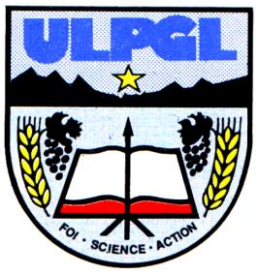
# UNIVERSITE LIBRE DES PAYS DES GRANDS LACS

# FACULTE DE SCIENCES ET TECHNOLOGIES APPLIQUEES

**DEPARTEMENT DE GENIE MECANIQUE**



BP. 368 GOMA

www.ulpgl.net


# ÉLABORATION ET MISE EN ŒUVRE D'UN BRAS ROBOTISÉ MOBILE POUR LA MANUTENTION ET LE TRANSPORT DES CHARGES


Par :

1. **ASIFIWE VAHVIRAKI DIEUDONNE**
2. **SALAMA KAHINDO MERVEILLE**


Travail présenté en vue de l'obtention du Diplôme de bachelor en Sciences de l'ingénieur

**Option** : Génie Mécanique

**Directeur** : Prof. BARAKA MUSHAGE

**ANNÉE ACADÉMIQUE 2024-2025**

# Epigraphe

« Donnez-lui les yeux, donnez-lui des membres, donnez-lui l’intelligence du mouvement, et l'objet deviendra un acteur. »

**Anonyme**

# Dédicace

Nous dédions ce travail à nos parents pour avoir toujours encouragé notre curiosité et à ceux qui inventerons le futur.

# Remerciements

Nos remerciements vont en premier au Dieu tout puissant pour le courage, la sante et la patience qu'il nous a donnés durant toutes ces longues années d'études et qui nous ont permis d'accomplir aujourd'hui ce modeste travail.

A notre Doyen de faculté et Directeur de notre travail le Professeur Docteur Ingénieur BARAKA MUSHAGE Olivier pour son orientation, ses conseils et le partage de ses connaissances durant toutes nos recherches et notre parcours académique. Nous tenons également à adresser nos remerciements a tout le corps enseignant de la faculté de science et technologie de l'ULPGL.

**Merveille** : je tiens à exprimer ma profonde gratitude à mes très chers parents Ladislas NDAKOLA et Régine NDAMWENGE pour leurs sacrifices inestimables et leurs amour inconditionnel ; à mes frères pour leurs soutiens et conseil constant ; à ma petite sœur pour sa présence réconfortante et ses encouragement sincère, à ma grand-mère Stella pour son encouragement, à mes oncles, mes tantes, à mes cousins et cousines pour leurs soutient moral tout au long de mon parcours académique.

**Dieudonné** tient à remercier ses très chers parents Kakule VAHAVIRAKI et Bernadette KAYUNGO pour avoir rendu ce parcours possible avec leurs soutient indéfectibles et leurs amours. A ses frères et sœurs pour l'avoir encouragé et le pousser à aller de l'avant.

A tous nos camarades et amis pour leurs soutiens et les échanges enrichissants tout au long de ce travail.

Enfin nous nous remercions mutuellement pour avoir partagé ce projet dans le respect, la bonne humeur et l'entraide, et pour avoir rendu cette expérience à la fois précieuse et inoubliable.

# Résumé

Ce travail porte sur **l'élaboration et la mise en œuvre d'un bras robotisé mobile** conçu pour optimiser les tâches de manutention et de transport de charges dans les milieux industriels et logistiques. La problématique centrale aborde la pénibilité des tâches manuelles et les limites des robots fixes traditionnels face à des environnements de travail dynamiques et semi-structurés

L'objectif général est de concevoir et de réaliser un système mécatronique combinant une plateforme de locomotion et un manipulateur, capable de déplacer et de déposer des charges avec précision tout en garantissant la sécurité des opérateurs. Pour ce faire, nous avons adopté une méthodologie base sur une étude théorique et documentaire sur la robotique et les manipulateurs mobiles, une phase de conception mécanique et électronique et phase de réalisation avec des matériaux accessibles à moindre coût et de validation expérimental du prototype.

Les résultats de la modélisation et des simulations confirment la capacité du robot à atteindre un espace de travail défini et validé la résistance structurelle. Les essais expérimentaux confirment la capacité du robot à se déplacer de manière stable et de manipuler une charge avec une bonne précision. Le système est piloté par un microcontrôleur **ESP32-CAM**, intégrant des capteurs pour la perception de l'environnement, offrant ainsi une solution à faible coût et adaptable aux besoins des ateliers et entrepôts locaux de la région du Kivu.
Mots clés : robotique mobile, bras robotisé, manutention, conception mécanique, modélisation cinématique.

# Abstract

This work focuses on the **development and implementation of a mobile robotic arm** designed to optimize handling tasks and load transport within industrial and logistical environments. The central problem addresses the physical strain of manual labor and the limitations of traditional fixed robots when faced with dynamic and semi-structured workspaces.

The general objective is to design and build a mechatronic system combining a locomotion platform and a manipulator, capable of moving and placing loads with precision while ensuring operator safety. To achieve this, we adopted a methodology based on a theoretical and literature review of robotics and mobile manipulators, a mechanical and electronic design phase, and a realization phase using low-cost accessible materials, followed by experimental validation of the prototype.

Modeling and simulation results confirm the robot's ability to reach a defined workspace and validate its structural integrity. Experimental tests confirm the robot's ability to move stably and manipulate loads with high precision. The system is controlled by an ESP32-CAM microcontroller, integrating sensors for environmental perception, thus offering a cost-effective and adaptable solution for the needs of local workshops and warehouses in the Kivu region.

Keywords: mobile robotics, robotic arm, material handling, mechanical design, kinematic modeling.

# Table des matières

## Table des matières

# Liste des abréviations

| | |
|---|---|
| Ddl | Degré de liberté |
| MCD | Modélisation Cinématique Directe |
| MCI | Modélisation Cinématique Inverse |
| DC | Courant Continu |
| PMW | Modulation de largeur d'impulsion (Pulse-Width-Modulation) |
| IMU | Unité de mesure inertiel (Inertial Measurement Unit) |
| LiDAR | Télédétection par laser (Light Detection and Randing) |
| SLAM | Cartographie et localisation simultanées (Simultaneous Localization and Mapping) |

# Liste des tableaux

# Liste des figures

# Chapitre 0 Introduction générale

## 0.1 Contexte

Dans un monde en constante évolution, la logistique joue un rôle stratégique au cœur des chaînes de production et de distribution. L'efficacité avec laquelle les matériaux, les pièces et les produits finis sont déplacés, stockés et livrés conditionne directement la performance industrielle et commerciale des entreprises. Face à la croissance des flux, à la diversité des tâches de manutention et aux exigences de rapidité et de flexibilité, les solutions traditionnelles atteignent leurs limites [1].

C'est dans ce contexte que l'automatisation s'impose comme une réponse incontournable. Initialement limitée aux processus fixes et répétitifs, elle s'est progressivement étendue à des domaines plus dynamiques grâce aux avancées technologiques. Parmi ces innovations, la robotique mobile et de manipulation occupe une place de choix, apportant une nouvelle dimension à la logistique moderne : celle de l'intelligence embarquée, de l'adaptabilité et de l'autonomie opérationnelle [2].

## 0.1 Identification et formulation du problème

Dans les secteurs de la logistique, la manutention industrielle et de la production, les tâches de déplacement et de manipulation de charges sont largement effectuées manuellement. Une grande partie de ces tâches sont contraignante, répétitive et éprouvante physiquement pour les opérateurs humains. Cette situation engendre plusieurs conséquences : fatigue, baisse de la productivité, accident, exposition à des troubles musculo-squelettique, etc. [3]. Les robots industriels traditionnels sont généralement fixes et conçus pour des environnements fortement structurés, tels que les lignes d'assemblage automatisées ou les cellules de production fermées. Leur efficacité repose sur la répétition de tâches précises dans un espace préconfiguré [1]. Cependant, avec la montée en puissance de l'industrie et de la logistique flexible, les environnements de travail évoluent vers des configurations plus dynamiques, imprévisibles et semi-structurées, où les objets, les obstacles et les tâches

varient fréquemment [4]. Dans ce contexte, les systèmes robotisés classiques montrent leurs limites. Il apparaît donc essentiel de développer un système mobile, flexible, fiable et autonome, capable de répondre aux nouveaux besoins industriels, notamment dans le contexte de la région du Kivu, où la promotion de la robotisation de l'industrie s'avère particulièrement importante.

## 0.2 Questions de recherche

Cette problématique nous pousse à nous poser ces questions :

1. Comment peut-on construire un bras robotisé mobile capable de soulever et déplacer des charges dans un espace de travail semi-structuré ?
2. Quelles méthodes peut-on utiliser pour que ce robot se déplace tout seul, évite les obstacles et travaille en sécurité avec des humains autour ?
3. Est-ce qu'un bras robotisé mobile peut vraiment aider à rendre le travail plus rapide et moins fatigant pour les ouvriers dans un environnement industriel ?

## 0.3 Formulation des hypothèses

- **Hypothèse 1** : La conception combinerait une base mobile et un bras robotique. Le robot intégrerait des capteurs qui permettraient la perception de l'environnement, et des algorithmes pour une commande précise et l'optimisation des mouvements.
- **Hypothèse 2** : Le robot utiliserait des capteurs pour la détection d'obstacles et permettrait ainsi une navigation autonome. Un ajout des normes de sécurité garantirait la sécurité des opérateurs, des protocoles d'arrêt d'urgence assurerait la sécurité en toutes circonstances.
- **Hypothèse 3** : Cette solution diminuerait la charge physique des opérateurs, c'est qui augmenterait le rendement et la productivité.

## 0.4 Justification du choix du sujet et motivations

Dans un contexte industriel où la recherche de performance et d'optimisation est devenue indispensable, l'automatisation constitue un levier essentiel pour accroître l'efficacité des processus tout en diminuant la pénibilité des tâches pour les opérateurs. Cependant, la conception et l'intégration de solutions véritablement adaptées aux environnements réels, tels que les ateliers et entrepôts caractérisés par des conditions dynamiques et parfois contraignantes, représentent encore un défi majeur.

Le choix de ce sujet se justifie par la volonté de proposer une réponse concrète à ces enjeux : le développement d'un bras robotique mobile. Ce système vise à améliorer la productivité et la flexibilité des opérations industrielles, tout en contribuant significativement à la réduction des risques d'accidents et de maladies professionnelles pour les travailleurs.

## 0.5 Énoncé des objectifs de recherche

### 0.5.1 L'objectif général

L'objectif principal de ce travail est la conception et la réalisation d'un bras robotise mobile destine à la manutention des charges dans les entrepôt et atelier locale pour y améliorer la productivité et l'efficacité.

### 0.5.2 Les objectifs spécifiques

Plus spécifiquement l'objectif de ce travail est :

- Étudier le fonctionnement d'un robot manipulateur mobile,
- Concevoir une architecture comprenant un bras manipulateur et une plateforme mobile,
- Intégrer à cette architecture des capteurs pour rendre le robot semi-autonome ou autonome, grâce à leur retour d'information,
- Concevoir un système de commande contenant une application mobile.

## 0.6 Méthodologie et délimitation du travail

Pour notre travail de recherche, nous utilisons les techniques et méthodes suivantes :

- La méthode analytique qui nous a permis définir les exigences du système bras manipulateur-base mobile, la conception mécanique, électronique et architecturale du bras manipulateur,
- La technique documentaire qui nous a permis une compréhension générale du fonctionnement de notre manipulateur, et surtout une compréhension spécifique pour chaque composant de celui-ci (actionneurs, pièces mécaniques, capteurs, ...),
- La technique d'expérimentation nous a permis de faire des tests sur chaque composant, sur leur interaction avant de faire l'assemblage finale du système et le testé.

## 0.7 Structure du mémoire/ Subdivision du travail

Le plan de ce travail est le suivant :

- Une introduction générale
- Le premier chapitre qui portera sur la revue de la littérature et de l'état de l'art
- Le second sur la conception du robot manipulateur et sa commande
- Le dernier parlera de la réalisation du bras manipulateur mobile et des résultats
- Nous finirons ensuite avec une conclusion générale

# Chapitre 1 GENERALITES

## 1.1 Introduction

Le domaine de la robotique a connu une évolution considérable au cours des dernières décennies. Elle s'impose comme une solution incontournable pour l'optimisation et l'automatisation de plusieurs tâches.

Dans ce chapitre, nous parlerons de la robotique en général et plus succinctement des manipulateurs mobiles.

## 1.2 Robotique

### 1.2.1 Définition

La robotique est l'ensemble des techniques permettant la conception et la réalisation de machines automatiques ou de robots [5].

D'après l'AFNOR (Association Française pour la Normalisation), un robot est un manipulateur commandé en position, reprogrammable, polyvalent, à plusieurs degrés de liberté, capable de manipuler des matériaux, des pièces, des outils et des dispositifs spécialisés, au cours de mouvements variables et programmés pour l'exécution d'une variété de tâches [6].

Un robot est un système qui exécute un certain nombre de fonctions délicates et ponctuelles d'une manière précise et sensible qu'un être humain ne peut facilement exécuter en raison

de difficultés d'accès au lieu, des risques encourus, des coûts de la main-d'œuvre humaine ou de la grande précision des tâches [7].

### 1.2.2 Brève historique

Le mot « robot » a été utilisé pour la première fois en 1921 par Karel Capek dans sa pièce R.U.R. (Rossum's Universal Robots). Il provient du mot tchèque robota qui signifie corvée, travail obligatoire. Le terme « robotique » a été employé pour la première fois par Isaac Asimov en 1941 [8].

### 1.2.3 Caractéristiques du robot

Les principales caractéristiques d'un robot reposent sur sa capacité à percevoir son environnement, à traiter des informations et à agir en conséquence. Voici quelques-unes des principales caractéristiques :

- Intelligence artificielle : L'une des caractéristiques les plus importantes des robots est leur capacité à utiliser l'intelligence artificielle (IA) pour effectuer des tâches. L'IA permet aux robots d'apprendre à partir de l'expérience, d'analyser des informations complexes et de prendre des décisions en fonction de ces informations.
- Mobilité : La mobilité est une autre caractéristique essentielle des robots. De nombreux robots sont conçus pour se déplacer, que ce soit sur le sol, dans les airs ou même dans l'eau.
- Interaction avec l'environnement : Les robots sont également capables d'interagir avec leur environnement. Grâce à leurs capteurs, ils peuvent détecter différents types d'informations, tels que la température, la pression, la luminosité, les sons, etc. Ces informations leur permettent de comprendre leur environnement et de réagir en conséquence.
- Communication : Certains robots sont même capables de communiquer avec les humains. Ils peuvent utiliser la parole, les gestes ou les expressions faciales pour transmettre des informations aux personnes avec lesquelles ils interagissent.

- Automatisation des tâches : L'un des principaux avantages des robots est leur capacité à automatiser des tâches répétitives ou dangereuses pour les humains [9].

### 1.2.4 Types de robots

Il n'est pas facile de définir ce que sont les robots, et il n'est pas facile de les catégoriser non plus. Chaque robot a ses propres caractéristiques uniques et, dans l'ensemble, les robots varient énormément en taille, en forme et en capacités. Pourtant, de nombreux robots partagent une variété de fonctionnalités.

Les entreprises de robotique ont regroupé les types de robots en six catégories : les robots mobiles autonomes (AMR), les humanoïdes, les robots articulés, les cobots, les véhicules à guidage automatique (AGV) et les hybrides [10].

1. **Robot mobile autonome (AMR)**

Les robots mobiles autonomes ou Autonomous Mobile Robots sont des véhicules autoguidés. Ils peuvent naviguer sans commande humaine dans un environnement défini et agir en temps réel. Les AMR s'orientent grâce aux plans numériques de l'entrepôt en utilisant des composants optiques et des caméras ultramodernes ainsi que l'intelligence artificielle. Ils détectent ainsi de manière autonome les obstacles et les évitent seuls [11].

*Figure 1-1: AMR [12]*

2. **Les humanoïdes**

Un humanoïde est une forme de robot qui a été conçu pour ressembler et se comporter comme un être humain. Il n'est pas nécessaire la ressemblance soit parfaite, mais les

caractéristiques générales sont généralement conservées. Les humanoïdes peuvent avoir un tronc, une tête, deux bras et deux jambes. Dans certains cas, ils peuvent également avoir un visage, des yeux et une bouche, bien que cela ne soit pas toujours nécessaire. Ils possèdent la capacité de reproduire certaines actions spécifiques aux hommes, comme la marche et la manipulation d'objets. Ils sont utilisés dans les industries, dans les domaines de la santé, dans l'éducation, etc. [10]

*Figure 1-2: Robot humanoïde [13]*

### 3. Les robots articulés

Les robots articulés, encore appelés bras robots, sont les robots industriels les plus utilisés, représentant environ 60 % des installations dans le monde. Ils ressemblent à un bras humain et ont une structure similaire à celle des épaules, des coudes et des poignets. Ils possèdent entre deux et dix articulations, ce qui leur confère la souplesse nécessaire pour accomplir des tâches dynamiques. Le bras peut être attaché à une pince, laquelle est équivalente à une main. La pince peut aller d'une simple ventouse à des options complexes telles que des forets, des lasers et d'autres équipements spécialisés. Ils sont utilisés dans de nombreuses applications, le plus souvent pour l'impression, l'emballage, le soudage, le travail des métaux, l'entretien des machines et la manutention [14].

*Figure 1-3: Robot articulé [15]*

### 4. Les cobots

Les cobots, ou robots collaboratifs, sont des robots conçus pour interagir avec un environnement de travail afin de libérer les opérateurs des tâches les plus répétitives, complexes ou dangereuses. Dans les grands entrepôts ou les centres de distribution, les cobots augmentent l'efficacité de la préparation des commandes en parcourant l'installation et en évitant que les préparateurs de commandes ne quittent leur zone de travail [16].

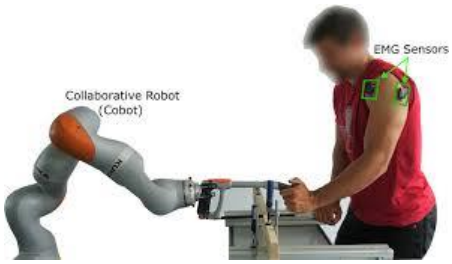



*Figure 1-4: Cobot [17]*

### 5. Véhicule à guidage automatique (AGV)

Un véhicule à guidage automatique ou véhicule autoguidé est un robot qui se déplace de façon autonome sans l'intervention humaine. Les AGV sont le plus souvent utilisés dans des applications industrielles pour déplacer de manière autonome des marchandises dans une usine, un entrepôt ou un atelier. Ainsi, l'expression « chariot automatique » désigne les AGV [18].


*Figure 1-5: AGV [19]*

### 6. Les hybrides

Plusieurs robots sont souvent utilisés pour la création de solutions hybrides qui peuvent assurer les tâches les plus complexes. Exemple d'un AMR qui a la possibilité d'être combiné avec un bras robotisé, ce qui permet d'avoir un robot qui assure la manipulation des colis dans un entrepôt. Les capacités de calcul sont consolidées dans des solutions uniques [11].

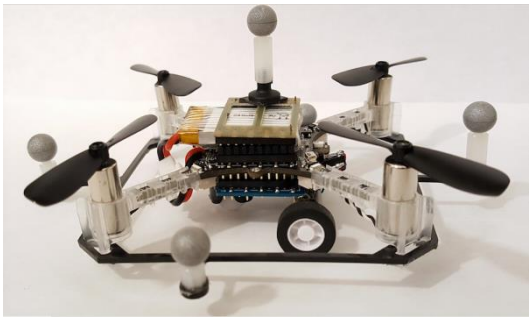


*Figure 1-6: Robot hybride [20]*

## 1.3 Manipulateur mobile

### 1.3.1 Concept Général

Les manipulateurs mobiles sont des robots dont une base mobile embarque un ou plusieurs bras robotisés. Les robots mobiles manipulateurs représentent une avancée significative dans le domaine de la robotique, apportant des solutions efficaces et polyvalentes à divers secteurs. Leurs capacités à effectuer des tâches répétitives, dangereuses et complexes, combinées à leur mobilité autonome, les rendent indispensables pour améliorer l'efficacité et la sécurité dans de nombreuses industries [11].

### 1.3.2 Constitution d'un manipulateur mobile

Un manipulateur mobile comprend un bras robotisé, une plate-forme mobile, ainsi que des capteurs pour la perception et des commandes pour le contrôle des actions, permettant au robot de se déplacer et d'interagir efficacement avec son environnement. Dans cette section, nous allons essayer de développer les grands points suivants : la base mobile, le bras manipulateur, sa perception et ses commandes.

### 1.3.3 La base mobile

Une base mobile est une plate-forme robotique capable de se déplacer (roues, chenilles ou autre système de locomotion), servant de support à d'autres composants tels que capteurs, actionneurs ou bras manipulateur. Elle assure la locomotion et permet de positionner efficacement les organes de perception dans l'environnement

Un robot mobile a besoin d'un mécanisme de locomotion qui lui permette de se déplacer sans restriction dans son environnement. Cependant, il existe une grande variété de façons possibles de se mouvoir et, par conséquent, le choix de l'approche de locomotion d'un robot est un aspect important de sa conception.

1. **Base mobile terrestre** : il en existe plusieurs sortes, à pattes, à chenilles, à roues, etc.

- ***Robot mobile à roues***, ce type de robot est le plus répandu actuellement. Il assure un déplacement avec une accélération et une vitesse rapides, mais nécessite un sol relativement plat. Il existe plusieurs types de robots à roues : uni cycle, différentiel, tricycle, de type voiture, omnidirectionnel et à traction synchrone [21].

*Figure 1-7: Robot mobile à  roues [22]*

- ***Robot mobile à chenilles***, l'utilisation de chenilles présente l'avantage d'une bonne adhérence au sol et d'une faculté de franchissement d'obstacles. L'utilisation est orientée vers l'emploi sur sol accidenté ou de mauvaise qualité au niveau de l'adhérence (présence de boue, herbe, etc. [23].

*Figure 1-8: Robot mobile à chenille [24]*

- ***Robot mobile à pattes***, inspiré de la morphologie d'un animal à quatre pattes, ces quadrupèdes déplacent chaque patte séparément. Il est plus délicat de faire avancer à la fois deux pattes opposées, car le polygone de sustentation devient étroit [25]

*Figure 1-9: Robot à patte [26]*

2. **Base mobile aquatique et aérienne** : elles comprennent des robots volants (multicoptères, voilure fixe), des robots nageurs (propulsion par hélice, locomotion bio-inspirée), etc.

- ***Les multicoptères*** sont des drones a plusieurs rotors (hexa rotors, quadri rotors, …), les voilures fixe sont des avions sans pilote, des drones a ailes, utilises souvent pour la cartographie et des missions longue portée.

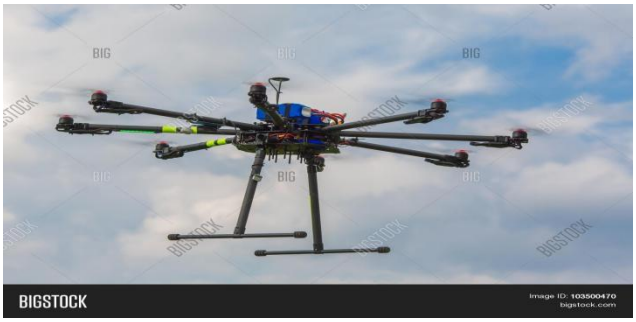



*Figure 1-10: Multicoptère [27]*

- ***Les robots nageurs*** sont soit propulse par hélice (la méthode la plus courante pour les sous-marin autonome) soit bio-inspirée imitant des poissons ou de méduse (souvent utilisé pour des applications discrètes).

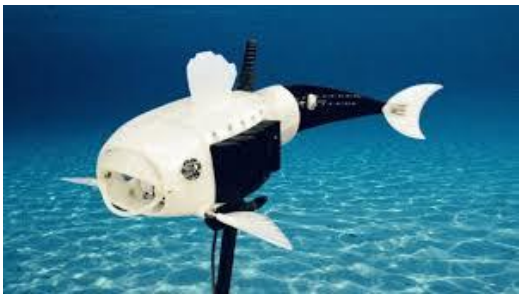


*Figure 1-11: Robot nageur [28]*

3. **Base mobile hybride et innovante** : locomotion modulaire, micro-robots et nano-robots, robot sauterelle.

- ***Les robots à locomotion modulaire*** sont des robots capables de changer de forme ou de changer de mode de déplacement.

*Figure 1-12: Robot à locomotion modulaire [29]*

- ***Les robots sauterelles*** sont des robots capables d'emmagasiner de l'énergie pour effectuer un grand saut

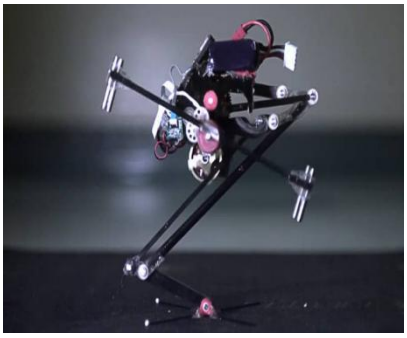


*Figure 1-13: Robot sauterelle [30]*

Dans les applications robotiques impliquant la manipulation et le transport de charges lourdes, la conception de la base mobile représente un élément central du système. Les contraintes liées à la masse embarquée, à la stabilité statique et dynamique, ainsi qu'à la résistance mécanique imposent le recours à des architectures de locomotion robustes et éprouvées. À cet effet, les bases mobiles terrestres à roues et à chenilles constituent les solutions les plus couramment retenues dans les contextes industriels et logistiques.

Les bases mobiles à chenilles se caractérisent par une excellente adhérence au sol et une répartition homogène des charges, ce qui leur confère une grande stabilité lors du déplacement et une aptitude accrue au franchissement d'obstacles. Ces propriétés en font

des plateformes particulièrement adaptées au support de bras manipulateurs de masse élevée ou à l'exécution de tâches en environnements contraints. En revanche, les bases mobiles à roues, notamment de type différentiel ou à traction synchrone, offrent une meilleure efficacité énergétique et des vitesses de déplacement plus élevées, à condition d'évoluer sur des surfaces relativement planes et structurées [31].

Les plateformes mobiles de petite taille ou à vocation expérimentale présentent quant à elles un intérêt principalement scientifique et technologique. Elles permettent de valider des modèles de locomotion, des stratégies de commande et des méthodes d'intégration entre la base mobile et le bras manipulateur, tout en réduisant les coûts et la complexité de mise en œuvre. Bien que leur capacité de charge soit limitée, ces systèmes constituent une étape essentielle dans le développement de solutions robotiques plus performantes et adaptées aux applications industrielles à différentes contraintes mécaniques [1].

### 1.3.4 Bras manipulateur

Un robot manipulateur est un bras articulé destiné principalement à la manipulation et équipé d'un certain nombre de degrés de liberté, ainsi que de possibilités élevées de programmation.
Cette propriété permet d'affecter le même robot à des tâches diverses. Un bras articulé a mécanisme ayant une structure plus ou moins proche de celle du bras humain.

1. **Structure et composant**

Un bras robotisé est composé d'une base et du corps du bras avec ses articulations, des actionneurs et d'un effecteur à son extrémité.
La base est le point qui supporte l'ensemble du bras et lui permet aussi une rotation dans le plan horizontal, les articulations du corps permettent son déplacement (épaule, coude, poignet).
Les actionneurs sont situés ente les articulations, ils fournissent la puissance nécessaire pour permettre les mouvements. L'effecteur terminale est conçus pour effectuer des tâches (pince, tourne vis, …) [32].

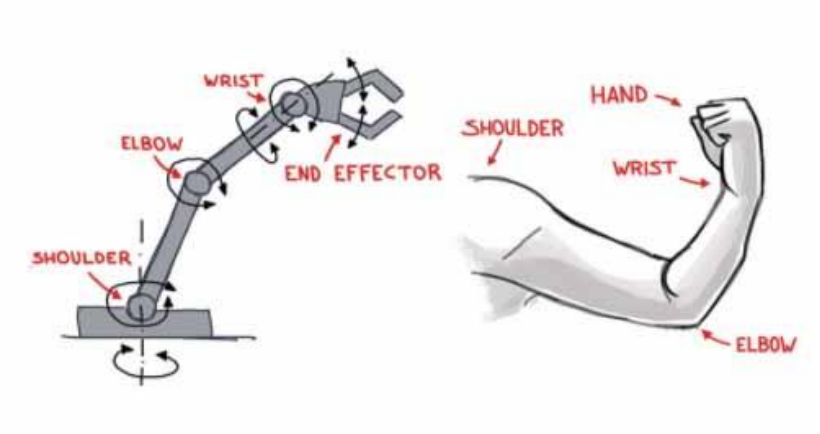



*Figure 1-14: Parties d'un bras robotique articulé [32]*

## 2. Classification des manipulateurs

La classification des manipulateurs repose sur leur configuration, leur degré de liberté et leur mode de déplacement.

- ***Robot cartésien(3ddl)*** : sa structure comporte trois articulations qui ne permettent que des mouvements de translation.

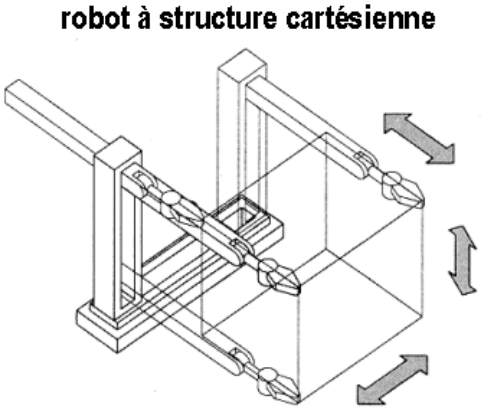



*Figure 1-15: Structure du robot cartésien [33]*

- ***Manipulateur cylindrique(3ddl)*** : il comprend une articulation rotative à sa base et deux autres articulations prismatiques.

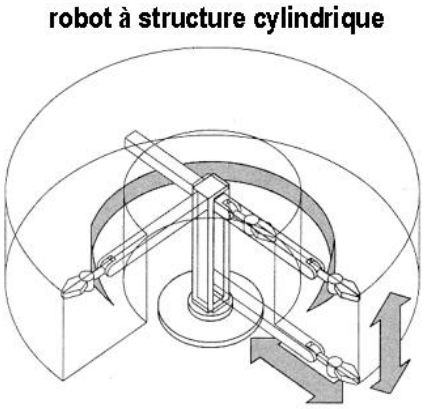



*Figure 1-16: Structure du Robot Cylindrique [33]*

- ***Manipulateur sphérique(3ddl)*** : il comprend deux articulations rotatives (une verticale et une horizontale) et une articulation linéaire. Il atteint des positions avec des coordonnées sphériques.

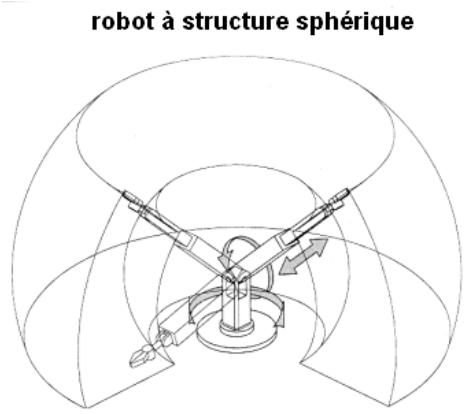



*Figure 1-17: Structure du robot Sphérique [33]*

- ***Manipulateur SCARA (Sélective Compliance Assembly Robot Arm) (4ddl)*** : sa structure comporte deux mouvements de rotation dans le plan horizontal et un mouvement linéaire vertical.

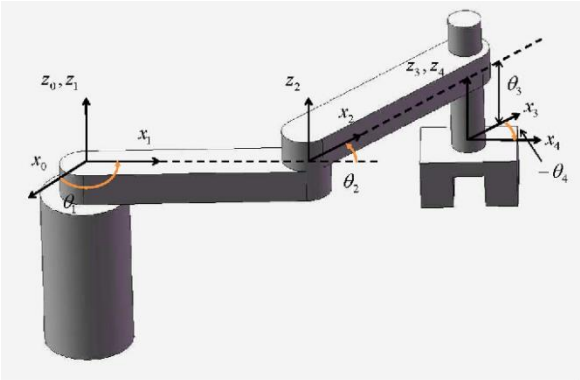



*Figure 1-18: Structure du manipulateur Scara [34]*

- ***Robot articulé (3 à 6 ddl)*** : appelé autrement bras robotique, il comprend entre 3 et 6 articulations rotatives, imitant les mouvements du bras humain.

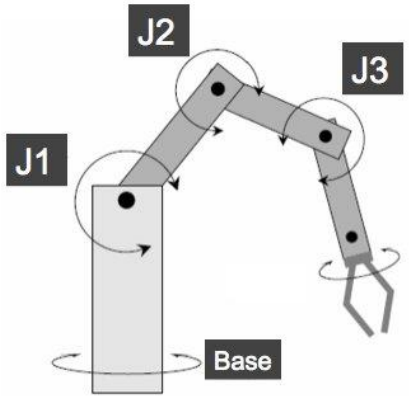



*Figure 1-19: Structure d'un bras articulé [35]*

## 1.4 Perception

La fonction perception consiste globalement à saisir un certain nombre d'informations sensorielles dans le but d'acquérir une connaissance et une compréhension du milieu d'évolution.

Pour un robot mobile, elle est le préalable indispensable aux étapes de localisation et de mise à jour de la carte de l'environnement [11].

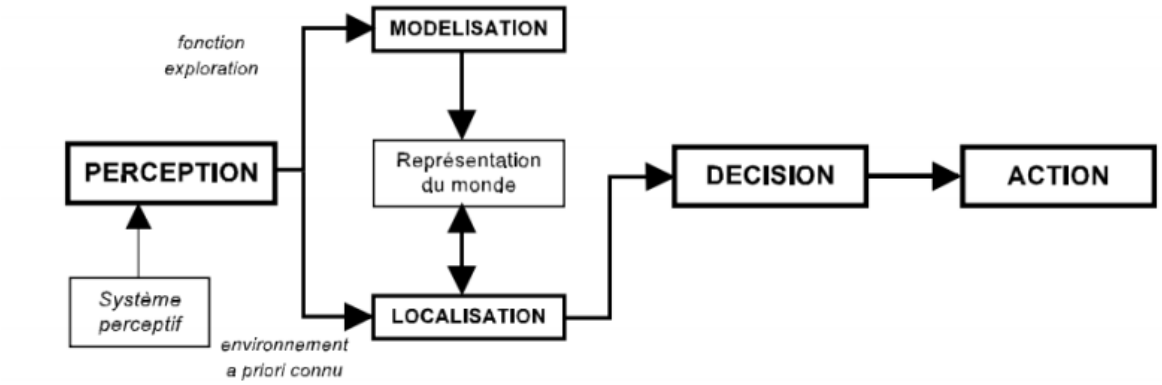



*Figure 1-20: Schéma de Perception d'un robot [36]*

Plusieurs constats peuvent être faits sur cet organe essentiel de la chaîne fonctionnelle de la navigation :

- *CONSTAT 1*

Le choix d'un système de perception est souvent dépendant du milieu d'évolution du robot mobile ainsi que des fonctionnalités mises en œuvre sur le robot pour qu'il puisse remplir sa mission.

- *CONSTAT 2*

Un système de perception constitué d'un unique capteur sera rarement suffisant pour percevoir correctement l'environnement. Le système de perception d'un robot mobile intégrera le plus souvent plusieurs capteurs qui seront de types complémentaires pour un enrichissement des informations sensorielles, ou de types redondants pour répondre au problème de fonctionnement en mode dégradé.

- *CONSTAT 3*

Coût de l'intégration de capteurs sur le véhicule autonome. La précision désirée et une fréquence d'acquisition élevée seront autant de facteurs qui augmenteront le coût d'un capteur. Il s'agit donc là d'une contrainte qui pèsera inévitablement sur le choix d'un système de perception.

## 1.5 Conclusion partielle

Ce chapitre a permis de présenter les bases théoriques de la robotique et de mettre en évidence l'importance croissante des manipulateurs mobiles dans les applications industrielles. Ces systèmes, alliant mobilité et capacité de manipulation, offrent des solutions efficaces pour l'automatisation des tâches complexes. Cette revue de littérature nous a permis de poser les bases théoriques de notre travail, orienté vers l'élaboration d'un manipulateur mobile destiné à la manutention.

# Chapitre 2 : METHODOLOGIE DE CONCEPTION DU ROBOT MANIPULATEUR

## 2.1 Introduction

La conception d'un robot manipulateur s'appuies sur une démarche méthodologique structurée visant à assurer la cohérence et la faisabilité du système. Ce chapitre présente la méthodologie adoptée pour la conception du robot manipulateur, depuis l'analyse des besoins jusqu'à la réalisation du prototype.

## 2.2 Objectif et spécification de conception

### 2.2.1 Objectif

Les objectifs de ce projet sont de concevoir un robot capable d'effectuer de tâches de manutention dans un environnement semi structuré (tels que les ateliers et entrepôts caractérisés par des conditions dynamiques). Le robot doit être capable de soulever et de déplacer des charges de manière semi autonome et de les déposer à un endroit bien définis. Il sera ainsi capable de réduire les efforts humains.

Outre la manutention, le robot sera conçu de manière à être polyvalent pour pourvoir accomplir certaines tâches simples tels que l'approvisionnement de poste de travail, l'assistance à l'opérateur, l'inspection visuelle simple et peut servir d'un robot de démonstration pédagogique.

De manière plus spécifique la conception vise à :

- Développer un bras robotisé offrant une bonne flexibilité de mouvement et une capacité d'orientation précise de l'organe terminal ;
- Assurer la mobilité du système grâce à une plateforme roulant stable et maniable ;
- Garantir une coordination efficace entre la base mobile et le bras manipulateur ;
- Concevoir un système simple, robuste et à faible cout, utilisant des composant facilement disponible ;

- Permettre une évolution future du prototype, notamment par l'ajout de capteurs ou des stratégies de commande plus avances.

### 2.2.2 Spécification fonctionnelle et technique

Afin de répondre aux objectifs définis, le robot manipulateur mobile doit satisfaire aux spécifications fonctionnelles et techniques suivantes :

- Se déplacer sur une surface plane ;
- Localiser l'objet à manipuler ;
- Saisir l'objet à l'aide d'une pince ;
- Transporter l'objet et le déposer avec une bonne précision ;
- Un bras robotisé de type sériel permettant l'orientation et le positionnement de la pince ;
- Une charge utile d'ordre de 30 à 50 kilogrammes ;
- Une portée du bras comprise entre 30 et 200 cm ;
- Une plateforme mobile à roues différentielles pour assurer une bonne maniabilité ;
- Une commande embarquée basée sur un microcontrôleur ;
- Une alimentation autonome par batterie rechargeable ;
- Des contraintes de stabilité et de sécurité lors du fonctionnement.

Ces spécifications constituent le cadre de référence pour les choix de conception.

## 2.3 Conception du système robotisé

La conception de ce système robotisé comprend trois grandes étapes cruciales pour répondre aux spécifications :

- ***Une conception mécanique*** : c'est robot avec un bras manipulateur à 5 dégrées de libertés et une plateforme mobile à roues,
- ***Une conception électronique*** : elle assure l'alimentation la commande des actionneurs et la gestion des capteurs des signaux,
- ***La commande*** : elle assure le pilotage et la coordination du mouvement, système robotisé. Cette partie garantie la précision, la stabilité et la cohérence du fonctionnement global du système.

### 2.3.1 Modélisation et Conception mécanique

#### 2.3.1.1 Modélisation du bras robotisé

La modélisation du bras robotique constitue une étape essentielle de la démarche de conception car elle permet de décrire mathématiquement la structure du bras manipulateur et l'analyse des capacités de mouvement avant la réalisation mécanique. Dans ce projet, le bras est modélisé comme une chaine cinématique sérielle compose de cinq articulations rotatives, correspondant à un manipulateur de type 5R.

La modélisation du bras permet de définir l'espace de travail (workspace) accessible et de vérifier que les mouvements requis pour les tâches de manutention sont réalisables. Elle permet aussi de faciliter la phase de simulation et de validation des choix de conception.

#### 2.3.1.2 Modélisation cinématique du bras robotisé

1. **Architecture et hypothèses**

Le bras robotisé étudié est un manipulateur sériel composé de **cinq degrés de liberté rotatifs (5R)** dédiés au positionnement et à l'orientation de l'organe terminal. La pince constitue un actionneur supplémentaire dédié à la préhension et n'est pas prise en compte dans la modélisation cinématique [31].

Les hypothèses suivantes sont retenues :

- Les liaisons sont parfaites (sans jeu) ;
- Les déformations mécaniques sont négligées ;
- Les axes des articulations sont rigides et bien définis ;
- La plateforme mobile est considérée immobile pendant la manipulation

2. **Paramétrage géométrique de Denavit –Hartenberg**

La méthode de Denavit–Hartenberg(D-H) est utilisée afin d'établir un modèle cinématique systématique du bras robotisé. Quatre règles guident la modélisation des repères D-H appelée convention de D-H [31]:

- L'axe z est l'axe de rotation d'une articulation rotative.
- L'axe x doit être perpendiculaire à la fois à l'axe z actuel et à l'axe z précédent.
- L'axe y est déterminé à partir des axes x et z en utilisant le système de coordonnées direct .
- L'axe x doit croiser l'axe z précédent.

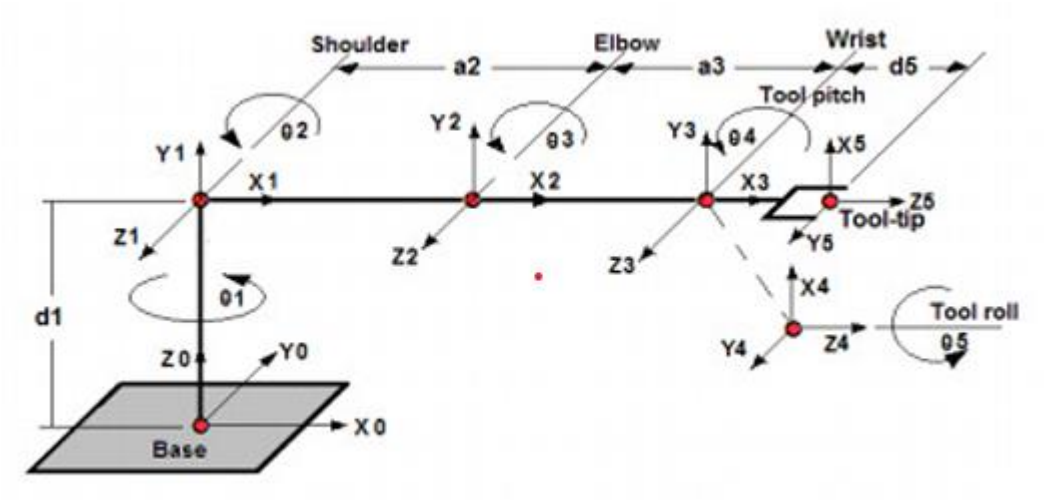



*Figure 2-1: Repères de D-H d'un bras 5R [37]*

Avec ces repères et leurs axes respectifs, quatre paramètres sont définis pour $i$ = 1 *à* $n$ : $\theta_i$, $\alpha_i$,$d_i$, $a_i$ étant respectivement *l'angle d'articulation*, la *torsion de lien*, le *décalage de lien* et la *longueur du lien*. Ils sont définis ci-dessous :

- $\alpha_i$ : l'angle de rotation entre les axes $Z_{i-1}$ et $Z_i$ correspondant à une rotation autour de l'axe $X_{i-1}$ ;
- $a_i$:la distance entre $Z_{i-1}$ et $Z_i$ mesurée le long de l'axe $X_{i-1}$ ;
- $\theta_i$ : l'angle de rotation entre les axes $X_{i-1}$ et $X_i$ correspondant à une rotation autour de l'axe $Z_i$ ;
- $d_i$ :la distance entre $X_{i-1}$ et $X_i$ mesurée le long de l'axe $Z_i$.

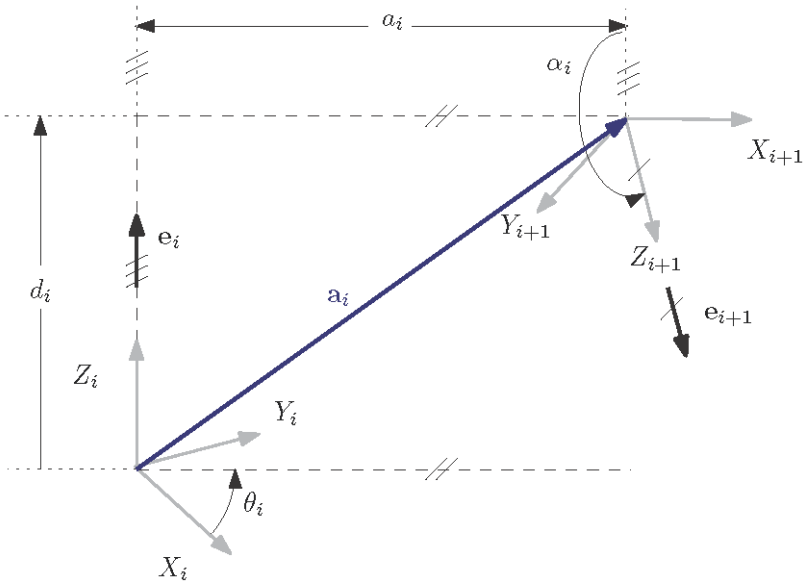



*Figure 2-2: Paramètre D-H [38]*

*Tableau 1: Tableau des paramètres D-H du bras robotisé*

| i | $a_i$ | $\alpha_i$ | $d_i$ | $\theta_i$ |
|---|---|---|---|---|
| 1 | 0 | $\pi/2$ | $d_1$ | $\theta_1$ |
| 2 | $a_2$ | 0 | 0 | $\theta_2$ |
| 3 | $a_3$ | 0 | 0 | $\theta_2$ |
| 4 | 0 | $\pi/2$ | 0 | $\theta_4$ |
| 5 | 0 | 0 | $d_5$ | $\theta_5$ |

## 3. Modèle cinématique direct (MCD)

La cinématique directe permet de déterminer la **position et l'orientation de l'organe terminal** à partir des variables articulaires. Le calcul du MCD repose sur la multiplication successive des matrices de transformation homogène. La matrice de transformation du repère $R_i$ dans $R_{i-1}$ est donnée par [39]:

$$T_{i-1}^{i} = \begin{pmatrix} \mathrm{Cos}\,\theta_i & -\sin\theta_i \cos\alpha_i & \sin\theta_i \sin\alpha_i & a_i \cos\theta_i \\ \sin\theta_i & \cos\theta_i \cos\alpha_i & -\cos\theta_i \sin\alpha_i & a_i \sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (1)$$

L'équation globale du robot **:** La pose finale (Position P et Orientation Q) est le produit de ces 5 matrices :

$$T_1^5 = \prod_{i=1}^{5} T_{i-1}^{i} = \begin{pmatrix} Q & P \\ 0^T & 1 \end{pmatrix} \quad (2)$$

- La matrice de transformation du repère zéro au premier repère est donnée par :

$$T_0^1 = \begin{pmatrix} \text{Cos}\,\theta_1 & 0 & \sin\theta_1 & 0 \\ \sin\theta_i & 0 & -\cos\theta_1 & 0 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (3)$$

- La matrice de transformation du premier au deuxième repère est donnée par :

$$T_1^2 = \begin{pmatrix} \text{Cos}\,\theta_2 & -\sin\theta_2 & 0 & a_2\cos\theta_2 \\ \sin\theta_2 & \cos\theta_2 & 0 & a_2\sin\theta_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (4)$$

- La matrice de transformation du deuxième au troisième repère est donnée par :

$$T_2^3 = \begin{pmatrix} \text{Cos}\,\theta_3 & -\sin\theta_3 & 0 & a_3\cos\theta_3 \\ \sin\theta_3 & \cos\theta_3 & 0 & a_3\sin\theta_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (5)$$

- La matrice de transformation du troisième au quatrième repère est donnée par :

$$T_3^4 = \begin{pmatrix} \text{Cos}\,\theta_4 & 0 & \sin\theta_4 & 0 \\ \sin\theta_4 & 0 & -\cos\theta_4 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (6)$$

- La matrice de transformation du quatrième au cinquième repère est donnée par :

$$T_4^5 = \begin{pmatrix} \text{Cos}\,\theta_5 & -\sin\theta_5 & 0 & 0 \\ \sin\theta_5 & \cos\theta_5 & 0 & 0 \\ 0 & 0 & 1 & d_5 \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (7)$$

Après développement des calculs, cette multiplication matricielle donne les équations qui représentent les positions de l'effecteur terminal (par rapport au repère zéro) qui sont les suivantes :

$$P_x = \cos\theta_1[a_2\cos\theta_2 + a_3\cos(\theta_2+\theta_3) + d_5\sin(\theta_2+\theta_3+\theta_4)] \quad (8)$$

$$P_y = \sin\theta_1[a_2\cos\theta_2 + a_3\cos(\theta_2+\theta_3) + d_5\sin(\theta_2+\theta_3+\theta_4)] \quad (9)$$

$$P_z = d_{1+}\, a_2\sin\theta_2 + a_3\sin(\theta_2+\theta_3) - d_5\cos(\theta_2+\theta_3+\theta_4) \quad (10)$$

## 4. Modélisation cinématique inverse (MCI)

La cinématique inverse consiste à déterminer les angles articulaires nécessaires pour atteindre une position donnée de l'organe terminal. Le MCI permet de déterminer les coordonnées articulaires $Q = [\theta_1, \theta_2, \theta_3, \theta_4, \theta_5]^T$ pour une pose de l'effecteur ($P_x$, $P_y$, $P_z$) et une orientation donnée. Une approche géométrique a été utilisée pour déterminer les paramètres des coordonnées articulaires [37].

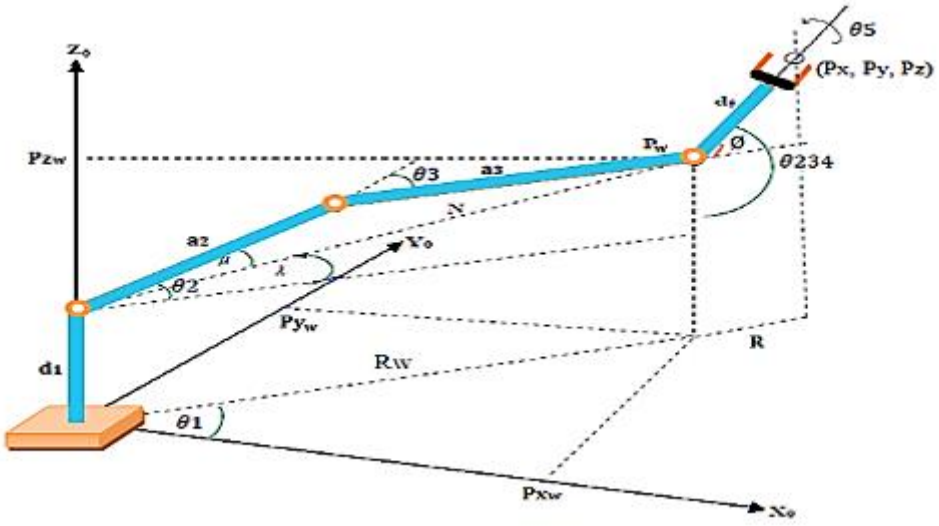



*Figure 2-3: Paramètre Géométrique du bras [37]*

L'orientation du poignet est caractérisée par l'angle global $\theta_{234}$, qui représente la somme des rotations des trois dernières articulations :

$$\theta_{234} = \theta_2 + \theta_3 + \theta_4 \quad (11)$$

Dans le cadre de cette étude, seule l'orientation selon l'axe de tangage (pitch) est considérée. La relation entre l'angle de pitch du poignet θ et l'angle global $\theta_{234}$ est donnée par :

$$\theta_{234} = 90^{\circ} \mp \theta \tag{12}$$

Cette hypothèse permet de simplifier le problème cinématique tout en restant adaptée aux applications de manutention.

La position du dernier segment est donnée par :

$$R = d_5 \cos\theta \tag{13}$$

Et la position du centre du poignet est donnée par :

$$\begin{cases} P_{xw} = P_x - \mathrm{R}\cos\theta_1 \\ P_{yw} = P_y - \mathrm{R}\sin\theta_1 \\ P_{zw} = P_z + d_5 \sin\theta \end{cases} \tag{14}$$

La distance horizontale du poignet par rapport à l'axe de la base est définie par :

$$R_w = \sqrt{P_{wx}^2 + P_{wy}^2} \tag{15}$$

La distance totale entre la deuxième articulation et le centre du poignet est alors :

$$N = \sqrt{(P_{zw} - d_1)^2 + R_w^2} \tag{16}$$

L'angle de base est obtenu par :

$$\theta_1 = \tan^{-1}\left(\frac{P_y}{P_x}\right) \tag{17}$$

On obtient l'angle de l'épaule en appliquant la loi des cosinus au triangle formé par $a_2$, $a_3$ et N :

$$\mu = \cos^{-1}\left(\frac{N^2 + a_2^3 - a_3^2}{2\, a_2\, N}\right) \tag{18}$$

$$\lambda = \tan^{-1}\left(\frac{P_{zw} - d_1}{R_w}\right) \tag{19}$$

$$\theta_2 = \lambda \mp \mu \tag{20}$$

L'angle du coude est obtenu par :

$$\theta_3 = \pm \cos^{-1}\left(\frac{N^2 - a_2^3 - a_3^2}{2\, a_2\, a_3}\right) \tag{21}$$

Les signes $\pm$ correspondent aux configurations *coude en haut* et *coude en bas*.

L'angle du poignet est donc donné par :

$$\theta_4 = \theta_{234} - \theta_2 - \theta_3 \tag{22}$$

L'angle $\theta_5$ correspond à la rotation de l'effecteur final autour de son axe longitudinal. Cet angle n'influence ni la position ni l'orientation en pitch de l'effecteur et peut donc être fixé indépendamment en fonction de la tâche à réaliser.

### 2.3.1.3 Modélisation dynamique du bras robotisé

#### 1. Contexte théorique

La dynamique complète d'un manipulateur robotique est régie par l'équation de Lagrange, qui relie les couples moteurs aux positions, vitesses et accélérations articulaires [31]:

$$\tau = M(q)\ddot{q} + C(q,\dot{q})\dot{q} + G(q) + F_v(\dot{q}) + F_s \quad (23)$$

- $M(q)$ Matrice d'inertie du robot
- $C(q,\dot{q})$ Représente les forces de Coriolis et centrifuges
- $G(q)$ Représente les couples dus à la gravité
- $F_v(\dot{q})$ et $F_s$ les frottements

#### 2. Hypothèses de modélisation

Pour l'étude dynamique des moteurs du bras, les hypothèses suivantes sont retenues :

- Mouvement plat
- Segments rigides
- Masse concentrée au centre de chaque lien
- Gravité dominante
- Étude quasi-statique

Ces hypothèses permettent de conserver un modèle simple tout en représentant les efforts principaux

#### 3. Couples gravitaires

- Couple gravitaire au niveau du moteur du lien 4, moteur de tangage (pitch) du poignet supportant le poids de la pince est obtenu par :

$$\tau_4 = m_4 g \left({}^{d_5}/_{2}\right) cos(\theta_2 + \theta_3 + \theta_4) \quad (24)$$

- Couple gravitaire au niveau du moteur du lien 3, moteur du coude supportant le poids de l'avant-bras et de la pince est obtenu par :

$$\tau^3 = m_3\, g\, ({}^{a_3}/_{2})cos(\theta_2 + \theta_3) + m_4 g \left[ a_3\ cos(\theta_2 + \theta_3) + \left({}^{d_5}/_{2}\right) cos(\theta_2 + \theta_3 + \theta_4)\right] \quad (25)$$

- Couple gravitaire au niveau du moteur du lien 2, moteur de l'épaule supportant le poids du bras, de l'avant-bras et de la pince est obtenu par :

$$\tau_2 = m_2\, g\, ({}^{a_2}/_{2})\cos(\theta_2) + m_3\, g\, [a_2 \cos(\theta_2) + ({}^{a_3}/_{2})\cos(\theta_2 + \theta_3)] + m_4 g\, [a_2\cos(\theta_2) + a_3\cos(\theta_2 + \theta_3) + ({}^{d_5}/_{2})\cos(\theta_2 + \theta_3 + \theta_4)]\ ] \quad (26)$$

### 2.3.2 Conception mécanique du bras manipulateur

Le bras robotisé est l'élément central du système de manipulation. Sa conception doit permettre d'assurer une bonne flexibilité de mouvement, une précision, une compatibilité avec la plateforme mobile et une bonne stabilité du système.

La conception de notre manipulateur repose sur une architecture sériel anthropomorphe a cinq degrés de liberté(5ddl). L'objectif principal est de concilier une grande amplitude de mouvement avec une structure légère et rigide.

#### 2.3.2.1 Les actionneurs

Dans un robot manipulateur mobile, les actionneurs sont les éléments essentiels qui convertissent l'énergie électrique en mouvement mécanique. Ils permettent à la fois de déplacer la plateforme et de contrôler les articulations du bras robotique, offrant ainsi la capacité d'interagir avec l'environnement. Le choix des actionneurs influence directement la précision, la vitesse, la charge utile et la stabilité du système [1].

1. **Actionneur du bras robotise**

Le bras robotique est constitué de plusieurs articulations chaque articulation nécessitant un actionneur pour contrôler son angle de rotation. Les actionneurs les plus couramment utilisés sont :

a. **Servomoteurs** : un servomoteur est un moteur électrique équipé d'un système de rétroaction (généralement un potentiomètre ou un encodeur), permettant un positionnement précis. Avec comme avantage le contrôle précis de la position et de l'angle et comme inconvénient la sensibilité à la surchauffe pour un usage intensif. Les servomoteurs sont adaptés aux charges légères, moyennes et lourde.
b. **Moteurs pas à pas (Stepper motors) :** un moteur pas à pas divise un tour complet en un nombre précis de pas, permettant un contrôle angulaire très fin. Ils offrent une très grande précision de positionnement et un bon couple a base vitesse. Les moteurs pas à pas sont adaptés aux charges moyennes et lourdes.
c. **Moteurs à courant continu avec encodeur :** un moteur DC classique couplé à un encodeur rotatif pour la rétroaction de position ou vitesse. Couplés à un réducteur, ils peuvent générer un couple élevé mais leur système de contrôle reste complexe. Ils sont utilisés sur des articulation à forte charge.

2. **Actionneurs de la plateforme mobile**

La plateforme mobile est responsable de la locomotion du robot.

a. **Moteurs à courant continu :** chaque roue est entraînée par un moteur DC, souvent couplé à un réducteur pour augmenter le couple. Ils ont l'avantage d'avoir une simplicité de commande (variation de tension ou PWM) et bonne capacité de charge mais leur précision de position est limitée sans encodeurs.
b. **Moteurs pas à pas pour :** chaque roue est contrôlée par un moteur pas à pas. Ils offrent un passionnément précis pour déplacements exacts mais sont moins adapté aux grandes vitesses.

Ainsi, le choix des actionneurs d'un robot manipulateur mobile résulte d'un compromis entre précision, couple, vitesse et complexité de commande. Les servomoteurs conviennent principalement aux articulations nécessitant un positionnement précis sous charges légères à moyennes, tandis que les moteurs pas à pas et les moteurs à courant continu avec encodeur

sont privilégiés pour les applications demandant un couple plus élevé ou une précision accrue. Pour la plateforme mobile, les moteurs à courant continu constituent une solution simple et efficace pour la locomotion, assurant la mobilité du robot.

### 2.3.2.2 Structure mécanique du bras manipulateur

La conception du bras a 5ddl repose sur le paradigme de la ***RIGIDITE SPECIFIQUE*** : maximiser la résistance aux déformations tout en minimisant la masse embarquée [40]. Cette approche est d'autant plus critique que sur le bras est monté sur une plateforme mobile, soumis donc à des perturbations dynamique (accélération de la base, vibration du sol).

**a.** ***Architecture des segments, optimisation de l'inertie de la section*** : pour garantir une précision millimétrique à la pince, chaque segment doit présenter une déformation minimale sous l'effet du moment de flexion. La conception privilégie des sections à géométrie repartie.

**b.** ***Conception du poignet et de la pince (organe de liaison)*** : l'extrémité du bras (poignée et pince) est la zone où les contraintes du poids sont les plus sévère (loi de masse ajoutée en bout de chaine).

- ***L'intégration compacte :*** les deux dernières articulations sont conçues avec des axes concourants pour minimiser les bras de levier internes, réduisant les couples de torsion parasite sur le segment principal ;
- ***La pince de préhension*** : l'organe de préhension retenu est une pince mécanique simple adaptée à la préhension d'objets et des masses varier. Ce choix s'explique par la s'implicite de fabrication et commande.

**c.** ***Choix Matériaux*** : Le choix des matériaux pour le bras manipulateur est effectué en fonction des charges mécaniques appliquées à chaque composant. Les éléments structurels sont principalement en aluminium en raison de son bon compromis entre légèreté, résistance mécanique et facilité d'usinage. Les pièces fortement sollicitées, telles que les axes d'articulation sont réalisés en acier afin de garantir une résistance élevée aux contraintes mécaniques et à la fatigue. En revanche, les composants faiblement sollicités sont fabriqués en plastiques techniques pour alléger l'ensemble.

d. ***Choix des actionneurs*** : les articulations principales du bras (épaule et coude) sont actionnées par des moteurs pas à pas hybrides à boucles fermées type NEMA 34 et NEMA 42, couplés à des réducteurs planétaires de précision (ratio 1 :50 à 1 :100) cette combinaison permet de générer un couple supérieur a 300Nm, indispensable pour compenser l'effet de levier exerces par une charge de 50Kg, Le moteur pas à pas NEMA 34 avec réducteur à couronne dentée. Ce choix est dicté par la nécessité de supporter le moment d'inertie massif de l'ensemble de mouvement. Les articulations du poignet dont la fonction est l'orientation de l'organes terminale, sont commandées par des moteurs pas à pas NEMA 23. Ils offrent la force nécessaire pour orienter l'Object saisie sans dérive, tout en conservant une compacité adaptée a l'extrémité du bras.

e. ***Adaptabilité à la plateforme mobile*** : la conception prend en compte l'aspect embarquée par deux caractéristiques majeures :
    - Capacité de repliement : la longueur des segments a été calculée pour permettre au bras de se replier totalement dans le périmètre de la plateforme, abaissant le centre de gravité global lors de phrase de translation rigide

Les choix de conception ont été orientés vers une structure robuste en alliage d'acier et aluminium, capable d'encaisser les moment d'inertie généré par le levage de 50kg. Par ailleurs la stabilité est renforcée par un abaissement stratégique du centre de gravité c'est-à-dire positionnement des batteries en base, et l'utilisation des roues a haute capacité de charge, garantissant une navigation fluide et sécurité en environnement industriel.

#### 2.3.2.3 Conception de la plateforme mobile

La plateforme mobile constitue le support du bras robotisé et assure le déplacement global du système dans l'environnement de travail. Sa conception doit garantir à la fois la stabilité du robot lors des phases de manipulation et une mobilité suffisante pour permettre l'accès aux différentes zones de travail. Dans ce projet, la conception de la plateforme mobile a été guidée par des contraintes de simplicité, de stabilité et de compatibilité avec un bras manipulateur embarqué.

### 1. Architecture de locomotion

La plateforme mobile retenue repose sur une architecture à roues de type différentiel composée de quatre roues motrices à haute capacité de charge. Pour déplacer un ensemble dont la masse totale atteint 150 à 200kg. Le système utilise des motoréducteurs a courant continue BDLC intégré dans chaque axe.

- **Transmission :** l'utilisation de réducteurs à engrenages métalliques à engrenages métallique (ratio 1 :30) permet de multiplier le couple de traction pour des démarrages fluides, même à pleine charge.
- **Pneumatique :** les roues mécanums (omnidirectionnelles) sont sélectionnées pour leur coefficient de friction élevé et leur dureté, évitant l'écrasement du pneu sous les charges utiles et garantissant une précision œdométrique.
- **Stabilité :** l'empattement a été élargi pour assurer un triangle de sustentation optimal, rendant le basculement physiquement impossible dans les limites de fonctionnement définis.

Le châssis est conçu à partir d'une structure mécano-soudée en acier industriel haute résistance. Ce choix est impératif pour garantir la rigidité structurelle nécessaire a la manipulation de charges allant jusqu'à 50kg. L'utilisation de plaque d'aluminium de forte épaisseur aux points de montage permet d'éliminer toute flexion du châssis, assurant ainsi une précision millimétrique du bras articule. La plateforme intègre de compartiment spécifique pour le lestage et le positionnement bas des batteries, créant un contrepoids naturel qui stabilise l'ensemble lors des extensions maximales du manipulateur.

### 2. Positionnement du bras sur la plateforme

Le bras robotisé est monté au centre géométrique de la plateforme mobile avec un léger recul par rapport à l'axe central. Ce positionnement permet de maintenir le centre de gravité du système à l'intérieur de la surface d'appui de la plateforme, réduisant ainsi les risques de basculement lors des mouvements du bras. Cette configuration (illustrée sur la Figure 2-4) facilite également la coordination entre les déplacements de la plateforme et les mouvements du bras.

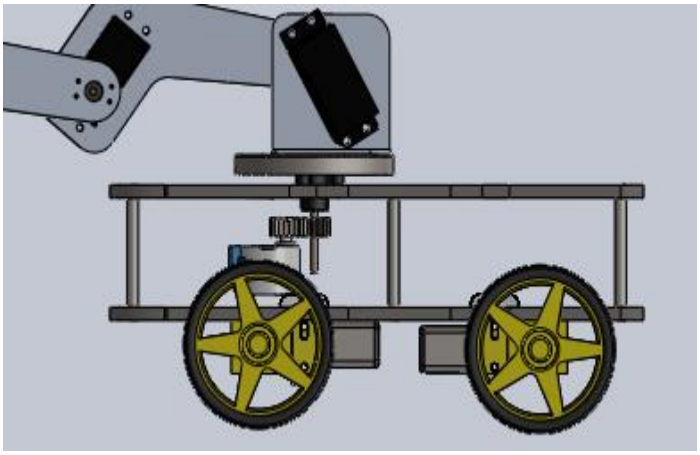


*Figure 2-4: positionnement du bras sur la plateforme*

### 2.3.3 Architecture électronique du système

L'architecture électronique du robot manipulateur mobile assure l'alimentation, la commande des actionneurs ainsi que l'acquisition des informations issues des capteurs. Elle constitue le lien entre la conception mécanique du système et la logique de commande mise en œuvre. Dans ce projet, l'architecture électronique a été conçue de manière modulaire afin de faciliter l'intégration des différents sous-systèmes et d'améliorer la maintenance du prototype.

1. **Unité de commande**

- **Commande centrale**

Le système repose sur une architecture de commande hybride. Le pilotage de haut niveau est assuré par un microcalculateur industriel **Raspberry-Pi 4**, choisi pour sa puissance de calcul, ses interfaces de communication sans fil intégrées et sa capacité à gérer les algorithmes des navigations autonomes (SLAM). Il fait office de cerveau central, traitant les commandes distantes et les données des capteurs.

Ce dernier communique via un bus industriel avec des contrôleurs des mouvements des actionneurs (drives) qui agissent comme contrôleur de bas niveau dédié à la gestion temps réel des actionneurs pour. Cette répartition permet d'isoler la gestion de la sécurité logiciel du contrôle direct des actionneurs.

- **Commande des actionneurs**

La gestion des moteurs pas à pas hybride est confié a des drivers industriels dédie communiquant via un bus de données haute vitesse. Ces contrôleurs intègrent leurs propres processeurs de signal (DSP) pour gérer l'avertissement en courant continue et la correction de la position en temps réel. Cette solution permet de réduire la charge de calcul du microcontrôleur et d'assurer une meilleure synchronisation des mouvements du bras, garantissant des mouvements des fluides et éliminant tout risque de résonance mécanique.
Les moteurs de la plateforme sont pilotés par des variateurs de puissance a pont en H renforcées, capables d'encaisser les courant de crête lors de démarrage en charge.

**2. Capteurs embarqués**

Le robot est équipé de plusieurs capteurs destinés à améliorer la perception de son état et de son environnement. Un capteur IMU permet de mesurer l'accélération et la vitesse angulaire du système, contribuant à l'estimation de l'orientation et à la stabilité de la plateforme. Un capteur LIDAR assure une couverture à 360 degrés pour l'évitement d'obstacles et la navigation SLAM.

**3. Alimentation et gestion de l'énergie**

L'alimentation du système est assurée par un pack de batteries LIFEPO4, sélectionné pour sa grande densité énergétique et sa capacité à fournir des courants des décharges élevées. Une batterie embarquée, fournissant l'énergie nécessaire aux moteurs pas à pas et à l'électronique de commande la tension minimale du système de 48v a été choisie pour minimiser les pertes par effet joule et garantir un couple constant aux actionneurs même lorsque les mouvements sont rapides. Une attention particulière a été portée à la séparation des alimentations de puissance et de commande afin de limiter les perturbations électriques et d'améliorer la fiabilité du système.

### 2.3.4 Logique de commande et coordination du système

La logique de commande du robot manipulateur mobile vise à assurer le contrôle coordonné du bras robotisé et de la plateforme mobile, tout en garantissant un fonctionnement simple, fiable et adapté. Dans ce projet, une approche hiérarchique de commande a été retenue,

distinguant la commande du bras robotisé, la commande de la plateforme mobile et la coordination entre ces deux sous-systèmes.

### 1. Commande du bras robotisé

La commande du bras robotisé repose sur le pilotage individuel des moteurs pas à pas associés aux différentes articulations. Chaque articulation est commandée en position à l'aide de signaux Drives générés par des modules HBS1108 (Nema 42), HBS86H (Nema 34) et IHSS57 (Nema 32 et 24) définies en fonction des mouvements souhaités du bras, tels que la prise, le déplacement et le dépôt d'un objet.

Ces moteurs sont commandés en position par comptage du nombre de pas effectués. Chaque pas correspond à un angle élémentaire connu, ce qui permet de déterminer la position angulaire de chaque articulation.

$$\theta_1 = N * \theta_{pas}$$

Avec $N\ et\ \theta_{pas}$ qui correspondent aux nombres des pas et à l'angle d'un pas.

Cette approche permet de simplifier la commande tout en garantissant une précision suffisante pour les tâches de manipulation envisagées.

Commande de la plateforme mobile

La plateforme mobile est piloté par des variateurs de puissance a pont en H renforcées. La commande repose sur des signaux PWM appliqués aux moteurs afin de réaliser les mouvements de base : translation avant et arrière, direction gauche/droite ainsi que rotation sur place.

La stratégie de commande adoptée privilégie des déplacements simples et maîtrisés, la vitesse de la plateforme étant volontairement limitée afin d'assurer la stabilité du système lors des phases de manipulation du bras robotisé.

### 2. Coordination bras–plateforme

La coordination entre le bras robotisé et la plateforme mobile constitue un aspect essentiel du système. Dans ce projet, une coordination séquentielle a été adoptée : la plateforme mobile se déplace pour positionner le robot à proximité de la zone de travail, puis le bras robotisé effectue la tâche de manipulation lorsque la plateforme est à l'arrêt.

Cette approche permet de réduire les interactions dynamiques entre le bras et la plateforme, simplifiant ainsi la commande globale du système et améliorant la stabilité lors des opérations de préhension.

### 3. Gestion des informations issues des capteurs

Les données fournies par les capteurs embarqués (IMU, capteur des distances, …) sont utilisées pour améliorer la sécurité et la fiabilité du fonctionnement du robot. Par exemple, le capteur ultrasonique permet d'éviter les obstacles lors des déplacements. Ces informations contribuent à une prise de décision simple mais efficace du système de commande.

## 2.4 Modélisation conceptuelle

Cette étape vise à vérifier la cohérence globale des choix effectués avant d'entamer la réalisation physique du prototype. Elle permet de s'assurer que l'architecture mécanique, électronique et fonctionnelle retenue répond aux objectifs et aux spécifications définis précédemment.

La modélisation mécanique du système a été réalisée à l'aide d'outils de conception assistée par ordinateur, notamment **SolidWorks**, permettant de représenter l'ensemble des sous-systèmes du robot, incluant le bras robotisé à cinq degrés de liberté, la plateforme mobile et les interfaces mécaniques associées. Cette modélisation tridimensionnelle a permis de visualiser les mouvements des articulations, de vérifier l'encombrement global du robot et d'identifier d'éventuelles interférences entre les composants.

Les résultats de cette phase de modélisation conceptuelle ont permis d'ajuster certains paramètres de conception avant la phase de réalisation.

Pour illustrer les différentes étapes de cette conception sous SolidWorks, la Figure 2-5 présente la modélisation des éléments constitutifs du châssis inférieur une fois assemblés.

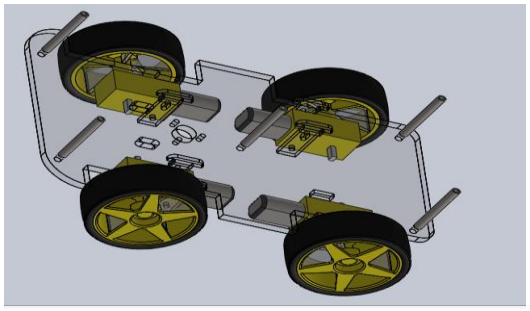


*Figure 2-5: Modélisation des éléments du châssis inferieur du robot assemblés*

L'intégration se poursuit avec le montage de la base du bras robotisé sur cette même plateforme mobile, comme le montre la Figure 2-6 ci-dessous.

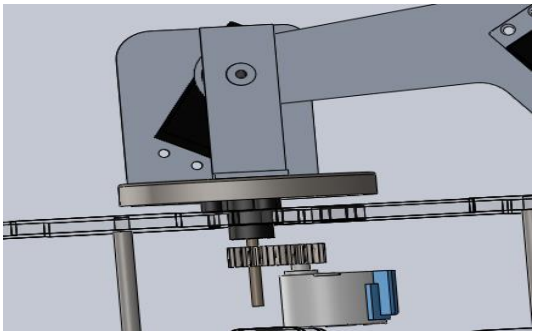


*Figure 2-6:Base du bras assemblé sur le châssis supérieur de la plateforme mobile*

La Figure 2-7 illustre ensuite l'assemblage complet de la structure principale du bras robotisé fixé sur le châssis.

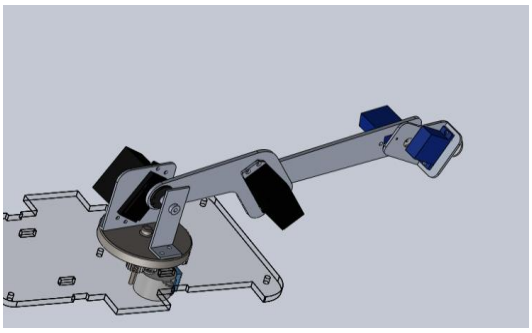


*Figure 2-7: Bras robotisé sur le châssis*

La conception détaillée de l'extrémité du robot, destinée à la manipulation, est mise en évidence par la Figure 2-8 qui présente l'assemblage du poignet et de la pince.

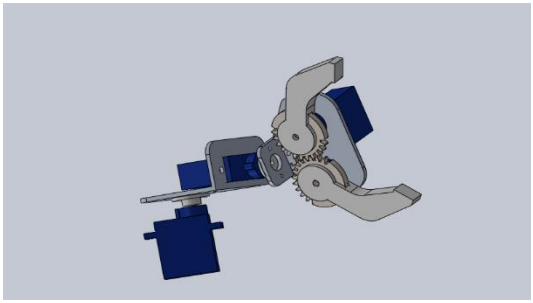


*Figure 2-8: Poignet et pince du bras*

Enfin, l'aboutissement de ce travail de modélisation est visible sur la Figure 2-9, qui dévoile la conception finale du robot manipulateur mobile entièrement assemblé.

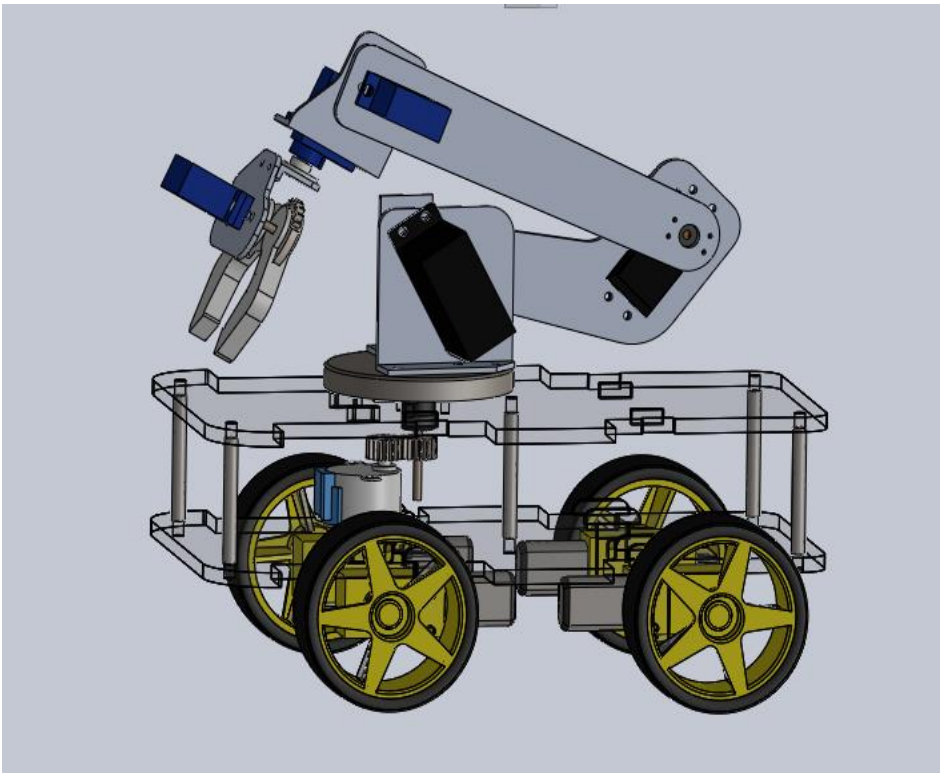


*Figure 2-9: Robot manipulateur mobile assemblé*

## 2.5 Conclusion partielle

Ce chapitre a présenté la méthodologie de conception adoptée pour le développement du robot manipulateur mobile. Il a permis de définir les objectifs et les spécifications du

système, puis de détailler les choix de conception relatifs au bras robotisé, à la plateforme mobile ainsi qu'à l'architecture électronique et à la logique de commande. L'ensemble de ces choix a été guidé par des contraintes de simplicité, de stabilité, de coût et de faisabilité, adaptées à la réalisation d'un prototype.

La modélisation conceptuelle a permis de vérifier la cohérence globale du système avant la phase de réalisation. Cette approche méthodologique assure une continuité logique entre l'analyse des besoins, la conception du robot et sa mise en œuvre pratique. Les simulations ainsi que la réalisation et la validation expérimentale du prototype sont présentées dans le chapitre suivant.

# Chapitre 3 REALISATION ET VALIDATION DU PROTOTYPE

## 3.1 Introduction

Après avoir établi les bases théoriques et la conception du robot dans les chapitres précédents, ce troisième chapitre expose la phase de mise en œuvre pratique. L'objectif est de transformer le modèle conceptuel en prototype physique fonctionnel capable de se déplacer et de manipuler des objets. Nous détaillons les simulations, la fabrication mécanique, l'intégration des composants électronique, l'architecture de commande logicielle, et enfin, la phase de validation technique qui permet de mesurer les performances réelles du systèmes face aux objectifs fixés.

## 3.2 Validation du robot par simulations statique, dynamique et cinématique

Dans le cadre de la conception du robot manipulateur mobile, des simulations numériques ont été réalisées afin de vérifier les performances mécaniques et fonctionnelles du système avant sa mise en œuvre physique. Ces simulations permettent de réduire les risques d'erreurs de conception, d'optimiser les dimensions des composants et d'anticiper le comportement réel du robot en fonctionnement.

Trois types d'analyses complémentaires ont été effectués :

- Une simulation statique pour évaluer la résistance mécanique de la structure avec le logiciel **SOLIDWORKS** [41] ;
- Une simulation dynamique pour analyser le comportement du système en mouvement avec le logiciel **SOLIDWORKS** ;
- Une simulation de cinématique inverse pour valider la précision du positionnement de l'effecteur avec le logiciel **MATLAB** [42].

Les deux premières simulations ont été effectuées avec le bras en position critique, c'est-à-dire totalement allongé. Cette position, qui engendre le maximum de contraintes sur la structure, est illustrée par la Figure 3-1.

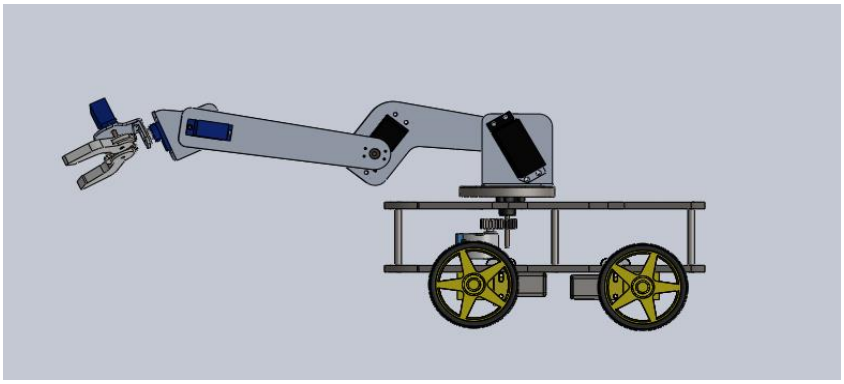


*Figure 3-1 : Robot avec le bras en position critique*

### 3.2.1 Simulation statique de la structure

La simulation statique a consisté à analyser le comportement mécanique du robot soumis à des charges représentant le poids propre des éléments ainsi que la charge maximale manipulée qui est de 200g. Cette étude vise principalement à vérifier la résistance des matériaux, la rigidité de la structure et l'absence de déformations excessives pouvant compromettre la précision du robot.

Les résultats de la simulation montrent que les contraintes de Von Mises (stress) restent localisées au niveau des pivots avec un stress maximal de 48.66 MPA (Méga pascale). La Figure 3-2 permet de localiser visuellement ces zones de concentration de contraintes au niveau des articulations de la base, de l'épaule et du coude.

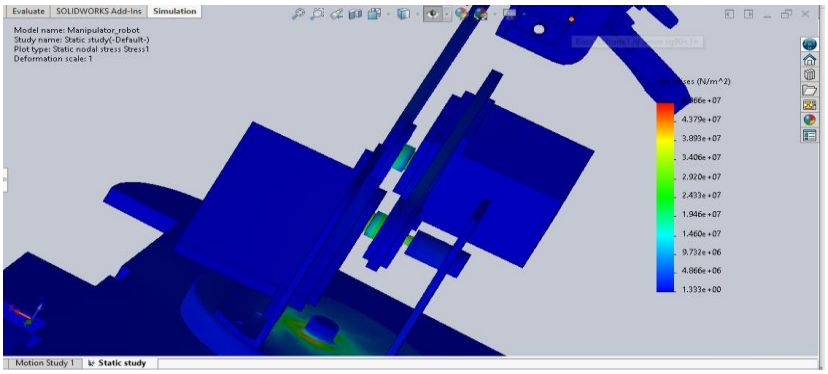



*Figure 3-2: Stress au niveau des articulations base-épaule-coude du bras robotisé*

De la même manière, la Figure 3-3 détaille l'impact de ces contraintes mécaniques directement sur les articulations du poignet.

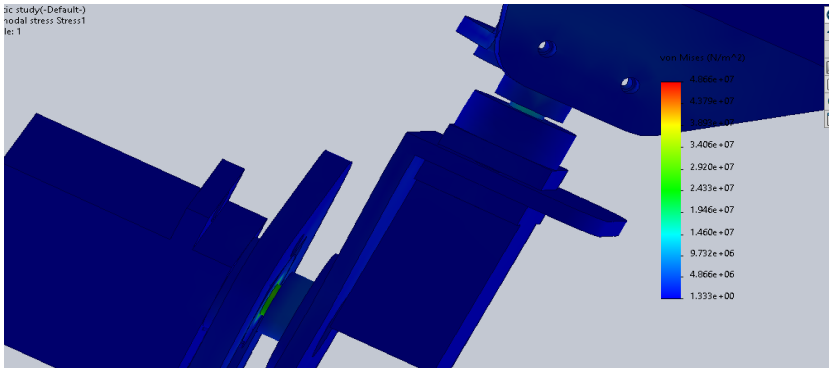



*Figure 3-3 : Stress au niveau des articulations du poignet*

Pour interpréter ces zones (concentrations des contraintes), la Figure 3-4 fournit l'échelle de couleurs correspondant aux valeurs des contraintes de Von Mises (en notation scientifique).

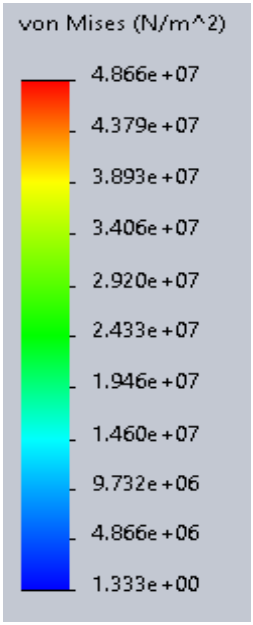



*Figure 3-4 : Echelles de couleur des valeurs contraintes de Von Mises du robot (en notation scientifique)*

Le déplacement maximal observé en bout de bras est d'à peu près 3mm, ce qui n'altère pas la précision du positionnement global. L'ampleur de ce déplacement sur l'ensemble de la structure est présentée sur la Figure 3-5.

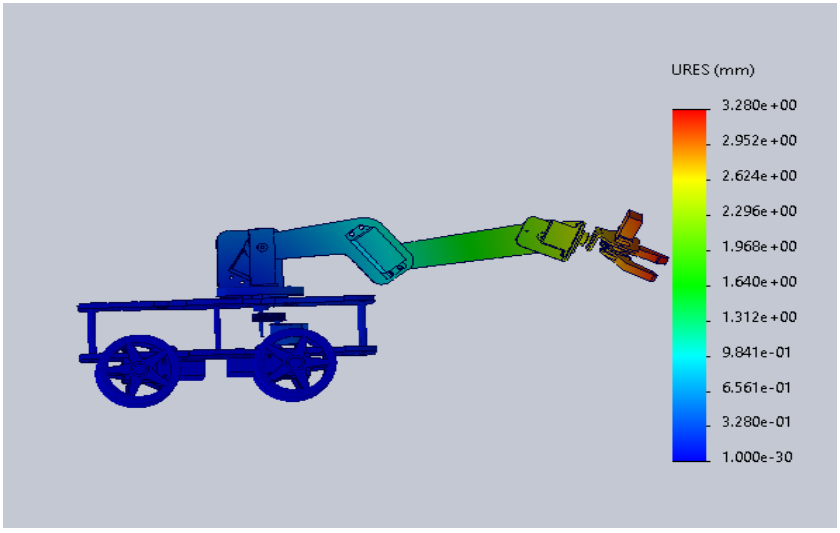



*Figure 3-5 : Déplacement globale de la structure avec une échelle de couleur*

Le critère de validation final repose sur le coefficient de sécurité. Comme le démontre la Figure 3-6, avec une valeur minimale de 3.9, la structure est considérée comme robuste pour les applications de manipulation envisagées.

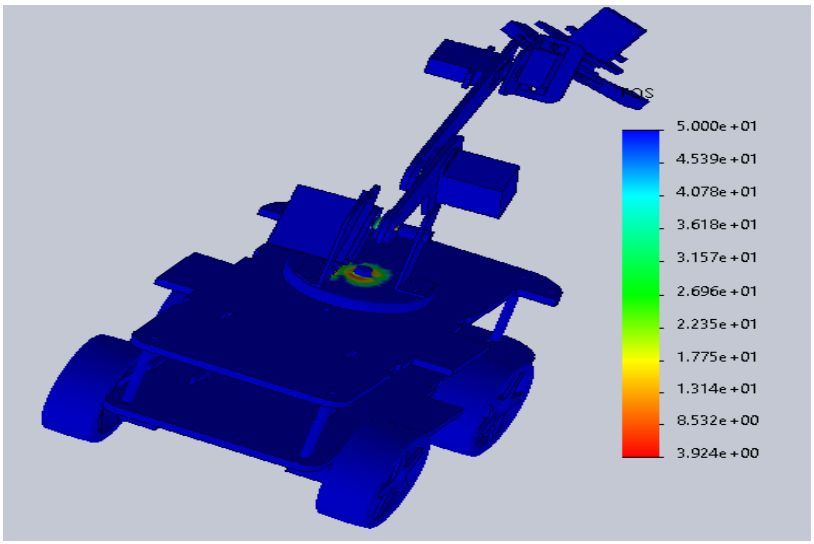



*Figure 3-6 : Facteur de sécurité avec une  échelle de couleur*

### 3.2.2 Simulation dynamique du mouvement

Une étude dynamique a été réalisée à l'aide du module Motion Study de SOLIDWORKS afin d'évaluer le comportement du robot en mouvement. Cette simulation prend en compte les masses, les inerties, les liaisons mécaniques et les actionneurs.

L’objectif principal de cette analyse était :

- De vérifier la stabilité du robot lors des déplacements de la plateforme mobile ;
- D’analyser les efforts transmis aux articulations du bras

En appliquant une charge utile de 200g sur la pince, nous avons pu tracer l'évolution du couple au niveau des articulations. Les résultats ont permis de confirmer que le dimensionnement des moteurs est adapté aux exigences fonctionnelles du système. Les résultats ont permis de déterminer les couples les plus critique pour le choix de chaque actionneur. Par exemple, la Figure 3-7 illustre l'évolution du couple moteur requis au niveau de l'articulation de la base du bras.

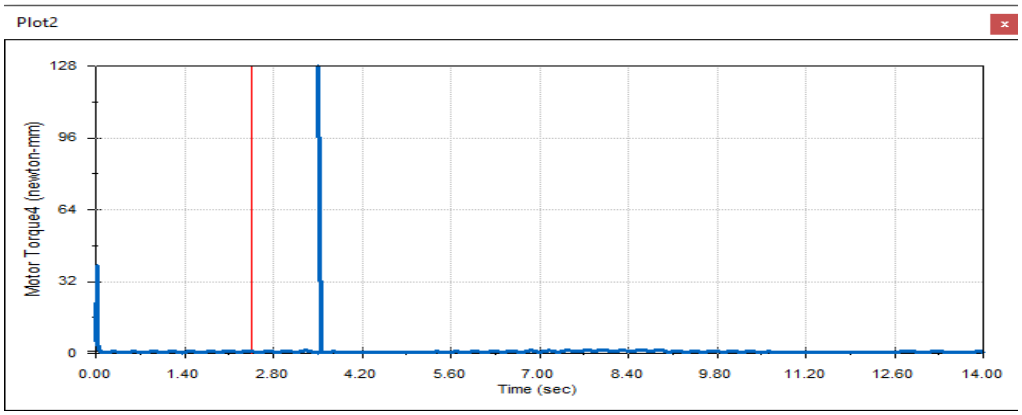



*Figure 3-7 : Couple moteur en N/mm de l'articulation de la base du bras*

La figure 3-8 montre le couple le plus critique (de 91Nmm) lors de cette simulation et montre que plus le segment est proche de sa position d’équilibre (en position verticale) plus le couple requis est moindre.

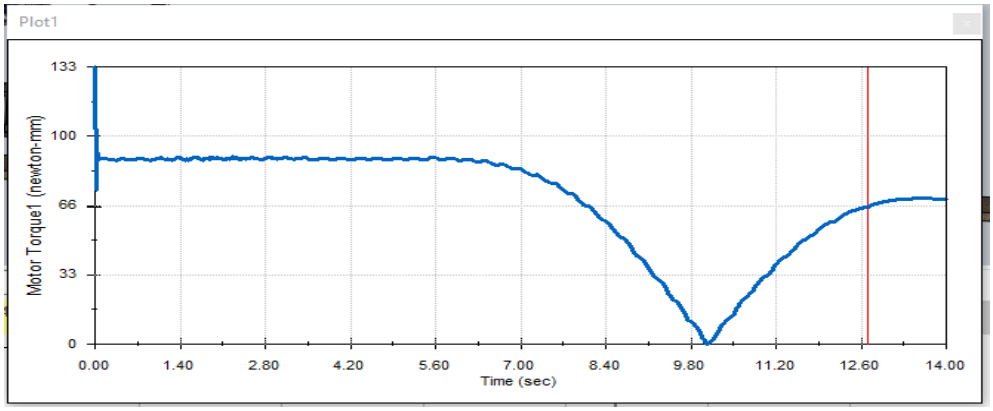



*Figure 3-8 : Couple moteur en N/mm de l'articulation de l'épaule, moteur le plus sollicité*

Enfin, concernant l'extrémité du bras, la Figure 3-9 trace les variations du couple moteur nécessaires pour actionner l'articulation de tangage (pitch) du poignet lors des mouvements.

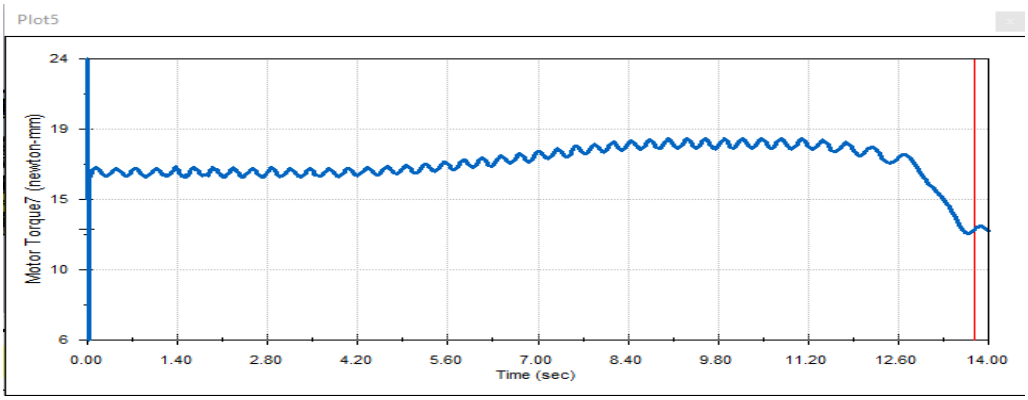



*Figure 3-9 : Couple moteur en N/mm de l'articulation du tangage du poignet(pitch)*

### 3.2.3 Simulation de la cinématique inverse

La simulation de la cinématique inverse a été effectuée sous MATLAB afin de déterminer les angles articulaires nécessaires pour atteindre une position donnée de l'effecteur et d'analyser l'espace de travail (Workspace) atteignable par le robot.

Cette étude est essentielle pour la commande du robot manipulateur, car elle permet de traduire les coordonnées cartésiennes de la position souhaitée en paramètres de commande des servomoteurs.

Le modèle mathématique du robot, basé sur ses paramètres géométriques, a été implémenté dans l'environnement de calcul. Les simulations ont permis de :

- Valider les équations de la cinématique inverse ;
- Vérifier la cohérence des mouvements ;
- S'assurer de l'absence de singularités dans l'espace de travail étudié.

*Tableau 2 : Tableau des paramètres D-H du bras robotisé étudié ($a_i$ et $d_i$ en 'mm')*

| i | $a_i$ | $\alpha_i$ | $d_i$ | $\theta_i$ |
|---|---|---|---|---|
| 1 | 0 | $\pi/2$ | 40 | $\theta_1$ |
| 2 | 100 | 0 | 0 | $\theta_2$ |
| 3 | 130 | 0 | 0 | $\theta_2$ |
| 4 | 0 | $\pi/2$ | 0 | $\theta_4$ |
| 5 | 0 | 0 | 100 | $\theta_5$ |

L'interface de simulation permet d'entrer une position cible cartésienne P (x, y, z) ainsi qu'une orientation désirée pour l'effecteur (Pitch). L'algorithme développé calcule en temps réel les angles articulaires nécessaires pour atteindre cette cible. Le simulateur génère également les multiples solutions cinématiques, permettant à l'utilisateur de choisir entre une configuration "Coude Haut" ou "Coude Bas". L'ensemble de ces fonctionnalités est regroupé au sein de l'interface graphique développée sur MATLAB, qui est présentée à la Figure 3-10.

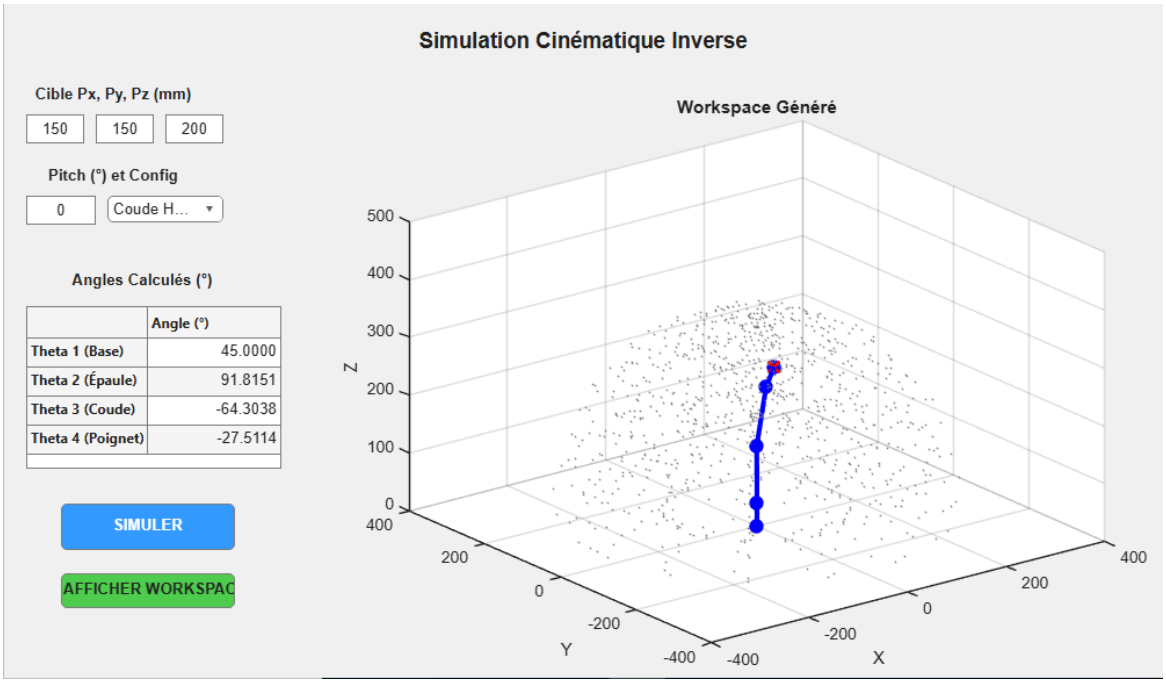



*Figure 3-10:Interface de simulation MATLAB et calcul du MCI pour une position cible et affichage du Workspace*

Comme illustré sur la Figure 3-10, le modèle 3D permet de visualiser la posture du robot. L'espace de travail (Workspace) représente l'ensemble des points que l'effecteur final du robot peut atteindre physiquement avec une porte maximale de 330cm et minimale de 30cm.

## 3.3 Réalisation mécanique

### 3.3.1 Montage du bras Manipulateur 5R

Les 5 actionneurs sont repartis comme suit :

- **Base pivotante (rotation horizontale**) : Moteur pas à pas 28BYJ-48 son couple est de 3 kg.cm a une tension de 5V [43]. Ce choix permet un positionnement angulaire avec précision et un couple de maintien stable. Il correspond au couple maximale de démarrage de la rotation de la base trouver après simulation dynamique.
- **Epaule** : Servomoteur MG996R. ses pignons métalliques et son couple élevé de 13 kg.cm à une tension de 6V pouvant atteindre un courant de maintien de couple de 2,5A. Il supporte la charge totale du bras et assure le mouvement vertical principal.

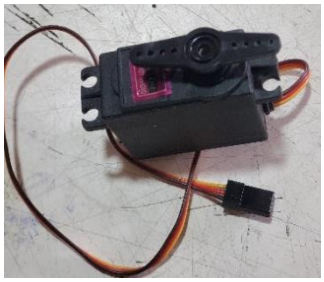


*Figure 3-11 : Servomoteur MG996R hi torque*

- **Coude** : un servomoteur MG996R. assure la puissance nécessaire lors de l'extension horizontales maximales, gère également et la rétraction du bras.
- **Poignet** : un servomoteur MG90S et un servomoteur SG90 permettent d'orienter l'effecteur avant la saisie. Avec leurs couples qui valent respectivement 2,2 kg et 1,6kg à 6V ; et un courant de maintien de couple pouvant atteindre 1A.
- **Pince** : La pince utilise un système de transmission par pignons à denture droite actionner par un servomoteur SG90. Sa conception permet une ouverture et une fermeture symétriques pour assurer une prise stable sur des objets de forme variées. Les mors de la pince sont recouvert d'une couche de matériaux souple (caoutchouc noir) pour augmenter le coefficient de friction. Cela permet de saisir des Object lisse avec une force de serrage maximale.

La forme incurvée des doigts a été choisie pour offrir une polyvalence de saisie, permettant de manipuler des objets cylindriques ou prismatiques. L'organe terminal réalisé, avec ses mors adaptés à la préhension, est détaillée sur la Figure 3-12.

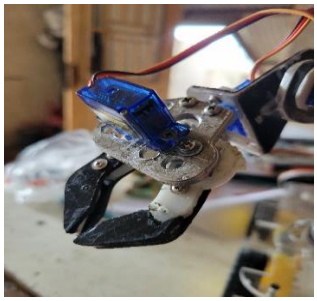


*Figure 3-12 : Pince*

La structure du bras articulé a été réalisée à partir de profilés en alliage d'aluminium 6061 recyclé, d'une épaisseur de 3 mm. La pince, quant à elle, a été fabriquée en plastique technique recyclé.

Les pièces ont été réalisées par usinage manuel.
L'assemblage des éléments du bras a été réalisée par vissage pour garantir une réparation rapide et une facilitée d'assemblage.

### 3.3.2 Montage de la plateforme

- **Base (châssis)** : La base est constituée de deux plaques en acrylique (de 5 mm d'épaisseur pour assurer une base rigide) découpée sur une fraiseuses CNC pour assurer une meilleur précision d'assemblage. L'acrylique étant transparent facilite l'inspection visuelle des composants internes pendant le déplacement du robot. Elle est réalisée pour supporter à la fois le bras robotisé, les composants électroniques et la source d'énergie. Le résultat de cet usinage précis est visible sur la Figure 3-13.

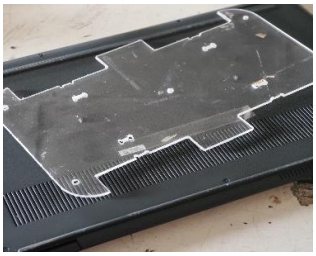


*Figure 3-13 : Support des roues après découpe sur CNC*

- **Roue** : chaque roue (65mm de diamètre) est associée à un moteur à courant continu (DC) muni d'un réducteur de vitesse et fonctionnant entre 3 et 12V. La répartition des quatre roues aux extrémité du châssis offre une base de sustentions large, indispensable pour supporter le poids du bras lors des manipulations. La Figure 3-14 montre l'une de ces roues couplées à son moteur de traction.

*Figure 3-14 : Roue associer au moteur DC*

L'aboutissement de l'intégration de toutes ces parties mécaniques est présenté sur la Figure 3-19, qui dévoile le robot manipulateur mobile entièrement assemblé.

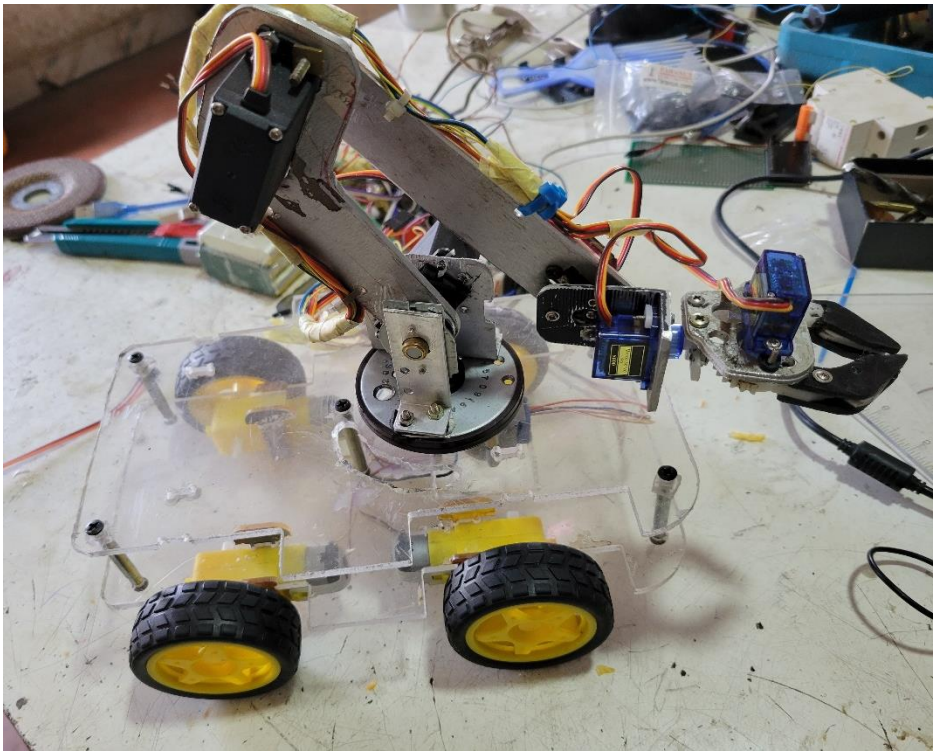


*Figure 3-15 : Robot totalement assemblé*

## 3.4 Réalisation Electronique

La réalisation électronique assure la distribution de l'énergie et les traitements des signaux de commande. Elle est conçue pour garantir la stabilité du bras lors des déplacements de la base.

Pour maximiser les performances, les systèmes électroniques est divise en deux unités de traitement distinctes communiquant entre elles.

## 1. Capteurs embarqués

Afin d'améliorer l'autonomie et la sécurité du robot, plusieurs capteurs ont été intégrés :

- **Capteur ultrasonique HC-SR04 :** Utilisé pour la détection d'obstacles et l'évitement de collision lors des déplacements, ayant une portée de détection allant de 2 à 400cm [44].
- **Unité de mesure inertielle MPU-6050(IMU) :** Permet d'estimer l'inclinaison et la stabilité de la plateforme mobile. Il combine un capteur gyroscopique 3 axes et un accéléromètre 3 axes qui font de lui un capteur 6 axes [45].

Ces capteurs fournissent des informations essentielles pour la prise de décision et la sécurité du système.

## 2. Régulation de tension via le module XL4015

Au lieu d'un simple pont diviseur, peu efficace pour la gestion du courant, nous avons intégré deux module XL4015(convertisseur DC-DC) un pour la puissance et l'autre pour le circuit de commande.

- Fonction : ils abaissent la tension de la batterie principale
- Avantage technique : le XL4015 peut fournir jusqu'à 5A, ce qui est indispensable pour alimenter les 5 servomoteurs du bras. Contrairement à un pont diviseur de tension, il ne dissipe pas l'énergie superflue sous forme de chaleur excessive et maintient une tension constante même quand les moteurs forcent [46].
- Protection : il isole les composants sensibles des fluctuations des tensions générées par les quatre moteurs de la base mobile. Ce composant essentiel pour la gestion énergétique et la protection du circuit est illustré par la Figure 3-15.

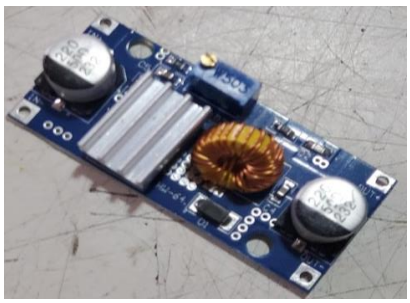


*Figure 3-16: Module XL4015*

## 3. Synthèse du câblage réalise

Les bornes d'alimentations (positives et négatives) des servomoteurs sont connectées ensemble pour simplifier le câblage. Le module XL4015 est place en amont pour servir de filtre énergétique. La communication entre l'ESP32 et l'Arduino s'effectue par les bronches RX/TX, permettant une commande bidirectionnelle.

## 4. Système d'alimentation et gestion d'énergie

L'alimentation constitue le pilier énergétique du prototype, assurant la cohabitation entre la puissance mécanique et la précision électronique.

Pour assurer le couple nécessaire à la base et au bras, nous avons opté pour une configuration de trois batteries Li-Po de 10000mAH montée en série(3s).

- Tension nominale : ce montage délivre une tension totale de 11,1V (3*3.7V).
- Compatibilité de charge : le choix d'une configuration 3S permet l'utilisation d'un chargeur standard de 12,6V, ce qui correspond au seuil de sécurité des cellules.
- Capacité de courant : ce pack peut fournir un courant de charge continue suffisant pour alimenter simultanément les moteurs de traction et les servomoteurs du bras.

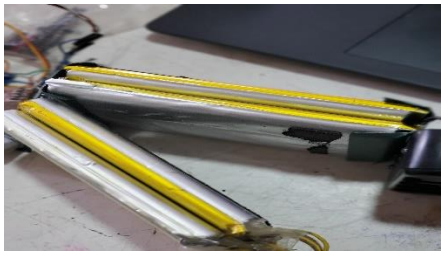


*Figure 3-17 : Batterie Li-Po utilisé*

## 3.5 Système de commande et logiciel

Le pilotage du robot repose sur un programme développe sous Arduino IDE. Il permet de coordonner les mouvements de la base et du bras.

### 1. Unité de commande Motrice (Arduino Uno)

L'Arduino Uno sert de contrôleur de bas niveau. Sa stabilité est idéale pour gérer les interruptions et les signaux PWM en temps réel.

- Gestion du bras : la carte génère 5 signaux PWM pour régler précisément l'angle des servomoteurs via le module PCA9685 pouvant supporter un courant total de 10 ampères indispensable pour la commande des servomoteurs l'Arduino ne pouvant pas supporter leurs courants de maintien.
- Pilotage de la base mobile : elle envoie les commandes aux deux drivers double pont h TB6612FNG pour contrôler les quatre moteurs DC, chaque driver pouvant délivrer jusqu'à 1,2 ampère par canal.
- Pilotage de la base du bras : elle envoie les commandes au driver ULN2003 pour le contrôle du moteur pas à pas.
- Communication : elle reçoit des ordres de hauts niveau (ex : Avancer, Saisir) via une liaison série provenant de l'ESP32.

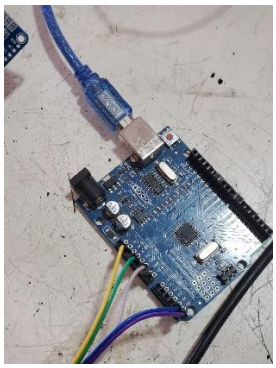


*Figure 3-18: Carte Arduino Uno*

### 2. Unité de Vision et connectivité (ESP32-CAM)

L'ESP32-CAM agit comme le superviseur du robot.

- Flux Vidéo : grâce à son capteur OV2640, elle capture et diffuse une vidéo en temps réel via un serveur Web hébergé localement.
- Intelligence embarquée : sa puissance de calcul supérieure permet d'envisager des fonctions simples des reconnaissances de forme ou de couleurs pour guider la pince.
- Lecture des capteurs : elle lit les informations provenant des capteurs pour la prise de décision en temps réel.
- Interface WI-FI : elle sert de passerelle entre l'utilisateur (Smartphone ou PC) et le robot, transmettant les commandes de pilotage a l'Arduino [47].


*Figure 3-19: Module  ESP-32 CAM*

### 3. Logique de commande

Le logiciel est conçu pour assurer des mouvements fluides et sécurises :

- Mouvement de la base : le code utilise une direction différentielle. Pour tourner, les roues d'un côté s'inversent par rapport à l'autre.
- Protection du bras : des limites d'angles sont fixes dans le code pour empêcher la pince de heurter le châssis ou les roues.
- Vitesse progressive : pour éviter les secousses, la vitesse des moteurs augmente graduellement.

## 3.6 Résultats expérimentaux et validation

L'évaluation des performances du robot manipulateur mobile a été réalisée à travers des essais expérimentaux portant sur la mobilité, la manipulation, la précision et la stabilité du système. Cette analyse vise à vérifier la conformité entre les objectifs de conception et les résultats obtenus lors de la réalisation.

### 3.6.1 Performances de manipulation du bras robotisé

Le bras robotisé à cinq degrés de liberté offre une capacité de positionnement suffisante pour des opérations de préhension d'objets. La cinématique inverse permet d'atteindre les positions cibles dans l'espace de travail défini.
Les essais ont permis de constater :

- Une coordination satisfaisante des articulations ;
- Une capacité de levage de 200g avec le bras en position horizontal ;
- Une bonne stabilité lors de la manipulation à faible vitesse.

La précision du positionnement reste toutefois limitée par les jeux mécaniques des articulations (backlash).

### 3.6.2 Performances de mobilité de la plateforme

La plateforme mobile à quatre roues motrices assure le déplacement du robot sur une surface plane avec une bonne stabilité. Le pilotage permet des déplacements rectilignes, des virages contrôlés ainsi que des rotations sur place.

Les essais ont montré que le robot est capable de :

- Se déplacer de bonne manière à vitesse modérée ;
- Effectuer des rotations avec un rayon faible ;
- Transporter le bras robotisé sans perte d'équilibre.

Cependant, des limites apparaissent lors des changements brusques de direction, notamment des phénomènes de glissement et une légère dérive de trajectoire due aux variations de vitesse entre les moteurs.

### 3.6.3 Performances du système de commande

L'architecture de commande basée sur une unité de bas niveau pour l'exécution des mouvements et une unité de haut niveau pour la supervision et la communication, assure une coordination efficace entre la base mobile et le bras manipulateur.

La communication entre les unités (ESP, Arduino et interface de commande) garantit la transmission fiable des instructions de pilotage et permet le contrôle à distance du robot.

### 3.6.4 Discussion des résultats obtenus

Les résultats expérimentaux obtenus lors des phases de simulation et de réalisation montrent que le robot manipulateur mobile répond globalement aux objectifs fixés lors de la conception. La combinaison d'une plateforme mobile à quatre roues motrices et d'un bras robotisé à cinq degrés de liberté a permis d'obtenir un système capable de se déplacer et de manipuler des objets dans son environnement de travail.

Lors des essais réels, le prototype a démontré une bonne stabilité lors des déplacements et des opérations de manipulation à faible vitesse. La coordination entre la base mobile et le bras robotisé s'est révélée satisfaisante, permettant d'exécuter des tâches simples de préhension et de transport d'objets.

Cependant, des écarts entre les résultats simulés et les performances réelles ont été observés. Ces différences sont principalement dues aux simplifications introduites lors de la modélisation, notamment la négligence des jeux mécaniques, des frottements et des déformations structurelles. De plus, l'absence de capteurs de retour de position limite la précision du positionnement et empêche la correction automatique des erreurs.

Malgré ces contraintes, le prototype réalisé constitue une preuve de concept fonctionnelle démontrant la faisabilité d'un robot manipulateur mobile à faible coût. Les résultats obtenus ouvrent des perspectives d'amélioration, notamment l'intégration de capteurs supplémentaires, l'optimisation des actionneurs et le développement d'une navigation autonome.

## 3.7 Conclusion partielle

Ce chapitre a présenté les différentes étapes ayant conduit à la réalisation et à la validation du robot manipulateur mobile conçu dans ce travail. Les simulations réalisées ont permis de vérifier en amont la faisabilité mécanique et cinématique du système. L'analyse statique sous SOLIDWORKS a confirmé la résistance de la structure aux charges prévues, tandis que l'étude dynamique a permis d'évaluer le comportement du bras lors des mouvements. Par ailleurs, la simulation de la cinématique inverse sous MATLAB a validé la capacité du manipulateur à atteindre des positions cibles dans son espace de travail.

La réalisation mécanique a consisté en la fabrication du bras robotisé et l'assemblage de la plateforme mobile, en tenant compte des contraintes de rigidité, de masse et d'encombrement. L'intégration électronique a permis d'assurer la distribution de l'énergie et la coordination des différents actionneurs et capteurs grâce à une architecture basée sur Arduino Uno, ESP32-CAM et le module PCA9685.

L'implémentation de la commande a permis d'obtenir un fonctionnement coordonné entre la base mobile et le bras manipulateur. Les essais expérimentaux ont confirmé la capacité du robot à se déplacer, à manipuler des objets et à exécuter les tâches prévues, tout en mettant en évidence certaines limitations liées à la précision des actionneurs et à l'absence de retour d'information.

# Conclusion générale

## Contributions

Ce travail de fin de cycle visait à concevoir un système automatisé pour la manutention afin de réduire la pénibilité du travail humain. À l'issue de notre démarche conceptuelle et de nos simulations, nous pouvons apporter des réponses concrètes à nos questions de recherche :

- **Sur la construction d'un manipulateur mobile adapté :** Nous avons démontré qu'il est possible de construire un bras mobile robuste pour un environnement semi-structuré en combinant une base à roues différentielles et un bras sériel anthropomorphe à 5 degrés de liberté. L'usage de l'aluminium pour la structure et de servomoteurs couplés à un moteur pas-à-pas a permis d'optimiser le rapport poids-puissance.
- **Sur la navigation et la sécurité :** Les méthodes employées reposent sur une architecture électronique hiérarchisée. L'utilisation d'une carte ESP32-CAM agissant comme superviseur, couplée à des capteurs ultrasoniques (HC-SR04) pour l'évitement d'obstacles et à une unité inertielle (MPU-6050) pour le contrôle de l'inclinaison, garantit des déplacements stables et sécurisés pour l'environnement humain immédiat.
- **Sur l'impact industriel :** Les simulations statiques et dynamiques réalisées sous SolidWorks, associées à la modélisation cinématique sous MATLAB, confirment que le système peut soulever et transporter avec précision une charge de 200g. Les faits mettent en évidence qu'à plus grande échelle, un tel dispositif diminue drastiquement la charge physique des ouvriers et augmente le rendement logistique.

## Critique du travail

Malgré la validation de nos hypothèses lors des simulations, ce travail présente certaines limites au cadre d'un prototype. Premièrement, le choix de l'acrylique pour le châssis limite la capacité de charge maximale et la résistance aux fortes vibrations industrielles. Deuxièmement, la coordination bras-plateforme reste séquentielle (le robot s'arrête pour manipuler) ; une manipulation dynamique (manipuler tout en roulant) n'a pas pu être implémentée à ce stade. Enfin, la détection des obstacles par ultrasons manque de précision pour cartographier un environnement complexe et réellement imprévisible

## Travaux futurs de recherche/Perspectives

Les limites de ce travail constituent autant d'opportunités pour des recherches ultérieures. Pour améliorer ce prototype, les travaux futurs pourraient se concentrer sur les axes suivants :

- **Amélioration de la perception** : Exploiter pleinement la caméra de l'ESP32-CAM en y intégrant des algorithmes de vision par ordinateur
- **Navigation autonome** avancée : Remplacer les capteurs ultrasoniques par un LiDAR 2D pour implémenter un algorithme SLAM (Simultaneous Localization and Mapping), permettant au robot de cartographier son atelier en temps réel. Ajouter des encodeurs au niveaux de roues pour une estimation précise de la distance parcourue. Ajouter de capteur des capteurs de position au niveau du bras et une caméra au niveau de l'effecteur pour une correction automatique de la position.
- **Optimisation mécanique** : Étudier l'intégration de matériaux composites avancés pour alléger la structure, et optimisé l'emplacement des actionneurs du bras pour augmenter la charge maximale du robot.
- **Amélioration de la commande** : Développer un meilleur algorithme de commande pour rendre le robot totalement autonome.

# Bibliographie


[1] B. Siciliano and O. Khatib, Springer HandBook of Robotics, Berlin: Springer, 2016.

[2] P.R.Wuman, R.D'Andrea and M.Mountz, Coordinating Hundreds of cooperative Autonomous Vehicles in warehouses, AI Magazine, 2008.

[3] Institut National de Recherche et de Securite(INRS), Troubles musculosqueletique(TMS) et manutention manuelle: prevention des risques profetionnel, Paris, 2019.

[4] N.Ghodsian, D.Obukov, L.Ciarlietta and P.Freass, “Mobile manipulators in Industry 4.0: A review of developments for industrial applications,”, Sensors, 2023.

[5] Boutaani, Systheme mecanique articule, 2020.

[6] AFNOR/ ISO, Robot et dispositif robotique- vocabulaire, Paris: AFNOR, 2021.

[7] Le Robert, Le Petit Robert de la Langue Francaise, Paris: Le Robert, 2023.

[8] J. M. Jordan, Robots, Cambrigde,MA: The MIT Press, 2016.

[9] La Redaction, «Qu'est qu'un robot? Decouvrez ses caracteristiques et fonctionnalites,» UMVIE, 11 Octobre 2023. [En ligne]. Available: https://umvie.com/quest-ce-quun-robot-decouvrez-les-caracteristiques-et-fonctionnalites/.. [Accès le 12 Novembre 2025].

[10] M. A. Haj Salah, «Quels sont les different types de robots?,» Robotic Tech, 1 Novemdre 2022. [En ligne]. Available: https://www.robotique.tech/blog/quels-sont-les-different-types-de-robots/. [Accès le 14 Novembre 2025].

[11] R. Siegwart, I. R. Nourbakhsh and D. Scaramuzza, Introduction to Autonomous Mobile Robots, 2nd ed, Cambridge, MA, USA: The MIT Press, 2011.

[12] blog manutention, «"Tout savoir sur les robots mobiles autonomes (AMR),",» Blog Manutention, 15 octobre 2021. [En ligne]. Available: https://www.blog-manutention.fr/tout-savoir-sur-les-robots-mobiles-autonomes-amr.html.. [Accès le 09 fevrier 2026].

[13] Engineered Arts, «Ameca - Le robot humanoïde le plus avancé au monde,» Engineered Arts, juin 2024. [En ligne]. Available: https://engineeredarts.com/fr/robots/ameca.. [Accès le 09 fevrier 2026].

[14] TE Connectivity, «Les differents types de robots pour la production industrielles,» 2025. [En ligne]. Available: https://www.te.com/fr/technologies/industrial-automation/understanding-indistrial-robots.html. [Accès le Novembre 15 2025].

[15] G. Franke, «Robots articulés | Robotique et manutention | Applications,» Franke-gmbh.fr, 2024. [En ligne]. Available: : https://www.franke-gmbh.fr/roulements-a-billes/applications/robotique-et-manutention/robots-articules/.. [Accès le 09 fevrier 20226].

[16] Mecalux Belgique, «Cobots: la robotique collaborative dans l'entrepot,» 25 septembre 2025. [En ligne]. Available: https://www.schmalz.com/fr/support/ savoir-faire/ glossaire/ cobot. [Accès le 10 decembre 2025].

[17] L. Vianello, S. Ivaldi, A. Aubry and L. Peternel, The effects of role transitions and adaptation in human–cobot collaboration », , Journal of Intelligent Manufacturing, 2023.

[18] T. Albercht and G. Ulrich, Automated Guided Vehicle system: A guide - With Pratical Applications- About the Technology-For Planning, Berlin: Springer-Vieweg, 2015.

[19] Intech Group, "Robot AGV intech-group.vn.," [Online]. Available: https://intech-group.vn/robot-agv.htm. [Accessed 09 fevrier 2026].

[20] T. Pappas, "Hybrid Driving-Flying Robot Swarms Could One Day Help in Disasters," Live Science, 27 juin 2017. [Online]. Available: https://www.livescience.com/59689-hybrid-driving-flying-robot-swarms.html.. [Accessed 09 fevrier 2026].

[21] T. Sun, X. Xiang, W. Su, H. Wu and Y. Song, A transformable wheel-legged mobile robot: Design, analysis and experiment », Robotics and Autonomous Systems, volume 98, 2017.

[22] Fdatabot, «Top 20 des meilleurs fabricants et fournisseurs de robots mobiles dans le monde,» 2924. [En ligne]. Available: https://www.fdatabot.com/fr/best-20-mobile-robot-manufacturers-and-suppliers-worldwide/. [Accès le 09 fevrier 2026].

[23] Z. Li, L. Chen, Q. Zheng, X. Dou and L. Yang, « Control of a path following caterpillar robot based on a sliding mode variable structure algorithm », volume 186, 2019.

[24] Salewireov, «Bras robotique avec unité de commande - Kit de construction 5 DDL,» Salewireov.click, 2024. [En ligne]. Available: https://salewireov.click/product_details/39551874.html.. [Accès le 09 fevrier 2026].

[25] R. Buchanan, T. Bandyopadhyay, M. Bjelonic, L. Wellhausen, M. Hutter and N. Kottege, Walking posture Adaptation for Legged Robot Navigation in Confined Spaces », Robotics and Automation Letters, 2019.

[26] J. Dupeyroux, J. R. Serres and S. Viollet, « AntBot: A six-legged walking robot able to navigate like desert ants in unknown outdoor environments », Science Robotics, 2019.

[27] Shutterstock, «Multicoptère : images, photos et objets vectoriels de stock,» [En ligne]. Available: https://www.shutterstock.com/fr/search/multicopte. [Accès le 09 02 2026].

[28] R. Siddall, R. Zufferey and S. F. Armanini, « The Natural Robotics Contest: Crowdsourced biomimetic design », Bioinspiration & Biomimetics,, Université de Surrey, 2023.

[29] E. Back, «Ces robots modulaires conçus pour l'exploration spatiale se transforment à volonté !,» science futura, 2023 juin 17. [En ligne]. Available: https://www.futura-sciences.com/tech/actualites/robotique-ces-robots-modulaires-concus-exploration-spatiale-transforment-volonte-105900/. [Accès le 09 fevrier 2026].

[30] D. W. Haldane, M. M. Plecnik, J. K. Yim and R. S. Fearing, Robotic vertical jumping agility via series-elastic power modulation, Science Robotics, 2016.

[31] J. J. Craig, Introduction to Robotics: Mechanics and Control, Pearson Education, 2005.

[32] EVS, Quelles sont les pieces et composant des robots industriels, Wenling: EVST, 2020.

[33] Office québécois de la langue française, Robotique : structures de base, Montréal: Le Grand dictionnaire terminologique, 1988.

[34] X. Zang, S. Iqbal, Y. Zhu, L. Fan and J. Zhao, Applications of Chaotic Dynamics in Robotics, International Journal of Advanced Robotic System, 2016.

[35] C. Jensen and D. DeStefano, Utilizing Failure Information for Mission Analysis for Complex Systems, Houston: International Mechanical Engineering Congress & Exposition (IMECE2015), 2015.

[36] N. Slimane, Systèmes de Perception en Robotique Mobile (Support de cours de Master 2 Automatique et Systèmes), Alger: Université de Batna 2, 2024.

[37] R. Ikeura et S. M. M. Rahman, Modélisation cinématique et analyse des coordonnées des liaisons du robot LabVolt 5150, Proceedings of the International Conference on Robotics and Biomimetics, 2017.

[38] S. S. Mohanty, S. K. Pradhan, S. S. Pattnaik and S. K. Pradhan, Recent Advances in Renewable Energy Systems, Singapore: Singapore: Springer Nature, 2022.

[39] W. Spong, S. Hutchunson and M. Vidyasagar, Robot Modeling and Control, John Wiley & Sons, 2005.

[40] Ashby and F. Michael, Materials Selection in Mechanical Design, Butterworth-Heinemann, 1992.

[41] Dassault Systemes, "SOLIDWORKS," Dassault Systemes, Vélizy-Villacoublay, 2025.

[42] The MathWork Inc., "MATLAB," The MathWork Inc., Natick, 2025.

[43] MikroElektronika, "28BYJ-48 Stepper Motor Datasheet," 2019. [Online]. Available: https://www.digikey.com.mx/htmldatasheets/production/1839399/0/0/1/28byj-48.html. [Accessed fevrier 2025].

[44] Cytron Technologies, HC-SR04 User's Manual, Grobotronics, 2013.

[45] InvenSense, MPU-6050: Six-Axis (Gyro + Accelerometer) MEMS MotionTracking™ Devices., TDK InvenSense, 2026.

[46] XLSEMI (Shanghai Xinlong Semiconductor Technology Co., Ltd)., 5A 180KHz 36V Buck DC to DC Converter, Alldatasheet.

[47] Espressif Systems, «ESP32 Series Datasheet: 2.4 GHz Wi-Fi and Bluetooth Combo Chip, Version 4.0,» Espressif Systems, 2023.

Annexe A

# Code ESP32-CAM

```
#include "esp_camera.h"
#include <WiFi.h>
#include <WebServer.h>
#include <ArduinoJson.h>
const char* ssid = "ROBOT_CAM";
const char* password = "robot12345";
const char* ap_ssid = "ESP32_ROBOT";
const char* ap_password = "12345678";
#define PWDN_GPIO_NUM     32
#define RESET_GPIO_NUM    -1
#define XCLK_GPIO_NUM      0
#define SIOD_GPIO_NUM     26
#define SIOC_GPIO_NUM     27
#define Y9_GPIO_NUM       35
#define Y8_GPIO_NUM       34
#define Y7_GPIO_NUM       39
#define Y6_GPIO_NUM       36
#define Y5_GPIO_NUM       21
#define Y4_GPIO_NUM       19
#define Y3_GPIO_NUM       18
#define Y2_GPIO_NUM        5
#define VSYNC_GPIO_NUM    25
#define HREF_GPIO_NUM     23
#define PCLK_GPIO_NUM     22
#define FLASH_LED 4
#define FOCAL_LENGTH_PX 500.0     // Focale en pixels
#define IMAGE_WIDTH 320.0         // Largeur image (QVGA)
#define IMAGE_HEIGHT 240.0        // Hauteur image (QVGA)
#define CENTER_X (IMAGE_WIDTH/2)  // Centre image X
#define CENTER_Y (IMAGE_HEIGHT/2) // Centre image Y
// Géométrie robot (hauteur caméra par rapport au sol)
#define CAMERA_HEIGHT 50.0       // Hauteur caméra en mm (à mesurer sur
robot)
#define CAMERA_ANGLE_DEG 0.0     // Angle inclinaison caméra vers le bas
(degrés)
```

```
// Seuils détection couleur (HSV)
struct ColorRange {
  uint8_t hMin, hMax;
  uint8_t sMin, sMax;
  uint8_t vMin, vMax;
};
// Couleurs prédéfinies
ColorRange COLOR_RED    = {0, 10, 100, 255, 100, 255};      // Rouge
ColorRange COLOR_BLUE   = {100, 130, 100, 255, 100, 255};   // Bleu
ColorRange COLOR_GREEN  = {40, 80, 100, 255, 100, 255};     // Vert
ColorRange COLOR_YELLOW = {20, 40, 100, 255, 100, 255};     // Jaune
ColorRange targetColor = COLOR_RED;  // Couleur cible par défaut
struct ObjectDetected {
  bool detected;
  int centerX;      // Position pixel X
  int centerY;      // Position pixel Y
  int area;         // Surface en pixels
  float worldX;     // Position réelle X (mm)
  float worldY;     // Position réelle Y (mm)
  float worldZ;     // Position réelle Z (mm)
  float distance;   // Distance capteur ultrason (cm)
};
ObjectDetected currentObject;
// État système
enum SystemState {
  STATE_IDLE,
  STATE_SCANNING,
  STATE_MOVING,
  STATE_MANIPULATING,
  STATE_DEPOSITING,
  STATE_ERROR
};
SystemState currentState = STATE_IDLE;
bool autoMode = false;
float depositX = 200.0;
float depositY = 100.0;
float depositZ = 100.0;
WebServer server(80);
void setup() {
  Serial.begin(115200);
  Serial.println("=== ESP32-CAM - VISION + COORDONNEES ===");
  pinMode(FLASH_LED, OUTPUT);
  digitalWrite(FLASH_LED, LOW);
  // Initialisation caméra
  if (initCamera()) {
```

```
    Serial.println("[OK] Caméra initialisée");
  } else {
    Serial.println("[ERROR] Échec init caméra");
  }
  // Connexion Wi-Fi
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  int attempts = 0;
  while (WiFi.status() != WL_CONNECTED && attempts < 20) {
    delay(500);
    Serial.print(".");
    attempts++;
  }
  if (WiFi.status() == WL_CONNECTED) {
    Serial.println("\n[OK] Wi-Fi connecté");
    Serial.print("IP: ");
    Serial.println(WiFi.localIP());
  } else {
    Serial.println("\n[INFO] Mode AP activé");
    WiFi.mode(WIFI_AP);
    WiFi.softAP(ap_ssid, ap_password);
    Serial.print("AP IP: ");
    Serial.println(WiFi.softAPIP());
  }
  setupWebServer();
  server.begin();
  Serial.println("[OK] Serveur web démarré");

  // Initialisation Arduino Nano
  sendCommand("<REST>");
  delay(2000);
  Serial.println("[READY] Système prêt - Vision activée\n");
}
void loop() {
  server.handleClient();
  // Mode automatique avec vision
  if (autoMode) {
    executeAutoSequenceWithVision();
  }
  // Lecture réponses Arduino
  if (Serial.available()) {
    String response = Serial.readStringUntil('\n');
    Serial.println("Arduino: " + response);
    // Parser réponse capteur distance
    if (response.startsWith("[DATA] DIST_FRONT:")) {
```

```
      float dist = response.substring(18).toFloat();
      currentObject.distance = dist;
    }
  }
  delay(50);
}
bool initCamera() {
  camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
  config.pin_d7 = Y9_GPIO_NUM;
  config.pin_xclk = XCLK_GPIO_NUM;
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;
  config.pin_href = HREF_GPIO_NUM;
  config.pin_sscb_sda = SIOD_GPIO_NUM;
  config.pin_sscb_scl = SIOC_GPIO_NUM;
  config.pin_pwdn = PWDN_GPIO_NUM;
  config.pin_reset = RESET_GPIO_NUM;
  config.xclk_freq_hz = 20000000;
  config.pixel_format = PIXFORMAT_RGB565;  // RGB pour traitement couleur
  // QVGA pour traitement rapide
  config.frame_size = FRAMESIZE_QVGA;  // 320x240
  config.jpeg_quality = 12;
  config.fb_count = 1;
  esp_err_t err = esp_camera_init(&config);
  if (err == ESP_OK) {
    // Configuration capteur
    sensor_t * s = esp_camera_sensor_get();
    s->set_brightness(s, 0);     // -2 to 2
    s->set_contrast(s, 0);       // -2 to 2
    s->set_saturation(s, 0);     // -2 to 2
    s->set_whitebal(s, 1);       // 0 = disable , 1 = enable
    s->set_awb_gain(s, 1);       // 0 = disable , 1 = enable
    s->set_wb_mode(s, 0);        // 0 to 4
  }
  return (err == ESP_OK);
}
```

```
bool detectObjectByColor(camera_fb_t* fb, ColorRange color) {
  if (!fb || fb->format != PIXFORMAT_RGB565) return false;
  uint16_t* pixels = (uint16_t*)fb->buf;
  int width = fb->width;
  int height = fb->height;
  // Variables pour calculer centre de masse
  long sumX = 0;
  long sumY = 0;
  long count = 0;
  int maxArea = 0;
  // Parcourir image et détecter pixels de la couleur cible
  for (int y = 0; y < height; y++) {
    for (int x = 0; x < width; x++) {
      uint16_t pixel = pixels[y * width + x];
      // Conversion RGB565 → RGB888
      uint8_t r = ((pixel >> 11) & 0x1F) << 3;
      uint8_t g = ((pixel >> 5) & 0x3F) << 2;
      uint8_t b = (pixel & 0x1F) << 3;
      // Conversion RGB → HSV (simplifié)
      uint8_t maxC = max(r, max(g, b));
      uint8_t minC = min(r, min(g, b));
      uint8_t delta = maxC - minC;
      // V (Value)
      uint8_t v = maxC;
      // S (Saturation)
      uint8_t s = (maxC == 0) ? 0 : (delta * 255 / maxC);
      // H (Hue)
      uint8_t h = 0;
      if (delta != 0) {
        if (maxC == r) {
          h = ((g - b) * 60 / delta) % 360;
        } else if (maxC == g) {
          h = ((b - r) * 60 / delta + 120) % 360;
        } else {
          h = ((r - g) * 60 / delta + 240) % 360;
        }
      }
      h = h / 2;  // Ramener à [0-180] pour compatibilité OpenCV
      // Vérifier si pixel correspond à couleur cible
      bool matchColor = (h >= color.hMin && h <= color.hMax &&
                         s >= color.sMin && s <= color.sMax &&
                         v >= color.vMin && v <= color.vMax);
      if (matchColor) {
        sumX += x;
        sumY += y;
```

```
        count++;
       }
     }
   }
   // Si assez de pixels détectés
   if (count > 100) {  // Seuil minimum de pixels
     currentObject.detected = true;
     currentObject.centerX = sumX / count;
     currentObject.centerY = sumY / count;
     currentObject.area = count;
     Serial.print("[VISION] Objet détecté: Pixel(");
     Serial.print(currentObject.centerX);
     Serial.print(", ");
     Serial.print(currentObject.centerY);
     Serial.print(") Area: ");
     Serial.println(currentObject.area);
     return true;
   }
   currentObject.detected = false;
   return false;
}
void pixelToWorldCoordinates(int pixelX, int pixelY, float distanceCm) {
   float distanceMm = distanceCm * 10.0;
   // Correction angle caméra (inclinée vers le bas)
   float angleRad = CAMERA_ANGLE_DEG * PI / 180.0;
   // Décalage pixel par rapport au centre image
   float deltaX = pixelX - CENTER_X;
   float deltaY = pixelY - CENTER_Y;
   // Calcul coordonnées dans repère caméra
   float camX = (deltaX * distanceMm) / FOCAL_LENGTH_PX;
   float camY = (deltaY * distanceMm) / FOCAL_LENGTH_PX;
   float camZ = distanceMm;
   // Transformation repère caméra → repère robot (caméra inclinée)
   // X robot = X caméra (latéral, pas d'effet)
   // Y robot = Z caméra * cos(angle) - Y caméra * sin(angle) (profondeur
corrigée)
   // Z robot = hauteur caméra - (Z caméra * sin(angle) + Y caméra *
cos(angle))
   currentObject.worldX = camX;
   currentObject.worldY = camZ * cos(angleRad) - camY * sin(angleRad);
   currentObject.worldZ = CAMERA_HEIGHT - (camZ * sin(angleRad) + camY *
cos(angleRad));
   Serial.print("[COORDONNEES] Monde: X=");
   Serial.print(currentObject.worldX);
   Serial.print(" Y=");
```

```
  Serial.print(currentObject.worldY);
  Serial.print(" Z=");
  Serial.println(currentObject.worldZ);
}
bool scanForObject() {
  Serial.println("[SCAN] Recherche objet...");
  // 1. Capturer image
  camera_fb_t* fb = esp_camera_fb_get();
  if (!fb) {
    Serial.println("[ERROR] Échec capture image");
    return false;
  }
  // 2. Détecter objet par couleur
  bool detected = detectObjectByColor(fb, targetColor);
  // Libérer buffer image
  esp_camera_fb_return(fb);
  if (!detected) {
    Serial.println("[SCAN] Aucun objet détecté");
    return false;
  }
  // 3. Mesurer distance avec ultrason
  sendCommand("<DIST>");
  delay(100);  // Attendre réponse Arduino
  // currentObject.distance est mis à jour dans loop() via réponse Arduino
  // 4. Calculer coordonnées réelles
  if (currentObject.distance > 0 && currentObject.distance < 200) {
    pixelToWorldCoordinates(currentObject.centerX, currentObject.centerY,
                            currentObject.distance);
    Serial.println("[SUCCESS] Objet localisé avec coordonnées calculées");
    return true;
  }
  Serial.println("[ERROR] Distance ultrason invalide");
  return false;
}
void executeAutoSequenceWithVision() {
  static unsigned long lastAction = 0;
  unsigned long now = millis();
  switch (currentState) {
    case STATE_IDLE:
      Serial.println("[AUTO] Phase 1: Scan environnement");
      currentState = STATE_SCANNING;
      lastAction = now;
      break;
    case STATE_SCANNING:
      if (now - lastAction > 500) {
```

```
    // Scanner avec vision + ultrason
    if (scanForObject()) {
      Serial.println("[AUTO] Objet détecté, calcul coordonnées...");
      currentState = STATE_MOVING;
    } else {
      Serial.println("[AUTO] Aucun objet, rotation recherche...");
      sendCommand("<RIGHT:100>");
      delay(500);
      sendCommand("<STOP>");
    }
    lastAction = now;
  }
  break;
case STATE_MOVING:
  if (now - lastAction > 500) {
    Serial.println("[AUTO] Phase 2: Approche objet détecté");
    // Calculer déplacement nécessaire
    // Si objet à gauche (X < 0), tourner à gauche
    // Si objet à droite (X > 0), tourner à droite
    if (abs(currentObject.worldX) > 30) {  // Désalignement > 30mm
      if (currentObject.worldX < 0) {
        sendCommand("<LEFT:120>");
      } else {
        sendCommand("<RIGHT:120>");
      }
      delay(300);
      sendCommand("<STOP>");
      // Rescanner après rotation
      currentState = STATE_SCANNING;
    } else {
      // Aligné, avancer jusqu'à distance optimale
      if (currentObject.worldY > 200) {  // Trop loin
        sendCommand("<FWD:150>");
        delay(500);
        sendCommand("<STOP>");
        currentState = STATE_SCANNING;  // Rescanner
      } else {
        currentState = STATE_MANIPULATING;
      }
    }
    lastAction = now;
  }
  break;
case STATE_MANIPULATING:
  if (now - lastAction > 500) {
```

```
        Serial.println("[AUTO] Phase 3: Saisie objet");
        // Ouvrir pince
        sendCommand("<GRIP:0>");
        delay(500);
        // Position bras vers objet détecté (coordonnées réelles
calculées!)
        String cmd = "<IK:" + String(currentObject.worldX) + "," +
                             String(currentObject.worldY) + "," +
                             String(currentObject.worldZ) + ",-90,0>";
        sendCommand(cmd);
        delay(3000);
        // Fermer pince
        sendCommand("<GRIP:1>");
        delay(1000);
        // Lever
        String cmdUp = "<IK:" + String(currentObject.worldX) + "," +
                               String(currentObject.worldY) + ",200,-90,0>";
        sendCommand(cmdUp);
        delay(2000);
        currentState = STATE_DEPOSITING;
        lastAction = now;
      }
      break;
    case STATE_DEPOSITING:
      if (now - lastAction > 500) {
        Serial.println("[AUTO] Phase 4: Dépose objet");
        // Aller vers zone dépôt
        sendCommand("<FWD:150>");
        delay(2000);
        sendCommand("<STOP>");
        // Position dépôt
        String cmd = "<IK:" + String(depositX) + "," +
                             String(depositY) + "," +
                             String(depositZ) + ",-90,0>";
        sendCommand(cmd);
        delay(3000);
        // Ouvr pince
        sendCommand("<GRIP:0>");
        delay(1000);
        // Repos
        sendCommand("<REST>");
        delay(2000);
        // Reculer
        sendCommand("<BWD:150>");
        delay(2000);
```

```
        sendCommand("<STOP>");
        Serial.println("[AUTO] Séquence terminée avec succès");
        autoMode = false;
        currentState = STATE_IDLE;
      }
      break;
    case STATE_ERROR:
      Serial.println("[ERROR] Erreur système");
      sendCommand("<ESTOP>");
      autoMode = false;
      currentState = STATE_IDLE;
      break;
  }
}
void setupWebServer() {
  server.on("/", HTTP_GET, handleRoot);
  server.on("/stream", HTTP_GET, handleStream);
  server.on("/api/scan", HTTP_GET, handleScan);
  server.on("/api/move", HTTP_POST, handleMove);
  server.on("/api/arm", HTTP_POST, handleArm);
  server.on("/api/grip", HTTP_POST, handleGrip);
  server.on("/api/auto", HTTP_POST, handleAuto);
  server.on("/api/stop", HTTP_POST, handleStop);
  server.on("/api/status", HTTP_GET, handleStatus);
  server.on("/api/color", HTTP_POST, handleSetColor);
}
void handleRoot() {
  server.send(200, "text/html", "<h1>Interface Web ici</h1>");
}
void handleStream() {
}
void handleScan() {
  bool found = scanForObject();
  StaticJsonDocument<200> doc;
  doc["detected"] = found;
  if (found) {
    doc["pixelX"] = currentObject.centerX;
    doc["pixelY"] = currentObject.centerY;
    doc["worldX"] = currentObject.worldX;
    doc["worldY"] = currentObject.worldY;
    doc["worldZ"] = currentObject.worldZ;
    doc["distance"] = currentObject.distance;
  }
  String json;
  serializeJson(doc, json);
```

```
  server.send(200, "application/json", json);
}
void handleMove() {
  StaticJsonDocument<200> doc;
  deserializeJson(doc, server.arg("plain"));
  int speedL = doc["speedL"];
  int speedR = doc["speedR"];
  String cmd = "<MOVE:" + String(speedL) + "," + String(speedR) + ">";
  sendCommand(cmd);
  server.send(200, "text/plain", "OK");
}
void handleArm() {
  StaticJsonDocument<200> doc;
  deserializeJson(doc, server.arg("plain"));
  float x = doc["x"];
  float y = doc["y"];
  float z = doc["z"];
  float theta = doc["theta"];
  float theta5 = doc["theta5"];
  String cmd = "<IK:" + String(x) + "," + String(y) + "," + String(z) + ","
+
               String(theta) + "," + String(theta5) + ">";
  sendCommand(cmd);
  server.send(200, "text/plain", "OK");
}
void handleGrip() {
  StaticJsonDocument<200> doc;
  deserializeJson(doc, server.arg("plain"));
  int state = doc["state"];
  sendCommand("<GRIP:" + String(state) + ">");
  server.send(200, "text/plain", "OK");
}
void handleAuto() {
  autoMode = true;
  currentState = STATE_IDLE;
  server.send(200, "text/plain", "Auto mode avec vision activé");
}
void handleStop() {
  autoMode = false;
  sendCommand("<STOP>");
  sendCommand("<REST>");
  currentState = STATE_IDLE;
  server.send(200, "text/plain", "Arrêt");
}
void handleStatus() {
```

```
  String state;
  switch (currentState) {
    case STATE_IDLE: state = "En attente"; break;
    case STATE_SCANNING: state = "Scan vision"; break;
    case STATE_MOVING: state = "Déplacement"; break;
    case STATE_MANIPULATING: state = "Manipulation"; break;
    case STATE_DEPOSITING: state = "Dépose"; break;
    case STATE_ERROR: state = "Erreur"; break;
  }
  StaticJsonDocument<300> doc;
  doc["state"] = state;
  doc["objectDetected"] = currentObject.detected;
  if (currentObject.detected) {
    doc["x"] = currentObject.worldX;
    doc["y"] = currentObject.worldY;
    doc["z"] = currentObject.worldZ;
  }
  String json;
  serializeJson(doc, json);
  server.send(200, "application/json", json);
}
void handleSetColor() {
  StaticJsonDocument<200> doc;
  deserializeJson(doc, server.arg("plain"));
  String color = doc["color"];
  if (color == "red") targetColor = COLOR_RED;
  else if (color == "blue") targetColor = COLOR_BLUE;
  else if (color == "green") targetColor = COLOR_GREEN;
  else if (color == "yellow") targetColor = COLOR_YELLOW;
  server.send(200, "text/plain", "Couleur cible: " + color);
}
void sendCommand(String cmd) {
  Serial.println(cmd);
  delay(50);
}
```

Annexe B

# Code Arduino

```
#include <Wire.h>
#include <Stepper.h>
#include <Adafruit_PWMServoDriver.h>
#include <MPU6050.h>
// PARAMÈTRES D-H (mm)
#define DH_D1 40.0
#define DH_A2 100.0
#define DH_A3 130.0
#define DH_D5 100.0
// Limites articulaires (degrés)
const float THETA_MIN[5] = {-70, -90, -90, -90, -90};
const float THETA_MAX[5] = { 70,  90,  90,  90,  90};
// CONFIGURATION PINS - BASE MOBILE (2× TB6612FNG, 1 par côté)
// Driver GAUCHE
#define PWM_LEFT   3
#define DIR1_LEFT  2
#define DIR2_LEFT  8
//  Driver DROIT
#define PWM_RIGHT  5
#define DIR1_RIGHT 6
#define DIR2_RIGHT 7
//  STANDBY
#define STBY 10
// CONFIGURATION PINS - STEPPER 28BYJ-48 (rotation base bras)
#define STEPS_PER_REV 2048
#define STEPPER_IN1 11
#define STEPPER_IN2 12
#define STEPPER_IN3 13
#define STEPPER_IN4 A0
// CONFIGURATION PINS - HC-SR04
#define TRIG_PIN_FRONT A1
#define ECHO_PIN_FRONT 4
#define TRIG_PIN_BACK  A4
#define ECHO_PIN_BACK  A5
// CONFIGURATION PCA9685
```

```
#define SERVOMIN 150
#define SERVOMAX 600
#define SERVO_FREQ 60
// CANAUX PCA9685
#define SERVO_EPAULE      4  // CH4 - Épaule (θ₂)
#define SERVO_COUDE       0  // CH0 - Coude (θ₃)
#define SERVO_POIGNET_INC 2  // CH2 - Poignet Inclinaison (θ₄)
#define SERVO_POIGNET_ROT 3  // CH3 - Poignet Rotation (θ₅)
#define SERVO_PINCE       1  // CH1 - Pince
// OBJETS GLOBAUX
Adafruit_PWMServoDriver pwm = Adafruit_PWMServoDriver();
Stepper stepper(STEPS_PER_REV, STEPPER_IN1, STEPPER_IN3, STEPPER_IN2,
STEPPER_IN4);
MPU6050 mpu;
// STRUCTURES DE DONNÉES
struct ArmPosition {
  float theta1;  // Rotation base (stepper)
  float theta2;  // Épaule
  float theta3;  // Coude
  float theta4;  // Poignet inclinaison
  float theta5;  // Pince rotation
  float pince;   // Ouverture pince
};
// VARIABLES POUR CONTRÔLE VITESSE SERVOS
// Angles actuels de chaque servo
float currentAngles[5] = {0, 90, 90, 90, 90};
// Vitesse des servos (degrés par pas)
float servoSpeed = 0.8;  //
// Configuration inversion servos (true = inverse)
bool servoInverted[5] ={
  false,
  false,
  true,
  false,
  true,
};
ArmPosition currentPos = {0, 90, 90, 90, 90, 90};
ArmPosition restPos    = {0, 0, 180, 90, 90, 45};
bool emergencyStop = false;
bool systemReady   = false;
// SETUP
void setup() {
  Serial.begin(115200);
  Serial.println(F("=== ARDUINO NANO - NIVEAU BAS v2 ==="));
  Serial.println(F("Initialisation..."));
```

```
  //  Pins base mobile
  pinMode(STBY,       OUTPUT); digitalWrite(STBY, HIGH);
  pinMode(PWM_LEFT,   OUTPUT);
  pinMode(DIR1_LEFT,  OUTPUT);
  pinMode(DIR2_LEFT,  OUTPUT);
  pinMode(PWM_RIGHT,  OUTPUT);
  pinMode(DIR1_RIGHT, OUTPUT);
  pinMode(DIR2_RIGHT, OUTPUT);
  //  HC-SR04
  pinMode(TRIG_PIN_FRONT, OUTPUT);
  pinMode(ECHO_PIN_FRONT, INPUT);
  pinMode(TRIG_PIN_BACK,  OUTPUT);
  pinMode(ECHO_PIN_BACK,  INPUT);
  //  I2C + PCA9685
  Wire.begin();
  pwm.begin();
  pwm.setPWMFreq(SERVO_FREQ);
  delay(10);
  Serial.println(F("[OK] PCA9685 initialisé"));
  // Stepper
  stepper.setSpeed(10);
  Serial.println(F("[OK] Stepper configuré"));
  // MPU6050
  mpu.initialize();
  if (mpu.testConnection()) {
    Serial.println(F("[OK] MPU6050 connecté"));
  } else {
    Serial.println(F("[WARN] MPU6050 non détecté"));
  }
  // - osition repos
  Serial.println(F("Position de repos..."));
  setArmPosition(restPos);
  delay(1000);
  systemReady = true;
  Serial.println(F("[READY] Système prêt - En attente commandes"));
  Serial.println(F("========================================\n"));
}
// LOOP PRINCIPAL
void loop() {
  if (emergencyStop) {
    stopAllMotors();
    delay(100);
    return;
  }
  if (Serial.available() > 0) {
```

```
    String cmd = Serial.readStringUntil('\n');
    cmd.trim();
    parseCommand(cmd);
  }
  delay(20);
}
// CONTRÔLE BASE MOBILE - DIFFÉRENTIEL
//  driveMotorSide(pin_pwm, pin_dir1, pin_dir2, speed), speed > 0  → avant,
speed < 0  → arrière, speed = 0  → stop (frein actif)
void driveMotorSide(int pwmPin, int dir1, int dir2, int speed) {
  if (speed > 0) {
    digitalWrite(dir1, HIGH);
    digitalWrite(dir2, LOW);
  } else if (speed < 0) {
    digitalWrite(dir1, LOW);
    digitalWrite(dir2, HIGH);
    speed = -speed;
  } else {
    // Frein actif
    digitalWrite(dir1, LOW);
    digitalWrite(dir2, LOW);
  }
  analogWrite(pwmPin, constrain(speed, 0, 255));
}
// Commande différentielle principale : speedL et speedR ∈ [-255, 255]
void driveBase(int speedL, int speedR) {
  driveMotorSide(PWM_LEFT,  DIR1_LEFT,  DIR2_LEFT,  speedL);
  driveMotorSide(PWM_RIGHT, DIR1_RIGHT, DIR2_RIGHT, speedR);
}
void moveForward(int speed)  { driveBase( speed,  speed); }
void moveBackward(int speed) { driveBase(-speed, -speed); }
void turnLeft(int speed)     { driveBase(-speed,  speed); }
void turnRight(int speed)    { driveBase( speed, -speed); }
void arcLeft(int speedInner, int speedOuter)  { driveBase(speedInner,
speedOuter); }
void arcRight(int speedInner, int speedOuter) { driveBase(speedOuter,
speedInner); }
void stopAllMotors() { driveBase(0, 0); }
// PARSING COMMANDES
void parseCommand(String cmd) {
  if (cmd.length() == 0) return;
  if (cmd.startsWith("<") && cmd.endsWith(">")) {
    cmd = cmd.substring(1, cmd.length() - 1);
    int sepIdx = cmd.indexOf(':');
```

```
    if (sepIdx == -1) { Serial.println(F("[ERROR] Format invalide"));
return; }
    String command = cmd.substring(0, sepIdx);
    String params  = cmd.substring(sepIdx + 1);
    if (command == "MOVE") {
      int comma  = params.indexOf(',');
      int speedL = params.substring(0, comma).toInt();
      int speedR = params.substring(comma + 1).toInt();
      driveBase(speedL, speedR);
      Serial.println(F("[ACK] MOVE"));
    }
    else if (command == "FWD")  {
moveForward(params.toInt());  Serial.println(F("[ACK] FWD")); }
    else if (command == "BWD")  { moveBackward(params.toInt());
Serial.println(F("[ACK] BWD")); }
    else if (command == "LEFT") {
turnLeft(params.toInt());     Serial.println(F("[ACK] LEFT")); }
    else if (command == "RIGHT"){
turnRight(params.toInt());    Serial.println(F("[ACK] RIGHT")); }
    else if (command == "ARC") {
      int c1 = params.indexOf(',');
      int c2 = params.lastIndexOf(',');
      int si = params.substring(0, c1).toInt();
      int so = params.substring(c1 + 1, c2).toInt();
      String dir = params.substring(c2 + 1);
      if (dir == "L") arcLeft(si, so); else arcRight(si, so);
      Serial.println(F("[ACK] ARC"));
    }
    else if (command == "STOP") { stopAllMotors(); Serial.println(F("[ACK]
STOP")); }
    else if (command == "IK") {
      float x, y, z, theta_deg, theta5_deg;
      int idx = 0, lastIdx = 0;
      idx = params.indexOf(',', lastIdx); x = params.substring(lastIdx,
idx).toFloat(); lastIdx = idx + 1;
      idx = params.indexOf(',', lastIdx); y = params.substring(lastIdx,
idx).toFloat(); lastIdx = idx + 1;
      idx = params.indexOf(',', lastIdx); z = params.substring(lastIdx,
idx).toFloat(); lastIdx = idx + 1;
      idx = params.indexOf(',', lastIdx); theta_deg  =
params.substring(lastIdx, idx).toFloat(); lastIdx = idx + 1;
      theta5_deg = params.substring(lastIdx).toFloat();
      ArmPosition target;
      if (inverseKinematics(x, y, z, theta_deg, theta5_deg, target)) {
        moveArmSmooth(currentPos, target, 2000);
```

```
        Serial.println(F("[ACK] IK OK"));
      } else {
        Serial.println(F("[ERROR] IK failed"));
      }
    }
    else if (command == "ARM") {
      ArmPosition target;
      int idx = 0, lastIdx = 0;
      idx = params.indexOf(',', lastIdx); target.theta1 =
params.substring(lastIdx, idx).toFloat(); lastIdx = idx + 1;
      idx = params.indexOf(',', lastIdx); target.theta2 =
params.substring(lastIdx, idx).toFloat(); lastIdx = idx + 1;
      idx = params.indexOf(',', lastIdx); target.theta3 =
params.substring(lastIdx, idx).toFloat(); lastIdx = idx + 1;
      idx = params.indexOf(',', lastIdx); target.theta4 =
params.substring(lastIdx, idx).toFloat(); lastIdx = idx + 1;
      idx = params.indexOf(',', lastIdx); target.theta5 =
params.substring(lastIdx, idx).toFloat(); lastIdx = idx + 1;
      target.pince = params.substring(lastIdx).toFloat();
      if (checkJointLimits(target)) { setArmPosition(target);
Serial.println(F("[ACK] ARM")); }
      else { Serial.println(F("[ERROR] Limites dépassées")); }
    }
    else if (command == "GRIP") { controlGripper(params.toInt());
Serial.println(F("[ACK] GRIP")); }
    else if (command == "ROT")  { rotateArmBase(params.toFloat());
Serial.println(F("[ACK] ROT")); }
    else if (command == "DIST") {
      float d = readUltrasonic(TRIG_PIN_FRONT, ECHO_PIN_FRONT);
      Serial.print(F("[DATA] DIST_FRONT:")); Serial.println(d);
    }
    else if (command == "DIST_BACK") {
      float d = readUltrasonic(TRIG_PIN_BACK, ECHO_PIN_BACK);
      Serial.print(F("[DATA] DIST_BACK:")); Serial.println(d);
    }
    else if (command == "DIST_ALL") {
      float df = readUltrasonic(TRIG_PIN_FRONT, ECHO_PIN_FRONT);
      float db = readUltrasonic(TRIG_PIN_BACK,  ECHO_PIN_BACK);
      Serial.print(F("[DATA] DIST_ALL:")); Serial.print(df);
Serial.print(","); Serial.println(db);
    }
    else if (command == "IMU") {
      int16_t ax, ay, az, gx, gy, gz;
      mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);
      Serial.print(F("[DATA] IMU:"));
```

```
      Serial.print(ax); Serial.print(","); Serial.print(ay);
Serial.print(","); Serial.print(az); Serial.print(",");
      Serial.print(gx); Serial.print(","); Serial.print(gy);
Serial.print(","); Serial.println(gz);
    }
    else if (command == "REST") {
      moveArmSmooth(currentPos, restPos, 2000);
      stopAllMotors();
      Serial.println(F("[ACK] REST"));
    }
    else if (command == "ESTOP") { emergencyStop = true; stopAllMotors();
Serial.println(F("[ALERT] EMERGENCY STOP")); }
    else if (command == "RESET") { emergencyStop = false;
Serial.println(F("[ACK] RESET")); }
    else { Serial.print(F("[ERROR] Commande inconnue: "));
Serial.println(command); }
  }
}
// CINÉMATIQUE INVERSE
bool inverseKinematics(float Px, float Py, float Pz, float theta_pitch_deg,
float theta5_deg, ArmPosition &result) {
  float theta_pitch = radians(theta_pitch_deg);
  float theta1 = atan2(Py, Px);
  float R   = DH_D5 * cos(theta_pitch);
  float Pwx = Px - R * cos(theta1);
  float Pwy = Py - R * sin(theta1);
  float Pwz = Pz + DH_D5 * sin(theta_pitch);
  float Rw = sqrt(Pwx * Pwx + Pwy * Pwy);
  float N  = sqrt((Pwz - DH_D1) * (Pwz - DH_D1) + Rw * Rw);
  if (N > (DH_A2 + DH_A3) || N < abs(DH_A2 - DH_A3)) {
    Serial.println(F("[ERROR] Position hors atteinte"));
    return false;
  }
  float cos_mu = (N * N + DH_A2 * DH_A2 - DH_A3 * DH_A3) / (2 * DH_A2 * N);
  if (abs(cos_mu) > 1.0) return false;
  float mu = acos(cos_mu);
  float lambda = atan2(Pwz - DH_D1, Rw);
  float theta2 = lambda + mu;  // Coude haut
  float cos_theta3 = (N * N - DH_A2 * DH_A2 - DH_A3 * DH_A3) / (2 * DH_A2 *
DH_A3);
  if (abs(cos_theta3) > 1.0) return false;
  float theta3 = acos(cos_theta3);  // Coude haut
  float theta234 = radians(90) - theta_pitch;
  float theta4   = theta234 - theta2 - theta3;
  result.theta1 = degrees(theta1);
```

```
  result.theta2 = degrees(theta2);
  result.theta3 = degrees(theta3);
  result.theta4 = degrees(theta4);
  result.theta5 = theta5_deg;
  result.pince  = currentPos.pince;
  return checkJointLimits(result);
}
// VÉRIFICATION LIMITES ARTICULAIRES
bool checkJointLimits(ArmPosition pos) {
  float angles[5] = {pos.theta1, pos.theta2, pos.theta3, pos.theta4,
pos.theta5};
  for (int i = 0; i < 5; i++) {
    if (angles[i] < THETA_MIN[i] || angles[i] > THETA_MAX[i]) {
      Serial.print(F("[ERROR] θ")); Serial.print(i + 1);
      Serial.print(F(" = ")); Serial.print(angles[i]);
      Serial.print(F("° hors limites [")); Serial.print(THETA_MIN[i]);
      Serial.print(F("°, ")); Serial.print(THETA_MAX[i]);
Serial.println(F("°]"));
      return false;
    }
  }
  return true;
}
// CONTRÔLE SERVOS PCA9685
void setServoAngle(uint8_t channel, float targetAngle) {
  // Contraindre l'angle cible
  targetAngle = constrain(targetAngle, 0, 180);
  if (servoInverted[channel]){
    targetAngle = 180-targetAngle;
  }
  // Angle actuel du servo
  float currentAngle = currentAngles[channel];
  // Si déjà à l'angle cible, ne rien faire
  if (abs(currentAngle - targetAngle) < 0.5) {
    return;
  }
  // Interpolation linéaire (mouvement progressif)
  while (abs(currentAngle - targetAngle) > 0.5) {
    // Calculer le pas (direction + vitesse)
    float step;
    if (targetAngle > currentAngle) {
      step = min(servoSpeed, targetAngle - currentAngle);
    } else {
      step = -min(servoSpeed, currentAngle - targetAngle);
    }
```

```
    currentAngle += step;
    int pulse = map((int)currentAngle, 0, 180, SERVOMIN, SERVOMAX);
    pwm.setPWM(channel, 0, pulse);
    delay(20);  //
  }
  // Sauvegarder la position finale
  currentAngles[channel] = targetAngle;
  int finalPulse = map((int)targetAngle, 0, 180, SERVOMIN, SERVOMAX);
  pwm.setPWM(channel, 0, finalPulse);
}
void setArmPosition(ArmPosition pos) {
  setServoAngle(SERVO_EPAULE,      pos.theta2);
  setServoAngle(SERVO_COUDE,       pos.theta3);
  setServoAngle(SERVO_POIGNET_INC, pos.theta4);
  setServoAngle(SERVO_POIGNET_ROT, pos.theta5);
  setServoAngle(SERVO_PINCE,       pos.pince);
  currentPos = pos;
}
void moveArmSmooth(ArmPosition from, ArmPosition to, int duration) {
  const int STEPS = 50;
  int delayTime = duration / STEPS;
  for (int i = 0; i <= STEPS; i++) {
    ArmPosition mid;
    mid.theta1 = map(i*100, 0, STEPS*100, from.theta1*100, to.theta1*100) /
100.0;
    mid.theta2 = map(i*100, 0, STEPS*100, from.theta2*100, to.theta2*100) /
100.0;
    mid.theta3 = map(i*100, 0, STEPS*100, from.theta3*100, to.theta3*100) /
100.0;
    mid.theta4 = map(i*100, 0, STEPS*100, from.theta4*100, to.theta4*100) /
100.0;
    mid.theta5 = map(i*100, 0, STEPS*100, from.theta5*100, to.theta5*100) /
100.0;
    mid.pince  = map(i*100, 0, STEPS*100, from.pince*100,  to.pince*100)  /
100.0;
    setArmPosition(mid);
    delay(delayTime);
  }
}
// STEPPER + PINCE
void rotateArmBase(float angle) {
  int steps = (int)((angle / 360.0) * STEPS_PER_REV);
  stepper.step(steps);
  currentPos.theta1 += angle;
}
```

```
void controlGripper(int state) {
  if (state == 1) { setServoAngle(SERVO_PINCE, 90);  currentPos.pince =
90;  }
  else            { setServoAngle(SERVO_PINCE, 180); currentPos.pince =
180; }
}
// HC-SR04
float readUltrasonic(int trigPin, int echoPin) {
  digitalWrite(trigPin, LOW);  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH); delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  long duration = pulseIn(echoPin, HIGH, 30000);
  if (duration == 0) return -1;
  return (duration * 0.034) / 2;
}
```