République **A**lgérienne **D**émocratique et **P**opulaire
Ministère de L'**E**nseignement **S**upérieur et de la
Recherche **S**cientifique

Université **M**ouloud **M**ammeri De **T**izi-**O**uzou

Faculté De **G**énie **E**lectrique Et D'informatique
DEPARTEMENT D'AUTOMATIQUE

Projet de Fin d'Etude
De MASTER PROFESSIONNEL
Spécialité : Automatique industrielle

Présenté par
Jugurtha LAOUEDJ
Abderrahmane DJERIANI

Mémoire dirigé par **Mr. MELLAH Rabah**

Thème

Commande vocale d'un rover via Bluetooth à base d'Arduino

Mémoire soutenu publiquement le 30 / 09 / 2018 devant le jury composé de :

M Said GUERMAH
Grade, Lieu d'exercice, Président

M Rabah MELLAH
Grade, Lieu d'exercice, Rapporteur

M Mohamed ALI BEY
Grade, Lieu d'exercice, Examinateur

2017/2018

Remerciements

D'abord nous remercions le bon Dieu à nous avoir donné la force, la volonté, et le courage pour réaliser ce travail.

Nous tenons à remercier vivement notre promoteur Mr. MELAH RABAH pour son orientation et sa disponibilité constante tout au long de notre travail

Nos remerciements vont également aux membres de jury qui ont accepté d'évaluer notre travail.

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

Dédicace

Ma Mère, Mon Père

Affable, honorable, aimable, vous représentes pour moi
Le symbole de la bonté par excellence, la source de tendresse
Et l'exemple du
Dévouement qui n 'a pas cessé de m'encourager et de prier
Pour moi.
Soyez sûrs que je continuerai mon chemin.
Je vous dédie ce travail en témoignage de mon profond amour.
Puisse Dieu, le tout puissant, vous préserver et vous accorder santé,
Longue vie et bonheur.
A mes sœurs
A mes frères
A tous les membres de ma famille, petits et grands
A tous mes amis de proches
Veuillez trouver dans ce modeste travail l'expression de mon
Affect

A.DJERIANI

Dédicace

Je dédie ce travail à mes très chers parents qui ont toujours était là pour moi, et qui m'ont donné un magnifique modèle de labeur et Persévérance

A ma chère mère qui a éclairé mon chemin en témoignage de ma Profonde gratitude et de mon incontestable reconnaissance, pour tous les sacrifices qu'elle me contente, toute la confiance qu'elle m'accorde et tout l'amour dont elle m'entoure.

A mon cher père qui m'a indiqué la bonne voie

A ma compagnonne de vie qui a été toujours avec moi en me Rappelant que la volonté fait toujours les grands hommes aussi Grâce à son encouragement, son soutien moral en exprimant mes Gratitudes.

Que Dieu vous garde.

À Toute mes frères et mes sœurs.

À Tous ma famille.

À tous mes amis.

À Toute les étudiants.

À Toute la Promotion Automatique 2017.

À Toute les pompiers de tizi ouzou

L.DJUGURTHA

SOMMAIRE

SOMMAIRE

CHAPITRE II : CARTES ARDUINO & SYSTÈME ANDROID

CHAPITRE III : RÉALISATION & MISE EN ŒUVRE DU ROVER

SOMMAIRE

Liste des figures

CHAPITRE I : GÉNÉRALITÉ SUR LA ROBOTIQUE

CHAPITRE II : CARTES ARDUINO & SYSTÈME ANDROID

Liste des figures

CHAPITRE III : RÉALISATION & MISE EN ŒUVRE DU ROVER

Liste des tableaux

Introduction générale

La robotique permet d'aider l'homme dans les tâches difficiles, répétitives ou Pénibles. De plus elle constitue le rêve de substituer la machine à l'homme dans ces tâches.
Les facultés de perception et de raisonnement des robots progressent chaque jour actuellement et plus encore dans l'avenir, ils sont appelés à jouer un rôle de plus en plus
Important dans notre vie.
La robotique comporte deux grands pôles d'intérêt, la robotique de manipulation (robotique industrielle) et la robotique mobile. Un des problèmes majeurs de la robotique mobile est la planification de mouvement. Autour de ce problème de planification de mouvement de nombreuses études ont été réalisées dans le but de développer des méthodes générales pour guider les robots.
Les robots de la première génération n'étaient que des simples automates capables d'accomplir des tâches répétitives dans des milieux parfaitement connus. La deuxième génération a vu apparaître des robots équipés de capteurs leur permettant de percevoir les modifications de l'environnement et d'agir d'une manière simple. Depuis, les robots se sont largement développés et généralisés. Ils sont dotés de moyens de perception et de décision leur permettant de comprendre et d'évoluer dans différents types d'environnement. Certains sont même utilisés pour l'exploration de planètes lointaines (Lune, Mars). La dernière génération des robots devrait être à haut degré d'autonomie, elle fait l'objet des recherches intensives. On s'est de plus en plus rendu compte, que pour plusieurs années encore, une coopération entre l'homme et la machine est inévitable.
C'est dans ce cadre que se situe notre projet de fin d'études intitulé « commande d'un rover avec la voix via Bluetooth ». Notre but principal est de commander un rover on utilise la voix via Bluetooth pour permettre aux utilisateurs de contrôler ce rover par l'intermédiaire d'un Smartphone Android. Pour réaliser notre objectif on a utilisé une carte Arduino ATmega2560, ainsi que d'autres équipements tels que le double pont-H, des moteurs, des capteurs, un module Bluetooth.

Notre projet s'articule autour de deux parties :

- La partie contrôle : c'est l'ensemble des composants mécaniques, électroniques et informatique composant le rover, l'essentiel de cette partie est la carte Arduino connectée avec le Shield.
- La partie commande : c'est une application Android qui permet aux clients de commander le rover en utilisant sa voix avec une liaison Bluetooth.

Ces deux parties sont relies par un module Bluetooth qui permet d'assurer la communication.

Ainsi que notre travail est divisé en trois chapitres :

- Le premier chapitre consiste à introduire une description sur la robotique.
- Le second chapitre est consacré à l'Arduino et le système Android.
- Le troisième chapitre illustre la conception optée pour la réalisation de notre projet.

Enfin on achève ce mémoire par une conclusion générale récapitulant tout notre travail ainsi que ses perspectives.

I.1 Introduction :

Ces dernières années, les robots manipulateurs ont un impact considérable sur de nombreux aspects de la vie moderne, de la fabrication industrielle aux soins de santé, le transport et l'exploration de l'espace et les profondeurs de la mer. Ceci s'explique par le fait que de plus en plus, les processus industriels sont automatisés, afin de répondre à des problématiques de maîtrise de la qualité produite, de la productivité et des coûts. Une des méthodes d'automatisation repose sur la robotisation. Dans les prochaines années, des robots seront aussi omniprésents dans des processus industriels comme les ordinateurs personnels. En outre, ces robots utilisent des capteurs pour bien mener les déférentes tâches et accomplir leurs missions.

Dans le but d'expliquer l'utilité des robots, ce chapitre sera consacré à la présentation des généralités sur la robotique.

I.2 La Robotique :

I.2.1 Historique de la robotique :

La robotique est passée par plusieurs générations comme suit **[1]** :

- 1947 : Premier manipulateur électrique télé-opéré.
- 1954 : Premier robot programmable.
- 1961 : Utilisation d'un robot industriel, commercialisé par la société UNIMATION (USA), sur une chaîne de montage de General Motors.
- 1961 : Premier robot avec contrôle en effort.
- 1963 : Utilisation de la vision pour commander un robot.
- 1978 : Le robot ARGOS. Développé à l'Université Paul Sabatier de Toulouse (France). Le robot ARGOS simule la navigation d'un robot mobile équipé d'un système de vision au fur et à mesure de ses déplacements.
- 1979 : Le robot HILARE. Les chercheurs du L.A.A.S. de Toulouse (France) étudièrent la planification des trajectoires d'un robot mobile ponctuel, dans un environnement totalement connu.
- 1981 : Le robot VESA. Ce robot, construit à l'I.N.S.A (France) de Rennes, il est équipé d'un arceau de sécurité pour réaliser la détection d'obstacles dans un environnement totalement inconnu.

- 1984 : Le robot FLAKEY. Ce robot, conçu et construit au Stanford Research Institute est le reflet des améliorations apportées par 14 années de développement. Le robot FLAKEY est équipé de deux roues motrices avec encodeurs, mais sa vitesse maximale est de 66 cm/s au lieu de quelques centimètres par seconde. Ce robot est capable de naviguer dans des environnements réels.
- 1993 : Les robots ERRATIC et PIONNER. Le robot ERRATIC a été conçu par Kurt Konolige, au Stanford Research Institue, comme un robot mobile de faible coût pour ses cours de robotique.
- Les robots mobiles actuels : A présent la plupart des travaux de recherche portent sur les problèmes de perception. Ainsi, la planification de trajectoires, l'analyse et la modélisation de l'environnement de robot, sont appliquées sur des robots mobiles commerciaux. Également la recherche actuelle s'oriente vers la conception mécanique des robots mobiles pour des applications hautement spécialisées, comme l'exploration sous-marine, les robots volants et le micro robot.

I.2.2 Définition :

Le terme (Robot) prend son origine du mot slave (Paboma) (se prononce robota) qui veut dire en russe travail ou en tchèque corvée ou travail forcé. II désigne aussi une machine à l'aspect humain, capable de se mouvoir et d'agir grâce à un mécanisme automatique pouvant effectuer certaines opérations, et capable par fois de modifier de lui-même son cycle de fonctionnement et d'exercer un certain choix.

I.2.3 Les types des robots :

Il existe deux grandes familles de robots :

- Les robots manipulateurs (fixe).
- Les robots mobiles.

I.2.3.1 Les robots manipulateurs :

Un robot manipulateur se présente sous forme d'un bras doté d'un certain nombre de segments articulés. Il est conçu pour manipuler ou déplacer des matériaux, outils et pièces sans contact humain direct. En parallèle, les robots sont des dispositifs qui permettent aux humains d'interagir avec des objets dans un environnement en toute sécurité. En effets, les

robots manipulateurs sont utilisés dans des applications industrielles pour effectuer efficacement des tâches telles que l'assemblage, soudage, traitement de surface, et le forage.

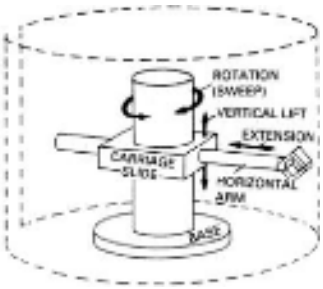
I.2.3.2 Les types des robots manipulateurs :

Les robots manipulateurs se présentent sous plusieurs formes, qui sont réparties en cinq grandes catégories :

- Robots cylindriques
- Robots rectilignes
- Robots sphériques
- Robots articulés
- Robots SCARA

a) Robots cylindriques :

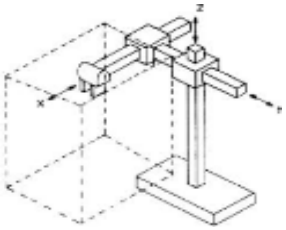
Le robot cylindrique, comme l'illustre la figure I.1, est composé essentiellement de deux axes de mouvement, dont le premier pour le mouvement en haut et en bas et le second est représenté par la rotation à travers la jonction de base. De plus, le bras horizontal peut se déplacer à l'intérieur et à l'extérieur, ce qui donne un troisième axe de mouvement **[2].**



FigureI.1 *: Robot cylindrique.*

b) Robots rectilignes :

Le robot rectiligne est représenté à la figure I.2possède trois axes de mouvement (x, y, z). Pour cette raison, il est parfois appelé Robot cartésien. Les robots de type de configuration exploitent le vérin pneumatique **[2].**



FigureI.2 *: Robot rectiligne.*

c) Robots sphériques :

Le robot sphérique donné à la figure I.3 est de grande taille avec un bras télescopique qui assure un mouvement à l'intérieur ou à l'extérieur. Les mouvements de base du robot sphérique sont des rotations qui s'effectuent à la base et la rotation angulaire du bras horizontal, soit vers le haut ou vers le bas en haut ou en bas **[2].**

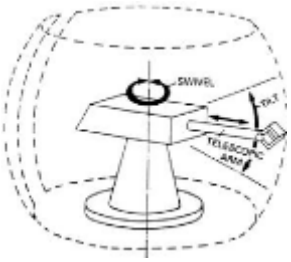


Figure I.3*: Robot sphérique.*

d) Robots articulés :

Le bras articulé du robot comme l'illustre la figure I.4 ressemble à un bras humain. Il se compose de deux éléments, nommés l'avant-bras et le bras supérieur **[3]**

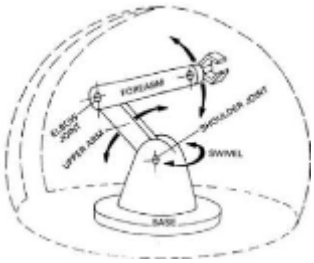


Figure I.4*: Robot articulé.*

e) Robots SCARA :

Le terme « SCARA » signifie « Sélective Compliance Arm for Robot Assembly »
Un robot SCARA est défini dans la norme ISO, en tant qu'un robot comportant deux liaisons pivotes parallèles pour fournir conformément à un plan sélectionné comme la montre la figure I.5. Il est peut-être considéré comme un cas particulier d'un robot cylindrique **[4].**

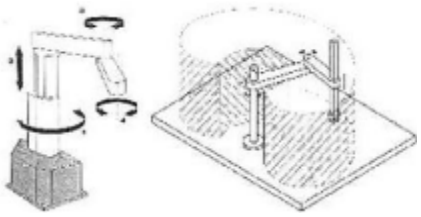


Figure I.5*: Robot SCARA.*

I.2.3.3 Les robots mobiles :

Un robot mobile est celui qui peut se déplacer dans son environnement de façon autonome. Pour ce faire, le robot doit pouvoir naviguer. Par conséquent, la portée et la précision de navigation requise varie en fonction de la taille du robot et du type de la tâche à réaliser.

I.2.3.4 Architecture des robots mobiles :

L'architecture des robots mobiles, est présentée à la figure I.6. Elle composée de quatre éléments à savoir :

- La structure mécanique et la motricité.
- Les organes de sécurité.
- Le système de traitement des informations et gestion des tâches.
- Le système de localisation.

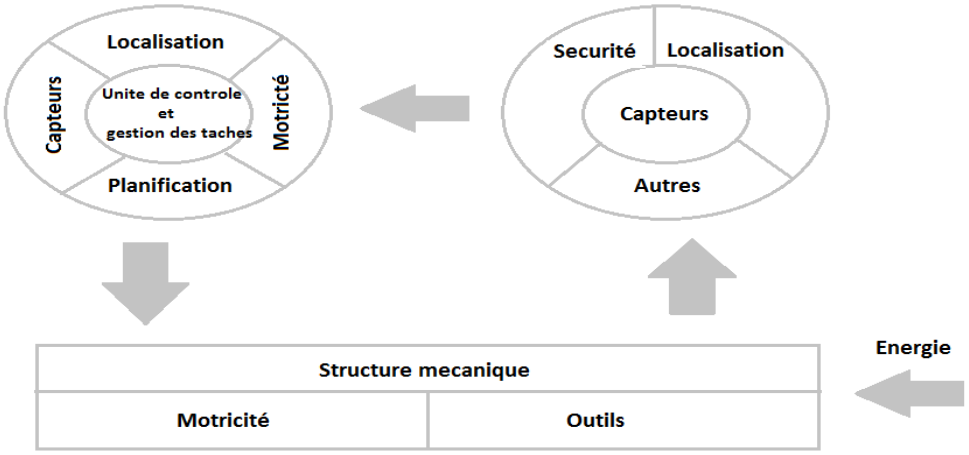


***Figure I.6**: Architecture d'un robot mobile.*

I.2.3.5 Classification des Robots Mobiles :

Selon le système de locomotion, on peut distinguer quatre types de robots mobiles :

1. **Les robots mobiles à roues :**

La mobilité par roues est la structure mécanique la plus utilisée. Ce type de robot assure un déplacement avec une accélération et une vitesse rapide mais nécessite un sol relativement plat. On distingue plusieurs classes de robots à roues, déterminées principalement par la position et le nombre de roues utilisées.

Nous citerons ici les quatre classes principales de robots mobiles à roues **[5]** :

a) Robot unicycle :

Un robot de type unicycle est actionné par deux roues indépendantes comme la montre la figure I.7, il possède éventuellement des roues folles pour assurer sa stabilité. Son centre de rotation est situé sur l'axe reliant les deux roues motrices.
En effet, le fait que c'est un robot non-holonome, il est impossible de le déplacer dans une direction perpendiculaire aux roues de locomotion.
Sa commande peut être très simple, car il est assez facile de le déplacer d'un point à un autre par une suite de rotations simples et de lignes droites **[6].**

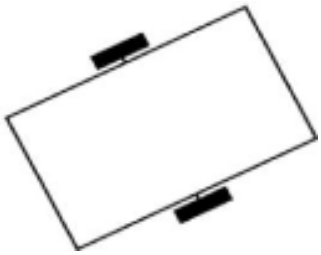


***Figure I.7**: Robot mobile unicycle.*

b) Robot tricycle

Un robot de type tricycle représenté par la figure I.8 est constitué de deux roues fixes placées sur un même axe et d'une roue centrée orientable placée sur l'axe longitudinal. Le mouvement du robot est donné par la vitesse des deux roues fixes et par l'orientation de la roue orientable. Son centre de rotation est situé à l'intersection de l'axe contenant les roues fixes et de l'axe de la roue orientable.

Né au moins, le fait que c'est un robot non-holonome, il est impossible de le déplacer dans une direction perpendiculaire aux roues fixes. Sa commande est plus compliquée. Il est en général impossible d'effectuer des rotations simples à cause d'un rayon de braquage limité de la roue orientable **[6].**

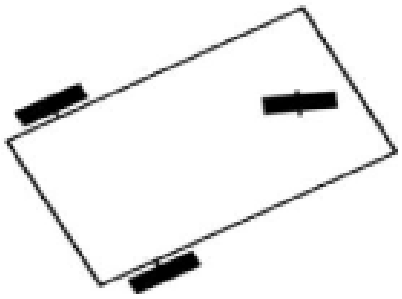


***Figure I.8**: Robot mobile tricycle.*

c) Robot voiture :

Un robot de type voiture donné à la figure I.9 est semblable au tricycle, il est constitué de deux roues fixes placées sur un même axe et de deux roues centrées orientables placées elles aussi sur un même axe.

Le robot de type voiture est cependant plus stable puisqu'il possède un point d'appui supplémentaire.

Toutes les autres propriétés du robot voiture sont identiques au robot tricycle, en raison que le robot voiture peut être ramené au robot tricycle en remplaçant les deux roues avant par une seule placée au centre de l'axe, et ceci de manière à laisser le centre de rotation inchangé**.**

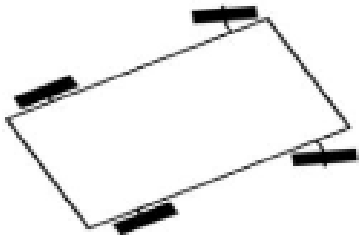


Figure I.9*: Robot mobile voiture*

e) Robot omnidirectionnel :

Un robot omnidirectionnel représenté à la figure I.10 est un robot qui peut se déplacer librement dans toutes les directions. Il est en général constitué de trois roues décentrées, orientables et placées en triangle équilatéral.

L'énorme avantage du robot omnidirectionnel est qu'il est holonome puisqu'il peut se déplacer dans toutes les directions. Mais ceci au détriment d'une complexité mécanique bien plus grande **[6].**

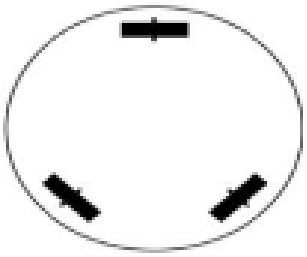


Figure I.10*: Robot mobile omnidirectionnel.*

- **Comparaison des différents types de robots mobiles à roues :**

Nous pouvons observer dans le tableau ci-dessous un récapitulatif des avantages et des inconvénients des différents types de robots à roues :

Type du robot	Avantage	Inconvénient
Unicycle	* Stable * Rotation sur soi-même * Complexité mécanique faible	* Non-holonome
Tricycle	* Complexité mécanique Modérée	* Non-holonome * Peu stable * Pas de rotation sur soi-même
Voiture	* Stable * Complexité mécanique modérée	* Non-holonome * Pas de rotation sur soi-même
Omnidirectionnel	* Holonome * Stable * Rotation sur soi-même	* Complexité mécanique Importante

Tab I.1 : différents types de robots à roues

2. Les robots mobiles à chenilles :

Les robots mobiles à chenilles illustrés par la figure I.11, présentent l'avantage d'une bonne adhérence au sol et d'une faculté de franchissement d'obstacles. L'utilisation de ces robots est orientée vers l'emploi sur sol accidenté ou de mauvaise qualité au niveau de l'adhérence (présence de boue, herbe...).

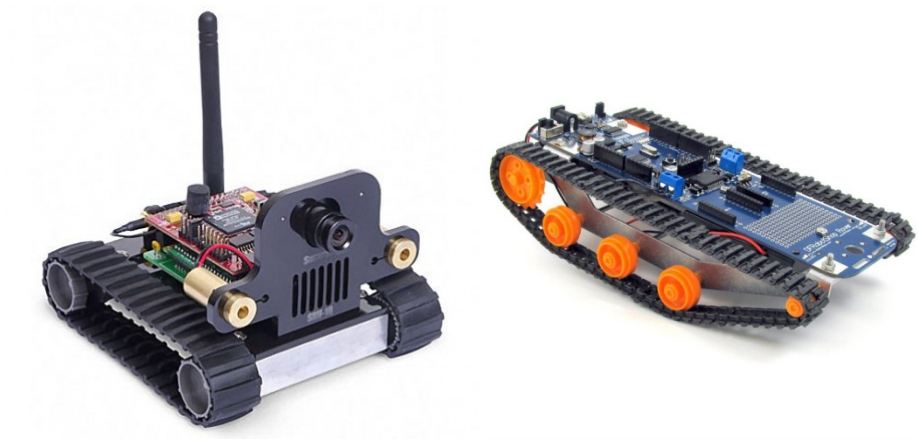


Figure I.11*: Robots mobiles à chenilles.*

3. Les robots mobiles marcheurs :

Les robots mobiles marcheurs donnés à la figure I.12 sont destinés à réaliser des tâches variées dont l'accès à un site est difficile et dangereux pour l'être humain. Leur structure est dotée de plusieurs degrés de liberté ce qui permet un rapprochement avec les robots manipulateurs. On distingue les robots marcheurs (figure à deux jambes (humanoïdes), à quatre pattes (type cheval), et à six pattes (type araignée).

***Figure I.12**: Robots mobiles marcheurs.*

4. Les robots mobiles rampants :

La reptation est une solution de locomotion pour un environnement de type « tunnel » qui conduit à réaliser des structures filiformes. Le système est composé d'un ensemble de modules ayant chacun plusieurs mobilités. Ici aussi les techniques utilisées découlent des méthodes de locomotion des animaux et des insectes comme la montre la figure I.13.
Les applications de ce type de robots sont très spécialisées et les architectures des robots sont en général spécifiques à l'application visée **[7]**.

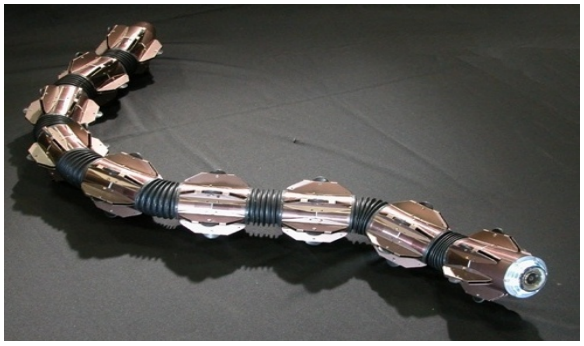


***Figure I.13**: Robot mobile rampant.*

I.2.4 Domaines d'utilisation des robots :

- **Les robots industriels :** sont des robots utilisés dans un environnement de fabrication industrielle. Ils sont utilisés dans la fabrication des automobiles, des composants et des pièces électroniques, des médicaments et de nombreux produits.

- **Robots domestiques ou ménagers :** Robots utilisés à la maison. Ce type de robots comprend de nombreux appareils très différents, tel que les aspirateurs robotiques, robots nettoyeurs de piscines, balayeuses, nettoyeurs gouttières et autres robots qui peuvent faire différentes tâches.
- **Robots en médecine et chirurgie :** Les robots semblent avoir de l'avenir à l'hôpital. Par exemple le Robodoc aide à réaliser certaines opérations de chirurgie. Le robot infirmier est encore en projet. Le cyber squelette HAL aide les personnes à se déplacer. Ainsi, le robot patient permet aux futurs chirurgiens-dentistes d'apprendre à soigner sans faire des dégâts.

I.2.5 Avantages et inconvénients des robots :

Un système robotique consiste non seulement en des robots mais aussi d'autres dispositifs et systèmes qui sont utilisés avec le robot pour effectuer la tâche nécessaire.

- **Les avantages des robots sont [8] :**
  - La robotique et l'automatisation peut dans de nombreuses situations accroître la productivité, la sécurité, l'efficacité, la qualité et la cohérence des produits.
  - Les robots peuvent travailler dans un environnement dangereux, sans besoin de soutien de vie, ou de préoccupations concernant la sécurité.
  - Les Robots n'ont pas besoin d'éclairage, de climatisation, de ventilation ou de protection contre le bruit.
  - Les robots travaillent continuellement, sans ressentir une fatigue ou l'ennui, et ne nécessitent pas une assurance médicale ou de vacances.
  - Les robots sont de précision répétable à tous les moments.
  - Les robots peuvent être beaucoup plus précis que les êtres humains. Précision linéaire d'un robot typiquement est de 10 à 20 microns.
  - Ils se perfectionnent sans cesse.

❖ **L'inconvénient des robots :**

- Ils manquent de capacité de réagir en cas d'imprévu, à moins que les situations soient comprises et les réponses soient incluses dans le système. Les mesures de sécurité sont nécessaires pour s'assurer qu'ils ne blessent pas les opérateurs et n'endommagent pas les machines. On peut également citer d'autres inconvénients comme **[8] :**
- Réponse inadéquate.
- Le manque de prise de décision,
- Consommation de l'énergie.
- Ils peuvent causer des dommages à d'autres appareils, et la blessure à l'homme.
- Les robots sont coûteux en raison du coût initial de l'équipement, d'installation, le besoin de périphériques, le besoin de formation et la nécessité de la programmation (les robots ont besoin des humains).
- Peur de voir les robots voler nos emplois
- Les capacités des robots sont limitées

I.3 Conclusion

A travers ce chapitre nous avons présenté une introduction aux deux catégories principales des robots, ainsi que d'autres spécificités de la robotique. Nous avons ensuite abordé le domaine d'utilisation des robots, ainsi que leurs avantages et inconvénients. Nous avons pu Constater l'évolution actuelle des robots proposés par divers constructeurs permettant de présenter une alternative aux machines-outils en utilisant une commande adéquate.

Le chapitre suivant s'intéresse ainsi à la présentation des cartes Arduino et le système Android.

II.1 Introduction

Aujourd'hui, l'électronique est de plus en plus remplacée par de l'électronique programmée.
L'Arduino est une carte basée sur un microcontrôleur (mini-ordinateur) Atmel ATMEGA8 ou ATMEGA168. Elle dispose dans sa version de base de 1 Ko de mémoire vive, et 8Ko de mémoire flash pour stocker ses programmes. Elle peut être connectée à 13 entrées ou sorties numériques, dont 3 PWM (pouvant donner 3 sorties analogiques et 6 entrées analogiques convertissant en 10 bit.
Dans la version la plus courante, la communication avec l'ordinateur se fait par un port USB. Il existe plusieurs versions de l'Arduino, dont une version miniaturisée, et d'autres projets sont également en gestation. La carte dispose d'un logiciel système interne (modifiable) et des programmes utilisateur.

II.2 Arduino

II.2.1 Présentation Arduino :

Arduino est une plate-forme de prototypage d'objets interactifs à usage créatif constituée d'une carte électronique et d'un environnement de programmation. Sans tout ne connaître ni tout comprendre de l'électronique, cet environnement matériel et logiciel permet à l'utilisateur de formuler ses projets par l'expérimentation directe avec l'aide de nombreuses ressources disponibles en ligne.
Pont tendu entre le monde réel et le monde numérique, Arduino permet d'étendre les capacités de relations humain/machine ou environnement/machine.
Arduino est un projet en source ouverte, la communauté importante d'utilisateurs et de concepteurs permet à chacun de trouver les réponses à ces questions **[9] :**

- ➢ Ça sert à quoi ?
- ➢ C'est pour qui ?
- ➢ Qu'est-ce que c'est ?
- ➢ Arduino est-il un microcontrôleur ?
- ➢ Comment ça marche ?
- ➢ Faut-il des connaissances en électronique ?

II.2.1.1 Arduino : ça sert à quoi ?

Arduino est un circuit imprimé open source. Vous connaissez déjà sûrement ce qualificatif pour des logiciels gratuits, où chacun peut prendre part au développement du projet et y apporter sa contribution. Ce mode de collaboration, réunissant beaucoup de personnes intéressées et engagées, possède un fort potentiel et fait clairement avancer les choses. Les résultats parlent d'eux-mêmes et n'ont rien à envier à ceux des projets commerciaux.

Sous ce nom Arduino se cachent non seulement du matériel mais aussi un logiciel. On parle alors de Physical Computing, qui désigne la construction de systèmes interactifs permettant de connecter le monde physique à celui des ordinateurs. Le monde dans lequel nous vivons est considéré comme un système analogique, alors que les ordinateurs agissent dans un environnement numérique ne connaissant que les états logiques O et I. C'est à nous, individus créatifs, qu'il appartient d'établir une liaison entre ces deux mondes et de montrer par des actions et des faits de quoi nous sommes capables.

Ce chapitre traite de deux thématiques fondamentales, dont nous ne pourrions-nous affranchir :

- L’électronique (composants et fonctions).
- Le microcontrôleur (la carte Arduino).

Toutes les expériences sont réalisées avec une tension d’alimentation de5ou12v.

- **Interagir avec le monde réel**

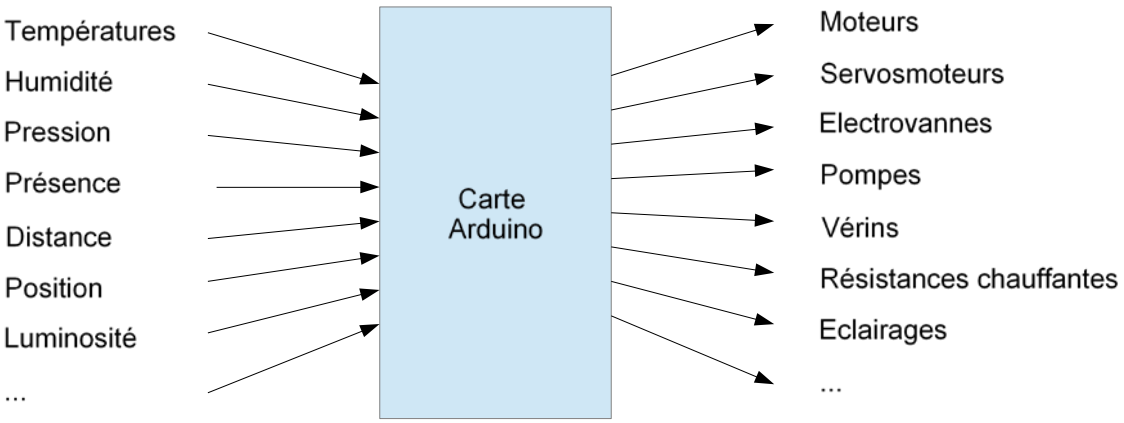


Figure II.1 : *Interagir de la carte Arduino avec le monde réel.*

II.2.1.2 Arduino : c'est pour qui ?

Le projet Arduino est né en hiver 2005. Massimo Banzi enseigne dans une école de Design à Ivrea en Italie, et souvent ses étudiants se plaignent de ne pas avoir accès à des solutions bas prix pour accomplir leurs projets de robotique. Banzi en discute avec David Cuartielles, un ingénieur Espagnol spécialisé sur les microcontrôleurs…

Ils décident de créer leur propre carte en embarquant dans leur histoire un des étudiants de Banzi, David Mellis qui sera chargé de créer le langage de programmation allant avec la carte. En deux jours David écrira le code, Trois jours de plus et la carte était créé…Ils décidèrent de l'appeler Arduino (un bar fréquenté par les élèves à proximité de l'école) …

Ça devient un hit tout de suite auprès des étudiants. Tout le monde arrive à en faire quelque chose très rapidement sans même avoir de connaissances particulières ni en électronique ni en informatique, réponse à des capteurs, faire clignoter des leds, contrôler des moteurs… Ils publient les schémas, investissent 3000 euros pour créer le premier lot de carte.

Les 50 premières partent directement à des élèves de l'école. En 2006 5000 cartes vendues…En 2007 plus de 30 000, en 2011 >120 000, sans compter les clones.

II.2.1.3 Arduino : qu'est-ce que c'est ?

La carte Arduino repose sur un circuit intégré (un mini-ordinateur appelé également microcontrôleur) associée à des entrées et sorties qui permettent à l'utilisateur de brancher différents types d'éléments externes. Les caractéristiques de la carte Arduino sont les suivantes :

- ➢ Une plate-forme de développement et de prototypage Open Source.
- ➢ Le rôle de la carte Arduino est de stocker un programme et de le faire fonctionner.
- ➢ Shields (cartes d'extension) avec des fonctions diverses qui s'enfichent sur la carte Arduino :
- ➢ Relais, commande de moteurs, lecteur carte SD, ...
- ➢ Ethernet, WIFI, GSM, GPS, …
- ➢ Afficheurs LCD, Écran TFT, ...
- ➢ IDE (Environnement de Développement Intégré) multi OS :
- ➢ Édition du programme

- Compilation du programme
- Transfert du programme dans la carte via le port USB

II.2.1.4 Arduino est-il un microcontrôleur ?

La réponse est oui, sans aucun doute. II possède bien tous les composants dont nous avons parlé et les réunit en son sein. Mais il cohabite aussi avec d'autres composants sur une carte compacte, dont nous allons parler dans ce chapitre.

- **C'est quoi un microcontrôleur ?**

Un microcontrôleur est un circuit intégré (ou IC, Integrated Circuit), qui rassemble sur une puce plusieurs éléments complexes dans un espace réduit. Au temps des pionniers de l'électronique, on soudait un grand nombre de composants encombrants, tels que les transistors, les résistances ou les condensateurs, sur des cartes plus ou moins grandes. Aujourd'hui, tout peut loger dans un petit boîtier en plastique noir muni d'un certain nombre de broches. Ces dernières sont les connexions du circuit intégré au moyen des quelles s'effectue la communication. La figure II.2 montre un microcontrôleur ATmega328, qu'on trouve sur la carte Arduino. Avec ses dimensions réduites, il dispose pourtant d'une grande puissance de calcul. En fait, il suffit de le souder sur une carte et de le mettre sous tension pour pouvoir l'utiliser. Certes, il manque encore quelques composants (par exemple, des stabilisateurs de tension, des connexions pour la programmation, et d'autres. Mais il est cependant sous cette forme déjà (presque) prêt à l'emploi **[10].**

Figure II.2 : *microcontrôleur ATmega328*

- **Sa sert à quoi ?**

À cela, les possibilités sont innombrables et dépendent uniquement de votre créativité.

Les microcontrôleurs jouent un rôle prépondérant dans les domaines suivants cette liste est loin d'être exhaustive et sert surtout à se faire une idée des diverses possibilités d'utilisation.

- Fonctions de surveillance dans des environnements critiques, par exemple dans des cages thoraciques (température, humidité, fréquence cardiaque, pression sanguine du prématuré…).
- Commande de chauffage : contrôle de la température externe ou interne pour le chauffage optimal de locaux.
- Stimulateurs cardiaques : surveillance de la fréquence cardiaque et, le cas échéant, stimulation du cœur.
- Appareils ménagers : par exemple, commande par programme enregistré dans des lave-linge ou lave-vaisselle modernes.
- Électronique de loisirs : lecteurs MP3, téléphones portables, appareils photo...
- Robotique : par exemple, commande de robots industriels pour le montage de pièces automobiles.

Cette liste peut ainsi se poursuivre à l'infini, mais nous pouvons d'ores et déjà remarquer une chose, les microcontrôleurs perçoivent des influences extérieures par le biais de capteurs, les traitent en interne à l'aide d'un programme, puis envolent des ordres de commande correspondants vers l'extérieur. Ils font donc preuve d'une certaine intelligence, qui dépend bien évidemment du programme mis en œuvre. Un microcontrôleur peut assurer des fonctions de mesure, de commande et de régulation.

Regardons maintenant de plus près le fonctionnement d'une boucle de régulation. Elle se compose d'un processus en boucle fermée comportant une perturbation. Un capteur transmet cette dernière au microcontrôleur qui réagit alors en fonction de son programme.

➢ Structure d'un microcontrôleur :

La structure générale d'un microcontrôleur et regardons les différents composants de la puce. Le microcontrôleur était déjà prêt à l'emploi. Mais Où se trouve son programme et où stocke-t-il ses données ? C’est un ordinateur complet sur un espace réduit au maximum, avec donc les éléments suivants

- Unité centrale (CPU)
- Mémoire de travail
- Mémoire de données
- Horloge interne

- Ports d'entrée et de sortie.

Un microcontrôleur se divise grossièrement en trois parties :

- Unité centrale (CPU),
- Mémoires (ROM et RAM)
- Ports d'entrée et de sortie.

L'horloge interne, ou l'oscillateur qui permet de piloter I 'unité centrale, a été laissée de côté pour le moment. Les éléments qui composent un microcontrôleur sont comparables aux périphériques d'un ordinateur. La différence réside dans le fait que les trois parties citées précédemment sont intégrées au microcontrôleur. Elles se trouvent toutes dans le même boîtier, ce qui est plus simple et plus compact.

Jetons maintenant un coup d'œil au schéma fonctionnel de notre microcontrôleur **[10] :**

Schéma d'un ***UC***

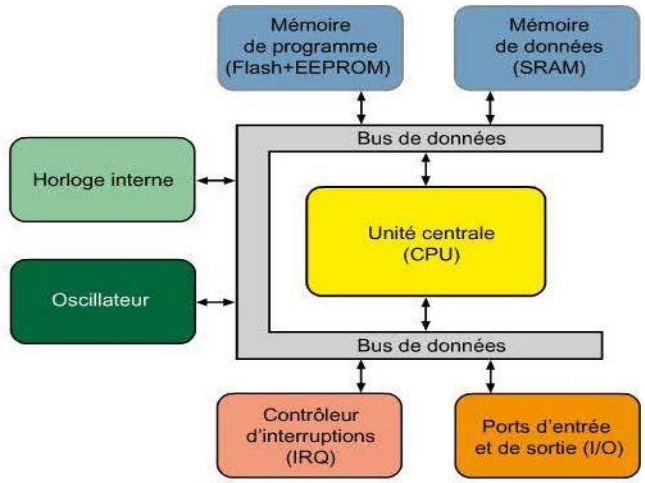


***Figure II.3** : schéma fonctionnel d'un microcontrôleur.*

- **Que signifient les différents blocs dans le schéma et quelle est leur fonction exacte ?**

- ✓ **L'unité centrale (CPU) :**

L'élément le plus important dans un microcontrôleur est l'unité centrale, appelée également CPU (Central Processing Unit). Sa fonction principale consiste à décoder et à exécuter des instructions. Elle peut adresser des mémoires, gérer des entrées ou sorties et réagir à des Interruptions (interrupts). Une Interruption (IRQ, ou Interrupt Request) est un signal qui demande au CPU d'interrompre un cycle de calcul en cours pour pouvoir réagir à un certain événement.

- ✓ **Le bus de données :**

Le bus de données sert à transporter les données d'un bloc à un autre. Par exemple, le CPU demande des données provenant de la mémoire, qui sont prises en charge par le bus et immédiatement mises à disposition pour traitement, Lorsque le résultat du calcul est disponible, il est à nouveau transféré sur le bus et transmis à un port de sortie qui, par exemple, pilote un moteur de robot pour atteindre un but précis. Cette structure de bus est une autoroute de données utilisable en commun par tous ceux qui sont desservis.

- ✓ **Les zones de mémoire :**

En principe, il existe deux types de mémoires d'un microcontrôleur :

- ➢ La mémoire de programme
- ➢ La mémoire de données.

La première accueille le programme que le CPU doit exploiter, alors que la seconde est utilisée pour gérer les résultats de calcul du moment. Il y a un problème quelque part. Quand j'éteins mon ordinateur, tous les programmes qui se trouvent dans la Mémoire s'effacent et je dois les recharger depuis mon disque dur pour pouvoir travailler avec. C'est vrai et c'est en cela que la mémoire de programme d'un microcontrôleur est particulière.

Un microcontrôleur n'a bien sûr pas de disque dur, mais il garde son programme en mémoire en I 'absence de tension d'alimentation externe. Un type de mémoire particulier est utilisé à cet effet, qu'on appelle mémoire flash. Comme son nom l'indique, c'est une mémoire non volatile, c'est-à-dire que bits et octets ne « s'envolent » pas en cas de coupure d'alimentation et restent disponibles. Vous avez déjà utilisé cette forme de mémoire des milliers de fois sur votre ordinateur.

Le BIOS est hébergé dans une mémoire flash de type EEPROM, et ses données peuvent être écrasées au besoin par l'insertion d'une nouvelle version. On dit alors que le BIOS est « reflashé » Le contraire se produit dans les mémoires de données dites SRAM. Ces dispositifs sont volatils et les données mémorisées sont perdues dès qu'il y a coupure d'alimentation. Mais rien de grave puisque ces dernières ne sont nécessaires que lorsque le programme est exécuté. Quand le microcontrôleur est sans courant, il n'a rien besoin de calculer. Mais la mémoire SRAM a sur la mémoire flash un avantage très important : elle offre un accès plus rapide.

✓ **Les ports d'entrées-sorties**

Les ports d'entrées-sorties sont les connexions qui relient le microcontrôleur au monde extérieur. Ils constituent une interface à laquelle la périphérie peut être connectée. On entend par périphérie tout ce qui peut être avantageusement raccordé à l'interface. Il peut s'agir, par exemple, des composants électroniques ou électriques suivants :

- LED (diode électroluminescente) ;
- Bouton-poussoir
- Commutateur
- LDR (Light DépendantRésistor ou photorésistance)
- Transistor
- Résistance
- Haut-parleur ou élément piézoélectrique
- etc.

En principe, les ports d'entrées-sorties sont soit numériques, soit analogiques

✓ **Le contrôleur d'interruption**

Un microcontrôleur est équipé d'un contrôleur dit d'interruption. Qu'est-ce que c'est et à quoi sort-il **[10] ?**

Un commutateur surveillant l'état d'une vanne est raccordé au port d'entrée numérique. Notre microcontrôleur pourrait ainsi être programmé pour interroger l'état du commutateur à intervalles brefs et réguliers. Cette interrogation cyclique appelée polling (ou interrogation) est, dans ce cas, plutôt inefficace car l'unité centrale est sollicitée pour rien. Une surveillance par interruption s'avère ici bien plus avantageuse. L'unité centrale suit le cours normal de son programme et ne réagit que si une interruption se produit. Le travail de fond s'interrompt un court instant et bascule dans une routine d'interruption (ISR, ou Interrupt Service Routine). Celle-ci contient des instructions qui indiquent l'action à effectuer. Après cela, on revient au travail de fond et à l'endroit précis où l'interruption s'est produite, comme si rien ne s'était passé.

II.2.1.5 Arduino : comment ça marche ?

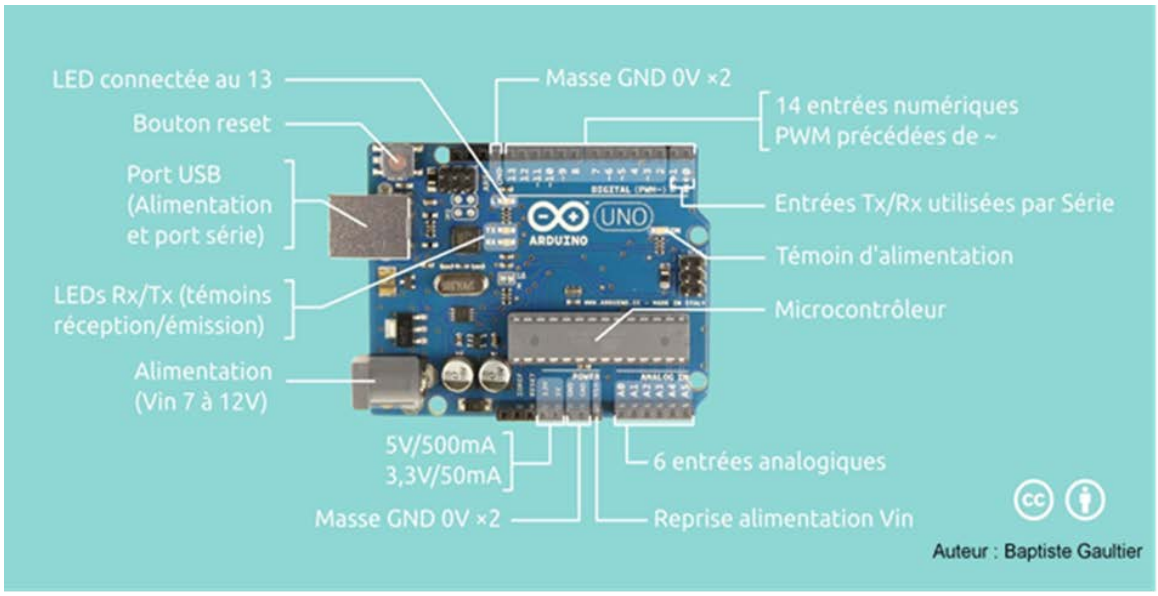


Figure II.4 : *les éléments de la carte Arduino.*

Ces éléments sont les plus importants de la carte Arduino mais, bien entendu, cela ne veut pas dire que les autres sont à négliger.

II.2.1.6 Arduino : faut-il des connaissances en électronique ?

- Pas ou peu si on utilise des cartes et des modules tout faits.
- La communauté francophone est très active sur le forum.
- Entraide, tutoriels, exemples de réalisations, ...
- Il faut des connaissances en électronique si on veut optimiser ou faire du sur-mesure.

II.2.2 Types des cartes Arduino

Il existe trois types des cartes :

a. Les « officielles » qui sont fabriquées en Italie par le fabricant officiel (Smart Projects).
b. Les « compatibles » qui n'est pas fabriqués par (Smart Projects), mais qui sont totalement compatibles avec les Arduino officielles.
c. Les « autres » fabriquées par diverse entreprise et commercialisées sous un nom différent (Freeduino, Seeduino, Femtoduino, ...).

II.2.3 Les différentes cartes Arduino

Les cartes Arduino doivent satisfaire des exigences diverses et variées. Certains utilisateurs souhaiteront effectuer du prototypage et tester de nouveaux montages ou idées, la carte Arduino Uno disposant d'assez d'entrées-sorties pour les projets d'envergure raisonnable. De par sa taille, la carte devrait aussi pouvoir être utilisée à l'avenir dans des projets plus ambitieux. D'autres auront besoin d'un grand nombre de ports afin de pouvoir raccorder de nombreux capteurs ou actionneurs. Une troisième catégorie d'utilisateurs ne cherchera qu'à attirer l'attention et à transmettre des signaux à leurs semblables à l'aide de diodes clignotantes. Ces exemples ne constituent qu'une partie des attentes que les makers auront vis-à-vis de leur carte Arduino. La forme, la taille ou les possibilités de connexion jouent un rôle décisif dans le choix de la carte adaptée. C'est pourquoi les développeurs d'Arduino ont mis au point un vaste choix de cartes à microcontrôleur afin que chacun trouve le modèle qui réponde à ses besoins.

II.2.4 Diverses Cartes Arduino

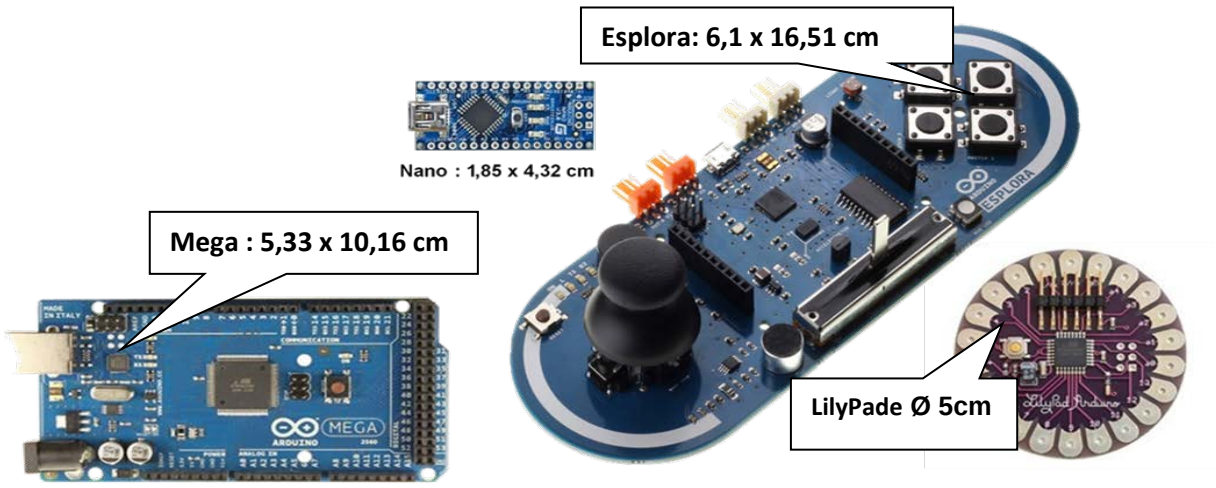


Figure II.5 : *différends cartes Arduino.*

II.2.5 Diverses Shields Arduino

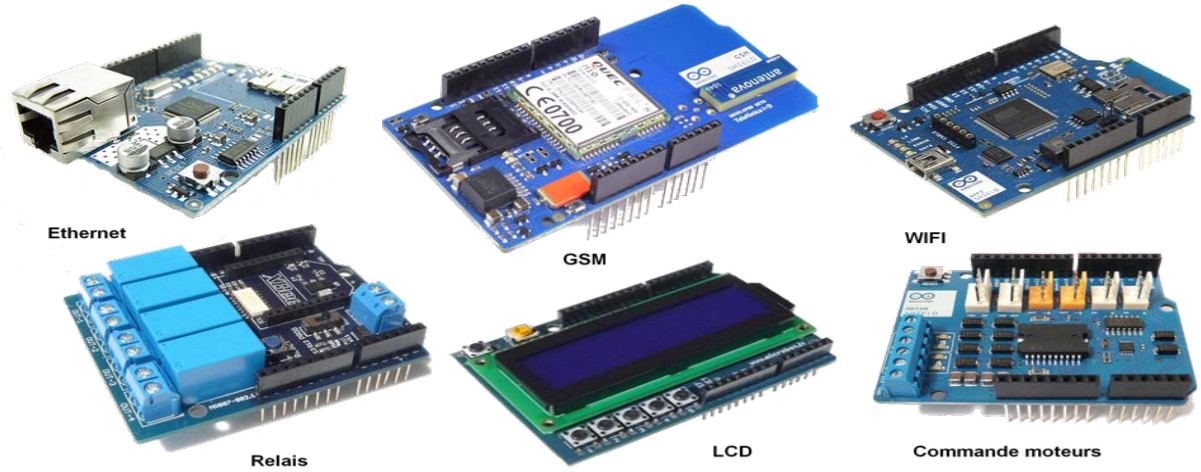


Figure II.6 : *différends Shields moteur.*

II.2.6 Carte Arduino + Shields

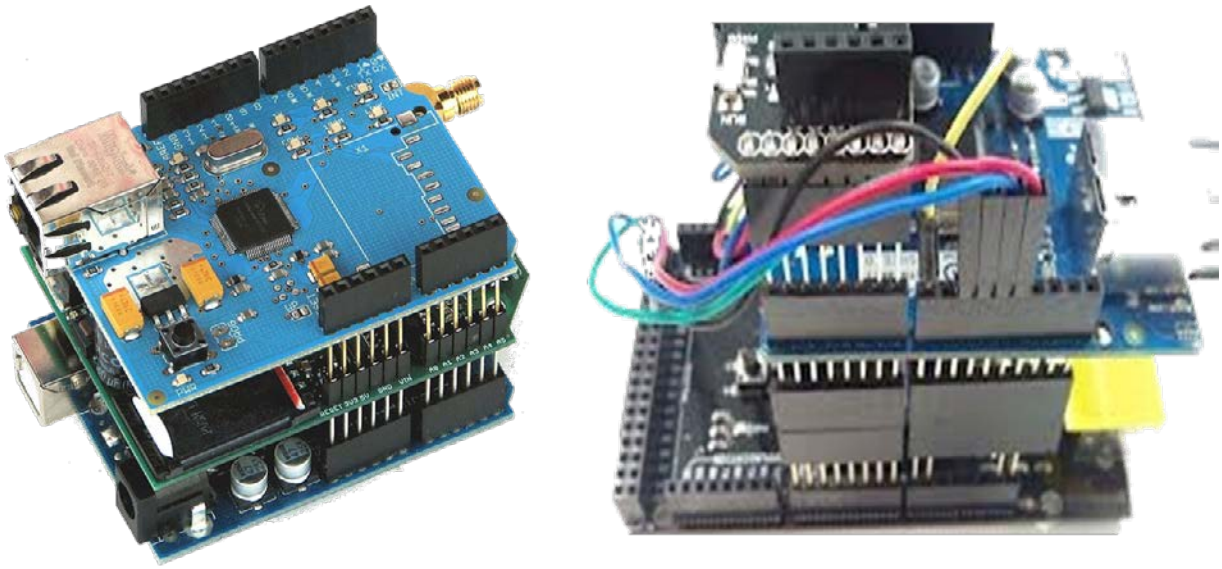


Figure II.7: *montage de la carte Arduino + Shields.*

Un environnement de développement intégré fonctionnant sur divers systèmes d'exploitation (Windows, Mac OS, Gnu/Linux*)* qui permet d'éditer le programme sur un ordinateur et de le transférer via le port USB.

II.2.7 les principales caractéristiques de la carte Arduino :

- Microcontrôleur ATmega
- Tension de service 5 V
- 14 entrées et sorties numériques (6 sorties commutables en MLI)
- 6 entrées analogiques (résolution 10 bits)
- 32 Ko de mémoire flash (0,5 Ko occupé par le chargeur d'amorçage ou bootlader)
- 2 Ko de SRAM
- 1 Ko d'EEPROM
- Fréquence d'horloge 16 MHz
- Interface USB

Comme vous pouvez le constater, un certain nombre d'entrées ou de sorties sont disponibles pour communiquer avec la carte Arduino. Elles constituent l'interface avec le monde extérieur et permettent d'échanger des données avec le microcontrôleur, Comme l'indique le schéma **[10]** :

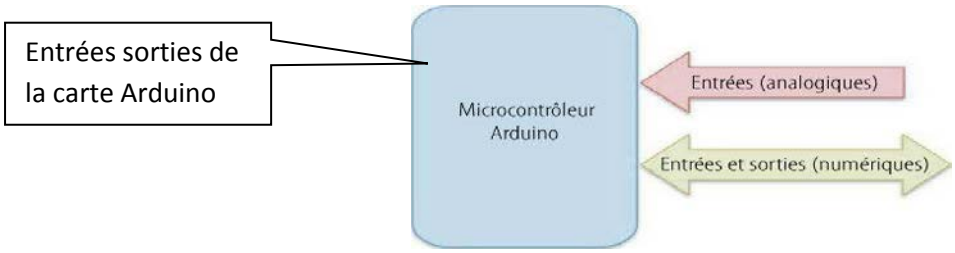


Figure II.8 : *communication de microcontrôleur Arduino.*

Le microcontrôleur Arduino, représenté en bleu à gauche, peut communiquer avec nous via certaines interfaces. Certains ports servent d'entrées (flèche rose), et d'autres d'entrées et de sorties (flèche verte). Un port est ici un chemin d'accès défini au microcontrôleur, pratiquement une porte vers l'intérieur qu'il est possible d'actionner.

Vous apercevez également des réglettes de raccordement noires sur ses bords supérieur et inférieur.

II.2.8 Programmation

L'environnement de programmation Arduino (IDE en anglais) est une application écrite en Java inspirée du langage Processing. L'IDE permet d'écrire, de modifier un programme et de le convertir en une série d'instructions compréhensibles pour la carte.

II.2.8.1 Structure d'un programme

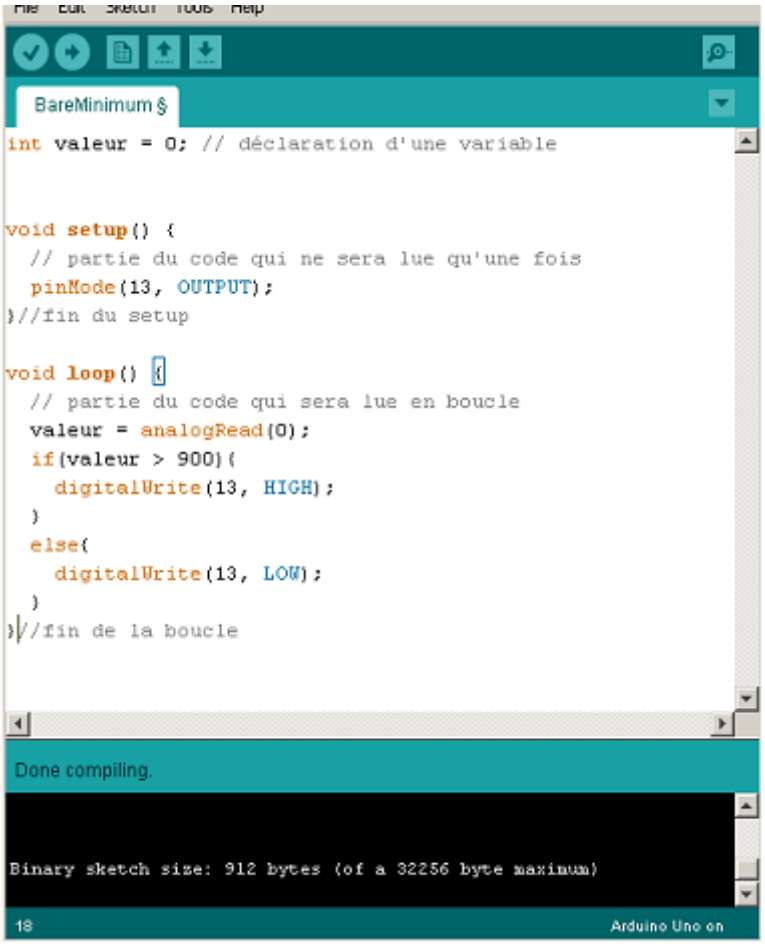


Figure II.9 : *Capteur d'écran d'un programme sur Arduino.*

Le programme est lu par le microcontrôleur de haut vers le bas. Une variable doit être déclarée avant d'être utilisée par une fonction. La structure minimale est constituée :

– en tête : déclaration des variables, des constantes, indication de l'utilisation de bibliothèques etc...

– un setup : (= initialisation) cette partie n'est lue qu'une seule fois, elle comprend les fonctions devant être réalisées au démarrage (utilisation des broches en entrées ou en sortie, mise en marche du midi, du port série de l'I2C etc.…)

– une loop (boucle) : cette partie est lue en boucle, C'est ici que les fonctions sont réalisées. En plus de cette structure minimale, on peut ajouter :

– des « sous-programmes » ou « routines » qui peuvent être appelées à tout moment dans la boucle, très pratiqué pour réaliser des morceaux de codes répétitifs.

– Des « callbacks », ce sont des fonctions qui sont rappelées automatiquement depuis une bibliothèque.

II.2.8.2 Références

référence :

Structure		
Fon Ces deux fonctions sont obligatoires dans tout programme en langage Arduino : • void setup() • void loop()	**Structures de contrôle** • if • if...else • for • switch case • while • do... while • break • continue • return • goto	**Syntaxe de base** • ; (point virgule) • {} (accolades) • // (commentaire sur une ligne) • /* */ (commentaire sur plusieurs lignes) • #define • #include
Opérateurs arithmétiques • = (égalité) • + (addition) • - (soustraction) • * (multiplication) • / (division) • % (modulo)	**Opérateurs de comparaison** • == (égal à) • != (différent de) • < (inférieur à) • > (supérieur à) • <= (inférieur ou égal à) • >= (supérieur ou égal à)	**Opérateurs booléens** • && (ET booléen) • || (OU booléen) • ! (NON booléen)
Pointeurs • * pointeur • & pointeur Voir également : • Manipulation des Ports	**Opérateurs bit à bit** • & (ET bit à bit) • |(OU bit à bit) • ^(OU EXCLUSIF bit à bit) • ~(NON bit à bit) • << (décalage à gauche) • >> (décalage à droite)	**Opérateurs composés** • ++ (incrémentation) • -- (décrémentation) (à revoir) • += (addition composée) • -= (soustraction composée) • *= (multiplication composée) • /= (division composée) • &= (ET bit à bit composé) • |= (OU bit à bit composé)

Tab II.1 : *Les différentes références.*

II.2.8.3 Variables et constantes

Variables et constantes

Les variables sont des expressions que vous pouvez utilisez dans les programmes pour stocker des valeurs, telles que la tension de sortie d'un capteur présente sur une broche analogique.

Constantes prédéfinies

Les constantes prédéfinies du langage Arduino sont des valeurs particulières ayant une signification spécifique.

- HIGH | LOW
- INPUT | OUTPUT
- true | false

A ajouter : constantes décimales prédéfinies

Expressions numériques

- Expressions numériques entières
- Expressions numériques à virgule

Types des données

Les variables peuvent être de type variés qui sont décrits ci-dessous.

Synthèse des types de données Arduino

- boolean
- char
- byte
- int
- unsigned int
- long
- unsigned long
- float (nombres à virgules)
- double (nombres à virgules)
- Les chaînes de caractères
- objet String NEW
- Les tableaux de variables
- le mot-clé void (fonctions)
- word
- PROGMEM

Voir également :

- Déclaration des variables

Pour info : les types de données avr-c

Conversion des types de données

- char()
- byte()
- int()
- long()
- float()
- word()

Portée des variables et qualificateurs

- Portée des variables
- static
- volatile
- const

Utilitaires

- sizeof() (opérateur sizeof)

Référence

- Code ASCII

Tab II.2 : *variables et constantes de programmation.*

II.2.8.4 Les fonctions

Fonctions

Entrées/Sorties Numériques
- pinMode(broche, mode)
- digitalWrite(broche, valeur)
- int digitalRead(broche)

Entrées analogiques
- int analogRead(broche)
- analogReference(type)

Sorties "analogiques" (génération d'impulsion)
- analogWrite(broche, valeur) - PWM

Entrées/Sorties Avancées
- tone()
- noTone()
- shiftOut(broche, BrocheHorloge, OrdreBit, valeur)
- unsigned long pulseIn(broche, valeur)

Communication
- Serial

Temps
- unsigned long millis()
- unsigned long micros()
- delay(ms)
- delayMicroseconds(us)

Math
- min(x, y)
- max(x, y)
- abs(x)
- constrain(x, a, b)
- map(valeur, toLow, fromHigh, toLow, toHigh)
- pow(base, exposant)
- sq(x)
- sqrt(x)

Pour davantage de fonctions mathématiques, voir aussi la librairie math.h : log, log10, asin, atan, acos, etc...

Nombres randomisés (hasard)
- randomSeed(seed)
- long random(max)
- long random(min, max)

Trigonométrie
- sin(rad)
- cos(rad)
- tan(rad)

Bits et Octets
- lowByte()
- highByte()
- bitRead()
- bitWrite()
- bitSet()
- bitClear()
- bit()

Interruptions Externes
- attachInterrupt(interruption, fonction, mode)
- detachInterrupt(interruption)

Interruptions
- interrupts()
- noInterrupts()

Voir également la librairie interrupt.h.

Tab II.3 : *Les différents fonctions.*

II.2.10 Les langages de programmation

Pour que la communication avec la carte Arduino se déroule sans problème, les développeurs doivent convenir d'une base de langage, afin qu'ils puissent se comprendre entre eux et exploiter un flux d'informations. C'est la même chose que lorsque vous allez à l'étranger et que vous ne maîtrisez pas la langue. Dans ce cas, vous devez vous adapter d'une façon ou d'une autre, peu importe la manière (gestes...).

Le microcontrôleur ne connaît à son niveau d'interprétation que le langage machine, appelé aussi code natif, composé exclusivement de valeurs numériques. Il est très difficile à

comprendre, car nous avons appris tout petit à échanger à l'aide de mots et de phrases, et non de valeurs numériques. Nous devons donc trouver un moyen de pouvoir communiquer de manière compréhensible avec le microcontrôleur.

C'est pourquoi un environnement de développement traduisant les commandes dans un langage dit évolué autrement dit, se présentant sous une forme semblable à notre langage a été créé. Pour autant, nous ne sommes pas plus avancés puisque le microcontrôleur ne comprend pas ce langage. En effet, il manque une sorte de traducteur servant de lien entre lui et l'environnement de développement. C'est le rôle du compilateur qui convertit un programme écrit en langage évolué en un langage cible compréhensible par le destinataire (ici, le CPU, ou Central Processing Unit, de notre microcontrôleur). Presque tous les langages de programmation font appel au vocabulaire anglais ; nous n'avons donc pas d'autre choix que de nous y mettre. Une autre étape de traduction sera donc nécessaire, mais je pense que l'anglais scolaire suffira ici dans la plupart des cas. Les instructions autrement dit, les ordres que l'environnement de développement comprend sont concis et semblables à celles du langage militaire, et représentent ce qu'il faut faire. **[11]**

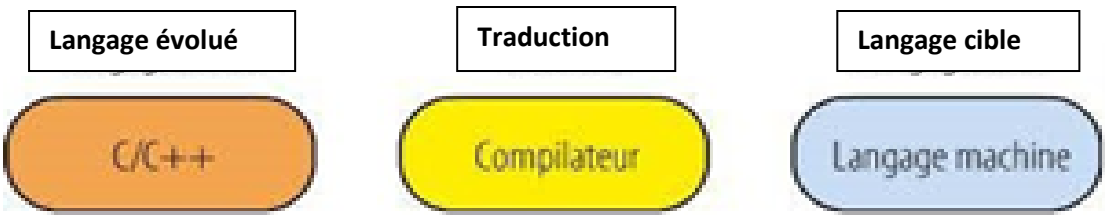


***Figure II.10** : langages de programmation.*

Il existe d'autres outils facilitant la programmation.

- **Ardublock (programmation en mode graphique) :**

  C'est un outil qui se greffe au logiciel Arduino. Il suffit de créer des blocs et de les paramétrer. Ce logiciel est vraiment un outil de qualité pour démarrer facilement sur Arduino, sans connaissances en programmation.

- **Scratch pour Arduino (programmation en mode graphique) :**

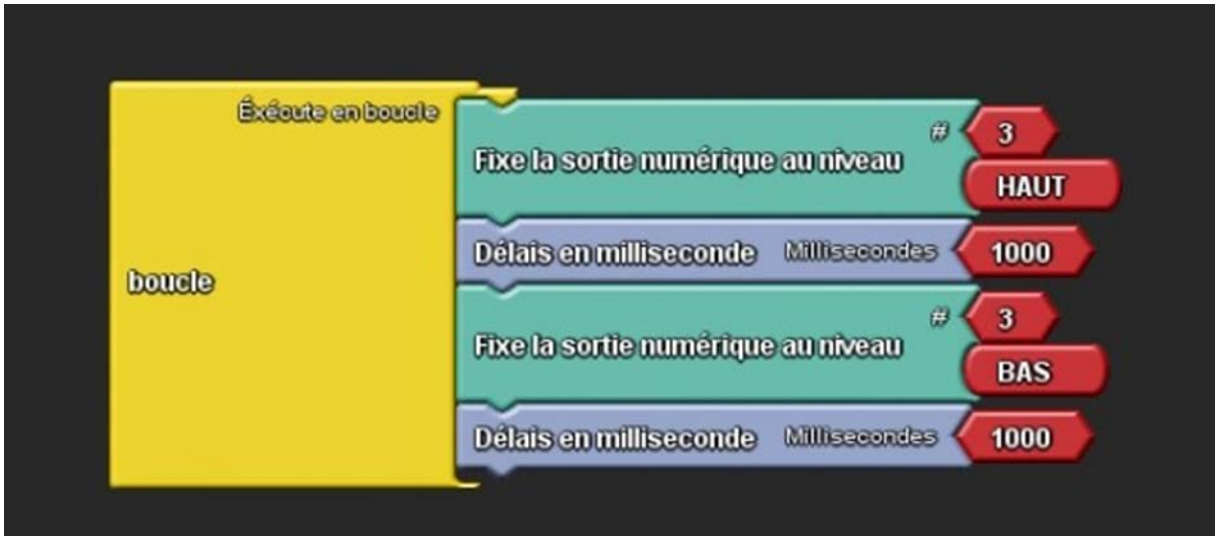


Figure II.11 : *programmation en mode graphique sur scratch.*

Permet de piloter un Arduino à partir du code SCRATCH et de ce fait rend accessible à tout public la programmation d'un robot à partir d'un environnement aussi ludique, visuel et intuitif que celui de SCRATCH.

- **Scratch pour Arduino**

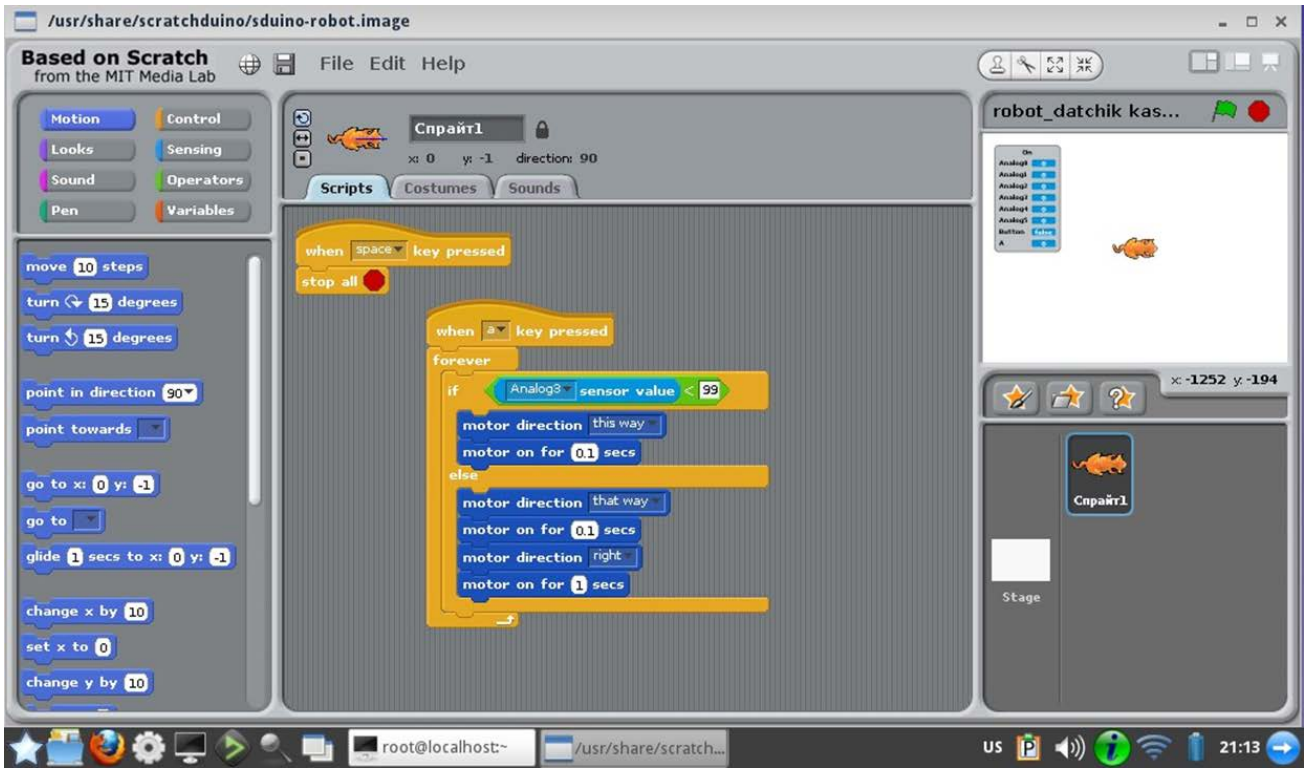


Figure II.12 : *schéma globale sur le scratch.*

II.3 Android

II.3.1 Historique

Android commence en octobre 2003, où la société Android Inc. est créée.
Officiellement, elle développe des logiciels pour mobiles. Mais en réalité, elle se préparait à sortir un tout nouveau système d'exploitation pour Smartphones. En 2005, Google rachète cette entreprise, et sort une première version béta en novembre 2007, avant de lancer la version 1.0 en septembre 2008 avec le HTC Dream. À partir de ce moment-là, le rythme des nouvelles sorties est très élevé : pas moins de 34 versions différentes sont apparues à nos jours**. [12]**

II.3.2 Définition

Android est une plateforme complète pour appareil mobile (téléphone, PDA, netbook, tablettes, etc.). La plateforme Android est un OS (Operating System) basée sur un kernel linux, sous licence open source. Elle est composée d'un système d'exploitation, de librairies (middleware), et d'un ensemble d'applications : un client mail, un navigateur, un calendrier, etc.

Les services offerts par Android facilitent notamment l'exploitation des réseaux de télécommunications GSM, Bluetooth, WIFI et UMTS, la manipulation des médias, notamment de la vidéo, de l'audio et des images JPEG ainsi que d'autres formats, l'exploitation des senseurs tels que les capteurs de mouvements, la caméra, la boussole et le récepteur GPS, l'utilisation de l'écran tactile, le stockage en base de données, l'affichage de pages web, l'exécution multitâche des applications et l'envoi de messages.

***Figure II.13** : Logo Android.*

II.3.3 Les versions d'Android

A ce jour, Android est disponible en version 6.0, (Marshmallow). Les versions se succèdent rapidement et les changements qui les accompagnent sont souvent conséquents en termes de nouvelles fonctionnalités et d'améliorations.

Les différentes versions ont toutes des noms de desserts « en anglais » depuis la sortie de la version 1.5 et suivent une logique alphabétique (de A vers Z).

Version	Nom de la version	Date de sortie	Évolutions
6.0	Marshmallow	09/10/2015	Autonomie en veille, Animation de démarrage, système UI Tuner, gestion des autorisation, AndroidPay…
5.0	Lollipop	23/11/2014	Moteur d'exécution ART, support de 64 bits, Android TV, économiseur de batterie...
4.4	KitKat	31/10/2013	Interface translucide, Framework pour imprimer, Framework pour la gestion des fichiers.
4.1	Jelly Bean	09/07/2012	Assistance vocale, accessibilité : mode gestuel 'Braille', WIFI-Direct service discovery, vsync timing
4.0	IceCream Sandwich	19/10/2011	WI-FI direct, Bluetooth Health Devie profile, Control over network data, Grid Layout.
3.2	Honeycomb	22/02/2011	Support des processeurs Qualcomm, Support des tablettes tactiles de 7 pouces
2.3	Gingerbread	06/12/2010	Support de la VoIP et SIP. Gestionnaire de téléchargement, support de plusieurs cameras.
2.2	Froyo	20/05/2010	Implémentation de JIT, partage de connexion USB.
2.0	Eclair	26/10/2009	Bluetooth, support de plus de taille d'écran.
1.6	Donut	15/09/2009	Google navigation (GPS gratuit)

Tab II.4 : *Les différentes versions d'Android.*

II.3.4 Architecture d'Android :

L'environnement de développement est basé sur une architecture autour du noyau Linux. La plateforme Android est composée de cinq couches principales, Ce sont le noyau Linux, les bibliothèques et le moteur d'exécution Android, le cadre de l'application et la couche application.

La figure présente les différentes couches de la pile logicielle d'Android : **[13]**

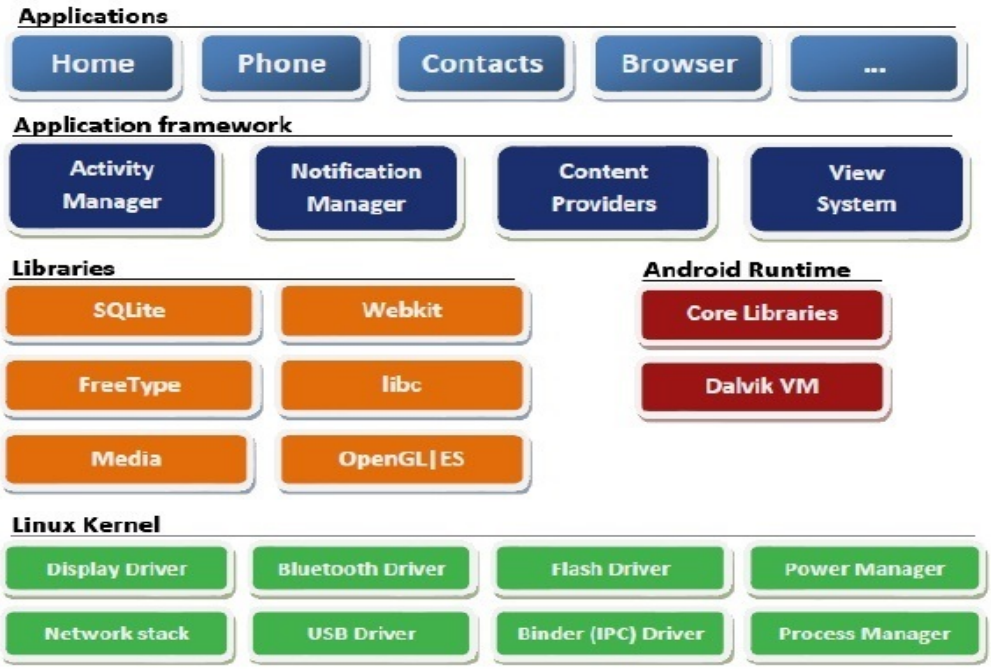


Figure II.14 : *Architecture d'Android.*

Android bénéficie d'une architecture en couche complète faisant de lui une plateforme riche, dédiée aux appareils mobiles.

Il est à base du noyau linux qui permet de faire le pont entre la partie matérielle et la partie Logicielle et profitant des services système de base tels que la sécurité, la gestion mémoire, Gestion de processus, etc. A un niveau supérieur se trouvent un ensemble de librairies écrites en C/C++ jouant le rôle d'un middleware (on en cite le système de bibliothèque C, les médiathèques, le SGL, etc.). C'est sur cette couche que se greffe l'Android Runtime, Comprenant la machine virtuelle java et ses bibliothèques. Vient ensuite la plateforme logicielle, nommée aussi Framework de développement, écrite en java et permettant de mutualiser les ressources entre les applications Java. Elle offre aux développeurs la possibilité de produire des applications diverses et innovantes à travers un ensemble d'API.

Enfin, et à un niveau plus supérieur se situe un ensemble d'applications sous forme de paquets apk. Les applications fournies par Android sont telles qu'un navigateur web, un client mail, un calendrier, un gestionnaire de contacts, etc.

II.3.5 Le moteur d'exécution d'Android

A partir de la version 5.0 sortie en 2014, l'environnement d'exécution Android RunTime (ART) remplace la machine virtuelle Dalvik. Cet environnement d'exécution plus performant a été développé par Google pour pallier le potentiel limité de Dalvik, créé en 2007. Avec ART, contrairement à la machine virtuelle java, les fichiers. Apk ne sont plus lancés directement, mais décompressés et lancés avec de nouvelles bibliothèques et API ; les applications prennent ainsi plus de place (+20 %), mais les gains en performance et en autonomie des batteries sont conséquents.

II.3.6 Application Android

Une application Android est une collection de composants liés via un fichier de configuration, il y'a quatre types de composants :

- Les activités.
- Les vues et contrôles.
- Les ressources.
- Le fichier de configuration appelé également manifeste.

➢ **Les activités** :

Il s'agit d'une partie de l'application présentant une vue à l'utilisateur Une application peut avoir une ou plusieurs activités (par exemple pour une application de messagerie on pourrait avoir une Activity pour la liste des contacts et une autre pour l'éditeur de texte).

➢ **Les vues et contrôles** :

Les vues sont les éléments de l'interface graphique que l'utilisateur voit et sur lesquels) il pourra agir. Les vues contiennent des composants organisés selon diverses mises en pages. Quant aux contrôles (boutons, champs de saisie, case à cocher…) sont eux-mêmes un sous ensemble des vues. Ils ont besoin d'accéder aux textes et aux images qu'ils affichent, ces textes et ces fichiers seront puisés dans les fichiers ressources de l'application.

- **Le fichier de configuration :**

Il est indispensable à chaque application qui décrit entre autres :

- Le point d'entrée de notre application (quel code doit être exécuté au démarrage de l'application).
- Quels composants constituent ce programme.
- Les permissions nécessaires à l'exécution du programme (accès à internet, accès à l'appareil photo, accès au BLUETOOTH pour notre cas).

II.3.7 Conclusion

Dans ce chapitre, notre étude s'est focalisée sur la présentation de la carte (Arduino), et on a commencé par un bref historique et quelques détails concernant la création et l'utilité de cette carte. Puis on a parlé de son côté matériel, enfin on a présenté le logiciel et le langage de programmation qui est devenu un outil de commande et de programmation de cette dernière.

La compréhension de données exposées dans ce chapitre nous offre des bases de connaissances pour l'usage de la carte d'acquisition ce qui nous donne la possibilité d'allier les performances de la programmation à celles de l'électronique en vue de commander notre robot. Dans le chapitre suivant nous allons faire la conception de notre système.

III.1 Introduction

Ce présent chapitre a pour objectif de présenter la réalisation de notre robot commandé avec la voix en utilisant un Smartphone Android via le Bluetooth. Nous allons donner un schéma synoptique global pour expliquer le fonctionnement du système.

❖ **Schéma synoptique :**

Chaque partie du travail est simulée avant d'être mise en application.

III.2 Environnement matériel

Pour réaliser notre projet, on a utilisé le matériel suivant :

- Carte Arduino atmega2560
- Shield moteur
- Bluetooth **HC-05**
- Servo moteur
- Capteur ultrason **HC-SR04**
- Des câbles métalliques
- Un Smartphone
- Robot mobile 2 moteurs
- Deux batteries rechargeables

III.2.1 Carte Arduino Atmega2560

III.2.1.1 Présentation d'Arduino Méga 2560

La carte Arduino repose sur un circuit intégré (un mini-ordinateur appelé également microcontrôleur) associé à des entrées et sorties qui permettent à l'utilisateur de brancher différents types d'éléments externes **[14]** :

- ✓ **Côtés entrés**, des capteurs qui collectent des informations sur leur environnement comme la variation de température via une sonde thermique, le mouvement via un détecteur de présence ou un accéléromètre, le contact via un bouton-poussoir, etc.
- ✓ **Côtés sortis**, des actionneurs qui agissent sur le monde physique telle une petite lampe qui produit de la lumière, un moteur qui actionne un bras articulé, etc. Comme le logiciel Arduino, le circuit électronique de cette plaquette est libre et ses plans sont disponibles sur internet. On peut donc les étudier et créer des dérivés.

Plusieurs constructeurs proposent ainsi différents modèles de circuits électroniques programmables et utilisables avec le logiciel Arduino. Il existe plusieurs variétés de cartes Arduino.

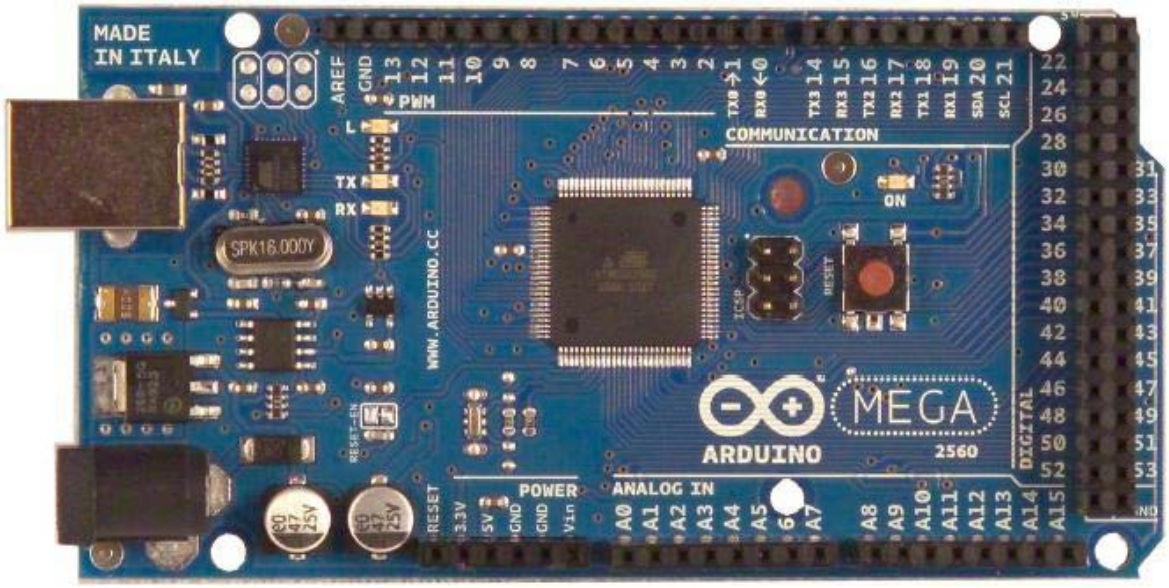


Figure III.1 : *Arduino Méga 2560.*

III.2.1.2 Caractéristiques de l'Arduino Méga 2560

Cette carte dispose :

- De 54 broches numériques d'entrées/sorties (dont 14 peuvent être utilisées en sorties).
- PWM (largeur d'impulsion modulée).
- De 16 entrées analogiques (qui peuvent également être utilisées en broches entrées/sorties numériques).
- De 4 UART (port série matériel).
- D'un quartz 16Mhz.
- D'une connexion USB.
- D'un connecteur d'alimentation jack.
- D'un connecteur ICSP (programmation "in-circuit").
- Et d'un bouton de réinitialisation (reset).

Elle contient tout ce qui est nécessaire pour le bon fonctionnement du microcontrôleur. Ainsi, pour pouvoir l'utiliser et se lancer, il suffit simplement de la connecter à un ordinateur à l'aide d'un câble USB (ou de l'alimenter avec un adaptateur secteur ou une pile, mais ceci n'est pas indispensable, l'alimentation étant fournie par le port USB**).**

III.2.2 Shield moteur

Est un moteur facile à utiliser pour l'Arduino, capable de piloter différents moteurs.

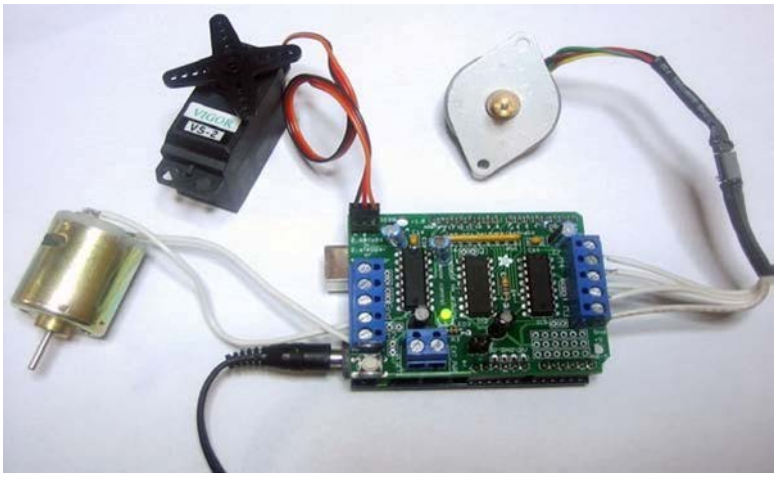


Figure III.2 : *montage d'un Shield moteur avec différents moteurs.*

De gauche à droite : moteur à courant continu, servomoteur, moteur pas à pas.

III.2.3 L293D

III.2.3.1 Comprendre le composant L293D

Le composant L293D est un pont de puissance composé de plusieurs transistors et relais qui permet d'activer la rotation d'un moteur. Ce composant se trouve facilement sur et ne coûte pas très cher.

Le L293D est un double pont-H, ce qui signifie qu'il est possible de l'utiliser pour commander quatre moteurs distincts (dans un seul sens) grâce à ses 4 canaux.

Il est également possible de constituer deux pont-H afin de piloter deux moteurs distincts, dans les deux sens et indépendamment l'un de l'autre.

Il est important de noter que le L293D peut délivrer au maximum 600mA, ce que le choix des moteurs doit se faire en conséquence.

III.2.3.2 Caractéristiques techniques du L293D

Voici les caractéristiques techniques du composant L293D :

- Nombre de pont-H : 2
- Courant Max Régime continu : 600mA (x2)
- Courant de pointe Max < 2ms : 1200mA
- VS Max Alim moteur : 36v
- VSS Max Alim logique : 7v
- Nombre de Broches : 16 DIP
- Perte de tension : 1.3v

III.2.3.3 Branchement du L293D

Le schéma suivant détaille les différentes broches du composant L293D :

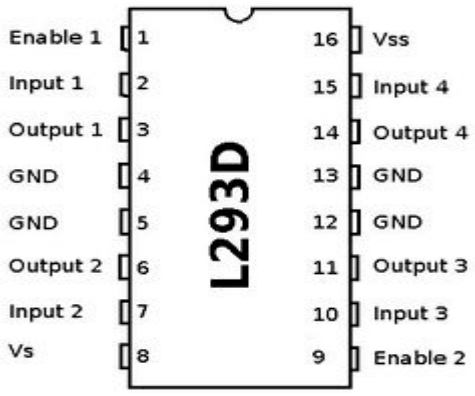


Figure III.3 : *différentes broches du composant L293D.*

1. Enable1 : permet d'envoyer (ou pas) la tension sur les sorties du moteur via OUTPUT1et OUTPUT2 et commande l'activation/désactivation du premier Pont-H. Si
ENABLE1 = GND, le pont-H est déconnecté et le moteur ne fonctionne pas. Par contre siENABLE1 = VSS, le pont-H est connecté aux sorties et le moteur fonctionne dans un sens ou l'autre ou pas en fonction des tensions appliquée sur INPUT1 & INPUT2.

2. Input1 : avec Input 2, sont les broches de commande du Pont-H Output1/Output2. Il sera directement branché à votre Arduino pour commander le sens du courant entre Output 1 et Output 2.

3. Ouput1 : avec Output 2, sera branché directement sur le moteur.

4. GND : qui doit être raccordé à la masse de la source d'alimentation de puissance VS et à la masse de la source d'alimentation de la logique "VSS" (donc GND Arduino).

5. GND.

6. Ouput2 : avec Output 1, sera branché directement sur le moteur.

7. Input2 : avec Input 1, sont les broches de commande du Pont-H Output1/Output2. Il sera directement branché à votre Arduino pour commander le sens du courant entre Output 3 et Output 4.

8. VS : Alimentation de puissance des moteurs.

9. Enable2 : commande l'activation du second pont-H constitué d'Output3/Output4.

10. Input3 : avec Input 4, sont les broches de commande du Pont-H Output3/Output4. Il sera directement branché à votre Arduino pour commander le sens du courant entre Output 3 et Output 4.

11. Ouput3 : avec Output 4, sera branché directement sur le moteur.

12. GND

13. GND

14. Ouput4 : avec Output 3, sera branché directement sur le moteur.

15. Input4 : avec Input 3, sont les broches de commande du Pont-H Output3/Output4. Il sera directement branché à votre Arduino pour commander le sens du courant entre Output 3 et Output 4.

16. VSS : Alimentation de la logique de commande (5V). A raccorder à la borne +5V de votre Arduino.

Veuillez noter que les pins Enable1 et Enable2 permettent moduler la vitesse du moteur en utilisant des broches PWM de votre Arduino. Si vous ne souhaitez pas moduler la vitesse du moteur DC, il vous suffit de brancher Enable1 sur la broche VSS de votre Arduino.

III.2.3.4 Fonctionnement du L293D

Le tableau suivant vous permettra de faire fonctionner votre moteur DC en utilisant le composant L293D :

Enable 1	Input 1	Input 2	Fonction
High	Low	High	Tourne dans le sens horlogique
High	High	Low	tourne dans le sens anti-horlogique
High	Low	Low	Stop
High	High	High	Stop
Low	Non applicable	Non applicable	Stop

***Tab III.1** : fonctionnement du moteur DC en utilisant L293D.*

III.2.3.5 Utilisation du L293D avec Arduino et 2 moteurs DC

Avec un seul pont L293D et un Arduino on va être capable de piloter 2 moteurs à courant continu indépendamment l'un de l'autre. Si la puissance de vos moteurs est faible vous pourrez même utiliser le 5V en sortie de votre Arduino pour alimenter vos moteurs DC.

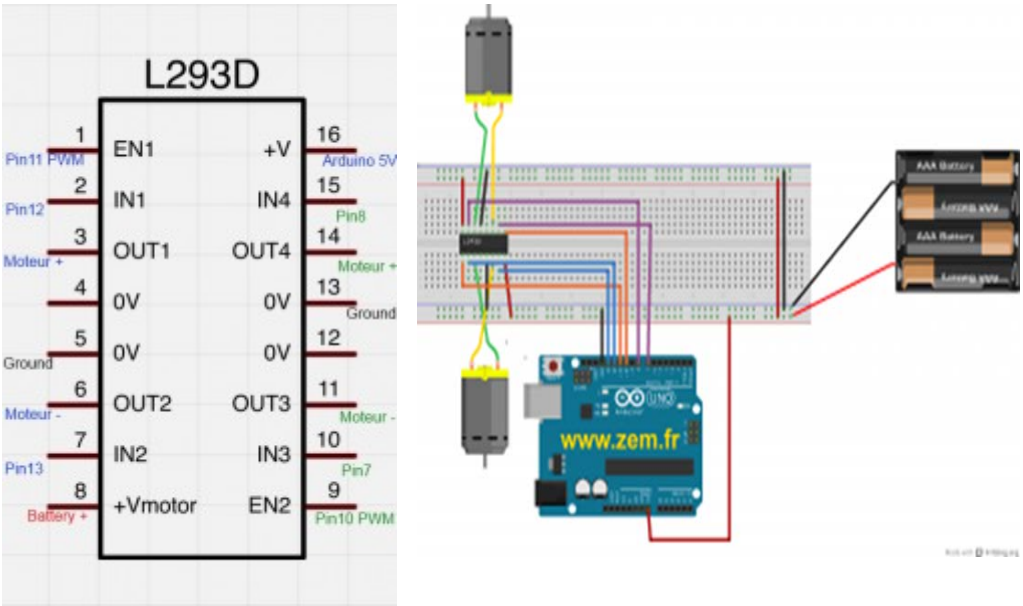


***Figure III.4:** Brochages du L293D avec 2 moteurs sous Arduino.*

Il existe plusieurs puces de circuits intégrés qui sont peu coûteuses et faciles à intégrer dans un circuit. Le double contrôleur de moteur commun L293D est une puce IC 16 DIP qui

contient deux circuits d'attaque protégés capables de fournir jusqu'à 600 mA de courant continu à chaque moteur jusqu'à 36 VCC. Le L298N est une puce similaire capable de délivrer 2 ampères à chaque moteur. Ces puces (et autres) acceptent les signaux d'entrée standard 0-5V et disposent de portes logiques internes pour éviter une surcharge accidentelle et la commande du contrôleur en état destructeur.

Il y a quatre séries d'entrées et de sorties, intitulées 1A (entrée) et 1Y (sortie) à 4A et 4y. L'état numérique de ces broches d'entrée détermine l'état numérique de leur broche de sortie amplifiée correspondante. En pratique, lorsque vous appliquez un signal 5V (VCC1) à la broche 1A, vous obtenez un signal 12v (VCC2) sur la broche 1Y. Il y a aussi une broche "enable" pour chaque ensemble d'entrées / sorties. La broche 1-2 Enable contrôle l'état des broches de sortie 1Y et 2Y simultanément, et la broche d'activation 3-4 contrôle les sorties 3Y et 4Y. Vous pouvez utiliser des broches numériques sur l'Arduino pour contrôler les quatre broches d'entrée et définir la direction du moteur, tout en utilisant un signal PWM sur chaque broche d'activation pour définir la vitesse de chaque moteur.

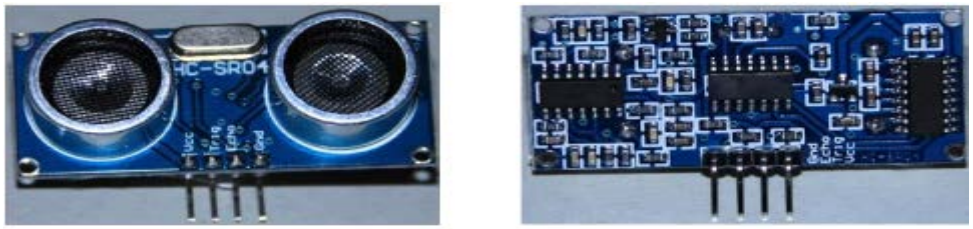
III.2.3.6 Programme de Controller des 2 Motors

```
int motor1_enablePin =11;//pwm

int motor1_in1Pin =13;

int motor1_in2Pin =12;

int motor2_enablePin =10;//pwm

int motor2_in1Pin =8;

int motor2_in2Pin =7;

Voidsetup ()

{

//on initialise les pins du moteur 1

pinMode (motor1_in1Pin, OUTPUT);

pinMode (motor1_in2Pin, OUTPUT);

pinMode (motor1_enablePin, OUTPUT);
```

```
  //on initialize les pins du moteur 2
pinMode (motor2_in1Pin, OUTPUT);
pinMode (motor2_in2Pin, OUTPUT);
pinMode (motor2_enablePin, OUTPUT);
}
 Voidloop ()
{
SetMotor2(175, true);
SetMotor1(255, false);
}
//Function qui set le moteur1
void SetMotor1(intspeed, Boolean reverse)
{
analogWrite (motor1_enablePin, speed);
digitalWrite (motor1_in1Pin, reverse);
digitalWrite (motor1_in2Pin, reverse);
}
//Fonction qui set le moteur2
void SetMotor2(intspeed, Boolean reverse)
{
analogWrite (motor2_enablePin, speed);
digitalWrite (motor2_in1Pin, reverse);
digitalWrite (motor2_in2Pin, reverse);
}
```

III.2.4 Le capteur à ultrason HC-SR04

Le capteur HC-SR04 utilise les ultrasons pour déterminer la distance d'un objet. Il offre une excellente plage de détection sans contact, avec des mesures de haute précision et stables. Son fonctionnement n'est pas influencé par la lumière du soleil ou des matériaux sombres, bien que des matériaux comme les vêtements puissent être difficiles à détecter.

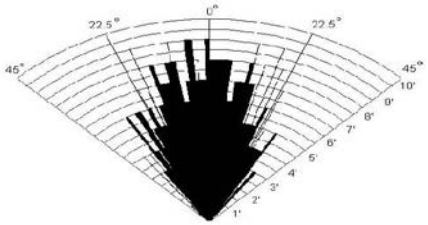


Le capteur de face le capteur de dos

Figure III.5 : *Ultrason HC-SR04.*

III.2.4.1 Caractéristique du HC-SR04

- Plage de mesure : 2 cm à 400 cm.
- Résolution de la mesure : 0.3 cm.
- Angle de mesure efficace : -7.5° à +7.5 °.
- Signal de détection : impulsion de 10us TTL.
- Signal d'écho : sortie TTL PWL.
- Dimensions : 45 mm x 20 mm x 15 mm.

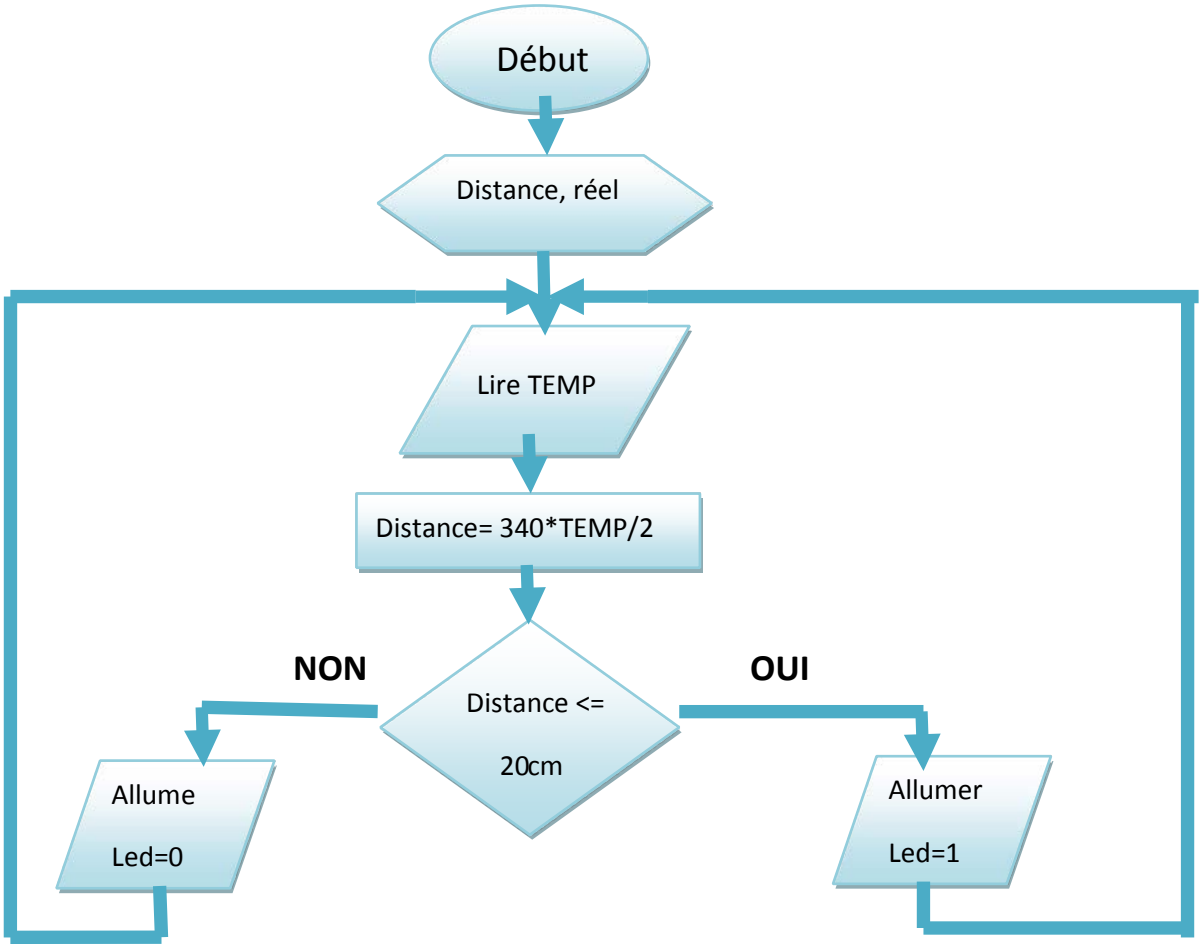


Practical test of performance,
Best in 30 degree angle

III.2.4.2 Configuration des broches du HC-SR04

- Vcc = Alimentation +5 V CC.
- Trig = Entrée de déclenchement de la mesure (Trigger input).
- Echo = Sortie de mesure donnée en écho (Echo output).
- GND = Masse de l’alimentation.

III.2.4.3 Organigramme :



Détection d'obstacle

III.2.5 Câbles métalliques

Pour brancher les différents capteurs vers le microcontrôleur on s'est servis de différent câbles, très pratiques et faciles à brancher.

***Figure III.6**: Câbles métalliques.*

III.2.6 Châssis rover 4WD

Pour notre projet nous avons choisis le Châssis 4WD 4 roues, c'est un châssis très léger facile à assembler et à manipuler, il comporte :

- 4 roues en forme de pneu de voiture avec des trous pour encodeur éventuel.
- 4 Moteurs CC (courant continu) de 6V.
- 2 plaques avec trous de montage.
- Vis nécessaires pour le montage.

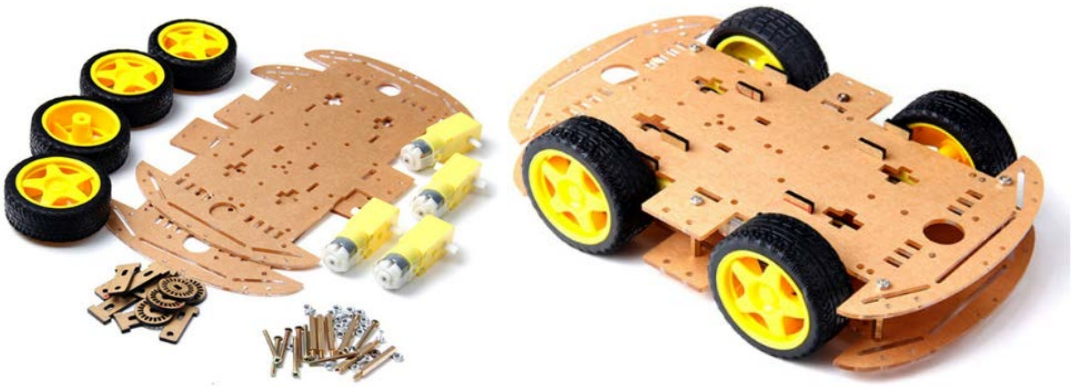


Figure III.7: *Chassis rover 4WD.*

III.2.7 Smartphone SAMSUNG Galaxy S5

Ce Smartphone est utilisé pour le test de l'application (interface) développée pour commander le Rover.

Ces caractéristiques techniques :

Système : Android 5.0

Fréquence processeur : 2.5 GHz

Taille (diagonal) (pousse) : 5.10

Figure III.8*: SAMSUNG Galaxy S5.*

III.2.8 Le module Arduino Bluetooth

Le Bluetooth est un protocole de communication sans fil. Il a vu le jour à la fin des années 1990 et n'a vraiment percé que dans les années 2000.

Ce protocole est un cousin du Wifi. En effet, ils utilisent la même gamme de fréquences : 2.4 GHz (tout comme les téléphones portables et le Zigbee par exemple). C'est une communication bidirectionnelle, deux modules peuvent communiquer ensemble en même temps. Le comportement utilisé est « maître/esclave » **[15].**

Un esclave pourra parler avec un seul maître, mais un maître pourra dialoguer avec plusieurs esclaves Pour son utilisation, elle se passe en plusieurs étapes :

1. Le maître se met en mode « reconnaissable »
2. L'esclave trouve le maître et demande à s'y connecter
3. Le maître accepte la connexion
4. Les périphériques sont alors appariés (ou associés)
5. la communication peut commencer

III.2.8.1 Présentation du module Bluetooth HC-05 :

HC-05 et est trouvable assez facilement pour quelques euros (via des sites d'import de Chine). Il est aussi gros que le pouce et est en fait un montage d'un module Bluetooth sur un petit PCB (printed circuit board). Cela permet de s'affranchir de certaines contraintes comme la soudure du module (qui est très délicate), la conversion 5V -> 3.3V, la régulation de l'alimentation (3.3V de nouveau) ou encore l'ajout de LED de signal. Tout cela est déjà intégré **[16].**

Alors que trouvons-nous sur ce petit module ?

Tout d'abord, un ensemble de broches. VCC et GND pour l'alimentation (5V), Rx/Tx pour la communication. On y voit aussi une broche « Key » (ou bien Enable « EN ») qui servira à envoyer des commandes de configuration au module. La dernière broche nommée « Led » (ou bien state) permet de brancher une LED pour obtenir un signal sur l'état du module.

Led sert à connecter une LED de statut, vous pouvez la laisser déconnectée cela n'influencera pas le comportement du module. Key sert à utiliser le mode « commande » du module. Avec cette dernière vous pourrez reconfigurer la voie série (vitesse, parité etc…) et d'autres options liées au Bluetooth (nom du module, mot de passe d'appairage, mode esclave/maître…). Cette broche est à connecter à n'importe quelle sortie numérique de l'Arduino **[17].**

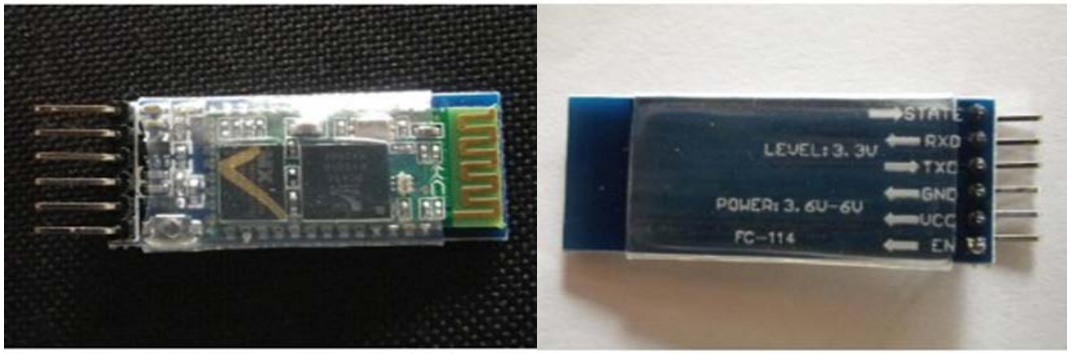


Figure III.9: *module Bluetooth HC-05.*

III.2.9 Batteries

Pour l'alimentation du Rover nous avons utilisé deux batteries rechargeables :

- Une batterie d'une capacité de 5000 mAh, sortie (5v) pour alimenter l'ensemble de l'électronique.
- Une batterie de 9,6v pour alimenter les Moteurs.

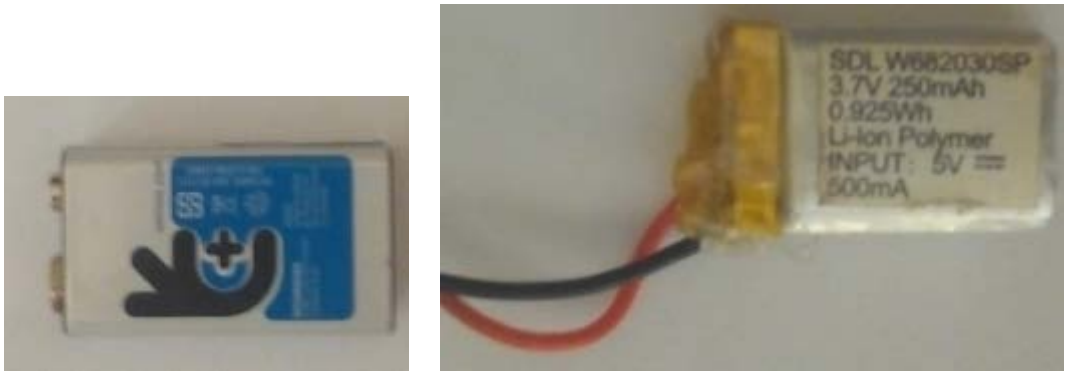


Figure III.10 : *Batteries d'alimentation du système.*

III.2.10 Servo-moteur :

Un servomoteur est un type spécial de moteur à courant continu qui utilise un codeur pour déterminer la position de l'arbre de sortie. Les servomoteurs Hobby se composent d'un petit moteur à courant continu, d'un réducteur de vitesse, d'un codeur d'arbre de potentiomètre et d'un circuit de commande du moteur, ce qui facilite leur interface directe avec l'Arduino. Le circuit de commande du moteur n'est pas seulement utilisé pour décoder le signal d'entrée (impulsion asservie R / C), mais également pour entraîner le moteur CC. Ces moteurs peuvent se déplacer rapidement vers une position spécifique et ne nécessitent que trois fils pour s'interfacer avec l'Arduino (signal, alimentation et terre). Le signal utilisé pour faire fonctionner le servo est une impulsion électrique synchronisée avec précision qui varie de 1 milliseconde à 2 millisecondes - où une impulsion de 1,5 ms donne la position centrale du servomoteur. Le servomoteur recherche une mise à jour avec une nouvelle impulsion environ 50 fois par seconde ou toutes les 20 millisecondes.

***Figure III.11** : servomoteur.*

Une vue démontée du plus gros servomoteur qui est à droite indique les engrenages, le potentiomètre, le circuit imprimé et le moteur à courant continu qui sont normalement emballés dans le boîtier en plastique.

III.2.11 Moteurs à engrenages

La puissance d'un moteur à une énergie donnée peut être conçue pour avoir une vitesse élevée ou un couple élevé. Pensez à un vélo de montagne à 18 vitesses pour voir comment cela fonctionne, si vous mettez le vélo en première vitesse, votre pédalage fournira un couple élevé qui vous permettra de monter facilement sur une côte escarpée au détriment de la lenteur. Si vous mettez la moto en dix-huitième vitesse, il sera presque impossible de pédaler sur une pente raide, mais vous obtiendrez une excellente vitesse sur terrain plat ou en descente. La puissance produite par un moteur électrique peut être manipulée de la même manière. Pour convertir la puissance, nous devons utiliser une série d'engrenages connectés à l'arbre de sortie du moteur ou acheter un moteur avec des engrenages intégrés à l'arbre de sortie, appelé un moteur à engrenages. Un moteur à engrenages réduit la vitesse de l'arbre moteur de sa vitesse normale (1 000 à 20 000 tr / min) à une vitesse de sortie plus lente et plus facile à gérer pour un robot mobile.

Figure III.12 : *Moteur à engrenage*

Vous pouvez voir ici un petit moteur à engrenage à courant continu (à gauche) et à nouveau avec le moteur à courant continu retiré de la boîte à engrenages en plastique. Un moteur à engrenages peut être tout type de moteur électrique, à condition qu'il soit équipé d'un réducteur réduisant la vitesse de sortie de l'arbre du moteur. Chaque motoréducteur doit avoir un rapport de démultiplication qui spécifie le rapport entre la vitesse d'entrée et la vitesse de sortie de l'arbre de sortie du moteur. Par exemple, un motoréducteur avec un rapport de transmission de 100 : 1 implique que l'arbre de sortie du moteur à courant continu réel doit tourner 100 fois pour que l'arbre de sortie à engrenage effectue un tour complet.

III.2.11.1 Conduite d'un moteur

Le moteur à courant continu est le moteur le plus simple à mettre sous tension. Appliquer un signal positif à un fil et un signal négatif à l'autre et votre moteur doit bouger, comme illustré à la (figure III.12). Si vous échangez la polarité des fils, le moteur tournera dans la direction opposée. La vitesse du moteur dépend du niveau de tension d'alimentation positif - plus la tension est élevée, plus l'arbre du moteur tourne vite. La puissance du moteur est sa capacité à maintenir sa vitesse, même sous une charge et qui est déterminée par l'ampérage disponible à partir de la source d'alimentation. Lorsque la charge de travail du moteur augmente, l'ampérage des batteries augmente.

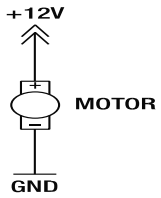


***Figure III.13** : un moteur CC met sous tension.*

Pour alimenter un moteur à courant continu, connectez simplement un fil à l'alimentation positive et l'autre à l'alimentation négative. Certains de nos robots ont des moteurs puissants pouvant fonctionner jusqu'à 24 Vcc. Si l'ensemble des 24 V est appliqué au moteur en même temps, il risque de faire tourner les pneus ou de faire un tour ! Nous ne voulons casser aucun de nos équipements ou frapper quelqu'un à proximité car notre robot se lance lorsque nous allumons les moteurs. Nous allons donc utiliser un contrôleur de moteur pour faire varier la tension du moteur de 0v à la tension d'alimentation (en la plupart des cas, 6v, 12v ou 24v). Cela permet au bot de démarrer lentement et d'atteindre son plein régime, ce qui réduit la pression sur les batteries au démarrage et permet un contrôle plus précis. Nous pouvons faire varier le niveau de tension aux moteurs en utilisant un signal PWM de modulation de largeur d'impulsion pour déterminer le cycle de fonctionnement de sortie ou le pourcentage de temps d'activation. Parce que l'utilisation de PWM signifie que la sortie est soit totalement activée soit complètement éteinte, les moteurs reçoivent autant d'ampérage que le permet l'alimentation pour le cycle de travail donné, et la puissance peut varier de 0% à 100% pour un contrôle de vitesse complet.

III.2.11.2 Le pont en H :

Lorsque vous conduisez un moteur à courant continu dans une seule direction, nous n'avons pas besoin de circuits spéciaux pour allumer ou éteindre le moteur, un simple commutateur en série avec une borne de moteur fera l'affaire. Mais pour inverser la polarité de la tension de la borne du moteur, il faut un circuit en demi pont ou un pilote push-pull. Ce circuit utilise deux commutateurs (S1 et S3), comme illustré à la Figure III.13.

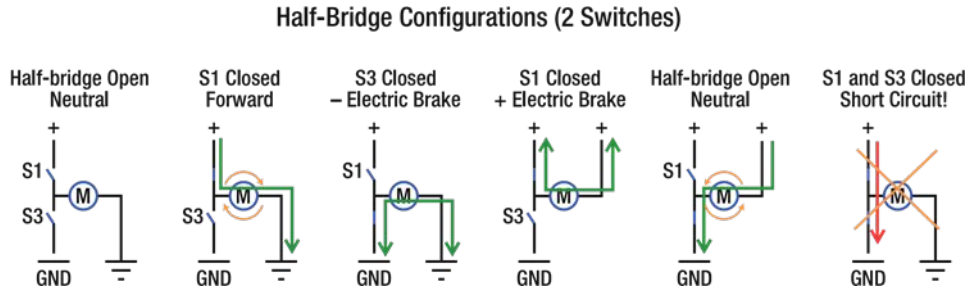


Figure III.14 : *un circuit demi pont*

Pour fournir un chemin entre une borne de moteur et l'alimentation de tension positive ou négative (masse). En utilisant un seul de ces commutateurs à la fois, un court-circuit est évité, l'autre terminal du moteur est connecté en permanence à VIN ou GND. Le pont sert à acheminer la polarité correcte aux bornes du moteur au moment opportun. Pour éviter un court-circuit, vous ne devez jamais fermer les deux commutateurs du même côté du pont (à la fois le positif et le négatif) en même temps.

Pour contrôler la polarité aux deux bornes du moteur, nous avons besoin de deux demi ponts identiques disposés dans un pont en H (voir la figure III.14).

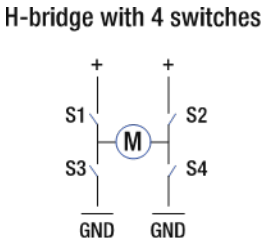


Figure III.15: *le pont H*

Pour que le moteur tourne, le courant de la batterie doit provenir de l'alimentation positive, du moteur et de la masse pour compléter le circuit. Pour que cela se produise, nous devons ouvrir un commutateur de chaque côté du pont, un côté bas et un côté haut opposé - cela

signifie que nous pouvons soit activer S1 et S4 pour avancer, soit activer S2 et S3 pour aller en arrière. La direction du flux de courant à travers les bornes du moteur détermine la direction dans laquelle le moteur tourne. Nous pouvons manipuler le flux du courant en fermant les deux commutateurs correspondants pour nous donner le contrôle directionnel du moteur. Si les quatre commutateurs sont ouverts (déconnectés), le moteur tourne en roue libre, ce qui signifie qu'il n'y a pas de chemin à parcourir pour le courant.

III.3 L'environnement de la programmation :

Le logiciel de programmation de la carte Arduino sert d'éditeur de code (langage proche du C). Une fois, le programme tapé ou modifié au clavier, il sera transféré et mémorisé dans la carte à travers de la liaison USB. Le câble USB alimente à la fois en énergie la carte et transporte aussi l'information ce programme appelé IDE Arduino.

III.3.1 Structure générale du programme (IDE Arduino) :

Comme n'importe quel langage de programmation, une interface souple et simple est exécutable sur n'importe quel système d'exploitation Arduino basé sur la programmation en C.

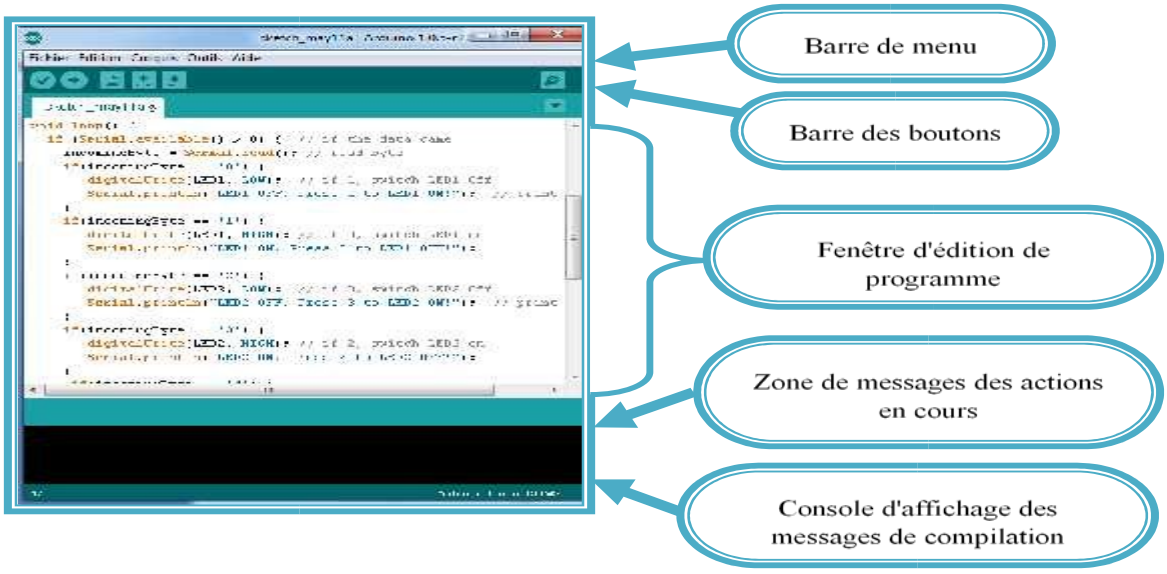


Figure III.16 : *Interface IDE Arduino*

III.3.2 Injection du programme :

Avant d'envoyer un programme dans la carte, il est nécessaire de sélectionner le type de la carte (Arduino Mega) et le numéro de port USB (COM 3) comme à titre d'exemple cette figure suivante.

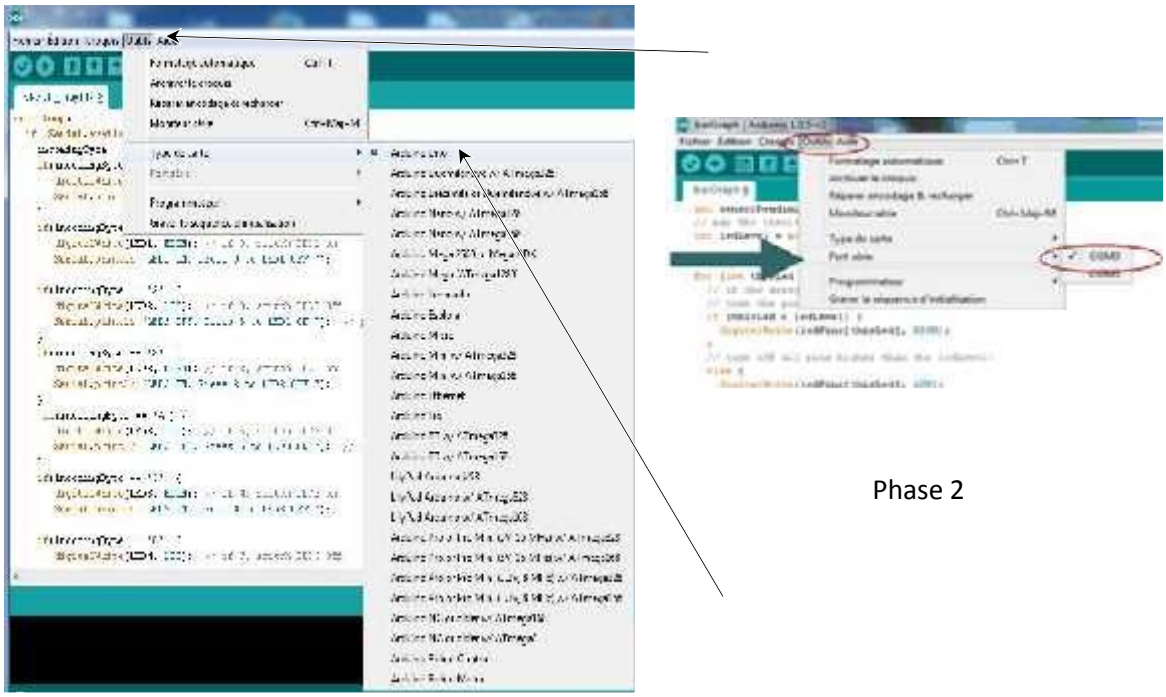


Figure III.17 : *Paramétrage de la carte Arduino.*

III.3.3 Description du programme :

Un programme Arduino est une suite d'instructions élémentaires sous forme textuelle (Ligne par ligne). La carte lit puis effectue les instructions les unes après les autres dans l'ordre défini par les lignes de codes.

III.3.4 Commentaires :

Les commentaires sont, en programmation informatique, des portions du code source ignorées par le compilateur ou l'interpréteur, car ils ne sont pas censés influencer l'exécution du programme.

```
1 /* programme de command DC moteur   avec Smartphone via Bluetooth----------
2  /*et fait également clignoter la diode de test de la carte----------
```

3 */----------

III.3.5 Définition des variables :

Pour notre montage, on va utiliser une sortie numérique de la carte qui est par exemple la 3ème sortie numérique, cette variable doit être définie et nommée ici moteur pin 3, la syntaxe est pour désigner un nombre entier est **Int.**

4 Int moteur 1 = 3 ; // mettre le moteur au pin 3----------

III.3.6 Configuration des entres et des sorties void setup () :

Les broches numériques de l'Arduino peuvent aussi bien être configurées en entrées numériques ou en sorties numériques, ici on va configurer moteur pin en sortie, pin mode (nom, état) est une des quatre fonctions relatives aux entrées-sorties numériques.

5 Void setup () {----------

6 // mettre le moteur 1 comme sortie : ----------

7 PinMode (moteur 1, OUTPUT) ; // lorsque le pin 3 est activé le moteur tourne----------
8} ----------

III.3.8 Programmation des interactions voidloop :

Dans cette boucle, on définit les opérations à effectuer dans l'ordre **digital write** (nom, état) est une autre des quatre fonctions relatives aux entrées -sorties numériques.

- Delay (temps en milliseconde) est la commande d'attente entre deux instructions.
- Chaque ligne d'instruction est terminée par un point-virgule.
- Ne pas oublier les accolades qu'encadre la boucle.

9 void loop () {----------

10 digital write (moteur1, HIGH); ----------

11 delay (3000) ----------

12 digital Write (moteur 1, LOW) ; ----------

13 delay (1000) ----------

14 }----------

III.3.9 Les étapes de téléchargement du programme :

Une simple manipulation enchaînée doit être suivie afin d'injecter un code vers la carte Arduino via le port USB.

1. On conçoit ou on ouvre un programme existant avec le logiciel IDE Arduino.
2. On vérifie ce programme avec le logiciel Arduino (compilation).
3. Si des erreurs sont signalées, on modifie le programme.
4. On charge le programme sur la carte.
5. On câble le montage électronique.
6. L'exécution du programme est automatique après quelques secondes.
7. On alimente la carte soit par le port USB, soit par une source d'alimentation autonome. (Pile 9 volts par exemple).
8. On vérifie que notre montage fonctionne.

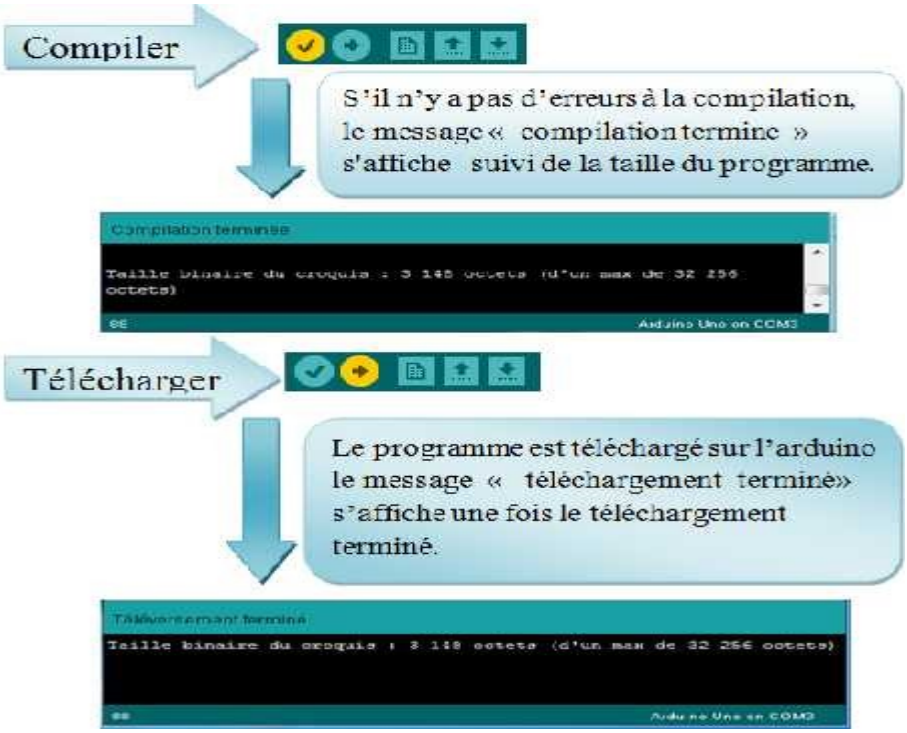


Figure III.18 : *Les étapes de téléchargement du code.*

III.4 Fonctionnement

Dans cette partie les commandes sont données vocalement à travers une application androïde connecté via un module Bluetooth, lié à l'Arduino.

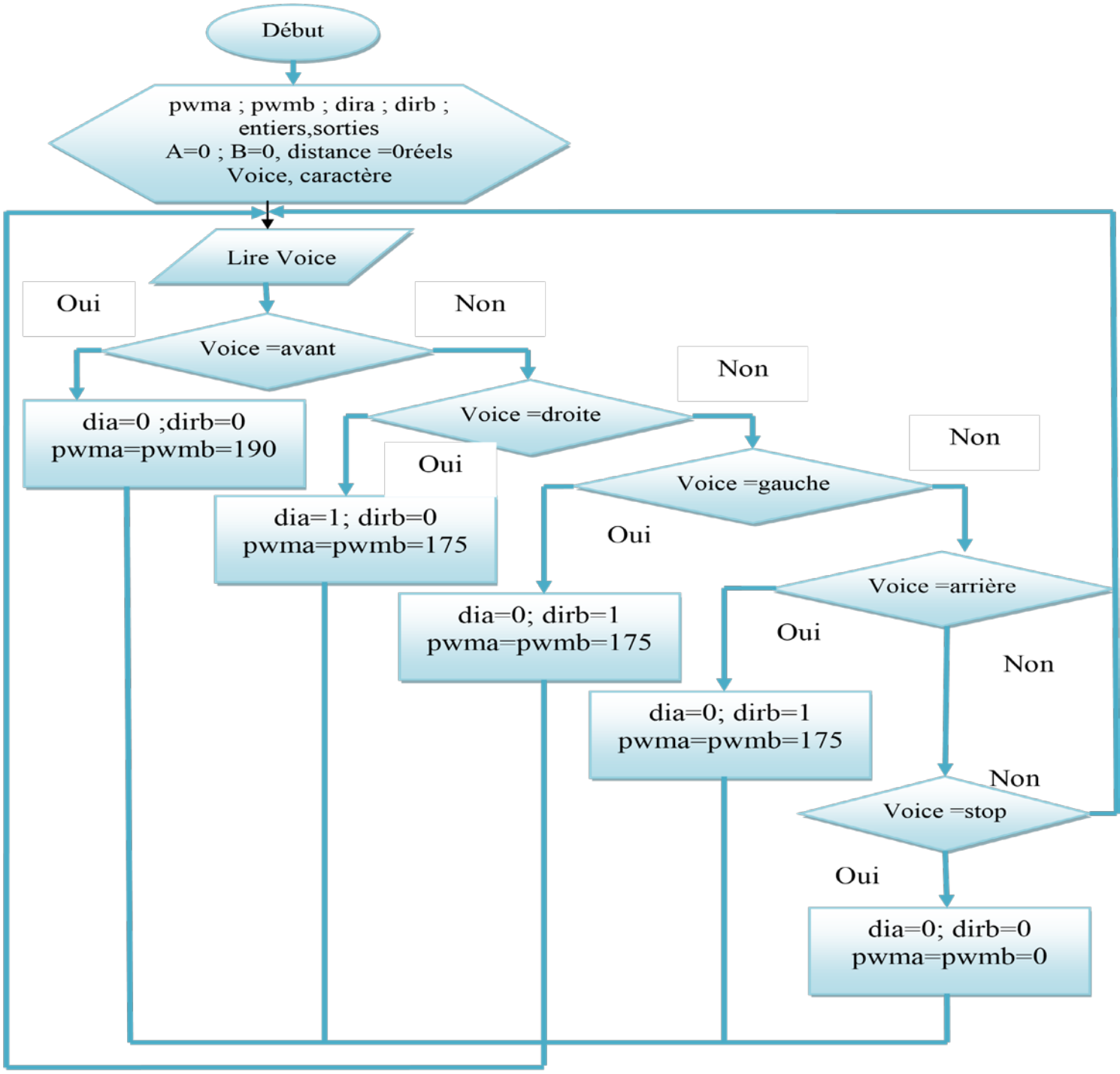
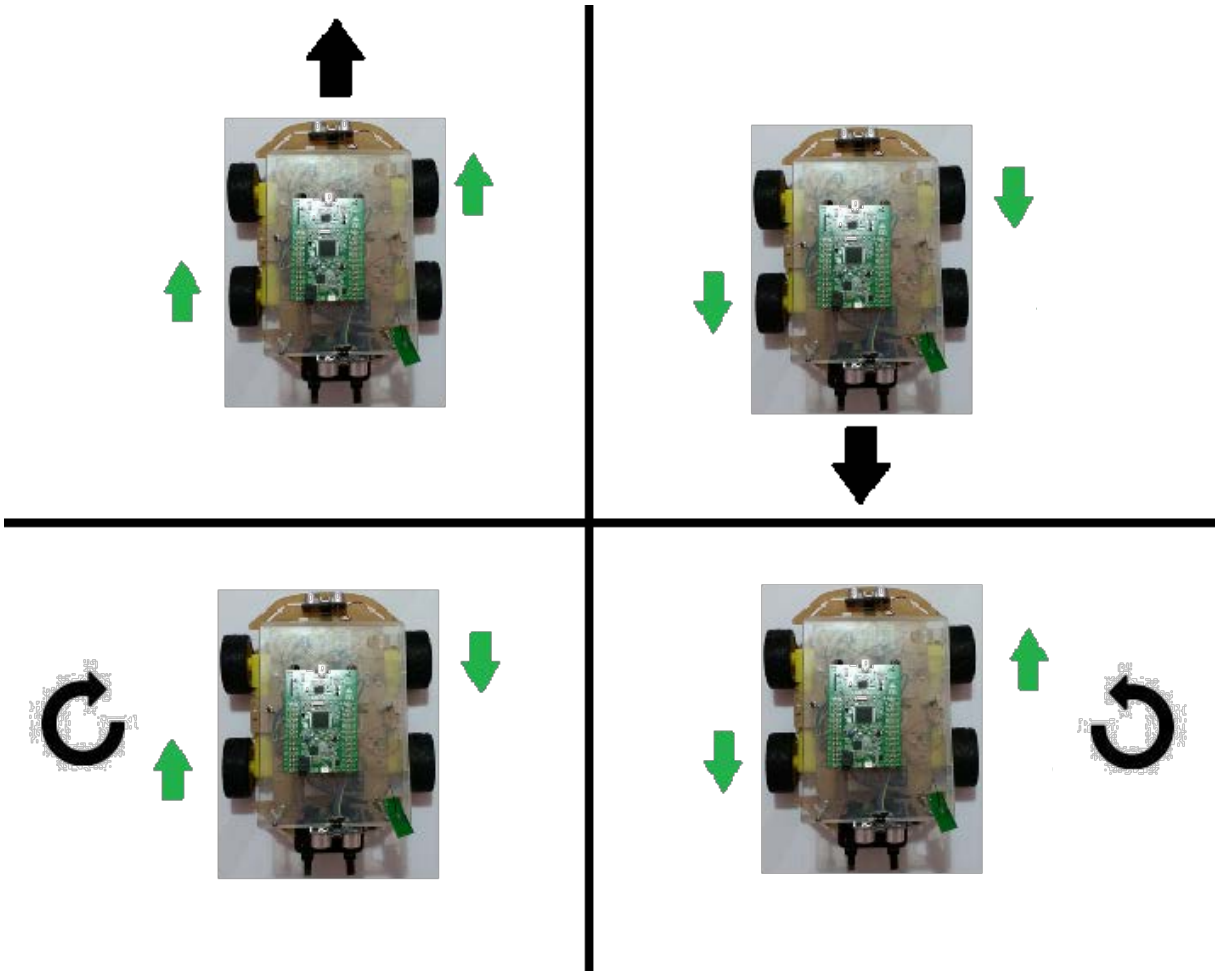


Diagramme de fonctionnement

- **Le comportement du rover selon le sens de rotation des roues**



III.5 Conclusion

Dans ce dernier chapitre, nous avons d'abord présenté et explicité une synthèse de différentes étapes qui nous ont permis de réaliser ce Rover, ainsi la description des composants utilisés et leurs tâches. Par la suite, on a présenté aussi une description générale du programme IDE implémenté sur la carte Arduino Atmega 2560.

Conclusion Générale

Ces dernières années l'automatique a fait l'objet d'une partie intégrante de notre vie et tend à se généraliser à tous les domaines où la miniaturisation, la mobilité, la puissance de calcul et informatique sont nécessaires.

Dans ce cadre, nous avons développé un système, qui permet de commander un rover à distance en utilisant un Smartphone. Ce rover peut être utilisé dans plusieurs domaines tel que l'exploration, la sécurité, le domaine militaire…etc.

La mise en œuvre de notre travail a exigé des connaissances très approfondies en programmation, ainsi que la maitrise de l'environnement d'Android.

En arrivant au terme de notre travail, on peut dire que nous avons atteint un double objectif, dont le premier est la réalisation de notre système qui est très complexe, et le second est enrichissement de nos connaissances en la matière grâce à plusieurs essais expérimentaux effectués.

Ce travail ne constitue qu'un début dans le domaine de la robotique mobile. Des fonctionnalités importantes manquent au rover comme l'autonomie de déplacement, intelligence artificielle, la caméra…etc.

Nous souhaitons dans un future proche aborder ces concepts pour fabriquer un robot mobile de qualité.

BIBLIOGRAPHIE

- [1] : A. ALLOUI, A. HADJ BRAHIM. « Proposition d'une solution multi-agent pour la commande et la coopération multi -robot mobile ». Mémoire d'ingénieur d'état en automatique, Université Biskra, Juin 2007.
- [2]: Mechatronics; Auteur: V.S. Bagad, ISBN9788184314908, Edition 2008.
- [3]: Computer Aided Manufacturing, C. Elanchezhian, G.ShanmugaSundar, First Edition 2005, Second Edition 2007.
- [4]: Robotics for Electronics Manufacturing: Principles and Applications in cleanroom automation; Auteur: Karl Mathia, ISBN978-0-521-87652-0Hardback.
- [5]: Robotics and control; Auteur: Mittal &Nagrath, ISBN 0-07-048293-4.
- [6] : BelkhadriaKhemisti, "commande d'un robot mobile par réseaux de neurones artificiels" Mémoire en vue de l'obtention du diplôme de magister en électronique, Option : Robotique. Université El HadjLakhdarBatna.
- [7] : A. PRUSKI. « Robotiquegénérale » Edition Ellipase 1988.
- [8] : Robotics (par AppuuKuttan) ; Auteur : AppuuKuttan, ISBN978-81-89866-38-9
- [9] : https:// arduino.technologiescollege.fr/IMG/PDF/cahier_ () _initialisation.
- [10] :Le grand livre d'Arduino, chapitre I, page 3,4,5 et 6.
- [11] : Le grand livre d'Arduino, chapitre III, page 35.
- [12] : http:// fr.scribd.com/doc/109899931/Cours-système-d'exploitation-Android.
- [13] : Nouha kheyar, « Locate my car » école nationale des sciences de l'informatique, Tunis.
- [14] : Le grand livre d'Arduino, chapitre II, page 20.
- [15] : J. P. Haton, J. M. Pierrel, G. Perennou, J. Caelen, et J. L. Gauvain, "Reconnaissance automatique de la Parole" dunod, 1991.
- [16] :http://eskimon.fr/2498-arduino-annexes-g-utiliser-module-bluetooth-hc-05.
- [17]: Patel, S., Jung, S.-H., Ostrowski, J. P., Rao, R., & Taylor, C. J. (2002). Sensor based door navigation for a nonholonomic vehicle. In IEEE International Conference on Robotics.

Résumé :

Ce projet concerne la conception, la réalisation et la commande d'un robot mobile à quatre roues à l'aide d'une carte électronique" Arduino", son rôle est d'étendre les capacités d'action des humains dans des environnements variés, où le robot peut agir pour rendre des services directement ou à distance via une interface decommunication.

Son principe peut être utilisé aussi pour concevoir un fauteuil roulant pour les gens tétraplégies, commandé vocalement pour que ces gens puissent se déplacer sans la nécessité de l’accompagner.

Abstract:

This project concerns the design, production and control of a four-wheeled mobile robot using an electronic board "Arduino", Its role is to extend the senses and the capacities of action of humans in various environments, where the robot can act to render services directly or remotely via a communication interface.

It’s an example which can be used to make a voice controled wheelchair for personnes who had a quadriplegia, so they can move without the need to accompany.