

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



Université A. MIRA-BEJAIA

Faculté de Technologie

Département de Génie Electrique

# Mémoire de Fin d'études

En vue de l'obtention du diplôme de Master en Automatique

Spécialité : Automatique et Informatique Industrielle

*Thème*

---

**Etude et réalisation d'un bras de robot à base de carte Arduino**

---

**Soutenu le : 25/06/19**

**Présenté par :**

**Mr : SAHIR Koceila**  
**Mr : BOUATMANE Hakim**

**Encadré par :**

**Mr : MENDIL Boubekeur**

**Membres du jury :**

**Mr : MOKRANI Karim**  
**Mme : BELLAHSENE Nora**

**Année universitaire 2018/2019**

# REMERCIEMENTS

D'abord nous remercions Dieu notre créateur de nous avoir donné la santé, la force, la volonté et le courage de pouvoir réussir et d'accomplir ce modeste mémoire.

Nous tenons à exprimer nos sincères remerciements et notre reconnaissance à Mr B. MENDIL pour son encadrement et sa précieuse disponibilité qu'il a bien voulu nous accorder. Son expérience académique, sa rigueur intellectuelle et ses conseils dans le but de réaliser ce travail, nous ont permis de nous développer et de réussir ce projet.

Nous remercions très respectueusement Mr MOKRANI Karim et Mme BELLAHSENE Nora, de nous avoir fait l'honneur de juger notre modeste travail.

On ne saura clôturer ces remerciements sans exprimer toute notre affection pour les enseignants de l'Université de Bejaia, qui ont bien voulu partager leurs savoir durant nos études.

Enfin, nous remercions tous ceux qui ont contribué de prêt ou de loin à la réalisation de ce projet.

# DEDICACES

Avec l'expression de ma reconnaissance, je dédie ce travail à ceux à qui, quels que soient les termes embrassés, je n'arriverais jamais à exprimer mon amour sincère :

- À l'homme, ma précieuse offre du dieu, qui doit ma vie, ma réussite et tout mon respect : mon cher père Rachid.
- À la femme qui a souffert sans me laisser souffrir, qui n'a jamais dit non à mes exigences et qui n'a épargné aucun effort pour me rendre heureux : mon adorable mère Malika.
- À mes très chers frères leurs femmes, sœurs et leurs enfants, qui n'ont pas cessé de me conseiller, encourager et soutenir tout au long de mes études, que Dieu les protège et leur offrent la chance et le bonheur.
- À tous les cousin(e)s, les voisin(e)s et les ami(e)s que j'ai connu jusqu'à maintenant. Merci pour leurs encouragements.

Sans oublier mon binôme et toutes personnes qui ont contribué de près ou de loin par leur soutien moral tout au long de ce projet

*Koceila*

# DEDICACES

Je dédie ce modeste mémoire :

A mes adorables parents Mouloud et Hayat qui m'ont toujours encouragé durant toute ma vie.

A mes frères et sœurs à qui je souhaite beaucoup de bonheur et de réussite.

A tous mes ami(e)s et à toutes les personnes qui ont une place spéciale dans mon cœur et ma vie.

A tous les étudiants Master 2 Automatique.

*Hakim*

# SOMMAIRE

SOMMAIRE .....	i
LISTE DES ABREVIATIONS .....	v
LISTE DES SYMBOLES .....	viii
LISTE DES FIGURES .....	x
LISTE DES TABLEAUX.....	xiii
INTRODUCTION GENERALE.....	1
<b>1. CHAPITRE 1 : GENERALITES</b>	
1.1.Introduction .....	2
1.2.Définitions .....	2
1.2.1. Définition d'un robot.....	2
1.2.2. Définition de la robotique .....	2
1.3.L'histoire de la robotique .....	2
1.4.Types de robots .....	5
1.4.1. Robots mobiles.....	5
1.4.2. Robots manipulateurs.....	6
1.5.Robots industriels.....	6
1.6.Bras manipulateurs.....	6
1.7.Constituant d'un robot.....	6
1.7.1. Les actionneurs.....	7
1.7.2. Le système mécanique articulé (S.M.A) .....	7
1.7.3. Les articulations .....	8
1.7.3.1.Les articulations prismatiques.....	8
1.7.3.2.Les articulations rotoïdes .....	8
1.7.4. Organe terminal.....	8
1.7.5. Les capteurs.....	9
1.7.6. La commande .....	9
1.8.Classification des robots.....	9
1.8.1. Point de vue fonctionnel.....	9
1.8.1.1.Manipulateur à commande manuelle .....	9
1.8.1.2.Manipulateur automatique à cycles prééglés .....	10
1.8.1.3.Robots programmables.....	10
1.8.1.4.Robots intelligents.....	11

1.8.2. Classification géométrique.....	12
1.8.2.1.Structure cartésienne (PPP).....	12
1.8.2.2.La structure cylindrique (RPP) ou (PRP).....	12
1.8.2.3.La structure sphérique ou polaire à axe de rotation orthogonale (RPP) ....	13
1.8.2.4.La structure dite SCARA .....	13
1.8.2.5.La structure anthropomorphe (RRR) .....	14
1.9.Domains d'application de la robotique.....	14
1.9.1. Industriel .....	14
1.9.2. Médical.....	15
1.9.3. Spatial.....	15
1.10. Conclusion.....	15
<b>2. CHAPITRE 2 : DESCRIPTION ET MODELISATION DU BRAS MANIPULATEUR</b>	
2.1.Introduction .....	16
2.2.Description mécanique du bras manipulateur .....	16
2.2.1. La structure mécanique du bras.....	16
2.2.1.1.La base .....	16
2.2.1.2.Le bras .....	17
2.2.1.3.La pince .....	18
2.2.1.4.Le robot .....	18
2.2.2. Description des parties constitutives du bras manipulateur .....	19
2.2.2.1.Les actionneurs du bras manipulateur.....	19
2.2.2.2.Les paramètres techniques des servomoteurs .....	20
2.2.2.3.La commande des servomoteurs .....	20
2.2.2.4.Transmission entre articulations : les engrenages .....	21
2.2.2.5.Les capteurs.....	21
2.2.2.6.Les paramètres techniques du récepteur bluetooth .....	22
2.3.Modélisation.....	22
2.3.1. Repères et référentiels .....	22
2.3.1.1.Coordonnées cartésiennes .....	23
2.3.1.2.Coordonnées cylindriques .....	23
2.3.1.3.Coordonnées sphériques.....	24
2.3.2. Position et orientation d'un solide.....	24
2.3.3. Modèle géométrique direct .....	25
2.3.3.1.Paramètre de Denavit-Hatenberg (DH).....	26
2.3.3.2.Modélisation géométrique direct de notre réalisation .....	27
2.3.3.3.Identification des paramètres de DH de notre bras manipulateur .....	27

2.3.4. Le modèle géométrique inverse (MGI).....	28
2.3.4.1. Le modèle géométrique inverse de notre réalisation.....	29
2.3.5. Modélisation cinématique direct (MCD) .....	30
2.3.5.1.Modèle cinématique direct de notre réalisation .....	30
2.3.6. Modèle cinématique inverse (MCI) .....	32
2.3.6.1.Modèle cinématique inverse de notre réalisation.....	32
2.3.7. Modèle dynamique .....	33
2.3.7.1.Modélisation dynamique de notre réalisation .....	34
2.3.8. Modélisation des actionneurs électriques .....	37
2.4.Conclusion.....	38
<b>3. CHAPITRE 3 : COMMANDE ET SIMULATION DU BRAS MANIPULATEUR</b>	
3.1.Introduction .....	39
3.2. Présentation de la carte Arduino .....	39
3.2.1. La carte Arduino UNO.....	39
3.2.2. Les caractéristiques de la carte Arduino UNO.....	40
3.2.3. L’environnement de programmation Arduino (IDE).....	40
3.3.Réalisation de la carte de commande .....	41
3.4.Types de commande du bras manipulateur .....	44
3.4.1. La commande manuelle .....	44
3.4.2. La commande sans fil.....	45
3.4.2.1.Le concept d’App Inventor .....	45
3.4.2.2.Réalisation de l’application Android .....	46
3.4.2.3.L’interface finale de notre application .....	49
3.4.2.4.Communication de l’application avec l’Arduino .....	49
3.4.3. Commande par angles préprogrammés .....	52
3.5.Implémentation et simulation du projet sous ISIS .....	54
3.5.1. Présentation du logiciel ISIS (Intelligent Schematiic Input System).....	55
3.5.2. Les différentes phases à suivre pour l’élaboration d’un projet .....	55
3.5.3. Implémentation et simulation.....	56
3.6.Tests et résultats pratiques.....	57
3.6.1. Commande manuelle.....	57
3.6.2. Commande par angles préprogrammés .....	60
3.7.Simulation du bras manipulateur avec la commande PID .....	61
3.7.1. Le régulateur PID .....	61
3.7.1.1.La composante proportionnelle.....	62
3.7.1.2.La composante intégrale .....	62

3.7.1.3.La composante dérivée..... 63

3.7.2. Calcul des paramètres du régulateur PID..... 63

3.7.2.1.Simulation avec SimMechanics (MATLAB SIMULINK)..... 65

3.7.2.2.Résultats ..... 66

3.8.Conclusion..... 66

CONCLUSION GENERALE ..... 67

BIBLIOGRAPHIE ..... 68

ANNEXES ..... ..

RESUME..... ..

# LISTE DES ABREVIATIONS

## #123

.dxf = Drawing eXchange Format

3D = Trois dimensions

## A

A.F.R.I. = Association Française de Robotique Industrielle

ATAN2 : En trigonométrie, atan2 à deux arguments est une variation d'ARC tangente

## C

CPU = Central Processing Unit

## D

Ddl = Degré de liberté

DH = Denavit-Hartenberg

## E

E/S = Entrée / Sortie

EEPROM = Electrically Erasable Programmable Read-Only Memory

EPFL = Ecole Polytechnique Fédérale de Lausanne

## G

GM = General Motors

## I

IDE = Integrated Development Environment

ISIS = Intelligent Schematic Input System

ISO = International Organizations for Standardization

**J**

JC = Jésus Christ

JSC = Johnson Space Center

**L**

LED = Light-Emitting Diode

**M**

MCC = Moteur à Courant Continu

MCD = Modèle Cinématique Direct

MCI = Modèle Cinématique Inverse

MDD = Modèle Dynamique Direct

MGD = Modèle Géométrique Direct

MGI = Modèle Géométrique Inverse

**P**

P = Prismatique

PID = Proportionnel, Intégrale, Dérivé

PPP = Trois articulations prismatiques consécutives

PWM = Pulse With Modulation

**R**

R = Rotoïde

RAM = Random Access Memory

ROM = Read-Only Memory

RF = Radio Fréquence

RPP (ou PRP) = Une articulation rotoïde et deux articulations prismatiques

RRP = Deux articulations rotoïdes et une articulation prismatique

## **S**

S.M.A. = Système Mécanique Articulé

SCARA = Selective Compliance Articulated Robot for Assemblage

SRAM = Static Random-Access Memory

## **U**

UART = Universal Asynchronous Receiver Transmitter

USB = Universal Serial Bus

# LISTE DES SYMBOLES

SYMBOLES	SIGNIFICATIONS
$a_i$	Distance entre $z_{i-1}$ et $z_i$ , le long de $x_i$
$\alpha_i$	Angle entre $z_{i-1}$ et $z_i$ , autour de $x_i$
$C(\theta, \dot{\theta})$	Matrice appelée courbe de Coriolis
$C(q)$	Vecteur des forces de charges gravitationnelles
$D(q)$	Matrice d'inertie liée à l'accélération
$D_c(t)$	Composante dérivée
$d_i$	Déplacement de l'articulation
$E_c(t)$	Erreur
$f_e$	Vecteur de l'effort extérieur
$G(\theta)$	Vecteur des forces de gravité
$H(q, \dot{q})$	Vecteur des forces centrifuges
$h_i$	Hauteur du centre de masse
$I_c(t)$	Composante intégrale
$I_j$	Courant de l'induit du moteur j
$J(q)$	Matrice jacobienne
$J_j$	Inertie propre du moteur j
$k_d$	Constante dérivée du système
$k_i$	Constante intégrale du système
$k_p$	Constante proportionnelle du système
$K$	Energie cinétique
$K_{a_j}$	Constante proportionnelle
$K_{c_j}$	Coefficient du couple moteur j
$L$	Lagrangien

$M(q)$	Matrice d'inertie
$m_i$	Paramètres de système
$N_j$	Rapport de réduction
$\theta_i$	Angle entre $x_{i-1}$ et $x_i$ , autour de $z_{i-1}$ .
$P$	Energie potentielle
$P_c(t)$	Composante proportionnelle
$q$	Vecteur des variables articulaires
$Q$	Force généralisée
$T_n$	Matrice de transformation
$\Gamma$	Vecteur des couples des actionneurs
$u_j$	Tension d'entrée
$V_m$	Vitesse linéaires
$\omega_m$	Vitesse angulaire
$\omega_n$	Pulsation naturelle
$X$	Coordonnées opérationnelles
$\zeta$	Facteur d'amortissement

## CHAPITRE 1 : Généralités

<b>Fig.1.1.</b> De gauche à droite : (a) Horloge a eau de Clépsydre, (b) Théâtre mobile de Héron D'alexandrie, (c) Canard digérateur de Jacques de Vaucanson .....	3
<b>Fig.1.2.</b> Unimate, premier robot industriel.....	3
<b>Fig.1.3.</b> Opportunity.....	4
<b>Fig.1.4.</b> Robonaut.....	4
<b>Fig.1.5.</b> Graphique représentant l'estimation des expéditions annuelles mondiales de robots industriels par régions .....	5
<b>Fig.1.6.</b> Robot mobile .....	5
<b>Fig.1.7.</b> Bras Manipulateur .....	6
<b>Fig.1.8.</b> Constituant d'un robot .....	7
<b>Fig.1.9.</b> Représentation d'une articulation prismatique .....	8
<b>Fig.1.10.</b> Représentation d'une articulation rotoïde.....	8
<b>Fig.1.11.</b> Manipulateur à commande manuelle.....	10
<b>Fig.1.12.</b> Manipulateur à cycle préréglé .....	10
<b>Fig.1.13.</b> Robot programmable .....	11
<b>Fig.1.14.</b> Robot intelligent .....	11
<b>Fig.1.15.</b> Structure PPP .....	12
<b>Fig.1.16.</b> Structure RPP.....	13
<b>Fig.1.17.</b> Structure RRP .....	13
<b>Fig.1.18.</b> Structure SCARA .....	14
<b>Fig.1.19.</b> Structure RRR.....	14

## CHAPITRE 2 : Description et Modélisation du Bras Manipulateur

<b>Fig.2.1.</b> Les pièces constituant le robot.....	16
<b>Fig.2.2.</b> Base du robot conçu sur 3D Builder.....	17
<b>Fig.2.3.</b> Bras du robot conçu sur 3D Builder .....	17
<b>Fig.2.4.</b> Pince du robot conçu sur 3D Builder.....	18
<b>Fig.2.5.</b> Bras robot entier conçu sur 3D Builder .....	18
<b>Fig.2.6.</b> Schéma d'un actionneur .....	19
<b>Fig.2.7.</b> De gauche à droite : (a) servomoteur du type MG90S, (b) servomoteur du Type SG90 .....	20
<b>Fig.2.8.</b> Exemple de signaux en position pour un servomoteur .....	21
<b>Fig.2.9.</b> Schéma d'un engrenage .....	21

<b>Fig.2.10.</b> De gauche à droite : (a) récepteur bluetooth, (b) schéma d'un servomoteur.....	22
<b>Fig.2.11.</b> Le système de coordonnées cartésiennes.....	23
<b>Fig.2.12.</b> Le système de coordonnées cylindriques .....	23
<b>Fig.2.13.</b> Le système de coordonnées sphériques .....	24
<b>Fig.2.14.</b> Position d'un solide dans l'espace et son repère associé.....	25
<b>Fig.2.15.</b> Schématisation du lien entre le corps $i-1$ et $i$ .....	26
<b>Fig.2.16.</b> Placement des repères selon le modèle DH.....	27
<b>Fig.2.17.</b> Schéma fonctionnel d'un actionneur .....	37

### CHAPITRE 3 : Commande et Simulation du Bras Manipulateur

<b>Fig.3.1.</b> La carte Arduino UNO .....	40
<b>Fig.3.2.</b> Environnement de développement Arduino .....	41
<b>Fig.3.3.</b> Les composants constituant la carte de commande manuelle (non soudés).....	42
<b>Fig.3.4.</b> Montage des pins et sources d'alimentation sur la carte de commande manuelle .....	42
<b>Fig.3.5.</b> Test de continuité à l'aide d'un multimètre de la carte de commande manuelle .....	43
<b>Fig.3.6.</b> Carte de commande manuelle finalisée .....	43
<b>Fig.3.7.</b> Logigramme des phases explicatives de la commande manuelle.....	45
<b>Fig.3.8.</b> App Inventor LOGO.....	46
<b>Fig.3.9.</b> Partie design de notre application sous AppInventor .....	46
<b>Fig.3.10.</b> Définition des éléments de l'objet Bluetooth .....	47
<b>Fig.3.11.</b> Blocs de contrôle des curseurs.....	48
<b>Fig.3.12.</b> Blocs des deux boutons .....	48
<b>Fig.3.13.</b> Interface finale de l'application .....	49
<b>Fig.3.14.</b> Schéma synoptique simplifié du système .....	49
<b>Fig.3.15.</b> Schéma de la communication avec le module Bluetooth HC-05 .....	50
<b>Fig.3.16.</b> Logigramme résumant la commande sans fil .....	52
<b>Fig.3.17.</b> Logigramme résumant la commande à angles préprogrammés .....	54
<b>Fig.3.18.</b> Logiciel Proteus ISIS .....	55
<b>Fig.3.19.</b> Logigramme des phases explicatives de l'utilisation du logiciel ISIS .....	56
<b>Fig.3.20.</b> Simulation du bras manipulateur sous ISIS .....	57
<b>Fig.3.21.</b> Variation de l'entrée analogique de la première articulation en fonction du temps .	57
<b>Fig.3.22.</b> Variation de l'angle $\theta_1$ en fonction du temps .....	58
<b>Fig.3.23.</b> Variation de l'entrée analogique de la deuxième articulation en fonction du temps	58
<b>Fig.3.24.</b> Variation de l'angle $\theta_2$ en fonction du temps .....	58
<b>Fig.3.25.</b> Variation de l'entrée analogique de la troisième articulation en fonction du temps	59
<b>Fig.3.26.</b> Variation de l'angle $\theta_3$ en fonction du temps .....	59

<b>Fig.3.27.</b> Test du bras manipulateur pour déplacement d'objet.....	61
<b>Fig.3.28.</b> Schéma du régulateur PID .....	62
<b>Fig.3.29.</b> Modèle du robot dans SimMechanics.....	65
<b>Fig.3.30.</b> Les composants du sous système pour la première articulation .....	65
<b>Fig.3.31.</b> Réponses des trois articulations à différentes amplitudes pour un signal sinusoïdale .....	66

# LISTE DES TABLEAUX

CHAPITRE 1 : Généralités

CHAPITRE 2 : Description et Modélisation du Bras Manipulateur

<b>Tab.2.1.</b> Tableau des caractéristiques des servomoteurs du bras .....	20
<b>Tab.2.2.</b> Tableau des caractéristiques du récepteur Bluetooth du bras .....	22
<b>Tab.2.3.</b> Types d'articulations et d'angles de rotations possibles du bras réalisé .....	27
<b>Tab.2.4.</b> Tableau des paramètres de DENAVIT- HATENBERG de notre réalisation .....	28

CHAPITRE 3 : Commande et Simulation du Bras Manipulateur

<b>Tab.3.1.</b> Caractéristiques de l'Arduino UNO .....	40
---	----



# **INTRODUCTION GÉNÉRALE**

La plupart des chercheurs ont mis l'accent sur le contrôle des robots et des applications de la robotique. La conception des robots est généralement concentrée dans des domaines tels que l'éducation, l'industrie ou la médecine. [1]

Commander un robot manipulateur consiste à lui permettre d'atteindre un état désiré ou de réaliser une tâche bien spécifique avec une grande précision dans un environnement réel. La phase de programmation est une phase très importante dans la mesure où elle permet de générer les trajectoires nécessaires pour réaliser ces tâches.

L'objectif principal de notre travail consiste en l'étude théorique, ainsi qu'en la réalisation pratique d'un bras robot manipulateur à trois degrés de liberté, en utilisant des servomoteurs permettant à ce dernier d'avoir des déplacements d'une grande précision. Après la réalisation, on a procédé à la programmation, notamment la configuration du Bluetooth et l'implémentation d'une commande à distance avec smartphone à travers une application Android.

Pour mieux présenter notre travail, le mémoire est organisé en trois chapitres distincts. Le premier chapitre concerne les généralités, où l'on présente la définition de la robotique, son historique et les différents types de robots manipulateurs en précisant leurs domaines d'utilisation.

Le deuxième chapitre est consacré à la description et la modélisation du bras manipulateur étudié. La première partie porte sur la description mécanique du bras et les différents composants utilisés pour sa réalisation. La deuxième traite de la modélisation et de l'élaboration des modèles géométriques, cinématiques, et dynamiques, ainsi que de la modélisation des actionneurs électriques.

Le troisième chapitre expose la commande et la simulation du bras manipulateur. La première partie concerne la présentation de la carte de commande et son fonctionnement, ainsi que les types de commande qu'on peut utiliser avec notre réalisation. La deuxième partie traite de la simulation du bras commandé par un régulateur PID sous MATLAB.

Enfin, le mémoire se termine avec une conclusion résumant les connaissances acquises tout au long de ce projet, ainsi que les perspectives pouvant venir améliorer notre bras manipulateur et compléter notre travail dans le futur.

**CHAPITRE 1 :**

**GÉNÉRALITÉS**

## 1.1. Introduction

De nos jours, l'utilisation des machines dans l'industrie ou la recherche est de plus en plus importante, notamment l'utilisation de la robotique. Celle-ci permet de manipuler, avec beaucoup de précision et donc de manière plus sûre, divers produits ou objets, neutres ou dangereux.

Dans ce chapitre, on donnera un certain nombre de définitions ainsi qu'un bref historique sur la robotique. On introduira les différents types de robots et leurs constituants. Enfin, on évoquera la classification des systèmes robotiques, tout en ouvrant une courte parenthèse sur les domaines d'applications possibles.

## 1.2. Définitions

### 1.2.1. Définition d'un robot

Un robot est un mécanisme automatique capable d'exécuter certaines tâches selon un programme fixe ou modifiable [2].

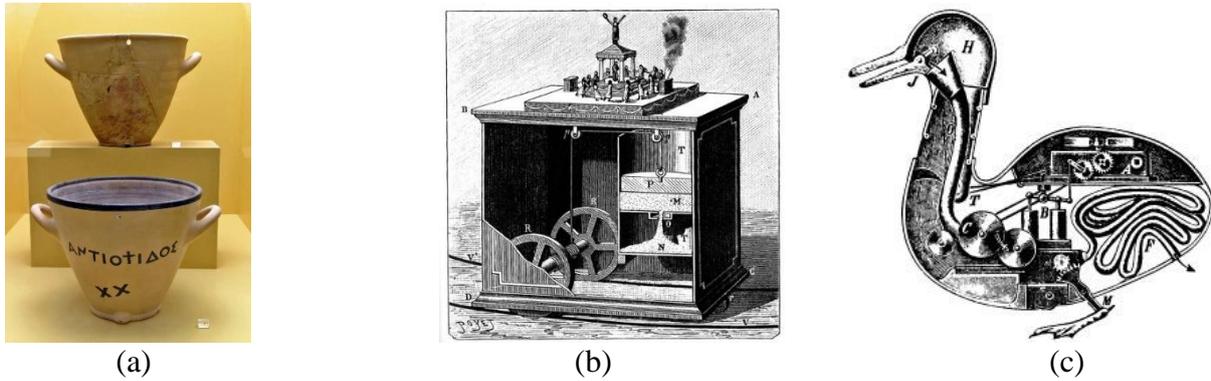
### 1.2.2. Définition de la robotique

Science et technique de la robotisation, de la conception et de la construction des robots [3].

## 1.3. L'histoire de la robotique

Depuis la nuit des temps, l'homme a cherché à se faire remplacer pour des tâches spécifiques pouvant être dangereuses, fastidieuses ou longues. Le concept du robot a été établi par de nombreuses réalisations historiques créatives, tel que : l'horloge à l'eau de Clepsydre introduite par les Babyloniens (1400 avant JC), le théâtre de l'automate du Héron d'Alexandrie (une centaine d'années après JC) et les automates de l'arabe Al-Jazari (1200 après JC) [4]. Un de ces spécimen très évolué fut présenté par Jacques de Vaucanson en 1738 : il représentait un homme jouant d'un instrument de musique à vent. Jacques de Vaucanson créa également un automate représentant un canard mangeant et refoulant sa nourriture après ingestion de cette dernière.

Cependant, l'apparition du robot physique a dû attendre l'arrivée de technologies avancées et plus adaptées au cours du XXe siècle.



**Fig.1.1.** De gauche à droite : (a) Horloge a eau de Clépsydre, (b) Théâtre mobile de Héron D'Alexandrie, (c) Canard digérateur de Jacques de Vaucanson [4].

- 1920 : L'origine du mot robot provient de la langue tchèque dans laquelle son ancêtre "robota" signifie travail forcé. Il a été introduit, en 1920, par l'écrivain tchèque Karel Capek dans la pièce de théâtre « Rossum's Universal Robots » [5].
- En 1954, George Charles Devol invente le premier robot industriel programmable. Il fonde en 1956 la première entreprise de robot, Unimation, avec l'ingénieur Joseph Engelberger. Leur robot, appelé Unimate, était capable de manipuler des objets avec des actionneurs hydrauliques. [5]



**Fig.1.2.** Unimate, premier robot industriel [5]

- En 1969, Victor Scheinman de l'Université de Stanford inventa un bras articulé sur 6 axes. Scheinman vendit son concept à Unimation plus tard [5].
- 1973 : KUKA, une entreprise allemande crée le premier robot industriel avec 6 axes contrôlés électromécaniquement : le Famulus [5].

- 1978 : Unimation propose le PUMA, un robot d'assemblage encore utilisé de nos jours dans plusieurs laboratoires de recherche [5].
- 1981 : Takeo Kanade construit le premier robot muni de moteurs installés directement aux articulations du bras. Cela a permis des robots plus rapides et plus précis qu'auparavant [5].
- 1998 : ABB, Suède, a développé le FlexPicker, le robot de préparation de commandes le plus rapide au monde basé sur le robot delta développé par Reymond Clavel, Institut fédéral de technologie de Lausanne (EPFL) [5].
- 2003 : Première mission d'exploration de la planète Mars impliquant deux robots mobiles, Spirit et Opportunity [5].

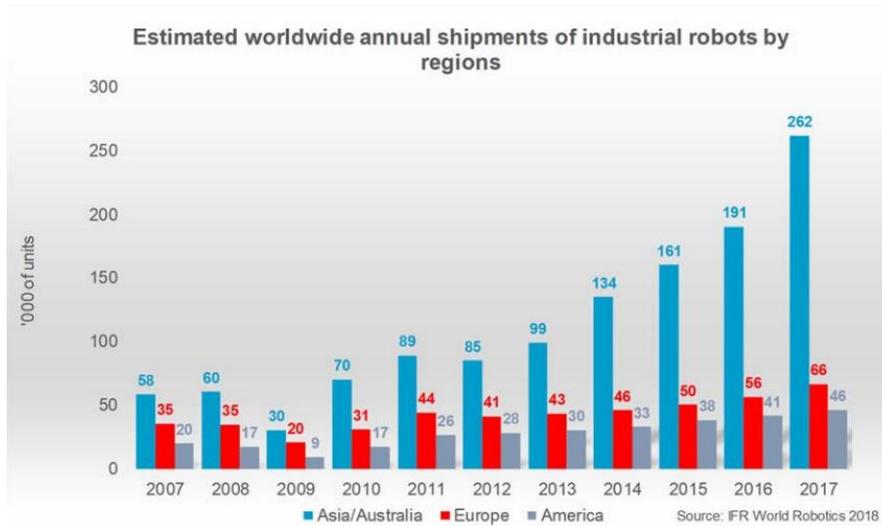


**Fig.1.3.** Opportunity [5]

- 2011 : Robonaut (R2B) premier robot humanoïde envoyé dans l'espace construit et conçu par la NASA au Johnson Space Center (JSC) à Houston (Texas), en collaboration avec General Motors (GM) et Oceaneering [5].



**Fig.1.4.** Robonaut [5]



**Fig.1.5.** Graphique représentant l'estimation des expéditions annuelles mondiales de robots industriels par régions [5].

## 1.4. Types de robots

Il existe deux types de robots : les robots mobiles et les robots manipulateurs

### 1.4.1. Robots mobiles

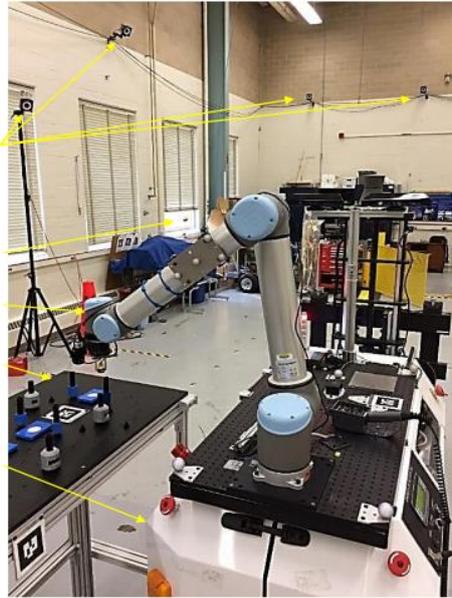
En général, on appelle robots mobiles l'ensemble des robots à base mobile. L'usage veut cependant que l'on montre le plus souvent par ce terme les robots mobiles à roues. Les autres robots mobiles sont en effet le plus souvent désignés par leur type de locomotion, qu'ils soient marcheurs, sous-marins ou aériens.



**Fig.1.6.** Robot mobile [6]

### 1.4.2. Robots manipulateurs

Les robots manipulateurs sont des robots intégrés dans un espace propre destiné au transfert de produits semi-finis entre les lignes de production. Le robot est généralement suspendu pour des questions d'encombrement et par contraintes d'implantation.



**Fig.1.7.** Bras Manipulateur [7]

### 1.5. Robot industriel

La définition diffère selon les pays, on se contentera de la version normalisée par l'organisation internationale de normalisation (ISO) :

« Une machine formée par un mécanisme incluant plusieurs degrés de libertés, ayant souvent l'apparence d'un ou de plusieurs bras se terminant par un poignet capable de tenir des outils, des pièces ou un dispositif d'inspection » [8].

### 1.6. Bras manipulateur

Un bras manipulateur est un bras d'un robot généralement programmable, avec des fonctions similaires à un bras humain. Il peut être autonome ou contrôlé manuellement et peut être utilisé pour effectuer une variété de tâches avec une grande précision. [9]

### 1.7. Constituant d'un robot [10]

On distingue classiquement 4 parties principales dans un robot manipulateur :

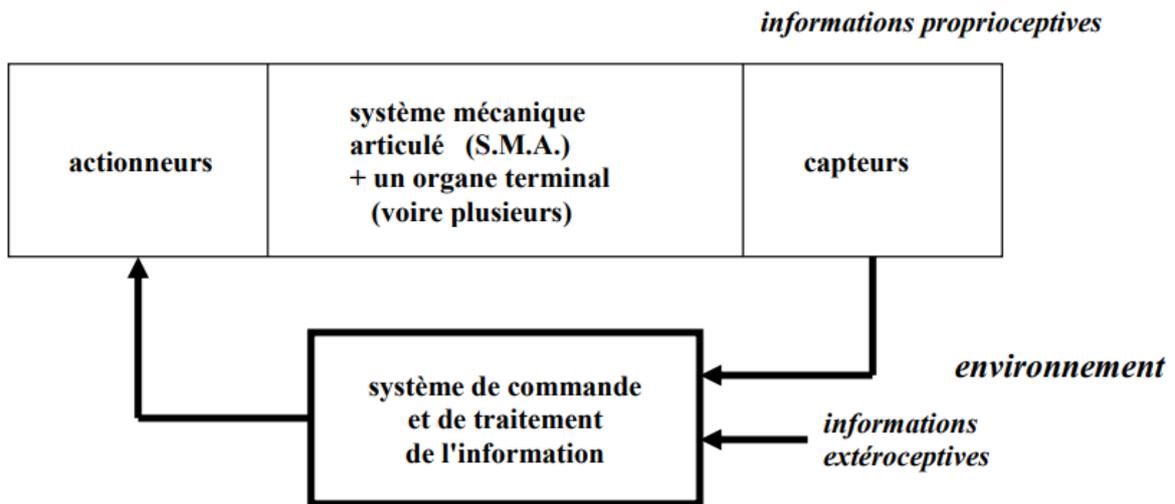


Fig.1.8. Constituant d'un robot [10]

### 1.7.1. Les actionneurs

Les actionneurs sont l'ensemble des moteurs associés à des transmissions (courroies crantées) pour animer le S.M.A. Ils utilisent souvent des moteurs électriques à aimant permanent, à courant continu et à commande par l'induit (la tension n'est continue qu'en moyenne, car en général l'alimentation est un hacheur de tension à fréquence élevée).

Les actionneurs sont le plus souvent hydrauliques pour les robots devant manipuler de très lourdes charges (par exemple, une pelle mécanique). Ils doivent agir en translation (vérin hydraulique) ou en rotation (moteur hydraulique). Il existe également des actionneurs pneumatiques ayant un usage général pour les manipulateurs à cycles.

### 1.7.2. Le système mécanique articulé (S.M.A.)

Il s'agit d'un mécanisme ayant une structure plus ou moins proche de celle du bras humain. Il permet de remplacer, ou de prolonger, son action. Le rôle du S.M.A. est de guider l'organe terminal dans une situation (position et orientation) donnée, selon des caractéristiques de vitesse et d'accélération données. Il est composé d'une chaîne cinématique de corps, généralement rigides, assemblés par des liaisons appelées articulations. Sa motorisation est réalisée par des actionneurs électriques, pneumatiques ou hydrauliques qui transmettent leurs mouvements aux articulations par des systèmes appropriés.

### 1.7.3. Les articulations

Une articulation lie deux corps successifs en limitant le nombre de degré de liberté de l'un par rapport à l'autre. Une articulation complexe peut se ramener à une combinaison d'articulations prismatiques ou rotoïdes.

#### 1.7.3.1. Les articulations prismatiques

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



Fig.1.9. Représentation d'une articulation prismatique [10]

#### 1.7.3.2. Les articulations rotoïdes

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



Fig.1.10. Représentation d'une articulation rotoïde [10]

### 1.7.4. Organe terminal

On regroupe tous les dispositifs destinés à manipuler des objets (dispositifs de serrage, dispositifs magnétiques, à dépression, ...), ou à les transformer (outils, torche de soudage, pistolet de peinture, ...). En d'autres termes, il s'agit d'une interface permettant au robot d'interagir avec son environnement. Un organe terminal peut être multifonctionnel, au sens où il peut être équipé de plusieurs dispositifs ayant des fonctionnalités différentes. Il peut aussi être monofonctionnel, mais interchangeable. Un robot, enfin, peut-être multi-bras, chacun des

bras portant un organe terminal différent. On utilisera indifféremment le terme organe terminal, préhenseur, outil ou effecteur pour nommer le dispositif d'interaction fixé à l'extrémité mobile de la structure mécanique.

### **1.7.5. Les capteurs**

Les capteurs sont les organes de perception. Ils sont dits proprioceptifs lorsqu'ils mesurent l'état interne du robot (positions et vitesses des articulations) et extéroceptifs lorsqu'ils recueillent des informations sur l'environnement (détection de présence, de contact, mesure de distance, vision artificielle).

### **1.7.6. La commande**

La partie commande synthétise les consignes des asservissements pilotant les actionneurs, à partir de la fonction de perception et des ordres de l'utilisateur.

## **1.8. Classification des robots**

La classification des systèmes robotiques est difficile, car il existe de nombreux critères pour leurs descriptions. Cependant nous allons les classer selon deux structures qui sont les suivantes :

- Point de vue fonctionnel
- Point de vue géométrique

### **1.8.1. Point de vue fonctionnel [11]**

Le nombre de classes et les distinctions entre celles-ci varient de pays à pays (6 classes au Japon, 4 en France). L'A.F.R.I. distingue 4 classes illustrées ci-dessous :

#### **1.8.1.1. Manipulateur à commande manuelle**

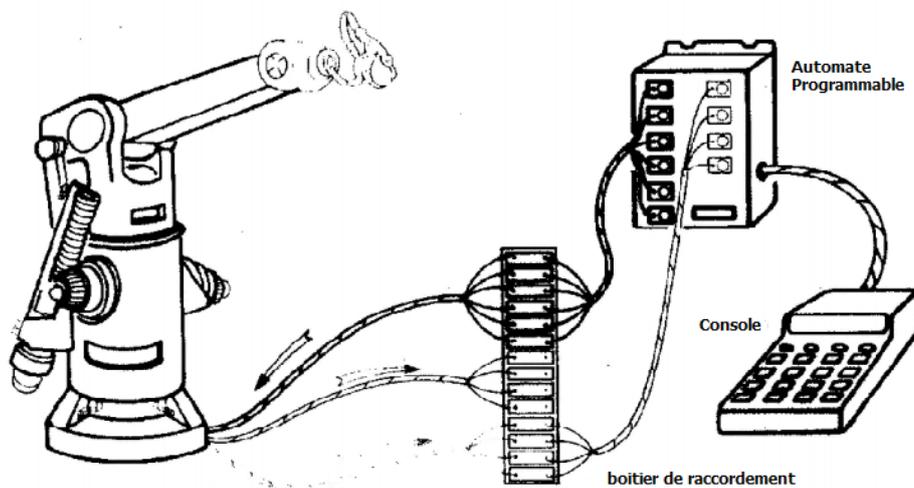
**Fig.1.11.** représente les manipulateurs à commande manuelle.



**Fig.1.11.** Manipulateur à commande manuelle [12]

### 1.8.1.2. Manipulateur automatique à cycles prééglés

Le réglage se fait mécaniquement par cames, butées, la commande peut se faire par automate programmable ; on peut distinguer entre manipulateurs à cycle fixe et manipulateurs à cycle programmable

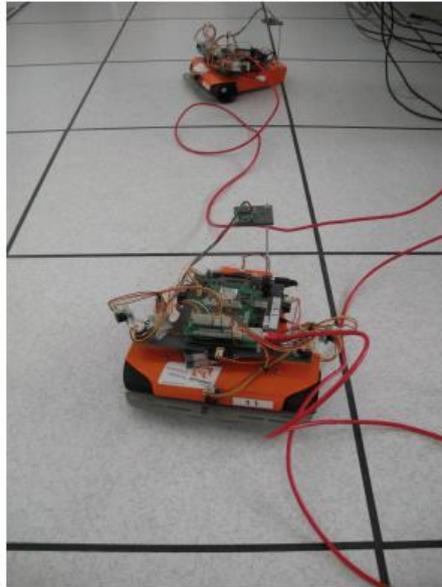


**Fig.1.12.** Manipulateur à cycle prééglé [11]

### 1.8.1.3. Robots programmables

Ils répètent les mouvements qu'on leur a appris ou programmés, sans informations sur l'environnement ou la tâche effectuée.

On peut aussi faire la distinction entre robots « play-back » qui reproduisent la tâche apprise et robots à commande numérique qui peuvent être programmés hors-ligne.



**Fig.1.13.** Robot programmable [13]

#### **1.8.1.4. Robots intelligents**

On trouve actuellement des robots de seconde génération qui sont capables d'acquérir et d'utiliser certaines informations sur leur environnement (systèmes de vision, détecteurs de proximité, capteurs d'efforts, ...). On étudie des robots de troisième génération, capables de comprendre un langage oral proche du langage naturel et de se débrouiller de façon autonome dans un environnement complexe, grâce à l'utilisation de l'intelligence artificielle.



**Fig.1.14.** Robot intelligent [14]

## 1.8.2. Classification géométrique

Il existe différentes architectures du porteur : la structure cartésienne (PPP), la structure cylindrique (RPP ou PRP), la structure sphérique ou polaire (RRP), la structure dite SCARA (RRP) et enfin la structure anthropomorphe (RRR).

### 1.8.2.1. Structure cartésienne (PPP)

Les caractéristiques d'une structure cartésienne (PPP) sont : trois axes deux à deux en série avec trois degrés de liberté, une très bonne précision et une grande lenteur. Le volume de travail est un parallélépipède dont les dimensions sont les translations permises par les trois liaisons prismatiques. [15]

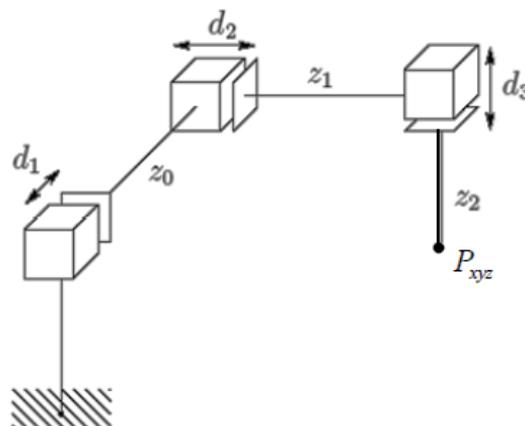
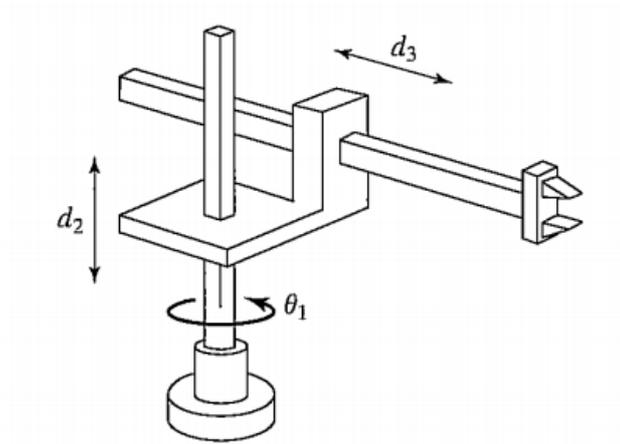


Fig.1.15. Structure PPP [16]

### 1.8.2.2. La structure cylindrique (RPP) ou (PRP)

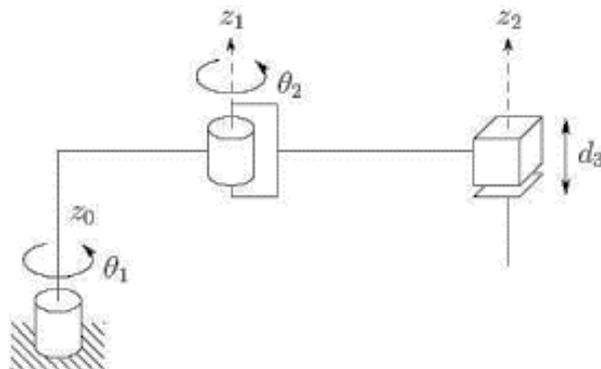
Les caractéristiques d'une structure cylindrique (RPP ou PRP) sont : trois axes non perpendiculaires en série avec trois degrés de liberté et une grande rapidité. Le volume de travail est un cylindre plein ou creux, autrement dit un tore à section rectangulaire, dont la hauteur L est la translation permise par une liaison prismatique.[16]



**Fig.1.16.** Structure RPP [17]

### 1.8.2.3. La structure sphérique ou polaire à axe de rotation orthogonale (RRP)

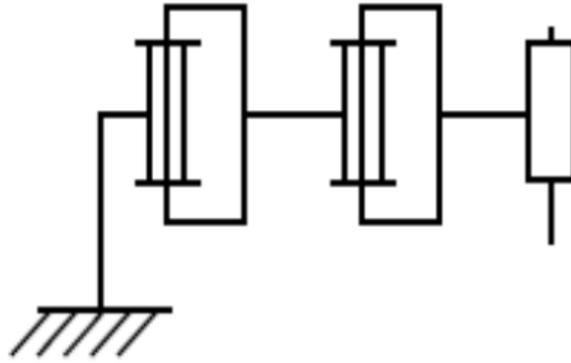
Le volume de travail est une sphère creuse, dont les rayons intérieur et extérieur sont fixés soit par la disposition de la liaison prismatique et la translation qu'elle permet, soit par les longueurs des deux parties du bras [19].



**Fig.1.17.** Structure RRP [18]

### 1.8.2.4. La structure dite SCARA

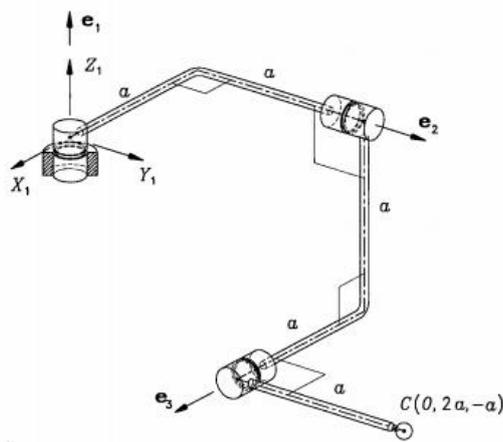
Les caractéristiques d'une structure SCARA (Selective Compliance Articulated Robot for Assemblage) sont : 3 axes en série cylindrique (RRP) ayant trois degrés de liberté. Les particularités de cette structure sont qu'elles sont précises et très rapides [19].



**Fig.1.18.** Structure SCARA [19]

### 1.8.2.5. La structure anthropomorphe (RRR)

Cette structure à une architecture plus généraliste reproduisant le bras humain. Son enveloppe de travail à une cinématique et dynamique complexes, mais sa configuration est plus flexible.



**Fig.1.19.** Structure RRR [20]

## 1.9. Domaines d'application de la robotique [21]

### 1.9.1 Industriel

Les robots sont très souvent utilisés dans le domaine de l'industrie comme, par exemple, dans le montage, la soudure ou la peinture.

### **1.9.2 Médical**

De nos jours, les médecins et plus précisément les chirurgiens sont assistés par les robots intelligents capables de pratiquer des incisions de plus en plus précises. On retrouve ces robots en chirurgie, télé-chirurgie et en macro-chirurgie.

### **1.9.3 Spatial**

L'espace étant très vaste, il est très difficile de l'explorer à l'échelle humaine. C'est pourquoi les robots sont très utilisés dans ce domaine.

## **1.10. Conclusion**

Dans ce premier chapitre, on a introduit les bases de la robotique en général. On va dès à présent commencer la partie technique et plus concrète de notre mémoire.

**CHAPITRE 2 :**

**DESCRIPTION  
ET  
MODÉLISATION  
DU BRAS  
MANIPULATEUR**

## 2.1. Introduction

L'une des premières étapes dans la réalisation d'un robot consiste à définir le mode de locomotion à mettre en œuvre ainsi que les différents éléments qui le composent. Dans ce second chapitre, on présentera l'architecture mécanique générale du bras manipulateur et on abordera les différents actionneurs et capteurs le constituant. On donnera également des notions théoriques qui nous aideront à l'aboutissement de la modélisation de notre système.

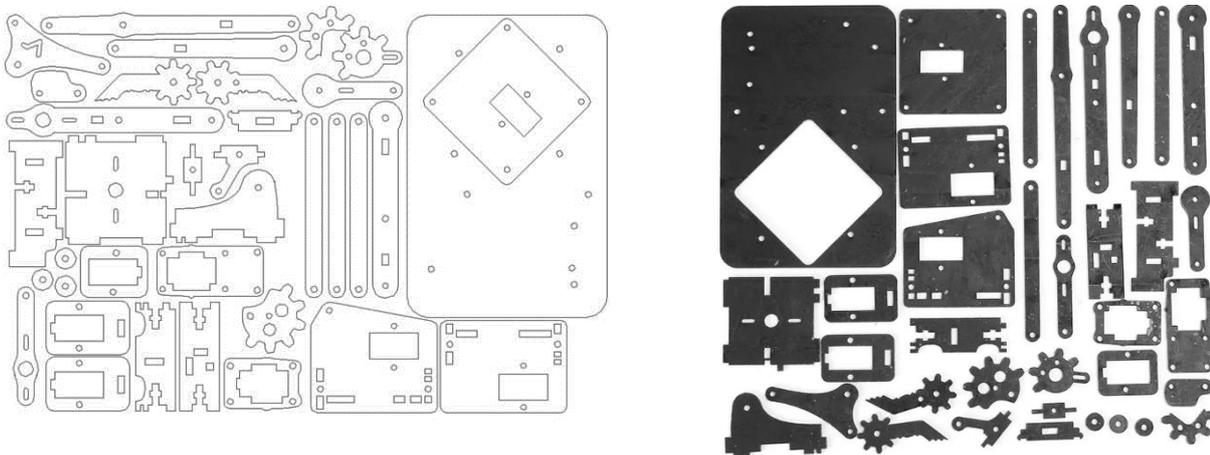
## 2.2. Description mécanique du bras manipulateur

### 2.2.1. La structure mécanique du bras

Pour la réalisation de notre robot, plusieurs choix s'offrent à nous. Notre premier choix s'est porté sur la technique à utiliser pour la fabrication du châssis. On a le choix entre deux techniques : l'impression en 3D ou la découpe laser. L'impression en 3D est rare, car les imprimantes 3D sont peu disponibles et ses bobines de recharge sont coûteuses. C'est donc pourquoi on a choisi la découpe laser.

Pour le matériau, on a opté pour du plexiglass, car celui-ci est connu pour son prix abordable et sa grande disponibilité.

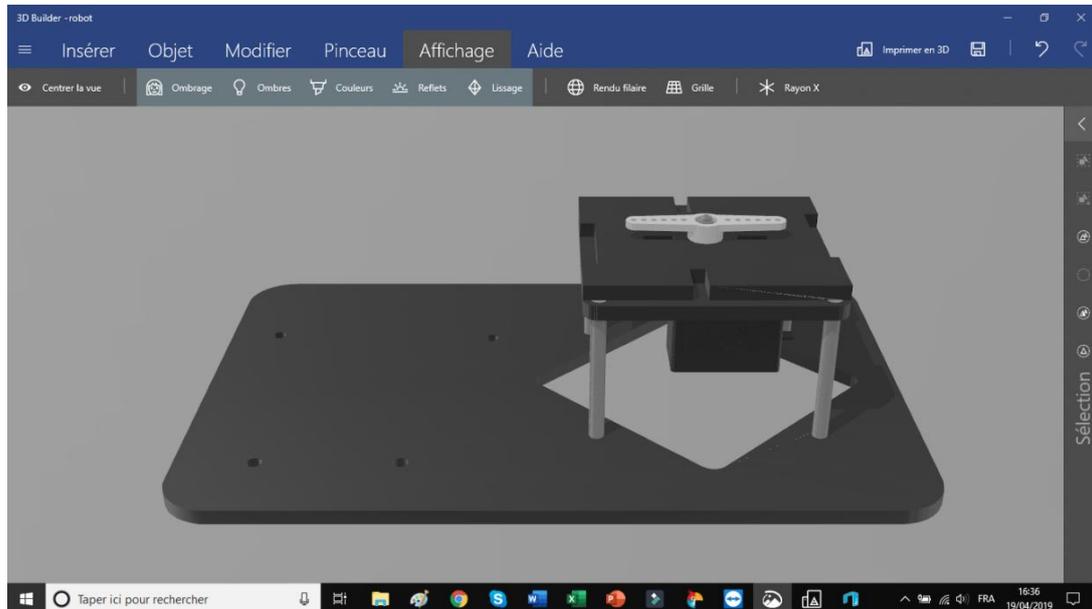
On a utilisé un modèle de bras en format .dxf pour découpe laser en téléchargement sur internet (<https://microbotlabs.com>) et on a procédé à la découpe des pièces (**Fig.2.1.**).



**Fig.2.1.** Les pièces constituant le robot

#### 2.2.1.1. La base

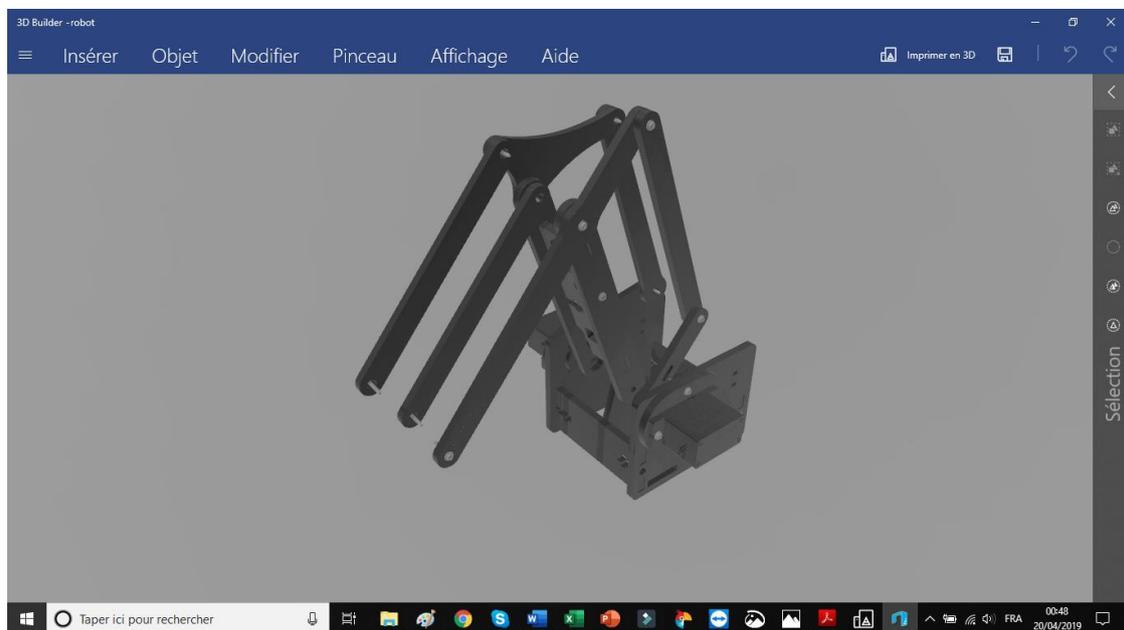
La base de notre robot est fixe. La première articulation possède un servomoteur qui permet au bras de se déplacer sur un angle de  $180^\circ$ , autour de la verticale.



**Fig.2.2.** Base du robot conçu sur 3D Builder

### 2.2.1.2. Le bras

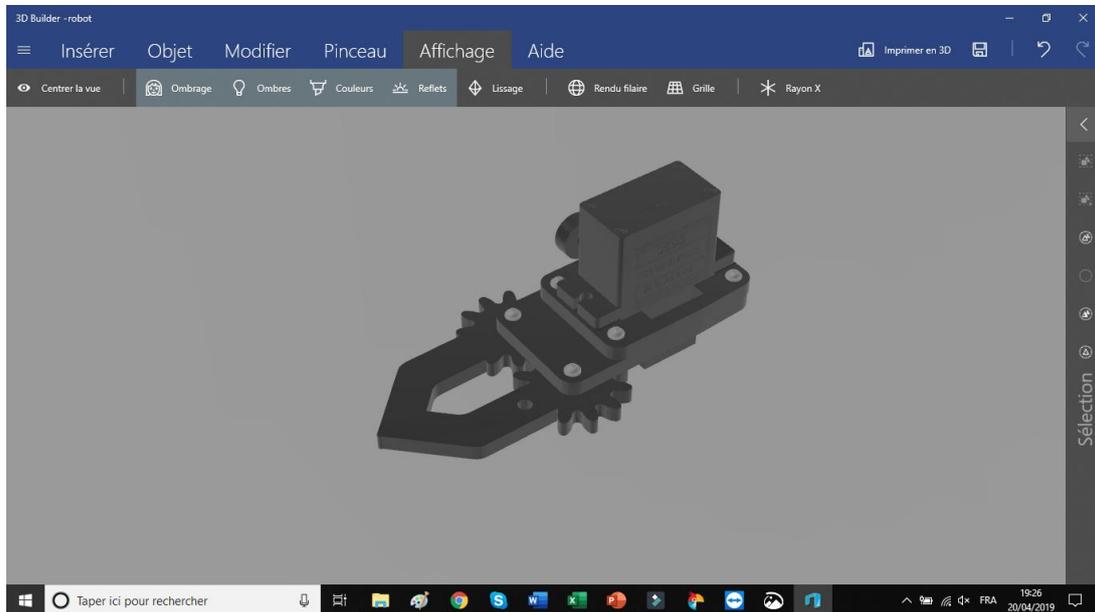
Il relie la base à la pince. Il se constitue d'une multitude de pièces en plexiglass reliées par des vis et de deux servomoteurs. Ils se trouvent sur les côtés comme l'illustre la **Fig.2.3**. Ceux-ci contrôlent les mouvements avant/arrière et haut/bas du bras.



**Fig.2.3.** Bras du robot conçu sur 3D Builder

### 2.2.1.3. La pince

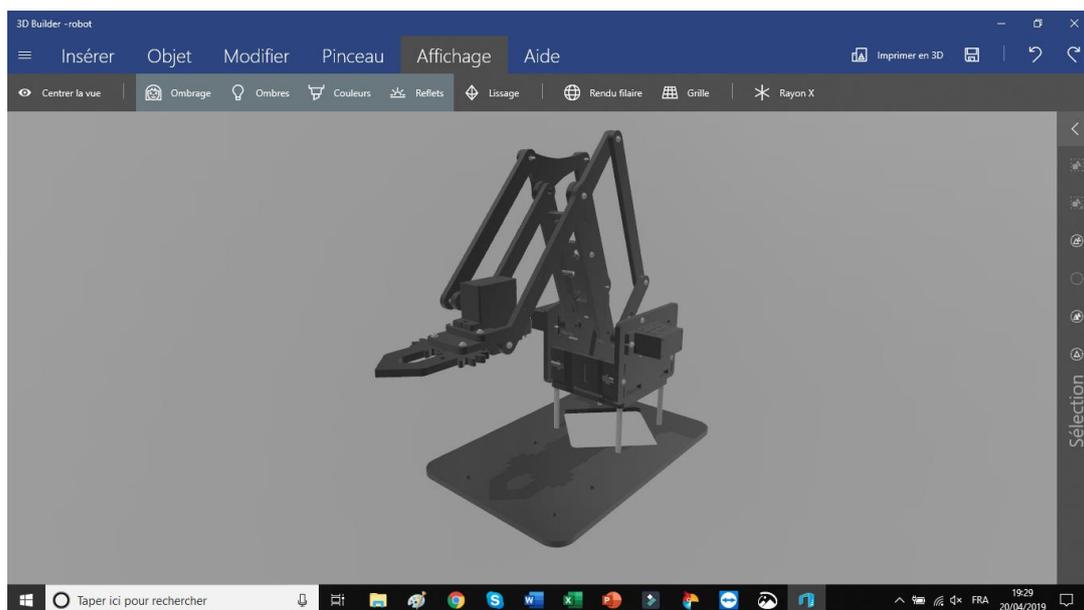
La pince est l'organe terminal de notre robot. En effet, elle permet de saisir les différents objets qu'on lui présente. La pince est constituée d'un servomoteur qui contrôle l'ouverture ou la fermeture de celle-ci grâce à un mécanisme d'engrenage.



**Fig.2.4.** Pince du robot conçu sur 3D Builder

### 2.2.1.4. Le robot

La structure globale du bras manipulateur avec pince est illustrée par **Fig.2.5.**



**Fig.2.5.** Bras robot entier conçu sur 3D Builder

Les dimensions exactes des pièces sont données dans l'Annexe E.

## 2.2.2. Description des parties constitutives du bras manipulateur

Dans cette partie, on va parler des types d'actionneurs et capteurs que nous avons utilisé pour notre robot.

### 2.2.2.1. Les actionneurs du bras manipulateur

Le rôle des actionneurs est d'exécuter une tâche. Ils permettent de fournir assez de force pour pouvoir provoquer un mouvement. On a choisi d'utiliser comme actionneurs quatre servomoteurs qui sont des moteurs à courant continu (MCC), asservis en position à l'aide d'un capteur de position et d'un circuit électronique interne au moteur.

L'asservissement est un moyen de gérer, corriger une commande en fonction d'une consigne et d'un capteur de position. Pour pouvoir réaliser la correction de l'angle du bras, le servomoteur utilise une électronique d'asservissement. Cette électronique est constituée d'un comparateur qui compare la position du bras du servomoteur à la consigne. Après une rapide comparaison entre la consigne et la valeur réelle de la position du bras, le servomoteur applique une correction si le bras n'est pas orienté à l'angle imposé par la consigne [22].

Les servomoteurs sont pilotés par un fil de commande (jaune) et alimentés par deux autres fils : le premier est relié à l'alimentation positive +5V ou +6V (rouge), le deuxième est relié à la masse (noir).

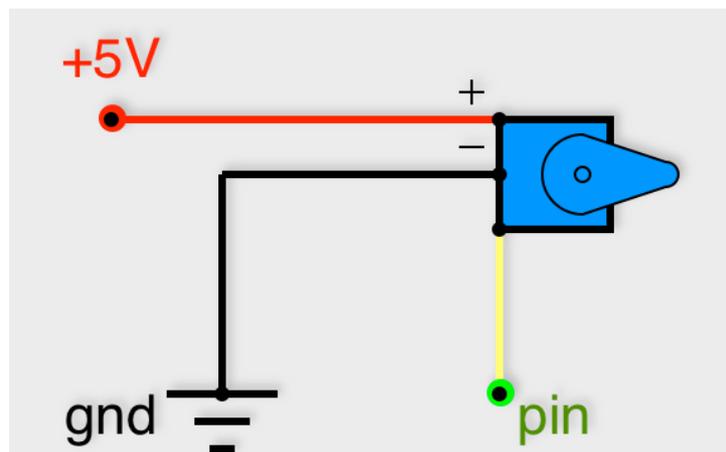


Fig.2.6. Schéma d'un actionneur



**Fig.2.7.** De gauche à droite : (a) servomoteur du type MG90S [23], (b) servomoteur du Type SG90 [24].

### 2.2.2.2. Les paramètres techniques des servomoteurs

Les paramètres techniques des servomoteurs utilisés sont donnés par **Tab.2.1.**

**Tab.2.1.** Tableau des caractéristiques des servomoteurs du bras [23][24].

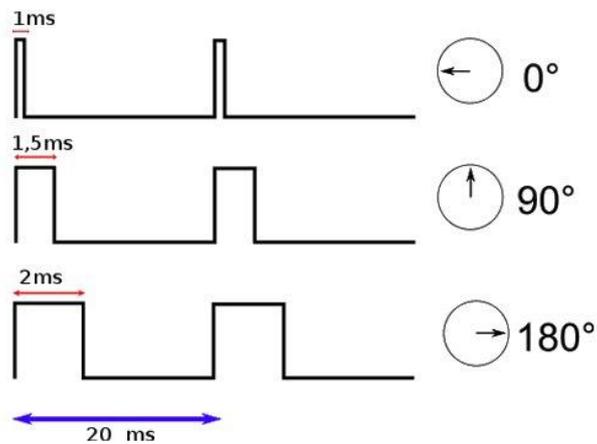
Références Caractéristiques	MG90S	SG90
Poids	13,4g	14,7g
Dimensions	22,5 x 12 x 35,5 mm	32 x 12 x 32 mm
Couples d'arrêt	1,8 kgf.cm (4,8V), 2,2 kgf.cm (6V)	2,5 kgf.cm
Tensions de fonctionnement	4,8V – 6V	4,8V – 6V
Vitesses	0,1s/60° (4,8V), 0,08s/60° (6V)	0,1s/60°
Angles de rotation	180°	180°

### 2.2.2.3. La commande des servomoteurs

La consigne envoyée au servomoteur est un signal électronique de type PWM. Il dispose cependant de deux caractéristiques indispensables pour que le servomoteur puisse fonctionner :

- La fréquence fixe : le signal qu'on va devoir générer doit avoir une fréquence de 50 Hz,

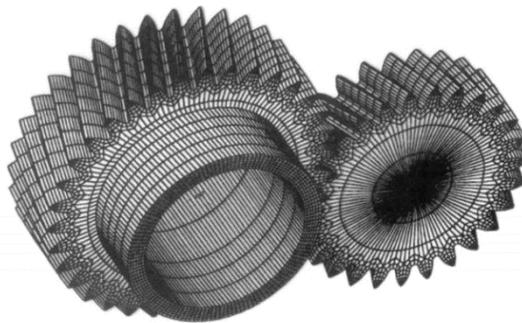
- La durée de l'état haut : cette durée indique au servomoteur l'angle précis qui est souhaité par l'utilisateur [22].



**Fig.2.8.** Exemple de signaux en position pour un servomoteur [22].

#### 2.2.2.4. Transmission entre articulations : les engrenages

On trouve, au niveau de la pince, des engrenages reliés au servomoteur. Les transmissions par engrenages sont des organes mécaniques utilisés pour transformer et transmettre à un organe récepteur le couple et le mouvement de rotation générés par un moteur. [25]



**Fig.2.9.** Schéma d'un engrenage [25]

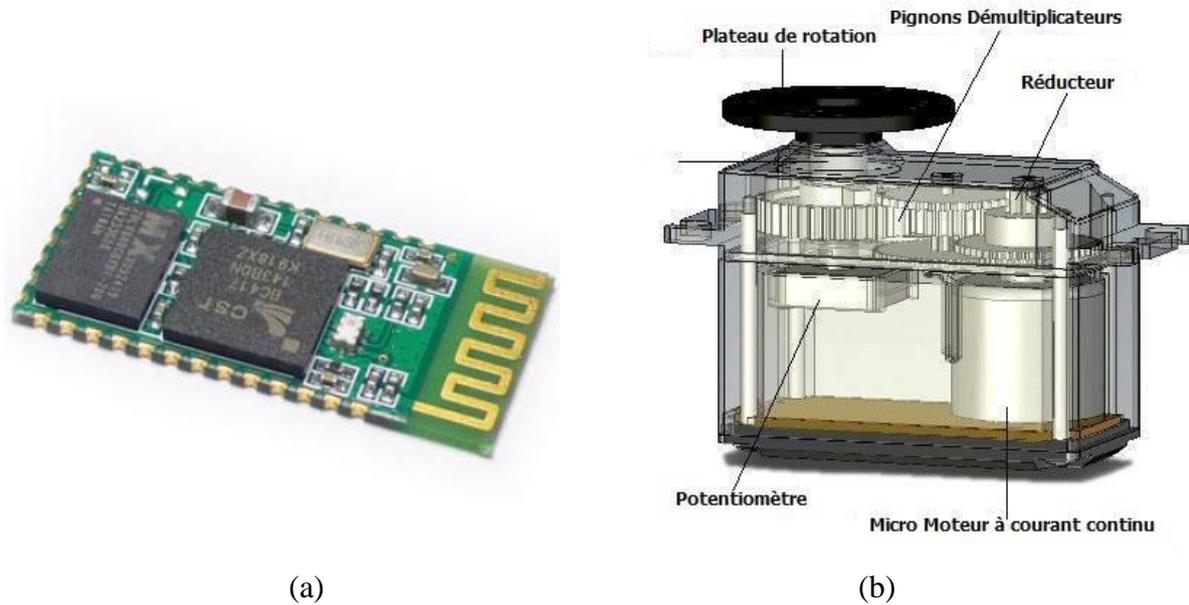
#### 2.2.2.5. Les capteurs

Le rôle des capteurs est de gérer les relations entre le robot et son environnement. Ils permettent de contrôler plus facilement les tâches qu'on lui ordonne.

Il est important de distinguer deux types de capteurs :

- Les capteurs proprioceptifs qui mesurent l'état mécanique interne du robot,
- Les capteurs extéroceptifs qui recueillent des informations sur l'environnement.

Dans notre projet, on suppose qu'on travaille dans un espace de travail sans obstacle. L'utilisation de capteurs de détection n'est donc pas nécessaire. On n'a aussi pas besoin de rajouter des capteurs de position, car ceux-ci sont déjà intégrés dans les servomoteurs. En revanche, on a utilisé un récepteur Bluetooth afin de pouvoir commander ses déplacements à l'aide d'une application mobile.



**Fig.2.10.** De gauche à droite : (a) récepteur bluetooth [26], (b) schéma d'un servomoteur [22].

### 2.2.2.6. Les paramètres techniques du récepteur bluetooth

Les spécifications techniques du composant Bluetooth utilisé sont données par **Tab.2.2.**

**Tab.2.2.** Tableau des caractéristiques du récepteur Bluetooth du bras [26].

Caractéristiques	Bluetooth HC-05
Sensibilité typique	-80dBm
Puissance d'émission RF	Jusqu'à +4 dBm
Tension de fonctionnement	1,8V - 3,6V

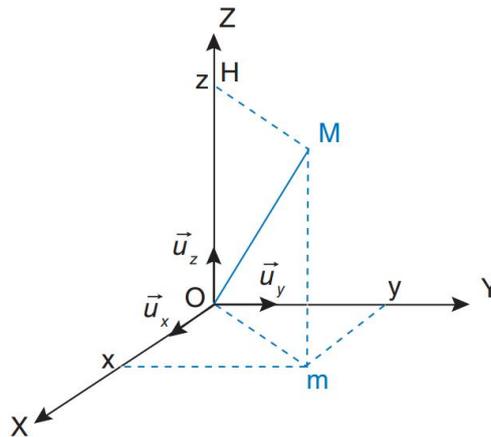
## 2.3. Modélisation

### 2.3.1. Repères et référentiels

À un instant donné, on repère la position du point M par le vecteur  $\vec{r} = \overrightarrow{OM}$  appelé vecteur position. Ce dernier est représenté par les valeurs algébriques des projections sur une base orthonormée [27].

### 2.3.1.1. Coordonnées cartésiennes

On considère un repère constitué de trois axes rattachés à un point origine  $O$ , caractéristique du solide de référence  $(R)$ . À ce repère on associe une base orthonormée directe  $(\vec{u}_x, \vec{u}_y, \vec{u}_z)$ . Les vecteurs  $\vec{u}_x, \vec{u}_y, \vec{u}_z$  sont alors les vecteurs unitaires des axes  $OX, OY$  et  $OZ$  respectivement [26]. Les coordonnées cartésiennes sont les plus utilisées en robotique, car elles sont les plus simples pour la mesure des distances.

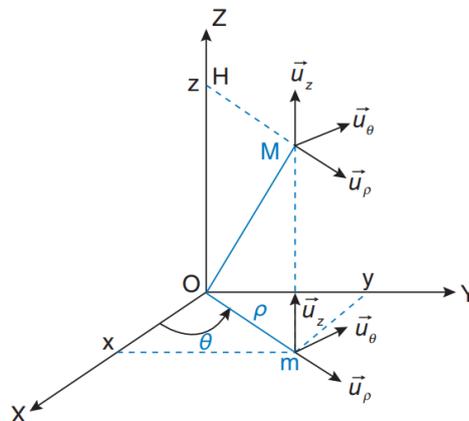


**Fig.2.11.** Le système de coordonnées cartésiennes [27]

### 2.3.1.2. Coordonnées cylindriques

La position du point  $M$  est définie dans un repère  $(O, \vec{u}_\rho, \vec{u}_\theta, \vec{u}_z)$ . On introduit la base  $(\vec{u}_\rho, \vec{u}_\theta, \vec{u}_z)$  orthonormée directe, associée aux coordonnées cylindriques  $(\rho, \theta, z)$ . Les relations entre les coordonnées cylindriques et cartésiennes sont les suivantes [27]

$$\begin{cases} x = \rho \cos(\theta) \\ y = \rho \sin(\theta) \\ z = z \end{cases} \quad (2.1)$$

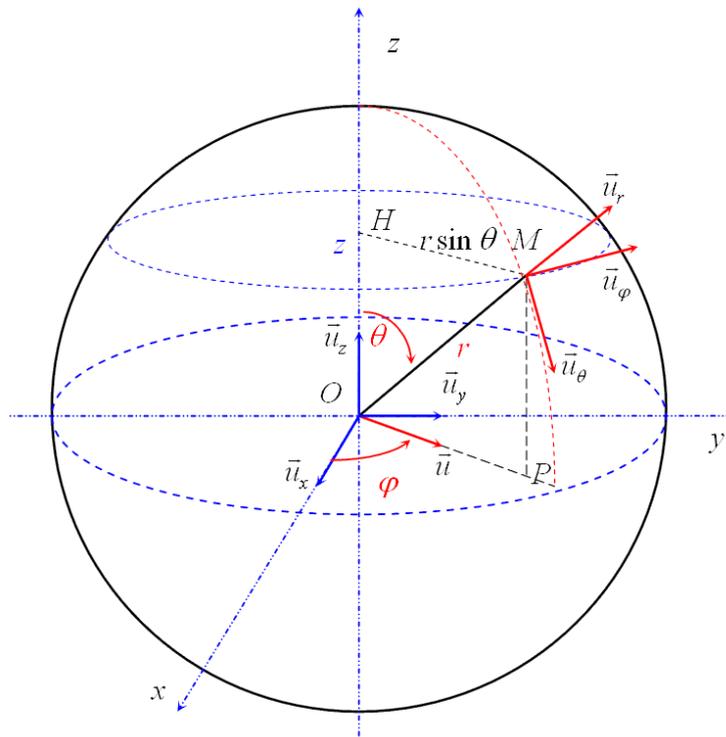


**Fig.2.12.** Le système de coordonnées cylindriques [27]

### 2.3.1.3. Coordonnées sphériques

La position du point M est définie dans un repère  $(O, \vec{u}_\rho, \vec{u}_\theta, \vec{u}_\varphi)$ . On introduit la base  $(\vec{u}_\rho, \vec{u}_\theta, \vec{u}_\varphi)$  orthonormée directe, associée aux coordonnées sphériques  $(\rho, \theta, \varphi)$ . Les relations entre les coordonnées sphériques et cartésiennes sont les suivantes [28] :

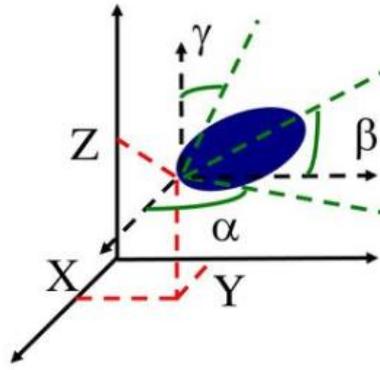
$$\begin{cases} x = \rho \sin(\theta) \cos(\varphi) \\ y = \rho \sin(\theta) \sin(\varphi) \\ z = \rho \cos(\theta) \end{cases} \quad (2.2)$$



**Fig.2.13.** Le système de coordonnées sphériques [28].

### 2.3.2. Position et orientation d'un solide

Si pour décrire la position d'un point, trois dimensions sont nécessaires et suffisantes, ce n'est plus le cas pour décrire la position d'un solide et son orientation. Il faut trois coordonnées pour positionner le centre d'inertie des solides et trois autres coordonnées pour son orientation dans l'espace. Six coordonnées sont donc nécessaires et suffisantes pour placer un objet dans l'espace [28].



**Fig.2.14.** Position d'un solide dans l'espace et son repère associé [29]

### 2.3.3. Modèle géométrique direct [21]

Le modèle géométrique direct (MGD) est l'ensemble des relations qui permettent d'exprimer la situation de l'organe terminal, c'est-à-dire les coordonnées opérationnelles du robot, en fonction de ses coordonnées articulaires. Dans le cas d'une chaîne ouverte simple, il peut être représenté par la matrice de transformation  ${}^0T_n$  :

$${}^0T_n = {}^0T_1(q_1) {}^1T_2(q_2) \dots {}^{n-1}T_n(q_n) \quad (2.3)$$

Le MGD du robot peut aussi être représenté par la relation :

$$X = f(q) \quad (2.4)$$

Avec,  $q$  : Vecteur des variables articulaires tel que :

$$q = [q_1 \ q_2 \ \dots \ q_n]^T \quad (2.5)$$

Les coordonnées opérationnelles sont définies par :

$$X = [x_1 \ x_2 \ \dots \ x_m]^T \quad (2.6)$$

Plusieurs possibilités existent pour définir le vecteur  $X$ . Par exemple, avec les éléments de la matrice  ${}^0T_n$  :

$$X = [P_x \ P_y \ P_z \ S_x \ S_y \ S_z \ n_x \ n_y \ n_z \ a_x \ a_y \ a_z]^T \quad (2.7)$$

Ou bien, sachant que  $s = n_x a$  :

$$X = [P_x \ P_y \ P_z \ n_x \ n_y \ n_z \ a_x \ a_y \ a_z]^T \quad (2.8)$$

### 2.3.3.1. Paramètre de Denavit-Hartenberg (DH) [21]

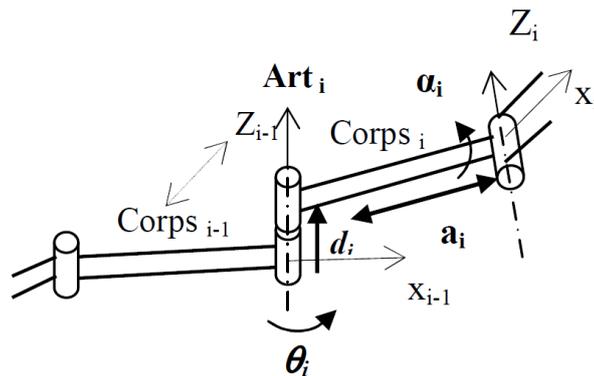
Les paramètres DH sont utilisés pour systématiser la modélisation de n'importe quel type de robot série. Ils permettent de simplifier le modèle géométrique, mais également d'établir une norme reconnue par tous.

Afin d'établir les paramètres de DH, chacun des repères est établi en se basant sur trois règles :

- L'axe  $Z_{i-1}$  est selon l'axe de mouvement de l'art- $i$ .
- L'axe  $x_{i-1}$  est aligné entre les articulations  $i$  et  $(i+1)$ . Si  $a_i$  est nul, alors  $x_{i-1}$  est perpendiculaire à  $z_{i-1}$  et  $z_i$ .
- L'axe  $Y_i$  complète le repère  $i$  pour former un repère « main droite » .

Pour passer de  $R_{i-1}$  à  $R_i$ , on peut suivre le chemin suivant :

- |                                     |  |
|-------------------------------------|--|
| 1- Rotation ( $z_{i-1}, \theta_i$ ) | Avec, $a_i$ : Distance entre $z_{i-1}$ et $z_i$ , le long de $x_i$ .     |
| 2- Translation ( $z_{i-1}, d_i$ )   | $\alpha_i$ : Angle entre $z_{i-1}$ et $z_i$ , autour de $x_i$ .          |
| 3- Translation ( $x_i, a_i$ )       | $\theta_i$ : Angle entre $x_{i-1}$ et $x_i$ , autour de $z_{i-1}$ .      |
| 4- Rotation ( $x_i, \alpha_i$ )     | $d_i$ : Distance de $O_{i-1}$ à l'intersection de $z_{i-1}$ avec $x_i$ . |



**Fig.2.15.** Schématisation du lien entre le corps  $i-1$  et  $i$  [21]

La matrice de passage d'une articulation à la suivante est donc :

$$T = T_{i-1}^i = T_{z,\theta_i} \cdot T_{z,d_i} \cdot T_{x,a_i} \cdot T_{x,\alpha_i} = \begin{bmatrix} c\theta_i & -c\alpha_i \cdot s\theta_i & s\alpha_i \cdot s\theta_i & a_i \cdot c\theta_i \\ s\theta_i & c\alpha_i \cdot c\theta_i & -s\alpha_i \cdot c\theta_i & a_i \cdot s\theta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.9)$$

Le MGD d'un bras de robot est obtenu en utilisant le produit :

$$T_{MG} = T_0^n = T_0^1 \cdot T_1^2 \cdot \dots \cdot T_{n-1}^n \quad (2.10)$$

### 2.3.3.2. Modélisation géométrique direct de notre réalisation

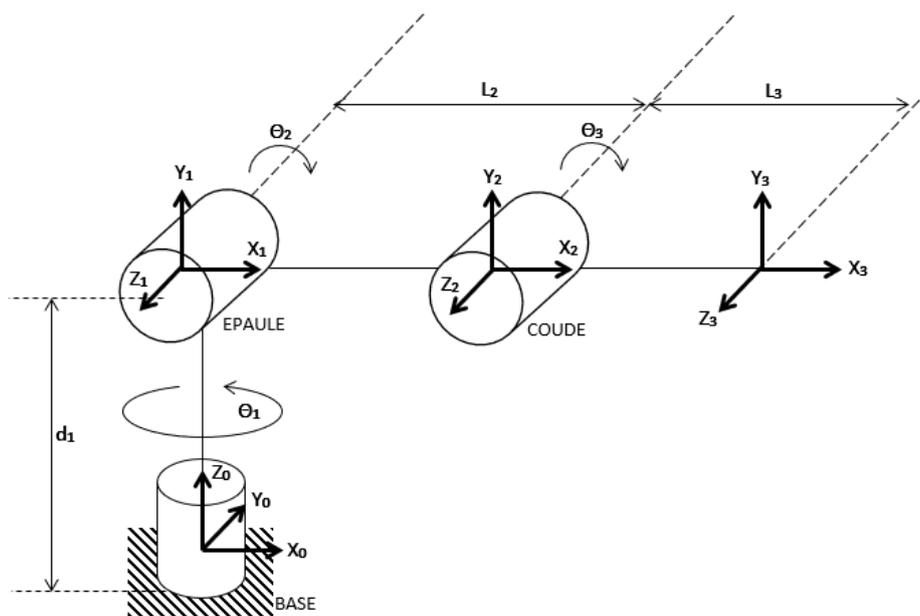
Notre réalisation est un bras manipulateur de trois ddl qui se compose d'un organe terminal et d'une structure mécanique articulée. L'organe terminal est constitué d'un servomoteur qui contrôle l'ouverture et la fermeture de la pince. La structure mécanique contient trois servomoteurs qui contrôlent les articulations (de types rotoïdes). Le rôle de cette structure est d'amener l'organe terminal dans une situation (position et orientation) donnée. Le **Tab.2.3.** donne le type d'articulations et les angles de rotations possibles de notre réalisation.

**Tab.2.3.** Types d'articulations et d'angles de rotations possibles du bras réalisé.

Type d'articulations, d'angles et de rotations de notre réalisation			
Nombre d'articulations		3	
Nombre d'actionneurs		4 servomoteurs	
Rotations		Angles	Types
La base		180°	Rotoïde
Le bras	L'épaule	180°	Rotoïde
	Le coude	180°	Rotoïde
Ouverture de la pince (en mm)		80	

### 2.3.3.3. Identification des paramètres de DH de notre bras manipulateur

**Fig.2.16.** représente le placement des repères, selon la convention DH, que l'on a utilisé pour élaborer le MGD du bras.



**Fig.2.16.** Placement des repères selon le modèle DH.

Les paramètres, selon DH, sont donnés par **Tab.2.4**.

**Tab.2.4.** Tableau des paramètres de DENAVIT- HATENBERG de notre réalisation.

Segment	$\theta_i$	$d_i$	$a_i$	$\alpha_i$
1	$\theta_1$	$d_1$	0	$90^\circ$
2	$\theta_2$	0	$L_2$	0
3	$\theta_3$	0	$L_3$	0

$$T_0^1 = \begin{bmatrix} c\theta_1 & -c\alpha_1 \cdot s\theta_1 & s\alpha_1 \cdot s\theta_1 & a_1 \cdot c\theta_1 \\ s\theta_1 & c\alpha_1 \cdot c\theta_1 & -s\alpha_1 \cdot c\theta_1 & a_1 \cdot s\theta_1 \\ 0 & s\alpha_1 & c\alpha_1 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} C_1 & 0 & S_1 & 0 \\ S_1 & 0 & -C_1 & 0 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.11)$$

$$T_1^2 = \begin{bmatrix} c\theta_2 & -c\alpha_2 \cdot s\theta_2 & s\alpha_2 \cdot s\theta_2 & a_2 \cdot c\theta_2 \\ s\theta_2 & c\alpha_2 \cdot c\theta_2 & -s\alpha_2 \cdot c\theta_2 & a_2 \cdot s\theta_2 \\ 0 & s\alpha_2 & c\alpha_2 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} C_2 & -S_2 & 0 & L_2 C_2 \\ S_2 & C_2 & 0 & L_2 S_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.12)$$

$$T_2^3 = \begin{bmatrix} c\theta_3 & -c\alpha_3 \cdot s\theta_3 & s\alpha_3 \cdot s\theta_3 & a_3 \cdot c\theta_3 \\ s\theta_3 & c\alpha_3 \cdot c\theta_3 & -s\alpha_3 \cdot c\theta_3 & a_3 \cdot s\theta_3 \\ 0 & s\alpha_3 & c\alpha_3 & d_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} C_3 & -S_3 & 0 & L_3 C_3 \\ S_3 & C_3 & 0 & L_3 S_3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.13)$$

Le MGD s'obtient à partir de l'équation (2.10) qui nous donne le résultat suivant :

$$T_0^3 = T_0^1 T_1^2 T_2^3 = \begin{bmatrix} C_1 C_{23} & -C_1 S_{23} & S_1 & C_1(L_3 C_{23} + L_2 C_2) \\ S_1 C_{23} & -S_1 S_{23} & -C_1 & S_1(L_3 C_{23} + L_2 C_2) \\ S_{23} & C_{23} & 0 & L_3 S_{23} + L_2 S_2 + d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.14)$$

### 2.3.4. Le modèle géométrique inverse (MGI) [21]

On a vu que le MGD d'un robot permettait de calculer les coordonnées opérationnelles donnant la situation de l'organe terminal en fonction des coordonnées articulaires. Le problème inverse consiste à calculer les coordonnées articulaires correspondant à une situation donnée de l'organe terminal.

On peut distinguer trois méthodes de calcul du MGI :

- La méthode de Paul qui traite séparément chaque cas particulier et convient pour la plupart des robots industriels,
- La méthode de Pieper qui permet à résoudre le problème des robots à six ddl possédant trois articulations rotoïdes d'axes concourants ou trois articulations prismatiques,

- La méthode générale de Raghavan et Roth, donnant la solution générale des robots à six articulations à partir d'un polynôme de degré, au plus, égal à 16.

### 2.3.4.1. Le modèle géométrique inverse de notre réalisation

Pour calculer  $\theta_1$ ,  $\theta_2$  et  $\theta_3$ , on a :

$$T_0^3 = \begin{bmatrix} r_{11} & r_{12} & r_{13} & r_{14} \\ r_{21} & r_{22} & r_{23} & r_{24} \\ r_{31} & r_{32} & r_{33} & r_{34} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} C_1 C_{23} & -C_1 S_{23} & S_1 & C_1(L_3 C_{23} + L_2 C_2) \\ S_1 C_{23} & -S_1 S_{23} & -C_1 & S_1(L_3 C_{23} + L_2 C_2) \\ S_{23} & C_{23} & 0 & L_3 S_{23} + L_2 S_2 + d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.15)$$

On remarque que le calcul de  $\theta_1$  est directement obtenue à partir de  $r_{13}$  et  $r_{23}$ , ce qui implique que :

$$\theta_1 = \text{atan2}(r_{13}, -r_{23})$$

Le calcul de  $\theta_2$  peut se faire en utilisant les composantes  $r_{14}$  et  $r_{34}$ , tel que :

$$r_{14} = (L_3 C_1 C_{23} + L_2 C_1 C_2) \quad (2.16)$$

$$r_{34} = (L_3 S_{23} + L_2 S_2 + d_1) \quad (2.17)$$

Ce qui implique que :

$$C_2 = \frac{r_{14} - L_3 C_1 C_{23}}{L_2 C_1} \quad (2.18)$$

$$S_2 = \frac{r_{34} - L_3 r_{31} - d_1}{L_2} \quad (2.19)$$

Ce qui nous donne :

$$\theta_2 = \text{atan2}(S_2, C_2) \quad (2.20)$$

On obtient le calcul de  $\theta_3$  à partir de  $r_{31}$  et  $r_{32}$  comme suit :

$$\text{On pose :} \quad \beta = \text{atan2}(r_{31}, r_{32}) \quad (2.21)$$

Ce qui nous donne :

$$\theta_3 = \beta - \theta_2 \quad (2.22)$$

### 2.3.5. Modélisation cinématique direct (MCD) [21]

Le MCD d'un robot manipulateur décrit les vitesses des coordonnées opérationnelles en fonction des vitesses articulaires. Il est noté :

$$\dot{X} = J(q)\dot{q} \quad (2.23)$$

Avec,  $J(q)$  : Matrice jacobienne de dimension  $(m \times n)$  du mécanisme.

On peut obtenir la matrice jacobienne par une méthode de calcul direct, fondée sur la relation entre les vecteurs des vitesses de translation et de rotation  $V_n$  et  $\omega_n$  du repère  $R_n$ , et les vitesses articulaires  $\dot{q}$  :

$$\begin{bmatrix} V_n \\ \omega_n \end{bmatrix} = J_n \dot{q} \quad (2.24)$$

#### 2.3.5.1. Modèle cinématique direct de notre réalisation

On commence par le calcul des vitesses angulaires :

$$\omega_0^0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad (2.25)$$

$$\omega_0^1 = \omega_0^0 + \theta_1^0 Z_0^0 = \begin{bmatrix} 0 \\ 0 \\ \dot{\theta}_1 \end{bmatrix} \quad (2.26)$$

$$\omega_1^1 = \omega_0^1 + R_1^0 \omega_0^1 = \begin{bmatrix} c_1 & s_1 & 0 \\ 0 & 0 & 1 \\ s_1 & -c_1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ \dot{\theta}_1 \end{bmatrix} = \begin{bmatrix} 0 \\ \dot{\theta}_1 \\ 0 \end{bmatrix} \quad (2.27)$$

$$\omega_1^2 = \omega_1^1 + \theta_2^1 Z_1^1 = \begin{bmatrix} 0 \\ \dot{\theta}_1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dot{\theta}_2 \end{bmatrix} = \begin{bmatrix} 0 \\ \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} \quad (2.28)$$

$$\omega_2^2 = R_2^1 \omega_1^2 = \begin{bmatrix} c_2 & s_2 & 0 \\ -s_2 & c_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} = \begin{bmatrix} \dot{\theta}_1 s_2 \\ \dot{\theta}_1 c_2 \\ \dot{\theta}_2 \end{bmatrix} \quad (2.29)$$

$$\omega_2^3 = \omega_2^2 + \theta_3^2 Z_2^2 = \begin{bmatrix} \dot{\theta}_1 s_2 \\ \dot{\theta}_1 c_2 \\ \dot{\theta}_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ \dot{\theta}_3 \end{bmatrix} = \begin{bmatrix} \dot{\theta}_1 s_2 \\ \dot{\theta}_1 c_2 \\ \dot{\theta}_2 + \dot{\theta}_3 \end{bmatrix} \quad (2.30)$$

$$\omega_3^3 = R_3^2 \omega_2^3 = \begin{bmatrix} c_3 & s_3 & 0 \\ -s_3 & c_3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 s_2 \\ \dot{\theta}_1 c_2 \\ \dot{\theta}_2 + \dot{\theta}_3 \end{bmatrix} = \begin{bmatrix} \dot{\theta}_1 (c_3 s_2 + s_3 c_2) \\ \dot{\theta}_1 (c_2 c_3 - s_2 s_3) \\ \dot{\theta}_2 + \dot{\theta}_3 \end{bmatrix} = \begin{bmatrix} \dot{\theta}_1 s_{23} \\ \dot{\theta}_1 c_{23} \\ \dot{\theta}_2 + \dot{\theta}_3 \end{bmatrix} \quad (2.31)$$

$$\begin{aligned}
\omega_0^3 = R_0^3 \omega_3^3 &= \begin{bmatrix} c_1 c_{23} & -c_1 s_{23} & s_1 \\ s_1 c_{23} & -s_1 s_{23} & -c_1 \\ s_{23} & c_{23} & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 s_{23} \\ \dot{\theta}_1 c_{23} \\ \dot{\theta}_2 + \dot{\theta}_3 \end{bmatrix} \\
&= \begin{bmatrix} 0 & s_1 & s_1 \\ 0 & -c_1 & -c_1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix} \quad (2.32)
\end{aligned}$$

Ensuite, on poursuit par les vitesses linéaires :

$$V_0^0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad (2.33)$$

$$V_0^1 = V_0^0 + \omega_0^1 P_0^1 = \begin{bmatrix} 0 \\ 0 \\ \dot{\theta}_1 \end{bmatrix} \times \begin{bmatrix} 0 \\ 0 \\ d_1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad (2.34)$$

$$V_1^1 = R_1^0 V_0^1 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \quad (2.35)$$

$$V_1^2 = V_1^1 + \omega_1^2 P_1^2 = \begin{bmatrix} 0 \\ \dot{\theta}_1 \\ \dot{\theta}_2 \end{bmatrix} \times \begin{bmatrix} L_2 c_2 \\ L_2 s_2 \\ 0 \end{bmatrix} = \begin{bmatrix} -\dot{\theta}_2 L_2 s_2 \\ \dot{\theta}_2 L_2 c_2 \\ -\dot{\theta}_1 L_2 c_2 \end{bmatrix} \quad (2.36)$$

$$V_2^2 = R_2^1 V_1^2 = \begin{bmatrix} c_2 & s_2 & 0 \\ -s_2 & c_2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -\dot{\theta}_2 L_2 s_2 \\ \dot{\theta}_2 L_2 c_2 \\ -\dot{\theta}_1 L_2 c_2 \end{bmatrix} = \begin{bmatrix} 0 \\ L_2 \dot{\theta}_2 \\ -\dot{\theta}_1 L_2 c_2 \end{bmatrix} \quad (2.37)$$

$$V_2^3 = V_2^2 + \omega_2^3 P_2^3 = \begin{bmatrix} 0 \\ L_2 \dot{\theta}_2 \\ -\dot{\theta}_1 L_2 c_2 \end{bmatrix} + \begin{bmatrix} \dot{\theta}_1 s_2 \\ \dot{\theta}_1 c_2 \\ \dot{\theta}_2 + \dot{\theta}_3 \end{bmatrix} \times \begin{bmatrix} L_3 c_3 \\ L_3 s_3 \\ 0 \end{bmatrix} \quad (2.38)$$

$$V_3^3 = R_3^2 V_2^3 = \begin{bmatrix} c_3 & s_2 & 0 \\ -s_3 & c_3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} -(\dot{\theta}_2 + \dot{\theta}_3) L_3 s_3 \\ L_2 \dot{\theta}_2 + L_3 c_3 (\dot{\theta}_2 + \dot{\theta}_3) \\ -\dot{\theta}_1 L_3 c_{23} \end{bmatrix} = \begin{bmatrix} \dot{\theta}_2 L_2 s_3 \\ \dot{\theta}_2 L_2 c_3 + L_3 (\dot{\theta}_2 + \dot{\theta}_3) \\ -\dot{\theta}_1 L_3 c_{23} \end{bmatrix} \quad (2.39)$$

$$\begin{aligned}
V_0^3 = R_0^3 V_3^3 &= \begin{bmatrix} c_1 c_{23} & -c_1 s_{23} & s_1 \\ s_1 c_{23} & -s_1 s_{23} & -c_1 \\ s_{23} & c_{23} & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_2 L_2 s_3 \\ \dot{\theta}_2 L_2 c_3 + L_3 (\dot{\theta}_2 + \dot{\theta}_3) \\ -\dot{\theta}_1 L_3 c_{23} \end{bmatrix} \\
&= \begin{bmatrix} -L_3 c_{23} s_1 & L_2 c_1 (s_3 c_{23} - c_3 s_{23}) - L_3 c_1 s_{23} & -L_3 c_1 s_{23} \\ L_3 c_{23} c_1 & L_2 s_1 (s_3 c_{23} - c_3 s_{23}) - L_3 s_1 s_{23} & -L_3 s_1 s_{23} \\ 0 & L_2 (s_3 s_{23} + c_3 c_{23}) + L_3 c_{23} & L_3 c_{23} \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix} \quad (2.40)
\end{aligned}$$

Le modèle cinématique dans  $R_3$  est donné par :

$$\begin{bmatrix} V_3^3 \\ \omega_3^3 \end{bmatrix} = \begin{bmatrix} 0 & L_2 s_3 & 0 \\ 0 & L_3 + L_2 c_3 & L_3 \\ -L_3 c_{23} & 0 & 0 \\ s_{23} & 0 & 0 \\ c_{23} & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix} \quad (2.41)$$

Le modèle cinématique final dans  $R_0$  est donné par :

$$\begin{bmatrix} V_0^3 \\ \omega_0^3 \end{bmatrix} = \begin{bmatrix} -L_3 c_{23} s_1 & L_2 c_1 (s_3 c_{23} - c_3 s_{23}) - L_3 c_1 s_{23} & -L_3 c_1 s_{23} \\ L_3 c_{23} c_1 & L_2 s_1 (s_3 c_{23} - c_3 s_{23}) - L_3 s_1 s_{23} & -L_3 s_1 s_{23} \\ 0 & L_2 (s_3 s_{23} + c_3 c_{23}) + L_3 c_{23} & L_3 c_{23} \\ 0 & s_1 & s_1 \\ 0 & -c_1 & -c_1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix} \quad (2.42)$$

### 2.3.6. Modèle cinématique inverse (MCI) [21]

L'objectif du MCI est de calculer, à partir d'une configuration  $q$  donnée, les vitesses articulaires  $\dot{q}$  qui assurent au repère terminal une vitesse opérationnelle  $\dot{X}$  imposée.

Pour obtenir le modèle cinématique inverse, on inverse le modèle cinématique direct en résolvant un système d'équations linéaires. La mise en œuvre peut être faite de façon analytique ou numérique. Les solutions analytiques réduisent le nombre d'opérations de façon remarquable par rapport aux solutions numériques. Mais, il faut traiter les cas singuliers distinctement. Les solutions numériques sont plus générales et traitent tous les cas de la même manière.

#### 2.3.6.1. Modèle cinématique inverse de notre réalisation

Pour ce faire, on peut utiliser la matrice  $J_{0(3 \times 3)}$  réduite, que l'on obtient en supprimant les lignes 3, 4 et 6 du MCD. En considérant la matrice jacobéenne réduite nous avons :

$$J_{0(3 \times 3)} = \begin{bmatrix} -L_3 c_{23} s_1 & L_2 c_1 (s_3 c_{23} - c_3 s_{23}) - L_3 c_1 s_{23} & -L_3 c_1 s_{23} \\ L_3 c_{23} c_1 & L_2 s_1 (s_3 c_{23} - c_3 s_{23}) - L_3 s_1 s_{23} & -L_3 s_1 s_{23} \\ 0 & -c_1 & -c_1 \end{bmatrix} \quad (2.43)$$

Son MCI est obtenu en calculant  $J_0^{-1(3 \times 3)}$  réduite avec le nouveau vecteur  $\dot{X}r = [v_x \ v_y \ v_y]^T$  :

$$\dot{\theta} = J_0^{-1(3 \times 3)} \dot{X}r \quad (2.44)$$

$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix} = \begin{bmatrix} -L_3 c_{23} s_1 & L_2 c_1 (s_3 c_{23} - c_3 s_{23}) - L_3 c_1 s_{23} & -L_3 c_1 s_{23} \\ L_3 c_{23} c_1 & L_2 s_1 (s_3 c_{23} - c_3 s_{23}) - L_3 s_1 s_{23} & -L_3 s_1 s_{23} \\ 0 & -c_1 & -c_1 \end{bmatrix}^{-1} \dot{X}_r \quad (2.45)$$

Le déterminant vaut :  $L_2 L_3 c_{23} c_1 (s_3 c_{23} - c_3 s_{23})$ .

Ce qui nous donne :

$$\begin{bmatrix} \dot{\theta}_1 \\ \dot{\theta}_2 \\ \dot{\theta}_3 \end{bmatrix} = \frac{1}{L_2 L_3 c_{23} c_1 (s_3 c_{23} - c_3 s_{23})} \times \begin{bmatrix} -L_2 s_1 c_1 (s_3 c_{23} - c_3 s_{23}) & L_2 c_1^2 (s_3 c_{23} - c_3 s_{23}) & 0 \\ L_3 c_{23} c_1^2 & L_3 s_1 c_1 c_{23} & -L_3^2 c_{23} s_{23} \\ -L_3 c_{23} c_1^2 & L_3 c_{23} (-L_3 s_{23} + L_2 (s_3 c_{23} - c_3 s_{23})) & L_3 c_{23} (L_3 s_{23} - L_2 (s_3 c_{23} - c_3 s_{23})) \end{bmatrix} \dot{X}_r \quad (2.46)$$

### 2.3.7. Modélisation dynamique [21]

Le modèle dynamique est la relation entre les couples (et/ou forces) appliqués aux actionneurs et aux positions, vitesses et accélérations articulaires. On représente le modèle dynamique par une relation de la forme :

$$\Gamma = F(q, \dot{q}, \ddot{q}, fe) \quad (2.47)$$

Avec,  $\Gamma$  : Vecteur des couples/forces des actionneurs, selon le type d'articulation (rotoïde ou prismatique).

$q$  : Vecteur des positions articulaires.

$\dot{q}$  : Vecteur des vitesses articulaires.

$\ddot{q}$  : Vecteur des accélérations articulaires

$fe$  : Vecteur représentant l'effort extérieur (forces et moments) qu'exerce le robot sur l'environnement.

Le modèle dynamique direct (MDD) exprime les accélérations articulaires en fonction des positions, des vitesses et des couples articulaires :

$$\ddot{q} = F(q, \dot{q}, \Gamma, fe) \quad (2.48)$$

### 2.3.7.1. Modélisation dynamique de notre réalisation

Plusieurs approches ont été proposées afin d'obtenir le modèle dynamique des robots. Les plus souvent utilisés dans la robotique sont : le formalisme de Lagrange et le formalisme de Newton-Euler.

L'approche de Newton-Euler est basée sur les forces et les moments qui agissent entre les liens. Cette formulation peut être considérée comme une approche basée sur l'équilibre des forces.

La formulation de Lagrange, quant à elle, est une approche basée sur l'énergie.

Dans cette étude on présente le formalisme de Lagrange. Ses équations sont généralement écrites sous la forme :

$$Q = \frac{d}{dt} \frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} \quad (2.49)$$

Avec : Q : Force généralisée,

q : Vecteur des variables articulaires, (se compose de l'angle d'articulation  $\theta_i$  et du déplacement  $d_i$ )

L : Lagrangien du système,

K : Energie cinétique,

P : Energie potentielle.

$$L = K - P \quad (2.50)$$

Comme on a trois ddl, il y a donc trois coordonnées généralisées ( $q_i$ ) dans (2.49), conduisant à 3 équations. Afin de calculer l'énergie cinétique du robot, on somme l'énergie cinétique de chaque lien. Ainsi, l'énergie cinétique totale devient :

$$k(\theta, \dot{\theta}) = \sum_{i=1}^3 k_i(\theta, \dot{\theta}) = \frac{1}{2} \dot{\theta}^T M(\theta) \dot{\theta} \quad (2.51)$$

Avec,  $M(q)$  : matrice d'inertie de dimension  $(n \times n)$ .

Les expressions de la forme de k sont appelées forme quadratique. L'énergie cinétique doit toujours être positive, c'est pourquoi la matrice d'inertie du manipulateur doit être une matrice définie positive.

On calcule l'énergie potentielle du robot :

$$P(\theta) = \sum_{i=1}^3 P_i = \sum_{i=1}^3 m_i g h_i(\theta) \quad (2.52)$$

Avec,  $h_i$  : la hauteur du centre de masse de la  $i^{\text{ème}}$  liaison.

Le Lagrangien devient donc :

$$L(\theta, \dot{\theta}) = \frac{1}{2} \dot{\theta}^T M(\theta) \dot{\theta} - P(\theta) \quad (2.53)$$

A des fins de contrôle, il est courant et plus pratique de réécrire le modèle dynamique Euler – Lagrangien du robot manipulateur en forme compacte ou matricielle comme suit :

$$\Gamma = M(\theta)\ddot{\theta} + C(\theta, \dot{\theta})\dot{\theta} + G(\theta) \quad (2.54)$$

Avec,  $\Gamma$  : Vecteur des couples d'actionneur,

$G(\theta)$  : Vecteur de forces de gravité,

$C(\theta, \dot{\theta})$  : Matrice appelée courbe de Coriolis

La courbe de Coriolis est donnée par :

$$C(\theta, \dot{\theta}) = \frac{1}{2} \left( \frac{\partial M_{ij}}{\partial \theta_k} + \frac{\partial M_{ik}}{\partial \theta_j} - \frac{\partial M_{kj}}{\partial \theta_i} \right) \dot{\theta}_k \quad (2.55)$$

Et

$$M(\theta) = \begin{bmatrix} M(1,1) & M(1,2) & M(1,3) \\ M(2,1) & M(2,2) & M(2,3) \\ M(3,1) & M(3,2) & M(3,3) \end{bmatrix} \quad (2.56)$$

$$M(1,1) = \frac{1}{2} m_1 r^2 + \frac{1}{3} m_2 l_2^2 \cos^2 \theta_2 + \frac{1}{3} m_3 l_3^2 \cos(\theta_2 + \theta_3)^2 + m_3 l_2^2 \cos^2 \theta_2 + m_3 l_2 l_3 \cos(\theta_2 + \theta_3) \cos \theta_2 \quad (2.57)$$

$$M(1,2) = 0 \quad (2.58)$$

$$M(1,3) = 0 \quad (2.59)$$

$$M(2,1) = 0 \quad (2.60)$$

$$M(2,2) = \frac{1}{3} m_2 l_2^2 + \frac{1}{3} m_3 l_3^2 + m_3 l_2^2 + m_3 l_2 l_3 \cos \theta_3 \quad (2.61)$$

$$M(2,3) = \frac{1}{3} m_3 l_3^2 + m_3 l_2^2 + \frac{1}{3} m_3 l_2 l_3 \cos \theta_3 \quad (2.62)$$

$$M(3,1) = 0 \quad (2.63)$$

$$M(3,2) = \frac{1}{3} m_3 l_3^2 + m_3 l_2^2 + \frac{1}{3} m_3 l_2 l_3 \cos \theta_3 \quad (2.64)$$

$$M(3,3) = \frac{1}{3} m_3 l_3^2 \quad (2.65)$$

Le Coriolis et les forces centrifuges sont calculés directement à partir de l'équation (2.55) :

$$C(\theta, \dot{\theta}) = \begin{bmatrix} c(1,1) \\ c(2,1) \\ c(3,1) \end{bmatrix} \quad (2.66)$$

$$C(1,1) = \left[ -\frac{4}{3} m_2 l_2^2 \sin 2\theta_2 - \frac{1}{3} m_3 l_3^2 \sin 2(\theta_2 + \theta_3) - m_3 l_2 l_3 \sin(2\theta_2 + \theta_3) \right] \dot{\theta}_2 \dot{\theta}_1 \quad (2.67)$$

$$+ \left[ -\frac{1}{3} m_3 l_3^2 \sin 2(\theta_2 + \theta_3) - m_3 l_2 l_3 \cos \theta_2 \sin(\theta_2 + \theta_3) \right] \dot{\theta}_3 \dot{\theta}_1$$

$$C(2,1) = [-m_3 l_2 l_3 \sin \theta_3] \dot{\theta}_2 \dot{\theta}_3 + \left[ -\frac{1}{2} m_3 l_2 l_3 \sin \theta_3 \right] \dot{\theta}_3^2 + \quad (2.68)$$

$$\left[ \frac{1}{6} m_2 l_2^2 \sin 2\theta_2 + \frac{1}{6} m_3 l_3^2 \sin 2(\theta_2 + \theta_3) + \frac{1}{2} m_3 l_2^2 \sin 2\theta_2 + \frac{1}{2} m_3 l_2 l_3 \sin(2\theta_2 + \theta_3) \right] \dot{\theta}_1^2$$

$$C(3,1) = \left[ \frac{1}{2} m_3 l_2 l_3 \sin \theta_3 \right] \dot{\theta}_2^2 + \left[ \frac{1}{6} m_3 l_3^2 \sin 2(\theta_2 + \theta_3) + m_3 l_2 l_3 \cos \theta_2 \sin(\theta_2 + \theta_3) \right] \dot{\theta}_1^2 \quad (2.69)$$

Le vecteur des forces de gravité du robot manipulateur peut être dérivé de l'équation :

$$G(\theta) = \frac{\partial P}{\partial \theta_i} \quad (2.70)$$

On aura donc :

$$G(1,1) = 0 \quad (2.71)$$

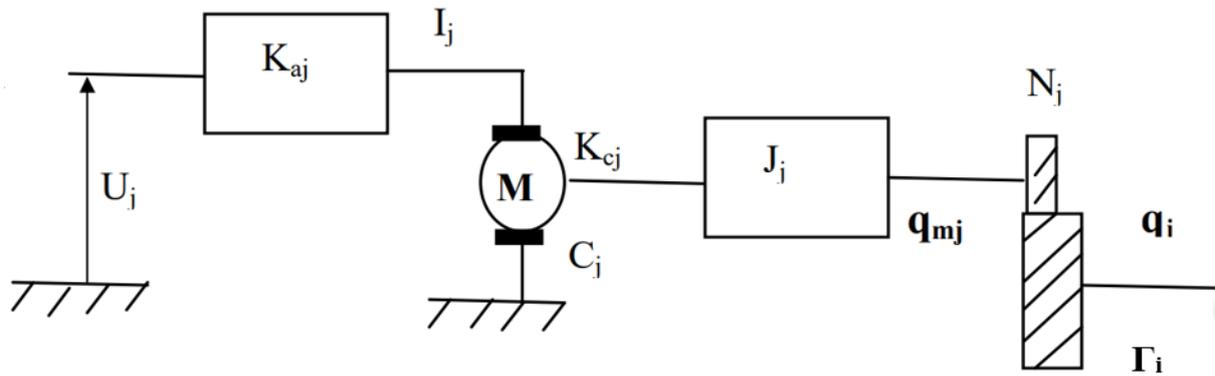
$$G(2,1) = \frac{1}{2} m_2 g l_2 \cos \theta_2 + \frac{1}{2} m_3 g l_3 \cos(\theta_2 + \theta_3) + m_3 g l_2 \cos \theta_2 \quad (2.72)$$

$$G(3,1) = \frac{1}{2} m_3 g l_3 \cos(\theta_2 + \theta_3) \quad (2.73)$$

### 2.3.8. Modélisation des actionneurs électriques [21]

Les servomoteurs, comme évoqué précédemment, sont des MCC qui sont asservis en position : ce sont des actionneurs électriques. C'est donc pour cela que l'on va consacrer cette partie à leur modélisation.

Les actionneurs électriques produisent les couples nécessaires aux articulations. Ces couples sont déterminés à partir des équations modélisant ces derniers.



**Fig.2.17.** Schéma fonctionnel d'un actionneur [21]

On peut écrire :

$$I_j = K_{aj} \cdot u_j \quad (2.74)$$

Avec,  $I_j$  : Courant de l'induit du moteur  $j$

$u_j$  : Tension d'entrée

$K_{aj}$  : Constante proportionnelle

Le courant, traversant l'induit du moteur, produit un couple moteur :

$$C_j = K_{cj} \cdot I_j = K_{cj} \cdot K_{aj} \cdot u_j = J_j \cdot \ddot{q}_{mj} + \frac{\Gamma_i}{N_j} \quad (2.75)$$

Avec,  $\ddot{q}_{mj} = N_j \cdot \ddot{q}_j$  et  $\dot{q}_{mj} = N_j \cdot \dot{q}_j$

$N_j$  : Rapport de réduction,

$K_{cj}$  : Coefficient du couple moteur  $j$ ,

$J_j$  : Inertie propre du moteur  $j$  ramené à l'arbre du moteur plus celle des organes de transmission,

$\dot{q}_{mj}$  : Vitesse angulaire produite par le moteur  $j$ ,

$\dot{q}_j$  : Vitesse angulaire transmise au corps  $j$  à travers le réducteur,

$\Gamma_i$  : Couple transmis à la  $j^{\text{ème}}$  articulation.

On a alors :

$$N_j \cdot K_{c_j} \cdot K_{a_j} \cdot u_j = (N_j^2 \cdot J_j) \cdot \ddot{q}_j + \Gamma_i = I_{a_j} \cdot \ddot{q}_j + \Gamma_i \quad (2.76)$$

En utilisant la forme matricielle on obtient :

$$N \cdot K_c \cdot K_a \cdot U = I_a \cdot \ddot{q} + \Gamma \quad (2.77)$$

Après avoir obtenu la modélisation des actionneurs électriques, on peut obtenir le modèle dynamique complet du robot, tel que :

$$\Gamma = M(q) \cdot \ddot{q} + C(q, \dot{q}) + G(q) \quad (2.78)$$

En substituant (2.78) dans (2.77), on obtient :

$$N \cdot K_c \cdot K_a \cdot U = [I_a + M(q)] \cdot \ddot{q} + C(q, \dot{q}) + G(q) \quad (2.79)$$

Avec,  $M(q)$  : Matrice d'inertie (nxn) symétrique, liée à l'accélération,

$C(q, \dot{q})$  : Vecteur des forces centrifuges et Coriolis non linéaire lié à la vitesse,

$G(q)$  : Vecteur des forces de charges gravitationnelles.

## 2.4. Conclusion

Dans ce second chapitre, on a commencé par décrire les différents organes constituant notre bras manipulateur. Par la suite, on a donné une modélisation aussi simpliste que possible de son modèle géométrique, cinématique et dynamique. Enfin, on a terminé par la modélisation des actionneurs électriques afin d'obtenir le modèle dynamique complet généralisé d'un bras manipulateur.

On va dès à présent débiter la partie concernant la commande de notre réalisation.

**CHAPITRE 3 :**

**COMMANDE  
ET  
SIMULATION  
DU BRAS  
MANIPULATEUR**

### 3.1. Introduction

Pour tout système de commande, la connaissance de l'ensemble des points de fonctionnement est primordiale. La commande d'un bras manipulateur nécessite la commande de ses articulations une à une, afin de réaliser une tâche bien déterminée.

Ce chapitre est dédié à l'étude des différentes commandes appliquées à notre robot manipulateur. Tout d'abord, on présente le microcontrôleur utilisé ainsi que le Shield conçu pour contrôler ce dernier. Ensuite, on va illustrer les différentes commandes et la procédure suivie pour l'emploi de celles-ci dans notre projet.

### 3.2. Présentation de la carte Arduino

La carte Arduino est un microcontrôleur open source, qui a été adopté par la communauté des concepteurs. Il s'agit d'un système numérique à processeur entouré d'une mémoire et des interfaces avec le monde extérieur.

Ces microcontrôleurs ont des performances réduites, mais sont de faible taille et consomment peu d'énergie, les rendant indispensables dans toute solution d'électronique embarquée (voiture, porte de garage, robots, ...). La carte Arduino n'est pas le microcontrôleur le plus puissant, mais son architecture a été publiée en open-source [30].

#### 3.2.1 La carte Arduino UNO

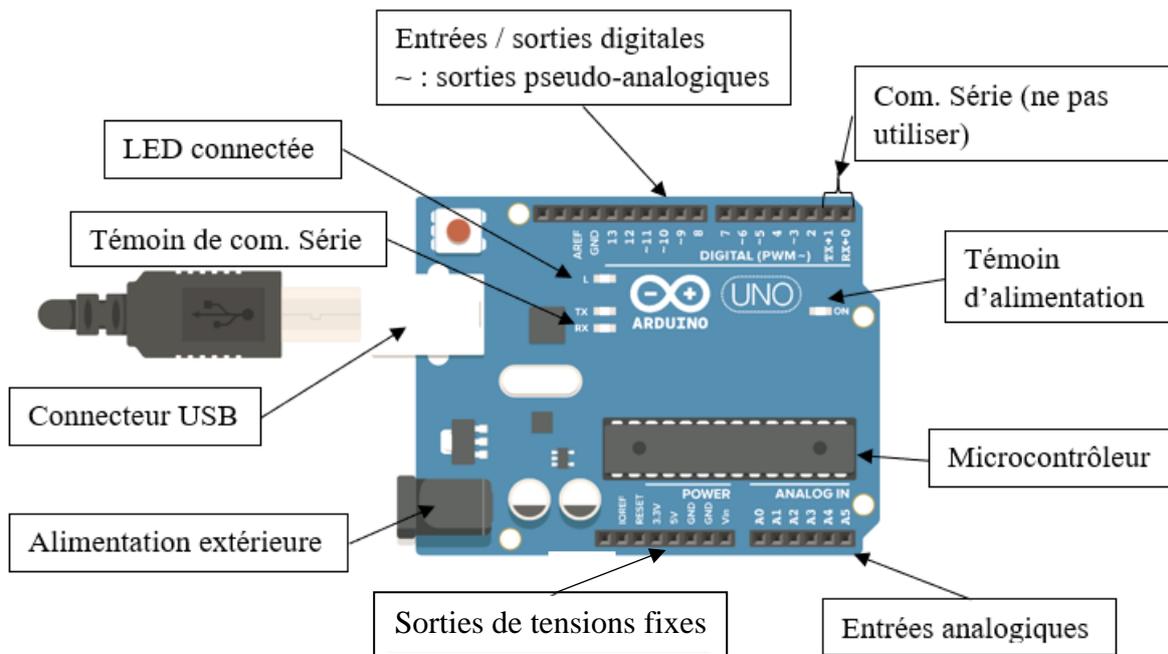
L'Arduino UNO est une carte de développement avec un microcontrôleur Atmel ATMEGA328P qui est couramment utilisé par le grand public.

L'avantage de cette carte est d'éviter d'utiliser des composants nus. Les composants périphériques nécessaires sont déjà présents, il n'y a donc pas besoin de les souder. De plus, l'ATMEGA est assez complet et facile à utiliser. Enfin, elle est également peu chère et connaît un grand succès.

Le microcontrôleur contient, notamment [31] :

- Un CPU (Central Processing Unit)
- Une mémoire RAM
- Une mémoire ROM
- Des entrées-sorties

Le brochage Atmega 328 P 32-pin TQFP est donné dans l'Annexe A.



**Fig.3.1.** La carte Arduino UNO

### 3.2.2. Les caractéristiques de la carte Arduino UNO

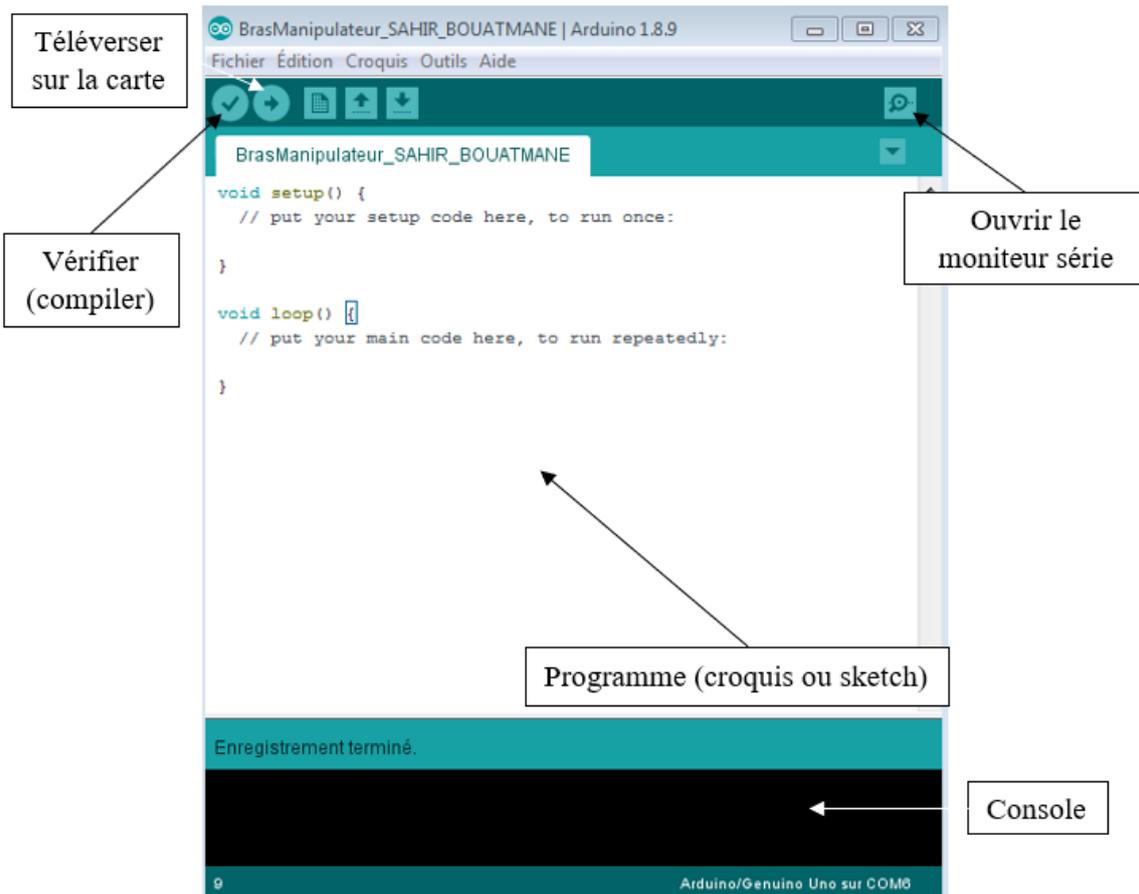
**Tab.3.1.** Caractéristiques de l'Arduino UNO [32].

Microcontrôleur	ATmega328
Tension de fonctionnement	5V
Tension d'alimentation	7-12V
Tension d'alimentation (limites)	6-20V
Broches E/S numériques	14 (dont 6 fournissent une sortie PWM)
Broches d'entrées analogiques	6
Intensité maxi disponible pour la sortie 3.3V	50 mA
Intensité maxi disponible pour la sortie 5V	40 mA
Mémoire programme Flash	32 KB (dont 0.5 KB utilisés par le chargeur de démarrage)
Mémoire SRAM (mémoire volatile)	2 KB
Mémoire EEPROM (mémoire non volatile)	1 KB
Vitesse d'horloge	16 MHz

### 3.2.3. L'environnement de programmation Arduino (IDE)

Le logiciel Arduino est un environnement de développement (IDE en anglais) open source et gratuit, téléchargeable sur le site officiel Arduino.

L'IDE Arduino permet d'éditer un programme, de le compiler dans le langage « machine » de la carte, de le téléverser dans la mémoire de l'Arduino et enfin de communiquer avec cette dernière grâce au terminal.



**Fig.3.2.** Environnement de développement Arduino

### 3.3. Réalisation de la carte de commande

Dans le but de commander notre bras manipulateur, on a décidé d'utiliser la carte Arduino UNO que l'on a introduit précédemment. On a également utilisé une deuxième carte qui vient se positionner directement sur les PINs de l'Arduino, sur laquelle on a réservé des places pour l'utilisation des potentiomètres afin qu'ils assurent la commande manuelle. On a aussi attribué des places pour l'alimentation, pour la commande des servomoteurs et pour la connexion du récepteur Bluetooth. Voici, ci-après, les étapes de soudure des composants sur la carte électronique.

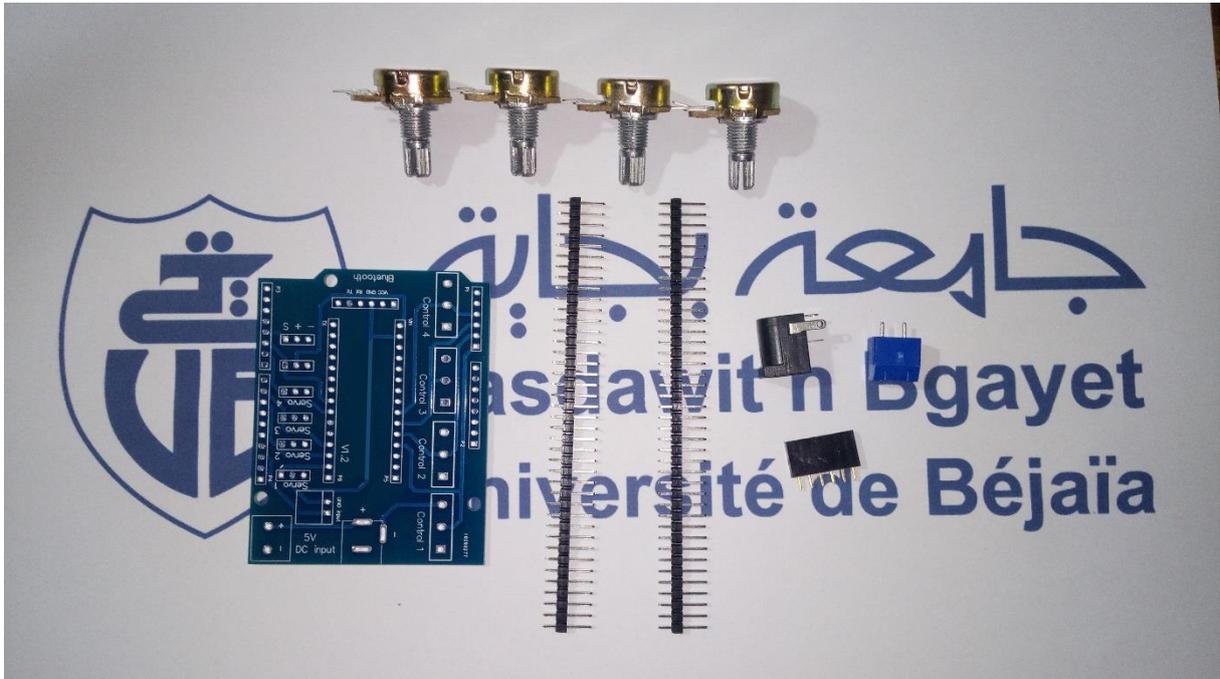


Fig.3.3. Les composants constituant la carte de commande manuelle (non soudés)

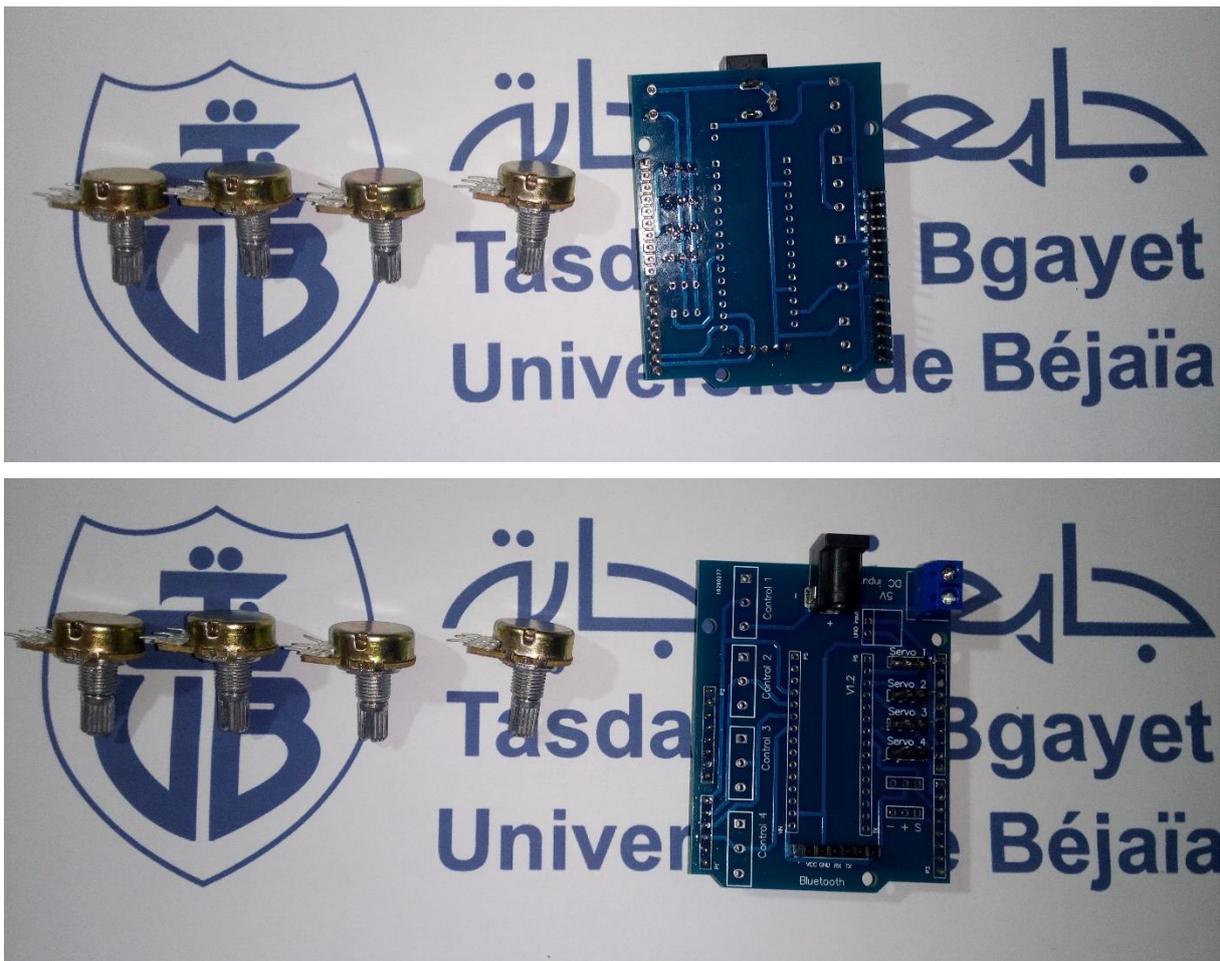
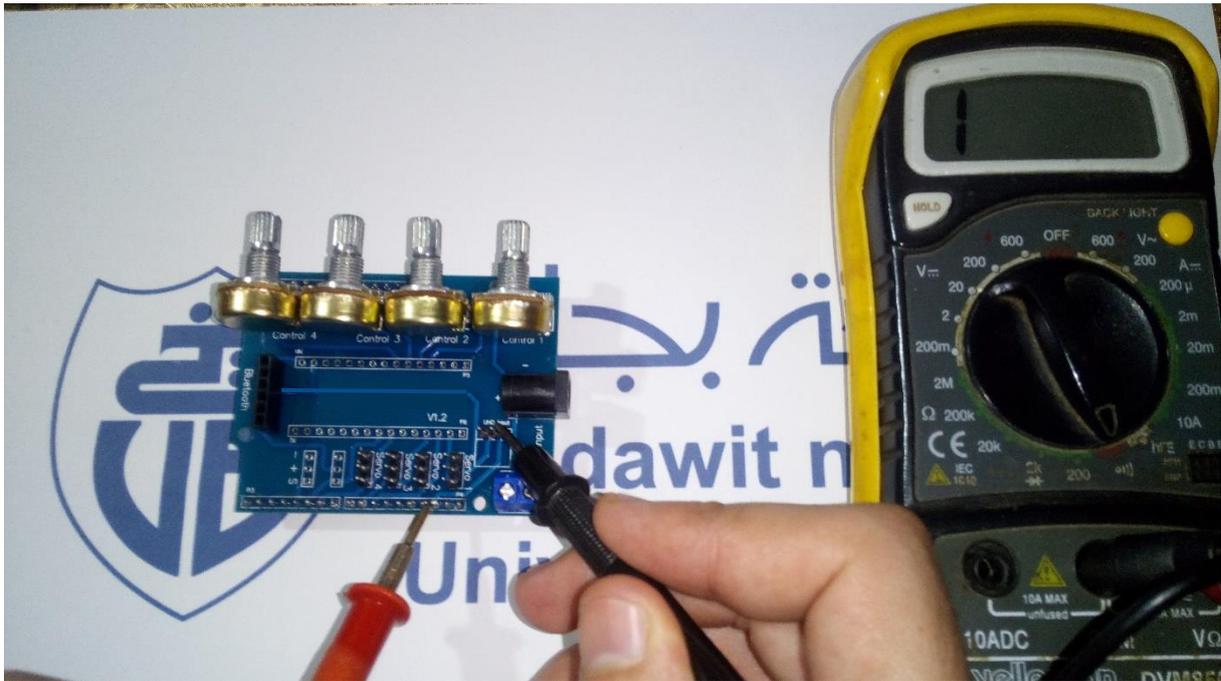
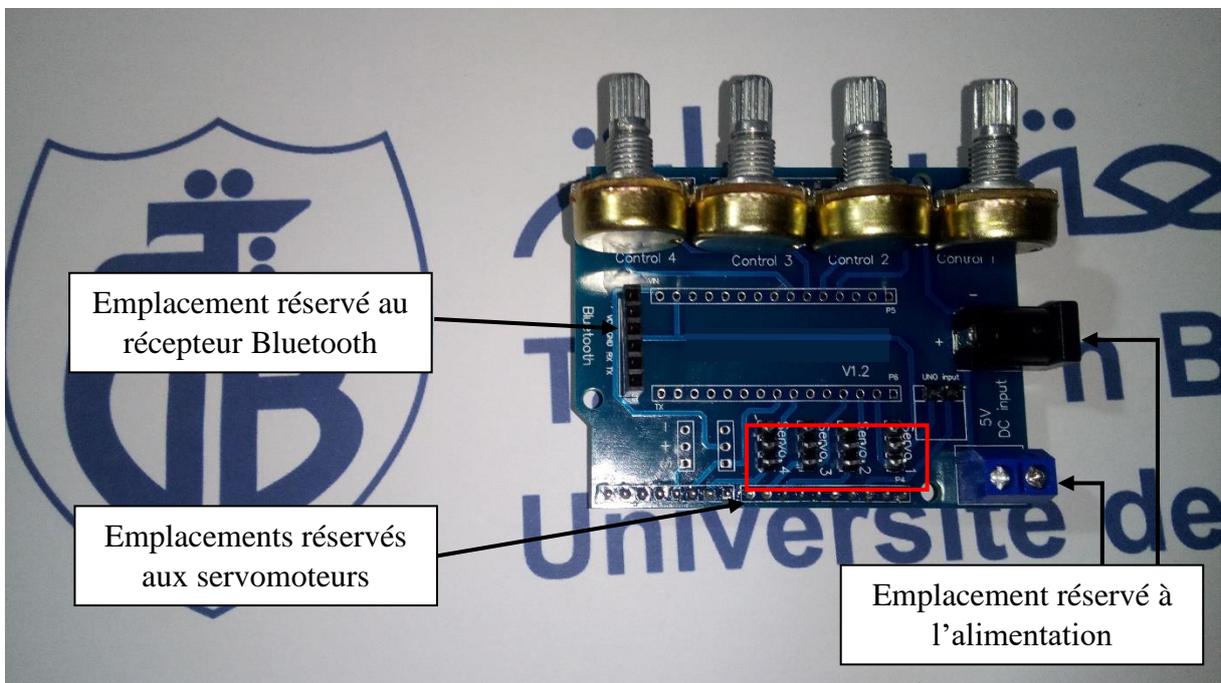


Fig.3.4. Montage des pins et sources d'alimentation sur la carte de commande manuelle

Une fois la carte finalisée avec l'installation des potentiomètres, on passe à l'étape de test de continuité à l'aide d'un multimètre, dans le but de vérifier le bon contact du circuit et des composants.



**Fig.3.5.** Test de continuité à l'aide d'un multimètre de la carte de commande manuelle.



**Fig.3.6.** Carte de commande manuelle finalisée.

### 3.4. Types de commande du bras manipulateur

Pour commander les quatre servomoteurs du bras manipulateur, on a réalisé trois types de commande : la commande manuelle, la commande via une application mobile et la commande par les angles des organes du robot, préprogrammés directement via l'environnement de développement IDE. Le dispositif assurant la tâche de la commande manuelle est le potentiomètre rotatif.

#### 3.4.1. La commande manuelle

Afin d'établir une commande manuelle au bras manipulateur, on a créé un programme qui permet de le commander par le biais de potentiomètres.

La connexion entre la carte électronique et l'environnement de programmation se fait via une liaison USB. On commence, tout d'abord, par déclarer la bibliothèque `<Servo.h>`, qui est une fonction prédéfinie qui nous permet d'utiliser les servomoteurs dans notre programme.

On a utilisé la fonction `analogRead()` qui permet de lire les entrées analogiques. Elle est définie comme suit :

```
valPot1 = analogRead(pot1);
```

On a également utilisé la fonction `map()` qui permet la conversion (ou le redimensionnement) des valeurs lues pour les adapter aux valeurs des servomoteurs (0 : 180).

```
valPot1 = map (valPot1, 0, 1023, 0, 180);
```

Le pseudocode suivant présente la démarche globale suivie pour l'élaboration du programme. Le programme détaillé de cette commande est donné dans l'annexe B.

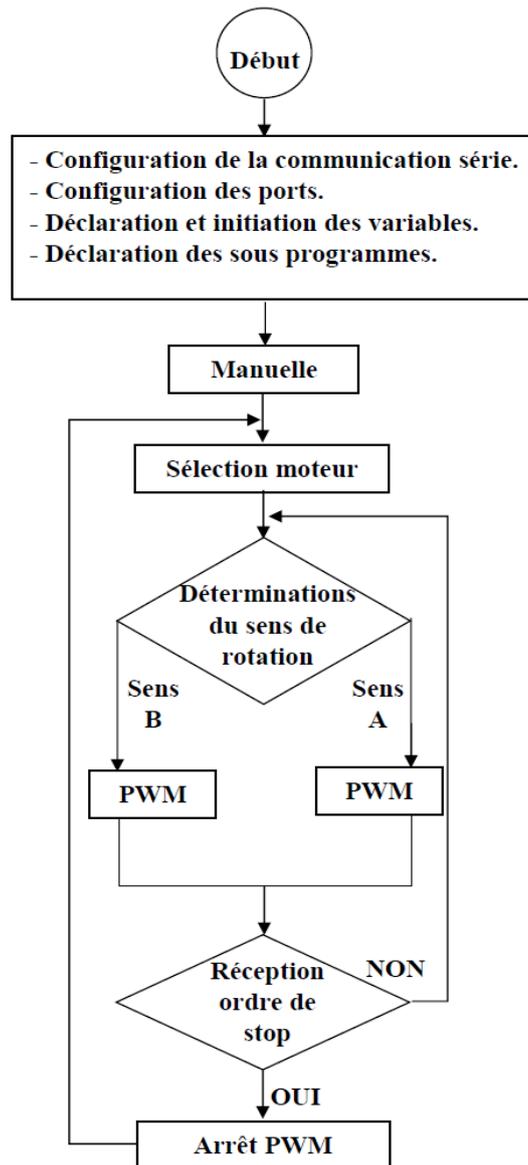
---

#### Algorithme 1 : Commande manuelle

---

1. Faire appel à la bibliothèque Servo
  2. Créer des constantes utilisables par la bibliothèque Servo.
  3. Déclarer les potentiomètres sur les PINs analogiques.
  4. Créer des variables indiquant les valeurs des potentiomètres.
  5. **Void setup() {**
  6. Attacher les servomoteurs aux PINs }
  7. **Void loop() {**
  8. Lire les valeurs des potentiomètres qui sont comprises entre 0 et 1023
  9. Adapter les valeurs lues de chaque potentiomètre, pour permettre d'aller de 0 à 180 degrés.
  10. Positionner les servomoteurs en fonction des valeurs des potentiomètres.
  11. Créer un délais de 15 millisecondes avant de refaire la boucle (loop) }
-

Le fonctionnement de la commande manuelle se résume dans l’organigramme ci-dessous :



**Fig.3.7.** Logigramme des phases explicatives de la commande manuelle

### 3.4.2. La commande sans fil

Afin d’établir une commande sans fil qui nous donnera un contrôle à distance pour notre réalisation, on a utilisé un récepteur Bluetooth. On a également conçu une application mobile Android grâce à l’outil de développement créé par Google App Inventor pour assurer la communication entre l’Arduino et l’interface réalisée.

#### 3.4.2.1. Le concept d’App Inventor

La programmation s’accomplit en ligne à l’aide d’un langage de bloc d’instructions et sous un environnement logiciel au choix (Mac, Linux ou Windows). Les informations sont

stockées sur des serveurs distants. La seule contrainte à laquelle nous sommes confrontés est de devoir posséder un compte Gmail pour pouvoir y accéder.

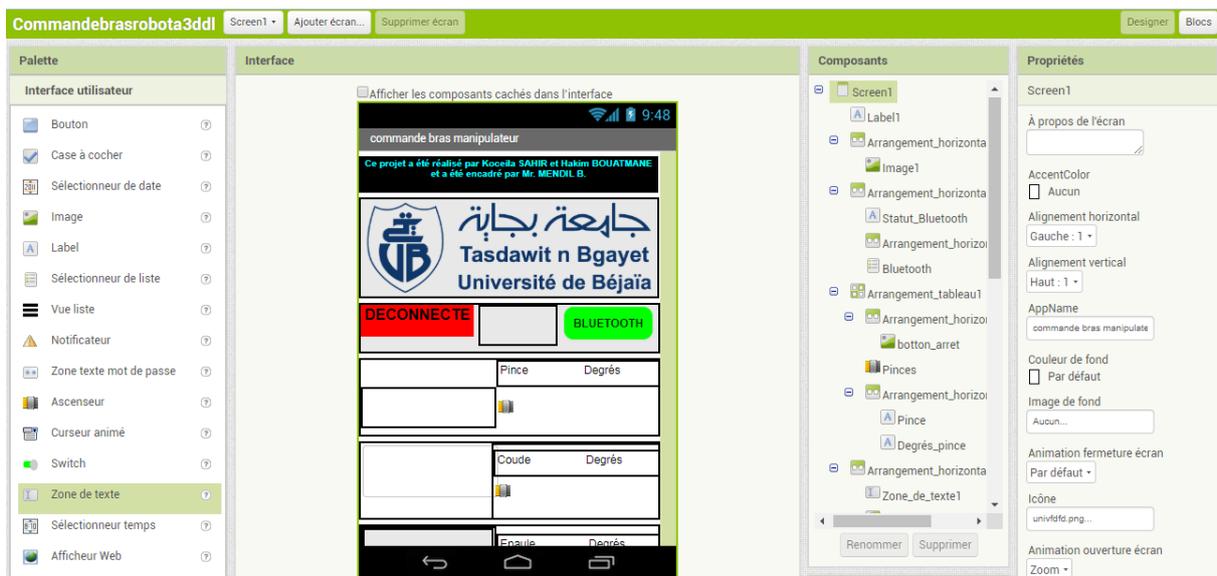


**Fig.3.8.** App Inventor LOGO

### 3.4.2.2. Réalisation de l'application Android

La réalisation de l'application sous AppInventor se divise en deux parties : la partie design pour le graphique et la partie bloc pour la programmation.

Dans la partie design, on dispose de divers blocs graphiques comme des boutons, des champs de texte, des images, des curseurs, mais aussi des layouts qui servent à délimiter les zones de notre application. Il existe également quelques autres fonctionnalités utiles de multimédia, de GPS, et de communication Bluetooth.

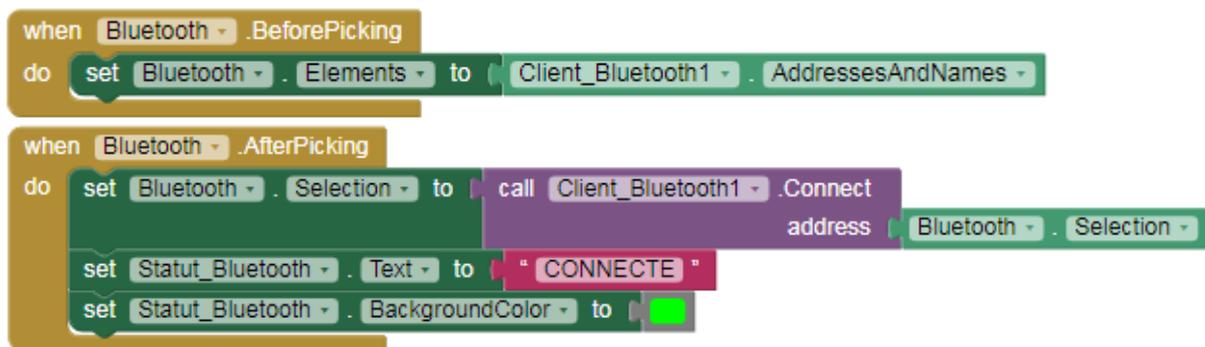


**Fig.3.9.** Partie design de notre application sous AppInventor

Dans la partie bloc, on retrouve toutes les fonctions mathématiques, logiques, ou évènementielles, ce rapportant aux objets utilisés dans la partie graphique. Cela nous permet de tirer profit d'une simplicité de programmation orientée objet sous la forme de blocs préprogrammés et qu'ils ne restent plus qu'à organiser en fonction du programme souhaité.

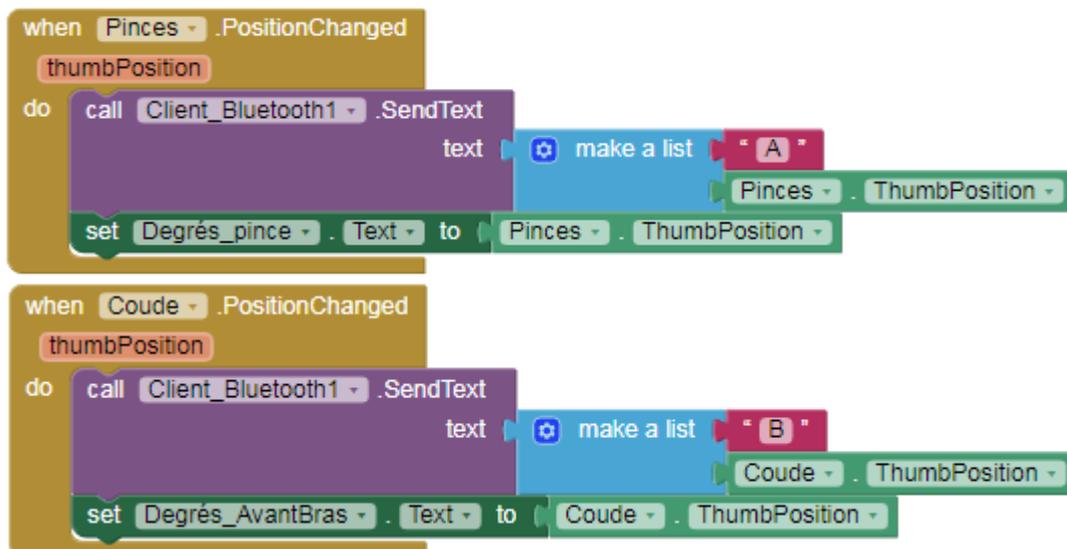
Lorsque ces outils sont utilisés correctement, on peut réaliser une application simple et efficace, avec un design propre qui répond à nos besoins. Dans ce qui suit, on va expliquer le fonctionnement des blocs de programme et leurs utilités dans le programme Arduino pour le contrôle du bras.

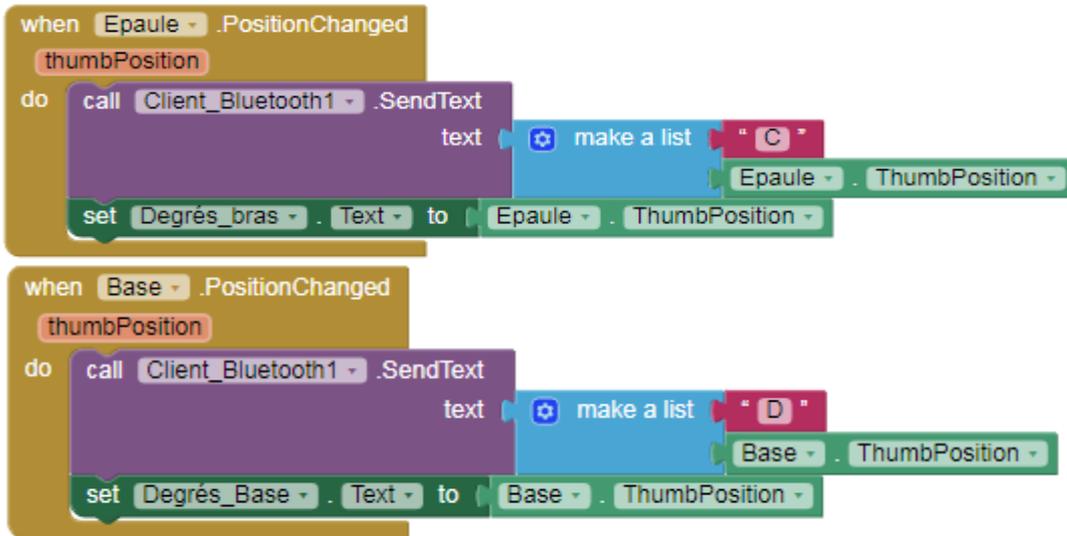
Pour commencer, on définit les éléments de l'objet Bluetooth de type *BluetoothListe* (comme les périphériques Bluetooth disponibles). Ensuite on établit la connexion entre le périphérique et l'application.



**Fig.3.10.** Définition des éléments de l'objet Bluetooth

Ensuite, on définit les blocs de contrôle des curseurs pour les variations d'angles des servomoteurs :

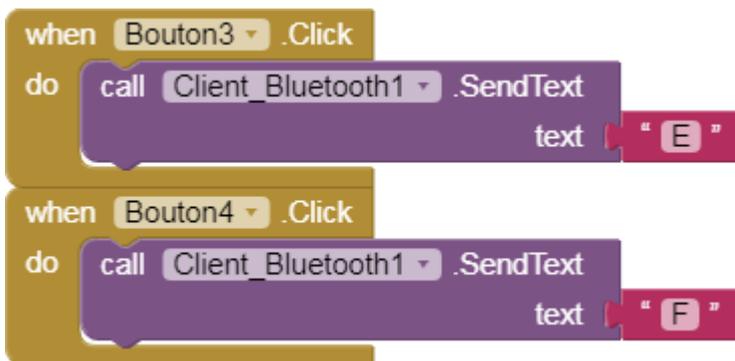




**Fig.3.11.** Blocs de contrôle des curseurs

Ce sont les blocs qui contrôlent les quatre servomoteurs du bras manipulateur (pince, coude, épaule et base). Si on change la position du curseur, en utilisant la fonction *Bluetooth.SendText*, on envoie un texte à l'Arduino. Ce texte se compose d'un préfixe qui indique quel curseur a été modifié (le préfixe « A » pour la pince jusqu'au préfixe « D » pour la base) ainsi que la valeur actuelle du curseur qui est *thumbPosition*.

Enfin on définit les blocs des deux boutons. Ces derniers envoient un texte à l'Arduino. Ce texte se compose d'un préfixe qui indique quel cycle est à exécuter.

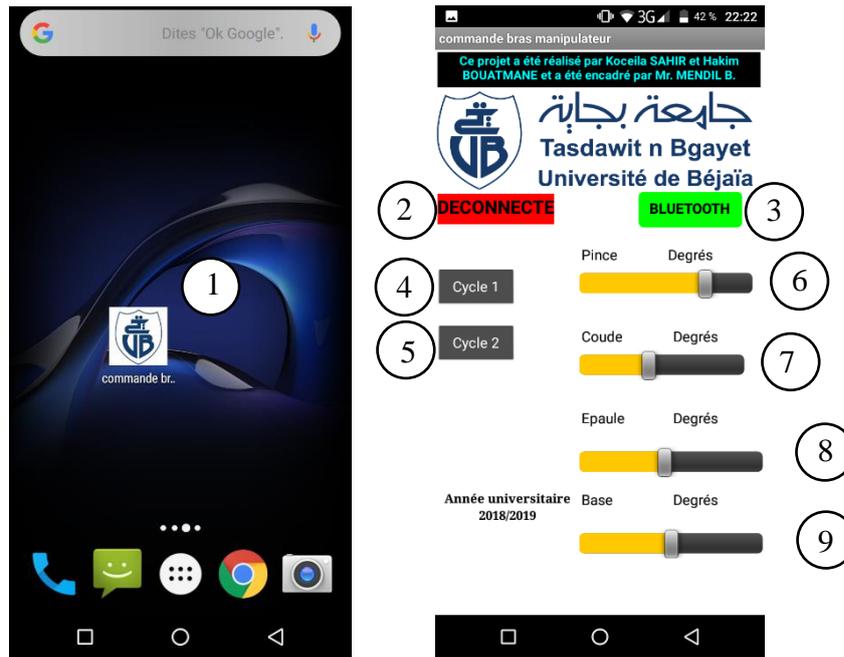


**Fig.3.12.** Blocs des deux boutons

Les deux boutons permettent d'exécuter une série d'instructions différentes. Celles-ci forment un cycle automatique, et peuvent être modifiés selon le besoin.

### 3.4.2.3. L'interface finale de notre application

Après le design et la programmation par bloc, on obtient l'application finale :

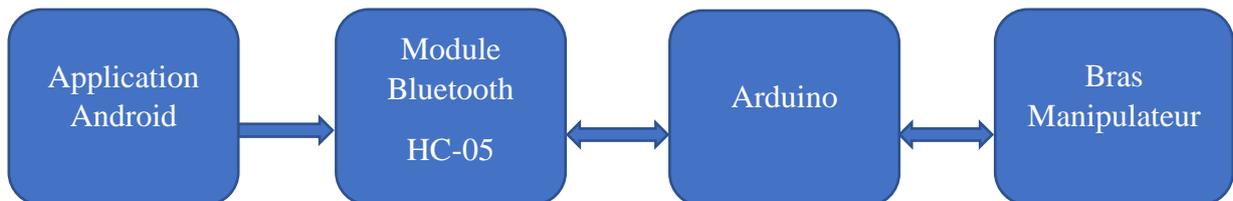


**Fig.3.13.** Interface finale de l'application

- |  |                       |
|--|-----------------------|
| 1 : Icône de l'application                         | 6 : Angle de la pince |
| 2 : Etat du bluetooth (connecté ou déconnecté)     | 7 : Angle du coude    |
| 3 : Connexion au Bluetooth                         | 8 : Angle de l'épaule |
| 4 : 1 <sup>er</sup> cycle d'exécution automatique  | 9 : Angle de la base  |
| 5 : 2 <sup>ème</sup> cycle d'exécution automatique |                       |

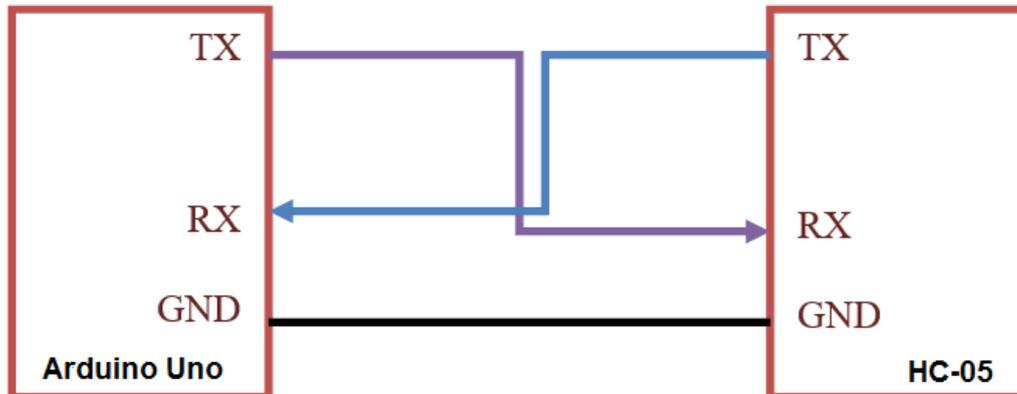
### 3.4.2.4. Communication de l'application avec l'Arduino

Après l'installation de notre application Android et après avoir établie la connexion avec le module Bluetooth, l'application envoie une information d'état au Bluetooth et qui lui-même, par la suite, la transmettra à l'Arduino à travers une liaison série (**Fig.3.14**).



**Fig.3.14.** Schéma synoptique simplifié du système

Afin de gérer la communication avec le module HC-05, on connecte les broches Rx et Tx de la carte aux broches d'émission/réception de notre module Bluetooth. La broche Rx du module est branchée sur la broche Tx de l'Arduino et inversement.



**Fig.3.15.** Schéma de la communication avec le module Bluetooth HC-05

Ainsi, chaque fois qu'un caractère/texte sera reçu, il sera envoyé sur la voie série de la carte de contrôle. Pour faire le lien avec l'application, on doit procéder par programmation en C++ qui permet de traduire les chaînes de caractères reçues via l'application en des informations pour les actionneurs du bras manipulateur.

Dans le programme, après avoir fait l'initialisation des servomoteurs et du module Bluetooth, on place le bras du robot dans sa position initiale en utilisant la fonction *write()* qui permet de déplacer simplement le servomoteur dans n'importe quelle position désirée.

On organise, ensuite, notre programme en un ensemble de fonctions d'exécution pour chaque servomoteur (moteur 1, moteur 2, moteur 3 et moteur 4), auquel on fait appel grâce à l'outil conditionnel « if », qui teste à chaque fois le préfixe envoyé par l'application et qui indique la position actuelle et le curseur modifié du servomoteur sélectionné. Le sous-programme est défini comme suit :

```
void loop() {
  if (Serial.available()){
    a = Serial.read();

    if(a=='A'){
      motor1();
    }

    if(a=='B'){
      motor2();
    }
  }
}
```

```

if(a=='C'){
  motor3();
}
if(a=='D'){
  motor4();
}

```

La fonction *Serial.available()* permet d'obtenir le nombre d'octets (caractères) disponibles pour la lecture à partir du port série. La fonction *serial.read()* nous permet de lire ces dernières. Chaque fonction doit être programmée de manière à convertir la chaîne de caractères reçue en un entier qui désignera l'angle de déplacement du servomoteur. Le sous-programme de la première fonction est donné ci-après :

```

void motor1(){
  delay(10);
  while (Serial.available()) {

  char b = Serial.read();
  readString += b;
  }
  if (readString.length() >0)
  {
    Serial.println(readString.toInt());
    servoP.write(readString.toInt());
    readString="";
  }
}

```

La fonction *Serial.println(readString.toInt())* nous permet la conversion de la chaîne de caractères en un entier. La fonction *write((readString.toInt()))* nous permet le déplacement du servomoteur vers l'entier (angle) obtenu après conversion.

Pour les deux boutons, on ordonne notre programme en deux fonctions d'exécution (cycle 1 et cycle 2), à qui on fait appel (une fois le bouton actionné) avec l'outil conditionnel « if ». Chaque cycle est créé selon une trajectoire choisie. Le sous-programme est défini comme suit :

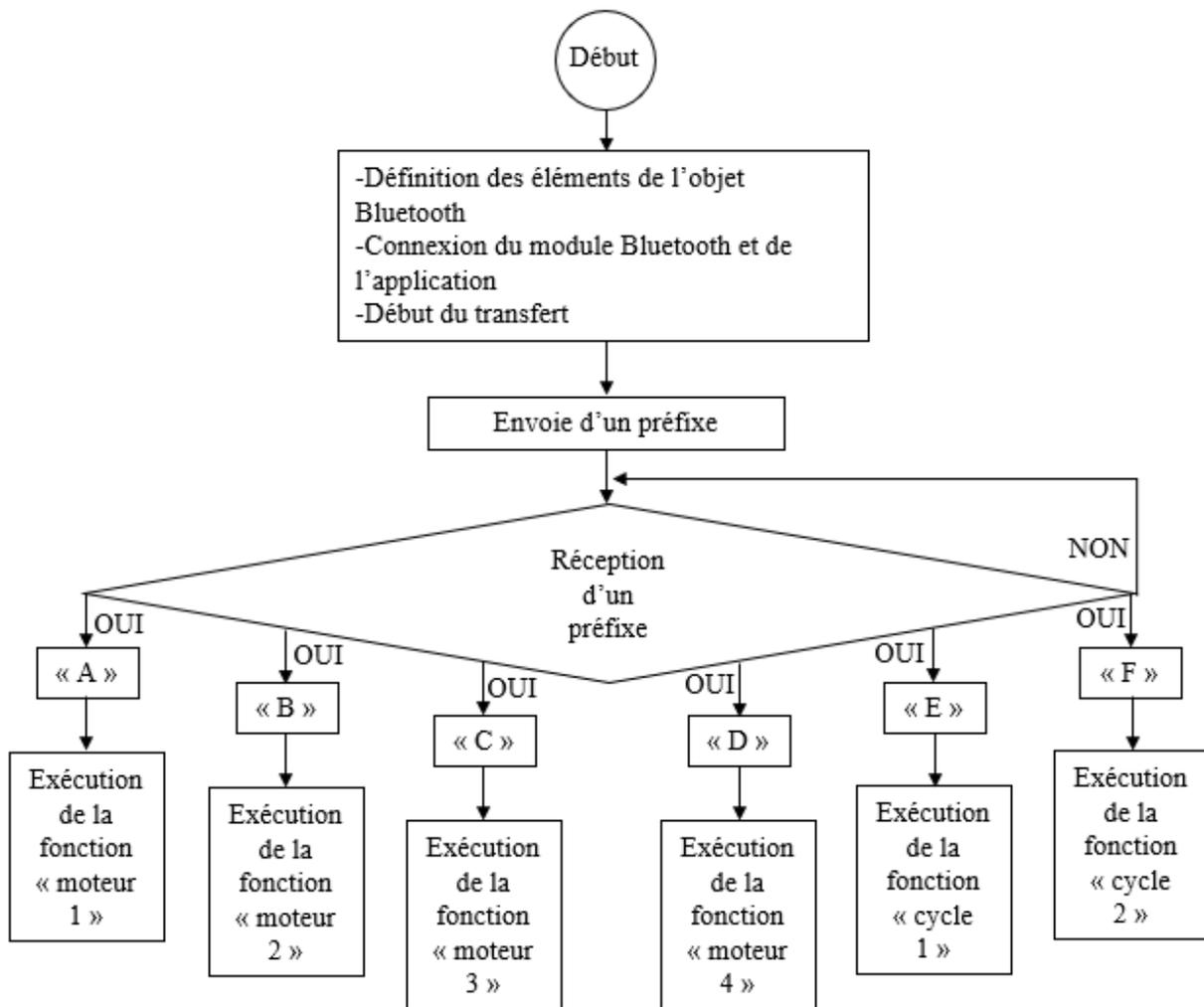
```

if(a=='E'){
  cyle1();
}
if(a=='F'){
  cyle2();
}

```

On retrouve, dans chacune de ses deux fonctions, des instructions nous menant vers la trajectoire choisie. Le programme détaillé utilisé pour la commande de notre bras manipulateur

à l'aide de l'application conçue est donné dans l'annexe C. Le logigramme de **Fig.3.16** résume la commande sans fil.



**Fig.3.16.** Logigramme résumant la commande sans fil

### 3.4.3 Commande par angles préprogrammés

La commande par angles préprogrammés est un programme qui va nous permettre de suivre directement des trajectoires désirées. Les différentes positions du bras ont été enregistrées directement dans un tableau par l'amorçage d'un premier bouton pour chaque variation des potentiomètres qui eux, désigne des angles pour les servomoteurs, ce qui nous fournit une séquence d'enregistrement et nous définit une trajectoire .

En amorçant le second bouton, le programme exécute quatre fonctions qui nous permettent de mettre en position initiale, prendre, déplacer et déposer des objets avec le bras robot selon les positions sauvegardées. Le sous programme des deux boutons est présenté ci-dessous :

Dans ce sous-programme, on illustre seulement trois positions de sauvegarde pour le bouton 1.

```
if(digitalRead(button1) == HIGH){
  button1pres++;
  switch(button1pres){
    case 1:
      servo1PosSave[0] = pot1Angle;
      servo2PosSave[0] = pot2Angle;
      servo3PosSave[0] = pot3Angle;
      servo4PosSave[0] = pot4Angle;
      break;
    case 2:
      servo1PosSave[1] = pot1Angle;
      servo2PosSave[1] = pot2Angle;
      servo3PosSave[1] = pot3Angle;
      servo4PosSave[1] = pot4Angle;
      delay(1500);
      break;
    case 3:
      servo1PosSave[2] = pot1Angle;
      servo2PosSave[2] = pot2Angle;
      servo3PosSave[2] = pot3Angle;
      servo4PosSave[2] = pot4Angle;
      delay(1500);
      break;
  }
}
```

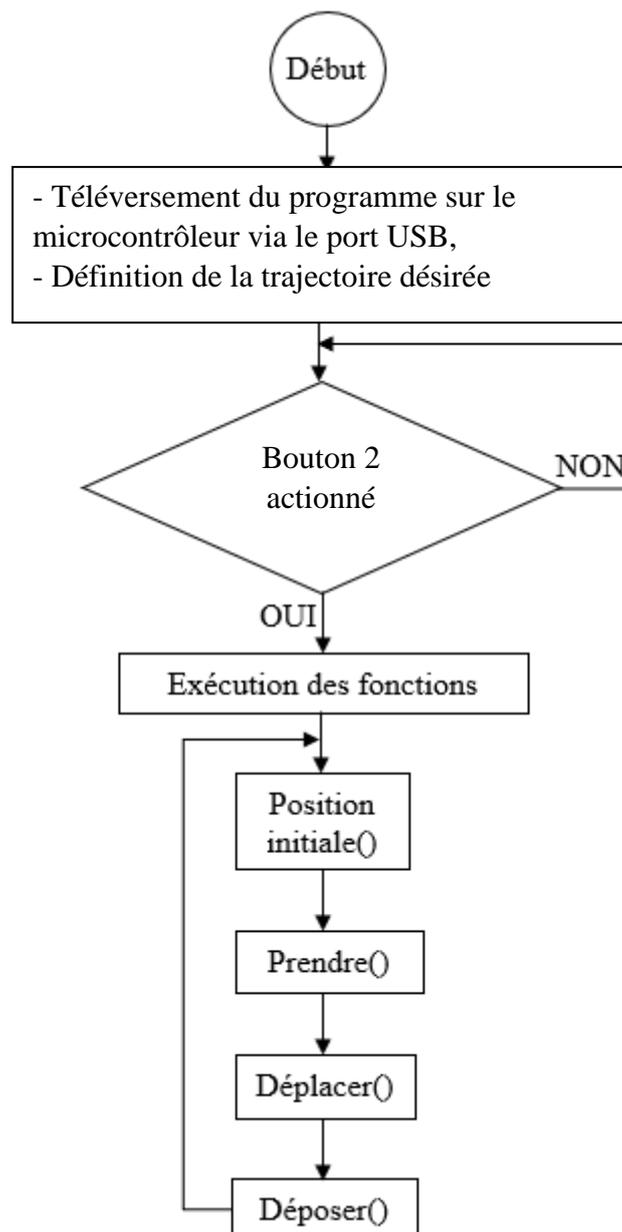
Le bouton deux nous permet d'exécuter huit positions de sauvegarde avec « i » allant de 0 à 7.

```
if(button2pres){
  for(int i=0; i<8; i++){
    servoB.write(servo1PosSave[i]);
    servoD.write(servo2PosSave[i]);
    servoG.write(servo3PosSave[i]);
    servoP.write(servo4PosSave[i]);
    delay(2000);
  }
}
```

Comme précisé précédemment, le programme exécute directement les angles désirés sauvegardés dans chaque articulation. L'inclusion des retards est faite afin d'éviter les déplacements rapides entre les différents organes du robot qui peuvent causer des dégâts matériels.

Le programme détaillé est donné dans l'annexe D.

Le logigramme de **Fig.3.17** illustre le principe de la commande à angles préprogrammés.



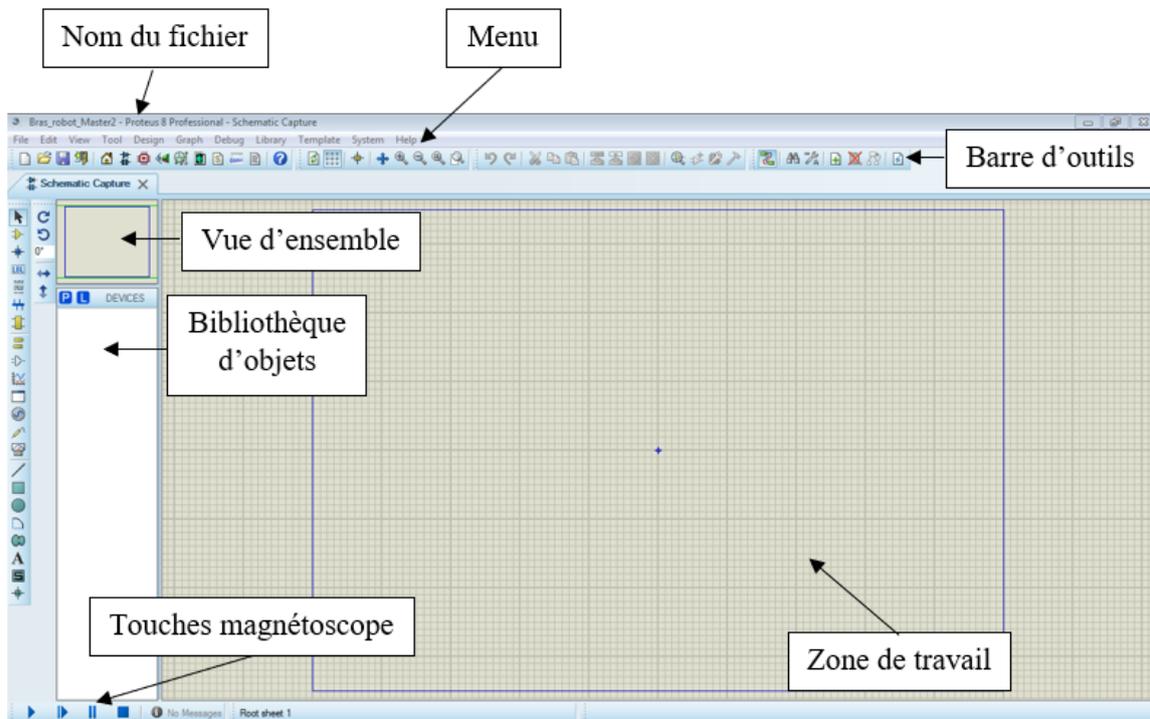
**Fig.3.17.** Logigramme résumant la commande à angles préprogrammés

### 3.5. Implémentation et simulation du projet sous ISIS

En réalité, avant la réalisation pratique, notre projet a fait l'objet d'une implémentation sous ISIS. Ce qui a permis de simuler le projet et de vérifier le bon fonctionnement des différentes parties avant l'implantation pratique.

### 3.5.1. Présentation du logiciel ISIS (Intelligent Schematic Input System)

Isis Proteus est un logiciel de développement et de simulation largement utilisé ces dernières années comme outil pédagogique, en particulier pour sa simplicité due à son

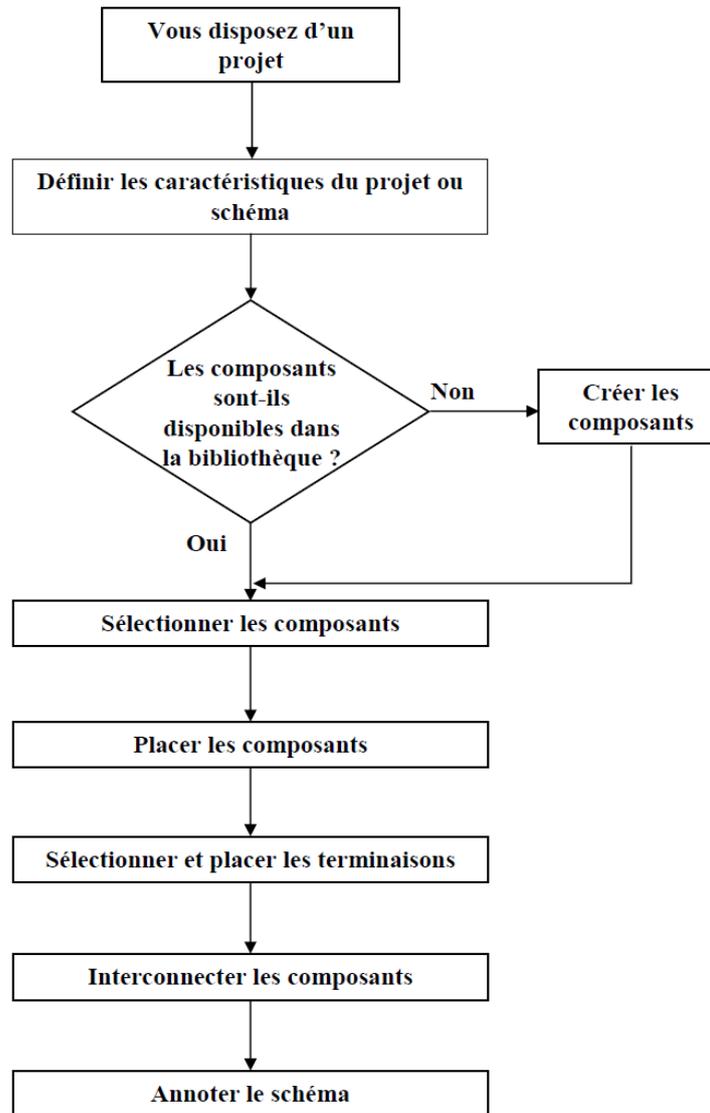


environnement graphique et interactif. Les différents outils mis à la disposition de l'utilisateur permettent, outre la simulation, des analyses dans des domaines différents (temporel, fréquentiel), ainsi que des régimes différents (statique, dynamique) [33].

**Fig.3.18.** Logiciel Proteus ISIS

### 3.5.2. Les différentes phases à suivre pour l'élaboration d'un projet

Les différentes étapes à suivre pour l'élaboration d'un projet sur ISIS sont illustrées dans le logigramme de **Fig.3.19**.



**Fig.3.19.** Logigramme des phases explicatives de l'utilisation du logiciel ISIS

### 3.5.3. Implémentation et simulation

La carte de commande est équipée d'une carte Arduino de type UNO, de quatre potentiomètres qui contrôlent manuellement le bras, de quatre servomoteurs qui permettent de le faire bouger et enfin d'un module Bluetooth HC-05 pour assurer la communication avec un téléphone mobile dans le but de le contrôler à distance par une application Android.

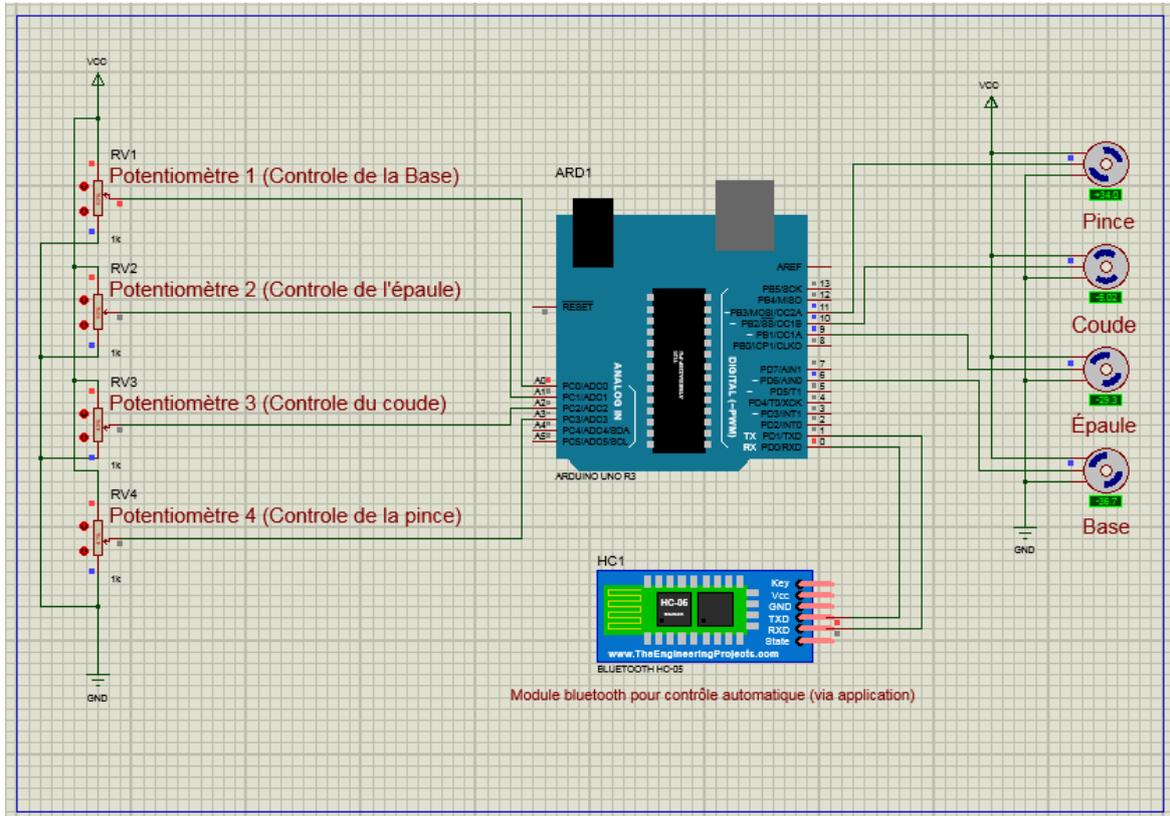


Fig.3.20. Simulation du bras manipulateur sous ISIS

### 3.6. Tests et résultats pratiques

#### 3.6.1. Commande manuelle

Dans cette partie, on teste les variations d’entrées des potentiomètres et leurs sorties obtenues. On a représenté les courbes en fonction des conversions obtenues entre l’entrée analogique et le résultat donné par la fonction *map* pour les angles articulaires.

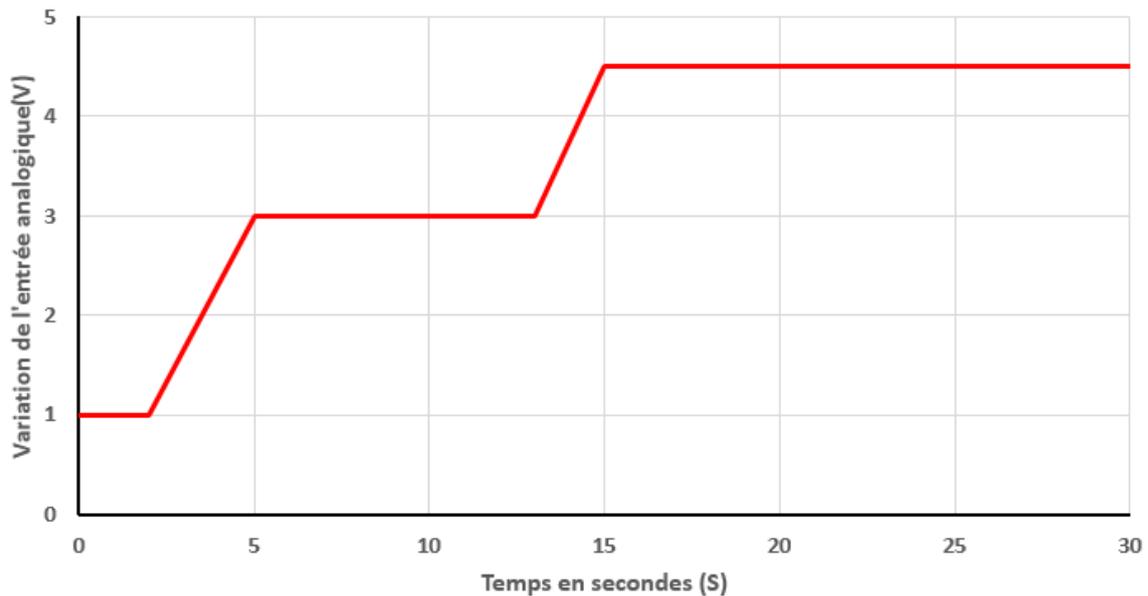
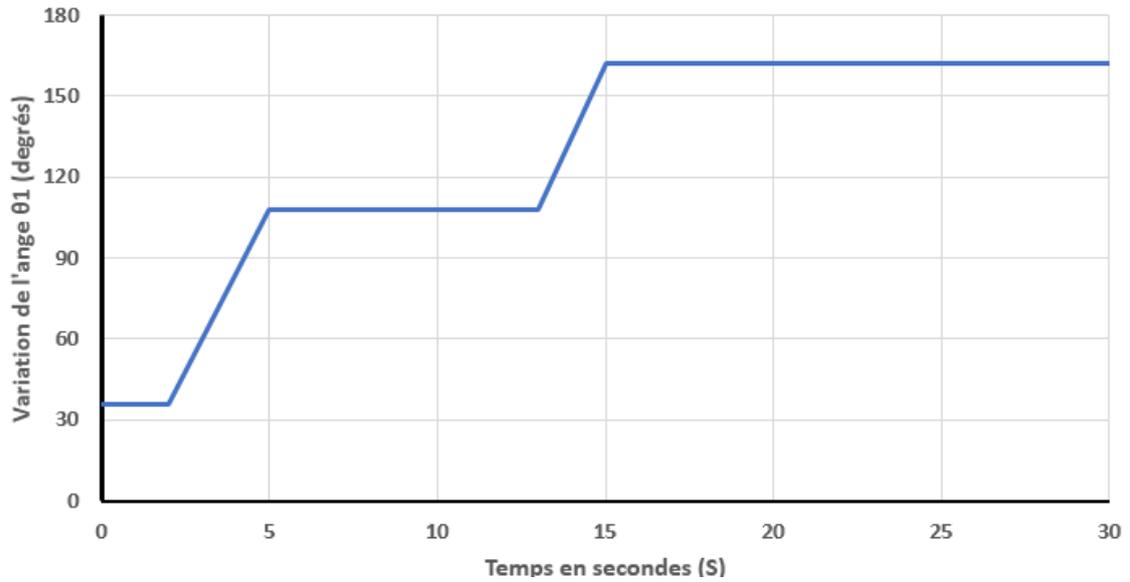
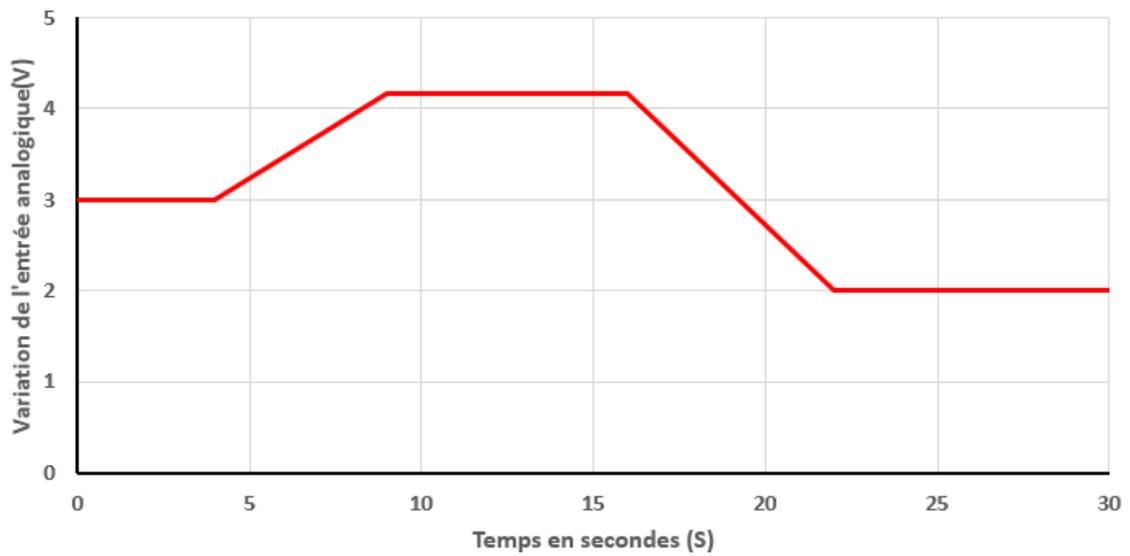


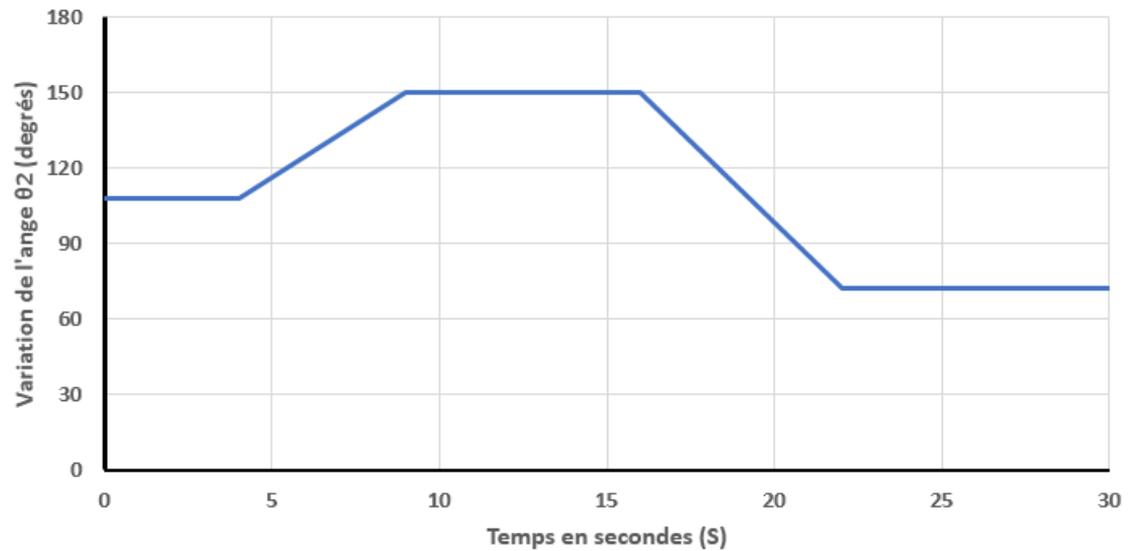
Fig.3.21. Variation de l’entrée analogique de la première articulation en fonction du temps



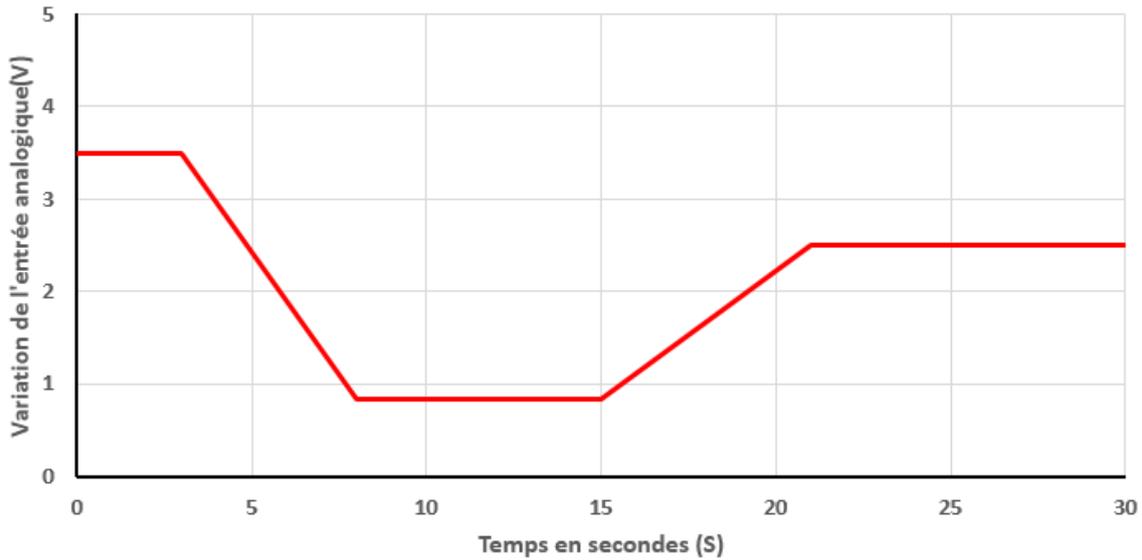
**Fig.3.22.** Variation de l'angle  $\theta_1$  en fonction du temps



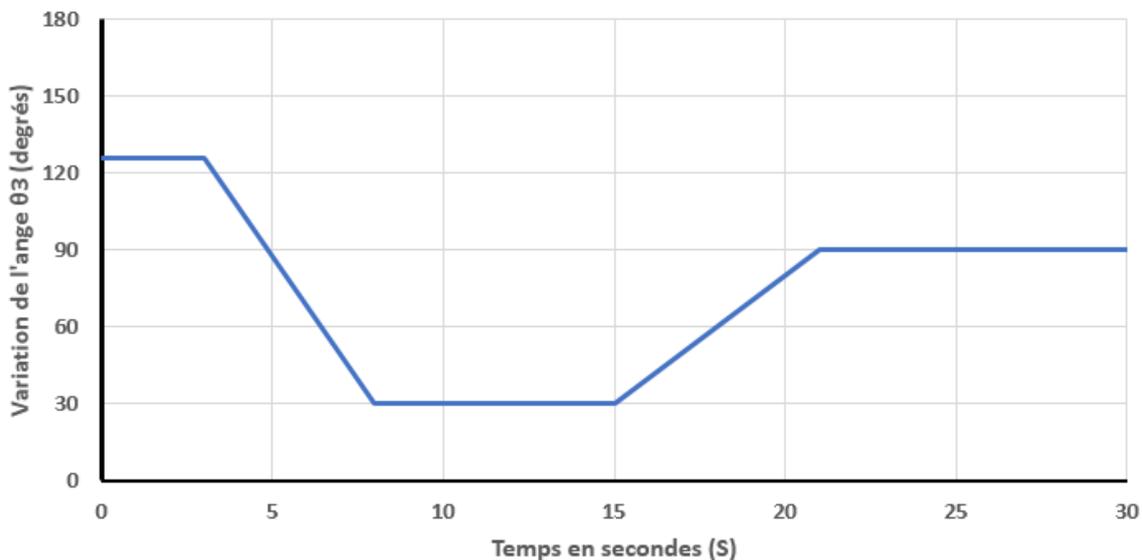
**Fig.3.23.** Variation de l'entrée analogique de la deuxième articulation en fonction du temps



**Fig.3.24.** Variation de l'angle  $\theta_2$  en fonction du temps



**Fig.3.25.** Variation de l'entrée analogique de la troisième articulation en fonction du temps



**Fig.3.26.** Variation de l'angle  $\theta_3$  en fonction du temps

D'après les résultats obtenus, on constate que les entrées analogiques varient de 0V à 5V. Celles-ci ont été converties grâce à un convertisseur A/N de l'Arduino, qui les a transformées en signaux numériques variant entre 0 et 1023. Elles ont ensuite été adaptées par la fonction *map*, de 0° à 180°, pour être utilisées comme angles de déplacement pour les servomoteurs.

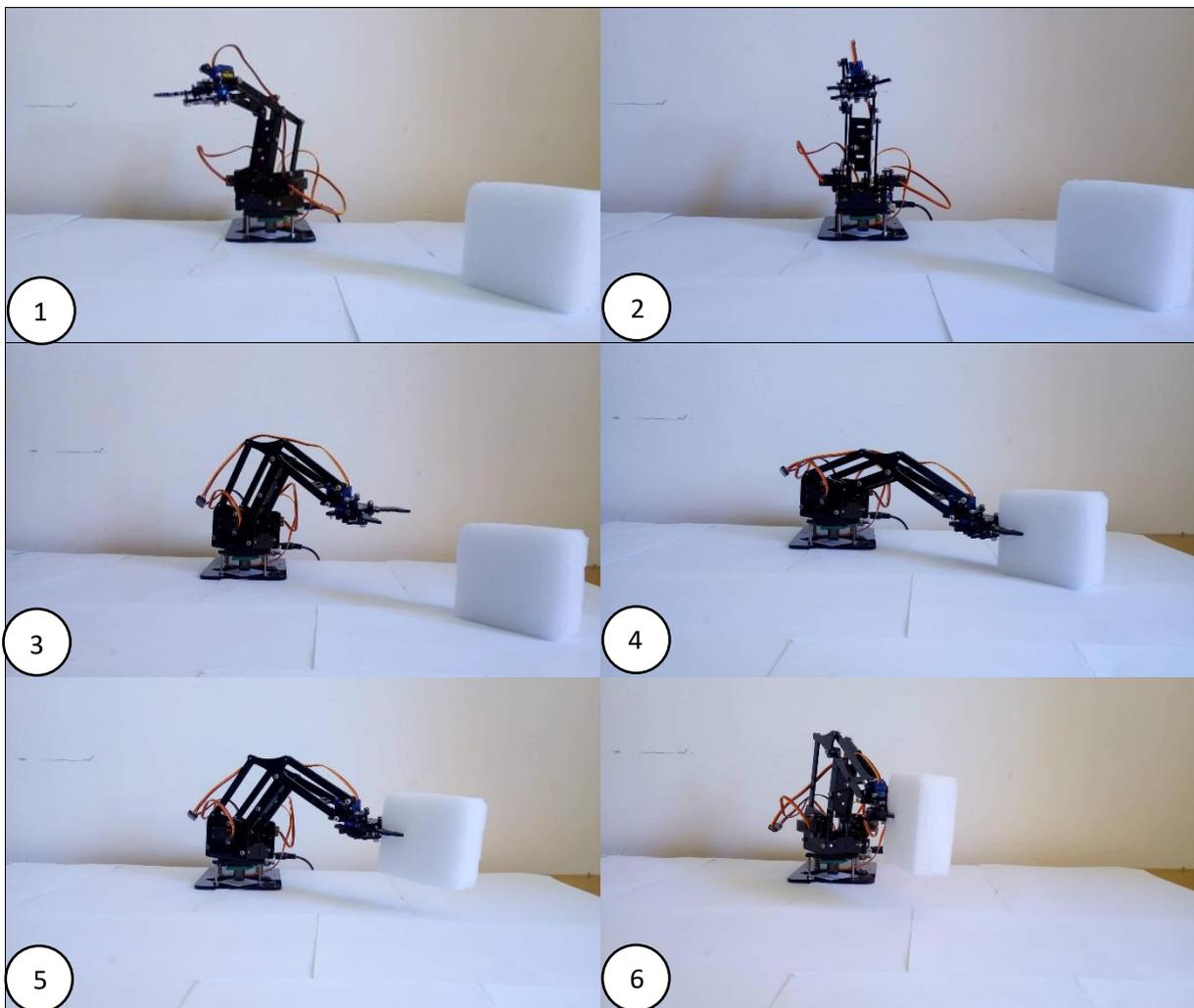
- Quand la variation de l'entrée analogique de l'articulation est constante, la variation de l'angle  $\theta$  est constante.
- Quand la variation de l'entrée analogique de l'articulation augmente, la variation de l'angle  $\theta$  augmente.

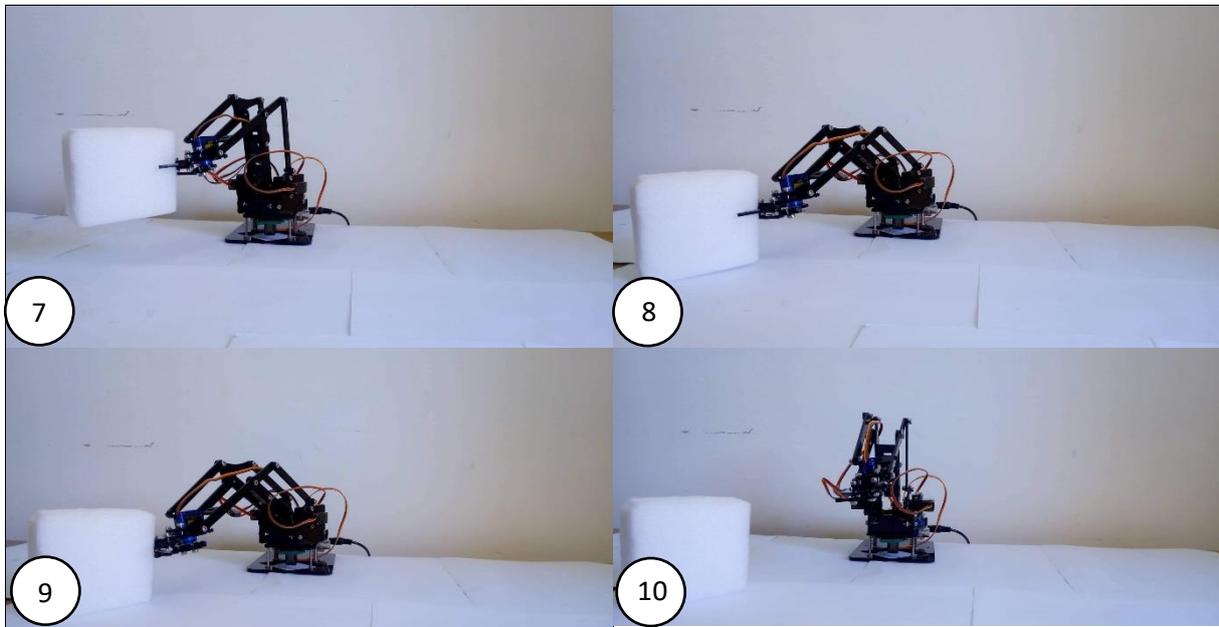
- Quand la variation de l'entrée analogique de l'articulation diminue, la variation de l'angle diminue.

Pour conclure, les potentiomètres nous permettent de faire varier les articulations du bras robot selon notre convenance.

### 3.6.2. Commande par angles préprogrammés

Dans cette partie, on teste notre bras manipulateur final avec la commande préprogrammée qui nous permet de déplacer un objet d'une position à une autre. Vu la difficulté pour l'obtention d'informations démonstratives décrivant l'exécution du programme, on a opté pour une séquence d'images afin de mieux illustrer les résultats.





**Fig.3.27.** Test du bras manipulateur pour déplacement d'objet.

D'après ce test illustré, on constate que le robot suit la trajectoire prévue et déplace l'objet à l'endroit désiré.

### 3.7. Simulation du bras manipulateur avec la commande PID

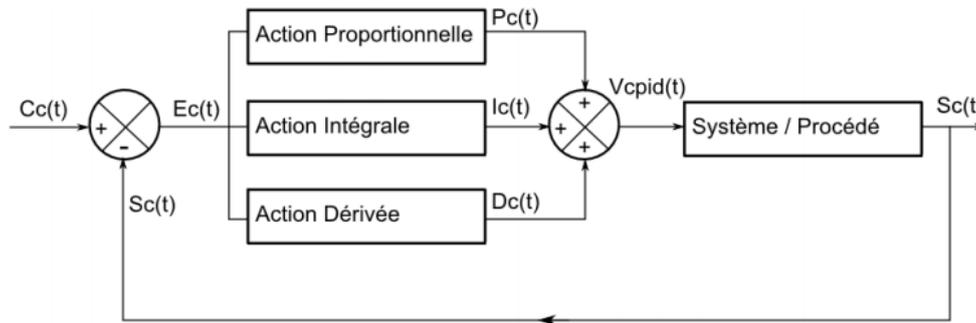
Toutes les commandes exposées et implémentées pratiquement sont réalisées en boucle ouverte. La commande en boucle fermée, plus utile en présence des perturbations, constitue une extension intéressante de ce travail. Néanmoins, on a abordé le sujet dans ce mémoire, par une étude théorique et une simulation sous SimMechanics (MATLAB SIMULINK).

Une fois le modèle dynamique du robot établi, le plus intéressant reste la commande du bras manipulateur. Cette matière a fait l'objet de nombreuses études. Certains y ont appliqué les réseaux de neurones artificiels, d'autres encore utilisent le réglage flou. La régulation n'étant pas l'objet principal de notre travail, on a opté pour un régulateur PID simple.

#### 3.7.1. Le régulateur PID

Le régulateur PID (Proportionnel, Intégral, Dérivé) est un régulateur, en boucle fermée, qui a pour rôle de diminuer l'erreur entre la consigne et la mesure et d'atteindre la valeur souhaitée pour l'une des variables du système (vitesse, position...) [34].

La régulation sert à maintenir des grandeurs constantes, en minimisant rapidement l'effet des perturbations en le compensant. La poursuite sert, quant à elle, à s'adapter rapidement aux nouvelles consignes. C'est ce qu'on appelle l'asservissement.



**Fig.3.28.** Schéma du régulateur PID [34].

### 3.7.1.1. La composante proportionnelle

Comme son nom l'indique, elle est directement proportionnelle à l'erreur  $E_c(t)$  et s'exprime selon la formule :

$$P_c(t) = k_p \cdot E_c(t) \quad (3.1)$$

La valeur du gain  $k_p$  doit être choisie en fonction du système. Plus  $k_p$  est important, plus l'erreur statique diminue et rend le système plus rapide, mais au risque de déstabiliser le système [35].

### 3.7.1.2. La composante intégrale

Elle est définie par :

$$I_c(t) = k_i \int_0^t E_c(t) dt \quad (3.2)$$

Avec  $k_i$  : constante intégrale du système.

L'intérêt principal de ce correcteur est d'ajouter, dans la chaîne de commande, une intégration pour annuler l'erreur statique. Plus la constante  $k_i$  est élevée, plus la réponse sera rapide. En revanche cela entraîne également un risque de dépassement de la consigne en raison d'une accumulation d'erreurs dans la partie intégrale [34].

### 3.7.1.3. La composante dérivée

Elle dépend des variations de l'erreur  $E_c(t)$ :

$$D_c(t) = k_d \frac{dE_c(t)}{dt} \quad (3.3)$$

Comme nous l'avons vu précédemment, l'introduction de la composante intégrale entraîne également un risque de dépassement de la consigne. En calculant la variation de l'erreur à chaque itération, la partie dérivée permet de limiter ce phénomène en ralentissant la commande lorsque la valeur de la sortie s'approche de la consigne [34].

### 3.7.2. Calcul des paramètres du régulateur PID

Le modèle dynamique inverse du bras est donné par la formule finale (2.54) du second chapitre, qui donne le vecteur de commande en fonction de l'accélération. Pour la simulation des mouvements du bras, on utilise le modèle direct

$$\ddot{\theta} = M^{-1}(\theta)[\Gamma - C(\theta, \dot{\theta})\dot{\theta} - G(\theta)] \quad (3.4)$$

Dans certaines conditions de fonctionnement à de faibles vitesses et avec des constituants légers, la commande peut être restreinte à une forme très simple qui correspond à un modèle linéarisé autour de point de fonctionnement avec des articulations complètement découplées. Chaque articulation est supposée indépendante et les termes des forces centrifuges et Coriolis sont complètement négligés.

Dans notre réalisation on a bien un modèle avec de faible vitesse et un poids léger qui est dû au matériau utilisé qui est du plexiglas. Chaque articulation donc peut être modélisée par :

$$\Gamma_{ii} = M_{ii}\ddot{\theta}_i \quad (3.5)$$

Ce qui permet d'écrire le modèle sous sa forme linéaire découplée :

$$\ddot{\theta} = M^{-1}(\theta)\Gamma \quad (3.6)$$

La loi de commande PID est de la forme :

$$\Gamma_i = k_{vi}\dot{e}_i + k_{pi}e_i + k_{ii}\int_0^t e_i(\Gamma)d\Gamma \quad (3.7)$$

Le modèle dynamique inverse s'écrit :

$$k_{vi}\dot{e}_i + k_{pi}e_i + k_{ii}\int_0^t e_i(\Gamma)d\Gamma = m_i\ddot{\theta} \quad (3.8)$$

Par application de la transformée de Laplace, on a :

$$m_iP^3 + k_{vi}P^2 + k_{pi}P + k_{ii} = 0 \quad (3.9)$$

Les versions les plus simple de commande n'utilise aucun paramètres de système. On pose donc  $m_i = 1$  :

$$P^3 + k_{vi}P^2 + k_{pi}P + k_{ii} = 0 \quad (3.10)$$

Dans la commande des bras de robots, on évite les réponses oscillatoires. On prend donc un facteur d'amortissement  $\zeta=1$ , qui correspond au début du régime aperiodique :

$$(P - P_3)(P^2 + 2\zeta\omega_n P + \omega_n^2) = 0 \quad (3.11)$$

Pour  $\zeta=1$ , il reste à fixer la valeur de la pulsation naturelle,  $\omega_n$ , et de calculer  $k_v$ ,  $k_p$  et  $k_i$ . Après étude, on a pris  $\omega_n = 3$  rad/sec, ce qui nous donne :

$$P_1 = P_2 = -3 \quad (3.12)$$

Le troisième pôle est choisi de manière que la réponse soit plus rapide que celles des deux autres pôles. On pose :  $P_3 = 5P_2$ . Ce qui donne :

$$(P + 15)(P^2 + 6P + 9) = 0 \quad (3.13)$$

$$P^3 + 21P^2 + 99P + 135 = 0 \quad (3.14)$$

Par comparaison avec (3.10) et identification :  $k_v = 21$ ,  $k_p = 99$  et  $k_i = 135$  :

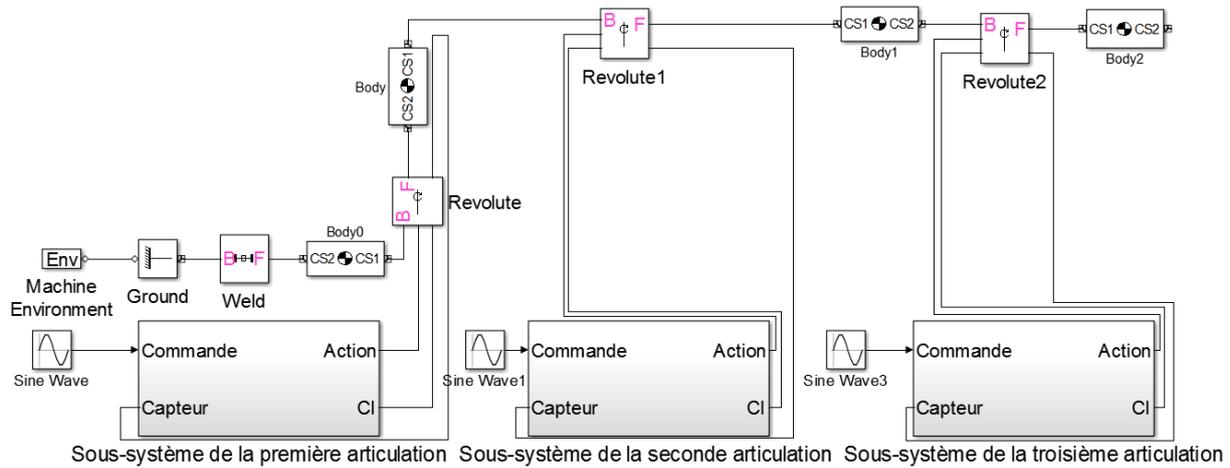
$$\Gamma_1 = 21\dot{e}_1 + 99e_1 + 135\int_0^t e_1(\Gamma)d\Gamma \quad (3.15)$$

$$\Gamma_2 = 21\dot{e}_2 + 99e_2 + 135\int_0^t e_2(\Gamma)d\Gamma \quad (3.16)$$

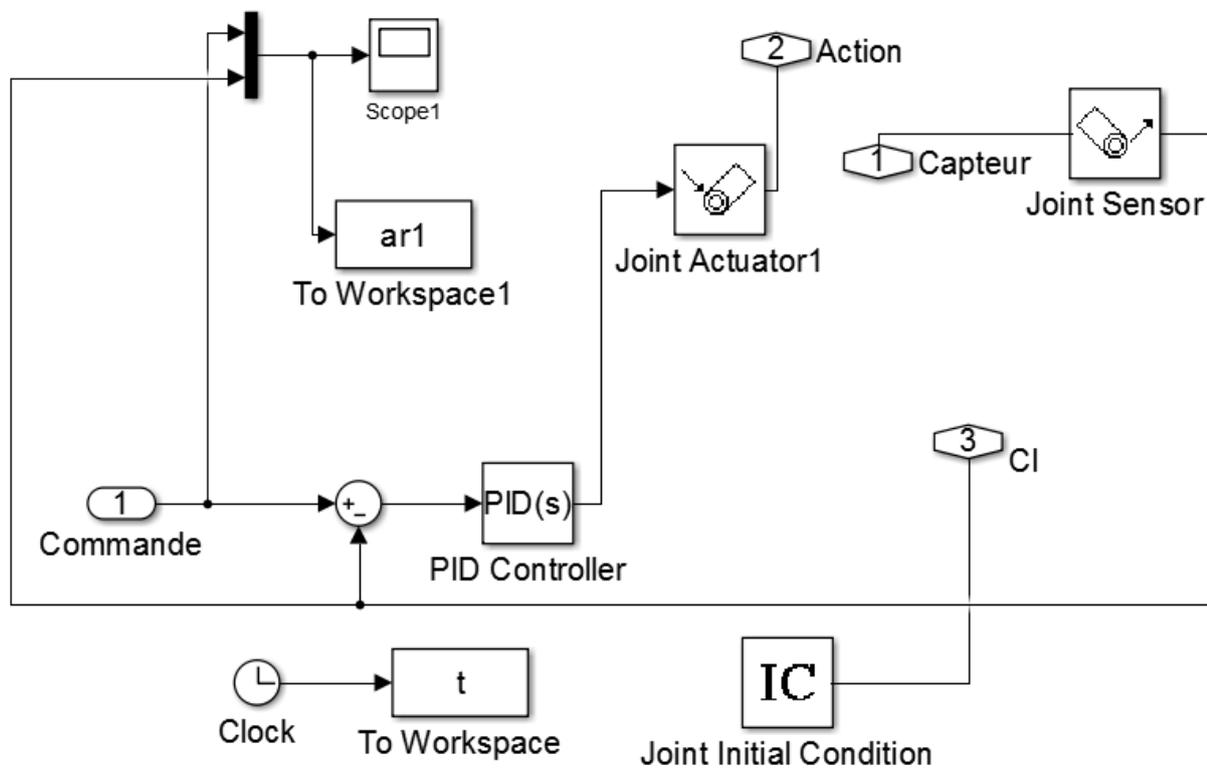
$$\Gamma_3 = 21\dot{e}_3 + 99e_3 + 135\int_0^t e_3(\Gamma)d\Gamma \tag{3.17}$$

**3.7.2.1. Simulation avec SimMechanics (MATLAB SIMULINK)**

On simule la modélisation dynamique du bras manipulateur en utilisant les paramètres du régulateur PID obtenu précédemment pour chaque articulation, avec une consigne sinusoïdale, sous SimMechanics.

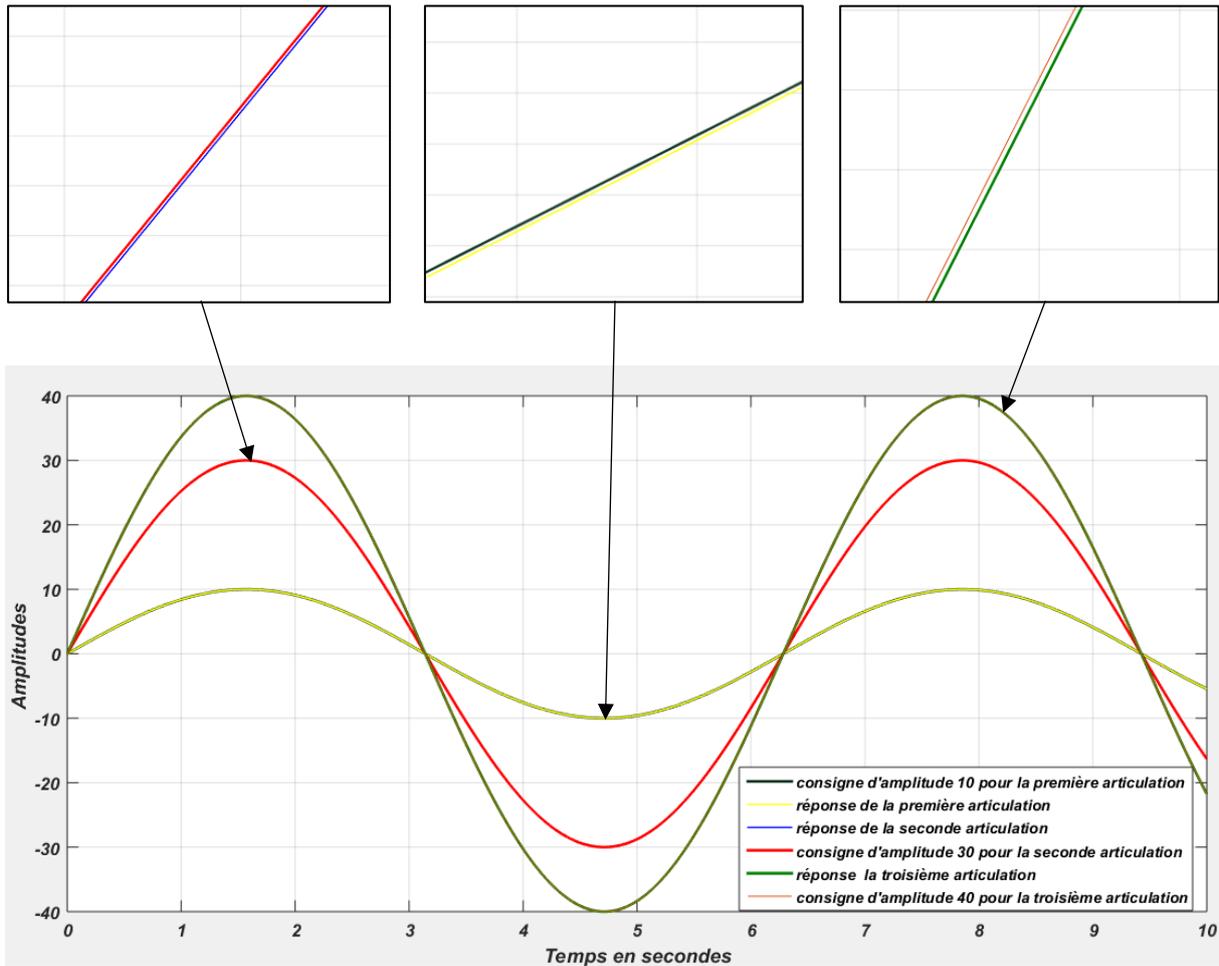


**Fig.3.29.** Modèle du robot dans SimMechanics



**Fig.3.30.** Les composants du sous système pour la première articulation

### 3.7.2.2. Résultats



**Fig.3.31.** Réponses des trois articulations à différentes amplitudes pour un signal sinusoïdale

Les résultats de la simulation montrent que le suivi de la consigne est assuré en un temps très court sans aucun dépassement apparaissant dans le temps. Cela revient aux pôles réels négatifs imposés dans la loi de commande PID.

### 3.8. Conclusion

Dans la première partie de ce chapitre, on s’est intéressé à la carte de commande Arduino et à son fonctionnement en décrivant les types de programmes implémentés à notre bras manipulateur (manuelle, sans fil et angles préprogrammés), puis on a exposé leurs tests et leurs résultats pratiques. Dans la seconde partie, on s’est penché sur la commande du bras manipulateur en boucle fermée par une simulation avec la commande PID sous SimMechanics.

---

# **CONCLUSION GÉNÉRALE**

Ce mémoire présente notre travail qui concerne l'étude et la réalisation d'un bras manipulateur à trois degrés de liberté, avec l'hypothèse d'un environnement de travail sans obstacles. Après une exploration générale du domaine de la robotique, on a abordé le vif du sujet en étudiant les différents organes constituant notre bras manipulateur. Par la suite, on s'est penché sur la modélisation et l'élaboration des modèles géométrique, cinématique et dynamique, qui ont pour rôle respectivement de générer des trajectoires de références, d'analyser les singularités et de commander et simuler le robot. Enfin, on a procédé à l'étude pratique et l'implémentation des techniques de commande prévues sur notre réalisation (par potentiomètres, application mobile et angles préprogrammés).

Pour la réalisation pratique, le châssis a été conçu par découpe laser. La programmation est faite à base du langage C++ et a fait l'objet d'une implémentation sur la carte Arduino UNO. Le contrôle du bras manipulateur se fait à l'aide d'une application mobile réalisée (sans fil) et grâce au shield incluant des potentiomètres. La dernière partie du travail concerne une simulation de la commande PID du bras robot sous MATLAB en utilisant SimMechanics.

Le manuscrit a été organisé en trois chapitres. Le premier présente des généralités. Le second porte sur la description et la modélisation du bras manipulateur étudié. Le troisième chapitre expose la commande et la simulation du bras manipulateur commandé par un régulateur PID sous MATLAB.

L'étude et la réalisation de ce projet nous ont permis d'approfondir nos connaissances théoriques et pratiques acquises durant notre formation, notamment en robotique et en systèmes embarqués. Nous avons pu appréhender différents points importants comme le fait de travailler en équipe et de concevoir et mener à bien un projet.

Bien que ce modeste travail ait atteint ses objectifs, on estime qu'il pourrait être encore amélioré. En effet, les travaux décrits dans ce mémoire peuvent se poursuivre sur plusieurs voies. Tout d'abord, il serait intéressant d'envisager un environnement de travail avec obstacles, ce qui n'est pas le cas dans cette étude. Il serait également intéressant d'implémenter des commandes en boucle fermée, plus utiles en présence des perturbations. Cela constitue une bonne extension de ce projet. Pour la réalisation du bras, il serait avantageux d'optimiser la construction en utilisant un matériau plus solide. On pourrait utiliser des servomoteurs offrant un couple moteur plus important, dans le but de soulever des objets plus lourds.

- [1] ALMUSAWI Ahmed R. J., KAPUCU Sadettin, DÜLGER Lale Canan « Robotic Arm Dynamic and Simulation with Virtual Reality Model (VRM) » The 3rd International Conference on Control, Decision and Information Technologies, Malte, Volume: 3, Avril 2016.
- [2] Dictionnaire Reverso, 2010, Edition en ligne.
- [3] Dictionnaire Larousse, 2008, Edition en ligne.
- [4] EYRAUD, Charles-Henri. « Horloges astronomiques au tournant du XVIIIe siècle : de l'à-peu-près à la précision ». Thèse de doctorat. Lyon 2, 2004.
- [5] Official website IFR INTERNATIONAL FEDERATION OF ROBOTICS.
- [6] EL HAMZAOUI Oussama « Localisation et Cartographie Simultanées pour un robot mobile équipé d'un laser à balayage : CoreSLAM », Thèse de Doctorat, École nationale supérieure des mines de Paris, 25 septembre 2012.
- [7] BOSTELMAN Roger « PERFORMANCE MEASUREMENT OF MOBILE MANIPULATORS », Thèse de Doctorat, UNIVERSITY OF BURGUNDY, 16 mars 2018.
- [8] ISO (International Standard Organization), 1947.
- [9] LAURENT Guillaume, CHAILLET Nicolas « Robotique industrielle » Cours, Université de Bourgogne Franche-Comté, 2017.
- [10] BOIMOND, Jean-Louis. « ROBOTIQUE » Cours, ISTIA, Université Angers, 2017.
- [11] BOUKLI HACENE Lotfi Fazil « Commande et supervision d'un ensemble de robots via internet », Mémoire en vue de l'obtention de Magister, Université Des Sciences Et De La Technologie D'oran, 26 avril 2012.
- [12] BAHLOUL Abdelkrim « Sur la commande des robots manipulateurs industriels en co-manipulation robotique », Thèse de Doctorat, Université Paris-Saclay, 7 décembre 2018.
- [13] LE BARS Fabrice « Analyse par intervalles pour la localisation et la cartographie simultanées ; Application à la robotique sous-marine », Thèse de Doctorat, UNIVERSITÉ DE BRETAGNE OCCIDENTALE, 23 novembre 2011.
- [14] FATOUX Julien « Mise en œuvre d'un robot humanoïde et contribution à la génération de marches dynamiques optimales », Thèse de Doctorat, UNIVERSITE DE POITIERS, 7 août 2006.

- [15] FISETTE P., BUYSE H., SAMIN J.C. « Introduction à la robotique » Cours, 19 février 2004.
- [16] RASHIDIFAR, Mohammad Amin, RASHIDIFAR, Ali Amin, et AHMADI, Darvish. « Modeling and Control of 5DOF Robot Arm Using Fuzzy Logic Supervisory Control », Article, vol. 2, no 2, 2089-4856, Janvier 2013, International Journal of Robotics and Automation (IJRA).
- [17] CRAIG, John J. « Introduction to robotics: mechanics and control », Pearson Education International, Livre, 2009.
- [18] EL-HUSSIENY Haitham « Robotics Engineering », Cours, Benha University, 2017.
- [19] GANGLOFF Jacques « Cours de Robotique », Cours, ENSPS 3A Master ISTI, 2018.
- [20] ANGELES Jorge « Fundamentals of robotic mechanical systems », Springer, Livre, 2002.
- [21] MENDIL B. « Cours de Robotique », Cours, Université Abderrahmane Mira, Béjaia, 2017.
- [22] FERRERO F. « Moteurs et transistors MOS », Cours, Université de Nice, 2017.
- [23] DATASHEET « MG90S Metal Gear Servo », Components 101, 2019.
- [24] DATASHEET « SG90 Servo Motor », Components 101, 2017.
- [25] BETTAIEB Mohamed Nizar « Modélisation du comportement mécanique d'engrenages. Couplages entre engrenement et éléments de structures flexibles », Thèse de Doctorat, Ecole Doctorale Des Sciences Pour L'Ingénieur De Lyon, 16 juin 2006.
- [26] DATASHEET « HC-05 Bluetooth to Serial Port Module », Components 101, 2018.
- [27] GAUTRON Laurent « PHYSIQUE TOUT LE COURS EN FICHES », Dunod, Livre, 2015.
- [28] PASQUIER Claude « Mécanique », Cours, Polytech Paris-Sud, 2012.
- [29] LAFOND Roger « Analyse dynamique du mouvement », Cours, 2018.
- [30] BOUQUET Frédéric, BOBROFF Julien « Microcontrôleur arduino », Cours, Université Paris Sud, printemps 2015.
- [31] RAO Michael « Introduction à l'Arduino », Cours, Université de Lyon, 2016.
- [32] DATASHEET « Arduino UNO », Components 101, 2018.

[33] LAKHDARI.F « Introduction à la simulation et routage des circuits avec le logiciel PROTEUS V7 et V8 », Cours, Université Des Sciences Et De La Technologie D'oran Faculté Du Génie Electrique Département D'électronique Licence, 2016/2017.

[34] DURAND-ESTEBE Baptiste « Optimisation Energétique du Rafraichissement des Datacenters », Thèse de Doctorat, Ecole Doctorale Des Sciences Physiques Et De L'Ingénieur, 4 juillet 2014.

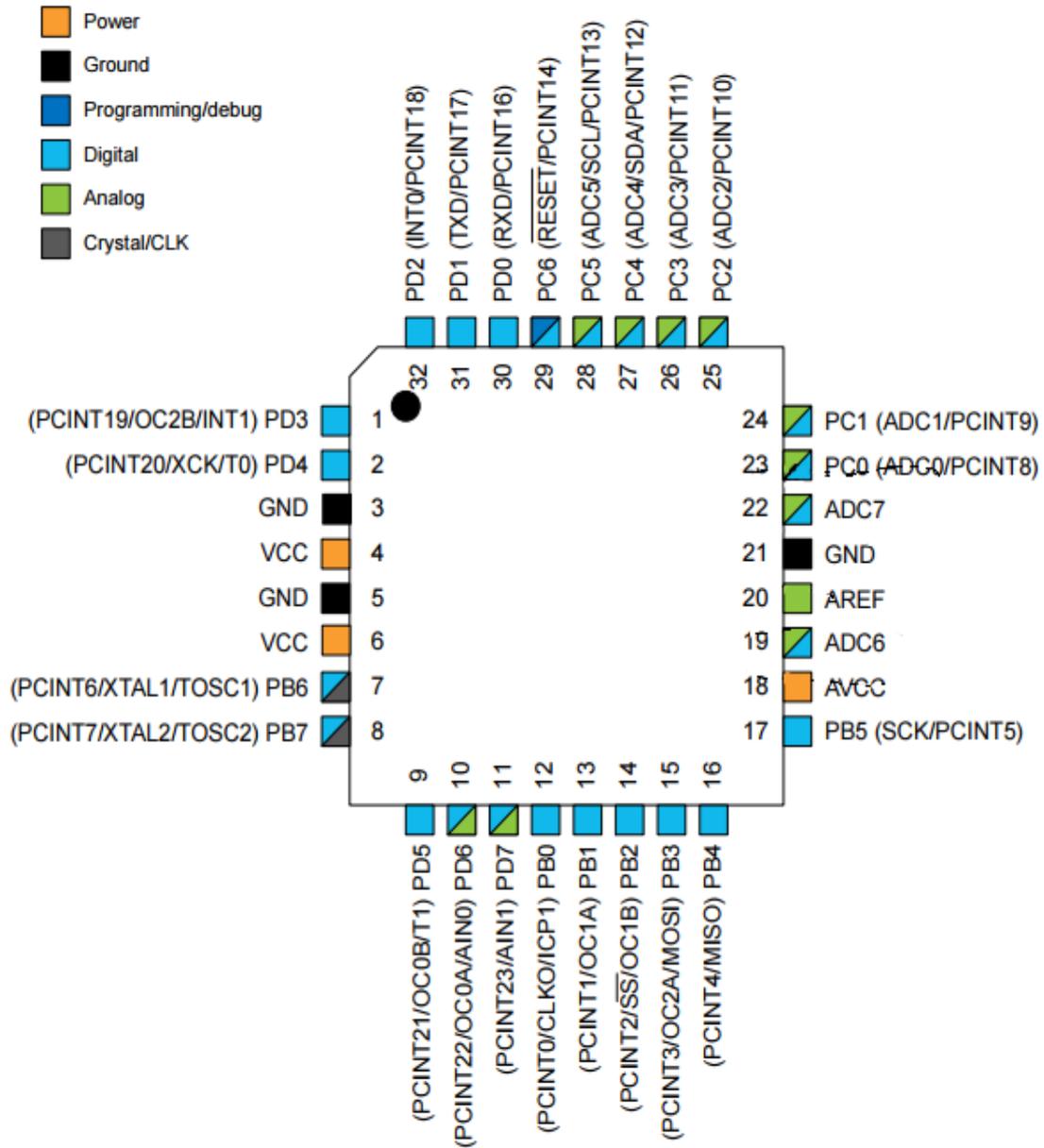


# **ANNEXES**

# Annexe A

Brochage Atmega 328 P 32-pin TQFP

Figure 5-3. 32-pin TQFP Top View



## Annexe B

Programme de contrôle par potentiomètre :

```
//inclure la bibliothèque des servomoteurs
#include <Servo.h>

//définition de nos servomoteurs
Servo servo1;
Servo servo2;
Servo servo3;
Servo servo4;

//définition de nos potentiomètres
int pot1 = A0;
int pot2 = A1;
int pot3 = A2;
int pot4 = A3;

//les variables utilisé pour la lecture des valeurs de la broche analogique (potentiomètres)
int valPot1;
int valPot2;
int valPot3;
int valPot4;
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
void setup()
{
  //attacher les servomoteurs aux pins qui fournissent la PWM (11 10 9 6)
  servo1.attach(11);
  servo2.attach(10);
  servo3.attach(9);
  servo4.attach(6);
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

void loop()
{
  //lire la valeur des potentiomètres (valeur entre 0 et 1023)

  valPot1 = analogRead(pot1);
  valPot1 = map (valPot1, 0, 1023, 0, 180); //changer de plage pour servo (valeur entre 0 et 180)
  servo1.write(valPot1); //régler la position du servo en fonction de la valeur mise à l'échelle
  delay(5);

  valPot2 = analogRead(pot2);
  valPot2 = map (valPot2, 0, 1023, 0, 180);
  servo2.write(valPot2);
  delay(5);

  valPot3 = analogRead(pot3);
  valPot3 = map (valPot3, 0, 1023, 0, 180);
  servo3.write(valPot3);
  delay(5);

  valPot4 = analogRead(pot4);|
  valPot4 = map (valPot4, 0, 1023, 0, 180);
  servo4.write(valPot4);
  delay(5);
}
```

## Annexe C

Programme de contrôle par application mobile Android sans les deux boutons :

```
//Inclure la bibliothèque des servomoteurs
#include <Servo.h>
//Définition de nos servomoteurs
Servo servoB;
Servo servoD;
Servo servoG;
Servo servoP;
////////////////////////////////////
//Déclaration des variables
char a;
String readString;
////////////////////////////////////
void setup() {
//Attacher les servomoteurs aux pins qui fournissent la PWM(11 10 9 6)
  servoB.attach(6);
  servoD.attach(9);
  servoG.attach(10);
  servoP.attach(11);
////////////////////////////////////
  Serial.begin(9600); //Démarrer le port série avec un taux de 9600 bauds.
////////////////////////////////////
//Position initiale des servomoteurs en degré
  servoD.write(20);
  servoP.write(126);
  servoB.write(90);
  servoG.write(70);
////////////////////////////////////
  delay(10); //Attendre 10 millisecondes
}
////////////////////////////////////

void loop() {
//Lire à partir du port RX(port 0), et envoyer à partir du port TX (port 1)
  if (Serial.available()){
    a = Serial.read(); //Lire les données séries entrantes

    if(a=='A'){
      motor1();
    }

    if(a=='B'){
      motor2();
    }

    if(a=='C'){
      motor3();
    }
    if(a=='D'){
      motor4();
    }
    if(a=='E'){
      cyle1();
    }
    // if(a=='F'){
    //cyle2();
    // }
  }
}
////////////////////////////////////
```

```

void motor1(){
  delay(10); //Attendre 10 millisecondes
  while (Serial.available()) {

  char b = Serial.read();//Lire les données séries entrantes pour le servomoteur 1
    readString += b;
  }
  if (readString.length() >0)//Vérification de la longueur de la chaîne de caractère
  {
    Serial.println(readString.toInt());//Convertir la chaîne de caractères en un entier
    servoP.write(readString.toInt());//Déplacement du servomoteur vers l'angle obtenu à partir de la conversion
    readString=""; //Effacer la chaîne de caractère
  }
}
//////////
void motor2(){
  delay(10);
  while (Serial.available()) {
    char b = Serial.read();
    readString += b;
  }
  if (readString.length() >0) {
    Serial.println(readString.toInt());
    servoG.write(readString.toInt());
    readString="";
  }
}
//////////
void motor3(){
  delay(10);
  while (Serial.available()) {
    char b = Serial.read();
    readString += b;
  }
  if (readString.length() >0) {
    Serial.println(readString.toInt());
    servoD.write(readString.toInt());
    readString="";
  }
}
//////////
void motor4(){
  delay(10);
  while (Serial.available()) {
    char b = Serial.read();
    readString += b;
  }
  if (readString.length() >0) {
    Serial.println(readString.toInt());
    servoB.write(readString.toInt());
    readString="";
  }
}
}

```

## Annexe D

Programme de la commande par angles préprogrammés :

```
#include <Servo.h>
//définition des servos
Servo servoB;
Servo servoD;
Servo servoG;
Servo servoP;
////////////////////////////////////
//définition des boutons pour l'enregistrement et l'exécution des positions
const int button1 = 12;
const int button2 = 13;
////////////////////////////////////
//définition d'une variable pour les valeurs des boutons
int button1pres = 0;
boolean button2pres = false;
////////////////////////////////////
//define potentiometers
int pot1 = A0;
int pot2 = A1;
int pot3 = A2;
int pot4 = A3;
////////////////////////////////////
//les variables utilisé pour la lecture des valeurs de la broche analogique(potentiometers)
int valPot1;
int valPot2;
int valPot3;
int valPot4;
////////////////////////////////////
//définition des variables pour les positions enregistrées des servos
int servo1PosSave[]={1,1,1,1,1,1,1,1};
int servo2PosSave[]={1,1,1,1,1,1,1,1};
int servo3PosSave[]={1,1,1,1,1,1,1,1};
int servo4PosSave[]={1,1,1,1,1,1,1,1};
////////////////////////////////////
void setup() {
  //attacher les servos aux pins qui fournissent la PWM (11 10 9 6)
  servoB.attach(11);
  servoD.attach(10);
  servoG.attach(9);
  servoP.attach(6);
  //////////////////////////////////////
  //définition des boutons comme entrés
  pinMode(button1, INPUT);
  pinMode(button2, INPUT);
  //////////////////////////////////////
  //position initiale des servos
  servoB.write(90);
  servoD.write(20);
  servoG.write(70);
  servoP.write(126);
}
```

```

////////////////////////////////////
void loop() {
  valPot1 = analogRead(pot1);
  valPot1 = map (valPot1, 0, 1023, 0, 180);
  servoB.write(valPot1);
  delay(5);
  valPot2 = analogRead(pot2);
  valPot2 = map (valPot2, 0, 1023, 0, 180);
  servoD.write(valPot2);
  delay(5);
  valPot3 = analogRead(pot3);
  valPot3 = map (valPot3, 0, 1023, 0, 180);
  servoG.write(valPot3);
  delay(5);
  valPot4 = analogRead(pot4);
  valPot4 = map (valPot4, 0, 1023, 0, 180);
  servoP.write(valPot4);
  delay(5);
  //////////////////////////////////
  //si le bouton est enfoncé (HIGH), on enregistre la position des potentiomètres
  if(digitalRead(button1) == HIGH){
    button1pres++;
    switch(button1pres){
      case 1:
        servo1PosSave[0] = pot1Angle;
        servo2PosSave[0] = pot2Angle;
        servo3PosSave[0] = pot3Angle;
        servo4PosSave[0] = pot4Angle;
        break;
      case 2:
        servo1PosSave[1] = pot1Angle;
        servo2PosSave[1] = pot2Angle;
        servo3PosSave[1] = pot3Angle;
        servo4PosSave[1] = pot4Angle;
        delay(1500);
        break;
      case 3:
        servo1PosSave[2] = pot1Angle;
        servo2PosSave[2] = pot2Angle;
        servo3PosSave[2] = pot3Angle;
        servo4PosSave[2] = pot4Angle;
        delay(1500);
        break;
      case 4:
        servo1PosSave[3] = pot1Angle;
        servo2PosSave[3] = pot2Angle;
        servo3PosSave[3] = pot3Angle;
        servo4PosSave[3] = pot4Angle;
        delay(1500);
        break;
      case 5:
        servo1PosSave[4] = pot1Angle;
        servo2PosSave[4] = pot2Angle;
        servo3PosSave[4] = pot3Angle;
        servo4PosSave[4] = pot4Angle;
        delay(1500);
        break;
    }
  }
}

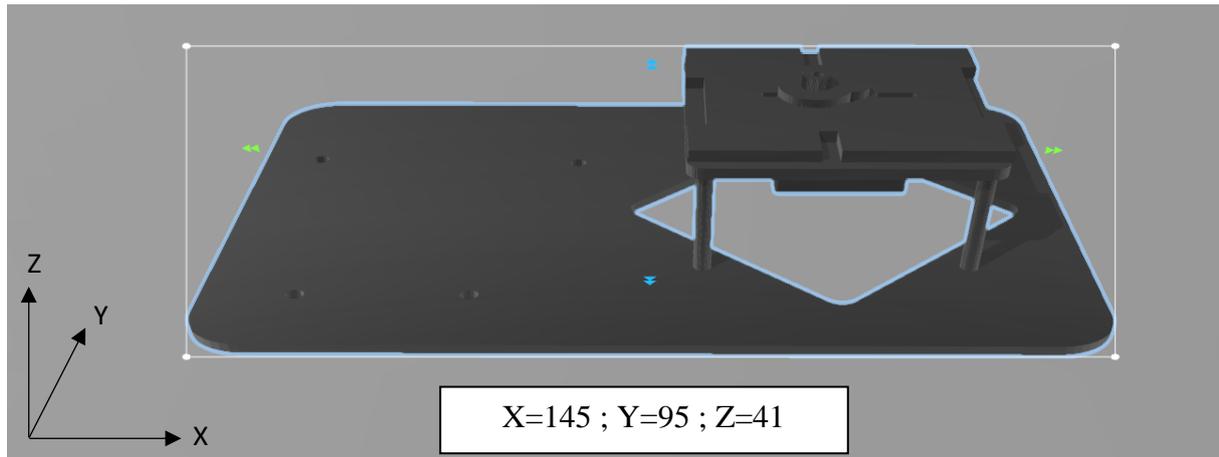
```

```

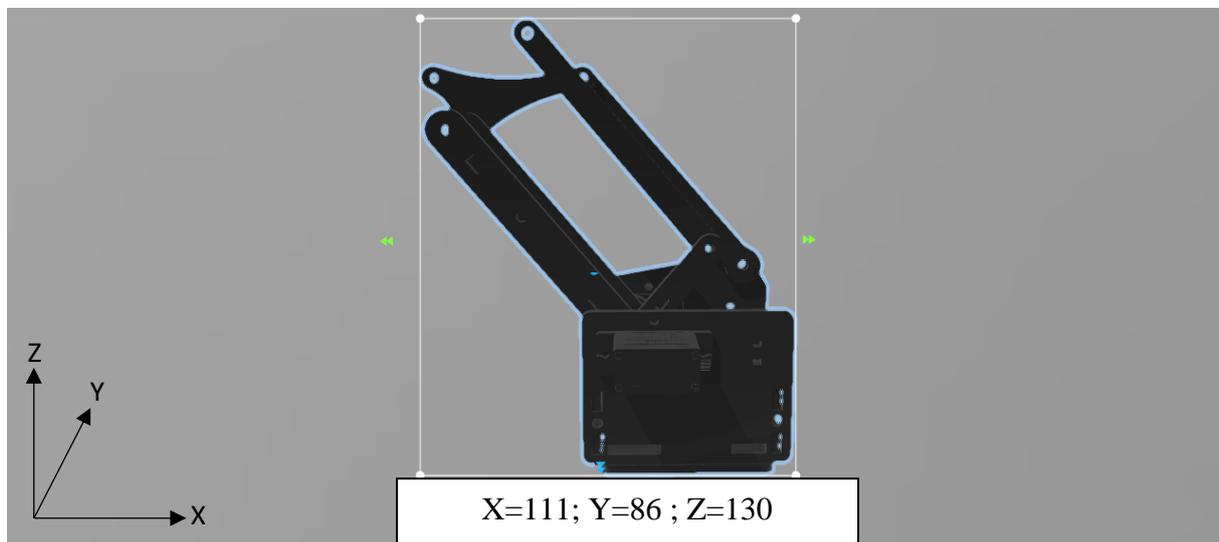
case 6:
  servo1PosSave[5] = pot1Angle;
  servo2PosSave[5] = pot2Angle;
  servo3PosSave[5] = pot3Angle;
  servo4PosSave[5] = pot4Angle;
  delay(1500);
break;
case 7:
  servo1PosSave[6] = pot1Angle;
  servo2PosSave[6] = pot2Angle;
  servo3PosSave[6] = pot3Angle;
  servo4PosSave[6] = pot4Angle;
  delay(1500);
break;
case 8:
  servo1PosSave[7] = pot1Angle;
  servo2PosSave[7] = pot2Angle;
  servo3PosSave[7] = pot3Angle;
  servo4PosSave[7] = pot4Angle;
  delay(1500);
break;
}
}
////////////////////////////////////
//si le bouton est enfoncé (HIGH), les servos on exécute les positions sauvegardées
if(digitalRead(button2) == HIGH){
  button2pres = true;
}
////////////////////////////////////
if(button2pres){
  for(int i=0; i<8; i++){
    servoB.write(servo1PosSave[i]);
    servoD.write(servo2PosSave[i]);
    servoG.write(servo3PosSave[i]);
    servoP.write(servo4PosSave[i]);
    delay(2000);
  }
}
delay(100);
}

```

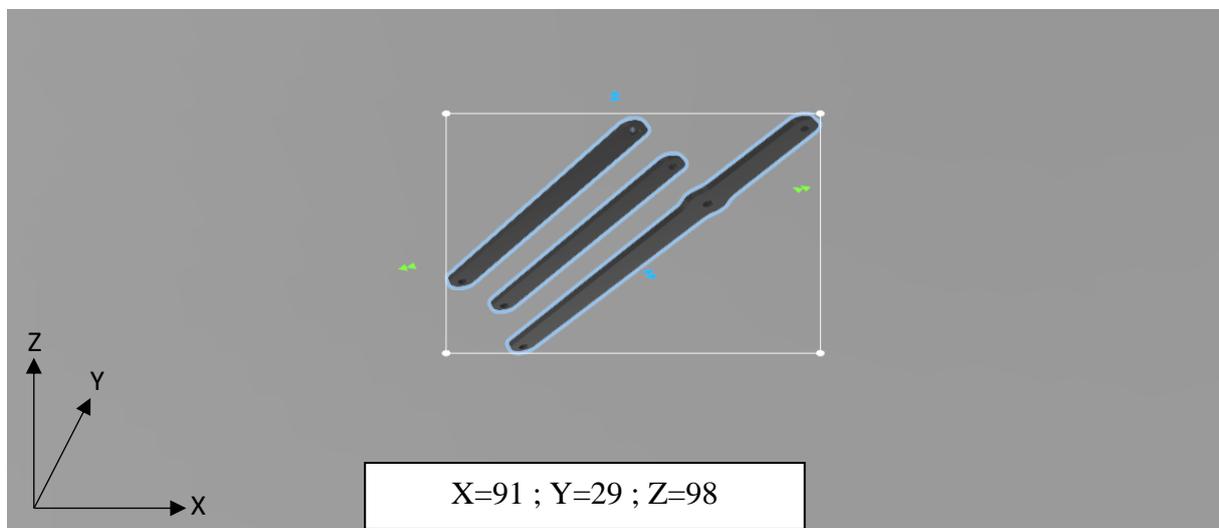
## Annexe E



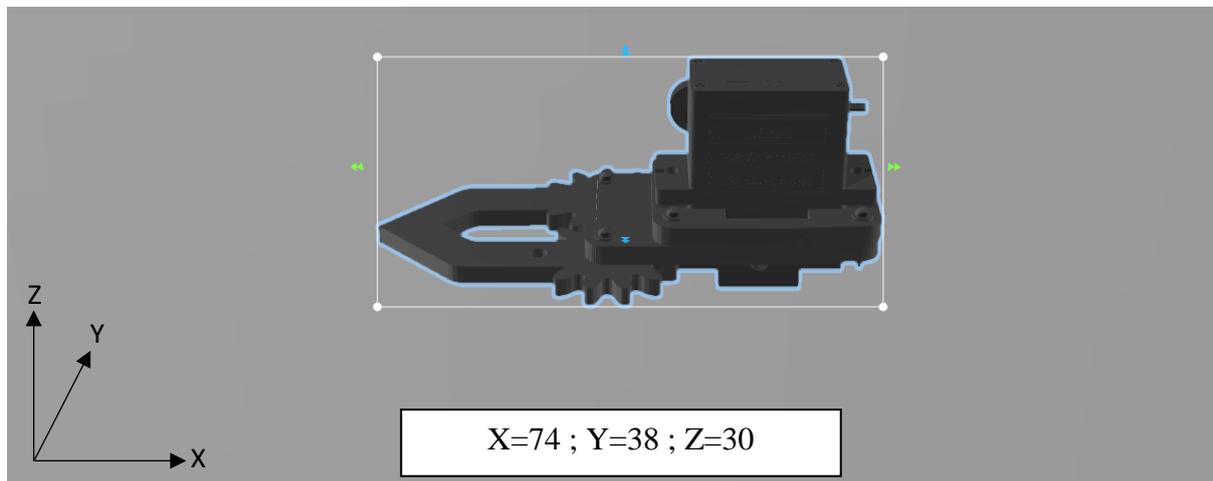
**Fig.E.1.** Dimensions de la base du bras manipulateur en millimètre



**Fig.E.2.** Dimensions du coude du bras manipulateur en millimètre



**Fig.E.3.** Dimensions de l'épaule du bras manipulateur en millimètre



**Fig.E.4.** Dimensions de la pince du bras manipulateur en millimètre

# RESUME

Ce projet de Master concerne l'étude et la réalisation d'un bras robot manipulateur à trois degrés de liberté à base d'une carte Arduino UNO. Après une exploration générale du domaine de la robotique, on a abordé le vif du sujet en étudiant les différents organes constituant notre manipulateur. Par la suite, on s'est penché sur sa modélisation et on a procédé à l'étude pratique et l'implémentation. Enfin, la simulation de la commande du bras robot par la commande PID est concrétisée avec SimMechanics (MATLAB SIMULINK). Pour la réalisation pratique, le châssis a été conçu par découpe laser. La programmation est faite à base du langage C++. Le contrôle du bras manipulateur se fait à l'aide d'une application mobile réalisée (sans fil) et grâce au shield incluant des potentiomètres.

Mots-clés : Bras Robotique ; Simulation ; Contrôleur PID ; Dynamique ; Cinématique ; Arduino.

## ABSTRACT

This Master project concerns the study and the achievement of a robot arm with three degrees of freedom, based on Arduino UNO board. After a review of the basics of robotics, different components constituting the manipulator have been studied and its different models have been established. The frame has been designed by laser cutting. After hardware achievement, different control modes have been implemented using C ++ code. Finally, the simulation of the control of the robot arm by PID was materialized with SimMechanics (MATLAB SIMULINK). The manipulator arm control can be performed using a mobile application (wireless) and thanks to the shield including potentiometers.

Keywords: Robot Arm; Simulation; PID control; Dynamics; Kinematics; Arduino.

## ملخص

يتضمن هذا العمل دراسة لذراع روبوت مكون من ثلاث درجات من الحرية. ولتحقيق ذلك وجب علينا تقسيم العمل إلى شطرين يتمثلان في دراسة نظرية وإدراك عملي.

الدراسة النظرية، يتم فيها عرض العاميات المتعلقة بأسس الروبوتات وخصائص مكونات الذراع المدروس.

تم تناولنا النماذج التي تميزه (الهندسية، الحركية والديناميكية). أخيراً، تم تجسيد المتحكم (PID) باستعمال SimMechanics(MATLAB SIMULINK)

الإدراك العملي قمنا فيه بتصميم الإطار عن طريق قطع الليزر. تم اجراء البرمجة استنادا إلى لغة البرمجة C++ وتم تنفيذها على البطاقة اردوينو اونو. التحكم في الذراع يتم باستخدام تطبيق أندرويد (لاسلكي) ومقياس الجهد الانزلاقي.

الكلمات المفتاحية: ذراع الروبوتات، المحاكاة، معرف المراقب، الحركة الديناميكية، الحركة المجردة، أردوينو