I : Généralité sur les robots

1. Introduction

La robotique représente l'une des avancées technologiques les plus marquantes de notre époque. Elle désigne l'ensemble des techniques et des sciences permettant de concevoir, programmer et contrôler des machines capables d'exécuter des tâches de manière autonome ou semi-autonome.

Depuis l'Antiquité jusqu'à aujourd'hui, le concept de robot a évolué en passant de simples automates mécaniques à des systèmes intelligents capables d'interagir avec leur environnement. Grâce aux progrès en électronique, en informatique et en automatique, les robots modernes sont désormais présents dans de nombreux secteurs d'activité.

Leur adoption croissante s'explique par les nombreux avantages qu'ils offrent en termes de productivité, de sécurité, de précision et de réduction de la pénibilité du travail. Qu'il s'agisse de robots industriels, médicaux, mobiles ou humanoïdes, leur impact est devenu incontournable.

Dans ce chapitre, nous allons étudier la robotique de manière générale, en présentant ses définitions fondamentales, ses composants principaux, ses différentes catégories ainsi que ses domaines d'application. Nous aborderons également les principes de base de l'asservissement et de la commande, qui sont essentiels pour comprendre les chapitres suivants.

2. L'histoire de la robotique

Depuis la nuit des temps, l'un des rêves fondamentaux de l'humanité a été de créer une machine intelligente à son image. Ce rêve, longtemps relégué au domaine de l'imaginaire, est aujourd'hui devenu une réalité concrète dans notre monde contemporain. Le concept de robot s'est construit à travers de nombreuses réalisations historiques qui ont su allier imagination scientifique et avancées technologiques**.**

La robotique trouve ses origines dans les premières tentatives de l'humanité à concevoir des machines capables d'imiter les fonctions humaines. Ce désir ancestral s'est transformé, au fil des siècles, en une discipline scientifique et technologique à part entière. Le terme *robot* a été introduit au début du XXe siècle pour désigner des entités artificielles conçues pour assister ou remplacer l'homme dans certaines tâches. Les premières applications concrètes de la robotique industrielle apparaissent dans les années 1950, notamment dans le secteur automobile, où les robots sont utilisés pour manipuler des charges lourdes avec une précision constante. [1]

Avec le développement des systèmes de commande et de l'intelligence embarquée, la robotique a progressivement évolué vers des machines capables de percevoir leur environnement

et d'y réagir de manière autonome. Un robot moderne est défini comme un système mécanique polyarticulé, équipé d'actionneurs et de capteurs, contrôlé par un calculateur, capable d'exécuter des tâches variées avec une grande précision, répétabilité et sans fatigue. Cette définition met en lumière l'écart croissant entre les capacités humaines et celles des machines, notamment en termes de force, d'endurance et de régularité. [5]

Les avancées récentes en modélisation géométrique, commande adaptative et planification de trajectoire ont permis l'essor des robots mobiles et manipulateurs, aussi bien dans l'industrie que dans des environnements dynamiques et complexes. Des travaux académiques réalisés dans plusieurs universités ont contribué à simuler, concevoir et tester des systèmes robotiques autonomes capables de naviguer sans intervention humaine, renforçant ainsi le rôle de la robotique dans les domaines critiques tels que la médecine, la sécurité, l'agriculture et l'exploration spatiale. [7]

3. Défénition du robot

Un robot est un appareil automatique capable de manipuler des objets ou d'exécuter des opérations selon un programme fixe, modifiable ou adaptable[14] doté des capteurs et d'effecteurs lui donnant une capacité d'adaptation et de déplacement proche de l'autonomie pour exécuter ces mouvement sans fatigue, en suivant le même chemin, avec haute précision qui reste inchangée et avec la même finesse, dépasse les capacités d'un être humain (précision, répétabilité, fatigue, force, ...)[5].On peut aussi le définir comme un système mécanique polyarticulé mû par des actionneurs et commandé par un calculateur qui estdestiné à effectuer une grande variété de tâches . [14]

Les compossants d'un robot

- Structure mécanique : articulations, degrés de liberté (DOF).
- Actionneurs : servomoteurs, moteurs DC, motoréducteurs
- Capteurs : position, force, vision, ultrason.
- Contrôleur : microcontrôleur ou API (ex : Arduino).
- Logiciel/contrôle : cinématique directe/inverse, trajectoire, asservissement (PID, sans capteur)
- Interface utilisateur : GUI, télécommande, etc.
- effecteur dans le cas du bras robot .

Les différentes catégories des robots

- Robots mobiles (les robots mobiles à roues, sous marins, volants, humanoïdes ..).
- Robots manipulateurs .

3.1 Robots mobiles

Un robot mobile est un système automatique-mécanique capables de se déplacer dans un environnement. Ils sont équipés ou non de manipulateurs suivant leur utilisation, il peux effectuer une ou plusieurs tâches dans un certain environnement, indépendamment, en mettant en œuvre un programme, [3] les robots mobiles peuvent être "auto-contrôlés" (robot mobile autonome), ce qui signifie qu'ils sont capables de naviguer dans un environnement incontrôlé sans avoir besoin de routeurs physiques ou électromécaniques[6]. Alternativement, les robots mobiles peuvent s'appuyer sur des routeurs qui leur permettent de naviguer sur un chemin de navigation prédéterminé dans un espace relativement contrôlé.

Différents types de robot mobile

- les robots mobiles à roues**. Figure I.1**
- Robot sous marins**. Figure I.2**
- Robot volants**.Figure I.3**
- Robot humanoïdes .**Figure I.4**

Figure I. 1 : Robot a roues

Figure I.2 : Robot sous marins.

Figure I. 3 : Robot volants.

Figure I. 4 : Robot humanoïde.

Domaine d'utilisation

Les robots mobiles sont largement utilisés dans de nombreux secteurs grâce à leur capacité à se déplacer de manière autonome ou télécommandée, ce qui leur permet d'intervenir dans des environnements parfois dangereux ou difficiles d'accès.

- Dans l'industrie nucléaire, ils sont essentiels pour surveiller des sites sensibles, manipuler des matériaux radioactifs et participer au démantèlement sécurisé des centrales.[16]
- Dans le domaine de la sécurité civile, ils permettent de neutraliser des activités terroristes, de procéder au déminage ou à la pose d'explosifs, et d'assurer la surveillance de zones à risque comme les entrepôts de munitions. Sur le plan militaire, ils sont utilisés pour des missions de surveillance, de patrouille, et pour la manipulation sécurisée de munitions ou d'explosifs. [7] [14]
- Dans l'industrie chimique, ils contribuent à la surveillance des installations et à la manipulation de substances toxiques. En médecine, ils assistent les secours d'urgence, soutiennent la mobilité des personnes handicapées et offrent une aide précieuse aux malvoyants ou aux aveugles. [8]
- Dans la lutte contre l'incendie, les robots localisent les foyers, détectent la fumée et participent à l'extinction ciblée des flammes. En milieu sous-marin, ils servent à poser des câbles, à explorer les fonds marins, à rechercher des nodules métalliques ou à localiser des navires immergés. [16]
- Dans le secteur agricole, ils assurent la récolte des fruits et légumes, la traite automatisée du bétail, la pulvérisation de produits et l'entretien des cultures comme la vigne. Le secteur du bâtiment bénéficie également de leur capacité à projeter du mortier ou à lisser automatiquement le béton. En nettoyage industriel, les robots interviennent sur les coques de navires ou dans de grandes surfaces difficiles à atteindre.

- Dans le domaine spatial, ils sont employés pour l'exploration planétaire et la collecte de données dans des environnements hostiles. Enfin, dans l'industrie en général, ils assurent le convoyage automatique de matériaux et la surveillance des lignes de production, contribuant ainsi à améliorer l'efficacité, la sécurité et la productivité. [8]

3.2 Les robots manipulateurs

Les robots manipulateurs sont des systèmes automatisés fixés de manière permanente à leur poste de travail. Ils sont conçus pour exécuter des tâches précises et répétitives, remplaçant ainsi l'intervention humaine dans des opérations souvent monotones, exigeantes ou nécessitant une grande précision. Qu'il s'agisse de transfert d'objets, de soudage, d'assemblage ou d'autres applications industrielles, un robot manipulateur est un mécanisme capable de déplacer et positionner un outil, appelé organe terminal ou point outil, dans une zone définie de l'espace appelée volume de travail [10]. La structure d'un robot manipulateur repose sur une architecture mécanique animée par des actionneurs, eux-mêmes contrôlés par un calculateur. Ce dernier génère les ordres de mouvement à partir des informations fournies par divers capteurs. Parmi ces capteurs, les capteurs extéroceptifs, qui permettent d'évaluer l'interaction entre le robot et son environnement via l'organe terminal, jouent un rôle croissant, notamment dans les applications à haute précision telles que la micro-manipulation ou la chirurgie assistée par robot[6][7].

Grâce à cette combinaison de structure rigide, de commande intelligente et de perception par capteurs, les robots manipulateurs occupent aujourd'hui une place essentielle dans l'automatisation avancée, aussi bien dans l'industrie que dans le secteur médical.

Types des robots manipulateurs

Classification géométrique

Les robots manipulateurs viennent sous plusieurs formes, adaptées aux différents besoins industriels tell que des robot manipulateur série (chaine ouvert)et parallèle (chaine fermé).

Figure I.5: manipulateur série

Figure I.6 : manipulateur parallèle

La plus grand partie de ces manipulateur est constituée des manipulateur série, dont la morphologie peut facilment etre comparée à celle de l'etre humain .On les classe généralement en cinq grandes catégories selon leur configuration géométrique et leur cinématique : cylindriques, rectilignes (cartésiens), sphériques, articulés et SCARA. Voici une présentation détaillée de chaque type.[17]

1. Robots cylindriques

Ces robots possèdent une architecture basée sur des mouvements de translation verticale, de rotation autour de l'axe vertical et de translation radiale. Leur espace de travail est de forme cylindrique.

Exemples d'utilisation : manipulation de pièces dans des ateliers de mécanique, opérations d'empilage, pick-and-place vertical.

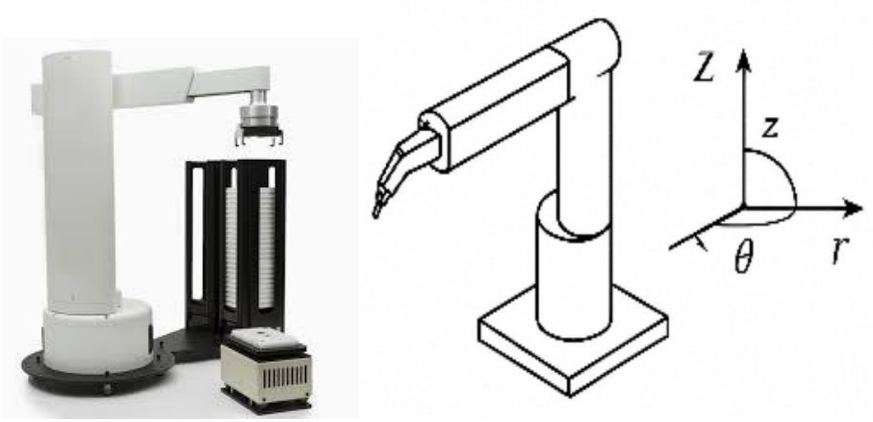


Figure I.7: Robots cylindriques

Avantage :

- Construction simple et robuste
- Espace de travail cylindrique idéal pour les tâches verticales
- Facilité d'intégration dans des lignes de production automatisées

2. Robots rectilignes

Ils se déplacent selon les trois axes linéaires X, Y et Z. Leur volume de travail est un parallélépipède rectangle. Ils offrent une très bonne précision et sont faciles à programmer.

Exemples d'utilisation : impression 3D, découpe CNC, lignes d'assemblage automatisées, pick-and-place horizontal.

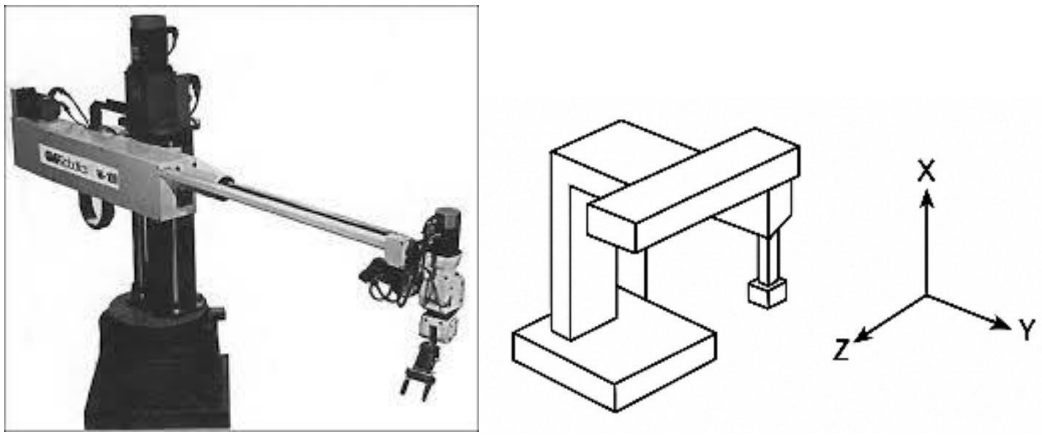


Figure I.8 : Robot rectilignes.

Avantage :

- Haute précision et répétabilité
- Idéal pour les environnements structurés
- Programmation simplifiée

3. Robots sphériques

Ce type des robots combinent une rotation autour de la base, un bras pivotant et une extension télescopique. Leur espace de travail est en forme de sphère partielle.

Exemples d'utilisation : manipulation de matériaux dans des environnements confinés, laboratoires automatisés.

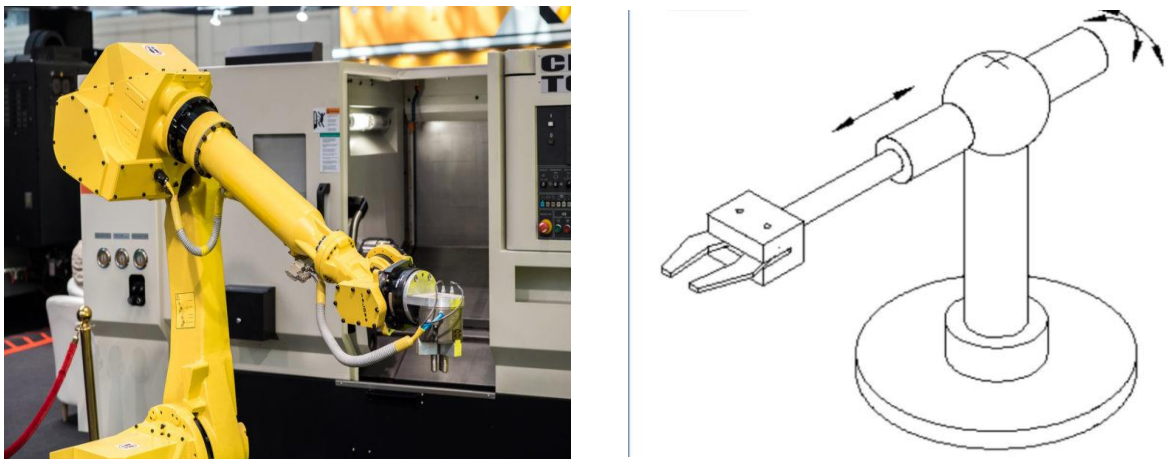


Figure I.9 : Robot sphérique

Avantage :

- Précision exceptionnelle
- Adaptabilité à des tâches variées
- Amélioration de l'efficacité et de la productivité

4. Robots SCARA

Ces robots sont rigides selon l'axe Z mais flexibles dans le plan XY, ce qui les rend idéaux pour des tâches d'assemblage rapide.

Exemples d'utilisation : insertion de composants électroniques, assemblage de petites pièces, emballage.

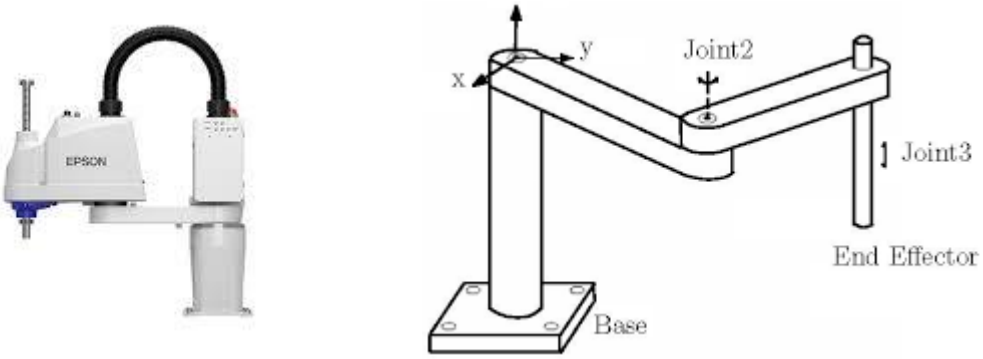


Figure I.10 : Robot SCARA.

Avantage :

- Fonctionnement à grande vitesse
- Excellente répétabilité

- Conception compacte adaptée aux espaces restreints

5 . Robots articulés

Ce sont les plus courants dans l'industrie. Ils sont composés de plusieurs segments reliés par des articulations rotatives, offrant une grande liberté de mouvement et un espace de travail étendu en forme de sphère.

Exemples d'utilisation : soudage, peinture industrielle, assemblage complexe, palettisation.

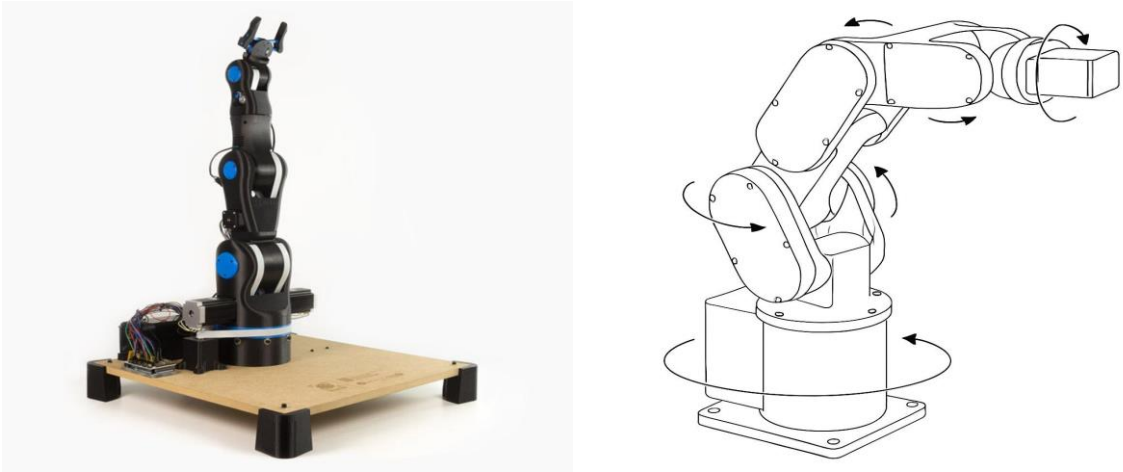


Figure I.11 : Robot articulés

Avantage :

- Polyvalent et adaptable
- Précis et exact
- Capable d'exécuter des mouvements complexes

Domaine d'utilisation des robots manipulateur

Les robots manipulateurs sont devenus des outils incontournables dans de nombreux domaines industriels et technologiques, grâce à leur capacité à exécuter des tâches répétitives avec précision, sécurité et efficacité. Ils sont largement utilisés :

- Dans l'industrie automobile pour les opérations d'assemblage, de soudage et de peinture, où la régularité et la rapidité sont cruciales.

- Dans le secteur pharmaceutique et médical, ils participent à la manipulation stérile de substances sensibles, à la préparation automatisée de doses, ainsi qu'à la chirurgie assistée par robot.
- En logistique, ils facilitent le conditionnement, le tri et le déplacement de colis dans les entrepôts. Dans le domaine de la métallurgie et de la transformation des matériaux, les robots manipulateurs manipulent des pièces lourdes ou dangereuses, parfois dans des environnements à haute température.
- Ils trouvent aussi leur place dans l'agriculture moderne, où ils peuvent récolter, trier ou emballer les produits agricoles.

Leur capacité à s'adapter à diverses tâches en fait des outils indispensables pour améliorer la productivité, réduire les erreurs humaines et intervenir dans des environnements risqués pour l'homme.[2]

3.3 Caractéristiques du bras robotique

Un bras robotique se distingue par plusieurs caractéristiques essentielles. Il est notamment capables de porter une certaine charge et d'exécuter des mouvements cycliques avec précision. Parmi ces caractéristiques, le nombre de degrés de liberté (DOF – Degrees of Freedom) constitue un paramètre fondamental. Chaque degré de liberté représente un axe ou un type de mouvement que le bras peut réaliser, en rotation ou en translation.

Dans le but de mimer les mouvements d'un bras humain, le bras robotique doit intégrer plusieurs articulations, chacune autorisant un type de mouvement particulier. Ainsi, plus le nombre de degrés de liberté est élevé, plus le bras est flexible et proche du comportement biomécanique humain

Les articulations du bras sont ici commandées par des moteurs pas à pas (stepper motors). Ces moteurs sont réputés pour leur capacité à effectuer des déplacements angulaires précis, maintenir des positions fixes sans rétroaction, et réagir de manière fiable aux impulsions de commande. Contrairement aux servomoteurs, les stepper motors ne nécessitent pas de capteurs internes pour connaître leur position actuelle, ce qui simplifie le contrôle dans certains systèmes ouverts.

3.4 Architecture du robot manipulateur

Le robot manipulateur est composé de plusieurs éléments mécaniques et électromécaniques agencés de manière hiérarchique, depuis la base fixe jusqu'à l'organe terminal. Voici la structure complète classée du bas vers le haut :

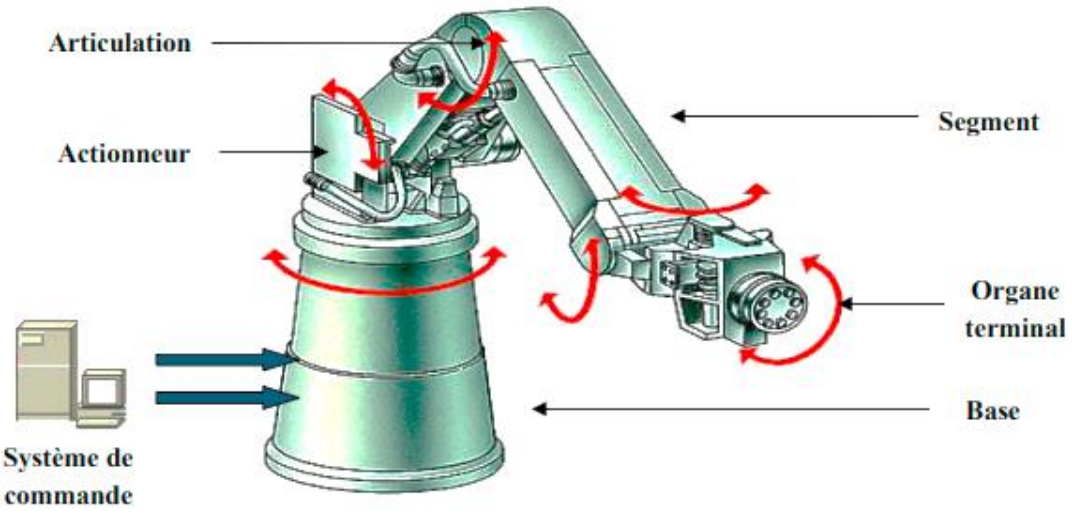


Figure I.12 : Architecture d'un bras robot

La base

C'est la partie inférieure du robot. Fixée au sol ou à une structure de travail, elle sert de support stable à tout le système. Elle ne bouge pas et joue le rôle de fondation.

Les articulations

Ce sont les points de connexion entre deux segments successifs. Elles permettent la rotation ou la translation. Leur rôle est de conférer au bras sa mobilité. Chaque articulation est généralement associée à un actionneur.[12]

- **Les articulation prismatiques :**Il s'agit d'une articulation de type glissière, notée P, réduisant le mouvement entre deux corps à une translation le long d'un axe commun. La situation relative entre les deux corps est mesurée par la distance le long de cet axe.

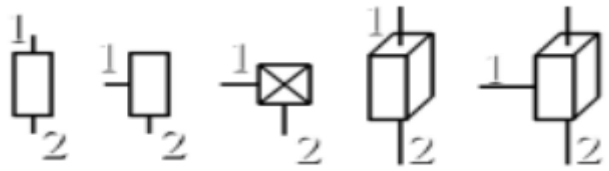


Figure I.13 : présentation s'une articulation prismatiques .

- **Les articulations rotoïdes :**Il s'agit d'une articulation de type pivot, notée R, réduisant le mouvement entre deux corps à une rotation autour d'un axe qui leur est commun. La situation relative entre les deux corps est donnée par l'angle autour de cet axe.

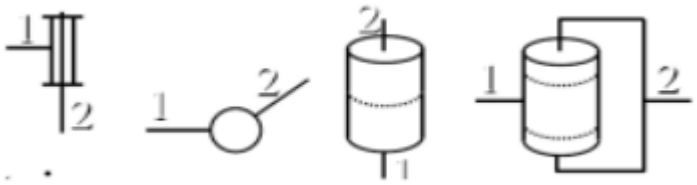


Figure 1.14 : présentation s'une articulation rotoïdes .

Les segments

sont les parties rigides qui relient les articulations. Ce sont les "os" du bras. Leur longueur et leur configuration déterminent la portée et l'agilité du bras manipulateur.

L'effecteur (organe terminal)

Fixé à la dernière articulation, l'effecteur est la partie fonctionnelle du bras. Il permet au robot d'interagir avec son environnement ,Exemples : pince (gripper), ventouse, outil de soudage, caméra, etc.. . Il peut être monofonctionnel ou multifonctionnel/interchangeable selon l'usage.

Les actionneurs

Fixés sur ou juste au-dessus de la base, les actionneurs sont les "muscles" du robot. Ils reçoivent des signaux de commande et créent le mouvement en agissant sur les articulations.

Types courants :Moteurs pas à pas ,Servomoteurs,Actionneurs pneumatiques ou hydrauliques

Les capteurs

Dans tout système automatisé, les capteurs jouent un rôle fondamental. Ils permettent de capter des informations physiques ,telles que la température, la position, la distance ou la lumière et de les convertir en signaux électriques utilisables par le système de commande [19]. Dans les systèmes mécatroniques notamment les robots, on utilise généralement ces types de capteurs :

- Les encodeurs : ils mesurent la rotation des moteurs et permettent de connaître la position exacte des articulations.
- Les capteurs de fin de course : détectent si une articulation a atteint sa limite physique
- Les capteurs de force ou de pression : utiles pour ajuster la force appliquée par un effecteur terminal comme une pince.

L'unité de commande

L'unité de commande est le cœur logique de tout les système mmécatronique Parmi les unités de commande les plus utilisées, la carte Arduino occupe une place centrale grâce à sa simplicité, sa flexibilité et son coût réduit. Arduino permet de recevoir des données depuis les capteurs, de traiter ces informations via un programme interne, puis d'envoyer des instructions aux moteurs ou autres composants. Ce fonctionnement est rendu possible grâce à son langage de programmation simple basé sur C/C++, et un environnement de développement facile à utiliser : l'Arduino IDE .[3][9]

3.5 Moteur pas à pas

Le moteur pas à pas est un moteur à courant continu sans balais dans lequel la rotation est divisée en un certain nombre de pas qui résultent de la structure du moteur. Généralement, une rotation d'arbre complète de 360° est divisée en 200 pas, ce qui signifie qu'un seul pas d'arbre est effectué tous les 1,8°. Il existe également des moteurs où la course de l'arbre est effectuée tous les 2 ; 2,5 ; 5, 15 ou 30°.[18]

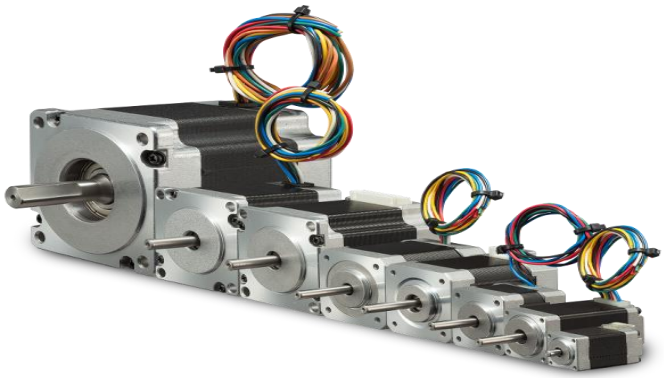


Figure I.15 : défférente types de moteur pas à pas .

Fonctionnement d'un moteur pas à pas

Un moteur pas à pas est constitué de deux parties principales :

- Le stator, fixe, composé de bobines (électroaimants) réparties autour du rotor.
- Le rotor, mobile, souvent muni d'aimants ou de dents en acier, fixé sur un arbre tournant.

Lorsque le courant traverse les bobines du stator, celles-ci génèrent un champ magnétique qui attire certaines parties du rotor. En alimentant les bobines successivement, le rotor se déplace par petits angles appelés "pas". En contrôlant précisément ces impulsions, on peut faire tourner le moteur dans un sens ou dans l'autre, avec une grande précision.Ce fonctionnement permet : des mouvements en pas complets (chaque bobine s'active à tour de rôle),en demi-pas (alternance entre une et deux bobines activées),ou en micropas, pour un mouvement encore plus fluide et précis grâce à une variation fine du courant dans les bobines.Ainsi, le moteur pas à pas convertit des impulsions électriques en mouvements angulaires précis, ce qui le rend idéal pour les applications de positionnement.

1) Fonctionnement en pas complet

En mode pas complet, le moteur se déplace par incréments fixes, généralement de 1,8° par pas, ce qui correspond à 200 pas pour un tour complet (200 × 1,8° = 360°).Chaque pas est réalisé en alimentant une ou deux bobines du stator :

- En activant une seule bobine, la consommation d'énergie est minimale.
- En activant deux bobines simultanément (mode biphasé), le moteur développe plus de couple et de stabilité, mais consomme davantage d'énergie.

Ce mode assure un bon compromis entre précision, couple et simplicité de commande.

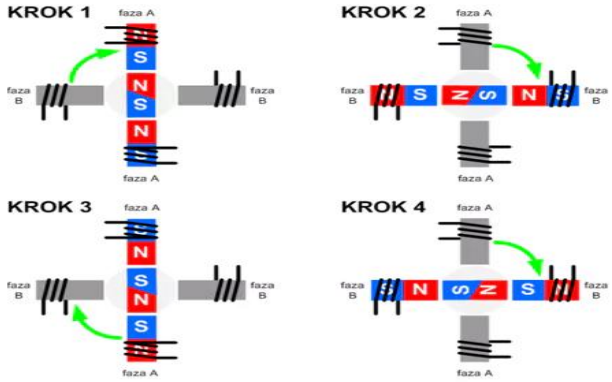


Figure I.16 : Principe de fonctionnement du moteur pas à pas en pas complet avec alimentation biphasée.

2) Fonctionnement en mode demi-pas

En mode demi-pas, chaque déplacement du rotor correspond à la moitié d'un pas complet, soit 0,9° au lieu de 1,8°, ce qui donne 400 pas pour un tour complet.

Ce mode alterne l'alimentation d'une seule bobine, puis de deux bobines simultanément, ce qui permet :

- une meilleure résolution angulaire (précision doublée) .
- un mouvement plus fluide.
- un couple légèrement supérieur à celui du mode monophasé.
- C'est un bon compromis entre précision, couple et régularité de mouvement.

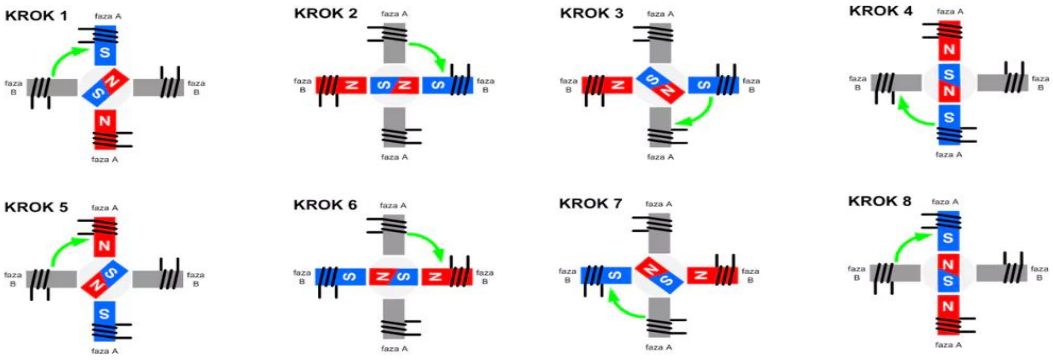


Figure I.17 : Principe de fonctionnement du moteur en mode demi-pas avec une alimentation biphasée.

3) Fonctionnement en mode micropas

Le mode micropas divise chaque pas en fractions encore plus petites qu'en demi-pas, jusqu'à 256 micro-divisions par pas complet. Cette finesse est obtenue en modulant

progressivement le courant dans les bobines, créant un champ magnétique lisse qui positionne le rotor avec une très haute précision.Ce mode est idéal pour les applications exigeant :

- des mouvements très fluides
- une grande précision de positionnement.

Cependant, il faut tenir compte d'un effet important : à haute vitesse, la réactance des bobines augmente avec la fréquence de commutation, ce qui réduit le courant moyen, donc le couple moteur. Cela peut entraîner :

- des pertes de pas.
- des arrêts involontaires.
- des oscillations.

Il est donc essentiel de consulter la courbe couple/fréquence du moteur dans sa fiche technique pour garantir un fonctionnement fiable en mode micropas.

Types des moteurs pas à pas

Les moteurs pas à pas, bien que mécaniquement simples comparés aux moteurs à courant continu à balais, nécessitent une réalisation précise et un contrôle rigoureux. Ils se rapprochent des moteurs BLDC modernes dans leur fonctionnement, notamment en termes de commande.[15] Ils se classent principalement selon :

- leur construction interne.
- le nombre de phases nécessaires à l'alimentation.

Ces critères influencent directement leur précision, leur couple et leur domaine d'application, les tableaux ci-après synthétisent les caractéristiques techniques des principales catégories de moteurs pas à pas, en comparant leur structure, performance et usage..

a) Moteur à aimant permanent (PM - Permanent Magnet)

Type de rotor	Aimant permanent
Type de bobinage	Unipolaire ou bipolaire selon les versions
Pas typique :	entre 7,5° et 15° (soit 24 à 48 pas par tour)
Nombre de phases	2 phases
Couple	Élevé à basse vitesse

Précision	Faible
Réponse dynamique	Lente
Coût	Faible
Applications :	dispositifs simples ne nécessitant pas une grande précision.

Tableau I.1 : Caractéristiques du moteur pas à pas à aimant permanent

b) Moteur à réluctance variable (VR - Variable Reluctance)

Type de rotor	Fer doux avec dents, sans aimant permanent
Type de bobinage	Multi-phases (souvent 3 à 4), sans prise médiane
Pas typique :	Variable selon la construction (souvent entre 5° et 15°)
Nombre de phases	3 à 4 phases
Couple	Faible
Précision	Moyenne à faible
Réponse dynamique	Moyenne
Coût	Très faible
Applications	Applications pédagogiques, démonstrations, anciens systèmes

Tableau I.2 : Caractéristiques du moteur pas à pas à réluctance variable (VR)

c) Moteur hybride (Hybrid Stepper Motor)

Type de rotor	Aimant permanent + dents
Type de bobinage	Unipolaire ou bipolaire (souvent bipolaire pour meilleure performance)
Pas typique :	Entre 0,9° et 3,6° (soit 400 à 100 pas par tour)
Nombre de phases	2 phases
Couple	Élevé
Précision	Haute
Réponse dynamique	Excellente
Coût	Moyen à élevé
Applications	Impression 3D, CNC, robotique, automatisation industriell.

Tableau I.3 : Caractéristiques du moteur pas à pas hybride.

d) Moteurs unipolaires

Type de rotor	Aimant permanent (similaire aux moteurs PM)
Type de bobinage	Avec prise médiane (centre tap)
Pas typique :	En général 1,8° (soit 200 pas par tour)
Nombre de phases	2 phases
Couple	Moyen (plus faible qu'un moteur bipolaire équivalent)
Précision	Moyenne
Réponse dynamique	Moyenne
Coût	Faible
Applications	Imprimantes, lecteurs CD/DVD, petits robots, scanners

Tableau I.4 : Caractéristiques du moteur pas à pas unipolaires .

e) Moteurs Bipolaires

Type de rotor	Aimant permanent (souvent hybride)
Type de bobinage	Sans prise médiane (bobines pleines)
Pas typique :	1,8° ou 0,9° selon la précision (200 à 400 pas par tour)
Nombre de phases	2 phases
Couple	Élevé
Précision	Haute
Réponse dynamique	Excellente
Coût	Moyen à élevé
Applications	CNC, imprimantes 3D, robotique, équipements médicaux, automatisation industrielle

Tableau I.5 : Caractéristiques du moteur pas à pas bipolaires.

4. Conclusion

Dans ce chapitre, nous avons donné un aperçu général de la robotique en découvrant ses notions fondamentales et les défférents domaines d'utilisation, ainsi que les différentes catégories

de robots. Une attention particulière a été accordée aux robots manipulateurs, qui constituent la base de nombreuses applications industrielles moderne et de comprendre les éléments essentiels qui composent un robot,ainsi que son fonctionement global. Nous avons ensuite concentré notre attention sur les moteurs pas à pas, qui sont essentiels au fonctionnement des robots manipulateurs en raison de leur précision et de leur commande en boucle ouverte. Cette base théorique constitue une étape essentielle pour aborder, dans le chapitre suivant, l'étude spécifique du robot manipulateur que nous avons conçu et développé dans le cadre de notre projet .

Chapitre II : Étude du robot BCN3D Moveo et modélisation géométrique

1. Introduction

Dans un contexte de développement rapide dans le domaine de l'automatisation et de la robotique, les bras robotiques sont devenus l'un des systèmes les plus essentiels, tant dans les applications industrielles qu'éducatives, en raison de leur capacité à reproduire les mouvements humains avec une grande précision et flexibilité. C'est dans ce cadre que s'inscrit notre projet, qui consiste en la conception et la réalisation d'un prototype modifier(3 DOF) de bras robotique open source de type BCN3D Moveo, disposant de trois degrés de liberté ainsi qu'un effecteur terminal (End Effector), basé sur des moteurs pas-à-pas .

Le projet est open-source, ce qui signifie que toutes les données nécessaires à la fabrication du robot sont accessibles au public. Cela permet à quiconque de construire, modifier et partager librement les améliorations du robot.

Ce chapitre s'appuie sur la conception mécanique (tell que les pièces mécaniques ont été fabriquées à l'aide de procédés d'impression 3D, ce qui permet de réduire les coûts et de faciliter les ajustements ou reproductions du système.),et électronique ainsi que l'implémentation de techniques de contrôle du mouvement et la modélisation géométrique et dynamique de notre robot .

2. Présentation du robot BCN3D Moveo

Le robot BCN3D Moveo est un bras manipulateur open-source développé initialement par BCN3D Technologies. Il est conçu pour l'éducation, la recherche et les applications d'apprentissage en robotique. Le robot est principalement fabriqué à l'aide de pièces imprimées en 3D et utilise des composants standards tels que des moteurs pas à pas (NEMA 17, NEMA 23) et commander par un carte Arduino Uno. Il dispose de 4 à 5 degrés de liberté, selon la version, et permet de simuler les mouvements d'un bras industriel classique.[16]

2.1. Architecture mécanique et composants

Le robot BCN3D Moveo est constitué de plusieurs segments reliés entre eux par des articulations. Sa structure comprend :

- Une base fixe
- Une épaule motorisée (axe 1 – rotation horizontale)
- Un bras supérieur (axe 2 – élévation verticale)
- Un avant-bras (axe 3 – extension du bras)
- Un poignet (axe 4 – orientation de l'outil)

- Un effecteur terminal (gripper ou pince, axe 5)

Chaque articulation est animée par un moteur pas à pas contrôlé par un carte arduino .

Figure II.1 : La strcture complet de bras robotique BCN3D

3. L'impression 3D

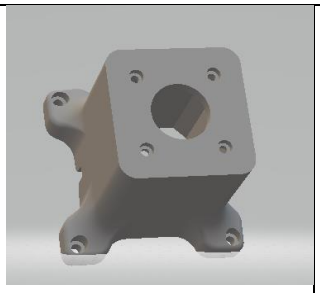
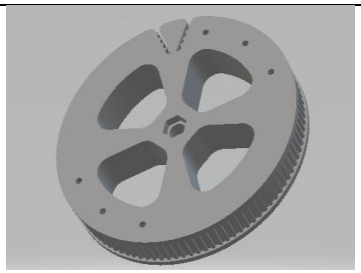
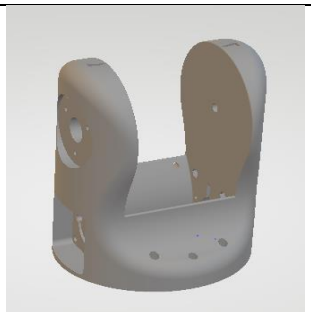
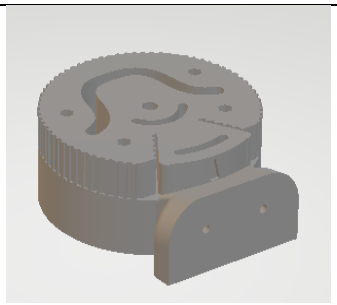
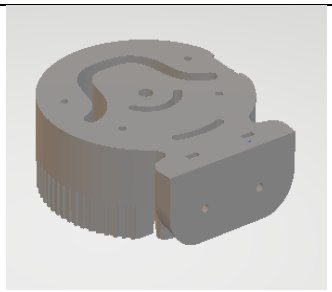
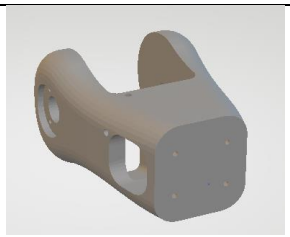
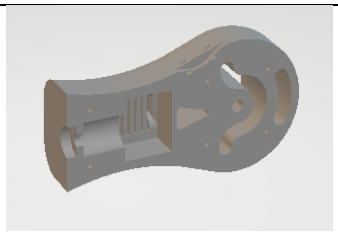
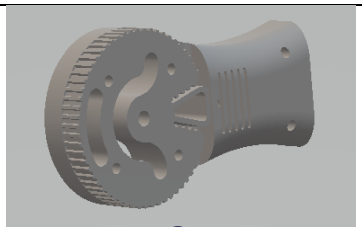
Lors de la fabrication d'un prototype robotique, l'importance de l'impression 3D des éléments conçus est primordiale. Ceci contribue à diminuer les dépenses et à accélérer la fabrication. Deux aspects étaient considérés : la sélection de la technologie et du matériau, l'infrastructure existante pour accomplir cette mission ainsi que les caractéristiques mécaniques désirées.la figure 2.1 présente les différentes parties du robot BCN3D imprimer en 3D :

Figure II.2 : Les différentes parties du robot BCN3D imprimer en 3D.

4. Description mécanique et éléctronique des parties constitutives du bras

4.1. Les défférentes pièces utiliser dans notre projet

Les défférent pièces présenter dans le tableau suivant **:**

La base	 Support de moteur de base Quantité : 1	Base rotative Quantité : 1	 Plaque rotative Quantité : 1
L'épaule	 Base d'épaule Quantité : 1	 Articulation de l'épaule Quantité : 1	 Articulation de l'épaule Quantité : 1
Le coude	 Base coudée Quantité : 1	 ½ articu du coude Quantité:1	 ½ articu du coude Quantité:1

Tableaux II.1 : les piéces mécanique utiliser dans notre robot (BCN3D modifier).[16]

4.2. Les moteurs

Le bras robotique BCN3D Moveo est conçu pour accueillir jusqu'à 6 moteurs, répartis sur ses différentes articulations. Bien qu'il dispose de 5 degrés de liberté (5 DOF), chaque articulation est mise en mouvement à l'aide d'un système de poulies et de courroies, garantissant une transmission fluide et un meilleur contrôle de la position de l'outil terminal. Cette configuration

mécanique permet d'optimiser la précision des déplacements tout en réduisant les contraintes directes sur les moteurs.

Pour contrôler les moteurs pas à pas, nous avons choisi des drivers de type DRV8825, réputés pour leur précision en micro-pas et leur capacité à gérer des courants élevés. Ces drivers sont reliés à un CNC Shield V3, une carte d'extension spécialement conçue pour l'Arduino Uno, permettant de piloter plusieurs moteurs de manière simultanée et organisée. Ce choix technique facilite l'architecture du système tout en garantissant un contrôle efficace et évolutif.

Les moteurs utiliser dans notre travaille

Nous allons étudier uniquement les trois premières articulations du robot(3DOF). Ce sont les différents moteurs que nous avons utilisés dans notre réalisation :

Moteur	Ampirage(A)	Torque(Kg .cm)
Nema 17 SM42HT47-1684	1.3	3.6
Nema 23 SM57HT76-3004A	3	20.29
Nema 17 SM42HT33- 1334A	1.33	2.2

Tableau II.2 : les moteurs utilisés et leurs spécifications.

- **Nema 17 SM42HT47- 1684 pour l'articulation 1**

SM42HT47-1684 Nema 17 : Ce moteur pas à pas hybride bipolaire a un angle de pas de 1,8° (200 pas par tour). Chaque phase utilise 1.6 A à 2.8 V,permettant ainsi un couple de Conservation de 3,6 kg-cm.

- **Nema 23 SM57HT76-3004A pour l'articulation 2**

Le moteur Nema 23 SM57HT76-3004A possède des dimensions de 57x57x76 mm. Il offre un couple de maintien de 19,27 kg.cm (267 oz.cm ou 1,89 N.m) et comporte 200 pas par révolution, avec un pas angulaire de 1,8°. Ce moteur consomme un courant de 3A par phase. Il constitue une solution efficace pour les applications où l'espace est limité, tout en nécessitant une faible vitesse ou un couple élevé.

- **Nema 17 SM42HT33- 1334A pour l'articulation 3**

Le moteur Nema 17 SM42HT33-1334A possède des dimensions de 42x42 mm. Il comporte 200 pas par révolution, avec un pas angulaire de 1,8°, et consomme généralement entre 1 et 2A par phase.

Les drivers

Un driver est un dispositif électronique qui gère l'actionnement d'un moteur en particulier un moteur pas à pas. Il agit comme un intermédiaire intelligent entre la carte de commande (Arduino) et le moteur lui-même. Le rôle principal du driver et de transmettant des impulsions électriques spécifiques aux bobines du moteur. Il convertit les signaux de commande (tels que les impulsions) en déplacements angulaires du moteur, autorisant ainsi un positionnement exact et maîtrisé. [9]

Les principaux drivers utilisés avec les moteurs pas à pas :

- A4988
- DRV8825
- TMC2208
- TMC2209

DRV8825:le type utiliser dans notre réalisation

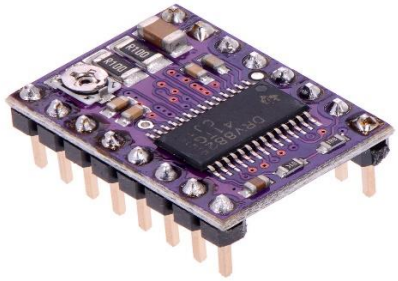


Figure II.3: Vue physique du driver DRV8825 [11]

Caractéristique	**Valeur / Description**
Microstepping	Jusqu'à 1/32 de pas
Courant maximal / phase	2,5 A (avec refroidissement) ; environ 1,5 A continus sans refroidissement
Tension d'alimentation	8,2 V à 45 V pour VMOT
Tension logique	Entre 2,5 V et 5,25 V
Protection intégrée	Shutdown thermique, détection surcourant, protection contre UVLO, broche Nfault
Consommation en veille	Très faible (courant de repos faible)
Dimensions du module	Environ 15,5 × 20,3 mm
Interface	Commande STEP/DIR standad(compatible ardino,CNC …)

Tableau II. 3 : récapitulatif des caractéristiques du DRV8825 [11]

CNC Sheild:

Cette carte d'extension, en tant que tel, peut servir de carte d'extension pour les pilotes et peut être utilisée avec des machines à graver ou des imprimantes 3D.

On compte quatre emplacements disponibles, capables de piloter quatre moteurs pas à pas. Chaque moteur pas à pas de route nécessite simplement deux ports IO. Autrement dit, trois moteurs pas à pas peuvent être correctement gérés par six ports d'entrée/sortie. Très pratique à utiliser.

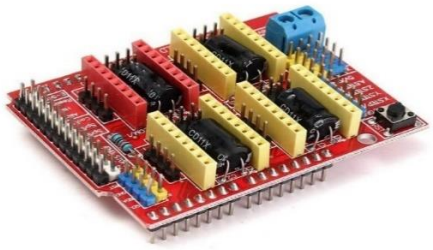


Figure II .4 : vue physique du CNC sheild .

4.3. L'arduino

C'est une carte électronique de développement qui inclut un circuit imprimé open source avec un microcontrôleur ATMEL intégré. Elle est conçue pour simplifier l'emploi de l'électronique interactive dans divers projets

Les avantages d'Arduino

- Coût de la licence.
- Simple à utiliser.
- Clarté du langage de programmation.
- Large éventail d'extensions disponibles.
- Accès à de nombreuses bibliothèques pour la plupart des extensions.
- Open source, ce qui facilite son évolution.
- Capable d'interagir avec des langages de programmation puissants comme JAVA, MATLAB et VB. NET.

Les défférents types des cartes arduino

Les cartes Arduino varient en fonction de leur format et du nombre de broches qu'on peut utiliser. Il existe plusieurs types de cartes Arduino, chacune possédant ses spécificités et domaines d'application. L'Arduino Uno, l'Arduino Nano et l'Arduino Mega sont trois modèles populaires. L'Uno est le plus utilisé et polyvalent, le Nano est plus compact tandis que le Mega offre davantage de mémoire et de broches.le tableau au dessous présente les principaux modèle d'arduino ,ainsi que les caractéristique de chaque modèle :

Type de Carte	Microcontrôleur Principal	Mémoire Flash (Ko)	Broches Numériques E/S (dont PWM)	Broches Analogiques E/S	Caractéristiques Spécifiques
NANO	ATmega328P ATmega628 SAMD21 RP2040	32 (Nano Every Nano 33 IoT) 256 (Nano 33 BLE Sense) 8 (Nano) 16 (Nano Every)	14 (Nano Every) 22 (Nano 33 BLE Sense)	8 (Nano Every) 8 (Nano 33 BLE Sense)	Très compacte, certains modèles incluent Wi-Fi/Bluetooth, capteurs embarqués (température, humidité, pression, geste, microphone), supporte MicroPython et ML, connexions similaires à l'UNO.
UNO	ATmega328P	32	14 (dont 6 PWM)	6	La plus courante, idéale pour les débutants, utilise l'Atmega16U2 pour un transfert rapide et plus de mémoire .
Mega	ATmega2560	256	54 (dont 14 PWM)	16	Grande capacité de mémoire et de broches d'E/S, idéale pour les projets complexes.

Tableau II.4 : les caractéristique des carte arduino les plus connu.

Structure de la carte arduino : se compose

- D’une entrée USB pour y télé verser un programme
- Une alimentation 9V pour fournir la tension nécessaire à la carte
- Un bouton reset pour relancer un programme
- Sortie d’alimentation en 3.3 ou 5V pour alimenter
- D’une entrée et sortie logique pour envoyer ou recevoir des informations
- 2 masses

5. La modélisation

5.1. Espace de travail

L'espace de travail d'un robot correspond à l'ensemble des positions que son effecteur final peut atteindre. Il dépend de la configuration du robot (par exemple : RRR, RRP), de la longueur de ses segments et des amplitudes autorisées pour les angles de rotation.[19]

L' analyser de cette espace permet de mieux appréhender les limites physiques du système, de prévenir les collisions et d'améliorer l'implantation du robot dans son environnement.

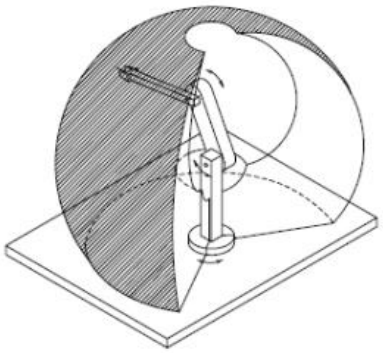


Figure II.5 : Espace de travail d'un bras manipulateur.

5.2. Modélisation des robots

La première question que l'on se pose lorsqu'on travaille avec un robot est la suivante il s'agit de savoir où se situera l'outil (ou l'effecteur terminal) du robot lorsque chaque articulation adopte un angle donné ou une certaine extension

La conception et la commande des robots impliquent le calcul de certains modèles mathématiques permettant de faire le lien entre les différents espaces de représentation du mouvement., tels que :

- l'espace opérationnel, où est définie la position et l'orientation de l'organe terminal (effecteur),
- l'espace articulaire, où est décrite la configuration du robot à travers les angles ou positions de ses articulations.

Parmi ces modèles, on distingue principalement :

5.2.1. Les modèles géométriques direct et inverse

Ces modèles permettent de déterminer soit la position et l'orientation de l'organe terminal à partir de la configuration articulaire (modèle direct),soit les valeurs articulaires nécessaires pour atteindre une position et orientation données dans l'espace (modèle inverse).

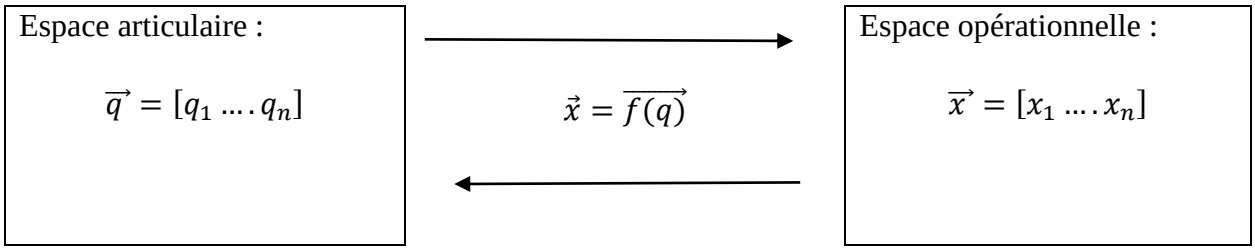


Figure II.6 : modéèle géométrique direct et inverse .

5.2.2. Les modèles cinématiques direct et inverse

Ils décrivent la relation entre les vitesses des articulations et la vitesse linéaire et angulaire de l'effecteur final :

- **Le modèle cinématique direct** permet de calculer la vitesse cartisienne d'un point généralement de l'organe terminal à partir des vitesses articulaires des joints .pour établire la relation entre les deux vitesse on utilise la matrice Jacobian :

$$dx = J(q)dq \qquad \text{(II.1)}$$

Ce qui done :

$$J(q) = \frac{\partial x}{\partial q} \qquad \text{(II.2)}$$

Le modèle CI devient :

$$\dot{X} = J(q).\dot{q} \qquad \text{(II.3)}$$

D'où :

$\dot{X}$: et la vitesse du l'outile terminale

$\dot{q}$: la vitesse de l'articulation .

L'équation dovient :

$$v = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ w_x \\ w_y \\ w_z \end{bmatrix} = \begin{bmatrix} v_x \\ v_y \\ v_z \\ w_x \\ w_y \\ w_z \end{bmatrix} = J(q).\dot{q} \quad \text{(II.4)}$$

Avec :

(v_x , v_y , v_z) représente la vitesse linéaire de l'effecteur terminal.

(w_x , w_y , w_z) représente la vitesse angulaire autour des axes x,y,z .

- **Le modèle inverse fait l'inverse :** il détermine les vitesses des articulations nécessaires pour atteindre un mouvement souhaitée de l'effecteur terminal.l'équation dovient :

$$[\dot{q}] = \left[J_{i,j}\right]^{-1} [\dot{X}] \quad \text{(II.5)}$$

5.2.3. Les modèles dynamiques

Ces modèles vont plus loin en tenant compte des forces, couples, masses et inerties du système. Ils expriment les équations du mouvement du robot, en reliant[13] :

- les couples et forces appliqués par les actionneurs.
- aux positions, vitesses et accélérations des différentes parties du robot.

Pour obtenir le modèle dynamique des robots on dispose un formalisme la plus souvent utiliser il s'agit de :

- formalisme de LAGRANGE qui d'écrit les équation en terms de travail et d'énergie du système.l'équation est défini sous la forme :

$$\Gamma_i = \frac{d}{dt}\left(\frac{dL}{d\dot{q}_i}\right) - \frac{dL}{dq_i} \quad \text{(II.6)}$$

Avec :

L : la fonction de LAGRANGE qui égale la défférence entre l'énergie cinétique Ec et l'énergie potentielle Ep du système.

5.3. Modèles géométrique directe

La modélisation géométrique directe permet de déterminer la position et l'orientation de l'organe terminal d'un robot à partir de ses coordonnées articulaires. Autrement dit, elle établit une relation mathématique qui, à partir des angles ou déplacements des articulations (coordonnées

généralisées), calcule la configuration spatiale de l'outil en bout de bras. Cette relation est généralement exprimée sous la forme suivante :

$$X_i = \mathrm{F}(\mathrm{q_i}) \qquad \text{(II.7)}$$

où $X_i$représente la position et l'orientation de l'organe terminal, et q_i l'ensemble des variables articulaires.

Repère Réfférentielles

pour construire un modèle géométrique cohérent, on commence par définir un référentiel de base (souvent noté), placé à la base du robot, puis on assigne à chaque articulation un référentiel local selon la convention de Denavit-Hartenberg (DH). Cette dernière fournit une méthode normalisée pour décrire la position relative des segments successifs du robot.[12]

Coordonnées homogènes

Soit un point P défini dans un repère cartésien par les coordonnées p_x, p_y, p_z (Figure II.1)On appelle coordonnées homogènes de ce point l'ensemble (wp_x, wp_y, wp_z) où w est un facteur d'échelle

En robotique, ce facteur est généralement fixé à w=1 Ainsi, le point Ps'exprime en coordonnées homogènes sous la forme :$p = \begin{pmatrix} p_x \\ p_y \\ p_z \\ 1 \end{pmatrix}$

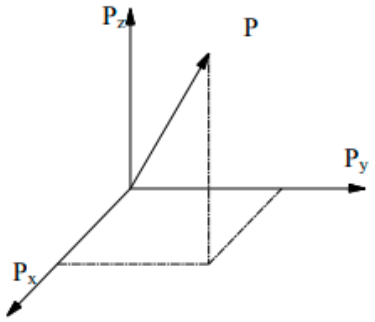


Figure II.7 : représentation d'un point .

Transformation des repères

Quand on veut exprimer un point ou un objet défini dans un repère R_j par rapport à un autre repère R_i, on utilise une matrice de transformation homogène notée T_i^j. Cette matrice permet de prendre en compte à la fois la rotation et la translation entre les deux repères, en une seule opération.

Par exemple, si le repère R_j est défini par trois vecteurs unitaires $\overrightarrow{U_j}\ \overrightarrow{V_j}\ \overrightarrow{W_j}$ (les axes x,y,z, du repère), et une origine $\overrightarrow{P_j}$,

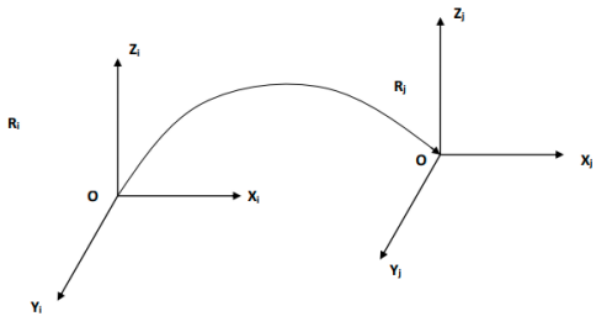


Figure II.8 : transformation des repères .

Alors la matrice de transformation homogène est :

$$T_i^j = \begin{bmatrix} U_x & V_x & W_x & P_x \\ U_y & V_y & W_y & P_y \\ U_z & V_z & W_z & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(II.8)}$$

Matrices homogènes

Les matrices homogènes sont des outils qui permettent de décrire une transformation (rotation + translation) entre deux repères, dans une seule multiplication matricielle.

On ajoute une 4e coordonnée (égale à 1) au vecteur pour pouvoir appliquer cette transformation

$$T_0^1 = \begin{bmatrix} R_0^1 & \vec{P} \\ 000 & 1 \end{bmatrix} \quad \text{(II.9)}$$

Pour exprimer les coordonnées d'un point P définies dans le repèreR_1, par rapport au repère R_0, on applique la matrice de transformation homogène T_0^1 . La relation est alors donnée par :

$$P_0 = T_0^1 \, . P_1 \qquad \text{(II.10)}$$

5.3.1.Notation standard de Denavit-Hartenberg (DH)

L'approche la plus utilisée pour modéliser géométriquement les robots à chaînes ouvertes est la méthode de Denavit-Hartenberg, qui permet de décrire la position relative entre les différentes liaisons à l'aide de quatre paramètres.[12]

Dans cette modélisation, on assigne un repère à chaque liaison du manipulateur, pour exprimer la position et l'orientation de l'organe terminal par rapport à la base, en fonction des coordonnées articulaires. Le résultat est un système d'équations non linéaires permettant de déterminer :

- le vecteur de position cartésienne p de l'effecteur,
- la matrice de rotation orthogonale R (3×3), définie par trois vecteurs unitaires correspondant aux axes X, Y et Z de l'organe terminal exprimés dans le repère de base.

Ces deux informations sont souvent combinées dans une matrice homogène 4×4 qui regroupe à la fois la position et l'orientation du point terminal du robot dans l'espace.

$$T_{i-1}^{i} = \begin{bmatrix} R_{3\times3} & P_{3\times\mathbf{1}} \\ 000 & 1 \end{bmatrix} \qquad \text{(II.11)}$$

Dans cette notation deux règles fondamentales s'appliquent :

1. L'axe x_i du repère i doit croiser l'axe z_{i-1} du repère précédent.
2. Cet axe x_i doit également être perpendiculaire à z_{i-1}.

Selon cette convention, un bras robotique possédant N articulations est modélisé par N + 1 liaisons (numérotées de 0 à N), où :

- Le lien 0 correspond à la base,
- Le lien N correspond à l'effecteur final.

Chaque articulation **i** connecte le $lien_{\,i-1}$ au $lien_{\,i}$ et permet de le mettre en mouvement.

Les quatre paramètres DH sont :

- a_i : Longueur de la liaison (distance entre les axes z_{i-1} et z_i le long de x_i).
- α_i : Angle de torsion entre z_{i-1} et z_i autour de x_i.
- d_i : La distance entre les axes x_{i-1} et x_i le long de z_{i-1}.
- θ_i : Angle de rotation entre les axes x_{i-1} et xi autour de l'axe z_{i-1}.

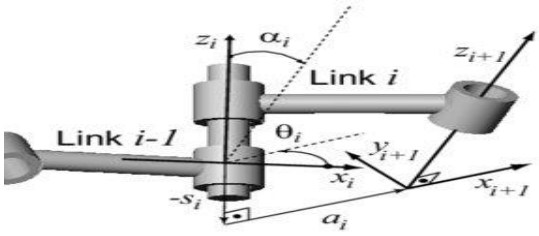


Figure II.9 : Représentation des repères et Paramètres de la liaison standard Denavit-Hartenberg

Convension de Denavit-Hartenberg

Comme illustré, le **repère i** est placé à l'extrémité du **lien i**, avec l'axe $\mathbf{z_i}$ aligné sur l'axe de l'articulation correspondante. Les paramètres qui décrivent la relation entre deux repères successifs (appelés paramètres DH) sont :

La transformation homogène entre deux repères consécutifs **i−1** et **i** est représentée par une matrice Ai−1, obtenue par la combinaison de rotations et translations élémentaires :

$$A^{i}_{\{i-1\}(\theta_i,d_i,a_i,\alpha_i)} = R_{z(\theta_i)} \cdot T_{z(d_i)} \cdot T_{x(a_i)} \cdot R_{x(\alpha_i)} \qquad \text{(II.12)}$$

Ce produit donne une matrice 4x4 qui décrit la position et l'orientation du repère i par rapport au repère i−1 , Cette matrice s'écrit comme suit:

$$A^{i}_{i-1} = \begin{bmatrix} \cos(\theta i) & -\sin(\theta i)\cos(\alpha i) & \sin(\theta i)\sin(\alpha i) & ai\cos(\theta i) \\ \sin(\theta i) & \cos(\theta i)\cos(\alpha i) & -\cos(\theta i)\sin(\alpha i) & ai\sin(\theta i) \\ 0 & \sin(\alpha i) & \cos(\alpha i) & di \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \text{(II.13)}$$

Dans un manipulateur en chaîne ouverte composé de N liaisons et N articulations (avec le lien 0 fixé à la base), chaque articulation offre un degré de liberté, défini par la variable articulaire qi.

La géométrique directe consiste à enchaîner les transformations successives entre les repères pour obtenir la pose finale (position et orientation) de l'effecteur T_0^N par rapport à la base :

$$\mathrm{T}_0^{\mathrm{N(q)}} = \mathrm{A}_0^{1(q_1)} \cdot \mathrm{A}_1^{2(q_2)} \cdot \ldots \cdot \mathrm{A}_{\{N-1\}}^{N(q_N)} \quad \text{(II.14)}$$

Ce calcul se fait de manière récursive et systématique, en multipliant les matrices DH une à une, chacune dépendant uniquement d'un paramètre articulaire.

5.4. Modèles géometrique inverse

La géométrique inverse vise à déterminer les valeurs des articulations (angles ou longueurs) permettant à l'effecteur terminal d'atteindre une position et une orientation donner.Elle constitue l'opération inverse de la géométrique directe, qui calcule la position de l'effecteur à partir d'une configuration donnée du robot..[20]Ce processus est fondamental pour traduire les mouvements souhaités dans l'espace opérationnel en ordres de mouvement articulaires. Cette relation est généralement exprimée sous la forme suivante :

$$\mathrm{r_i} = F(Xi) \quad \text{(II.15)}$$

Plusieurs méthodes sont disponibles pour le calcul du modèle géométrique invese d'un robot manipulateur. Parmi les plus connues et les plus utilisées, on peut citer la méthode géométrique classique , et la méthode géométrique analytique.

5.4.1. La méthode géométrique classique

Cette méthode basée sur l'analyse des triangles et des segments d'un bras robotique en utilisant des lois trigonométriques pour déterminer les angles. Elle est efficace pour les manipulateurs à deux ou trois degrés de liberté.

5.4.2. La méthode géométrique analytique(méthode de richard.paul) :

La méthode de Paul utilise les transformations homogènes et les paramètres DH pour modéliser analytiquement les liaisons du robot, ce qui la rend adaptée aux structures complexes.[20]

Principe de la méthode de Paul

Soit la matrice de transformation homogène

$$\mathrm{T}_0^{\mathrm{N}} = \mathrm{T}_0^1 . \mathrm{T}_1^2 \dots \dots . . \mathrm{T}_{N-1}^{\mathrm{N}} \quad \text{(II.16)}$$

Soit U_0 la situation désirée telle que :

$$U_0 = \begin{pmatrix} S_X & N_X & A_X & P_X \\ S_Y & N_Y & A_Y & P_Y \\ S_Z & N_Z & A_Z & P_Z \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad \text{(II.17)}$$

Equation générale à résoudre :

$$U_0 = \mathrm{T}_0^1 . \mathrm{T}_1^2 \dots \dots . . \mathrm{T}_{N-1}^{\mathrm{N}} \quad \text{(II.18)}$$

La méthode pour la résolution analytique de l'équation (II.15) consiste à pré multiplier successivement les deux membres de l'équation (II.15) par la matrice $\mathrm{T}_{\mathrm{i}}^{\mathrm{i}-1}$pour i variant de 1 à N-1,la succession des équations permettant le calcul de tous les variables articulaires est la suivante :

$$\left.\begin{array}{c} U_0 = \mathrm{T}_0^1 . \mathrm{T}_1^2 . \mathrm{T}_2^3 . \dots \dots . . \mathrm{T}_{N-1}^{\mathrm{N}} \\ \mathrm{T}_1^0 . U_0 = \mathrm{T}_1^2 . \mathrm{T}_2^3 . \dots \dots . . \mathrm{T}_{N-1}^{\mathrm{N}} \\ \mathrm{T}_2^1 . \mathrm{T}_1^0 . U_0 = \mathrm{T}_2^3 \dots \dots . . \mathrm{T}_{N-1}^{\mathrm{N}} \\ . \\ . \\ . \\ \mathrm{T}_3^2 . \mathrm{T}_2^1 . \mathrm{T}_1^0 \dots . \mathrm{T}_{N-1}^{\mathrm{N}-2} . U_0 = \mathrm{T}_{N-1}^{\mathrm{N}} \end{array}\right\} \quad \text{(II.19)}$$

Avec : $\mathrm{T}_{\mathrm{i}}^{\mathrm{i}-1}$ l'inverse de $\mathrm{T}_{\mathrm{i}-1}^{\mathrm{i}}$

Si :

$$\mathrm{T}_{\mathrm{i}-1}^{\mathrm{i}} = \begin{pmatrix} R_{i-1}^i & \overrightarrow{O_{\iota-1}^{\iota-1} O_\iota} \\ 000 & 1 \end{pmatrix} \quad \text{(II.20)}$$

Alors :

$$\mathrm{T}_{\mathrm{i}}^{\mathrm{i}-1} = \begin{pmatrix} {R_{i-1}^i}^T & -{R_{i-1}^i}^T . \overrightarrow{O_{\iota-1}^{\iota-1} O_\iota} \\ 000 & 1 \end{pmatrix} \quad \text{(II.21)}$$

Avec :

R_{i-1}^{i} : la matrice de rotation

$\overrightarrow{O_{i-1}^{i-1}O_i}$: le vecteur de décalage des origines

Après ces étapes, on obtient des expressions complexes qui relient les variables articulaires aux coordonnées souhaitées. Trouver ces équations et leurs solutions ne veut pas dire résoudre, dans le calcul du modèle géométrique inverse, trois situations peuvent se présenter :

- Absence de solution : Ce cas se produit lorsque la position ou l'orientation souhaitée se trouve en dehors de la zone accessible du robot. Cette zone est déterminée par plusieurs facteurs :
  - Le nombre de degrés de liberté du robot.
  - Les débattements articulaires, c'est-à-dire les limites d'angle des articulations.
  - La dimension des segments du robot.
- Infinité de solutions : Ce cas survient lorsque plusieurs configurations du robot peuvent atteindre la même position ou orientation. Cela peut se produire dans les situations suivantes :
  - Lorsque le robot est redondant par rapport à la tâche, disposant de plus de degrés de liberté que nécessaire.
  - Lorsque le robot se trouve dans certaines configurations singulières, où les équations de mouvement peuvent donner lieu à plusieurs solutions.
- Solutions en nombre fini : Un robot manipulateur est considéré comme résoluble lorsqu'il est possible de déterminer toutes les configurations qui permettent d'atteindre une position ou une orientation donnée. Dans ce cas, il existe un nombre limité de solutions valides .

6. La modélisation du notre robot (RRR)

6.1. Le modèle géometrique direct (MGD) de BCN3D (modifier)

Dans le cadre de cette étude, nous nous sommes limités sur les trois premières articulations du bras robotique.Nous allons donc déterminer uniquement les paramètres de Denavit-Hartenberg (DH) relatifs à ces trois articulations.

Prototype Denavit-Hartenberg Parameters

Le tableau suivant présente les paramètres DH définis pour les trois premières articulations du prototype étudié. Ces valeurs résultent d'un positionnement rigoureux des repères selon les règles de la convention DH, basé sur la configuration mécanique du bras BCN3D :

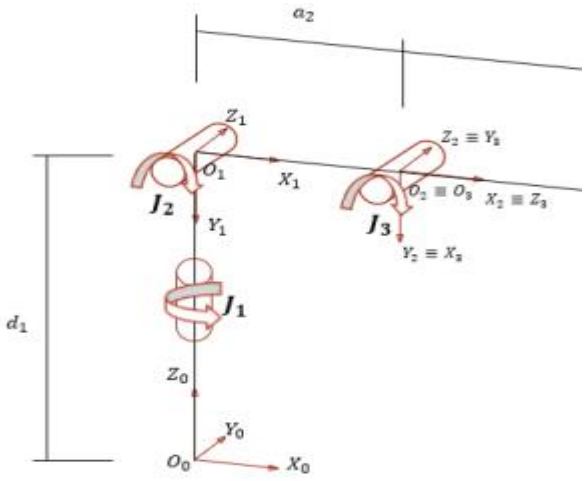


Figure II.10 : Représentation des repère associer a chaque articulation [12]

Joint	a_i(cm)	α_i (°)	$di(cm)$	θi(°)
1	0	-90	L1=23.2	θ_1 +0
2	L2=22.1	0	0	θ_2 +0
3	L3=13	0	0	θ_3 +0

Tableau II.5 : les paramètres DH du prototype.

Les matrix de transformation A_i de chaque articulation :

La matrice de transformation A_0^1 éxprime la relation entre le repère R_1et le réfférencielle R_0 :

$$A_0^1 = \begin{bmatrix} \mathrm{Cos}(\theta_1) & -\sin(\theta_1)\mathrm{Cos}(\alpha_1) & \sin(\theta_1)\sin(\alpha_1) & a_1\mathrm{Cos}(\theta_1) \\ \sin(\theta_1) & \mathrm{Cos}(\theta_1)\mathrm{Cos}(\alpha_1) & -\mathrm{Cos}(\theta_1)\sin(\alpha_1) & a_1\sin(\theta_1) \\ 0 & \sin(\alpha_1) & \mathrm{Cos}(\alpha_1) & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(II.22)}$$

En remplacent les valeur de chaque parammètre dans l'équation :

$$A_0^1 = \begin{bmatrix} \mathrm{Cos}(\theta_1) & 0 & \sin(\theta_1) & 0 \\ -\sin(\theta_1) & 0 & -\mathrm{Cos}(\theta_1) & 0 \\ 0 & -1 & 0 & L_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(II.23)}$$

La matrice de transformation A_1^2 éxprime la relation entre le repère R_2et le Repère précédent R_1 :

$$A_1^2 = \begin{bmatrix} \mathrm{Cos}(\theta_2) & -\sin(\theta_2)\mathrm{Cos}(\alpha_2) & \sin(\theta_2)\sin(\alpha_2) & a_2\mathrm{Cos}(\theta_2) \\ \sin(\theta_2) & \mathrm{Cos}(\theta_2)\mathrm{Cos}(\alpha_2) & -\mathrm{Cos}(\theta_2)\sin(\alpha_2) & a_2\sin(\theta_2) \\ 0 & \sin(\alpha_2) & \mathrm{Cos}(\alpha_2) & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(II.24)}$$

$$A_1^2 = \begin{bmatrix} \mathrm{Cos}(\theta_2) & -\sin(\theta_2) & 0 & L_2\mathrm{Cos}(\theta_2) \\ \sin(\theta_2) & \mathrm{Cos}(\theta_2) & 0 & L_2\sin(\theta_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(II.25)}$$

La matrice de transformation A_2^3 éxprime la relation entre le repère R_3et le repère précédent R_2 :

$$A_2^3 = \begin{bmatrix} \mathrm{Cos}(\theta_3) & -\sin(\theta_3)\mathrm{Cos}(\alpha_3) & \sin(\theta_3)\sin(\alpha_3) & a_3\mathrm{Cos}(\theta_3) \\ \sin(\theta_3) & \mathrm{Cos}(\theta_3)\mathrm{Cos}(\alpha_3) & -\mathrm{Cos}(\theta_3)\sin(\alpha_3) & a_3\sin(\theta_3) \\ 0 & \sin(\alpha_3) & \mathrm{Cos}(\alpha_3) & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(II.26)}$$

$$A_2^3 = \begin{bmatrix} \mathrm{Cos}(\theta_3) & -\sin(\theta_3) & 0 & L_3\mathrm{Cos}(\theta_3) \\ \sin(\theta_3) & \mathrm{Cos}(\theta_3) & 0 & L_3\sin(\theta_3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(II.27)}$$

Après la multiplication des matrices des transformations élémentaires, nous obtenons la matrice de transformation global T_0^3 qui exprime la relation entre la base et l'effecteur ou bien le modèle géométrique direct :

$$T_0^3 = T_0^1 + T_1^2 + T_2^3 = A_0^1 \,.\, A_1^2 \,.\, A_2^3 \quad \text{(II.28)}$$

$$T_0^3 = \begin{bmatrix} \mathrm{Cos}(\theta_1) & 0 & \sin(\theta_1) & 0 \\ -\sin(\theta_1) & 0 & -\mathrm{Cos}(\theta_1) & 0 \\ 0 & -1 & 0 & L_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \begin{bmatrix} \mathrm{Cos}(\theta_2) & -\sin(\theta_2) & 0 & L_2\mathrm{Cos}(\theta_2) \\ \sin(\theta_2) & \mathrm{Cos}(\theta_2) & 0 & L_2\sin(\theta_2) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(II.29)}$$

$$\begin{bmatrix} \mathrm{Cos}(\theta_3) & -\sin(\theta_3) & 0 & L_3\mathrm{Cos}(\theta_3) \\ \sin(\theta_3) & \mathrm{Cos}(\theta_3) & 0 & L_3\sin(\theta_3) \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_0^3 = \begin{bmatrix} C_1C_2C_3 - C_1S_2S_3 & -C_1C_2S_3 - C_1S_2C_3 & S_3 & L_2C_1C_2C_3 - L_2C_1S_2S_3 + L_2C_1C_2 \\ S_1C_2C_3 - S_1S_2S_3 & S_1S_2S_3 + S_1S_2C_3 & -C_1 & -L_2S_1C_2C_3 + L_3S_1S_2S_3 - L_2C_2S_1 \\ -S_2C_3 - C_2S_3 & S_2S_3 - C_2C_3 & 0 & -L_3C_3S_2 - L_3C_2S_3 - L_1 + L_2S_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{(II.30)}$$

Donc les paramètres de position de l'effecteur terminal sont donnés dans la dernière colonne de la matrice.

$$\begin{cases} px = L_2C_1C_2C_3 - L_2C_1S_2S_3 + L_2C_1C_2 \\ py = -L_2S_1C_2C_3 + L_3S_1S_2S_3 - L_2C_2S_1 \\ pz = -L_3C_3S_2 - L_3C_2S_3 - L_1 + L_2S_2 \end{cases} \quad \text{(II.31)}$$

Avec l'angle θ_i est connue.

6.2. Le modèle géometrique inverse (MGI)

La methode géométrique classique :

Après avoir établi le modèle géométrique direct du notre bras robotique,permettant de déterminer la position de l'effecteur final à partir des angles des articulations,nous passons maintenant à l'étude de la cinématique inverse,qui consiste à calculer les angles des articulations $\theta_1, \theta_2, \theta_3$ pour que l'effecteur atteigne a la position (p_x, p_y, p_z).

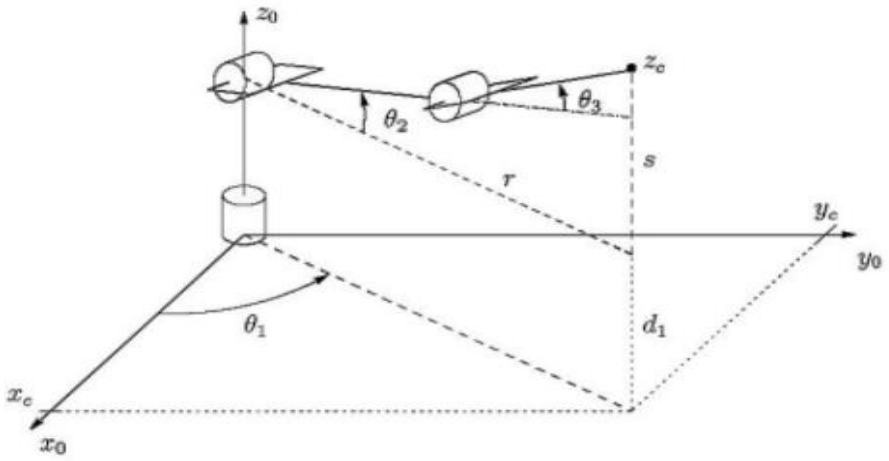


Figure II.11 : représentation géométrique du robot RRR dans le plan tridimensionnel .

Détermination du $\theta_1, \theta_2, \theta_3$:

L'objectif est de trouver les l angles articulaires $\theta_1, \theta_2, \theta_3$ qui donnent ce point $O_{c=}\ (x_c y_c z_c)^T$ spécifiée dans la dernière colonne de la matrice de transformation de Denavit-Hartenberg dans l'équation (II.22).

1/ Nous considérons la projection du point sur le plan $X_0 - Y_0$ pour simplifier l'analyse et on a choisir un point :

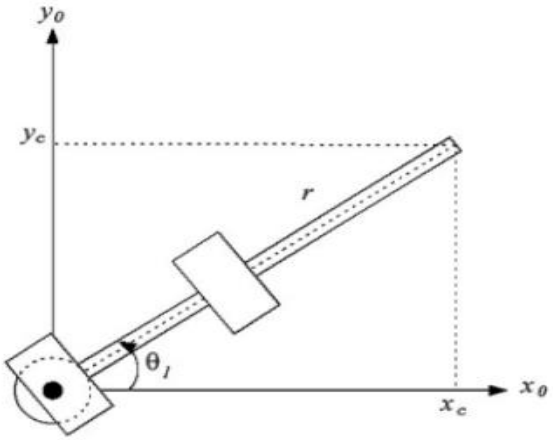


Figure II.12 : la projection du robot sur le plan x-y.

L'angle θ_1 correspondant à la rotation autour de l'axe vertical Z_0 peut etre obtenu grace à la formule suivante :

$$\theta_1 = \tan^{-1}\left(\frac{y_c}{x_c}\right) \quad \text{(II.32)}$$

Ou :

$$\theta_1 = \Pi + \tan^{-1}\left(\frac{y_c}{x_c}\right) \quad \text{(II.33)}$$

2/projection dans le plan r-z :

En définit maintenant les variable r et **s** tel que

r :est la distance horizantale entre la base du robot et la projection du point finale exprimer par la relation suivante :

$$r = \sqrt{{x_c}^2 + {y_c}^2} \quad \text{(II.34)}$$

s :distance vericale entre le point finale et la hauteur de la base qui egale :

$$s = z_c - d_1 \quad \text{(II.35)}$$

3 /calcul de θ_3 à l'aide de la loi du cosinus :

Les deux segment du bras a_1a_2 et la ligne droite vers le point finale formé une triangle ce qui permet d'appliquer la loi du cosinus suivante :

$$\cos\theta_3 = \frac{r^2 + s^2 - a_2^2 - a_3^2}{2a_2a_3} = D \tag{II.36}$$

On a :

$$cos^2(\theta) + sin^2(\theta) = 1 \tag{II.37}$$

Qui implique :

$$sin^2(\theta_3) = 1 - cos^2(\theta_3) = 1 - D^2 \tag{II.38}$$

Donc :

$$sin(\theta_3) = \sqrt{1 - D^2} \tag{II.39}$$

Et on a :

$$\tan(\theta_3) = \frac{\cos(\theta_3)}{\sin(\theta_3)} \tag{II.40}$$

Qui implique :

$$\theta_3 = \tan^{-1}\left(\frac{\sin(\theta_3)}{\cos(\theta_3)}\right) = \tan^{-1}\left(\mp\frac{\sqrt{1 - D^2}}{D}\right) \tag{II.41}$$

4 /maintenant on peut calculer l'angle θ_2 par géométrie :

$$\theta_2 = \tan^{-1}\left(\frac{s}{r}\right) - \tan^{-1}\left(\frac{a_3\sin(\theta_3)}{a_2 + a_3\cos(\theta_3)}\right) \tag{II.42}$$

7. Simulation MATLAB :

Pour simuler les modèles géométriques direct et inverse, nous avons utilisé le « Robotics Toolbox » de Peter Corke [21] pour visualiser le bras manipulateur en 3D. Nous avons développé une interface graphique sous MATLAB afin de tester et visualiser les mouvements du robot. Cette interface comprend trois champs pour saisir les coordonnées (q1 q2 q3 ou x y z) et trois autres pour afficher les coordonnées (x y z ou q1 q2 q3) en fonction du modèle sélectionné, en cliquant sur l'un des boutons (Forward Kinematics pour le modèle direct ou Inverse Kinematics pour le modèle inverse). Cette fenêtre permet également de visualiser la configuration du robot en 3D.

Les tests réalisés avec différentes valeurs de coordonnées ont démontré que le modèle établi est correct. Nous présentons ici quelques tests effectués pour les deux modèles dans les figures (**II.13** , **II.14** , **II.15**)

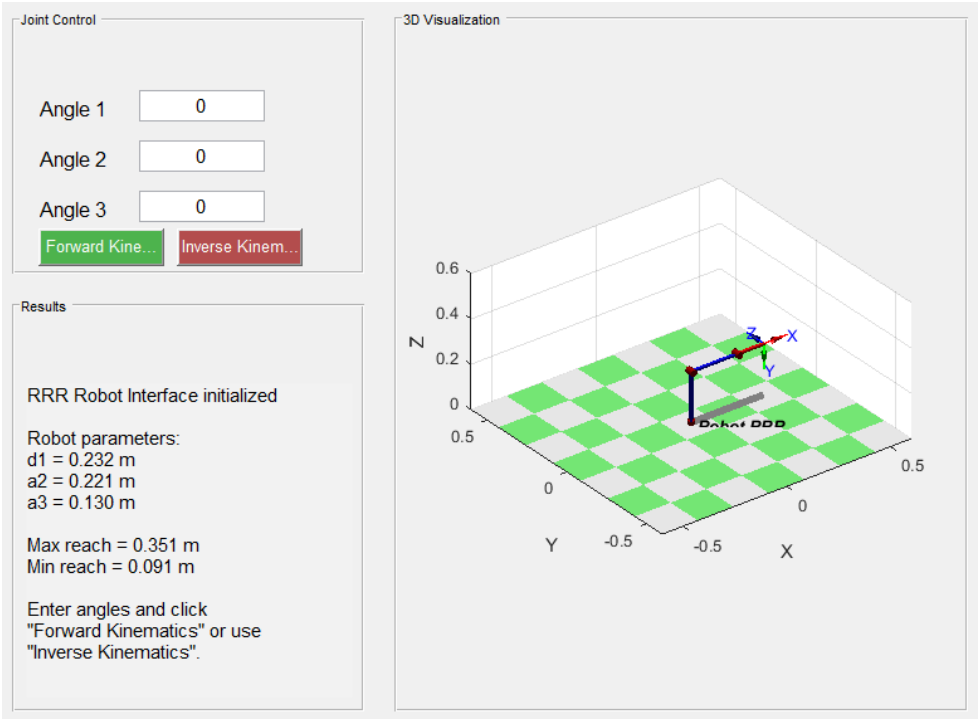


Figure II.13 : Configuration initiale du bras

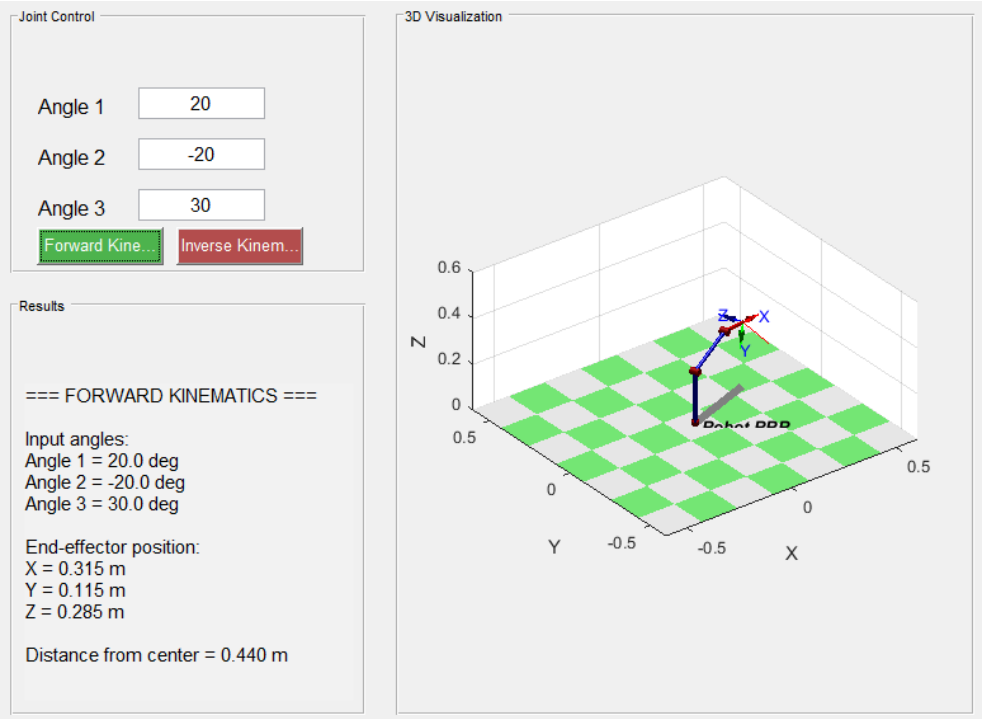
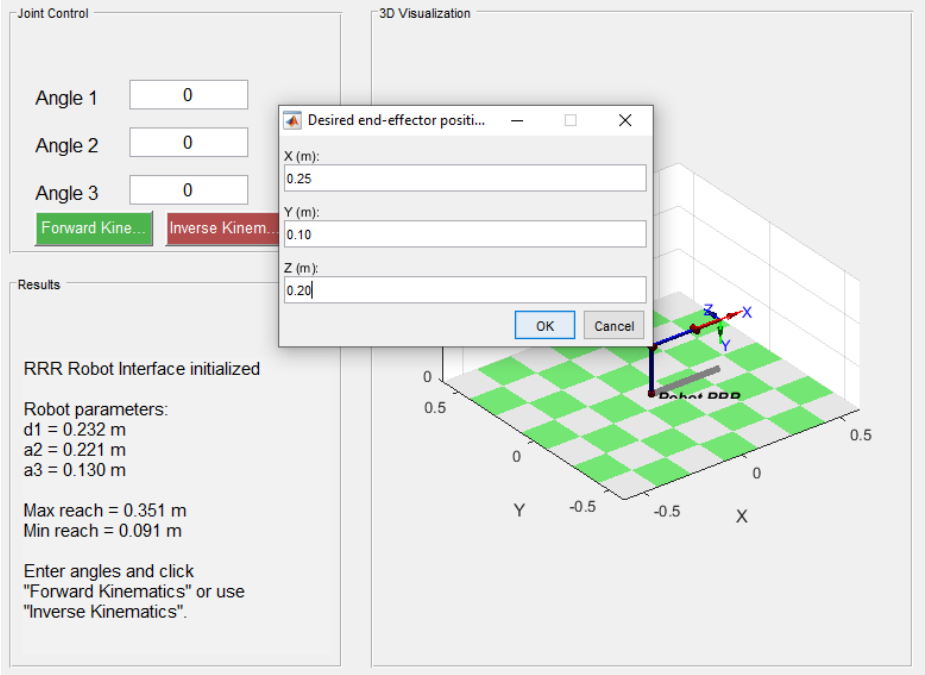


Figure II.14 : La simulation du modèle géométrique direct



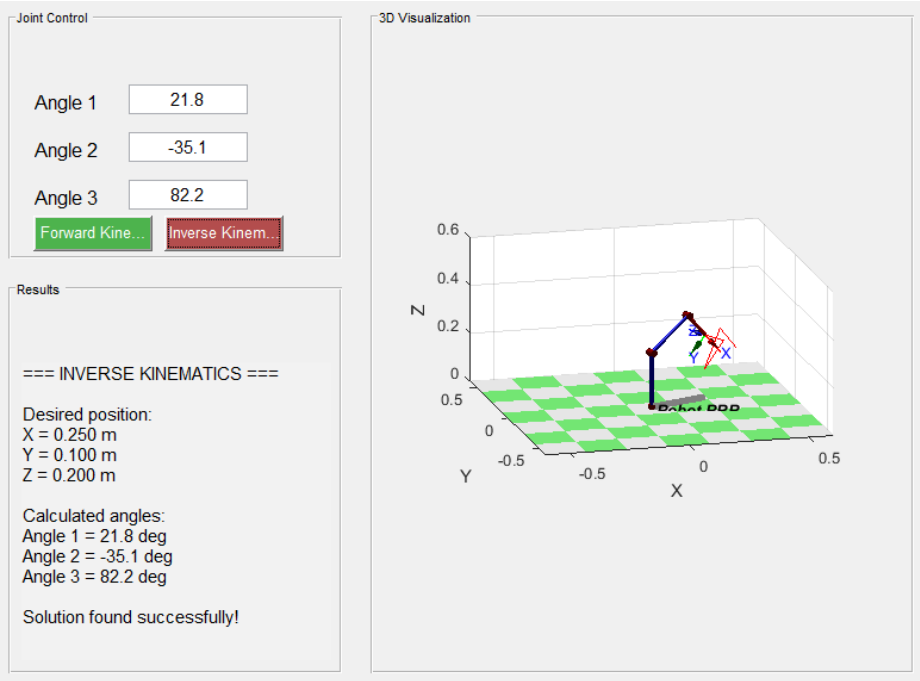


Figure II .15 : La simulation du modèle géométrique inverse .

8. Conclusion

Dans ce chapitre, nous avons présenté le bras robotique utilisé dans notre projet, en décrivant ses principales composantes mécaniques et électroniques, ainsi que son architecture globale. Nous avons ensuite abordé la modélisation du robot, en développant les modèles géométriques direct et inverse, qui nous permettent de déterminer avec précision la position et l'orientation de l'outil terminal en fonction des angles articulaires. Ces modèles ont été réalisés et simulés dans l'environnement MATLAB, à l'aide d'une interface graphique accesible. Les résultats de ces simulations nous ont permis de visualiser le comportement cinématique du robot, et seront exploités dans le chapitre suivant.

Chapitre III : Conception et réalisation du travai

1.Introduction

Ce chapitre est consacré à la réalisation concrète du projet avec détaille de l'ensemble des opérations nécessaire à la fabrication du pièces par l'imprission 3D,à l'assemblage du robot ainsi que l'instalation de tous les compossants éléctrique et éléctronique .enfin la programmation de la carte arduino et les premiers test de fonctionnement et réaliser l'objectif du projet, à savoir de manipuler un objet depuis de point A vers un point B .

2. Les outils utiliser pour la fabrication des pièces de robot

Objectif

Les pièces mécaniques du robot ont été imprimées en 3D à partir des fichiers STL[15] du projet open source BCN3D Moveo. Ces fichiers ont été traités avec le logiciel Creality Slicer pour générer un fichier G-code qui contient toutes les instructions nécessaires à l'imprimante : la température, la vitesse de déplacement, la hauteur des couches, etc. si le fichier G-code généré, nous l'avons copié sur une carte SD, que nous avons ensuite insérée dans l'imprimante 3D. L'impression a été réalisée à l'aide d'une Ender 3 Pro et de filament PLA high speed, ce qui nous a permis de produire l'ensemble des pièces nécessaires .

2.1.Imprimante Creality 3-Ender PRO

Cette imprémante offre un excellent compromis entre qualité d'impression ,facilité d'utilisatuon et accessibilite financière .

Figure III.1 : Imprimante Creality 3-Ender PRO utiliser pour fabriquer les pièces du robot.

2.2. Filament PLA High speed

On a choisi cette type de filament en raison de ses propriétés adaptées aux impression rapides et précise , il offre une excellente imprimabilité, un risque minimal de déformation, et une forte adhérence au plateau, garantissant des résultats fiables et de haute qualité à des vitesses accrues.

Figure III.2 : filament PLA high speed utiliser.

2.3.Creality slicer

Est un logiciel essentiel en impression 3D choisi pour sa simplicité, sa compatibilité avec les imprimantes et ses réglages précis. Il permet de convertir un fichier STL (modèle 3D) en G-code, un langage compréhensible par l'imprimante.

2.3.1.Installation et configuration du slicer

Le logiciel Creality Slicer a été téléchargé via le site officiel Creality. Lors de l'installation, nous avons sélectionné l'imprimante Ender 3 Pro, puis adapté les paramètres de base :

- Diamètre de buse : 0,4 mm.
- Dimensions du plateau : 220 x 220 x 250 mm.
- Type de filament : PLA (high speed) .
- Temp : nozzel = entre 200°C et 230°C, plateau = 60°C et 90 °C.

Préparation des modèles

- Les fichiers 3D (au format STL) sont importés dans le slicer.

- Positionnement : centrage automatique sur le plateau.
- Supports : (type radeau) sélectionné pour faciliter le retrait de pièce.

Réglages techniques

- Hauteur de couche : 0,2 mm pour un compromis entre qualité et vitesse.
- Vitesse d'impression : 80 mm/s (grâce au filament High Speed).
- Rétraction : activée avec Z-hop pour limiter les fils.
- Ventilation : 100% à partir de la 3ème couche pour meilleure qualité de surface

Génération du G-code et lancement de l'imréssion

Une fois les paramètres validés, le G-code est exporté sur carte SD. L'impression est ensuite lancée via l'imprimante. Le suivi est réalisé manuellement ou via l'écran LCD de l'Ender 3 Pro.

la figure suivant représente l'interface du logiciel Slicer compatible avec l'imprimante Ender 3 Pro, et réglage des paramètres d'impression ainsi que la position du modèle sur le plateau d'impression

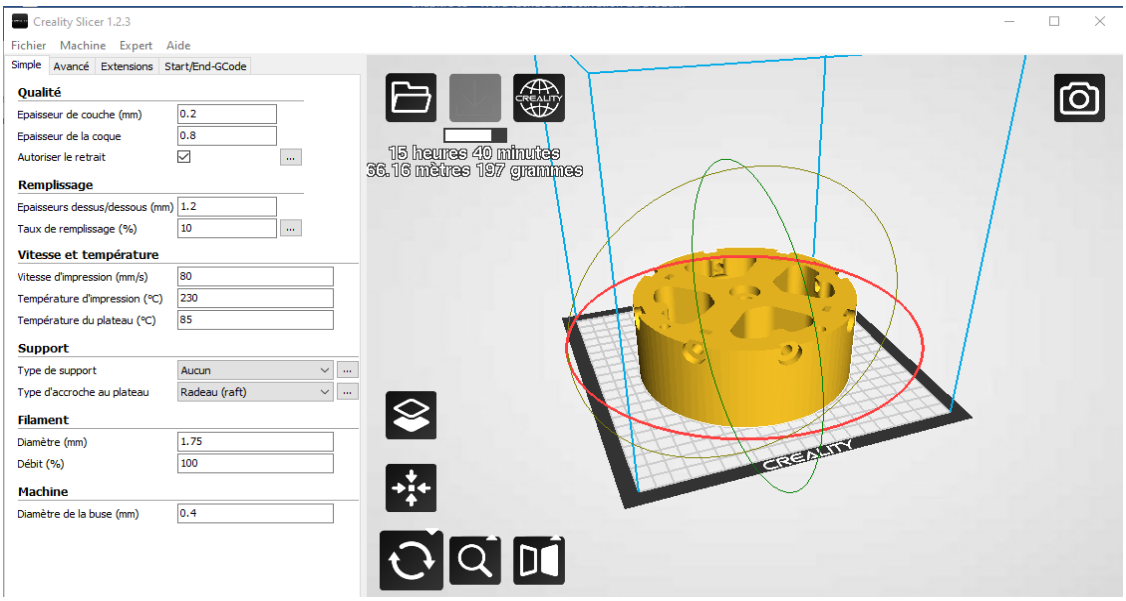


Figure III .3 : interfce de logiciel slicer adapté pour l'imprémante ender-3pro.

3. Assemblage mécanique et intégration électronique

Cette étape a consisté à assembler les différentes pièces mécaniques du bras robotique qui ont été préalablement imprimées en 3D. ainsi que les différents éléments électrique, électronique

et l'élément de fixation, comprenant vis, écrous, roulements,les courroies…en peux résumer les étapes d'assemblage comme suit :

a. Nous avons commencé par fixer la base sur un support stable et le moteur NEMA 17 le support ,la transmission du mouvement est assurée par un système à courroie crantée, reliant la poulie du moteur à la plaque tournante Comme illustré dans la (Figure III.4**)**,les compossant utiliser dans cette étape et identifier en detaill dans le tableau suivant :

N°	**Compossant**	**Quantité**
1	Base rotative	1
2	Plaque rotative	1
3	Support de moteur de base	1
4	Roulement 8 mm x 22 mm x 7 mm	2
5	Roulement 5 mm x 16 mm x 5 mm	8
6	Entretoise 5 mm x 10 mm	8
7	Entretoise 8 mm x 20 mm	1
8	Vis M8 x 65 mm	1
9	Vis M5 x 20 mm	8
10	Vis M3 x 8 mm	4
11	Vis M4 x 40 mm	4
12	Vis M4 x 30 mm	4
13	Écrou frein M4	12
14	Écrou frein M5	8
15	Écrou frein M8	1
16	Poulie pour arbre 5 mm	1
17	Nema 17 SM42HT47- 1684	1

Tableau III.1 : les élément utiliser pour rassembler la base.

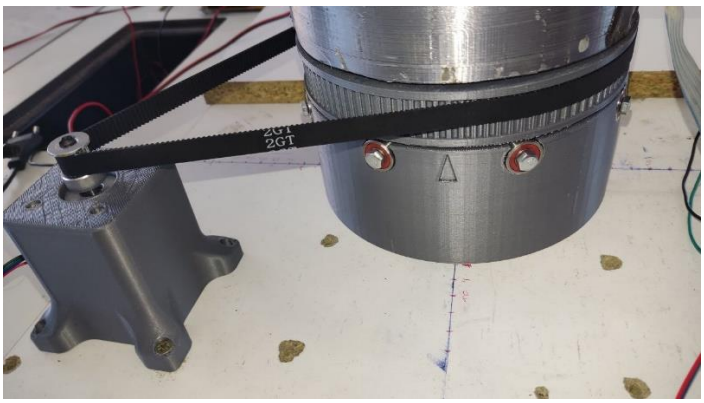


Figure III.4 : Assemblage de la base rotative.

b. l'étape suivante consiste à assembler l'articulation de l'épaule cette pièce est placée au-dessus de la base rotative Cette articulation est entraînée par un moteur pas à pas NEMA 23 le tableau suivant ilustre les compossant de cett articulation :

N°	Compossant	Quantité
1	Base d'épaule	1
2	Articulation d'épaule	2
3	Écrou M4	4
4	Écrou frein standard	4
5	Écrou frein M5	4
6	Vis M4 x 40 mm	4
7	Vis M3 x 25 mm	2
8	Vis M4 x 45 mm	6
9	Vis M5 x 14 mm	4
10	Nema 23 SM57HT76-3004A	1
11	Poulie 8mm T5	1
12	Roulement à billes 8×22×7 mm	2
13	Barre lisse (134 mm)	1

Tableau III.2 : les élément utiliser pour rassembler l'épaule .

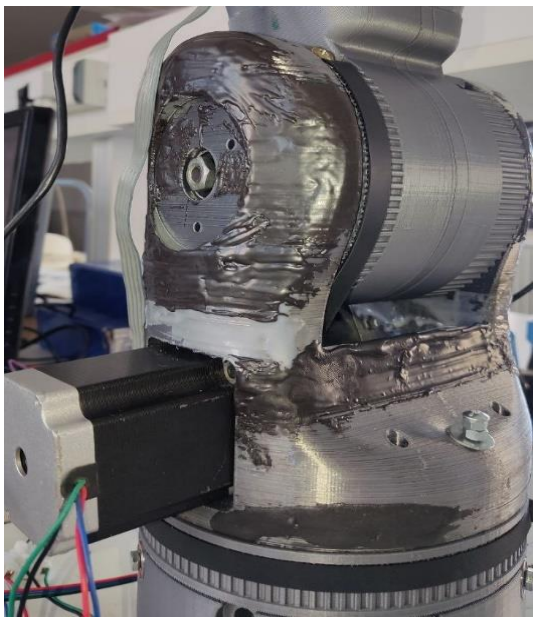


Figure III.5 : Fixation de l'articulation 1 sur la base .

c. la derniére étape consiste à assembler l'articulation du coude cette articulation reliée à l'épaule à l'aide d'une base coudée, le mouvement est assuré par un moteur Nema 17 SM42HT33- 1334A

N°	Compossant	Quantité
1	Base coudée	1
2	½ Articulation de coude	2
3	Vis M4 x 16 mm	4
4	Vis M3 x 40 mm	1
5	Écrou M3	4
6	Roulement 8 mm x 22 mm x 7 mm	2
7	Nema 23 SM57HT76-3004A	1
8	Poulie 8mm	1
9	Barre lisse (114 mm)	1

Tableau III.3 : les élément utiliser pour rassembler le coude.

Figure III.6 : Montage de la dernière articulation **.**

Après la finalisation de l'assemblage, le bras robotique est désormais complet , la figure suivante présente l'apparence finale du système prêt à etre tester.

Figure III.7 : Vue d'ensemble après l'assemblage des articulations.

4. Commande électronique et câblage

Après avoir finalisé la structure du bras, nous avons procédé au câblage électronique. La commande est assurée par une carte Arduino Uno**(Tableau II.4),** un Shield CNC **(Figure II .4)** et des drivers DRV8825**(Figure II.3)** pour chaque moteur pas à pas. Les signaux STEP et DIR sont envoyés depuis l'Arduino vers chaque driver, L'alimentation externe (12 V) alimente le shield et les moteurs de façon propre, sans solliciter directement l'Arduino.

Figure III.8 : vue finale d'assemblage mécanique et eléctronique du robot .

4.1. Software Arduino IDE

Le logiciel Arduino IDE a été utilisé pour écrire, compiler et téléverser le code de commande sur la carte Arduino Uno. Le programme, écrit en C/C++, permet de générer les signaux nécessaires (STEP et DIR) pour contrôler les moteurs pas à pas.

Figure III.9: Logiciel de programmation Arduino IDE

4.2. Test de fonctionnement

Après l'assemblage mécanique et électrique complet, nous avons procédé à un premier test avec un code Arduino simple ,Avant de commencer les essais, nous avons effectué une mesure expérimentale afin de déterminer la relation entre le nombre de pas et l'angle de rotation. En faisant tourner le moteur un tour complet de 200 pas, nous avons mesuré une rotation réelle de 40° de bras . Cette valeur a été retenue comme base de calcul.

Dans le programme Arduino, nous ne donnons pas directement le nombre de pas, mais l'angle souhaité. Le code utilise le rapport mesuré pour calculer automatiquement le nombre de pas à envoyer au moteur, selon la formule suivante :

$$200\ pas(moteur) \rightarrow rotation\ 40°\ de\ l'articulation$$

Pour calculer les nombre de pas d'une angle donnée en utilise la formule suivante :

$$\text{Nbr pas} = \frac{\text{angle donnée} \times 200}{40}$$

Ou bien utiliser le rapport de réduction mécanique :

$$\text{Gear}_{\text{ratio}} = \frac{360}{40} = 9$$

Les tests réalisés sur chaque articulation ont permis de valider cette approche. Voici le code arduino pour tester la fonctionnement de 3 moteur avec des angles donnée : $\theta_1 = -30°, \theta_2 = -100°, \theta_3 = 60°$

Structure de la programmation

1) Au niveau de la partie déclarative nous avons identifié les broches du CNC Shield pour les trois moteurs pas à pas du robot (base, épaule, coude) et défini les paramètres mécaniques, incluant le nombre de pas par tour, le rapport de réduction et les angles de rotation pour tester les mouvements de chaque axe, la figure III.10 représente la déclaration

test_de_fonct

```
#include <AccelStepper.h>

// CNC Shield pin assignments for axis motors
#define X_STEP_PIN     2    // Axis 1 (Base) - X on CNC Shield
#define X_DIR_PIN      5
#define Y_STEP_PIN     3    // Axis 2 (Shoulder) - Y on CNC Shield
#define Y_DIR_PIN      6
#define Z_STEP_PIN     4    // Axis 3 (Elbow) - Z on CNC Shield
#define Z_DIR_PIN      7

// Mechanical parameters (adjust based on your gear ratio)
#define STEPS_PER_REV      200    // 1.8° steppers (200 steps/revolution)
#define GEAR_RATIO         9      // 9:1 reduction (360°/40°)
#define DEGREES_PER_STEP   (360.0/(STEPS_PER_REV*GEAR_RATIO))  // 0.2° per step

// Movement parameters (in degrees)
#define AXIS1_ANGLE       -30.0    // Base rotation
#define AXIS2_ANGLE       -100.0    // Shoulder movement
#define AXIS3_ANGLE       60.0    // Elbow movement
#define PAUSE_DURATION    1000    // 1 second pause between movements
```

Figure III.10 : Déclaration des broches et des paramètres .

2) Cette section de code se concentre sur la configuration dynamique des moteurs pas à pas en spécifiant pour chaque axe la vitesse maximale et l'accélération appropriées selon les caractéristiques mécaniques de chaque articulation.la figure (III.11)correspond à cette configuration

test_de_fonct

```
// Initialize steppers with CNC Shield pins
AccelStepper axis1(AccelStepper::DRIVER, X_STEP_PIN, X_DIR_PIN);
AccelStepper axis2(AccelStepper::DRIVER, Y_STEP_PIN, Y_DIR_PIN);
AccelStepper axis3(AccelStepper::DRIVER, Z_STEP_PIN, Z_DIR_PIN);

// Test control variables
enum TestState {MOVE_AXIS1, PAUSE1, MOVE_AXIS2, PAUSE2, MOVE_AXIS3, TEST_COMPLETE};
TestState testState = MOVE_AXIS1;
unsigned long pauseStartTime = 0;

void setup() {
  // Configure stepper motors
  // Base (Axis1) - NEMA17 typically
  axis1.setMaxSpeed(500);        // Steps per second
  axis1.setAcceleration(300);    // Steps per second²

  // Shoulder (Axis2) - NEMA23 typically (higher torque, slower)
  axis2.setMaxSpeed(400);
  axis2.setAcceleration(200);

  // Elbow (Axis3) - NEMA17 typically
  axis3.setMaxSpeed(500);
  axis3.setAcceleration(300);
```

Figure III.11 : configuration dynamique des moteur pas à pas dans le code arduino .

3) Dans cette section, nous avons défini la séquence automatique des mouvements de test. Chaque moteur est activé successivement à un angle défini ,suivi d'une pause temporaire pour observer son comportement. Cela permet de tester et de valider individuellement le fonctionnement de chaque articulation du bras.la figure III.2 correspond a cette étape

test_de_fonct

```
  // Start first movement
  moveAxisTo(axis1, AXIS1_ANGLE);
}

void loop() {
  switch(testState) {
    case MOVE_AXIS1:
      if (axis1.run()) {
        // Still moving
      } else {
        startPause();
        testState = PAUSE1;
      }
      break;

    case PAUSE1:
      if (millis() - pauseStartTime >= PAUSE_DURATION) {
        moveAxisTo(axis2, AXIS2_ANGLE);
        testState = MOVE_AXIS2;
      }
      break;
```

Figure III .12 : teste individuelle de chaque articulation du bras .

4) Ce bloc contient les fonctions utilitaires essentielles pour le bon déroulement du test.comme indique dans la **(Figure III.13)**

```
// Move axis to specified angle in degrees
void moveAxisTo(AccelStepper &stepper, float degrees) {
  long steps = degrees / DEGREES_PER_STEP;
  stepper.moveTo(steps);
}

// Start pause between movements
void startPause() {
  pauseStartTime = millis();
}

// Emergency stop (hardware reset needed after this)
void emergencyStop() {
  axis1.stop();
  axis2.stop();
  axis3.stop();
  while(true); // Infinite loop until reset
}
```

Compilation terminée.

Figure III.13 : l'ensemble des fonctions utiliser dans la programme de test.

Résultat

Le code permet de déplacer les trois moteurs pas à pas de manière séquentielle, chaque moteur se déplaçant vers l'angle spécifié, avec une pause entre chaque mouvement et les moteurs réagissent exactement avec la formule 40° pour 200 pas.

4.3. Programme principal de manipulation : déplacement de A vers B :

Dans le cadre de notre projet nous avons utilisé la méthode géométrique inverse qui on a montrer dans le **chapitre II** pour calculer avec précision les angles des articulations nécessaires à atteindre une position cible définie par les coordonnées A(25,12,35) et B(25,-16,35)

Le calcul a été réalisé à l'aide du logiciel MATLAB, ce qui nous a permis de déterminer les valeurs des angles $\theta_1\theta_2\theta_3$ayant permis d'atteindre la position A et $\theta_1'\theta_2'\theta_3'$ pour passer a la position B . Ces angles ont ensuite été intégrés dans le programme Arduino, permettant ainsi de diriger le robot vers la position souhaitée

Le programme matlab

Command Window

```
>> % === Données d'entrée ===
x = 25;       % Coordonnée X de la position A
y = 12;       % Coordonnée Y de la position A
z = 35;       % Coordonnée Z de la position A
d1 = 23.2;    % Hauteur du premier axe (offset vertical)
a2 = 22.1;    % Longueur du premier bras (lien 2)
a3 = 13;      % Longueur du deuxième bras (lien 3)

% === 1. Calcul de θ1 ===
theta1 = atan2(y, x);            % Calcul de θ1 en radians
theta1_deg = rad2deg(theta1)     % Conversion de θ1 en degrés

% === 2. Calcul de r et s ===
r = sqrt(x^2 + y^2);             % Distance horizontale du point A
s = z - d1;                      % Hauteur relative du point A

% === 3. Calcul du terme D pour θ3 ===
D = (r^2 + s^2 - a2^2 - a3^2) / (2 * a2 * a3)   % Formule du cosinus pour θ3

% === 4. Calcul de θ3 (deux solutions possibles) ===
theta3_1 = atan2( sqrt(1 - D^2), D );    % Solution "coude en bas"
theta3_2 = atan2(-sqrt(1 - D^2), D );    % Solution "coude en haut"

theta3_deg_1 = rad2deg(theta3_1)         % θ3 en degrés (coude en bas)
theta3_deg_2 = rad2deg(theta3_2)         % θ3 en degrés (coude en haut)
```

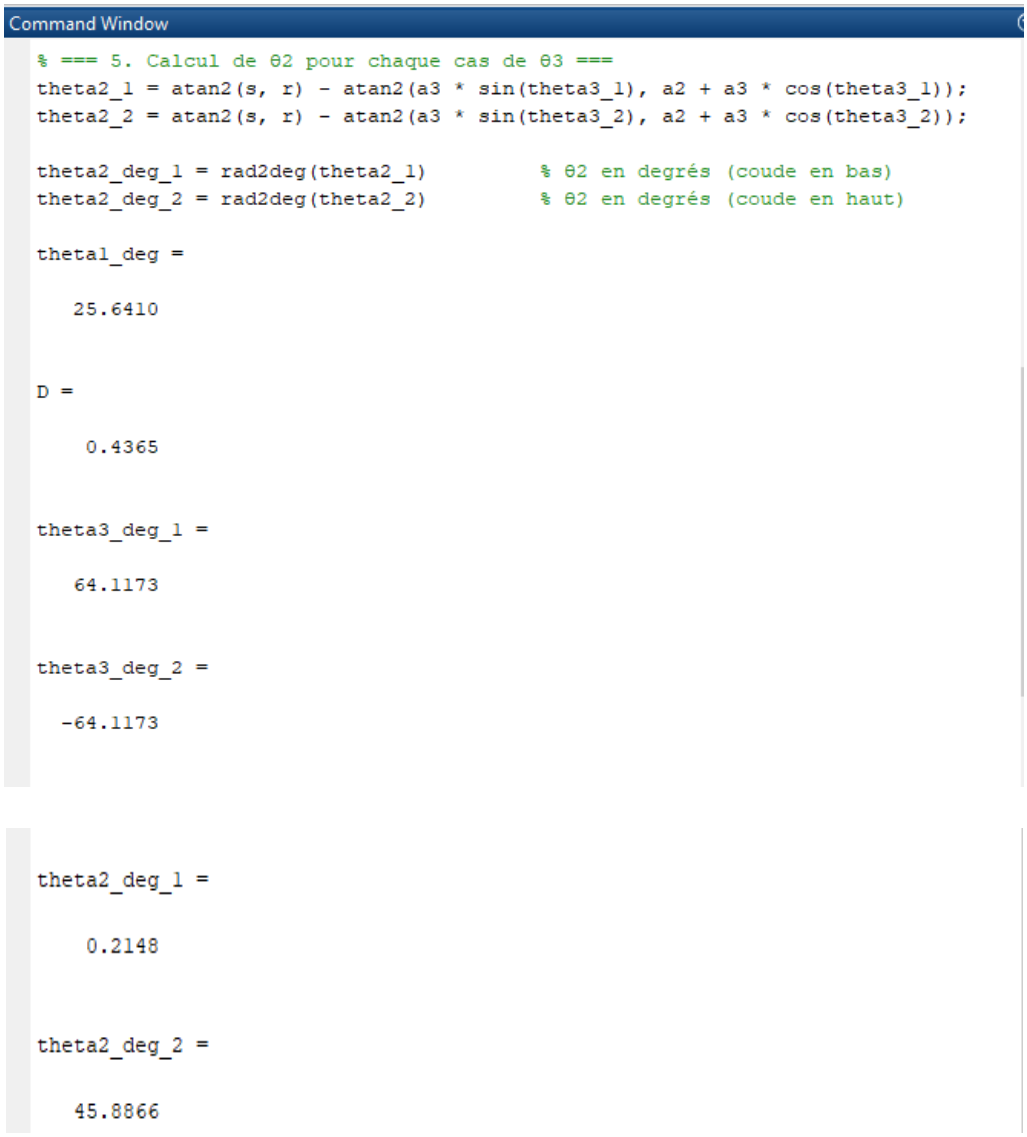


Figure III.14 : Code MATLAB pour calculer les angles θ1 θ2 et θ3

De la même manière en calculant $\theta_1'\theta_2'\theta_3'$ on obtient :

θ_1 =25.64°	θ_2 =45.88°	θ_3 =-64.11°
θ_1' =-32.61°	θ_2' =40.07°	θ_3' =-50.84°

Le code arduino

Le code ci-dessous constitue le programme final utilisé sur la carte Arduino. Il permet de déplacer le bras robotique vers les positions A et B en se basant sur les angles préalablement calculés. De plus, il offre la fonctionnalité de revenir à la position initiale

Partie 1

Cette partie représente la phase d'initialisation matérielle du bras robotique, incluant la configuration des broches de commande des moteurs, ainsi que la définition des paramètres essentiels (angles cibles, caractéristiques des moteurs et rapports de réduction) nécessaires à l'exécution des mouvements.

RRR_code

```
#include <AccelStepper.h>

// === Définition des broches ===
#define STEPPER1_STEP_PIN 2
#define STEPPER1_DIR_PIN 5
#define STEPPER2_STEP_PIN 3
#define STEPPER2_DIR_PIN 6
#define STEPPER3_STEP_PIN 4
#define STEPPER3_DIR_PIN 7

// === Initialisation des moteurs ===
AccelStepper stepper1(AccelStepper::DRIVER, STEPPER1_STEP_PIN, STEPPER1_DIR_PIN);
AccelStepper stepper2(AccelStepper::DRIVER, STEPPER2_STEP_PIN, STEPPER2_DIR_PIN);
AccelStepper stepper3(AccelStepper::DRIVER, STEPPER3_STEP_PIN, STEPPER3_DIR_PIN);

// === Angles pour les positions A et B ===
const float THETA1_A = 25.4;
const float THETA2_A = 45.88;
const float THETA3_A = -64.11;

const float THETA1_B = -32.61;
const float THETA2_B = 40.07;
const float THETA3_B = -50.84;

// === Paramètres moteurs ===
const int STEPS_PER_REV = 200;
const int MICROSTEPS = 16;



// === Paramètres moteurs ===
const int STEPS_PER_REV = 200;
const int MICROSTEPS = 16;

// === Gear ratios pour chaque moteur ===
// Déduit du fait que 1 tour moteur (360°) produit une rotation de 40° au niveau du joint.
// Donc, Gear Ratio = 360 / 40 = 9
const float GEAR_RATIO1 = 9.0;
const float GEAR_RATIO2 = 9.0;
const float GEAR_RATIO3 = 9.0;
```

Figure III.15 : La configuration matérielle du robot .

Partie 2

Cette section agit comme une interface de commande, permettant à l'utilisateur d'interagir avec le robot à travers des instructions envoyées via le port série. Le robot attend un caractère de commande spécifique pour effectuer un mouvement prédéfini :

- 'A' → mouvement vers la position A.
- 'B' → mouvement vers la position B.
- 'R' → retour à la position d'origine (0,0,0).

RRR_code

```
void setup() {
  Serial.begin(9600);

  stepper1.setMaxSpeed(1000);
  stepper1.setAcceleration(500);
  stepper2.setMaxSpeed(1000);
  stepper2.setAcceleration(500);
  stepper3.setMaxSpeed(1000);
  stepper3.setAcceleration(500);

  Serial.println("Système prêt. Commandes : A / B / R");
}

void loop() {
  if (Serial.available()) {
    char cmd = Serial.read();
    if (cmd == 'A') moveToPositionA();
    else if (cmd == 'B') moveToPositionB();
    else if (cmd == 'R') returnToOrigin();
  }
}
```

Figure III.16 : la partie de commande dans le code

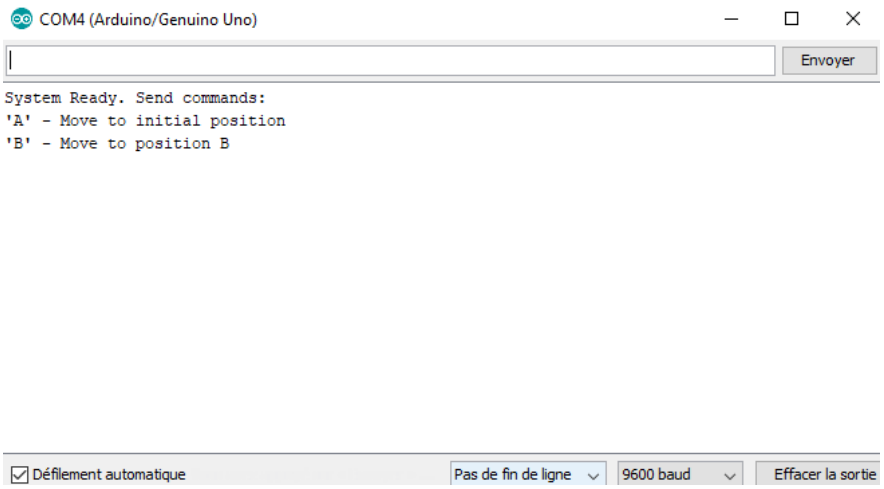


Figure III.17 : l'interface de commande dans le code (moniteur série)

Partie 03

Dans cette partie, le bras robotique exécute physiquement le déplacement vers la position souhaitée en utilisant les paramètres définis précédemment.

RRR_code

```
void moveToPositionA() {
  long s1 = degreesToSteps(THETA1_A, GEAR_RATIO1);
  long s2 = degreesToSteps(THETA2_A, GEAR_RATIO2);
  long s3 = degreesToSteps(THETA3_A, GEAR_RATIO3);
  stepper1.moveTo(s1);
  stepper2.moveTo(s2);
  stepper3.moveTo(s3);
  runAllMotors();
  Serial.println("Position A atteinte.");
}

void moveToPositionB() {
  long s1 = degreesToSteps(THETA1_B, GEAR_RATIO1);
  long s2 = degreesToSteps(THETA2_B, GEAR_RATIO2);
  long s3 = degreesToSteps(THETA3_B, GEAR_RATIO3);
  stepper1.moveTo(s1);
  stepper2.moveTo(s2);
  stepper3.moveTo(s3);
  runAllMotors();
  Serial.println("Position B atteinte.");
}
```

Figure III.18 : La partie qui représente le mouvement vers les point disérer

Partie 04

Dans cette partie, le bras robotique revient automatiquement à sa position d'origine, en réinitialisant chaque moteur. On y trouve également les fonctions qui permettent de transformer les angles de rotation en pas moteur, afin d'assurer un mouvement fluide et précis.

```
void returnToOrigin() {
  stepper1.moveTo(0);
  stepper2.moveTo(0);
  stepper3.moveTo(0);
  runAllMotors();
  Serial.println("Retour à la position d'origine.");
}

void runAllMotors() {
  while (stepper1.distanceToGo() != 0 ||
         stepper2.distanceToGo() != 0 ||
         stepper3.distanceToGo() != 0) {
    stepper1.run();
    stepper2.run();
    stepper3.run();
  }
}

long degreesToSteps(float degrees, float gearRatio) {
  return (degrees * gearRatio / 360.0) * STEPS_PER_REV * MICROSTEPS;
}
```

Figure III .19 : la représentation des fonctions

5. Difficultés techniques rencontrées et solutions apportées

Lors de la réalisation du bras robotique, plusieurs défis techniques ont été identifiés, tant au niveau de l'impression 3D que de l'électronique :

- **Impression 3D instable :** Certaines pièces se déformaient ou se détachaient du plateau durant l'impression en raison d'une mauvaise adhérence. Cela a nécessité plusieurs réimpressions, accompagnées d'ajustements des paramètres et de l'utilisation d'une colle pour améliorer la fixation comme il montré sur la **(Figure III.14)**

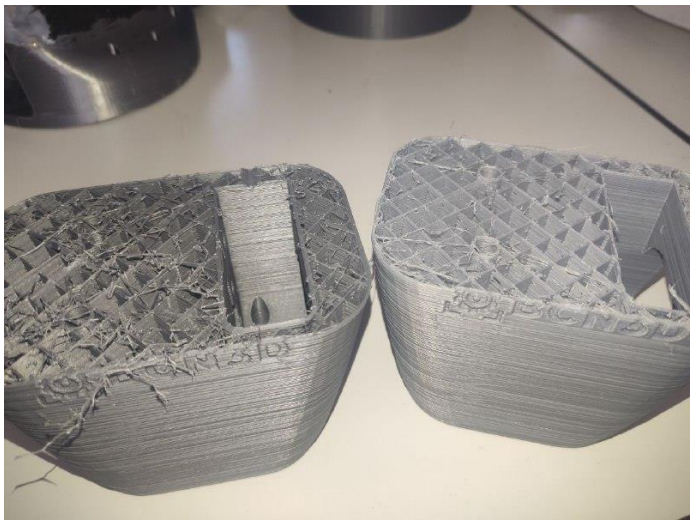


Figure III.20 : Déformation des pièces imprimé

- **Durée d'impression prolongée :** La fabrication des pièces, surtout les plus grandes, était très longue. Les échecs d'impression ralentissaient significativement le progrès du projet.
- **Problème au niveau du thermistor :** Le thermistor a surchauffé et grillé pendant l'impression, interrompant le processus. Son remplacement était crucial pour éviter des risques de surchauffe.la figure III.5 de l'ecrant du l'imprimante afficher l'erreur de ce cas

Figure III.21 : Erreur thermistor brûlé suite à une surchauffe

- **Alimentation électrique insuffisante :** Les moteurs pas à pas manquaient de puissance et étaient instables à cause d'une alimentation inadaptée, nécessitant des améliorations pour assurer une performance fiable.
- **Fragilité mécanique :** Certaines pièces se fissuraient ou se cassaient sous contrainte, nécessitant des réimpressions avec un taux de remplissage plus élevé et des matériaux plus résistants.comme illustré dans la(figure III.16)

Fidure III.22 : Casse mécanique au niveau des articulations

- **Manque de précision dans les mouvements :** Les moteurs perdaient en précision lors des mouvements descendants, en raison du poids des charges et de l'absence de système de rétroaction, rendant difficile la correction automatique de la position.

Solutions apportées

- Finalisation du montage des pièces restantes du robot.
- Amélioration des réglages d'impression pour éviter les défauts et réimpression des pièces concernées.
- Utilisation de moteurs plus puissants ou servo-moteurs intelligents pour une meilleure précision et charge utile.
- Ajout de capteurs de retour (encodeurs).
- Intégration d'un système de vision pour détecter les objets.
- Ajout d'un contrôle sans fil (Bluetooth/Wi-Fi).
- Utilisation d'algorithmes intelligents pour optimiser les mouvements.
- Intégration d'une alimentation autonome.

6. Conclusion :

Dans cette chapitre nous avons présenter La réalisation pratique de ce projet en passant par la fabrication et l'assemblage des pièces mécanique et électronique ainsi que la progra-mmation. Malgré les difficultés rencontrées, des solutions efficaces ont été mises en œuvre, et le bras robotique a pu être assemblé et testé avec succès. Ce travail ouvre la voie à plusieurs améliorations futures pour rendre le système plus performant, autonome et intelligent.

Conclusion générale

Conclusion générale

Ce projet permet de s'introduire à la robotique, En partant de notions théoriques sur les différents types de robots, leurs fonctions et leurs composants, nous avons progressivement construit un projet concret, en mettant en œuvre un bras robotique capable d'exécuter des mouvements précis vers des positions définies dans l'espace.

À travers cette réalisation, nous avons renforcé notre compréhension de la modélisation géométrique directe et inverse, ainsi que de la cinématique du robot . Ces modèles ont été développés et simulés dans l'environnement MATLAB, grâce à une interface graphique conviviale, ce qui nous a permis d'observer le comportement du robot dans un espace tridimensionnel. L'interaction entre les équations mathématiques et les mouvements physiques nous a donné une compréhension approfondie du rapport entre la théorie et la pratique

Tout au long de ce projet, nous avons rencontré plusieurs défis, qu'ils soient d'ordre théorique, technique ou pratique liés à la modélisation, à la fabrication ou à l'assemblage . Mais ces obstacles ont été autant d'opportunités d'apprentissage. Grâce à une approche progressive, nous avons modéliser le robot, et aussi assembler ses différentes parties, programmer ses mouvements, et tester son fonctionnement.

Ce projet n'est qu'un point de départ dans le développement de notre bras robotique . Il reste de nombreuses pistes d'amélioration que se soit sur le plan matériel, logicielle ou fonctionelle

Références et bibliographies

Références et bibliographies :

[1] D. Guthleben, “Le robot,” in La fabuleuse histoire des inventions : De la maîtrise du feu à l’immortalité, Paris, Dunod, 2021, pp. 244–245. ROBOTIQ DEF ROBO

[2] Dictionnaire Larousse, 2008, Edition en ligne DEFE ROBO

[3] EYRAUD, Charles-Henri. « Horloges astronomiques au tournant du XVIIIe siècle : de l'à peu-près à la précision ». Thèse de doctorat. Lyon 2, 2004. HISTORIQU

[4] NESSAH MOULOUD DEHANI NABIL Mémoire de Fin d’Etude « Utilisation d’un contrôleur à base du flou adaptatif pour commander un système non linéaire (application bras manipulateur à 2ddl) »

[5] BOUZIANE Fatima Zohra “Rétro-conception du bras horizontal de robot manipulateur de la cellule flexible (Tlemcen) »

[6] R. Gourdeau, “ELE4203 — Robotique : Modélisation des Robots Manipulateurs”,Département de génie ´électrique, Ecole Polytechnique de Montréal, 4 novembre 2010

[7] AFNOR, Cours de robotique industrielle, Centre Universitaire Amine El-Okkal, Tamanrasset, Algérie.

[8] Jean-Louis Boimond , ISTIA, Université Angers : cours de robotique

Site : https://fr.slideshare.net/souissi2013/cours-robotique-27452132

[9] N. Khellil, “Étude et Simulation d’un Système de Navigation d’un Robot Mobile Autonome via des Capteurs sans Fil,” Mémoire de Master, Univ. Blida 1, Blida, Algérie, 2011.

[10] M. Hazerchi and M. Lamine Tazir, “Planification de la trajectoire d'un robot mobile autonome dans un environnement dynamique,” Mémoire de Master, École Nationale Polytechnique, Alger, Algérie, 2012..

[11] Texas Instruments DRV8825 Datasheet (2015)

[12] R. M. C. Santos, Design and validation of a five degrees of freedom robotic arm prototype for car industry test application, Master’s dissertation, Faculdade de Engenharia da Universidade do Porto, 2019. Available: https://hdl.handle.net/10216/126046

[13] BAHLOUL Abdelkrim « Sur la commande des robots manipulateurs industriels en comanipulation robotique », Thèse de Doctorat, Université Paris-Saclay, 7 décembre 2018

[14] F. Morbidi, "Initiation à la Robotique," Licence Professionnelle Automatisme et Robotique, Session 2022, Amiens, Laboratoire MIS, Équipe Perception Robotique, Université de Picardie Jules Verne.

[15] Toglefritz, "Marlin firmware for BCN3D Moveo robot arm", GitHub, [Online]. Available: https://github.com/Toglefritz/Bertram_Robot_Arm/tree/master/Firmware/Marlin_BCN3D_Moveo. [Accessed: 1-Mars-2025].

[16] F .Z. Zndaoui, "conception et réalisation d'un robot manipulateur à 4 d.d .l dédie à la télé-Opération en réalitée augmentée", thèse de Doctorat en sciences, ecole nationale polythechnique, Alger, Algérie, 2020

[17] Dorna Robotics, "Types of Robotic Arms", Dorna Blog, 2023. [Online]. Available: https://dorna.ai/blog/types-of-robotic-arms/. [Accessed: 25-Mai-2025].

[18] TME, "Moteur pas à pas – types et exemples d'applications des moteurs pas à pas", TME Library Articles, 2021. [Online]. Available: https://www.tme.eu/fr/news/library-articles/page/41861/Moteur-pas-a-pas-types-et-exemples-dapplications-des-moteurs-pas-a-pas/. [Accessed: 11-Jun-2025].

[19] S. Moumeni, "Modélisation, étude et simulation d'un robot manipulateur," Mémoire de fin d'études, Promotion 2003, École Nationale Polytechnique, Alger.

[20] J-P. LALLEMAND & S. ZEGHLOUL Robotique modélisation mécanique CAO robotique commande édition MASSON Paris Milan .

[21] P. Corke, Robotics Toolbox for MATLAB, [en ligne]. Disponible : http://petercorke.com/wordpress/toolboxes/roboticstoolbox. Consulté en juin 2025.

Annex

Le code matlab pour la simulation du MGD et MGI :

```
function robot_gui()
    % Interface graphique pour robot RRR avec angles en degr?s
    % Cr?er la figure principale
    f = figure('Name','Interface Robot RRR','Position',[200 100 800 600],...
        'Resize','off','MenuBar','none','NumberTitle','off');

    % === Param?tres du robot ===
    d1 = 0.232;  % hauteur du premier joint
    a2 = 0.221;  % longueur du deuxi?me lien
    a3 = 0.13;   % longueur du troisi?me lien

    % === Cr?ation des contr?les UI ===
    % Panel pour les entr?es
    input_panel = uipanel('Title','Joint Control','Position',[0.02 0.6 0.35
0.35]);

    % ?tiquettes et champs des angles
    uicontrol('Parent',input_panel,'Style','text','Position',[20 120 80
20],...
        'String','Angle 1
(deg):','FontSize',12,'HorizontalAlignment','left');
    edit_q1 = uicontrol('Parent',input_panel,'Style','edit','Position',[100
120 100 25],...
        'String','0','FontSize',11,'BackgroundColor','white');

    uicontrol('Parent',input_panel,'Style','text','Position',[20 80 80
20],...
        'String','Angle 2
(deg):','FontSize',12,'HorizontalAlignment','left');
    edit_q2 = uicontrol('Parent',input_panel,'Style','edit','Position',[100
80 100 25],...
        'String','0','FontSize',11,'BackgroundColor','white');

    uicontrol('Parent',input_panel,'Style','text','Position',[20 40 80
20],...
        'String','Angle 3
(deg):','FontSize',12,'HorizontalAlignment','left');
    edit_q3 = uicontrol('Parent',input_panel,'Style','edit','Position',[100
40 100 25],...
        'String','0','FontSize',11,'BackgroundColor','white');

    % Boutons
    uicontrol('Parent',input_panel,'Style','pushbutton','String','Forward
Kinematics',...
        'Position',[20 5 100 30],'FontSize',10,'BackgroundColor',[0.3 0.7
0.3],...
        'ForegroundColor','white','Callback',@forward_callback);

    uicontrol('Parent',input_panel,'Style','pushbutton','String','Inverse
Kinematics',...
        'Position',[130 5 100 30],'FontSize',10,'BackgroundColor',[0.7 0.3
0.3],...
        'ForegroundColor','white','Callback',@inverse_callback);

    % Panel pour les r?sultats
    result_panel = uipanel('Title','Results','Position',[0.02 0.02 0.35
0.55]);
```

```
    txt_result = uicontrol('Parent',result_panel,'Style','text',...
        'Position',[10 10 260 250],'String','Results will appear here...',...
        'FontSize',11,'HorizontalAlignment','left',...
        'BackgroundColor',[0.95 0.95 0.95]);

    % Panel pour la visualisation 3D
    plot_panel = uipanel('Title','3D Visualization','Position',[0.4 0.02 0.57
0.93]);

    % === D?finition du robot avec Robotics Toolbox ===
    try
        % V?rifier si Robotics Toolbox est disponible
        L1 = Link([0 d1 0 -pi/2], 'standard');
        L2 = Link([0 0 a2 0], 'standard');
        L3 = Link([0 0 a3 0], 'standard');

        % Noms des joints
        L1.name = 'Joint 1';
        L2.name = 'Joint 2';
        L3.name = 'Joint 3';

        Rob = SerialLink([L1 L2 L3], 'name','Robot RRR');

        % Plot initial dans le panel
        axes_handle = axes('Parent',plot_panel);
        Rob.plot([0 0 0], 'workspace', [-0.6 0.6 -0.6 0.6 0 0.6],...
            'scale', 0.5, 'trail', 'r-');

        robotics_available = true;

    catch
        robotics_available = false;
        % Cr?er un axe pour affichage manuel si Robotics Toolbox n'est pas
disponible
        axes_handle = axes('Parent',plot_panel);
        plot_manual_robot([0 0 0]);
        set(txt_result, 'String', 'Note: Robotics Toolbox not detected. Using
manual plotting.');
    end

    %% === Fonction Cin?matique Directe ===
    function forward_callback(~,~)
        try
            % Lire les angles
            q1 = str2double(get(edit_q1, 'String'));
            q2 = str2double(get(edit_q2, 'String'));
            q3 = str2double(get(edit_q3, 'String'));

            % V?rifier la validit? des entr?es
            if isnan(q1) || isnan(q2) || isnan(q3)
                set(txt_result, 'String', 'Error: Please enter valid numeric
values.');
                return;
            end

            % Convertir en radians
            q_rad = deg2rad([q1 q2 q3]);

            % Calculer la position finale
```

```
            if robotics_available
                T = Rob.fkine(q_rad);
                pos = T(1:3,4);
                Rob.plot(q_rad);
            else
                pos = forward_kinematics_manual(q_rad);
                plot_manual_robot(q_rad);
            end

            % Afficher les r?sultats
            result_text = sprintf(['=== FORWARD KINEMATICS ===\n\n' ...
                'Input angles:\n' ...
                'Angle 1 = %.1f deg\n' ...
                'Angle 2 = %.1f deg\n' ...
                'Angle 3 = %.1f deg\n\n' ...
                'End-effector position:\n' ...
                'X = %.3f m\n' ...
                'Y = %.3f m\n' ...
                'Z = %.3f m\n\n' ...
                'Distance from center = %.3f m'], ...
                q1, q2, q3, pos(1), pos(2), pos(3), norm(pos));

            set(txt_result, 'String', result_text);

        catch ME
            set(txt_result, 'String', ['Error during calculation: '
ME.message]);
        end
    end

    %% === Fonction Cin?matique Inverse ===
    function inverse_callback(~,~)
        try
            % Demander la position d?sir?e
            answer = inputdlg({'X (m):','Y (m):','Z (m):'}, ...
                'Desired end-effector position', [1 50],
{'0.3','0.0','0.2'});

            if isempty(answer)
                return;
            end

            xc = str2double(answer{1});
            yc = str2double(answer{2});
            zc = str2double(answer{3});

            % V?rifier la validit? des entr?es
            if isnan(xc) || isnan(yc) || isnan(zc)
                set(txt_result, 'String', 'Error: Please enter valid numeric
values.');
                return;
            end

            % Calcul de la cin?matique inverse
            [q_deg, success, error_msg] = inverse_kinematics_manual(xc, yc,
zc);

            if ~success
                set(txt_result, 'String', ['Error: ' error_msg]);
                return;
            end
```

```
            % Mettre ? jour les champs d'entr?e
            set(edit_q1, 'String', sprintf('%.1f', q_deg(1)));
            set(edit_q2, 'String', sprintf('%.1f', q_deg(2)));
            set(edit_q3, 'String', sprintf('%.1f', q_deg(3)));

            % Visualiser la configuration
            q_rad = deg2rad(q_deg);
            if robotics_available
                Rob.plot(q_rad);
            else
                plot_manual_robot(q_rad);
            end

            % Afficher les r?sultats
            result_text = sprintf(['=== INVERSE KINEMATICS ===\n\n' ...
                'Desired position:\n' ...
                'X = %.3f m\n' ...
                'Y = %.3f m\n' ...
                'Z = %.3f m\n\n' ...
                'Calculated angles:\n' ...
                'Angle 1 = %.1f deg\n' ...
                'Angle 2 = %.1f deg\n' ...
                'Angle 3 = %.1f deg\n\n' ...
                'Solution found successfully!'], ...
                xc, yc, zc, q_deg(1), q_deg(2), q_deg(3));

            set(txt_result, 'String', result_text);

        catch ME
            set(txt_result, 'String', ['Error during calculation: '
ME.message]);
        end
    end

    %% === Cin?matique directe manuelle ===
    function pos = forward_kinematics_manual(q)
        % Calcul de la position de l'effecteur
        c1 = cos(q(1)); s1 = sin(q(1));
        c2 = cos(q(2)); s2 = sin(q(2));
        c23 = cos(q(2) + q(3)); s23 = sin(q(2) + q(3));

        x = c1 * (a2*c2 + a3*c23);
        y = s1 * (a2*c2 + a3*c23);
        z = d1 + a2*s2 + a3*s23;

        pos = [x; y; z];
    end

    %% === Cin?matique inverse manuelle ===
    function [q_deg, success, error_msg] = inverse_kinematics_manual(xc, yc,
zc)
        success = false;
        error_msg = '';
        q_deg = [0 0 0];

        % Calcul de ?1
        theta1 = atan2(yc, xc);

        % Calcul de r et s
```

```
        r = sqrt(xc^2 + yc^2);
        s = zc - d1;

        % V?rifier si la position est dans l'espace de travail
        max_reach = a2 + a3;
        min_reach = abs(a2 - a3);
        current_reach = sqrt(r^2 + s^2);

        if current_reach > max_reach
            error_msg = sprintf('Position too far (%.3f m > %.3f m)',
current_reach, max_reach);
            return;
        end

        if current_reach < min_reach
            error_msg = sprintf('Position too close (%.3f m < %.3f m)',
current_reach, min_reach);
            return;
        end

        % Calcul de ?3
        D = (r^2 + s^2 - a2^2 - a3^2) / (2*a2*a3);

        if abs(D) > 1
            error_msg = 'Position unreachable (impossible configuration)';
            return;
        end

        theta3 = atan2(sqrt(1 - D^2), D);  % Solution coude en bas

        % Calcul de ?2
        theta2 = atan2(s, r) - atan2(a3*sin(theta3), a2 + a3*cos(theta3));

        % Conversion en degr?s
        q_deg = rad2deg([theta1, theta2, theta3]);
        success = true;
    end

    %% === Trac? manuel du robot ===
    function plot_manual_robot(q)
        cla(axes_handle);

        % Calcul des positions des joints
        c1 = cos(q(1)); s1 = sin(q(1));
        c2 = cos(q(2)); s2 = sin(q(2));
        c23 = cos(q(2) + q(3)); s23 = sin(q(2) + q(3));

        % Positions des joints
        P0 = [0; 0; 0];                    % Base
        P1 = [0; 0; d1];                   % Joint 1
        P2 = [c1*a2*c2; s1*a2*c2; d1 + a2*s2];  % Joint 2
        P3 = [c1*(a2*c2 + a3*c23); s1*(a2*c2 + a3*c23); d1 + a2*s2 + a3*s23];
% Effecteur

        % Trac? du robot
        hold(axes_handle, 'on');

        % Liens
        plot3(axes_handle, [P0(1) P1(1)], [P0(2) P1(2)], [P0(3) P1(3)], 'b-',
'LineWidth', 3);
```

```
        plot3(axes_handle, [P1(1) P2(1)], [P1(2) P2(2)], [P1(3) P2(3)], 'r-',
'LineWidth', 3);
        plot3(axes_handle, [P2(1) P3(1)], [P2(2) P3(2)], [P2(3) P3(3)], 'g-',
'LineWidth', 3);

        % Joints
        plot3(axes_handle, P0(1), P0(2), P0(3), 'ko', 'MarkerSize', 8,
'MarkerFaceColor', 'k');
        plot3(axes_handle, P1(1), P1(2), P1(3), 'bo', 'MarkerSize', 8,
'MarkerFaceColor', 'b');
        plot3(axes_handle, P2(1), P2(2), P2(3), 'ro', 'MarkerSize', 8,
'MarkerFaceColor', 'r');
        plot3(axes_handle, P3(1), P3(2), P3(3), 'go', 'MarkerSize', 8,
'MarkerFaceColor', 'g');

        % Configuration de l'axe
        axis(axes_handle, 'equal');
        grid(axes_handle, 'on');
        xlabel(axes_handle, 'X (m)');
        ylabel(axes_handle, 'Y (m)');
        zlabel(axes_handle, 'Z (m)');
        title(axes_handle, 'Robot RRR');
        xlim(axes_handle, [-0.6 0.6]);
        ylim(axes_handle, [-0.6 0.6]);
        zlim(axes_handle, [0 0.6]);
        view(axes_handle, 3);

        hold(axes_handle, 'off');
    end

    % Message d'initialisation
    set(txt_result, 'String', sprintf(['RRR Robot Interface initialized\n\n'
...
        'Robot parameters:\n' ...
        'd1 = %.3f m\n' ...
        'a2 = %.3f m\n' ...
        'a3 = %.3f m\n\n' ...
        'Max reach = %.3f m\n' ...
        'Min reach = %.3f m\n\n' ...
        'Enter angles and click\n' ...
        '"Forward Kinematics" or use\n' ...
        '"Inverse Kinematics".'], ...
        d1, a2, a3, a2+a3, abs(a2-a3)));

end
```