REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR ET DE LA RECHERCHE SCIENTIFIQUE

UNIVERSITE M'HAMED BOUGARA-BOUMERDES

Faculté de Technologie

Département Ingénierie des Systèmes Electriques

Mémoire de Master

Présenté par

Mlle.ABLAOUI NESRINE

Mlle .MERMOUCHI HOURIA

Filière : Electronique

Spécialité : Electronique des Systèmes Embarqués

Etude et Réalisation d'un bras manipulateur à six degrés de libertés commandé par une application Androïde avec Arduino

Soutenu le 02 /07 /2020 devant le jury composé de:

AKROUM	HAMZA	MCB	UMBB	Président
NAFA	FARES	MCB	UMBB	Examinateur
MESSAOUDI	NOUREDDINE	MCA	UMBB	Promoteur
NICHE	ZINEB	MASTER	UMBB	Co-promotrice

Année Universitaire : 2019/2020

Remerciement

On tient tout d'abord à remercier en premier lieu Allah, le tout puissant, de nous avoir donné autant de courage, de patience et de volonté pour accomplir ce mémoire.

On tient à exprimer à notre promoteur Dr. Messaoudi Noureddine , et à notre Co- promotrice Mlle Niche Zineb, notre profonde gratitude d'avoir accepté notre thème et de nous avoir suivit tout au long de notre parcours. ils nous ont permis de nous initier à la recherche et d'apprendre à surmonter les difficultés rencontrées. leur disponibilité, leur grande connaissance ont été un autant déterminant pour mener à bien ce travail.

Nos remerciements vont également à tous les enseignants de la Faculté de technologie de Boumerdes qui nous ont soutenu.

Nos vifs remerciements s'étendent à tous les membres de club scientifique ELECTRO pour leur soutien.

Notre reconnaissance respectueuse vas à Mr Brahimi Lyés et BenSmain BenSbeur pour leurs

aide si précieuse lors de notre travail.

Tous nos remerciements vont également aux membres de jury pour avoir fait l'honneur de juger ce modeste travail.

Nos remerciements vont également à toutes les personnes qui nous ont aider et soutenue de près ou de loin dans la réalisation de ce travail.

Dédicace

Aux êtres les plus chers au monde « MES PARENTS» Aucune dédicace ne saurait exprimer l'amour, l'estime, le dévouement et le respect que j'ai

Toujours eu pour vous.

Rien au monde ne vaut les efforts fournis jour et nuit pour mon éducation et mon bien être .

Ce travail est le fruit de vos sacrifices que vous avez consentis pour mon éducation et ma formation.

A mes très chers frères et sœurs

Les mots ne suffisent guère pour exprimer l'attachement, l'amour et l'affection que je porte pour vous.

Ali et sa femme Alexandra, Noura et son époux Dahmane, Sonia et Saadaoui.

Ainsi qu'à mes adorables nièces : Lina, Maissa et Leila. Que DIEU leurs offre le bonheur.

A tout mes enseignants qu'on aider et toujours pousser ver le haut

A ma très chère binôme Nesrine et sa famille

A tous les membres de ma famille Mermouchi et Berkati petits et grands

A tous mes oncles : Mohamed, Saïd, Omar, Hamid et Mahfoudh et leur épouses

A tous mes amis

Aux personnes qui m'ont toujours aidé et encouragé, qui étaient toujours à mes côtés.

A tous ceux qui m'aime .

Houria

Dédicace

Aux êtres les plus chers au monde « MES PARENTS» Aucune dédicace ne saurait exprimer l'amour, l'estime, le dévouement et le respect que j'ai

Toujours eu pour vous.

Rien au monde ne vaut les efforts fournis jour et nuit pour mon éducation et mon bien être .

Ce travail est le fruit de vos sacrifices que vous avez consentis pour mon éducation et ma formation.

A mon très chers frère Aziz

Les mots ne suffisent guère pour exprimer l'attachement, l'amour et l'affection que je porte pour toi

A tout mes enseignants qu'on aider et toujours pousser ver le haut

A ma très chère binôme Houria et sa famille

A mon chere grand père longue vie a lui

A tous les membres de ma famille Ablaoui et Haircheur petits et grands

A toutes mes tentes :Zina , Nadjet ,Warda ,Khadidja , Sakina et leurs maries

A tous mes oncles Tentes : Azzouz , Mouloud , Hocine ,Said , Youcef et Cherif et leurs épouses

A tous mes mes cousins petits et grands

A tous mes chere amis : BenSabeur BenSmain , Adlene Tazerouti et Nezli Berkat.

Aux personnes qui m'ont toujours aidé et encouragé, qui étaient toujours à mes côtés.

A tous ceux qui m'aime .

Nesrine

Résumé

الملخص

يتكون عملنا من دراسة وتحقيق ذراع مناور من ست درجات من الحرية ،يتم التحكم فيه بواسطة تطبيق الاندرويد مع اتصال البلوتوث. وهي عبارة عن هيكل ميكانيكي يتم التحكم فيه بواسطة دائرة إلكترونية تتكون من لوحة الاردوينو اونو ووحدة البلتوث HC -05 ، وقد استخدمنا واجهة رسومية تعتمد على برنامج ٣د بيلدر لعرض أجزاء الروبوت الخاص بنا. يتميز الروبوت الذي أنجزناه بهيكل مفصلي متين يتم تشغيله بواسطة محركات مؤازرة ، وقد صممنا برنامج تحكم يتكون من وضعيين يدوي وتلقائي ونفذناه على برنامج الاردوينو ادي.انجزنا عدة اختبارات على ذراع مناور ومراقبة النتائج من أجل إدراك التناسق بين الدراسة النظرية وممارسة الجهاز.

Résumé

Notre travail consiste a l'étude et la réalisation d'un bras manipulateur de six degré de liberté commander par une application Androïde avec une communication Bluetooth . C'est une structure mécanique contrôler par un circuit électronique composé d'une carte Arduino Uno et un module Bluetooth HC-05. On a utilisée une interface graphique basée sur le logiciel 3D Builder pour la visualisation des pièces de notre robot.

Le robot que nous avons accompli se caractérise par un structure articulée solide actionné par des servomoteurs .On a conçue un programme de contrôle composé de deux modes : Manuelle et automatique et on l'a exécuter sur le logiciel Arduino IDE .Pour terminer nous avons fait des testes sur le bras manipulateur et observer les résultats afin de percevoir la cohérence entre l'étude théorique et la pratique de la machine .

Abstract

Our work consists in the study and the realization of a manipulator arm of six degrees of freedom controlled by an Android application with Bluetooth communication. It is a mechanical structure controlled by an electronic circuit consisting of an Arduino Uno board and a Bluetooth HC-05 module. We used a graphical interface based on 3D Builder software to view the parts of our robot.

The robot that we have accomplished is characterized by a solid articulated structure actuated by servomotors. We designed a control program composed of two modes: Manual and automatic and we executed it on the Arduino IDE software. To finish we did some tests on the manipulator arm and observe the results in order to perceive the coherence between the theoretical study and the practice of the machine.

Sommaire

Remerciement……..…..…...2

Dédicace……..…...…3

Résumé……...…....5

Liste des figures…….....9

Liste des tableaux…….....12

Liste des symboles……...13

Liste des abréviations…….….14

Introduction générale……..……..……..... 15

Chapitre 1

Généralités sur les robots

1.1.Introduction……..……..16

1.2.Définition du terme robot……..……16

1.3.Constitution du robot…….….17

1.3.1. Unités opérationnelles……..…...17

1.3.2. Unités informationnelles……17

1.4.Les domaines de la robotiques……..…..18

1.5. Les types de robots……..…...20

1.5.1. Les robots mobiles……..……20

1.5.2. Les robots bras manipulateurs……..……..20

1.6. Architecture d'un bras manipulateurs…….....21

1.7. Classification des bras manipulateurs…….....22

1.8. Caractéristique des bras manipulateurs……..…..…..24

1.9. Notion de chaine cinématique……24

1.9. 1.Chaine cinématique série …….....24

1.9.2. Chaine cinématique absorbante…….....25

1.9.3. Chaine cinématique parallèle……..……26

1.9.3.1.Avantage des robots parallèle……...27

1.10. Conclusion……..….28

Chapitre 2

Principe physique et modélisation d'un bras manipulateur

2.1. Introduction …….29

2.2. Coordonnées homogènes ……..…..29
2.2.1. Représentation d'un point ……....29
2.2.2. Représentation d'une direction ……..30
2.2.3. Représentation d'un plan ……...30
2.3. Transformations homogènes ……30
2.3.1. Transformation des repères ……30
2.3.2. Transformation des vecteurs ……..31
2.3.3. Transformation d'un plan ……..32
2.3.4. Matrice de transformation de translation pure …….....33
2.3.5. Matrices de transformation de rotation autour des axes principaux……..33
2.3.5.1. Rotation θ autour de l'axe X …….....33
2.3.5.2. Rotation θ autour de l'axe Y……...34
2.3.5.3. Rotation θ autour de l'axe Z ……..….34
2.3.6. Propriétés des matrices de transformation homogènes ……35
2.4. Modèles géométrique des robots manipulateurs……..….35
2.4.1. Description de la géométrie des robots à structure ouverte simple ……..35
2.4.2. Modèle géométrique direct ……..….36
2.4.3. Modèle géométrique inverse ……37
2.5. Conclusion ……...38

Chapitre 3

Outils électroniques du bras manipulateur

3.1. Introduction …….……..…..…..39
1.2.Déscription de la carte Arduino Uno …….…….39
3.3. Caractéristiques de la carte arduino……..……...40
3.4. Description du module Bluetooth HC-05……40
3.5. Caractéristiques du module HC-05……..….…….…..41
3.6. Description du servo moteur…….....42
3.6.1. Description servomoteur SG90…….……42
3.6.2. Caractéristiques du servomoteur SG90……..….....43
3.6.3. Description servomoteur MG996R…….....43
3.6.4. Caractéristiques du servomoteur MG996R……....44
3.6.5. Alimentation……...45
3.6.6. Fils de connexion ……...45

3.7. Définition du logiciel Arduino IDE …….....46
3.8. Définition du logiciel App inventor ……....47
3.9. Définition du logiciel Proteus ……..……..……..48
3.9.1. Caractéristiques du logiciel……48
3.10. Définition du logiciel 3D Builder ……..……..……49
3.11. Conclusion..……..…..50

Chapitre 4

Réalisation et teste du bras manipulateur

4.1. Introduction ……...51
4.2. La partie hardware ……..…51
4.2.1. La conception 3D du bras manipulateur ……52
4.2.3.Visualisation 3D des pièces du bras manipulateur……..….52
4.2.3.1. Visualisation de la base ……...…52
4.2.3.2. Visualisation des articulations…….....52
4.2.3.3. Visualisation du préhenseur ……..…..54
4.2.4 .Description de l'assemblage de bras manipulateur …….....54
4.3. La partie Software ……..……55
4.3.1. Simulation des composants du bras avec la carte Arduino Uno Sous Proteus…..55
4.3.1.1. Câblage et simulation du Module Bluetooth HC-05 avec La carte Arduino ...56
4.3.1.2. Câblage et simulation de la carte Arduino avec les servo moteurs……..…..57
4.3.1.3. Câblage et simulation globales des composants du bras manipulateur ……....58
4.4. Les étapes de création de l'application Androïde ……...58
4.5. Le fonctionnement du bras manipulateur …….…..63
4.5.1. La partie programme ……...63
4.5.2. Partie Test du bras manipulateur ……..…68
4.6. Conclusion ……..….69
Conclusion générale ……..…..70
Références bibliographiques……...71
Annexe…….73

Listes des figures

Figure 1.1 :Evolution des robots……..16

Figure 1.2 :Interface de communication des robots……...18

Figure 1.3 ;Les domaines d'application des robots……18

Figure 1.4 :Robot mobile…….....19

Figure 1.5 : Robot de rééducation de marche ……19

Figure 1.6 :Robot industriel…….....19

Figure 1.7 :Robot domotique…….…..….19

Figure 1.8 :Robot de chirurgie médicale……..19

Figure 1.9 : Robot de militaire de détresse……...20

Figure 1.10 :Robot domestique…….20

Figure 1.11: Robot manipulateur ABB……..20

Figure 1.12 : Robot collaboratif KUKA……..….20

Figure 1.13 : Principe de fonctionnement de bras manipulateurs…….21

Figure 1.14 : Architecture d'un bras manipulateur…….....21

Figure 1.15 :Visualisation de l'organe terminale……..…..22

Figure 1.16 :Chaine cinématique série…….….25

Figure 1.17 :Chaine cinématique arborescente…….…...26

Figure 1.18 :Chaine cinématique parallèle……...27

Figure 2.1 : Représentation d'un point……..…...30

Figure 2.2 :Transformation de repères……..31

Figure 2.3 : Transformation d'un vecteur de repère Ri à Rj……..……32

Figure 2.4 : Transformation d'un vecteur du repère Ri à Rj après transformation d'un point……...32

Figure 2.5 : Matrice de translation pure…….33

Figure 2.6 : Transformation de rotation pure autour de l'axe X……...….33

Figure 2.7 :Transformation de rotation pure autour de l'axe Y…….…….34

Figure 2.8 : Matrice de rotation autour de Z…….....….34

Figure 2.9 :Structure ouverte simple du bras manipulateur……...….36

Figure 2.10 : Paramètres géométrique dans le cas d'une structure ouverte simple……37

Figure.3.1 :La carte Arduino Uno 3 ……...39

Figure 3.2 : Le module Bluetooth HC-05…….41

Figure 3.3 :Le servomoteur SG90……42

Figure 3.4 : Le servomoteur MG996R……43
Figure.3.5 :Alimentation 9V……....44
Figure 3.6 :Les fils de connexion pour le bras manipulateur…….45
Figure 3.7 : Bras manipulateur avec les composants associer …….....….45
Figure. 3.8 : Le logiciel de programmation Arduino IDE…..…….……...…..46
Figure.3.9 :Logiciel de création d'application Android App inventer……..…....47
Figure 3.10 :Le logiciel de création de schéma électronique Proteus…….…...49
Figure 3.11 :Le logiciel 3D Builder……..….49
Figure 4.1 :Présentation des pièces du bras manipulateurs assemblées…….....51
Figure 4.2 :La base en 3D…….……52
Figure 4.3 :La base réel……..……...…52
Figure 4.4.'articulation 1 en 3D……53
Figure 4.5 :L'articulation 2 en 3D…….……...53
Figure 4.6 :L'articulation 3 en 3D…….……...……53
Figure 4.7 :L'articulation 4 en 3D…….…...53
Figure 4.8 :L'articulation 1 réel…….…..53
Figure 4.9 :L'articulation 2 réel…….….53
Figure 4.10 :L'articulation 3 réel……..…….….54
Figure 4.11 :L'articulation 4 réel……...……..……....54
Figure 4.12 . Le préhenseur en 3D…….…….....54
Figure 4.13 :Le préhenseur réel ……..……54
Figure 4.14 :l'assemblage de bras manipulateur en 3D…….……55
Figure 4.15 :Image réel l'assemblage de bras manipulateur……55
Figure 4.16 :Nouveau projet schématique sur le logiciel Proteus ……..…..…56
Figure 4.17 :Câblage de la carte Arduino Uno avec le module Bluetooth HC-05…….....56
Figure 4.18 :Câblage de la carte Arduino Uno avec le module Bluetooth HC-05……57
Figure 4.19 : Câblage globale des composants sous Proteus……..…57
Figure 4.20 :Fenêtre initial de l'écran de l'application dans logiciel App inventor …...59
Figure 4.21 :Ecrans modifiée la parie de connexion Bluetooth et servomoteurs …….…60
Figure 4.22 :Ecran modifiée des modes de fonctionnement de l'application 60
Figure 4.23 .Blocs de programme de connexion Bluetooth……61
Figure 4.24 :Bloc de programme des mouvements des servomoteurs dans le logiciel 62
Figure 4.25 :Bloc des modes de fonctionnement de l'application Androïde……63
Figure 4.26 :Déclarations des variables des composants Arduino IDE….....64

Figure 4.27 : initialisations des servomoteurs et module Bluetooth…….....64
Figure 4.28: lecture et stockage des donnes entrante de l'application Androïde…….....65
Figure 4.29 : Vérification des préfix des données entrantes……...65
Figure 4.30 : Contrôle de vitesse des servomoteurs……..…...65
Figure 4.31 : Sauvegarde des nouvelles positions des servomoteurs…….…66
Figure 4.32 : Exécution des positions enregistrer en boucle ……..…66
Figure 4.33: Remise a zéro des données stocké dans le tableau…….…67
Figure 4.34:Câblage du bras manipulateur avec les composants électroniques……...67
Figure 4.35 : Algorithme de fonctionnement du bras manipulateur 68

Liste des tableaux

Table 1.1 :les articulation du bras manipulateurs……..….22

Table 1.2 :Classification géométrique des bras manipulateurs……..23

Table 3.1 : Caractéristiques de la carte arduino……..40

Table 3.2 :Spécification du module Bluetooth HC-05……..…41

Table 3.3 : Caractéristiques du servo moteur SG90……43

Table 3.4 :Caractéristiques du servomoteur MG996R……44

Liste des symboles

- C0 : Corps initiale.
- Cj : Corps antécédent
- Cn : Corps qui porte l'organe terminal.
- Di : Distance entre zi-1 et zi.
- iTj : Matrice de transformation de repère.
- i V1: Vecteur de l'axe i.
- j V1: Vecteur de l'axe j.
- n : Nombre de corps.
- Px: Coordonnées cartésiennes de x.
- Py: Coordonnées cartésiennes de y.
- Pz: Coordonnées cartésiennes de z.
- Q: Vecteur ligne.
- QV : Produit matriciel
- Qj : Variable articulaire.
- ri : distance entre xi-1 et xi.
- Ri: Repère de l'axe i.
- Rj: Repère de l'axe j.
- Rot : Matrice de rotation.
- V: Point dans l'espace.
- Xi : Axe X.
- Yi : Axe Y.
- Zi : Axe Z.
- αi : Angle de l'axe z autour de x.
- θi : Angle de l'axe x autour de z.
- σi : Indique le type d'articulation .

Liste des abréviations

- App :Application .
- CAO : Conception assistées par ordinateur .
- EEPROM: Electrically Erasable Read Only Memory .
- GND : Ground .
- Hex : Hexadecimal .
- IDE : Integrated Developement Envirement .
- MAC:Macintosh .
- MGI: Model Géométrique Indirect .
- MHz : Mega Hertz.
- LED: Light Emitting Diode .
- OBJ: Object file .
- PMW: Pulse modulation width .
- PC: Portable Computer.
- PCB: Printable Circuit Board .
- RX: Receiving pin .
- SPP: Serial Port Protocol .
- SRAM: Static Random Access Memory .
- STL: Stereolithographie.
- TX: Transmitting pin.
- VCC: Voltage Common Collector .

Introduction générale

Un robot est définit comme étant un appareil automatique capable de manipuler des objets ou d'exécuter des opérations répétitive et parfois difficile pour l'homme, et cela s'effectue selon un programme fixe ou modifiable [1].

La première apparition du terme robot était dans le dix-neuvième siècle dans une pièce de théâtre Tchèque qui signifie : « corvée ». Le besoin de faire plusieurs taches simultanément et augmenter le rendement de productions dans les industries a fait que les robots prennent des tailles et des formes différentes tel que les bras manipulateurs, ces machines se caractérise par une modélisation géométriques et et leurs combinaison nous produit une machine programmable efficace et autonome, et la commande de ces machine à nécessiter le développement de plusieurs applications pour un fonctionnement plus efficace.

L'objectif de notre travaille est de réaliser un bras manipulateur de six degrés de liberté commander par une application Androïde capable de manipuler des objets selon un programme .Le fonctionnement de notre bras manipulateur se base sur la communication sans fil Bluetooth. Pour atteindre notre objectif et bien comprendre la structure et le comportement des bras manipulateurs, dans ce mémoire nous allons traiter quatre chapitres principaux qui sont:

- Le premier chapitre présente d'une manière générale les robots et la robotique et leur différents domaines d'application comme : la médecine ou militaire, les types qui existes dans le monde puis on s'intéressera aux bas manipulateurs en particulier, en décrivant leur caractéristiques qui les favorise le plus.
- Le deuxième chapitre porte sur les modèles géométriques d'un bras manipulateur, ainsi que toutes les matrices de transformations nécessaires pour la modélisation du robot.
- Le troisième chapitre présente les différents composants, utiliser a la création dans notre bras manipulateur, ainsi que les logiciels pour le commande : Proteus, App inventor, Arduino IDE et 3DBuilder pour la visualisation des pièces en trois dimensions.
- Le dernier chapitre présentera la visualisation de la conception des différentes articulations du bras manipulateur sous le logiciel 3DBuilder et l'assemblages des parties sous Proteus et aussi la création de l'application Androïde, à la fin nous avons effectuer des testes de fonctionnement .

Chapitre 1

Généralités sur les robots

1.1. Introduction

La robotique depuis sa découverte a toujours eu pour objectif de développer des technologies nouvelles au service de l'homme, de rendre les robots proches de nous, fonctionnels et sociale.

On appelle un robot tout objet automatique programmable constituer d'une partie matérielle et logiciel , et qui répond au besoin de l'utilisateur, il est apparu la première fois vers 1920 dans une pièce de théâtre du tchèque Karel Čapek ou il désignait des petits être artificielles anthropomorphes répandant parfaitement aux ordres de leurs maitre (robota signifie travail en tchèque), il a évolué au fil du temps et a pris des formes et des tailles différentes selon le besoin de l'homme [2].

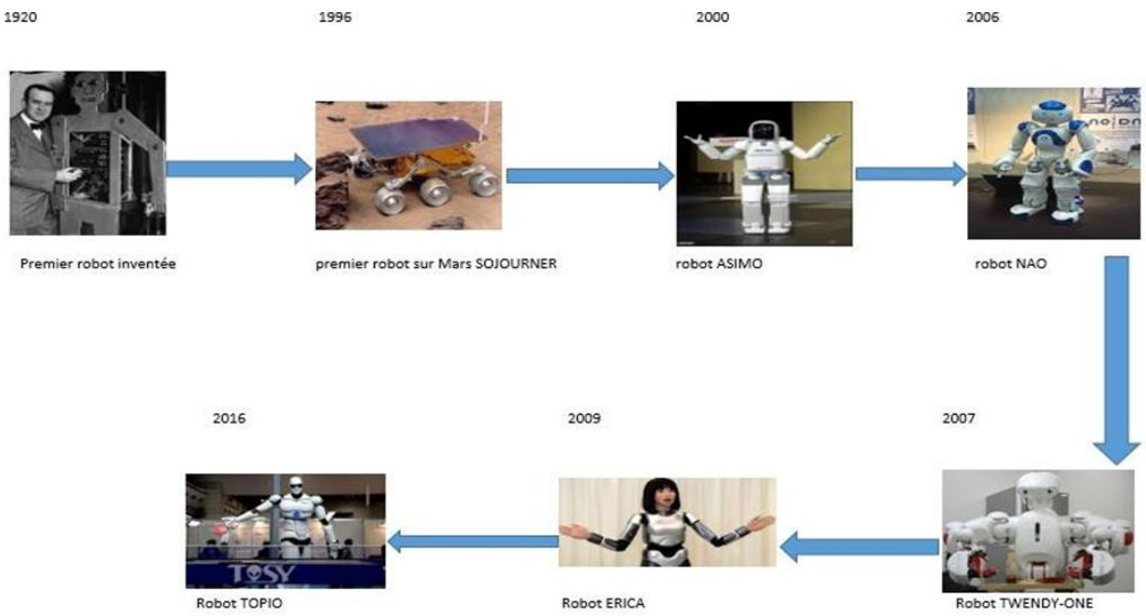


Figure 1.1 : Evolution des robots.

1.2. Définition du terme robot

Appareil automatique capable de manipuler des objets ou d'exécuter des opérations selon un programme fixe, modifiable ou adaptable [1].

1.3. Constitution d'un robot

Un robot est un assemblage complexe de pièces mécaniques et de pièces électroniques, le tout pouvant être piloté par une commande intelligente disposée comme suit :

1.3.1. Unité opérationnelle

- C'est la partie commande du robot et elle se constitue de plusieurs éléments comme :

- **Une source d'énergie** : elle est embarquée, généralement une batterie d'accumulateurs électriques.
- **Des capteurs** : Ils donnent une information à propos de l'environnement ou des composants internes (position d'un moteur ou d'un vérin, état d'une LED). Cette information est utilisée pour calculer l'ordre approprié à envoyer aux actionneurs.
- **Des actionneurs** : Ce sont des organes qui transforment l'énergie qui leur est fournie en un phénomène physique utilisable comme des mouvements.

1.3.2. Unité informationnelle

Elle représente tous les systèmes d'information du robot, composée de systèmes principaux qui sont :

- **L'ordinateur embarqué** : composé généralement des microprocesseurs à basse consommation d'énergie ou des microcontrôleurs. Il dispose d'un système d'exploitation qui facilite la gestion des tâches à effectuer. Plusieurs programmes peuvent être exécutés simultanément.

- **L'interface de communication** : elle possède également un transformateur d'information : envoyer et recevoir des informations d'après les interfaces entrée / sortie (liaison filaire, Sans fil, Ecran tactile...) comme il est indiquer dans la figure suivante [3] :

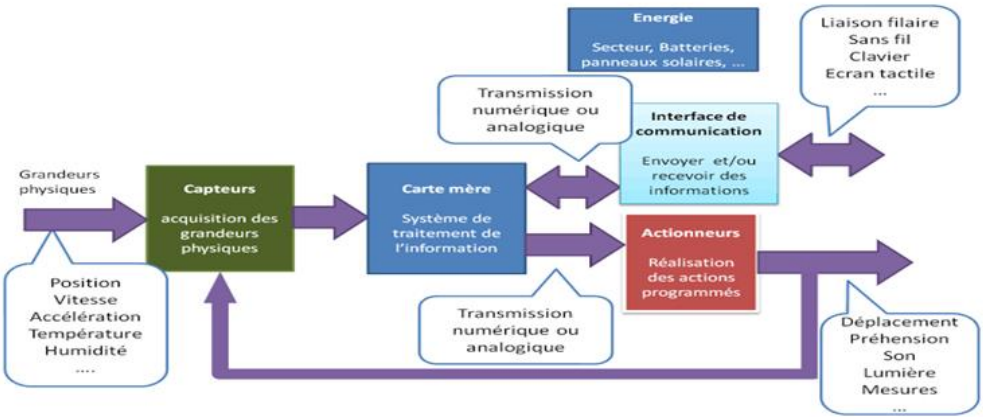


Figure 1.2: Interface de communication des robots.

1.4. Les domaines de la robotique

✓ Les robots sont utilisés dans une grande variété de domaine les plus importants comme la médecine sont présentés comme suit :

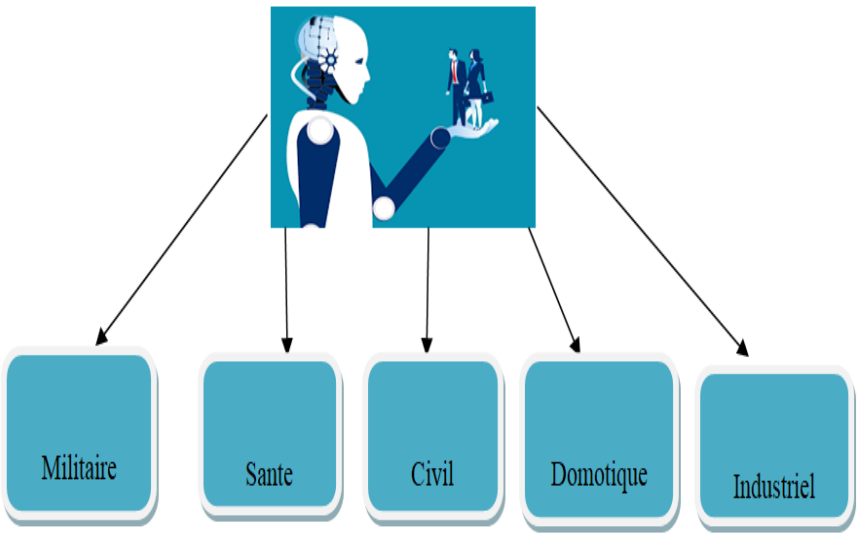


Figure 1.3 : Les domaines d'application des robots.

Figure 1.4 : Robot industriel.

Figure 1.5 : Robot domotique.

Figure 1.6 : Robot de chirurgie médicale.

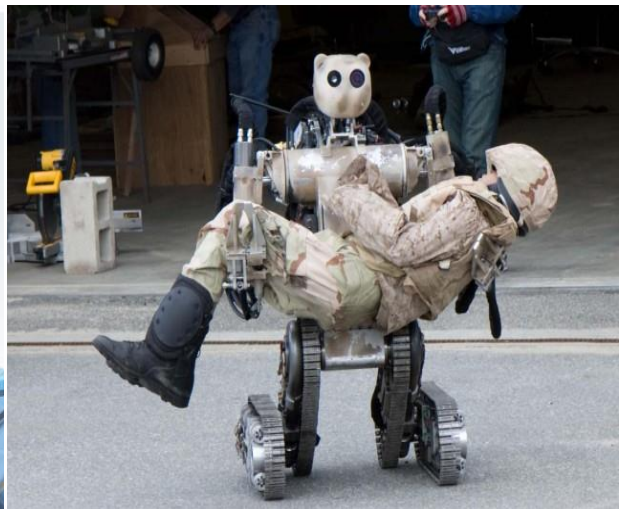


Figure 1.7 : Robot de détresse militaire.

Figure 1.8: Robot domestique.

1.5. Les types de robots

- Il existe plusieurs types de robots ils sont classe comme suit :

1.5.1 Les robots mobiles

Ce sont des robots capables de se déplace dans un environnement bien déterminé, ils sont équipée ou non de manipulateur suivent leur utilisation (les robots explorateur, les robots de services, les robots domestiques …).

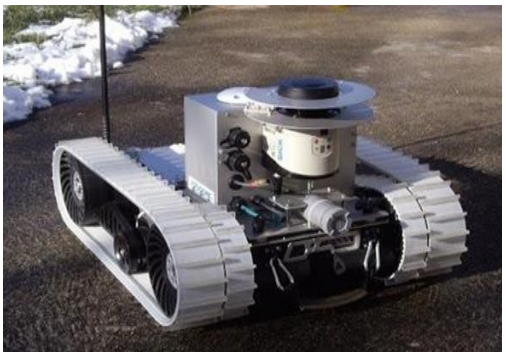


Figure 1.9 : Robot mobile.

Figure 1.10 : Robot de rééducation de la marche.

1.5.2. Les robots bras manipulateurs

C'est des robots ancrés physiquement à leur espace de travail et généralement mis en place pour réaliser une tache précise. Ce sont des manipulateurs automatiques programmés qui se substituent à l'homme pour l'accomplissement de tâches répétitives (fermeture de boites de médicament, pressage, …ect)[3] :

Figure 1.11 : Robot manipulateur ABB.

Figure 1.12 : Robot collaboratif KUKA.

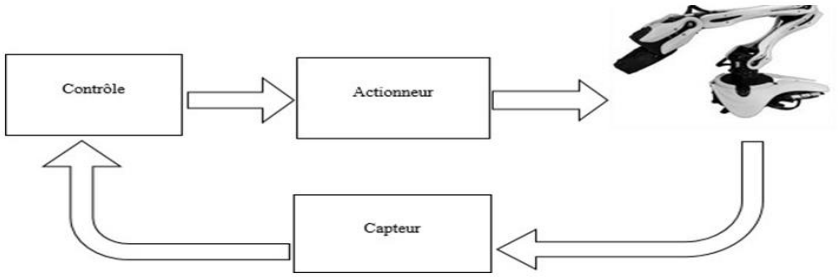


Figure 1.13*:* Principe de fonctionnement du bras manipulateur.

1.6. Architecture d'un bras manipulateur

L'architecture d'un bras manipulateur est très essentielle pour déterminer le fonctionnement de celui-ci elle se décompose comme suit :

- ✓ **La base :** La base du manipulateur est fixée sur le lieu du travail. Ceci est le cas de la quasi-totalité des robots industriels.
- ✓ **Le porteur :** Le porteur représente l'essentiel du système mécanique articulé (segment, articulation, actionneur, organe terminal), il a pour rôle d'amener l'organe terminal dans une situation imposée.
- ✓ **Le Segment :** Corps solides rigides susceptibles d'être en mouvement par rapport à la base du porteur, et les uns par rapport aux autres [3] :

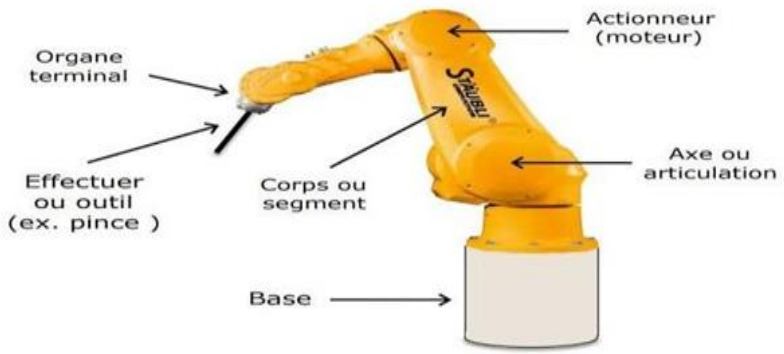


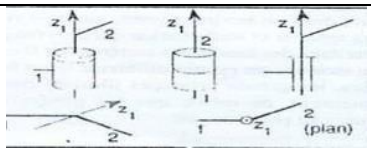
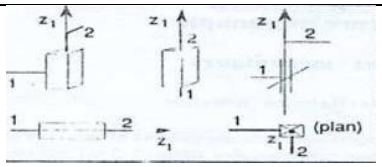
Figure 1.14 : Architecture du bras manipulateur.

- ✓ **L'articulation :** Une articulation lié deux corps successifs en limitant le nombre de degré de liberté, de l'un par rapport à l'autre.
- ✓ **Articulation rotoïde :** Il s'agit d'une articulation de type pivot, notée R, réduisant le mouvement entre deux corps à une rotation autour d'un axe commun.

✓ **Articulation prismatique :** 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 [2].

-Ces deux articulations sont représenter par des transformations pour bien les visualiser comme suite :

Tableau 1.1 : Les articulations du bras manipulateur.

Liaison	Schéma	Mouvement de 2\1	D D L
Rotoïde (Pivot)		Rotation autour de l'axe z1 de vitesse angulaire ϖz1.	1
Prismatique		Translation autour de l'axe z1 de vitesse Vz1.	1

✓ **L'organe terminal :** 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 [3].

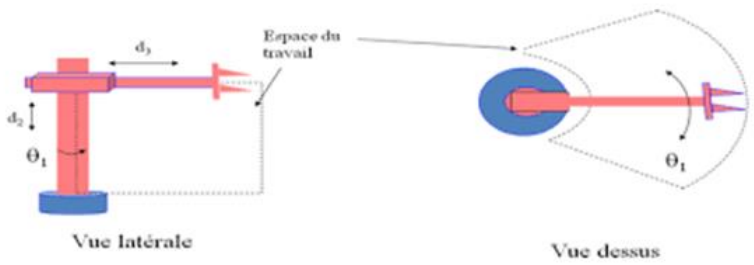


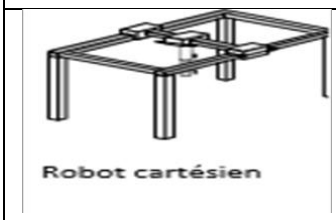
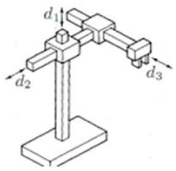
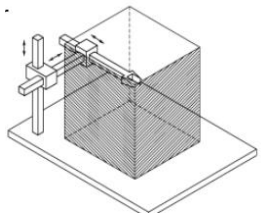
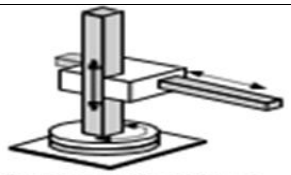
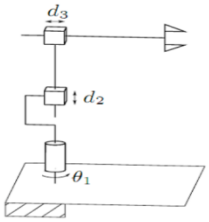
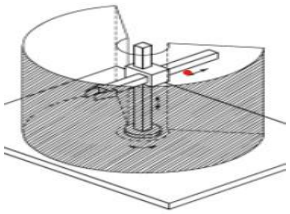
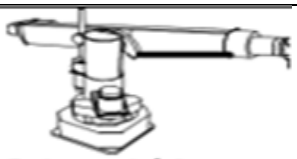
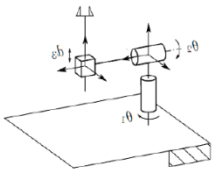
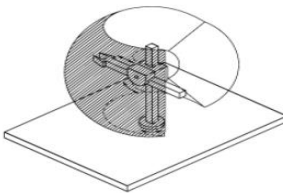
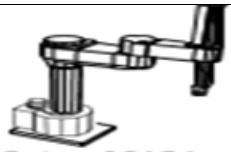
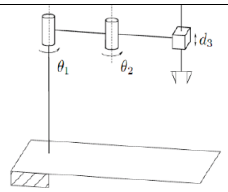
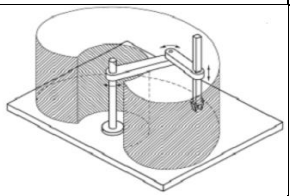
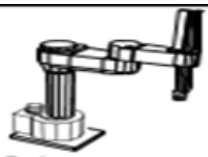
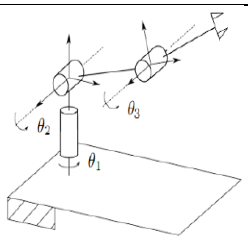
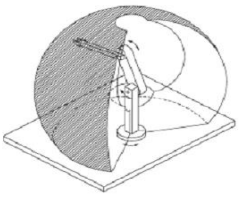
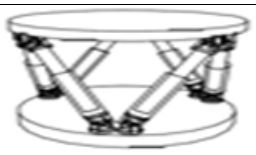
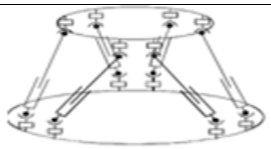
Figure 1.15 : Schématisation de l'organe terminale.

1.7.Classification des bras manipulateurs

Il y a plusieurs classifications qui existent pour définir un bras manipulateur mais la plus considérable est la classification géométrique selon les articulations rotoïde ou prismatique, la différence d'un bras manipulateur a un autre consiste dans l'ordre des articulations on distingue six bras manipulateurs les plus répondus qui se présente

comme suit :

Tableau 1.2: Classification géométrique des bras manipulateurs.

Type de robot	Structure cinématique	Espace de travail	Photo
 Robot cartésien			
 Robot cylindrique			
 Robot sphérique			
 Robot SCARA			
 Robot Anthropomorphe			
 Robot parallèle			

1.7. Caractéristiques du bras manipulateur

-On distingue le robot bras manipulateur des autres robots par ses caractéristiques qui lui sont propre et on les définit comme :

- La charge maximale transportable (de quelques kilos à quelques tonnes), à déterminer dans les conditions les plus défavorables.
- L'architecture de la structure mécanique articulée. Le choix est guidé par la tâche à réaliser.
- L'espace de travail est le volume de l'espace que le robot peut atteindre avec toutes les orientations possibles de l'effecteur (organe terminal). Cet espace de travail est un sous- ensemble de l'espace de travail maximal.
- La vitesse de déplacement (vitesse maximum en élongation maximum).
- La masse du robot.
- Le coût du robot.
- La maintenance [4].

1.8. Notion de chaine cinématique

Une chaîne cinématique est un ensemble de pièces liées par des liaisons mécaniques. On appelle chaîne cinématique simple toute chaîne cinématique où chaque pièce a une ou deux liaisons seulement. On appelle chaîne cinématique complexe toute chaîne cinématique Contenant une pièce (ou plusieurs pièces) ayant trois liaisons Les chaînes peuvent être soit ouvertes ou en série (tous les corps ont au plus deux liaisons), arborescentes (au moins l'un des corps a plus de deux liaisons) ou fermées (l'organe terminal est relié à la base du mécanisme par plusieurs chaînes).

1.8.1. Chaine cinématique série

Une chaîne cinématique simple ouverte est une succession de solides placés en série et liés entre eux un à un par des liaisons simples. Autrement dit lorsqu'une pièce (n) est liée par l'intermédiaire de $n-1$ pièces placées en série et si : $n-1$ pièces sont liées entre eux par des liaisons simples alors ces n pièces et n liaisons constituent une

chaîne cinématique simple ouverte ou des liaisons en série.

Les robots à chaine cinématique série ont pour avantage de disposer d'un grand volume de travail et d'être relativement simples sur le plan des calculs liés à leur commande .Leurs principaux défauts sont :

- Inertie élevée due aux masses réparties sur toute la chaîne cinématique (actionneurs, organes de transmission)
- Manque de rigidité par la mise en série d'éléments élastiques
- Fatigue et usure des liaisons de puissance assurant l'alimentation des actionneurs (câbles, tuyaux flexibles)
- Fatigue et usure des liaisons assurant la circulation des informations entre les capteurs et la commande, sur le plan sécurité, ce point est essentiel puisqu'une erreur de transmission peut avoir des conséquences désastreuses sur les mouvements du robot.

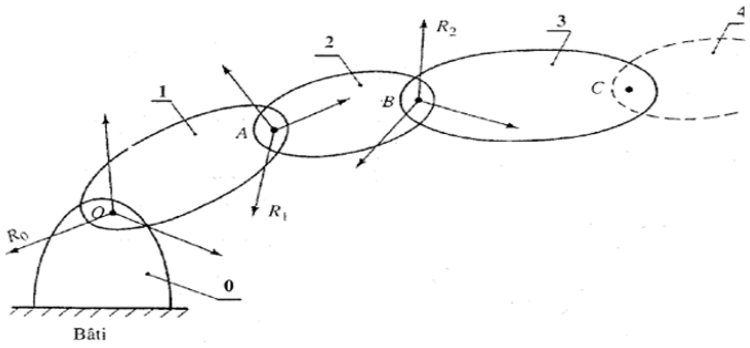


Figure 1.16 : Chaine cinématique série.

1.8.2. Chaine cinématique arborescente

Une chaîne cinématique arborescente est constituée par $n+1$corps, n articulations et '***r***' organes terminaux. Par convention, les corps et articulations sont numérotés de la manière suivante :

- La base est fixe et constitue le corps C_0.
- Les numéros des corps et articulations sont croissants sur chaque branche en partant de la base (corps C_0.), vers un organe terminal.
- Le corps C_j est articulé autour de l'articulation.
- Représente le corps antécédent sur la chaîne menant au corps Cj. en partant de la base.

- Un robot à chaîne cinématique arborescente est constitué de plusieurs chaînes cinématiques ouvertes reliées à un corps central.

En fonctionnement, un robot à chaîne cinématique arborescente peut devenir un robot à chaîne cinématique fermée par la saisie d'une pièce avec deux bras ou le contact avec le sol de deux ou plusieurs pattes.

Ce type de structure est davantage utilisé pour les robots de service (robots marcheurs, robot humanoïde) mais depuis les années 2010 des robots manipulateurs industriels à deux bras sont proposés pour réaliser des tâches particulièrement complexes.

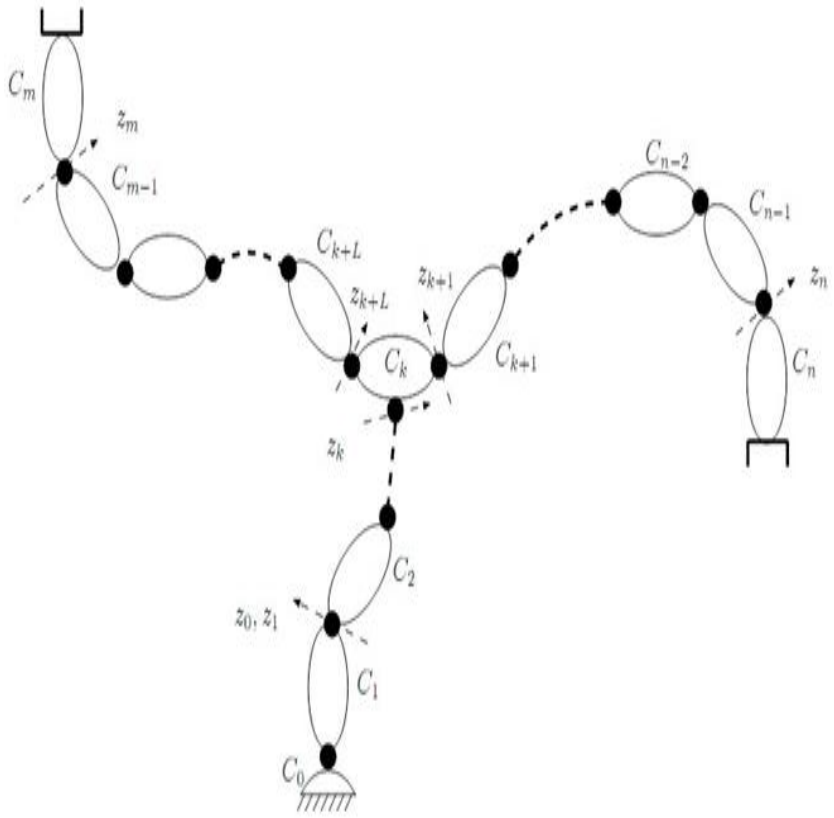


Figure 1.17 : Chaine cinématique arborescente.

1.8.3. Chaine cinématique parallèle

- Un robot parallèle est un manipulateur, en chaîne cinématique fermée, constitué d'un organe terminal à n degrés de liberté et d'une base fixe, reliés entre eux par des chaînes cinématiques indépendantes.
- Un ingénieur dénommé Pollard fut le premier à déposer un brevet de mécanisme pour peindre automatiquement les carrosseries de voitures en 1938.

- C'est la première idée de robot industriel (à structure parallèle). Le concept ne put aboutir faute de moyens électroniques et informatiques adéquats pour le commander. Le mécanisme (que l'on nomme maintenant Tripode) comprenait trois chaînes cinématiques. Les mobilités d'orientations étaient assurées par un poignet à trois degrés de liberté en série avec la structure parallèle.

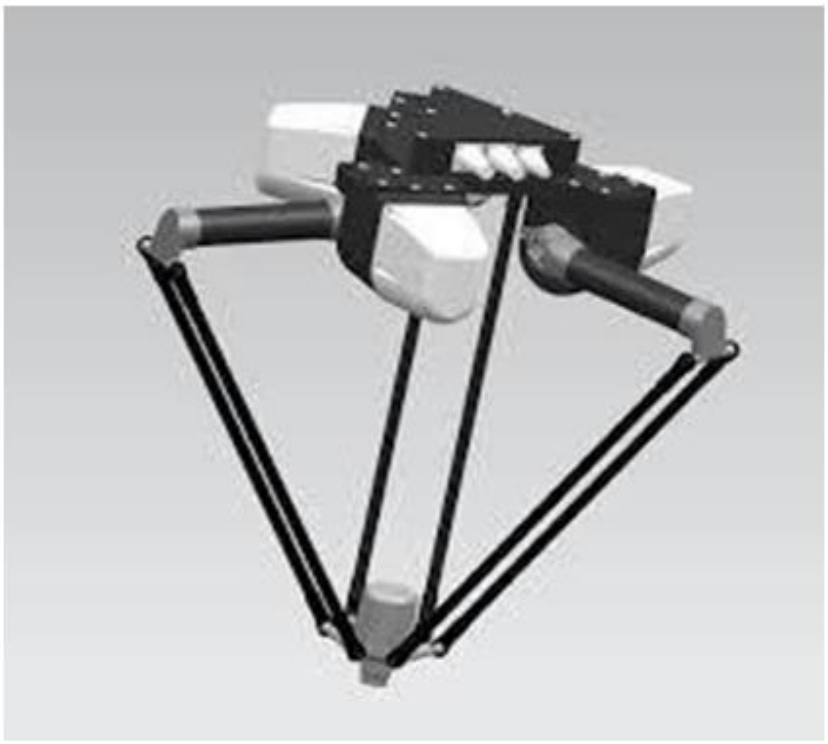


Figure 1.18: Chaine cinématique parallèle.

1.9.3.1. Avantages des robots parallèles

La mise en parallèle de plusieurs chaînes cinématiques entraînées chacune par un actionneur conduit généralement aux avantages suivants :

- Capacité de charge élevée,
- Possibilité de mouvements à haute dynamique (accélérations élevées),
- Rigidité mécanique élevée.
- Fréquence propre élevée, donc peu d'erreur de répétabilité due à une oscillation incontrôlée de la structure mobile,
- Grand choix de moteurs et de réducteurs par le fait que leur masse joue peu de rôle

dans l'inertie du manipulateur [5].

1.10.Conclusion

Dans ce chapitre nous avons traité les différents aspects de la robotique et des robots, passant par la description des différents types de robots ainsi que leurs domaines d'application de ces derniers, On a classé les bras manipulateurs selon la classification géométrique et citer les chaines cinématiques vue de leur importance dans l'élaboration du bras manipulateur, dans le chapitre qui suit nous allons introduire la modélisation du robot manipulateur.

Chapitre 2

Principe physique et modélisation d'un bras manipulateur

2.1. Introduction

En robotique, à chaque élément du poste de travail est associé un ou plusieurs repères. Ces repères sont généralement définis de telle sorte que leurs axes et leurs origines correspondent respectivement à des directions et à des points privilégiés ayant un rôle fonctionnel lors de l'exécution de la tâche.

La notion de changement de repère est donc fondamentale. Elle permet :

- D'exprimer les déplacements relatifs des différents éléments du mécanisme articulé les uns par rapport aux autres.
- De spécifier les situations successives que doit prendre le repère associé à l'organe terminal du robot pour réaliser une tâche donnée, ainsi que les vitesses correspondantes.
- De décrire et de contrôler les efforts mis en jeu lorsque le robot interagit avec son environnement.
- D'intégrer à la commande les informations sensorielles issues des capteurs ayant chacun leur système de référence propre [6].

2.2. Coordonnées homogènes

2.2.1. Représentation d'un point

En coordonnées homogènes, la représentation d'un point se fait par un vecteur colonne à quatre composantes.

Les trois premières sont les coordonnées cartésiennes du point, tandis que la composante supplémentaire est considérée comme un facteur d'échelle, égal à un en robotique, nous avons alors [6] :

$$P = \begin{bmatrix} px \\ py \\ pz \\ 1 \end{bmatrix}$$

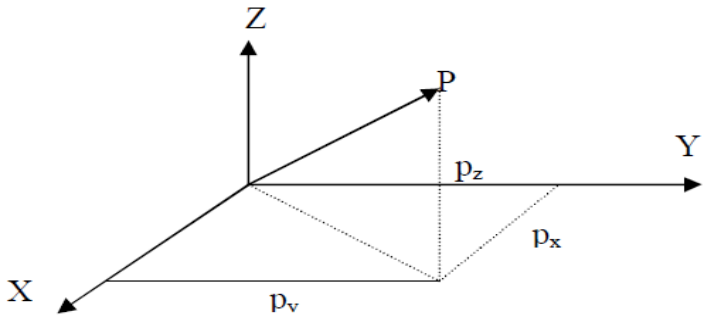


Figure 2.1. Représentation d'un point [6].

2.2.2. Représentation d'une direction

La représentation se fait par quatre composantes, trois coordonnées cartésiennes et la quatrième est nulle, soit [6] :

$$U = \begin{bmatrix} ux \\ uy \\ uz \\ 0 \end{bmatrix}$$

2.2.3. Représentation d'un plan

- ✓ Le plan $\alpha x + \beta y + \gamma z + \delta = 0$ est représenté par un vecteur ligne Q :
  $Q = [\alpha\ \beta\ \gamma\ \delta]$
- ✓ Pour tout point V appartenant au plan Q le produit matricielQV est nul :

$$QV = [\alpha\ \beta\ \gamma\ \delta] \begin{bmatrix} vx \\ vy \\ vz \\ 0 \end{bmatrix} = 0 \qquad (2.1)$$

2.3. Transformations homogènes

2.3.1. Transformation des repères

-La transformation (translation ou rotation) de repère Ri, vers le repère Rj se fait comme indique la figure suivante :

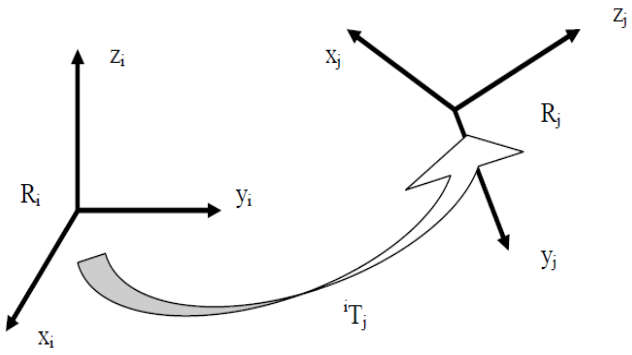


Figure.2 .2. Transformation de repère [6].

Cette transformation est définie par la matrice iTj appelée matrice de transformation homogène, de dimension (4x4), telle que :

$$^{i}Tj = [^{i}sj\ ^{i}nj\ ^{i}aj\ ^{i}pj] = \begin{bmatrix} sx & nx & ax & px \\ sy & ny & ay & py \\ sz & nz & az & pz \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (2.2)$$

Où ^{i}sj ,^{i}nj et ^{i}aj désignent les vecteurs unitaires suivant les axes xj, yj et zj respectivement du repère Rj exprimés dans le repère Ri et $^{i}p_j$ représente l'origine du repère Rj exprimée dans le repère Ri.

On peut dire également que la matrice iTj définit le repère Rj dans le repère Ri. Elle peut être écrite sous la forme suivante [6].

$$iTj = \begin{bmatrix} Ai & & & Pi \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} sj & nj & aj & pj \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (2.3)$$

2.3.2. Transformation des vecteurs

Le vecteur $jV1$ définissant le point $P1$ dans le repère , le vecteur $jV1$ peut être exprimé dans le repère Ri , ce qui revient à calculer les coordonner du point $P1$ dans Ri.

Deux cas sont à considérer [4] :

a)- La matrice de transformation iTj permet d'exprimer, dans le repère , les coordonnées d'un vecteur donné dans le repère Rj.

$$iV_1 = iTj\ .jV_1 \qquad (2.4)$$

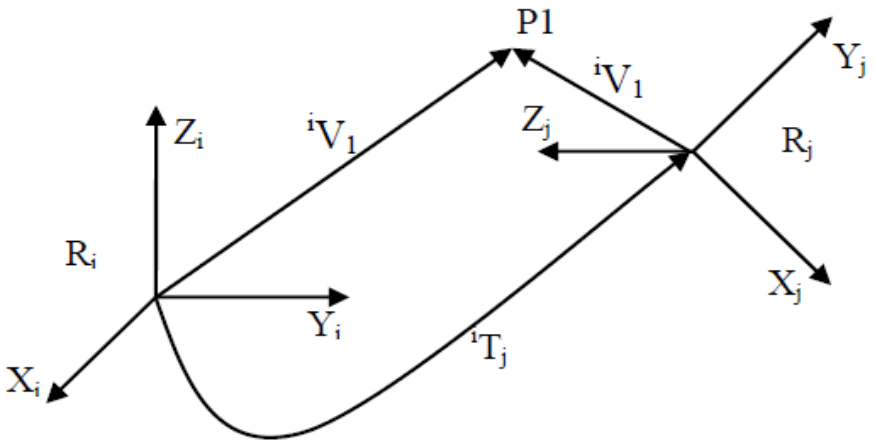


Figure 2.3. Transformation d'un vecteur de repère Ri à Rj[6].

b)- Faisons subir une transformation quelconque iTj au vecteur iV1, transformation qui amène le point P1 en P2.

$$iV_2 = iTj.jV_2 \qquad (2.5)$$

- on a iV1 = jV2 (égalité en terme de composantes), alors on peut écrire :

$$iV_2 = iTj.iV_2 \qquad (2.6)$$

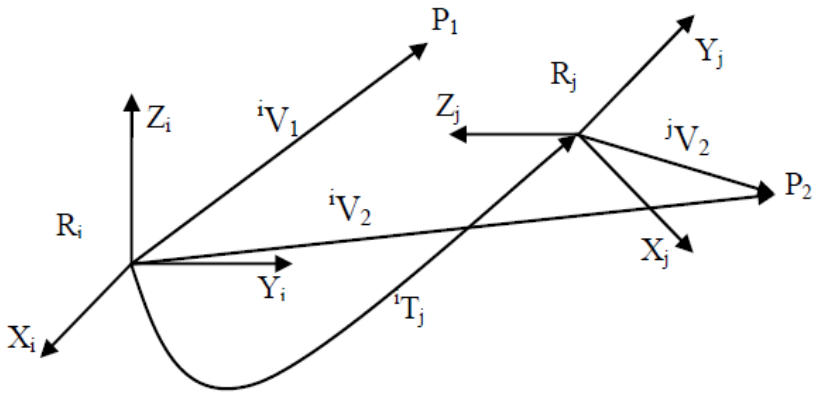


Figure. 2.4. Transformation d'un vecteur du repère Ri à Rj, après transformation d'un point[6].

2.3.3. Transformation d'un plan

-La position relative d'un point par rapport à un plan est indépendante des transformations appliquées à l'ensemble {point, plan}, ainsi :

$$jQ.jV = iQ.iV = iQ.iTj.jV \qquad (2.7)$$

il s'en suit que :

$$jQ = iQ.iTj \qquad (2.8)$$

2.3.4. Matrice de transformation de translation pure

-la matrice de transformation se présente comme suite :

$$iTj = \text{Trans}\,(a; b; c) = \begin{bmatrix} 1 & 0 & 0 & a \\ 0 & 1 & 0 & b \\ 0 & 0 & 1 & c \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (2.9)$$

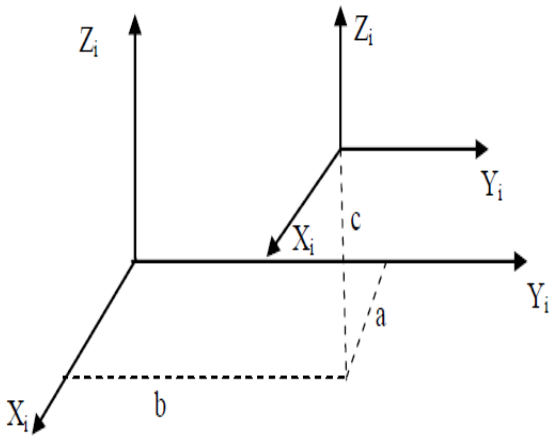


Figure. 2.5. Matrice de translation pure [6].

La matrice $Trans(a, b, c)$ peut être décomposée en produit de trois matrice tel que :

$$Trans(a, b, c) = Trans(X, a).Trans(Y, b).Trans(Z, c) \qquad (2.10)$$

2.3.5. Matrices de transformation de rotation autour des axes principaux

2.3.5.1. Rotation θ autour de l'axe X

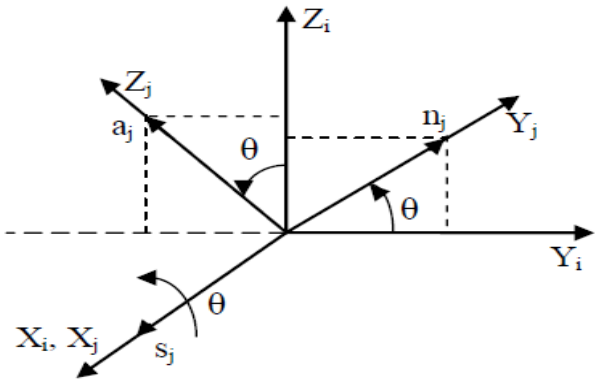


Figure. 2.6. Transformation de rotation pure autour de l'axe X [6].

$\boldsymbol{Rot(X, \theta)}$ **:** est une transformation correspondent à une rotation pure d'angle θ autour d'axe x.

La matrice de transformation s'écrite :

$$iTj = Rot(X,\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} & A(X,\theta) & & 0 \\ & & & 0 \\ & & & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.11)$$

Où $A(X,\theta)$ est la matrice d'orientation de dimension (3x3).

2.3.5.2. Rotation θ autour de l'axe Y

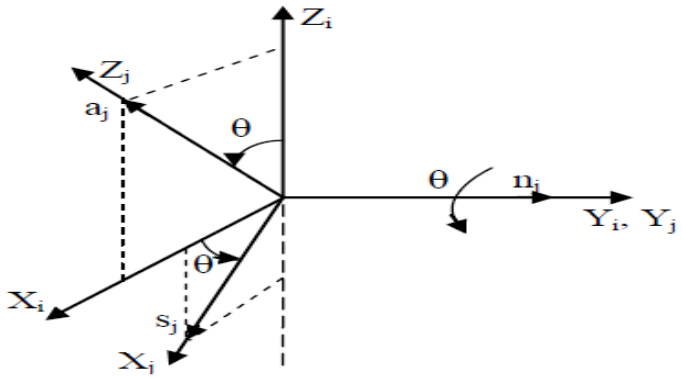


Figure 2.7. Transformation de rotation pure autour de l'axe Y[6].

La matrice de rotation autour de y se présente comme suite :

$$iTj = \begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} & A(Y,\theta) & & 0 \\ & & & 0 \\ & & & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.12)$$

2.3.5.3. Rotation θ autour de l'axe Z

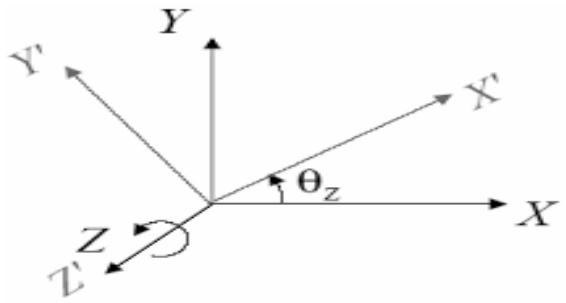


Figure 2.8: Matrice de rotation autour de z [6].

-La matrice de rotation autour de Z se présente comme suite :

$$iTj = Rot(Z,\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} & A(Z,\theta) & & 0 \\ & & & 0 \\ & & & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.13)$$

-On déduire que la rotation effectue par rapport à l'origine se fait comme suite:

$$R = Rot(Z,\theta z).Rot(Y,\theta y).Rot(X,\theta x)$$

$$R = \begin{bmatrix} cos\theta & -sin\theta & 0 & 0 \\ sin\theta & cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} cos\theta & 0 & -sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -sin\theta & 0 & cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\theta & -sin\theta & 0 \\ 0 & sin\theta & cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.14)$$

2.3.6. Propriétés des matrices de transformation homogènes

a) Une matrice de transformation peut se mettre sous la forme suivante :

$$T = \begin{bmatrix} sx & nx & ax & 0 \\ sy & ny & ay & 0 \\ sz & nz & az & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} & A & & p \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2.15)$$

La matrice A représente la rotation alors que la matrice colonne p représente la translation. Pour une transformation de translation pure $A = I3$ (I3 matrice unité d'ordre 3), tandis que pour une transformation de rotation pure $P = 0$

b) La matrice A est orthogonale, c'est à dire que la matrice inverse est égale à la matrice transposée. [6]

$$A^{-1} = A^{t} \quad (2.16)$$

2.4. Modèles géométrique des robots manipulateurs

Les modèles géométriques direct et inverse qui expriment la situation de l'organe terminal en fonction des variables articulaires du mécanisme et inversement [7].

2.4.1. Description de la géométrie des robots à structure ouverte simple

Une structure ouverte simple est composée de $(n+1)$ corps notés $C0$,…, Cn et de n articulations. Le corps $C0$ désigne la base du robot et le corps Cn le corps qui porte l'organe terminal.

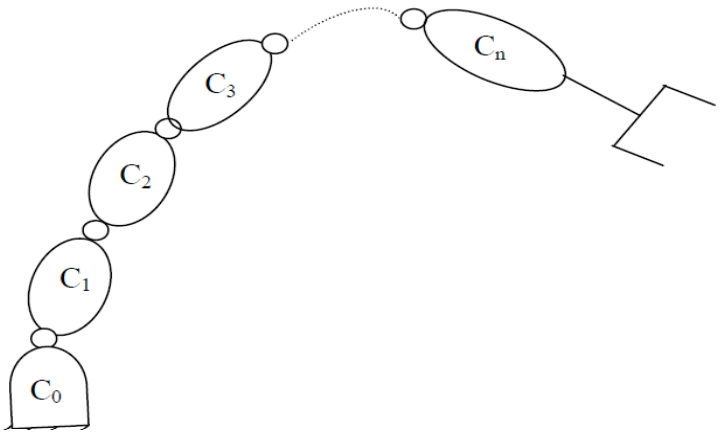


Figure 2.9. Structure ouverte simple du bras manipulateur.

2.4.2. Modèle géométrique direct

Afin de calculer l'équation géométrique directe d'un bras manipulateur a chaine ouverte simple, une méthode systématique et générale doit être appliquée pour définir la position et l'orientation relative de deux corps consécutifs, Les méthodes de Denavit-Hartenberg et de Khalil et Kleinfinger sont les plus rependus. Se reportant a la figure, la structure est composée de n corps Cj; $j = 1$ j a n, plus la base $(C0)$, relies entre eux par des liaisons rotoïde ou prismatique, l'articulation j relie le corps Cj au corps Cj $-$ 1 . Afin de définir la relation entre les positions des corps on associe à chaque corps Cj le repère $Rj(Oj; Xj; Yj; Zj)$ tels que :

- L'axe Zj est porte par l'axe de l'articulation j
- L'axe xj est porte par la perpendiculaire commune aux axes zj et zj + 1
- L'axe yj est forme par la règle droite pour compléter les coordonnées du système (xj, yj, zj)
- La matrice de transformation du repère Rj-1 au repère Ri s'exprime en fonction des paramètres
- αj : L'angle de rotation entre l'axe zj-1 et l'axe zj autour de xj-1
- dj : La distance entre l'axe zj-1 et l'axe zj le long de l'axe xj-1
- θj : L'angle de rotation entre les axes xj-1 et xi autour de zj
- rj : la distance entre les axes xi-1 et xi le long de l'axezj.

-La variable qj de l'articulation j, définissant l'orientation ou la position relative entre les articulations j-1 et i, est soi θj , soit rj , selon le type d'articulation est rotoïde ou prismatique respectivement. Ceci est définie par la relation [7] :

$$qj = \bar{\sigma}\theta j + \sigma jrj \qquad (2.17)$$

Avec :

$\sigma j = 0$ si l'articulation j est rotoïde.

- $\sigma j = 1$ si l'articulation j est prismatique.
- $\sigma j = 1 - \sigma j$

De la même façon on définit la variable qi :

$$\bar{q}j = \sigma\theta j + \sigma j r j \qquad (2.18)$$

La matrice de transformation définissant Ri dans Rj-1 est donnée par :

$$i - 1Ti = Rot(X, \alpha j)Trans(X, dj)Rot(Z, \theta j)Trans(Z, rj) \qquad (2.19)$$

$$j - 1Tj = \begin{bmatrix} \cos\theta j & -\sin\theta j & 0 & dj \\ \cos\alpha j \sin\theta j & \cos\alpha j \cos\theta j & -\sin\alpha j & -rj \sin\alpha j \\ \sin\alpha j \sin\theta j & \sin\alpha j \cos\theta j & \cos\alpha j & -rj \sin\theta j \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad (2.20)$$

- La représentation de la matrice de transformation $j - 1Tj$ est comme le montre la figure 2.10

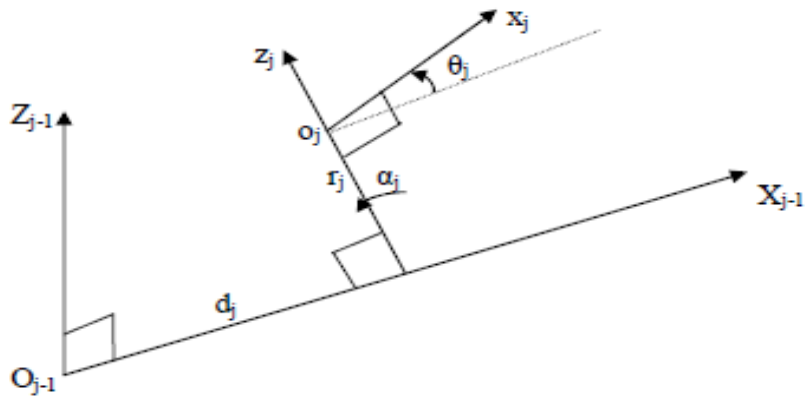


Figure.2.10. Paramètres géométrique dans le cas d'une structure ouverte simple [7].

2.4.3. Modèle géométrique inverse

Le modèle géométrique direct fournit l'emplacement de l'organe terminal en fonction des coordonnées articulaires. Le modèle géométrique inverse (MGI) consiste a déterminer les variables articulaires correspondant à une situation spécifique de l'organe terminal lorsqu'elles existent.

Trois méthodes de calcul de MGI sont répandues :

- La méthode de Paul, qui convient pour la plupart des robots industriels.

- La méthode de Peiper, qui permet de résoudre le problème pour les robots a six degrés de liberté avec trois articulations rotoïde ou trois articulation prismatiques.
- La méthode de raghavan et Roth, donnant la solution générale des robots a six articulations a partir d'un polynôme de degré au plus égale 16 [7] :

2.5. Conclusion

Dans ce chapitre, nous avons présenté une méthode de description des robots permettant de modéliser, avec des notations homogènes, des chaines ouvertes.

Nous avons aussi présenté la modélisation géométrique directe qui offre un modèle unique, donné sous forme d'équations explicites, ainsi que nous avons exposé certaines méthodes pour le calcul du modèle géométrique inverse. Dans le prochain chapitre nous allons décrire les composants de notre bras manipulateur et les logiciels utiliser.

Chapitre 3
Outils électroniques du bras manipulateur

3.1. Introduction

Le bras manipulateur est un robot qui réalise des tâches difficiles , répétitives et spécifiques que l'homme ne peut accomplie tout seule , et pour faire toutes ces fonctions il doit recevoir des informations suffisantes pour qu'il fasse bien son travail et dans des conditions favorables, et de la viens la commande du robot , cette dernière est faite par l'homme a travers différentes machines et programmes , chaque robot a spécifiquement une commande et un programme propre a lui , et tout cela ne peut se faire sans des matériaux bien déterminer comme un microcontrôleur , des servomoteurs , et la combinaison de la commande et des outils est appelé Hardware et Software .

3.2. Description de la carte Arduino Uno

Arduino est un microcontrôleur développé en tant qu'système open source. Il est alimenté par une puce et composé de différents composants soudés sur la carte. Il ressemble à une mini carte mère utilisée dans un éventail de projets. Arduino est également programmable en suivant des fonctions requises dans un projet. Les programmes seront utilisés pour affecter certaines broches pour exécuter des tâches spécifiques. Les pièces et les broches sont identifiées à l'aide des étiquettes imprimées sur la carte [8].

La carte Arduino Uno est basée sur un microcontrôleur ATMega328 cadencé à une fréquence de 16 MHz. C'est la plus récente et la plus économique carte à microcontrôleur d'Arduino.

Des connecteurs d'entrées et de sorties situés sur les bords extérieurs du circuit imprimé permettent d'afficher une série de modules complémentaires [9].

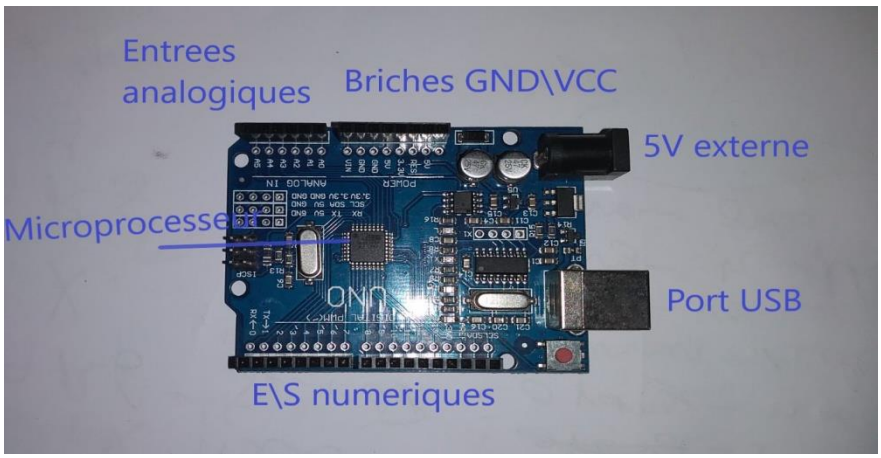


Figure.3.1. La carte Arduino Uno R3.

3.3. Caractéristiques de la carte arduino

- Les caractéristiques techniques de la carte arduino sont montrées dans le tableau suivant :

Tableau. 3.1. Caractéristiques de la carte arduino.

Caractéristique	**Description**
Microcontrôleur	ATmega328
Tension de fonctionnement	5V
Tension d'alimentation (recommandé)	7-12V
Tension d'alimentation (limites)	6-20V
Nombre de broches E/S numériques	14
Nombre de broches E/S numériques PMW	6
Nombre de broches d'entrée analogiques	6
Intensité maximale disponible par broches E/S (5v)	20mA
Intensité maximale disponible pour la sortie 3.3v	50mA
Mémoire programme flash	32 Ko (ATmega328P) dont 0,5 Ko utilisé par le chargeur de démarrage
Mémoire SRAM (mémoire volatile)	2 KB (ATmega328P)
Mémoire EEPROM (mémoire non volatile)	1 KB (ATmega328P)
Vitesse de l'horloge	16 MHz
Longueur	68.6 mm
Largeur	53.4 mm
Poids	25g

3.4. Description du module Bluetooth HC-05

Le module Bluetooth HC-05 est un module Bluetooth SPP facile à utiliser, conçu pour la configuration d'une connexion série sans fil transparente. Ce qui facilite l'interface avec le contrôleur ou le PC. Le module Bluetooth HC-05 fournit un mode de commutation entre les modes maître et esclave, ce qui signifie qu'il peut être utilisé comme pour réception et la transmission de données [10].

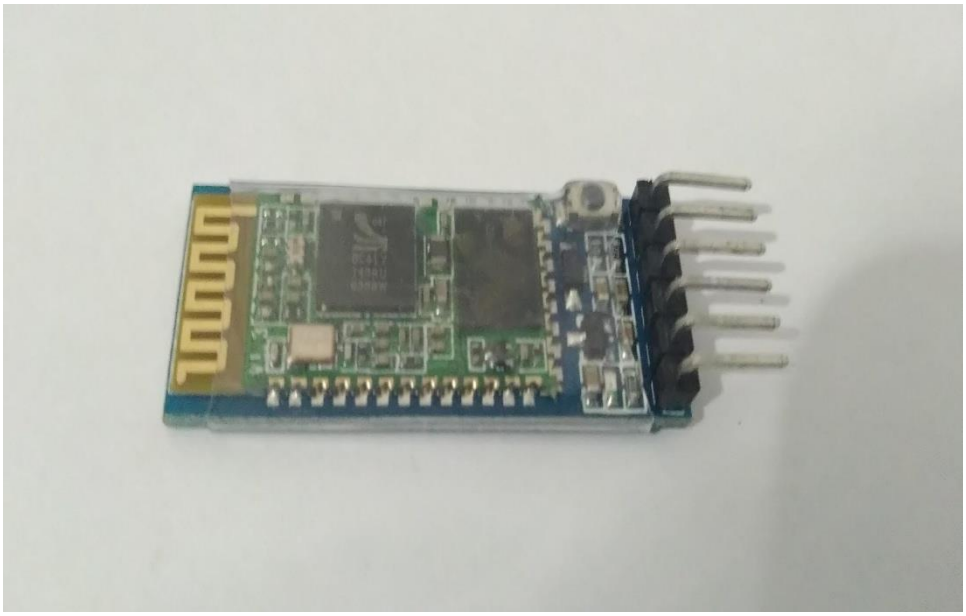


Figure 3.2 : Le module Bluetooth HC-05

3.5. Caractéristiques du module Bluetooth HC-05

- Les caractéristiques du module HC-05 sont montrées dans le tableau suivant

Tableau 3.2. Spécification du module HC-05.

Propriété	**Fonction**
Modèle	HC-05
Tension d'entrée	DC 5V
Méthode de communication	communication série
VCC	connecté à +5V
GND	connecté à la terre
TXD (broche d'envoi série Bluetooth)	connecter au microcontrôleur
RXD (broche de réception serial Bluetooth)	connecter au microcontrôleur
KEY	Si le niveau d'entrée est bas ou s'il est connecté à l'air, le module est en mode couplé ou en communication.

3.6. Description du servo moteur

Un servomoteur est comme son nom l'indique est un moteur mais avec quelques spécifications en plus, Contrairement à un moteur classique qui est utilisé pour tourner avec une vitesse proportionnelle à un courant ou à une tension, pour obtenir une position. Il effectue une rotation suivant un angle déterminé. Le plus souvent l'angle est compris entre 0 et 180° puis il garde cette position.

Les servomoteurs sont des actionneurs. Très utilisés dans l'industrie, ils ont comme caractéristique principale leur « couple », c'est-à-dire la force de rotation qu'ils peuvent exercer. Plus un servomoteur aura de couple et plus il pourra actionner des « membres » lourds comme déplacer un bras qui porte une charge. [1]

3.6.1. Description servomoteur SG90

Souvent mentionné comme "mini servo" ce moteur est en fait un micro servomoteur largement utilisé dans le monde du modélisme.[11]

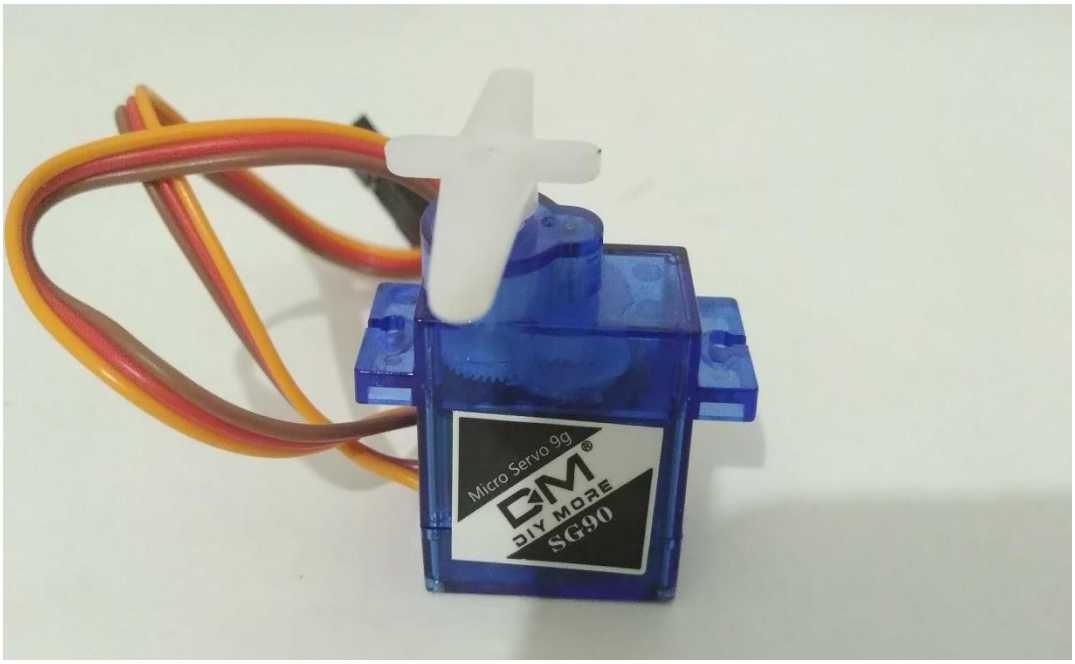


Figure 3.3. : Le servomoteur SG90.

3.6.2. Caractéristiques du servomoteur SG90

- Le servomoteur SG90 a les propriétés suivantes :

Tableau 3.3. Caractéristiques du servomoteur SG90.

Propriété	**Description**
Tension	4.8-6V
Amplitude	0 à 180° (pour être précis, de 5° à 175° selon nos constatations)
Couple	1.2kg/cm sous 4.8v
Vitesse	0.12 sec/60° sous 4.8v
Prise	Standard Graupner à 3 fils
Dimension	22mm x 11.5mm x 27mm
Poids	9g

3.6.3. Description servomoteur MG996R

MG996R est une version améliorée du servo MG995.Le nouveau système de contrôle PCB et IC qui le rend beaucoup plus précis dans son fonctionnement .Son engrenage interne et son moteur sont également mis à niveau pour améliorer la bande passante morte et le centrage [12].

Figure 3.4. Le servomoteur MG996R.

3.6.4. Caractéristiques du servomoteur MG996R

Tableau 3.4. Caractéristiques du servomoteur MG996R.

Propriété	**Description**
Tension	4.8-6V
Amplitude	180°
Couples	11Kg/cm sous 4.8V - 13Kg/cm sous 6.0V
Vitesse	0.16 sec/60° sous 4.8V - 0.13
Consommation	120mA sans charge 1 450 mA pour 11Kg/cm environ
Prise	servomoteur standard Orange Rouge Marron
Temps de réponse de largeur d'impulsion	5 microsecondes ou moins
Erreur de déviation angulaire	retour à 0 degrés, 45 ° autour de chaque 3 ° ou moins d'erreur
Dimension	40 x 20 x 36,5 mm

3.6.4. Alimentation

Figure.3.5. Alimentation 9V.

3.6.5. Fils de connexion

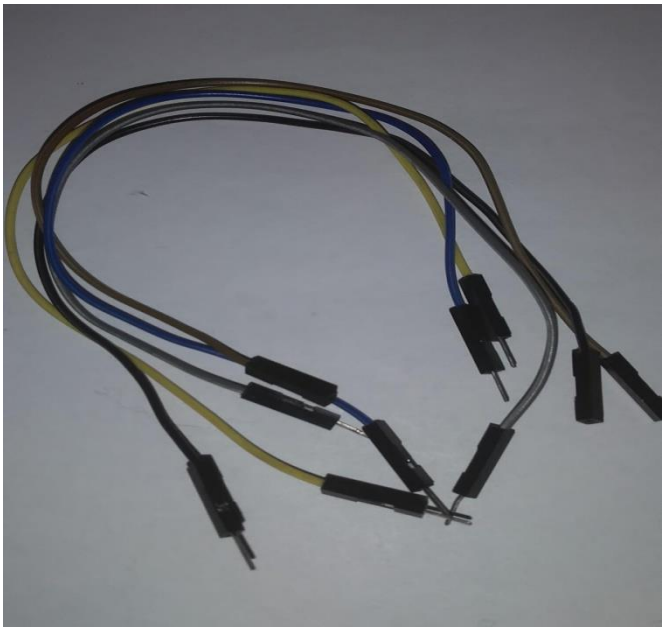


Figure 3.6 : Les fils de connexion pour le bras manipulateur.

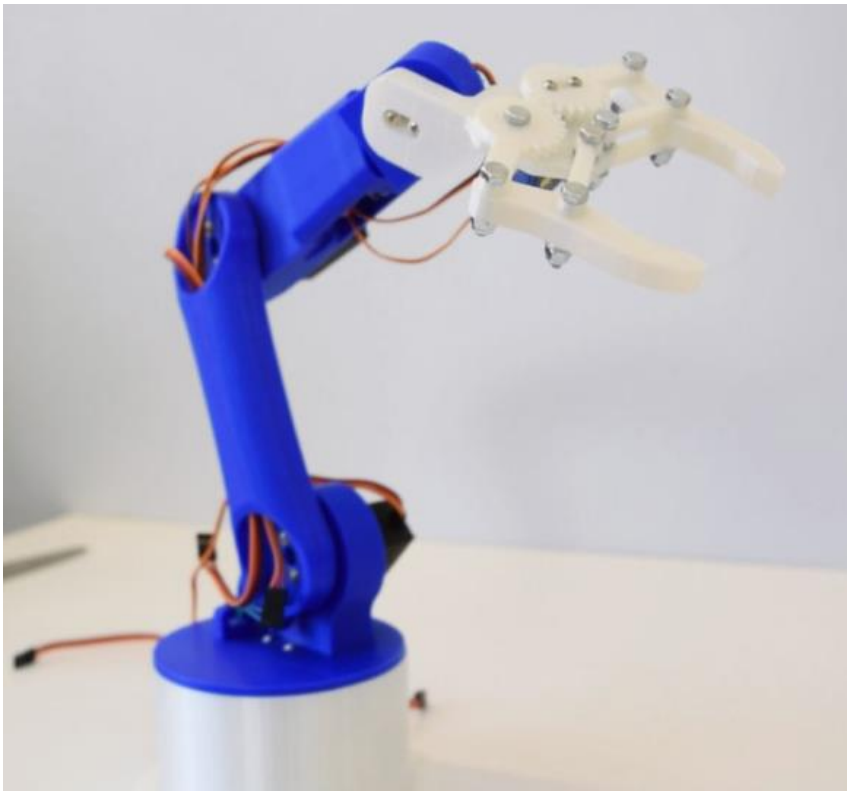


Figure 3.7 : Bras manipulateur avec les composants associer.

3.7. Définition du logiciel Arduino IDE

Arduino IDE est un logiciel open source qui est principalement utilisé pour écrire et compiler le code dans le module Arduino. Il s'agit d'un logiciel officiel Arduino, ce qui rend la compilation du code très facile pour que même une personne ordinaire sans connaissances techniques préalables puisse se frotter avec le processus d'apprentissage. Il est facilement disponible pour les systèmes d'exploitation comme MAC, Windows, Linux et s'exécute sur la plate-forme Java qui comprend des fonctions et des commandes intégrées qui jouent un rôle essentiel pour le débogage, l'édition et la compilation du code dans l'environnement.

Une gamme de modules Arduino disponibles, notamment Arduino Uno, Arduino Méga, Arduino Leonardo, Arduino Micro et bien d'autres.

Chacun d'eux contient un microcontrôleur sur la carte qui est réellement programmé et accepte les informations sous forme de code. Le code principal, également appelé croquis, créé sur la plate-forme IDE générera finalement un fichier hexadécimal (un fichier hexadécimal) qui sera ensuite transféré et téléchargé dans le contrôleur de la carte. L'environnement IDE contient principalement deux parties de base: l'éditeur et le compilateur .L'éditeur est utilisé pour écrire le code requis, le compilateur est utilisé pour compiler et télécharger le code dans le module Arduino donné. Cet environnement prend en charge les langages C et C++ [13].

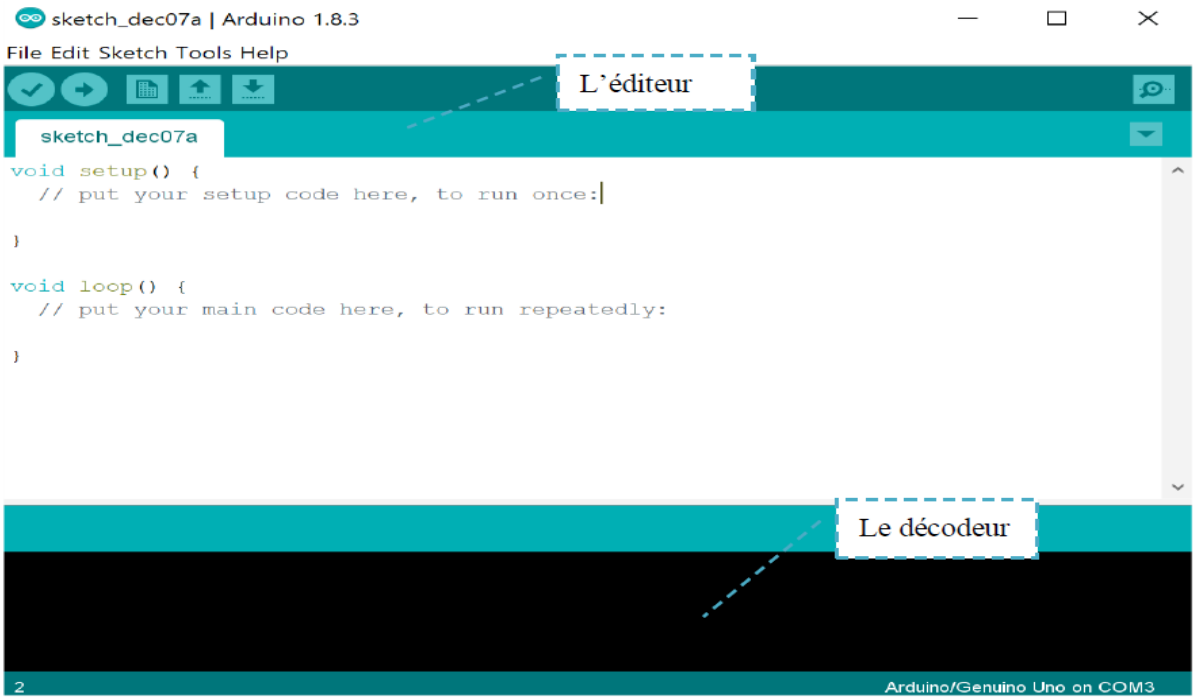


Figure. 3.8 : Le logiciel de programmation Arduino IDE.

3.8. Définition du logiciel App inventor

App Inventor est une plateforme digitale qui permet de développer des applications pour les téléphones Androïde à l'aide d'un navigateur Web et d'un téléphone ou d'un émulateur connecté. Les serveurs App Inventor stockent le travail et aide à suivre les projets.

L'App Inventor Designer, permet de sélectionner les composants pour les applications.

L'éditeur de blocs App Inventor, dans lequel vous assemblez des blocs de programme qui spécifient le comportement des composants. L'assemblage des programmes visuellement, en assemblant des pièces comme des pièces d'un puzzle.

L'application apparaît sur le téléphone étape par étape au fur et à mesure qu'on y ajoute des éléments, afin qu'on puisse tester votre travail .au fur et à mesure qu'on construit. Lorsque on terminé, on pout empaqueter l'application et produire une application autonome à installer.

Si on n'a pas de téléphone Androïde, on peut créer des applications à l'aide de l'émulateur, un logiciel qui s'exécute sur l'ordinateur et se comporte exactement comme le téléphone.

L'environnement de développement App Inventor est pris en charge pour les systèmes d'exploitation Mac OS, GNU / Linux et Windows et plusieurs modèles de téléphones Android populaires. Les applications créées avec App Inventor peuvent être installées sur n'importe quel téléphone [14].

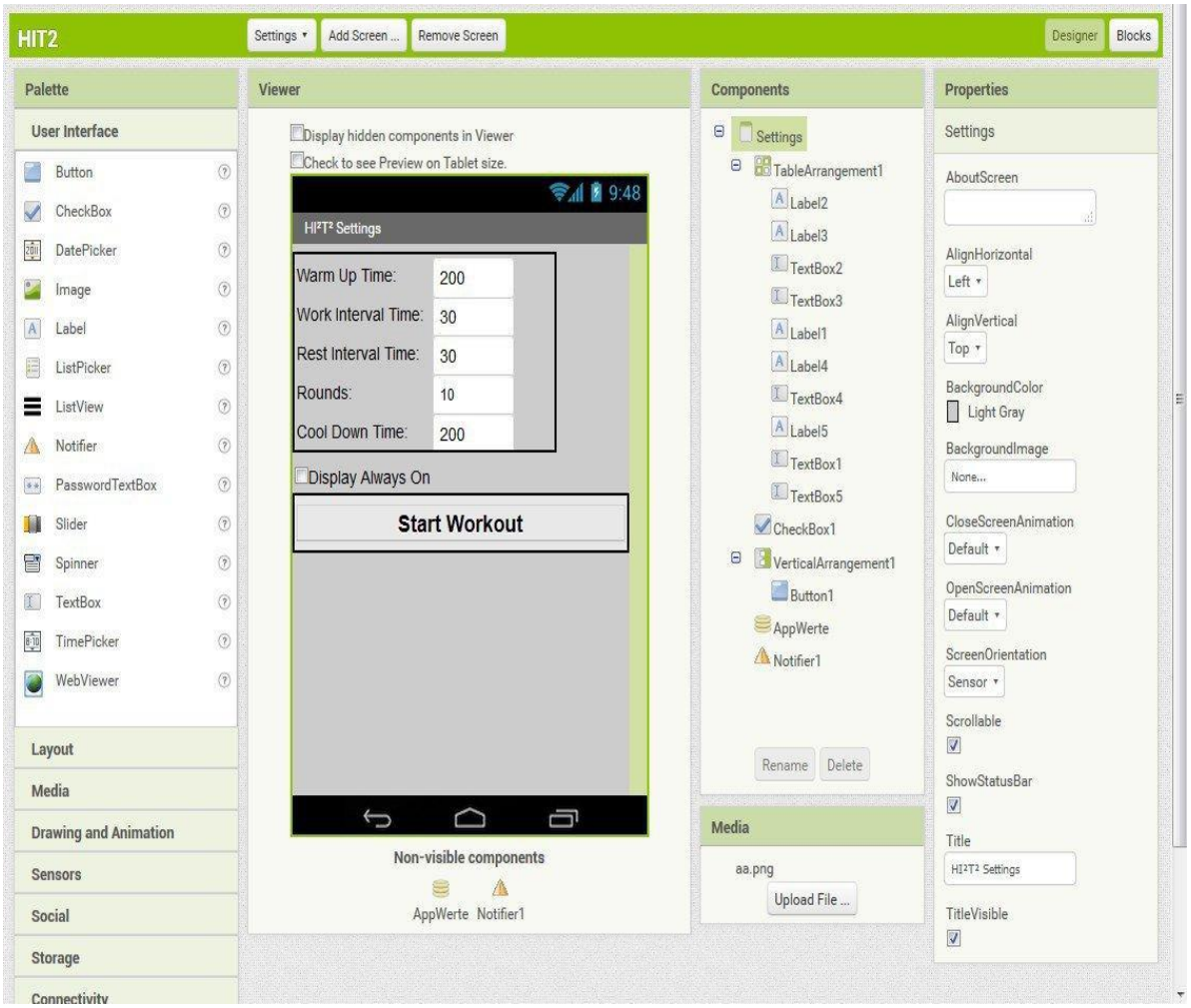


Figure.3.9. : Logiciel de création d'application Androïde App inventor.

3.9. Définition du logiciel Proteus

Proteus est un outil de simulation et de conception développé par Labcenter Electronics pour la conception de circuits électriques et électroniques. Il s'agit d'une suite logicielle contenant des schémas, des simulations ainsi que la conception de circuits électriques imprimés.

ISIS est le logiciel utilisé pour dessiner des schémas et simuler les circuits en temps réel. La simulation permet une intervention pendant l'exécution, fournissant ainsi une simulation en temps réel. ARES est utilisé pour la conception de circuits imprimés .Il a la fonction de visualiser la sortie en vue 3D du circuit imprimé conçu avec les composants.

3.9.1. Caractéristiques du logiciel :

- **ISIS** possède une large gamme de composants dans sa bibliothèque. Il dispose de sources, de générateurs de signaux, d'outils de mesure et d'analyse tels que l'

oscilloscope , le voltmètre, l'ampèremètre, etc., des sondes pour la surveillance en temps réel des paramètres du circuit, des commutateurs , des afficheurs , des charges comme des moteurs et des lampes, des composants discrets comme des résistances, des condensateurs, des inductances, transformateurs, circuits intégrés numériques et analogiques, commutateurs semi-conducteurs, relais, microcontrôleurs, processeurs, capteurs.

- **ARES** propose la conception de PCB jusqu'à 14 couches internes, avec montage en surface et packages de trous traversant. Il est intégré aux empreintes de différentes catégories de composants tels que les circuits intégrés, les transistors, les en-têtes, les connecteurs et d'autres composants discrets. Il offre des options de routage automatique et de routage manuel au PCB Designer. Le schéma dessiné dans l'ISIS peut être directement transféré sous forme de schéma ARES [15].

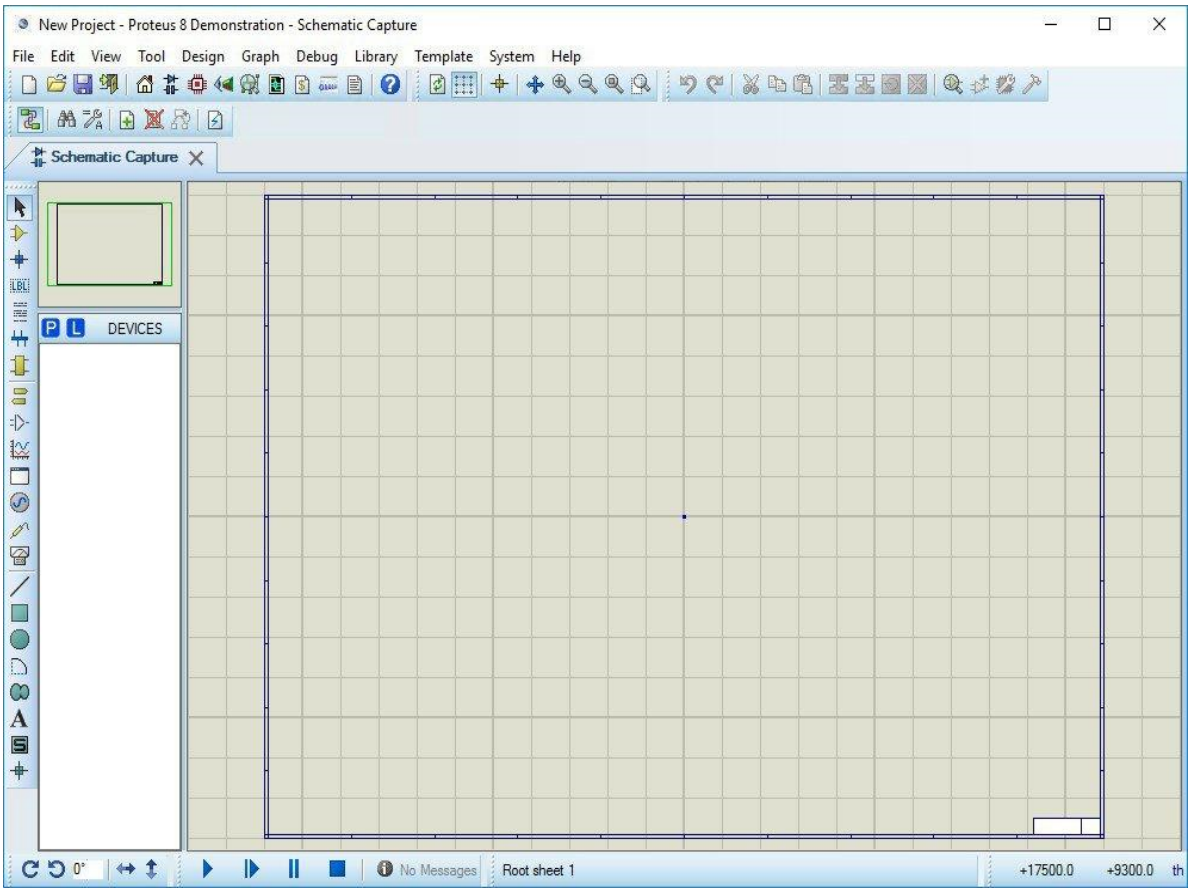


Figure 3.10 : Le logiciel de création de schéma électronique Proteus.

3.10. Définition de logiciel 3D Builder

3D Builder est un logiciel propriétaire de Microsoft permettant de faire de la CAO 3D . il permet de visualiser des fichiers STL et modifier ou crée des objets en trois dimension [16].

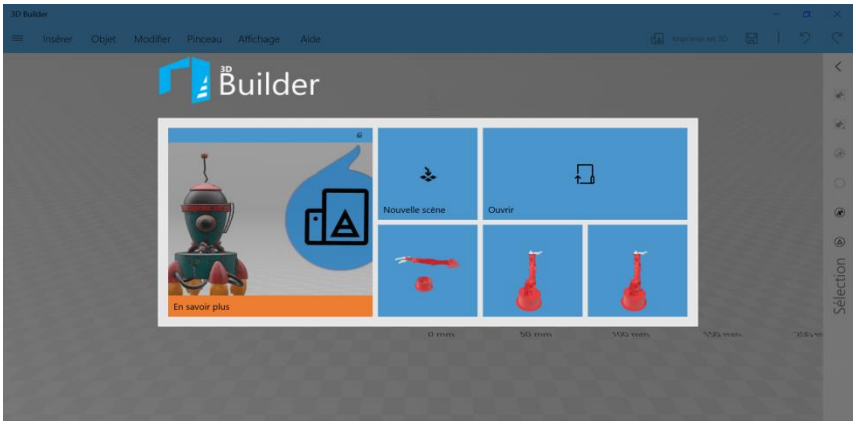


Figure 3.11: logiciel 3D Builder.

3.11. Conclusion

Dans ce chapitre, on s'est intéressées a la description matériel des différents composants essentiel au fonctionnement du robot, tel que : la carte Arduino, le module Bluetooth HC-05, les servomoteurs SG90 et MG996R, par la suite on a présenté les différents logiciels qu'on va l'utiliser qui sont : Arduino IDE, App inventor , 3D Builder et Proteus, dans le prochain chapitre on vas s'intéresser a la réalisation de bras manipulateur et son fonctionnement.

Chapitre 4

Realisation et teste du bras manipulateur

4.1. Introduction

Dans la conception des bras manipulateur la partie électronique joue un rôle très important dans le fonctionnement du système en effet pour permettre au robot de se déplacer et de communiquer avec l'extérieur.

Dans ce chapitre nous nous consacrons à décrire et à visualiser les différentes partitions matérielles de notre robot, ensuite nous allons simuler les composants du bras avec la carte Arduino Uno Sous le logiciel Proteus et enfin nous passerons à la création de l'application androïde pour faire un test du bras manipulateur.

4.2. Partie hardware

La réalisation de notre bras manipulateur s'est faite à partir de plusieurs pièces en plastique. La première articulation est assurée par un servomoteur de type MG966R qui se trouve au niveau de la base suivie par 2 servomoteur de même type définissants deux articulation 1 et 2 respectivement , ces derniers sont relier en sérés avec deux autre articulation 3 et 4 lies a de servomoteur SG90 , ainsi que de l'organe terminal comme le montre la figure ci-dessous :

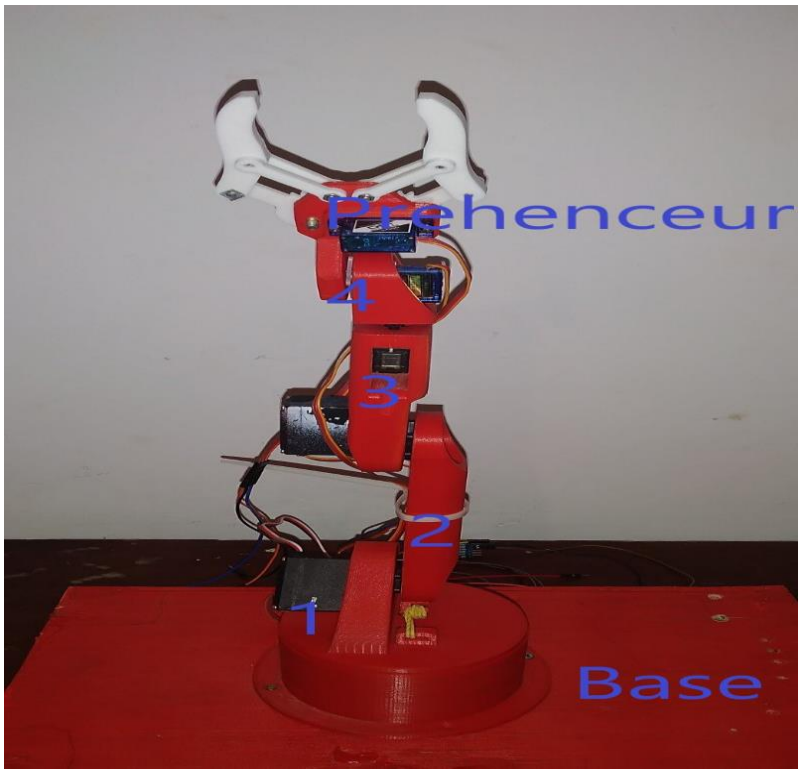


Figure 4.1 : Présentations des pièces du bras manipulateurs assemblées.

4.2.1 La conception 3D du bras manipulateur

Pour construire le bras manipulateur Nous avons fait appel au club scientifique ELECTRO il ont importer les fichiers STL du site web nommée howtoMecatronics sur un logiciel de design et impression 3D des constituants du bras sauf la pince qu'on nous même ramener puis ils ont imprimé les pièces en 3D de matière plastique PLA, par la suite ils ont assemblé Concernant le préhenseur nous l'avons imprimé dans une imprimerie 3D les huit pièces qui le constitues avec les fichiers STL correspondant, puis nous avons assemblé ces pièces avec des vices et des écrous de quatre millimètres aussi et nous l'avons ajouté au reste du corps du robot.

Afin de visualiser toute les composantes du bras manipulateur nous avons utilisé un logiciel de visualisation appeler 3DBuilder, nous avons créé une nouvelle scène puis importer les fichiers un par un jusqu'à la construction de tout le prototype et on l'a illustré dans les figures qui vont suivre.

4.2.3. Visualisation 3D des pièces du bras manipulateur

4.2.3.1. Visualisation de la base

La base de notre bras manipulateur est montrée dans la figure suivante :

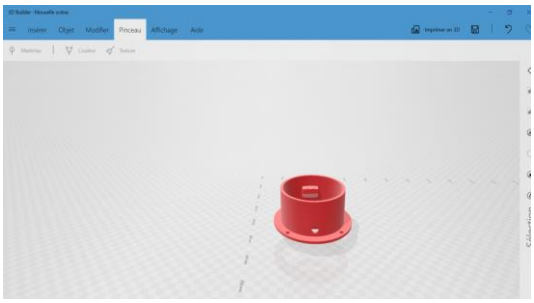


Figure 4.2: La base en 3D.

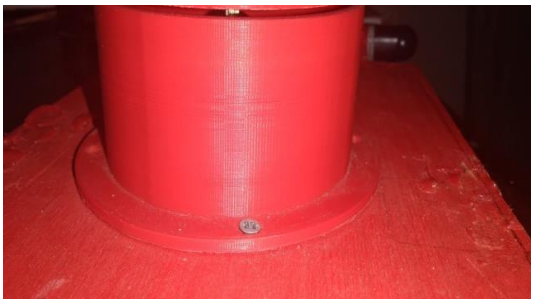


Figure 4.3: La base réel .

4.2.3.2. Visualisation des articulations

- Ces quatre figures représente les quatre articulations qui viens après la base visualiser en trois dimensions dans le logiciel 3DBuilder avant impression .

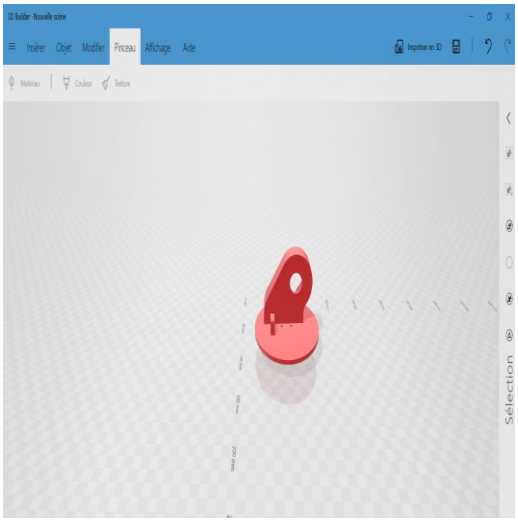


Figure 4.4:L'articulation 1 en 3D.

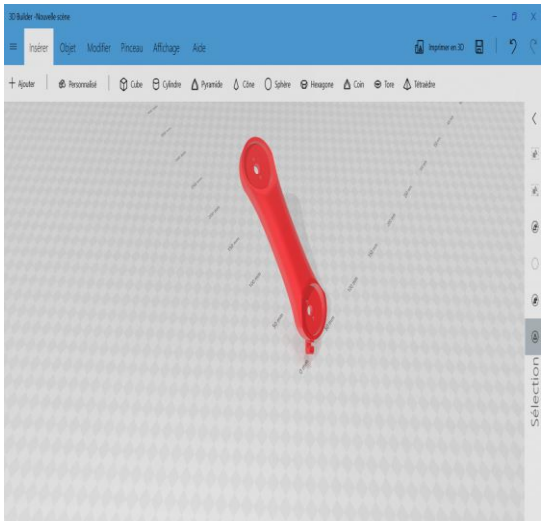


Figure 4.5:L'articulation 2 en 3D .

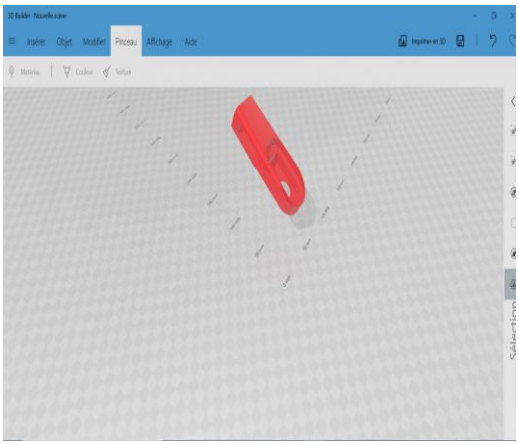


Figure 4.6:L'articulation 3 en 3D.

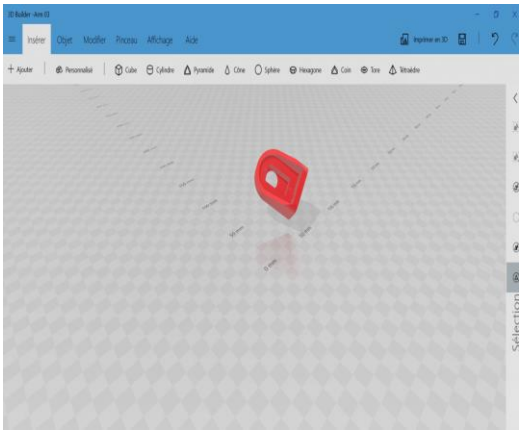


Figure 4.7: L'articulation 4 en 3D.

- Les figures suivantes montre les articulations réel du bras manipulateur.

Figure 4.8: L'articulation 1 réel.

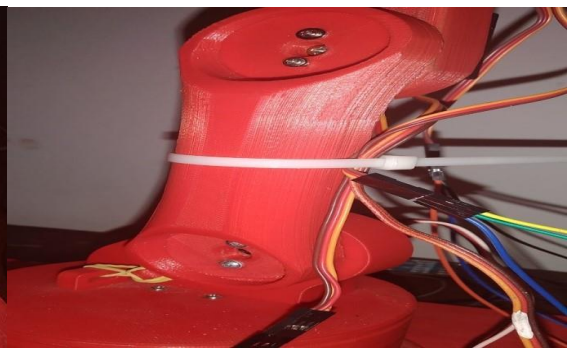


Figure 4.9: L'articulation 2 réel .

Figure 4.10: L'articulation 3 réel .

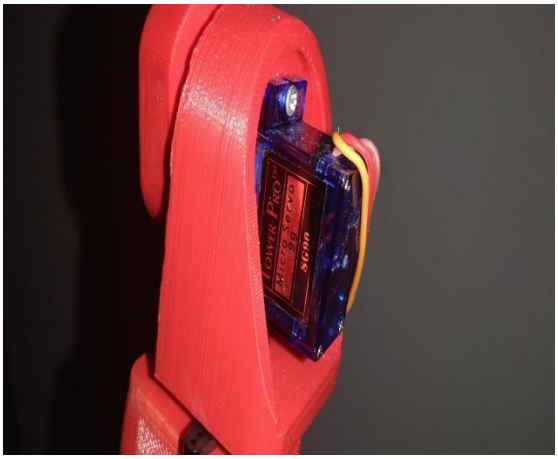


Figure 4.11: L'articulation 4 réel.

4.2.3.3. Visualisation du préhenseur

- La représentation en trois dimension et réel du bras manipulateur est comme suit :

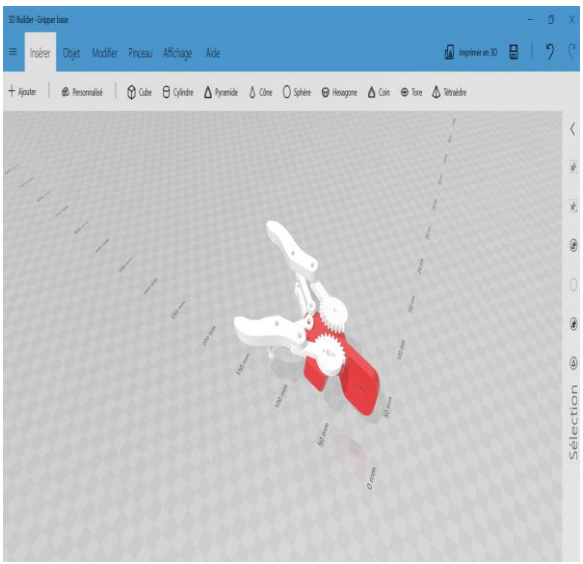


Figure 4.12 : Le préhenseur en 3D.

Figure 4.13 : Le préhenseur réel.

4.2.4.Description de l'assemblage de bras manipulateur

- Dans cette partie on a assemblé les déférents pièces a partir de la base jusqu'a l'organe terminal , et relier les servo moteurs avec le circuit électronique qui est la carte Arduino Uno et le module Bluetooth représenter comme suit :

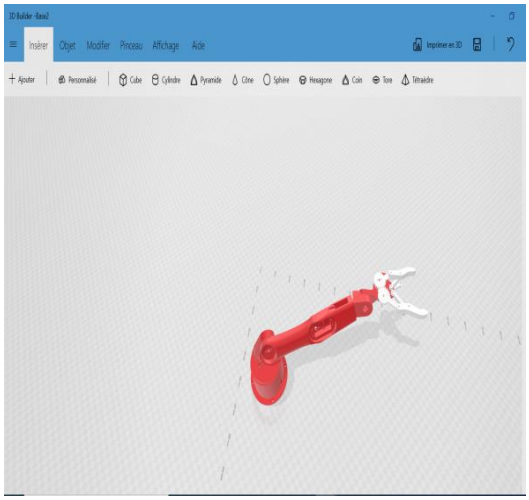


Figure 4.14: l'assemblage de bras bras manipulateur en 3D.

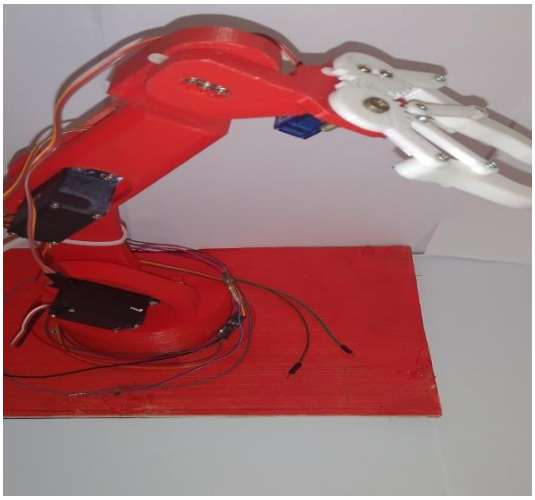


Figure 4.15 : Image réel l'assemblage de manipulateur .

4.3. Partie Software

Cette parie vas contenir la simulation du montage du bras manipulateur, les explications de la création de l'application Androide et de la programmation du robot .

4.3.1. Simulation des composants du bras avec la carte Arduino Sous Proteus

Nous avons d'abord crée un nouveau projet dans Proteus, puis nous avons téléchargé la bibliothèque de la carte Arduino Uno et du module Bluetooth HC-05, on a ouvert un nouveau projet dans le logiciel de simulation Proteus. Ensuite nous avons ajouté la carte Arduino Uno, le module Bluetooth et les servo moteur depuis la bibliothèque des composants du logiciel. Nous avons simulé chaque composant avec la carte, puis nous avons simulé le schéma global.

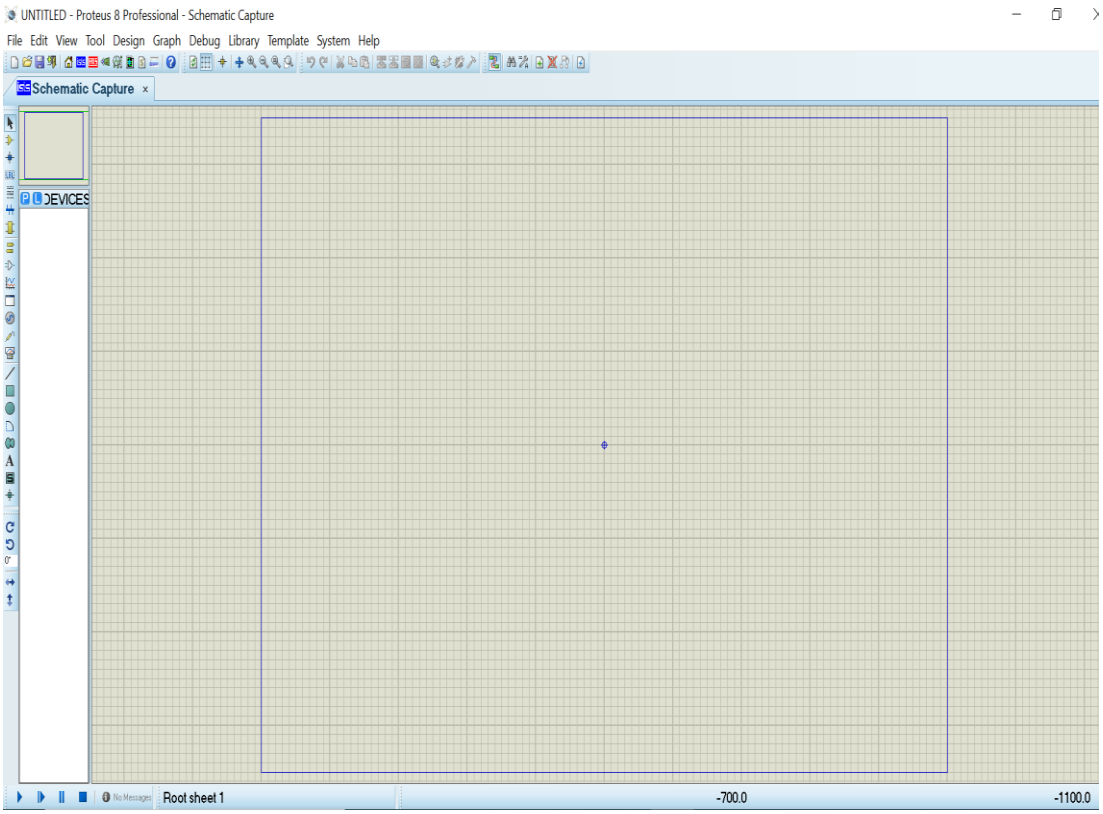


Figure 4.16 : Nouveau projet schématique sur le logiciel Proteus .

4.3.1.1. Câblage et simulation du Module Bluetooth HC-05 avec La carte Arduino Uno

On a placé la carte et le module HC-05 dans l'espace de travail du logiciel Proteus comme le montre la figure suivante :

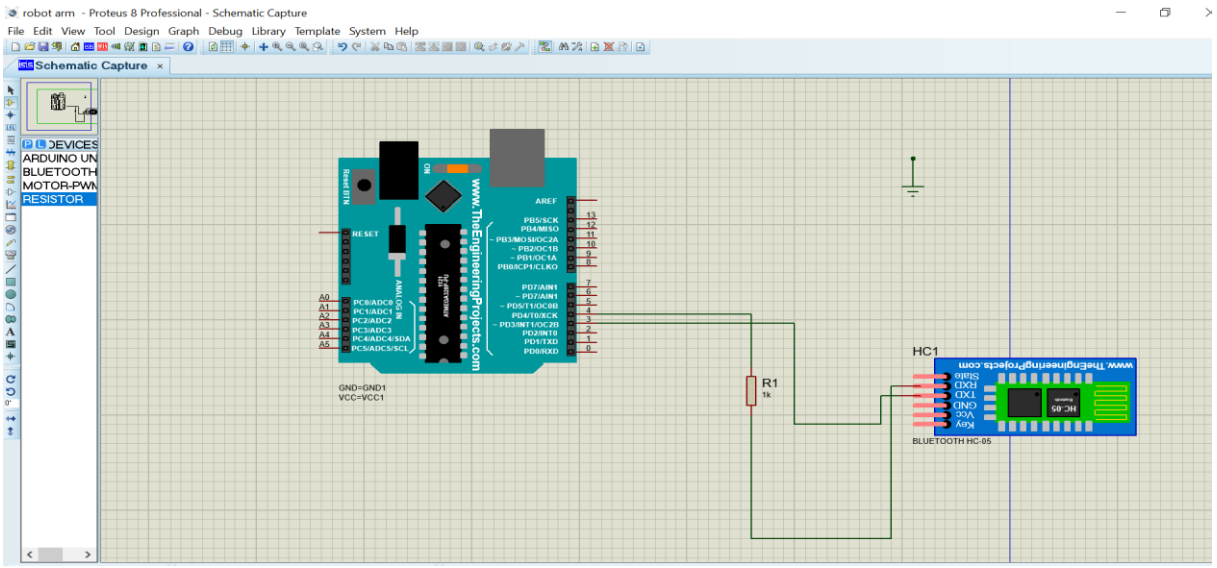


Figure 4.17 : Câblage de la carte Arduino Uno avec le module Bluetooth HC-05.

4.3.1.2. Câblage et simulation de la carte Arduino avec les servo moteurs

On a raccordé trois servo moteurs SG90 et trois MG996R avec la carte Arduino Uno afin de les visualiser comme suit :

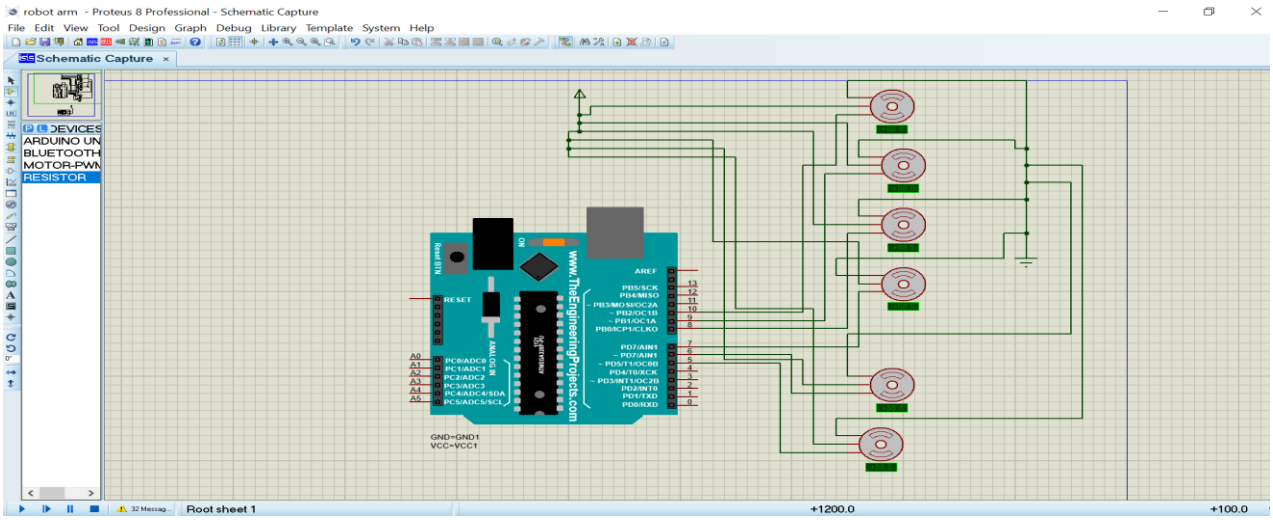


Figure 4.18 : Câblage de la carte Arduino Uno avec le module Bluetooth HC-05.

4.3.1.3. Câblage et simulation globales des composants du bras manipulateur

Apres avoir raccordé chaque composant avec la carte Arduino nous avons regrouper tous les composant précédant servo moteurs et module Bluetooth pour avoir le schéma globale et simuler le tout.

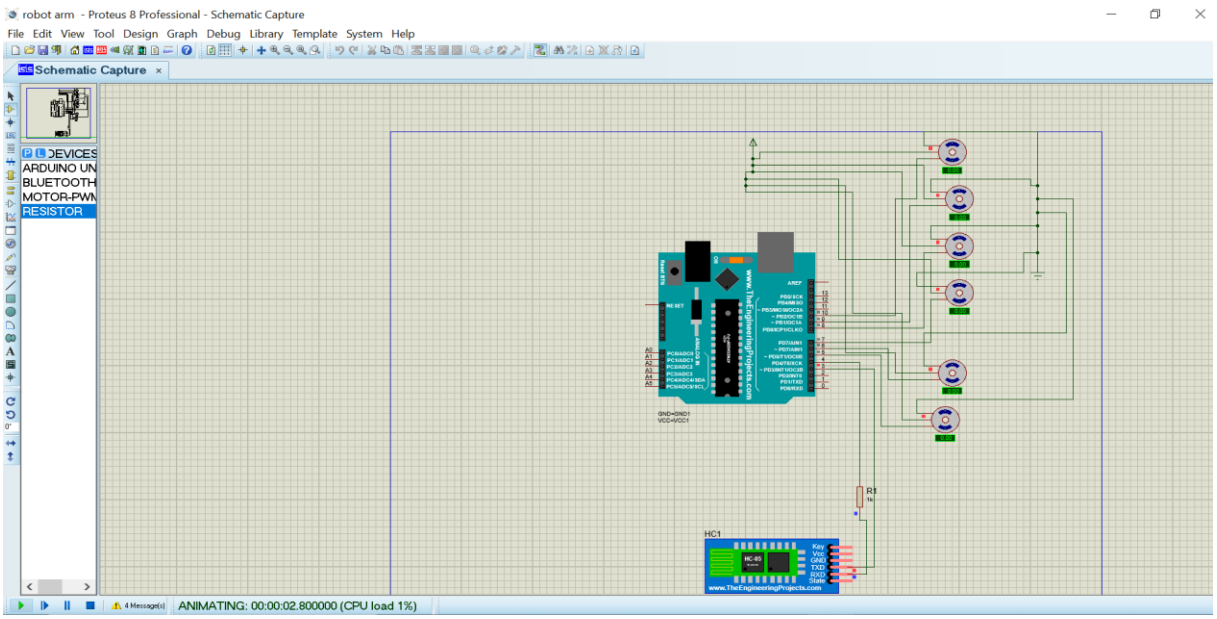


Figure 4.19 : Câblage globale des composants sous Proteus.

4.4. Les étapes de création de l'application Androïde

Nous avons suivit différentes étapes pour crée notre Application Bluetooth qui vont se présenter comme suit :

- Tout d'abord nous sommes allé au site app inventor, nous avons crée un compte avec une adresse mail après cela le site vas nous demander la permission d'utiliser cette adresse jusqu'à ce qu'on entre dans la fenêtre des projets.
- Nous avons crée un nouveau projet on cliquant sur l'icone a gauche« start new project » puis nous lui avons donné un nom « control bras robot arduino » sans espace, on a cliqué sur ok pour entrée dans la fenêtre principale de l'application.
- La fenêtre de notre application contient un menu et deux fenêtres une pour le design et une autre pour les blocs de programme, on a commencé par la fenêtre « designer »
  elle contient 4 blocs, le premier a gauche se nomme « Palette » qui contient les outils nécessaire pour l'application , le deuxième pour l'affichage de l'écran appeler « Viewer » , le troisième pour afficher la listes des composants utiliser dans l'application qui est « Components » et le dernier c'est le bloc des propriétés ou on modifie chaque outils utilisées .
- On a divises l'écran en plusieurs blocs pour pouvoir les éditer, on a pris de la liste « Layout » deux blocs horizontale surnommée « HorizontalArrangment » et un verticale du nom de « VerticalArrangement » , puis dans nous avons déplacer des zone textes dans les blocs précédant , le premier et deuxième blocs horizontale respectivement nous avons écrit le nom de l'application et déplacer les trois boutons de la liste des utilisateur qui est « UserInterface » pour les associer a la connexion et déconnexion du client Bluetooth.
- En ce qui concerne le bloc verticale nous avons modifier ces propriété pour qu'il soit de taille suffisantes pour contenir du coté gauche l'image du robot et de l'autres les curseur de mouvement des servomoteurs et la vitesse de ces dernier que nous avons déplacer de la listes des utilisateurs et modifier les propriétés afin qu'elles soit cohérente avec le programme.
- Apres les modifications que nous avons apporter a l'écran de l'application, nous avons passer au bloc des programme appeler « Blocs » qui contient deux fenêtres, une pour les blocs de programmation et l'autre est pour l'affichage et édition du programme.

- Les blocs de programme sons sous forme de puzzle en couleurs différentes, nous avons en premier plant nous suivis les étapes pour connecter le Smartphone au module Bluetooth du robot en déplaçant les blocs et en les associant a leur correspondant dans la fenêtre de design .
- En second plant nous avons fait les blocs de programme pour le déplacements des servo moteurs de la même manière que pour le module Bluetooth .
- En dernier plant nous avons éditer les blocs pour faire les modes de fonctionnement de l'application qui sont enregistre, marcher et redémarrer qui correspondes au programme Arduino . [17]

  - Les figures qui vont suivre représentes les modifications citer ci-dessus pour l'élaboration de l'application Androïde.

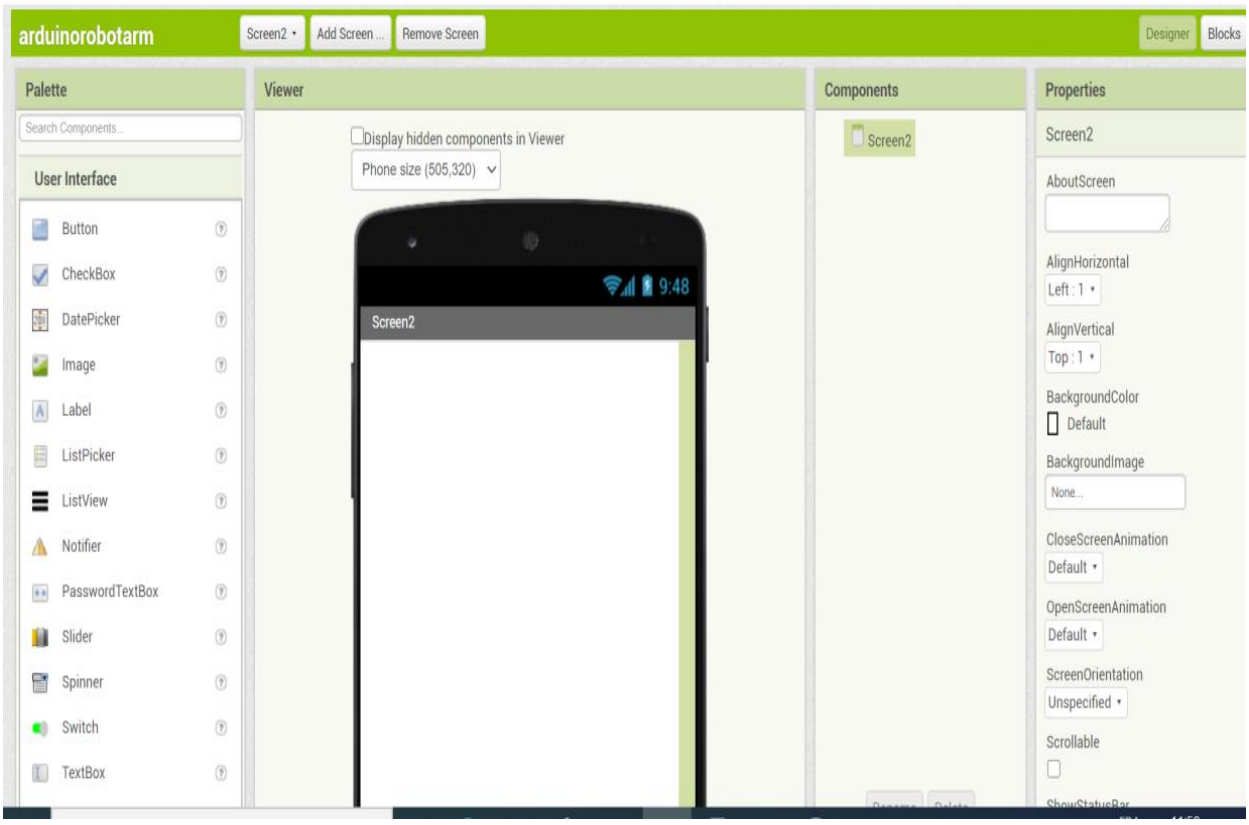


Figure 4.20 : Fenêtre initial de l'écran de l'application dans logiciel App inventor .

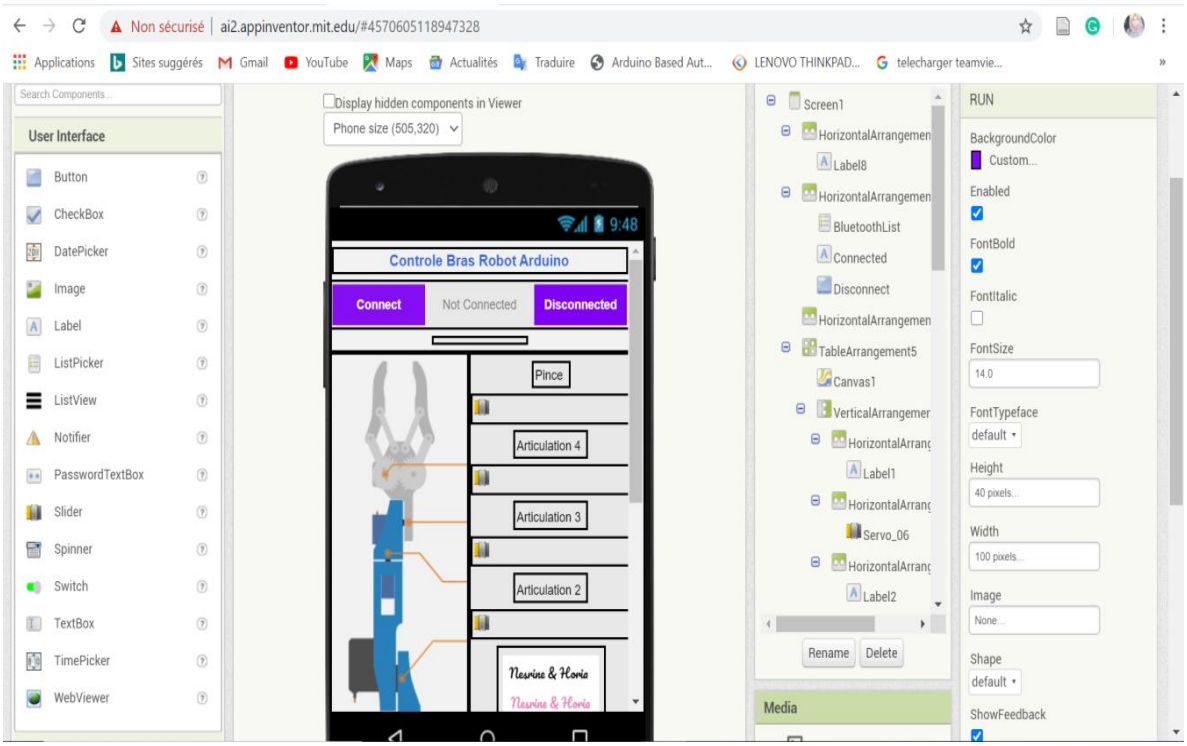


Figure 4.21 : Ecrans modifiée la parie de connexion Bluetooth et servomoteurs dans le logiciel .

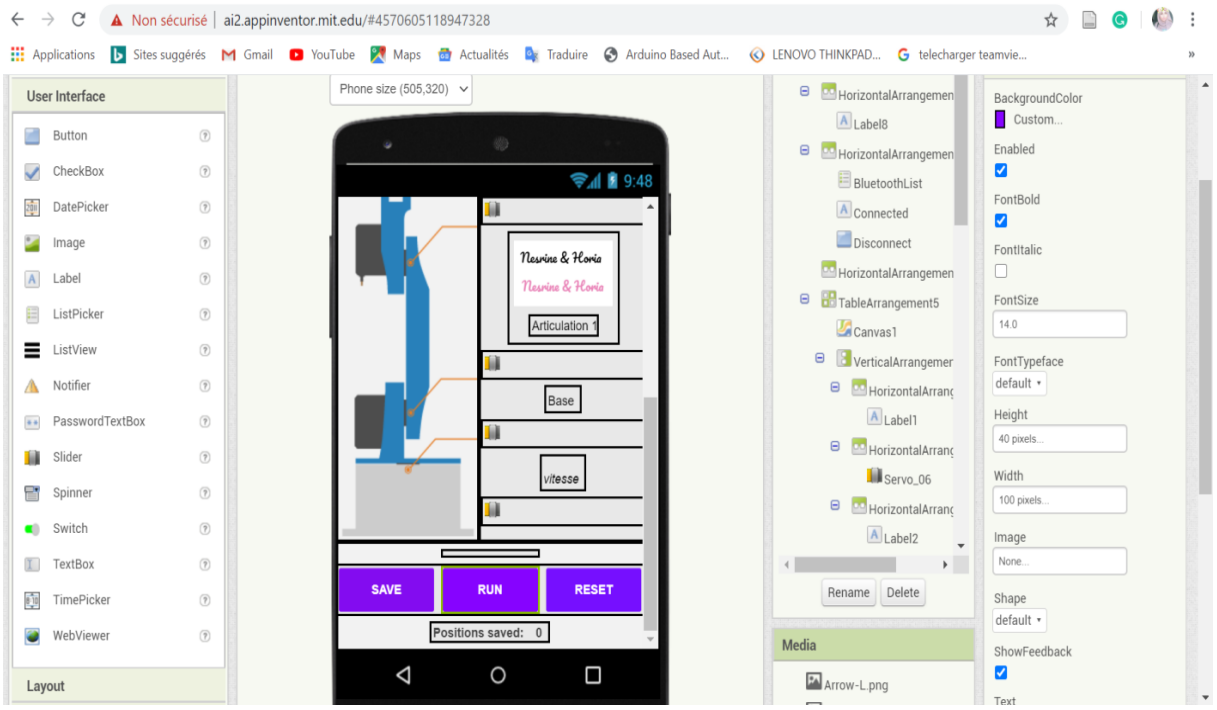


Figure 4.22 : Ecran modifiée de la partie des mode de fonctionnement de l'application .

- Nous avons assemblée les blocs logiques et de connexion Bluetooth pour faire connecter le module a l'application .

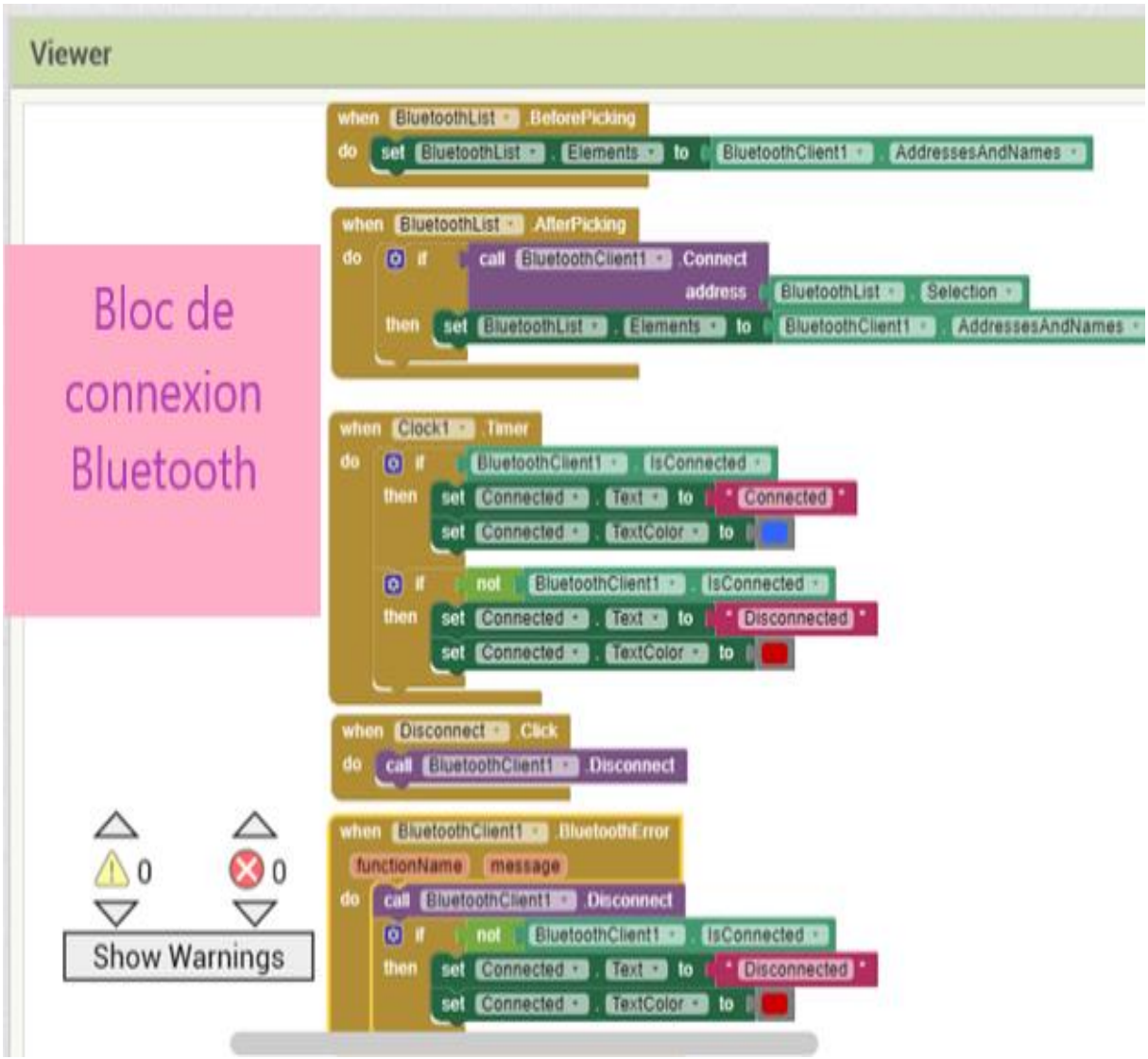


Figure 4.23 : Blocs de programme de connexion Bluetooth.

- Nous avons assembler les blocs logiques pour les mouvement des servomoteurs en accord avec le module HC-05 et la carte Arduino .

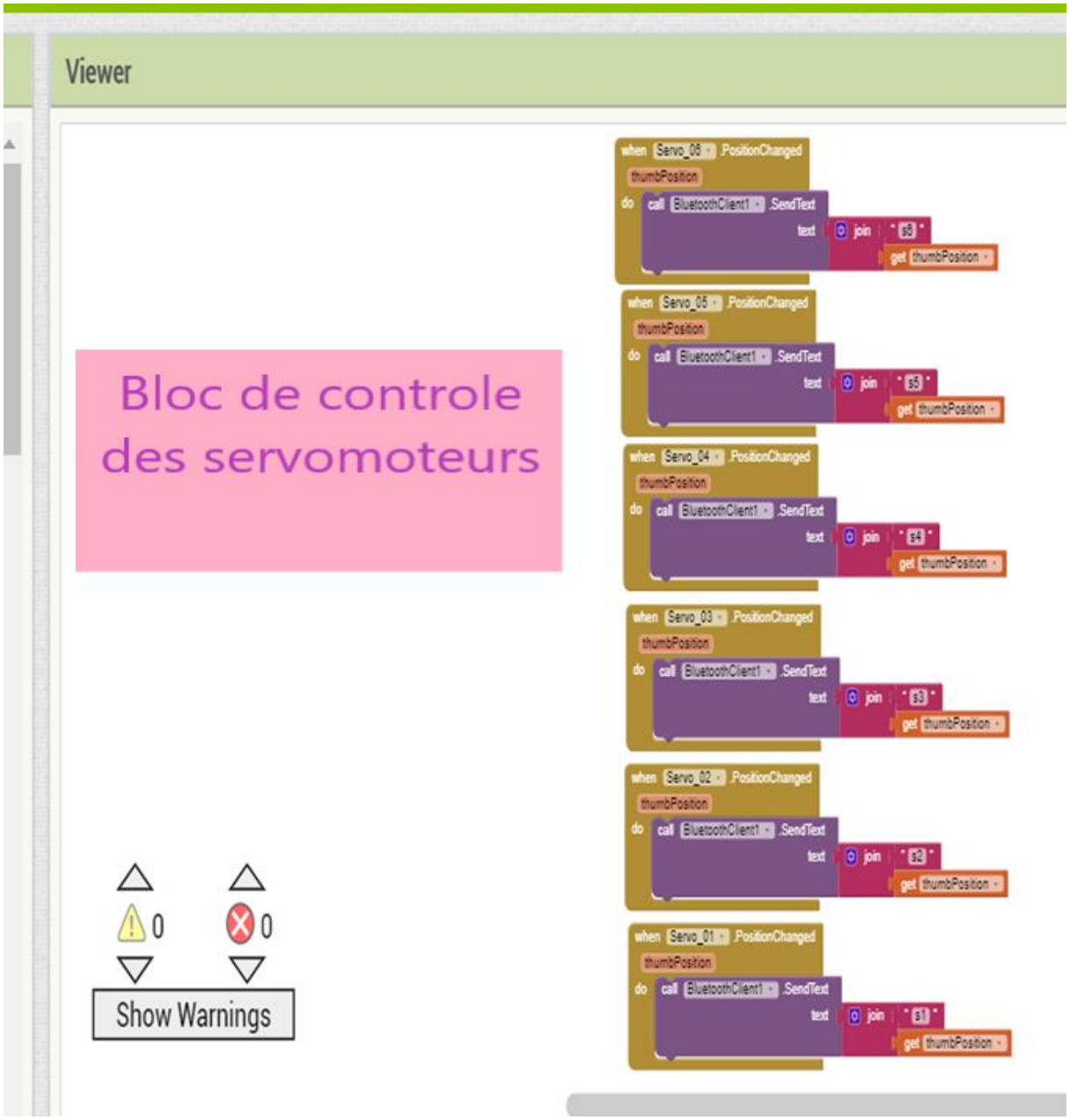


Figure 4.24 : Bloc de programme des mouvements des servomoteurs dans le logiciel.

- On a modifier le bloc des quatre modes de fonctionnement de l'application Androïde dans le logiciel afin qu'il marche parfaitement avec le programme Arduino en ajoutant les boutons « SAVE » pour la mémorisation , « RUN » pour l'exécution les positions enregistrer ,« RESET » pour la remise a zéro et « SPEED » pour régler la vitesse .

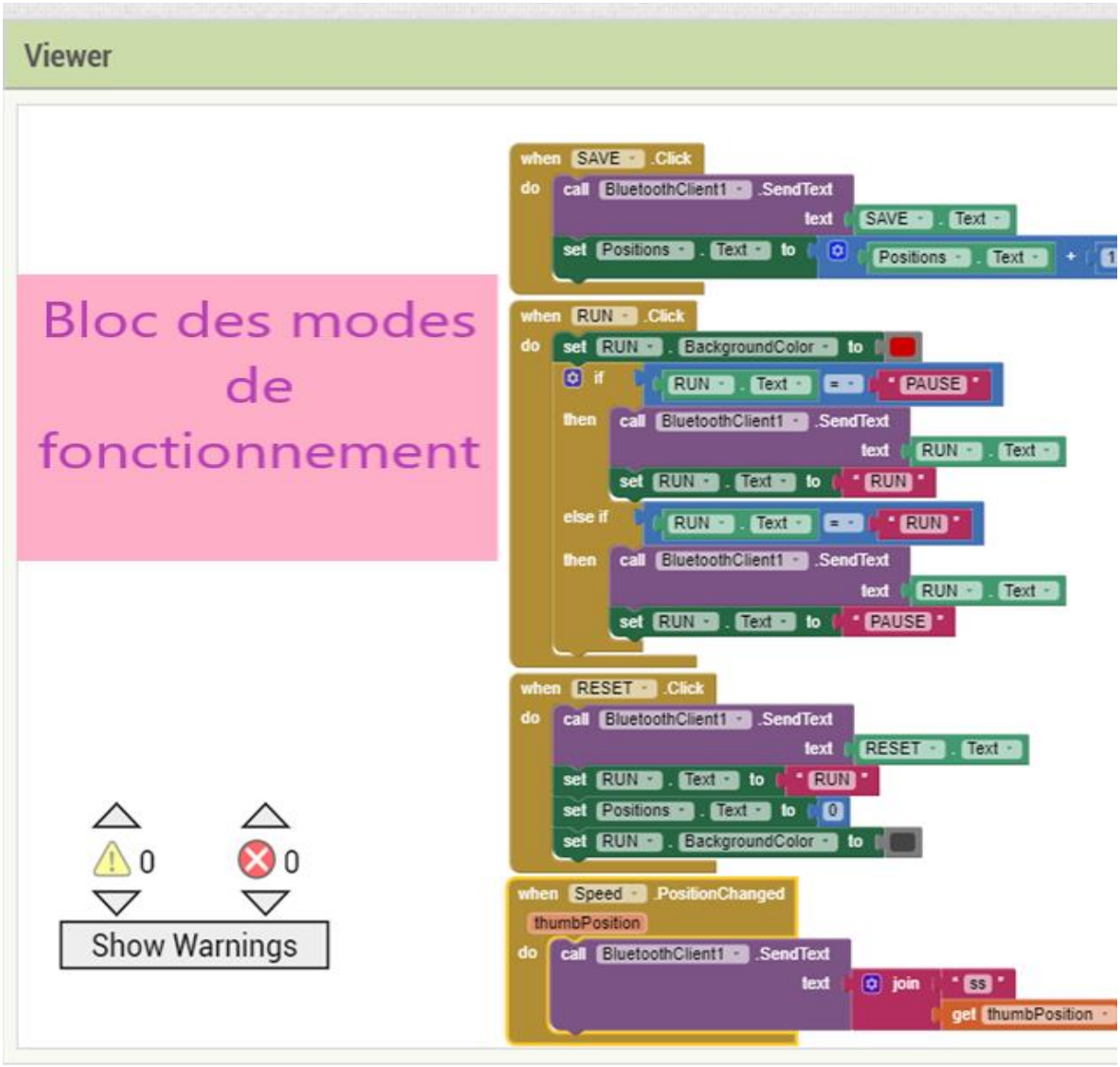


Figure 4.25: Bloc des modes de fonctionnement de l'application Androïde.

4.5. Le fonctionnement du bras manipulateur

4.5.1. La partie programme

Pour bien comprendre le programme de commande du bras manipulateur nous l'avons divisée en petites parties comme suit:

- Pour commencer ajouter dans le logiciel Arduino Ide la bibliothèque SoftwareSerial pour la communication série du module Bluetooth ainsi que la bibliothèque servo .
- Puis nous avons déclaré les variables des les six servomoteurs, le module Bluetooth, des variables pour stocker les positions actuelle et précédentes des servos, et des tableaux pour les positions et les modes automatique comme le montre la figure 4.26.

chapitre1_et_2_11-06-2020.docx | Arduino 1.8.10

Fichier Édition Croquis Outils Aide

chapitre1_et_2_11-06-2020.docx

```
#include <SoftwareSerial.h>
#include <Servo.h>

Servo servo01;
Servo servo02;
Servo servo03;
Servo servo04;
Servo servo05;
Servo servo06;

SoftwareSerial Bluetooth(3, 4); // Arduino(RX, TX) - HC-05 Bluetooth (TX, RX)

int servo1Pos, servo2Pos, servo3Pos, servo4Pos, servo5Pos, servo6Pos; // current position
int servo1PPos, servo2PPos, servo3PPos, servo4PPos, servo5PPos, servo6PPos; // previous position
int servo01SP[50], servo02SP[50], servo03SP[50], servo04SP[50], servo05SP[50], servo06SP[50]; // for storing positions/steps
int speedDelay = 20;
int index = 0;
String dataIn = "";
```

Figure 4.26 : Déclarations des différentes variables des composants dans Arduino IDE.

- nous avons fait l'initialisation de chaque servomoteur et de module Bluetooth et placer le bras du robot dans sa position initiale. Nous avons introduit la fonction write () qui déplace simplement le servo à n'importe quelle position de 0 à 180 degrés puis nous l'avons associée pour chaque servomoteur afin qu'il prend une nouvelle position a chaque mouvement comme le montre la figure 4.27.

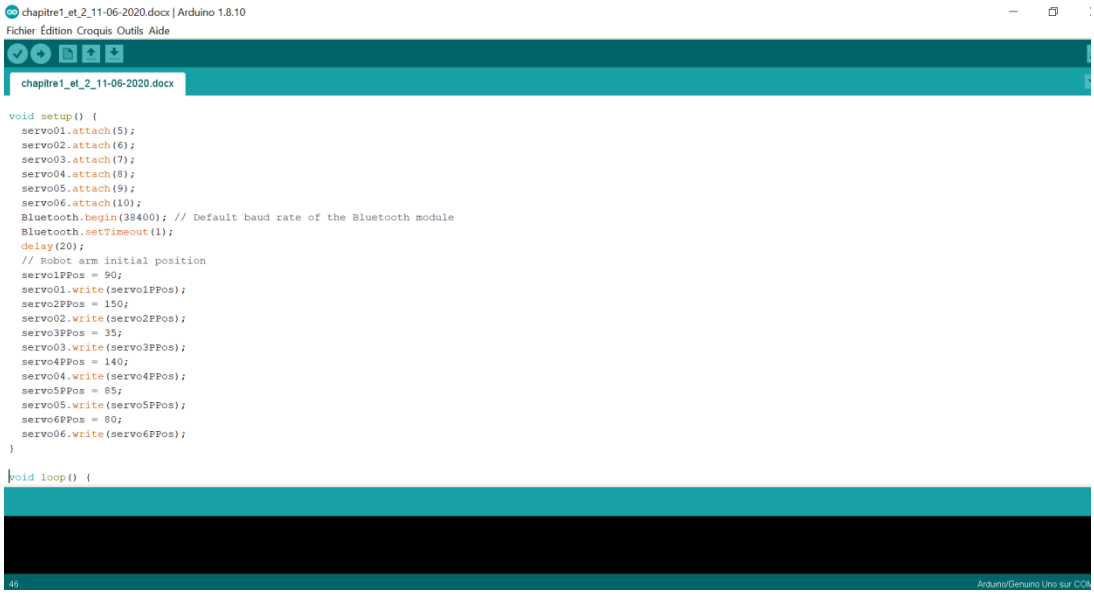


Figure 4.27 : initialisations des servomoteurs et module Bluetooth.

- Dans la boucle du programme appelée « void Loop » nous avons utilisé la fonction Bluetooth.available () pour vérifiée continuellement si il y'aurais des données

entrantes de notre l'application Androïde, puis après vérification et si il ya des nouvelles informations nous avons affecté la fonction readString () pour lire et stocker sous de chaine de caractères ces dernières dans le tableau dataIn.

```
void loop() {
  // Check for incoming data
  if (Bluetooth.available() > 0) {
    dataIn = Bluetooth.readString();  // Read the data as string
```

Figure 4.28 : lecture et stockage des donnes entrante de l'applications Androïde.

- On a utilisé la fonction startsWith () pour faire la vérifications du préfixe de chaque donnée entrante en conséquence nous saurons le quelle des six servo doit se déplacer En utilisant la fonction substring (), nous obtenons le texte restant, nous le convertissons en entier et utilisons la valeur pour déplacer les servos à cette position désignée par le préfix .

```
// If "Waist" slider has changed value - Move Servo 1 to position
if (dataIn.startsWith("s1")) {
  String dataInS = dataIn.substring(2, dataIn.length()); // Extract only the number. E.g. from "s1120" to "120"
  servo1Pos = dataInS.toInt();  // Convert the string into integer
```

Figure 4.29 : Vérification des préfix des données entrantes.

- On appeler la fonction pour pour que chaque servomoteur se déplace a la position choisit mais pour que ces derniers ne fonctionne pas a leur vitesse maximale nous avons contrôlé la vitesse des servomoteurs en ajoutant un retard pour chaque uns comme le montre la figure 4.30.

```
// We use for loops so we can control the speed of the servo
// If previous position is bigger then current position
if (servo1PPos > servo1Pos) {
  for ( int j = servo1PPos; j >= servo1Pos; j--) {   // Run servo down
    servo01.write(j);
    delay(20);     // defines the speed at which the servo rotates
  }
}
// If previous position is smaller then current position
if (servo1PPos < servo1Pos) {
  for ( int j = servo1PPos; j <= servo1Pos; j++) {   // Run servo up
    servo01.write(j);
    delay(20);
  }
}
servo1PPos = servo1Pos;   // set current position as previous position
```

Figure 4.30 : Contrôle de vitesse des servomoteurs.

- Pour la sauvegarde de chaque position des servomoteurs on a mis au point un bouton pour enregistrer continuellement.

```
// If button "SAVE" is pressed
if (dataIn.startsWith("SAVE")) {
  servo01SP[index] = servo1PPos;  // save position into the array
  servo02SP[index] = servo2PPos;
  servo03SP[index] = servo3PPos;
  servo04SP[index] = servo4PPos;
  servo05SP[index] = servo5PPos;
  servo06SP[index] = servo6PPos;
  index++;                        // Increase the array index
}
```

Figure 4.31 : Sauvegarde des nouvelles positions des servomoteurs.

- L'objectif du bouton « RUN », est pour faire appelle à la fonction personnalisée runservo () qui exécute les positions stockées en boucle jusqu'à ce que nous appuyions sur le bouton « RESET ». En utilisant la boucle FOR, pour parcourir toutes les positions stockées dans les tableaux et en même temps, nous vérifions des données entrantes . elles peuvent être le bouton « RUN » / « PAUSE », qui interrompt le robot . De plus, si nous modifions la position du curseur de vitesse, elle sera utilisée pour modifier le temps de retard entre chaque itération dans les boucles FOR ci-dessous, qui contrôle la vitesse des servomoteurs.

chapitre1_et_2_11-06-2020.docx

```
// Automatic mode custom function - run the saved steps
void runservo() {
  while (dataIn != "RESET") {   // Run the steps over and over again until "RESET" button is pressed
    for (int i = 0; i <= index - 2; i++) {  // Run through all steps(index)
      if (Bluetooth.available() > 0) {      // Check for incomding data
        dataIn = Bluetooth.readString();
        if ( dataIn == "PAUSE") {           // If button "PAUSE" is pressed
          while (dataIn != "RUN") {         // Wait until "RUN" is pressed again
            if (Bluetooth.available() > 0) {
              dataIn = Bluetooth.readString();
              if ( dataIn == "RESET") {
                break;
              }
            }
          }
        }
        // If speed slider is changed
        if (dataIn.startsWith("ss")) {
          String dataInS = dataIn.substring(2, dataIn.length());
          speedDelay = dataInS.toInt(); // Change servo speed (delay time)
        }
      }
      // Servo 1
      if (servo01SP[i] == servo01SP[i + 1]) {
      }
      if (servo01SP[i] > servo01SP[i + 1]) {
```

Figure 4.32 : Exécution des positions enregistrer en boucle .

- Nous avons insérer le bouton « RESET » il effacera toutes les données enregistrer dans le tableaux et les mettra a zéro , pour ensuite pouvoir enregistrer d'autres positions .[18]

```
// If button "RESET" is pressed
if ( dataIn == "RESET") {
  memset(servo01SP, 0, sizeof(servo01SP)); // Clear the array data to 0
  memset(servo02SP, 0, sizeof(servo02SP));
  memset(servo03SP, 0, sizeof(servo03SP));
  memset(servo04SP, 0, sizeof(servo04SP));
  memset(servo05SP, 0, sizeof(servo05SP));
  memset(servo06SP, 0, sizeof(servo06SP));
  index = 0;  // Index to 0
}
```

Figure 4.33 : Remise a zéro des données stocké dans le tableau.

4.5.2. Partie Test du bras manipulateur

On a tester notre robot en déplaçant un objet d'un point A vers un point B, premièrement on a connecté l'application Androïde depuis le smart phone avec le module Bluetooth HC-05 du montage , puis après la connexion nous avons régler la vitesse des servomoteurs pour ensuite déplacer les curseurs des six servomoteurs pour faire en sorte que le corps et la pince du robot se déplacent et soulèvent l'objet du point initiale au point finale , malheureusement on a rencontre des problèmes au niveau des servo des articulations leur mouvement n'étais pas prononcer ils ne faisais pas une rotation complète , on a essayer de les faire marche séparément et c'étais le même problème .

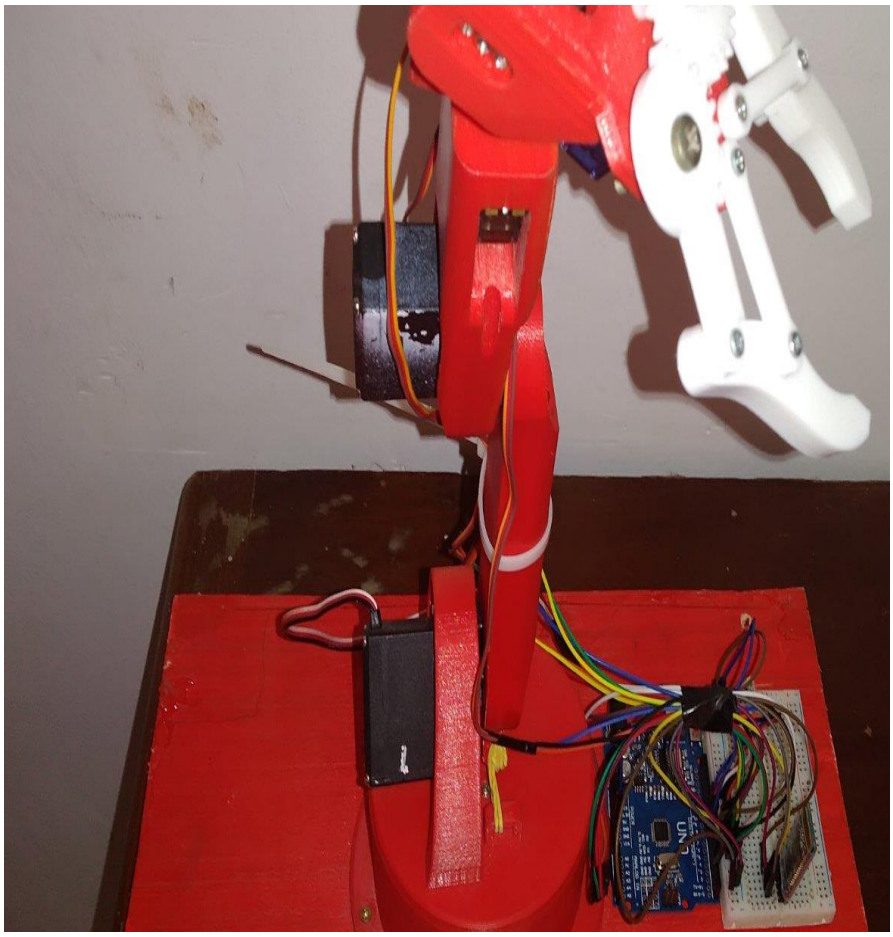


Figure 4.34 : Câblage du bras manipulateur avec les composant électronique .

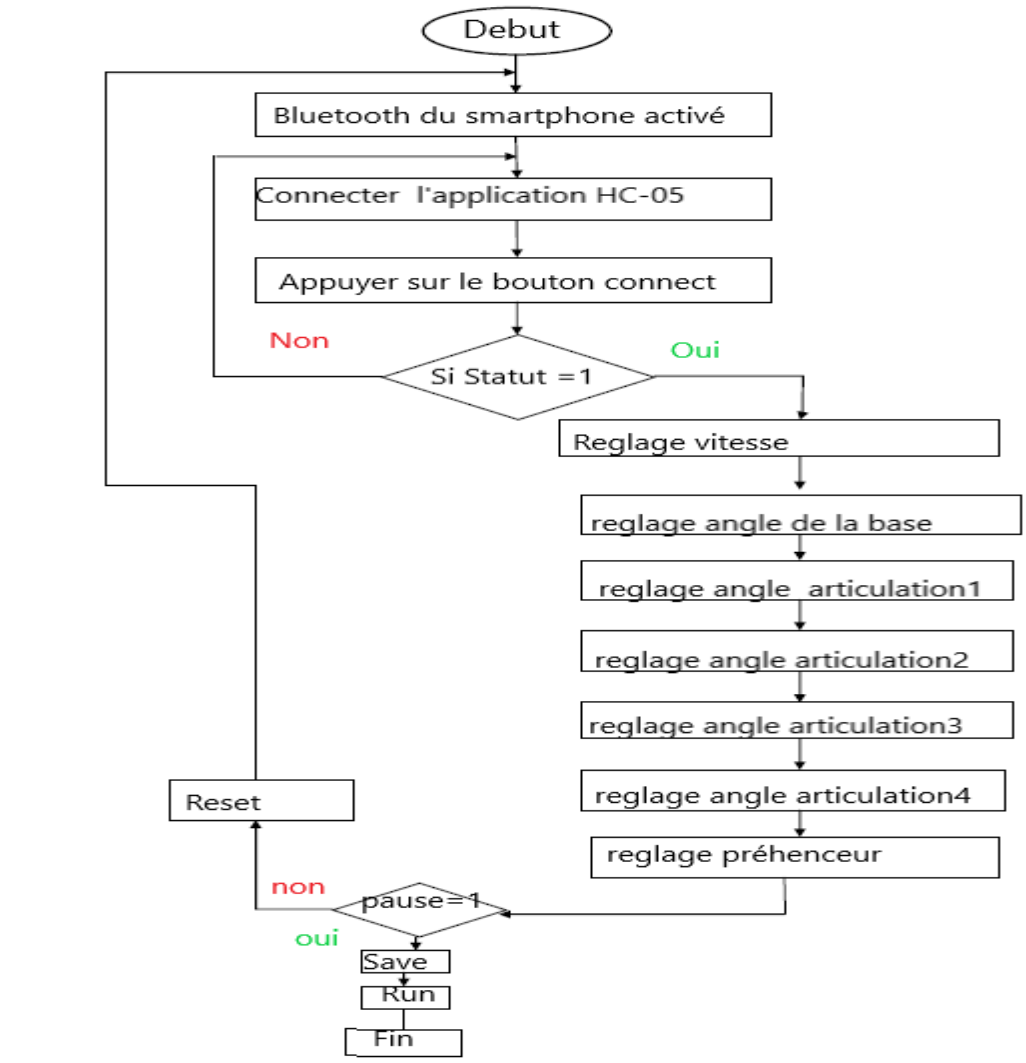


Figure 4.35 : Algorithme de fonctionnement du bras manipulateur.

4.6. Conclusion

Dans ce chapitre nous avons fait la description et la visualisation de la conception 3D des pièces du robot et de son assemblage finale, par la suite nous avons donner une explication détailler des étapes de création de notre application 3D, pour finir nous avons éclairci les diverses parties du programme de commande du bras manipulateur et expliquer le fonctionnement en donnant un exemple de déplacement d'un objet.

Conclusion générale

Ce travail nous a permis de réaliser un système électronique embarques qui est la commande d'un bras manipulateur anthropomorphe a six degrés de libertés avec une application Androïde par communication Bluetooth, capable de manipuler des objets ou d'exécuter des taches pénible.

Au début de ce travaille on a définit les robots de manière générale, et les bras manipulateurs de manière de manière détailler ainsi que leurs architecture complexe.

Nous avons déterminé les composants nécessaire et leur caractéristiques comme la carte Arduino Uno, le module Bluetooth HC-05 et les servomoteurs SG90 et MG996R.

Pour finir notre travail nous avons fait une description de la conception de notre bras manipulateur et nous avons expliqué la méthode de création de l'application Androïde qui le commande, et le fonctionnement du bras manipulateur.

On met en perspectives l'élaboration d'un bras manipulateur mobile qui peut réaliser plus de taches en un temps plus réduits et il pourras fonctionner dans un espace plus grand grâce a sa mobilité, également avec l'ajout de capteurs de sélection : de couleurs, et de matière .

Références bibliographique

[1] Definition d'un robot .[En ligne].Disponible sur : https://www.larousse.fr/dictionnaires/francais/robot/69647.
(consulté le 08/03/2020)

[2] Richa Braham, Bouyekhf Mouhamed Redha, «Etude et realisation d'un bras robot a 2DDL» ,Thèse de master, Faculté des Sciences et de la Technologie Département de Technologie, Khemis Miliana , Algerie, 2017.p4,6,7

[3] Fodil Rosa, Sayah Kenza, «Réalisation et Commande d'un Bras Manipulateur à Quatre Degrés de Liberté» ,Thèse de master, Faculté des Sciences de l'Ingénieur Département de maintenance industrielle Boumerdes , Algerie , 2017. p3,10,11

[4] Jean-Louis Boimond, Robotique, ISTIA, Université Angers, Angers. p7

[5] Belloufi Abderrahim , «Cours de robotique », Universite Kasdi Merbah Ouargla ,2015.

[6] Ahmed CHAÏB, « Cours ROBOTIQUE» ,Université M'hamed Bougara, Boumerdes, 2015 .p9,10

[7] Abdallah Ghoul, « Chapitre I Modélisation des robots manipulateurs», Université Tahri Mohammed Béchar, Février 2016. p1, 12 , 13

[8] Erik Savasgard , «Arduino 101 beguiners guide » Etats Unis ,2014.

[9] Arduino uno rev 3 .[En ligne].Disponible sur :https://store.arduino.cc/arduino-uno-rev3 .
(consulté le 4 Mai 2020)

[10]Arduino Bluetooth moule HC-05 pdf .[En ligne].Disponible sur :https://www.cours-gratuit.com/cours-arduino/arduino-bluetooth-module-hc-05-pdf-eng.consulter (consulté le 4 Mai 2020)

[11] Micro Servo Tower Pro SG90 .[En ligne].Disponible sur :https://shop.mchobby.be/fr/servo-moteurs/64-micro-servo-tower-pro-sg90-3232100000643.html?search_query=servomoteur+sg90&results=8
(consulté le 5 Mai 2020)

[12] MG996R Robot servo 180 .[En ligne].Disponible sur : http://www.towerpro.com.tw/product/mg995-robot-servo-180-rotation/ (consulté 5 Mai 2020)

[13] Introduction to arduino ide .[En ligne].Disponible sur : :https://www.theengineeringprojects.com/2018/10/introduction-to-arduino-ide.html
(consulté le 05 Mai 2020).

[14] What app inventor .[En ligne].Disponible sur :https://appinventor.mit.edu/explore/content/what-app-inventor consulter . (consulté le 5 Mai 2020)

[15] Proteus software introduction .[En ligne].Disponible sur : http://www.circuitstoday.com/proteus-software-introduction (consulté le 5 Mai 2020)

[16] 3D Builder . [En ligne] . Disponible sur : https://www.supinfo.com/articles/single/3464-3d-builder (consulté le 15 Juin 2020)

[17] Magesh Jayakumar ,« Arduino and Androide using Mit App inventor 2.0 » ,Etats Unis , 2016.

[18] DIY Arduino Robot Arm with Smartphone Control . [En ligne] . Disponible sur : https://howtomechatronics.com/tutorials/arduino/diy-arduino-robot-arm-with-smartphone-control/ (consulté le 10 Juin 2020)

Annexe

```
#include <SoftwareSerial.h>
#include <Servo.h>

Servo servo01;
Servo servo02;
Servo servo03;
Servo servo04;
Servo servo05;
Servo servo06;

SoftwareSerial Bluetooth(3, 4); // Arduino(RX, TX) - HC-05 Bluetooth (TX, RX)

int servo1Pos, servo2Pos, servo3Pos, servo4Pos, servo5Pos, servo6Pos; // current position
int servo1PPos, servo2PPos, servo3PPos, servo4PPos, servo5PPos, servo6PPos; // previous position
int servo01SP[50], servo02SP[50], servo03SP[50], servo04SP[50], servo05SP[50], servo06SP[50]; // for storing positions/steps
int speedDelay = 20;
int index = 0;
String dataIn = "";

void setup() {
  servo01.attach(5);
  servo02.attach(6);
  servo03.attach(7);
  servo04.attach(8);
  servo05.attach(9);
  servo06.attach(10);
  Bluetooth.begin(38400); // Default baud rate of the Bluetooth module
  Bluetooth.setTimeout(1);
  delay(20)
```

```
  // Robot arm initial position
  servo1PPos = 90;
  servo01.write(servo1PPos);
  servo2PPos = 150;
  servo02.write(servo2PPos);
  servo3PPos = 35;
  servo03.write(servo3PPos);
  servo4PPos = 140;
  servo04.write(servo4PPos);
  servo5PPos = 85;
  servo05.write(servo5PPos);
  servo6PPos = 80;
  servo06.write(servo6PPos);
}

void loop() {
  // Check for incoming data
  if (Bluetooth.available() > 0) {
    dataIn = Bluetooth.readString();  // Read the data as string

    // If "Waist" slider has changed value - Move Servo 1 to position
    if (dataIn.startsWith("s1")) {
      String dataInS = dataIn.substring(2, dataIn.length()); // Extract only the number. E.g.
from "s1120" to "120"
      servo1Pos = dataInS.toInt();  // Convert the string into integer
      // We use for loops so we can control the speed of the servo
      // If previous position is bigger then current position
      if (servo1PPos > servo1Pos) {
        for ( int j = servo1PPos; j >= servo1Pos; j--) {   // Run servo down
          servo01.write(j);
          delay(20);    // defines the speed at which the servo rotates
        }
```

```
  // If previous position is smaller then current position
  if (servo1PPos < servo1Pos) {
    for ( int j = servo1PPos; j <= servo1Pos; j++) {   // Run servo up
      servo01.write(j);
      delay(20);
    }
  }
  servo1PPos = servo1Pos;   // set current position as previous position
}

// Move Servo 2
if (dataIn.startsWith("s2")) {
  String dataInS = dataIn.substring(2, dataIn.length());
  servo2Pos = dataInS.toInt();

  if (servo2PPos > servo2Pos) {
    for ( int j = servo2PPos; j >= servo2Pos; j--) {
      servo02.write(j);
      delay(50);
    }
  }
  if (servo2PPos < servo2Pos) {
    for ( int j = servo2PPos; j <= servo2Pos; j++) {
      servo02.write(j);
      delay(50);
    }
  }
  servo2PPos = servo2Pos;
}
// Move Servo 3
if (dataIn.startsWith("s3")) {
  String dataInS = dataIn.substring(2, dataIn.length());
  servo3Pos dataInS.toInt();
```

```
  if (servo3PPos > servo3Pos) {
    for ( int j = servo3PPos; j >= servo3Pos; j--) {
      servo03.write(j);
      delay(30);
    }
  }
  if (servo3PPos < servo3Pos) {
    for ( int j = servo3PPos; j <= servo3Pos; j++) {
      servo03.write(j);
      delay(30);
    }
  }
  servo3PPos = servo3Pos;
}
// Move Servo 4
if (dataIn.startsWith("s4")) {
  String dataInS = dataIn.substring(2, dataIn.length());
  servo4Pos = dataInS.toInt();
  if (servo4PPos > servo4Pos) {
    for ( int j = servo4PPos; j >= servo4Pos; j--) {
      servo04.write(j);
      delay(30);
    }
  }
  if (servo4PPos < servo4Pos) {
    for ( int j = servo4PPos; j <= servo4Pos; j++) {
      servo04.write(j);
      delay(30);
    }
  }
  servo4PPos = servo4Pos;
}
// Move Servo 5
```

```
if (dataIn.startsWith("s5")) {
  String dataInS = dataIn.substring(2, dataIn.length());
  servo5Pos = dataInS.toInt();
  if (servo5PPos > servo5Pos) {
    for ( int j = servo5PPos; j >= servo5Pos; j--) {
      servo05.write(j);
      delay(30);
    }
  }
  if (servo5PPos < servo5Pos) {
    for ( int j = servo5PPos; j <= servo5Pos; j++) {
      servo05.write(j);
      delay(30);
    }
  }
  servo5PPos = servo5Pos;
}
// Move Servo 6
if (dataIn.startsWith("s6")) {
  String dataInS = dataIn.substring(2, dataIn.length());
  servo6Pos = dataInS.toInt();
  if (servo6PPos > servo6Pos) {
    for ( int j = servo6PPos; j >= servo6Pos; j--) {
      servo06.write(j);
      delay(30);
    }
  }
  if (servo6PPos < servo6Pos) {
    for ( int j = servo6PPos; j <= servo6Pos; j++) {
      servo06.write(j);
      delay(30);
    }
  }
```

```
      servo6PPos = servo6Pos;
    }
    // If button "SAVE" is pressed
    if (dataIn.startsWith("SAVE")) {
      servo01SP[index] = servo1PPos;  // save position into the array
      servo02SP[index] = servo2PPos;
      servo03SP[index] = servo3PPos;
      servo04SP[index] = servo4PPos;
      servo05SP[index] = servo5PPos;
      servo06SP[index] = servo6PPos;
      index++;                    // Increase the array index
    }
    // If button "RUN" is pressed
    if (dataIn.startsWith("RUN")) {
      runservo();  // Automatic mode - run the saved steps
    }
    // If button "RESET" is pressed
    if ( dataIn == "RESET") {
      memset(servo01SP, 0, sizeof(servo01SP)); // Clear the array data to 0
      memset(servo02SP, 0, sizeof(servo02SP));
      memset(servo03SP, 0, sizeof(servo03SP));
      memset(servo04SP, 0, sizeof(servo04SP));
      memset(servo05SP, 0, sizeof(servo05SP));
      memset(servo06SP, 0, sizeof(servo06SP));
      index = 0;  // Index to 0
    }
  }
}

// Automatic mode custom function - run the saved steps
void runservo() {
  while (dataIn != "RESET") {    // Run the steps over and over again until "RESET" button is pressed
```

```
for (int i = 0; i <= index - 2; i++) {  // Run through all steps(index)
  if (Bluetooth.available() > 0) {      // Check for incomding data
    dataIn = Bluetooth.readString();
    if ( dataIn == "PAUSE") {           // If button "PAUSE" is pressed
      while (dataIn != "RUN") {         // Wait until "RUN" is pressed again
        if (Bluetooth.available() > 0) {
          dataIn = Bluetooth.readString();
          if ( dataIn == "RESET") {
            break;
          }
        }
      }
    }
    // If speed slider is changed
    if (dataIn.startsWith("ss")) {
      String dataInS = dataIn.substring(2, dataIn.length());
      speedDelay = dataInS.toInt(); // Change servo speed (delay time)
    }
  }
  // Servo 1
  if (servo01SP[i] == servo01SP[i + 1]) {
  }
  if (servo01SP[i] > servo01SP[i + 1]) {
    for ( int j = servo01SP[i]; j >= servo01SP[i + 1]; j--) {
      servo01.write(j);
      delay(speedDelay);
    }
  }
  if (servo01SP[i] < servo01SP[i + 1]) {
    for ( int j = servo01SP[i]; j <= servo01SP[i + 1]; j++) {
      servo01.write(j);
      delay(speedDelay);
    }
```

```
}

// Servo 2
if (servo02SP[i] == servo02SP[i + 1]) {
}
if (servo02SP[i] > servo02SP[i + 1]) {
  for ( int j = servo02SP[i]; j >= servo02SP[i + 1]; j--) {
    servo02.write(j);
    delay(speedDelay);
  }
}
if (servo02SP[i] < servo02SP[i + 1]) {
  for ( int j = servo02SP[i]; j <= servo02SP[i + 1]; j++) {
    servo02.write(j);
    delay(speedDelay);
  }
}

// Servo 3
if (servo03SP[i] == servo03SP[i + 1]) {
}
if (servo03SP[i] > servo03SP[i + 1]) {
  for ( int j = servo03SP[i]; j >= servo03SP[i + 1]; j--) {
    servo03.write(j);
    delay(speedDelay);
  }
}
if (servo03SP[i] < servo03SP[i + 1]) {
  for ( int j = servo03SP[i]; j <= servo03SP[i + 1]; j++) {
    servo03.write(j);
    delay(speedDelay);
  }
```

```
// Servo 4
        if (servo04SP[i] == servo04SP[i + 1]) {
        }
        if (servo04SP[i] > servo04SP[i + 1]) {
          for ( int j = servo04SP[i]; j >= servo04SP[i + 1]; j--) {
            servo04.write(j);
            delay(speedDelay);
          }
        }
        if (servo04SP[i] < servo04SP[i + 1]) {
          for ( int j = servo04SP[i]; j <= servo04SP[i + 1]; j++) {
            servo04.write(j);
            delay(speedDelay);
          }
        }

        // Servo 5
        if (servo05SP[i] == servo05SP[i + 1]) {
        }
        if (servo05SP[i] > servo05SP[i + 1]) {
          for ( int j = servo05SP[i]; j >= servo05SP[i + 1]; j--) {
            servo05.write(j);
            delay(speedDelay);
          }
        }
        if (servo05SP[i] < servo05SP[i + 1]) {
          for ( int j = servo05SP[i]; j <= servo05SP[i + 1]; j++) {
            servo05.write(j);
            delay(speedDelay);
          }
        }

        // Servo 6
        if (servo06SP[i] == servo06SP[i + 1]) {
```

```
 }
 if (servo06SP[i] > servo06SP[i + 1]) {
  for ( int j = servo06SP[i]; j >= servo06SP[i + 1]; j--) {
    servo06.write(j);
    delay(speedDelay);
  }
 }
 if (servo06SP[i] < servo06SP[i + 1]) {
  for ( int j = servo06SP[i]; j <= servo06SP[i + 1]; j++) {
    servo06.write(j);
    delay(speedDelay);
  }
 }
}
}
```