**R**épublique **A**lgérienne **D**émocratique et **P**opulaire
**M**inistère de l'**E**nseignement **S**upérieur et de la **R**echerche **S**cientifique

**U**NIVERSITE **M**OULOUD **M**AMMERI DE **T**IZI-**O**UZOU

**F**ACULTE DU **G**ENIE **E**LECTRIQUE ET D'INFORMATIQUE
**DEPARTEMENT D'INFORMATIQUE**

# Mémoire de Fin d'Etudes de MASTER ACADEMIQUE

**Domaine :** Mathématiques et Informatique

**Filière :** Informatique

**Spécialité : Réseaux, Mobilité et Systèmes Embarqués (RMSE)**

Présenté par

**Salim BOUSSAA**

**Karim BOUZOUANE**

## Thème

# Commande d'un Rover via wifi

*Mémoire soutenu publiquement le 12/07/ 2016 devant le jury composé de :*

**Président : M. Hakim ACHOUR**
**Encadreur : M. Mahemmed DAOUI**
**Co-Encadreur : M. Smail DJIOUA**
**Examinateur : M. Rachida AOUDJIT**
**Examinateur : M. Mouhend REMDANE**

# *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 M. DAOUI MEHAMMED pour son orientation et sa disponibilité constante tout au long de notre travail.

Nous adressons tous nos remerciements à notre Co-promoteur de labo de développement « LABO DZ », ou nous avons effectué notre projet de fin d’études, M. DJIOUA SMAIL, qui nous a permis de mener à terme ce travail, par ses soutiens, ses précieux conseils et ses bien vaillances.

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.

# SOMMAIRE

**Introduction générale**

# Chapitre I

## Chapitre II

## Chapitre III



## Chapitre IV

# LISTE
# DES FIGURES

## Chapitre I



## Chapitre II



## Chapitre III

## Chapitre IV

# LISTE
# DES TABLEAUX

## Chapitre I



## Chapitre II



## Chapitre IV

# INRODUCTION GENERALE

# INTRODUCTION GENERALE

Les systèmes embarqués prennent une place de plus en plus importante dans notre société, ils servent à contrôler, réguler des dispositifs électroniques grâce à des capteurs, embarqués dans des robots, des véhicules etc... Ces systèmes embarqués sont souvent utilisés par le public dans la vie de tous les jours sans même qu'on ne s'en rende compte, par exemple dans les systèmes de freinage d'une voiture, le contrôle de vol d'un avion. L'un des domaines d'applications majeures de ces systèmes est la robotique.

Depuis le milieu des années 70, la robotique est devenue une science extrêmement populaire dans les milieux universitaire. Alliant un grand intérêt pédagogique et industriel,

Cette nouvelle science demande beaucoup de créativité et de connaissances pluridisciplinaires (mécanique, électronique numérique et analogique, électrotechnique, Programmation, Intelligence artificielle, Temps réel, Automatique…..).

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 « command d'un rover via wifi ».Notre but principal est de commander un rover via wifi pour permettre aux utilisateurs de contrôler ce rover par l'intermédiaire d'un Smartphone Android. Pour réaliser notre objectif on a utilisé le microcontrôleur STM32F4-D qui appartienne à la famille ST Microsystème, ainsi que d'autres équipements tel que le double pont-H, des moteurs, des capteurs, un module wifi etc…

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 le système embarque architecturé au tour de la carte STM32F4-D.
- La partie commande : c'est une application Android qui permet aux clients de commander le rover en utilisant une liaison wifi.

Ces deux parties sont relie par un module wifi qui permet d'instaurer la communication.

Ainsi que notre travail est divisé en quatre chapitres :

Le premier chapitre consiste en premier lieu, a introduire les notions générales sur les systèmes embarques ainsi que la plateforme Android. Le second chapitre est consacré à la robotique et aux capteurs.

Par ailleurs, la conception optée pour la réalisation de notre projet est présentée dans le troisième chapitre, le chapitre quatre illustre l'environnement de développement et les divers composants implémentés dans l'architecture de notre système.
Enfin on achève ce mémoire par une conclusion générale récapitulant tout notre travaille ainsi que ses perspectives.

# CHAPITRE I

# Systèmes Embarqués

## Introduction

Les systèmes électroniques et informatiques sont de plus en plus présents dans la vie courante. Les ordinateurs et micro-ordinateurs sont les systèmes les plus connus. Mais l'électronique et informatique se trouve maintenant embarquée dans de très nombreux objets usuels : les téléphones, les agendas électroniques, les voitures. Ce sont ces systèmes enfouis dans les objets usuels qui sont appelés systèmes embarqués.
Pour bien comprendre les notions de base de ces systèmes, ce premier chapitre donne une vue générale sur les systèmes embarqués ainsi qu'un aperçu sur les microcontrôleurs, et on l'achève par une présentation de l'Android.

## 1 Système embarqué

### 1.1 Historique

Les premiers systèmes embarqués sont apparus en 1971 avec l'apparition de l'Intel 4004. L'Intel 4004 était le premier microprocesseur, et le premier circuit intégré incorporant tous les éléments d'un ordinateur dans un seul boîtier : unité de calcul, mémoire, contrôleur d'entrées/sorties **[01].**
Alors qu'il fallait auparavant plusieurs circuits intégrés différents, chacun dédié à une tâche particulière, un seul microprocesseur pouvait assurer autant de travaux différents que possible. Très rapidement, des objets quotidiens tels que les fours à micro-ondes, les télévisions et les automobiles à moteur à injection électronique ne tardèrent pas à être équipés de microprocesseurs. Ce sont alors les débuts de l'informatique embarquée.
La figure I.1 illustre un bref historique des micro-processeurs selon leurs vitesses de calcul en mips et le nombre de transistors.

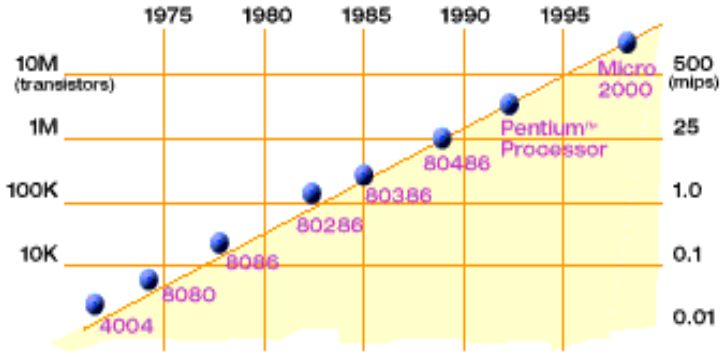



**Figure I.1** : Histoire des micro-processeurs.

### 1.2 Définition

Un système embarqué (Embedded system) est défini comme un système électronique et informatique autonome, spécialisé dans une tâche bien précise à la différence des ordinateurs personnels qui exécutent une variété d'applications [**01].**
Ce sont des systèmes dédiés à une fonction, possédant des ressources d'ordre spatial (taille limitée) et énergétique (consommation restreinte) limitées. Le terme de « Système Embarqué » désigne aussi bien le matériel que le logiciel utilisé.
Le logiciel créé pour les systèmes embarqués est appelé firmware. Étant enfoui et intégré (ce qui traduit le terme anglais Embedded) dans le matériel, Il est stocké dans une mémoire en lecture seule ou flash plutôt que dans un disque dur. Il fonctionne le plus souvent avec des ressources matérielles limitées : voire pas de clavier, un petit écran et peu de mémoire.
Le logiciel a une fonctionnalité fixe ; exécuter une application bien spécifique.

### 1.3 Architecture d'un système embarqué

L'architecture d'un système embarqué se défini par le schéma de la figureI.2**[02]** :

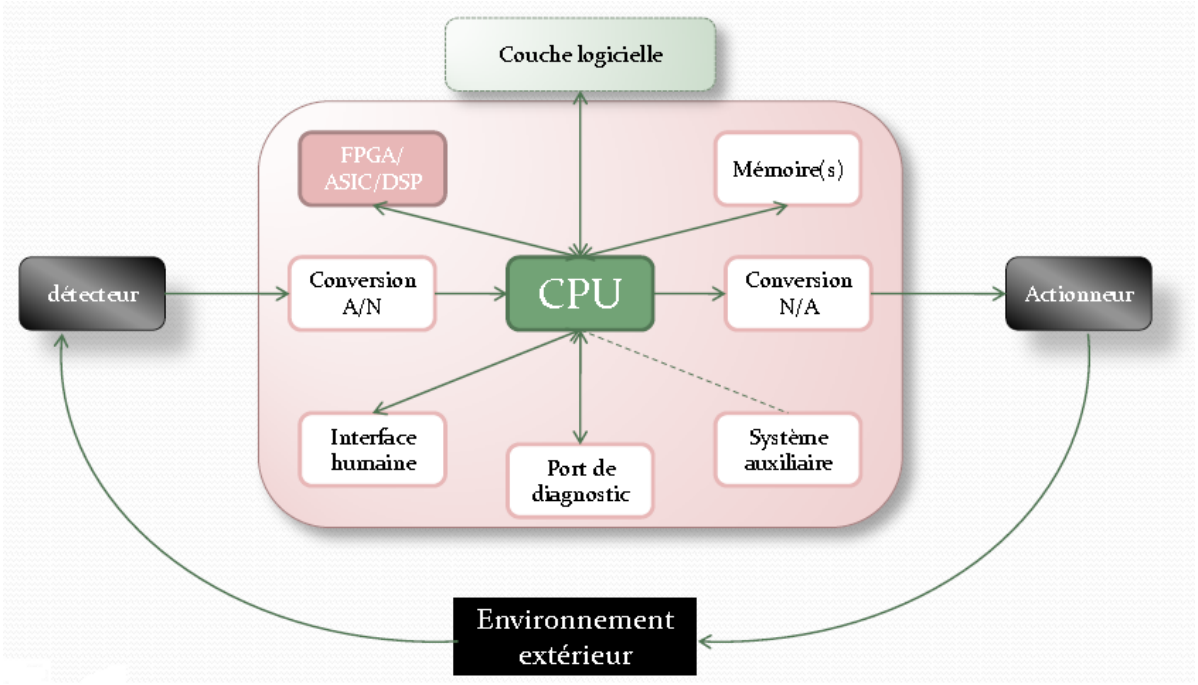



**Figure I.2** : Architecture d'un système embarqué.

Cette architecture peut varier selon les systèmes :

On peut par exemple, ne pas trouver de systèmes auxiliaires dans de nombreux systèmes embarqués autonome et indépendants. En revanche, l'architecture de base est la plupart du temps composée d'une unité centrale de traitement (CPU), d'un système d'exploitation qui se présente parfois uniquement sous forme d'un logiciel spécifique (ex : routeur), ou une boucle d'exécution (ex : ABS). De même l'interface IHM (interface homme-machine) n'est pas souvent existante, mais est souvent utile pour reconfigurer le système ou vérifier son comportement. Le fonctionnement du système se résume ainsi :

- Il reçoit des informations de l'environnement extérieur qu'il converti en signal numérique.
- L'unité de traitement composée du CPU, de la mémoire, du logiciel et éventuellement de système externe qui traite l'information.
- Le traitement génère éventuellement une sortie, les systèmes auxiliaires, les ports de monitoring ou l'IHM.

### 1.4 Caractéristiques des systèmes embarqués

Les systèmes embarqués sont principalement caractérisés par **[03]** :

- **La criticité** : les systèmes embarqués sont souvent critiques. En effet et, comme un tel système agit sur un environnement physique, les actions qu'il effectue sont irréversibles.
- **Exécution temps réel** : Ces systèmes doivent interagir avec leur environnement dans un laps de temps qui est imposée par ce dernier. Ceci induit donc des impératifs de temps de réponse. C'est pour cette raison que l'informatique embarquée est souvent basée sur un système temps réel.

- **L'autonomie** : Les systèmes embarqués doivent en général être autonomes, c'est à dire remplir leur mission pendant de longues périodes sans intervention humaine. Cette autonomie est nécessaire lorsque l'intervention humaine est impossible, mais aussi lorsque la réaction humaine est trop lente ou insuffisamment fiable.
- **La robustesse** : un système embarqué doit faire face à des conditions environnementales extrêmes, tels que les chocs, la variation de température, l'eau, le feu, etc. Ce dernier doit donc donner les mêmes performances dans n'importe quelles conditions environnementales, qu'elles soient extrêmes ou normales.
- **Tolérance aux fautes :** Certains systèmes embarqués doivent pouvoir remplir leurs fonctions malgré la présence de fautes, qu'elles soient d'origine physique ou humaine. Des méthodes de tolérance aux fautes sont nécessaires pour permettre au système de remplir ses fonctions en dépit des fautes pouvant affecter ses composants.
- **Fiabilité et sécurité de fonctionnement** : Les systèmes embarqués sont la plupart du temps dans des machines qui doivent fonctionner en continu pendant de nombreuses années, sans erreurs et, dans certains cas, réparer eux-mêmes les erreurs quand elles arrivent. Le système doit rester en marche pour des raisons de sécurité.

Et enfin ces systèmes sont soumis à des contraintes non fonctionnelles, comme :

- **L'espace mémoire** : La mémoire est une ressource limitée dans un grand nombre de systèmes embarqués, et par conséquent une bonne utilisation de la ressource mémoire est cruciale pour la gestion de ces systèmes.
- **Le packaging et l'encombrement** : difficulté de faire cohabiter dans un faible volume, l'électronique analogique et l'électronique numérique.
- **Consommation d'énergie** gardée le plus faible possible. Due à l'utilisation de batteries et, ou, de panneaux solaires, une consommation excessive augmente le prix de revient du système embarqué car il faut alors des batteries de plus forte capacité.

### 1.5 Complexité

Les systèmes embarqués requièrent souvent un faible encombrement (faible poids) comme les PDA (Personal Digital Assistant). Leur technologie fait alors appel à l'électronique et à des applications portables où l'on doit minimiser aussi bien l'encombrement que la consommation électrique **[04].** Par conséquent, la réalisation du packaging afin de faire cohabiter sur une faible surface de l'électronique analogique, de l'électronique numérique, des composantes RF (Radiofréquence) sans interférences est une tâche difficile. En effet, les performances des systèmes sur carte deviennent obsolètes dans le contexte des besoins actuels. Dans les stratégies de conceptions actuelles, un système embarqué est généralement intégré sur un support silicium unique constituant ainsi un système complet intégré sur une puce SoC (System on Chip).

### 1.6 Les contraintes de développement sur un système embarqué

Les contraintes de développements sur un système embarqué sont nombreuses et se rapprochent assez de celles rencontrées en informatique générale [05] :

- La complexité des algorithmes utilisés doit être relativement faible d'autant plus que les performances et la puissance des systèmes est moindre.
- L'absence de MMU limite la taille des applications.

- L'absence de media de stockage permanent a grande capacité.
- De nombreuses applications des systèmes embarquées sont des applications temps réels : On doit souvent ajouter une contrainte temps-réel.

### 1.7 Classification des systèmes embarqués

On peut classifier les systèmes embarqués comme suit **:**

- **Système Transformationnel :**

Assure des activités de calculs, qui lit ses données et ses entrées lors de son démarrage, qui fournit ses sorties, et meurt ensuite.

- **Système Interactif :**

Système en interaction quasi permanente avec son environnement, y compris après l'initialisation du système, la réaction du système est déterminée par les événements reçus et par l'état courant (fonction des événements et des réactions passés), le rythme de l'interaction est déterminé par le système et non par l'environnement.

- **Système Réactif ou Temps Réel :**

Système en interaction permanente avec son environnement, y compris après l'initialisation du système, la réaction du système est déterminée par les événements reçus et par l'état courant (fonction des événements et des réactions passées), mais le rythme de l'interaction est déterminé par l'environnement et non par le système, comme le montre la figure I.3 :

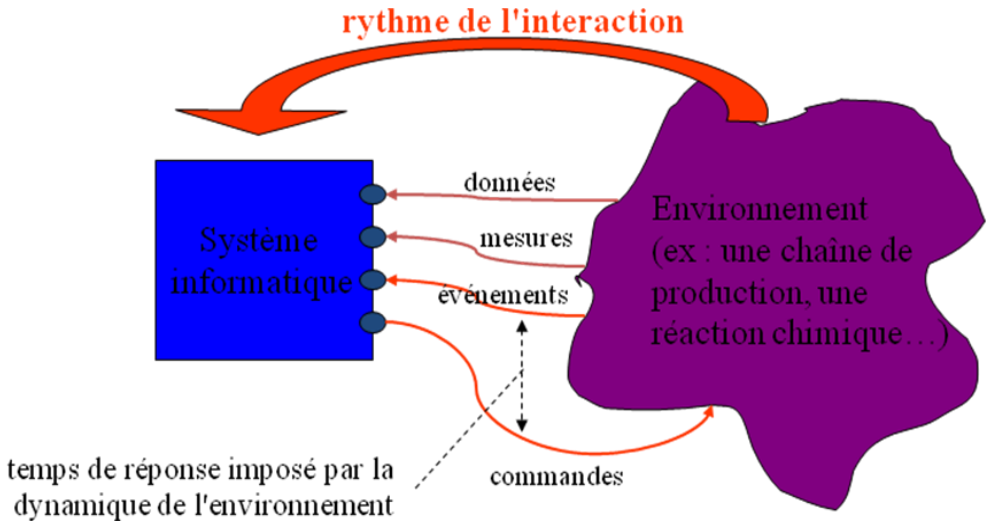



**Figure I.3 :** Architecture d'un système embarqué temps réel **[05].**

### 1.8 Domaines d'application des systèmes embarqués

Les domaines dans lesquels on trouve des systèmes embarqués sont de plus en plus nombreux tel que **[02]**:

- **Grand public** : Appareils photographiques et caméras, lecteurs DVD, chauffage et climatisation, éclairage, électroménager, domotique, sécurité (incendie, intrusion, surveillance, piscine), ascenseurs, HiFi, audio et vidéo, consoles de jeux, décodeurs, etc.
- **Transports** : Automobile, aéronautique, spatial, marine, assistance à la conduite ou au pilotage, maintenance, signalisation, contrôle du trafic aérien, maritime (aujourd'hui aide, demain automatique, objectif trafic autoroutier), distributeur de billets, radar, etc.
- **Défense** : Contrôle de trajectoire, lanceur, etc.

- **Secteur manufacturier et industrie** : Chaînes de production, automates, production et distribution d'électricité, réacteurs chimiques, réacteurs nucléaires, raffineries, dispositifs de sécurité, aide à la maintenance, etc.
- **Information et communication** : Imprimante, périphérique, téléphone, répondeur, fax, routeurs, téléphonie mobile, satellites, GPS, etc.
- **Santé** : Imagerie médicale, diagnostique, soins, implants, handicapés, etc.
- **Autres** : Carte à puce, distributeurs etc.

## 1.9 Système d'exploitation embarque

### 1.9.1 Définition

Un système d'exploitation (Operating System ou OS) embarqué est un système d'exploitation pouvant être installé sur un système embarqué. Ce système est conçu avec des spécificités afin de répondre à des besoins spécifiques au type de système embarqué.

### 1.9.2 Utilité d'un système d'exploitation embarque

Le rôle principal d'un OS embarqué est d'abstraire les ressources matérielles afin que l'application en ait une vision simplifiée. Cela se traduit en pratique par deux abstractions différentes. La première concerne les processeurs pour lesquels il s'agit de partager efficacement, Ce partage est réalisé grâce à un ordonnanceur (*scheduler*) qui sélectionne à un instant donné telle tâche pour une exécution immédiate.
La seconde concerne l'accès potentiellement partagé aux périphériques par l'intermédiaire de pilotes qui, d'une part, savent comment configurer et échanger des données avec le périphérique et qui, d'autre part, garantissent un accès par une unique tâche (accès dit atomique) si nécessaire **[04]**.

### 1.9.3 Exemples des systèmes d'exploitation embarqués

- **Linux embarqué**

Linux embarqué (Embedded Linux) est une adaptation du noyau Linux à un système embarqué à ressources limitées.
Les fonctionnalités du noyau Linux embarqué sont adaptées aux capacités du système embarqué [**06]** :

- Moins de mémoire requise
- Moins de services disponibles
- Boot depuis une mémoire ROM (FLASH)
- Pas de clavier ou de souris requis

Une version de Linux embarqué peut être configurée à la carte pour fonctionner sur une plateforme donnée.

- **TinyOS**

Tinyos développé et soutenu par l'université américaine de Berkeley, Il a été programmé en langage NesC. Est un système d'exploitation open-source conçu pour des réseaux de capteurs sans fil.

Il respecte une architecture basée sur une association de composants, réduisant la taille du code nécessaire à sa mise en place. Cela s'inscrit dans le respect des contraintes de mémoires qu'observent les réseaux de capteurs.

La bibliothèque de composant de TinyOS est particulièrement complète puisqu'on y retrouve des protocoles réseaux, des pilotes de capteurs et des outils d'acquisition de données. L'ensemble de ces composants peut être utilisé tel quel, il peut aussi être adapté à une application précise.

En s'appuyant sur un fonctionnement événementiel, TinyOS propose à l'utilisateur une gestion très précise de la consommation du capteur et permet de mieux s'adapter à la nature aléatoire de la communication sans fil entre interfaces physiques.

- **Windows CE:**

Windows CE (WinCE)est une variation de Windows pour les systèmes embarqués. Il utilise un noyau distinct plutôt qu'une version allégée et supporte les différentes architectures processeur (Intel x86, MIPS, ARM et Hitachi SH).

WinCE est optimisé pour les appareils possédant une faible capacité de stockage, le noyau peut tourner avec moins d'un mégaoctet de mémoire vive. Les systèmes sont souvent produits sans disque de stockage.

La figure I.4 présente l'espace de travail de Windows CE.


**Figure I.4** : Windows CE.

- **VxWorks**

VxWorks le leader de marché des SE embarques, est un système d'exploitation temps réel multitâche, développé par la firme Wind River, a été employé par la NASA pour les missions spatiales du programme Discovery, et les rovers martiens.
Il est principalement utilisé par la recherche et l'industrie (aéronautique, robotique, automobile, transport, télécommunication). Ce système peut gérer plusieurs cartes mères, chacune possédant à son tour des slots d'extension afin d'y ajouter des interfaces de toutes sortes.

## 1.10 Spécificités des logiciels embarqués

Plusieurs langages de programmation se veulent dédiés à l'embarqué parmi lesquels se trouve les langages proches de la machine comme le C et dans une moindre mesure le C++. Le langage assembleur reste encore un choix approprié pour les systèmes soumis à des contraintes sévères de temps réel.

D'autres langages orientés objet sont utilisés, et celui qui s'impose est le Java notamment Java Android qui est utilisé pour développer des applications destinées au système qui porte le même nom (Android), et qui est l'un des systèmes les plus connu pour la téléphonie mobile est les tablettes.

## 2 Les microcontrôleurs

### 2.1 Définition

Un microcontrôleur (µc ou MCU) est un circuit intégré assemblant dans un même boitier : un microprocesseur, plusieurs types de mémoires et des périphériques de communication (entrées-sorties).

Par rapport à des systèmes électroniques à base de microprocesseurs et autres composants séparés, les microcontrôleurs permettent de diminuer la taille, la consommation électrique et le coût des produits. Ils ont ainsi permis de démocratiser l'utilisation de l'informatique dans un grand nombre de produits et de procédés.

Il existe plusieurs familles de microcontrôleurs, qui se différencient par la vitesse de leur processeur et par le nombre de périphériques qui les composent. Toutes ces familles ont un point commun c'est de réunir tous les éléments essentiels d'une structure à base de microprocesseur sur une même puce **[07].**

Ils sont fréquemment utilisés dans les systèmes embarqués, comme les contrôleurs des moteurs automobiles, les télécommandes, les robots, l'électroménager, les jouets, la téléphonie mobile, etc.

### 2.2 Description et structure interne d'un microcontrôleur

Un microcontrôleur se présente sous la forme d'un circuit intégré réunissant tous les éléments d'une structure à base de microprocesseur. Voici généralement ce que l'on trouve à l'intérieur d'un tel composant :

- Un microprocesseur (C.P.U).
- De la mémoire de donnée (RAM et EEPROM).
- De la mémoire programme (ROM, OTPROM, UVPROM ou EEPROM).
- Des timers/compteurs pour générer ou mesurer des signaux avec une grande précision temporelle.
- Des convertisseurs analogiques-numériques (CAN).
- Des interfaces de communication (UART, $I^2C$, SPI, CAN, USB, Ethernet, etc.).
- Une sortie PWM (modulation d'impulsion).
- Un Watchdog : (surveillance du programme).

La figure I.5 donne les elements qu'on trouve generalement dans un microcontrôleur :

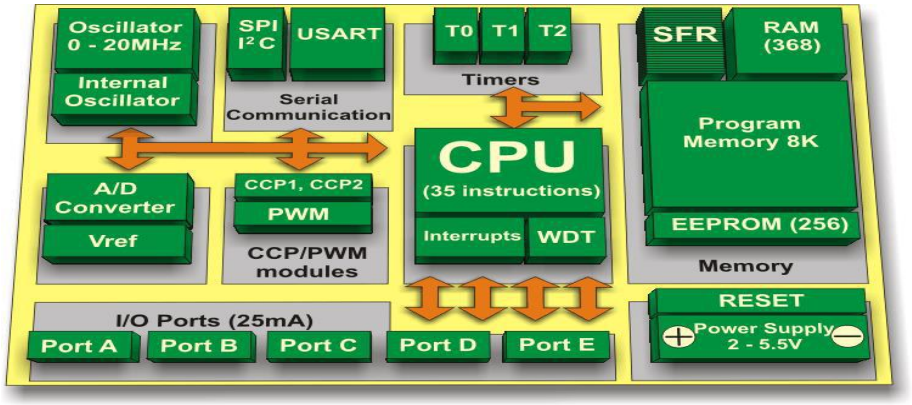



**Figure I.5** : Structure interne d'un microcontrôleur **[07]**.

### 2.3 Le microprocesseur

C'est l'unité intelligente de traitement des informations dans un microcontrôleur. Son travail consiste à lire des programmes (des suites d'instructions), à les décoder et à les exécuter. L'interfaçage du processeur avec l'extérieur nécessite 3 bus : un bus de données, un bus d'adresse et un bus de commande.

Le microprocesseur est composé de plusieurs éléments internes (figure I.6) dont:

- Une unité arithmétique et logique (ALU) permettant d'effectuer des opérations entre l'accumulateur et un opérande.
- Un compteur programme pointant l'adresse de la prochaine instruction à exécuter.
- Un ou plusieurs registres accumulateurs contenant temporairement les opérandes ainsi que les résultats des opérations,
- Un registre code condition indiquant certaines particularités en ce qui concerne le résultat de la dernière opération (retenu, zéro, interruption...).
- Des registres auxiliaires permettant de relayer les accumulateurs,
- Des registres d'index pour le mode d'adressage indirect,

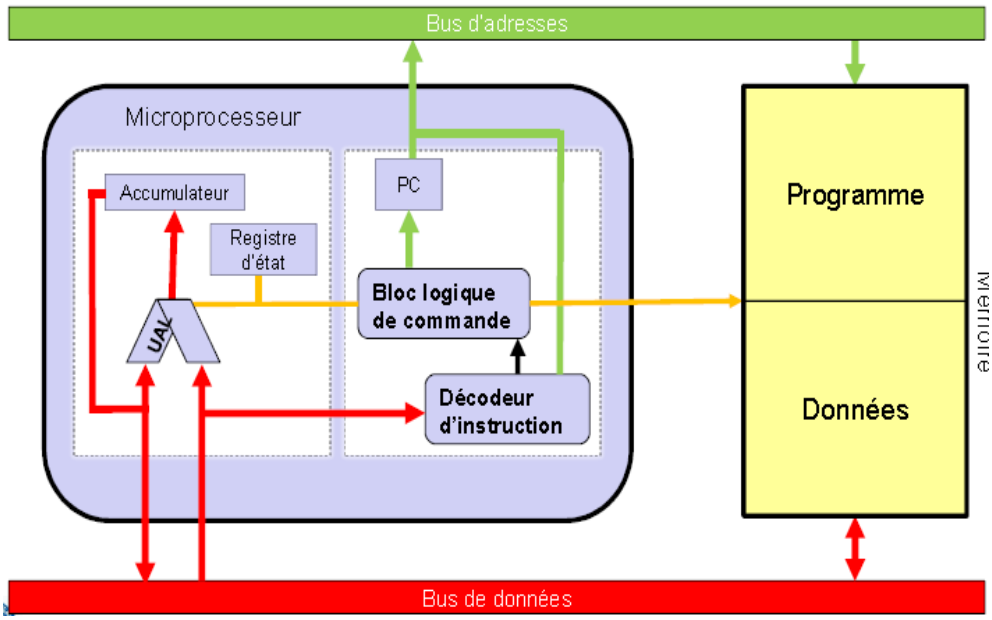



**Figure I.6** : Architecteur interne d'un microprocesseur **[08].**

Les microcontrôleurs améliorent l'intégration et le coût (lié à la conception et à la

réalisation) d'un système à base de microprocesseur en rassemblant ces éléments essentiels dans un seul circuit intégré.
On peut noter qu'il existe 2 catégories de microprocesseur : les CISC et les RISC.
CISC (Complex Instruction Set Computer) : Ce microprocesseur possède un nombre important d'instructions. Chacune d'elles s'exécute en plusieurs périodes d'horloges.
RISC (Reduced Instruction Set Computer) : Ce microprocesseur possède un nombre réduit d'instructions. Chacune d'elles s'exécute en un nombre fixe.

### 2.4 Les entrées/sorties GPIO

C'est l'abréviation de « General Purpose Input/Output », ou plus simplement entrées/sorties. Ces entrées sorties permettent d'étendre les fonctionnalités de microcontrôleur en lui donnant la possibilité d'agir sur l'environnement extérieur (leds, afficheurs LCD, lire l'état d'un interrupteur, d'un capteur…etc.).
Les GPIOs dispose de différents types de connexion :

- Des broches utilisables en entrée ou sortie numérique.
- Des broches pour une interface série, I2C, SPI, UART.
- De broches pouvant être utilisé en PWM (Pulse Width Modulation).

### 2.5 Les Interfaces de communication d'un microcontrôleur

#### 2.5.1 L'interface de communication parallèle

On désigne par liaison parallèle la transmission simultanée de N bits. Ces bits sont envoyés simultanément sur N voies différentes (une voie étant un fil, un câble ou tout autre support physique).
Les niveaux 0 et 1 sont envoyés simultanément sur chaque ligne, comme le montre la figure I.7.

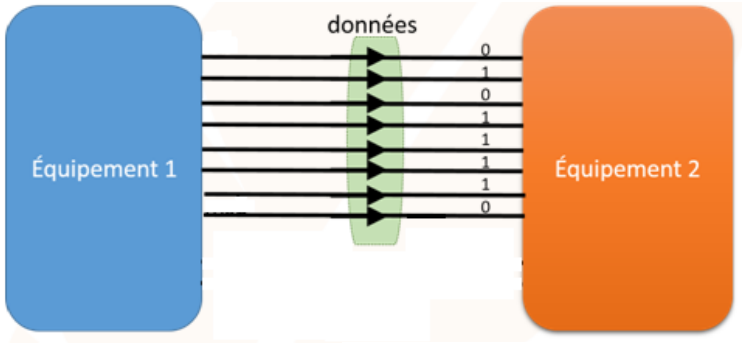



**Figure I.7** : L'Interface de communication parallèle.

#### 2.5.2 Les interfaces de communication série

L'interface de communication série permet au microcontrôleur de communiquer avec d'autres systèmes à base de microprocesseur. Les données envoyées ou reçues se présentent sous forme d'une succession de valeurs binaires. Il y a deux types de liaison série :

- **L'interface de communication série synchrone :**

Dans cette interface la transmission est synchronisée par un signal d'horloge émis par l'unité maître.

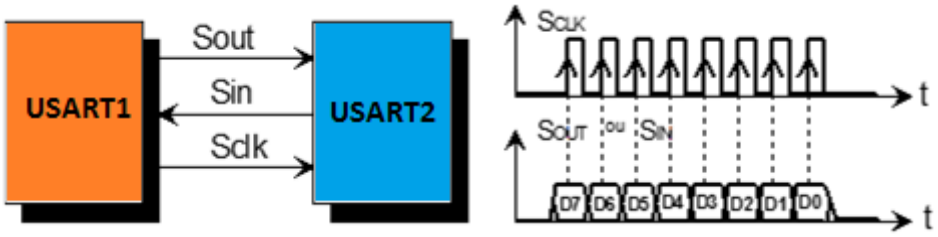



**Figure I. 8 :** L'interface de communication série synchrone.

- **L'interface de communication série asynchrone :**

Cette interface ne possède pas de signal d'horloge de synchronisation. Les unités en liaison possèdent chacune une horloge interne cadencée à la même fréquence. Lorsqu'une unité veut émettre un mot binaire, elle génère un front descendant sur sa ligne émettrice. A la fin de l'émission de ce mot, la ligne repasse au niveau haut. La donnée à transmettre peut contenir un bit supplémentaire appelé "parité" et servant à la correction d'erreurs.

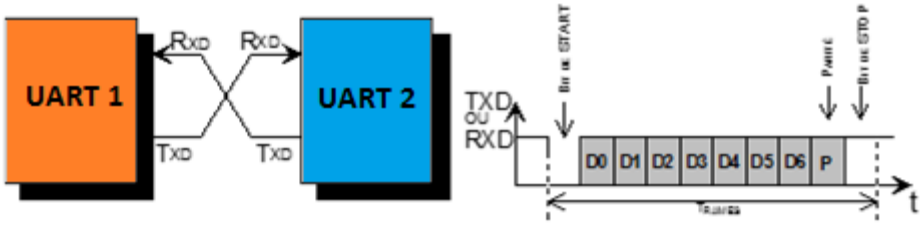



**Figure I.9** : L'interface de communication série asynchrone.

La combinaison des bits envoyer d'une interface de communication série a l'autre est appelée trame. Elle est définie par les spécifications suivantes **[09]**:

- Le niveau logique au repos entre les trames est le niveau 1 (haut).
- Un bit de START toujours à 0 sert à la synchronisation du récepteur.
- La taille des données peut varier généralement entre 7 et 8 bits, mais très souvent c'est 8 bits.
- Éventuellement un bit de parité paire ou impaire est ajouté.
- La trame est terminée par un bit STOP toujours à 1 (la durée peut varier entre 1, 1,5 et 2 temps bit).

La figure I.10 résume la constitution d'une trame :

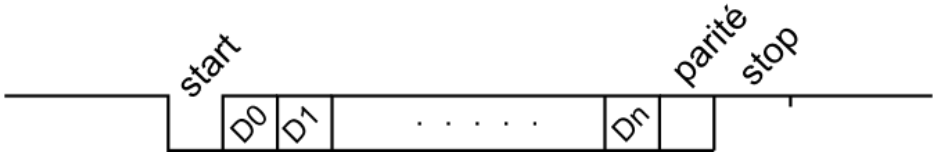



**Figure I.10 :** Constitution d'une trame

- ✓ **L'interface de communication I2C**

L'interface de communication I2C (Inter Integrated Circuit) a été développée au début des années 1980, par Philips pour minimiser les liaisons entre les circuits intégrés numériques. Elle fait partie de l'interface de communication série. Pour se connecter il faut trois fils :

- SDA (Signal DATA) : est utilisé pour transmettre les données.
- SCL (Signal CLOCK) : est utilisé pour transmettre un signal d'horloge de synchronisation.
- Un fil de masse(GND).

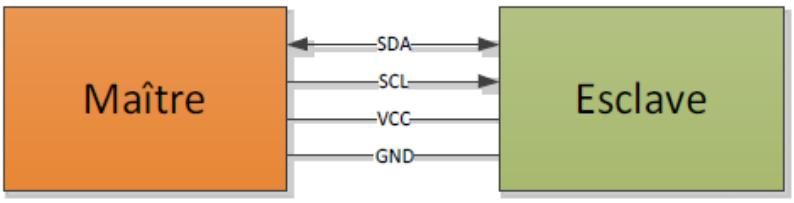



**Figure I.11 :** L'interface de communication I2C.

**Fonctionnement :**

L'interface de communication I2C fonctionne avec le mode adressage et de configuration à 7 bits.

Le microcontrôleur (Le maître) envoie un message synchronisé avec l'horloge (SCL) d'adressage à l'esclave et lorsque le message est envoyé, le microcontrôleur se met en mode écoute pour recevoir les valeurs de l'esclave ayant l'adresse envoyé **[10]**.

### 2.6 Les timers

Un timer est un compteur interne au microcontrôleur qui peut être incrémenté par un signal d'horloge interne ou externe, il peut y avoir un pré diviseur avent le timer, comme le montre la figure I.13 :

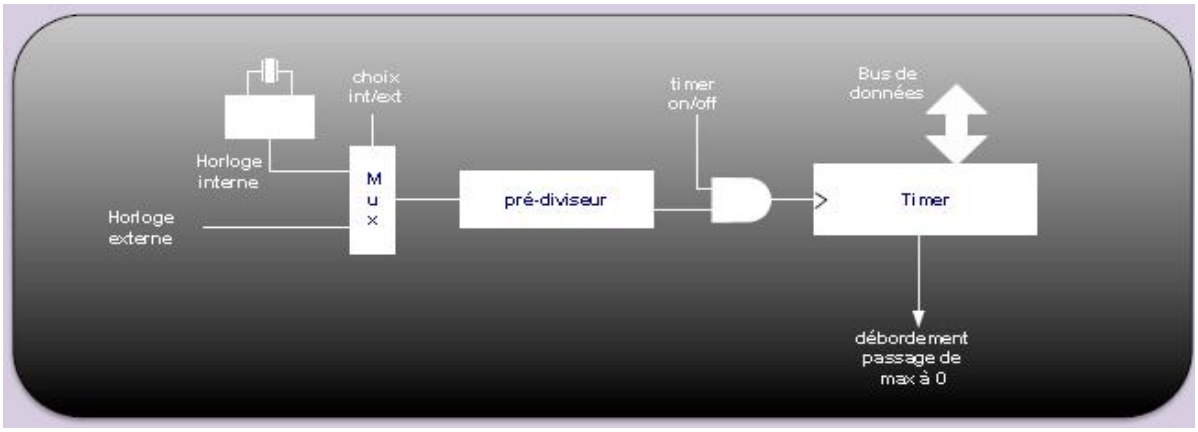



**Figure I.12 :** Structure d'un timer.

Un timer génère un drapeau de débordement à chaque passage de sa valeur maximal à zéro.

On peut lire à tout moment la valeur de timer.

Un Timer permet de réaliser les fonctions suivantes :

- Génération d'un signal périodique modulé ou non en largeur d'impulsion.
- Génération d'une impulsion calibrée.
- Temporisation.
- Comptage d'événements.

Plusieurs registres associés au Timer permettent de configurer les différents modes décrits précédemment.

### 2.7 Les caractéristiques principales d'un microcontrôleur

Les microcontroleurs sont des composants qui permettent la gestion des cartes, ils sont caractérisés par **[11]** :

- Un plus haut degré d'intégration
- Une vitesse de fonctionnement plus faible (de quelques mégahertz jusqu'à plus d'un gigahertz)
- Diminution de l'encombrement du matériel et du circuit imprimé
- Simplification du tracé du circuit imprimé (plus besoin de tracer de bus)
- Augmentation de la fiabilité du système
  - ✓ Moins de composants.
  - ✓ Moins de connexions composants/supports et composant circuit imprimé.
- Diminution de la consommation.
- Le microcontrôleur contribue à réduire les coûts à plusieurs niveaux (moins cher que les composants qu'il remplace)
- Diminution des coûts de main d'œuvre (conception et montage)
- Environnement de programmation et de simulation évolués

### 2.8 Les défauts des microcontrôleurs

- Le microcontrôleur est souvent surdimensionné devant les besoins de l'application
- Investissement dans les outils de développement
- Écrire les programmes, les tester et tester leur mise en place sur le matériel qui entoure le microcontrôleur
- Incompatibilité possible des outils de développement pour des microcontrôleurs de même marque.
- Fabrication uniquement en grande série >1000 **[11].**

### 2.9 Domains d'applications des microcontrôleurs

Les microcontrôleurs sont souvent utilisés dans l'élaboration de systèmes embarqués, nécessitant des traitements spécialisés (autoradios, téléphones portables, lecteur mp3, GPS, etc.).

Ces circuits intégrés sont également très prisés en robotique amateur et permettent de réaliser de nombreuses applications, y compris des robots autonomes, les automatismes en modélisme (maquettes de réseau ferroviaire).

### 2.10 Logiciels de développement

Le langage C est le langage de prédilection (avec l'assembleur) du développement sur microcontrôleurs. Une fois le programme compilé, le fichier binaire doit être envoyé au microcontrôleur. On utilise soit :

- Un programmateur de microcontrôleurs et souvent également d'EEPROM, on parle alors de programmateur universel.
- Un programmateur ISP qui a l'avantage de ne pas nécessiter de sortir le microcontrôleur du système électronique complet.

On peut alors utiliser le système. Toutefois, le programme qui a été envoyé peut comporter des bogues, aussi, pour parvenir à les détecter on utilisera par exemple un émulateur de circuit.

### 2.11 Différentes Familles de microcontrôleurs

On trouve plusieurs familles des microcontrôleurs citons :

- Le C167 de Siemens.
- La famille Hitachi H8.
- La famille Intel 8051.
- La famille des PIC de Microchip.
- La famille des ST, STM32 de STMicroelectronics.
- La famille MSP430 de Texas Instruments.
- La famille LPC de Philips.
- La famille Atmel AT91

.

Le tableau I.1 représente quelques exemples des microcontrôleurs :

| Reference | Fabricant | Vitesse | RAM | ROM / EPROM / FLASH | EEPROM | E / S Logiques | Timer | Entrees Analogiques | Particularite |
|---|---|---|---|---|---|---|---|---|---|
| 8051 | Intel | 12 Mhz | 128 o | 4 Ko | X | 32 | 2 | 0 | |
| 16C71 | Microchip | 20 Mhz | 36 o | 1Kx14 | X | 13 | 1 | 4 | RISC |
| 6805 S2 | Motorola | 4 MHz | 64 | 1 Ko | X | 16 | 2 | 8 | |
| 68HC11 A1 | Motorola | 8 MHz | 256 o | X | 512 | 22 | 1 | 8 | Etendu |
| AT90S 8515 | Atmel | 20 MHz | 512 o | 4 Ko | 512 | 32 | 3 | 8 | RISC |
| ST 6265 | Thomson | 8 MHz | 128 o | 4 Ko | 64 o | 21 | 2 | 13 | |

**Tableau I.1** : Exemples des microcontrôleurs.

## 3 Android

### 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]**.

### 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 I.13** : Logo Android.

### 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 (d A vers Z).

| Version | Nom de la version | Date de sortie | Évolutions |
|---|---|---|---|
| 6.0 | Marshmallow | 09/10/2015 | Autonomie en veille, Animation de demmarage, système UI Tuner, gestion des autorisation,Android Pay…. |
| 5.0 | Lollipop | 23/11/2014 | Moteur d'exéction 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 | Implementation 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) |

| | | | |
|---|---|---|---|
| 1.5 | Cupcake | 30/04/2009 | Envoi de vidéos vers YouTube et Picasa, rotation automatique |
| 1.1 | Banana bread | 22/10/2008 | Support pour sauvegarder les fichiers attachent aux MMS. |
| 1.0 | Apple pie | 11/11/2007 | Début de l'aventure Android. |

**Tableau I.2** : L'évolution de versions d'Android.

### 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 I.15 présente les différentes couches de la pile logicielle d'Android :

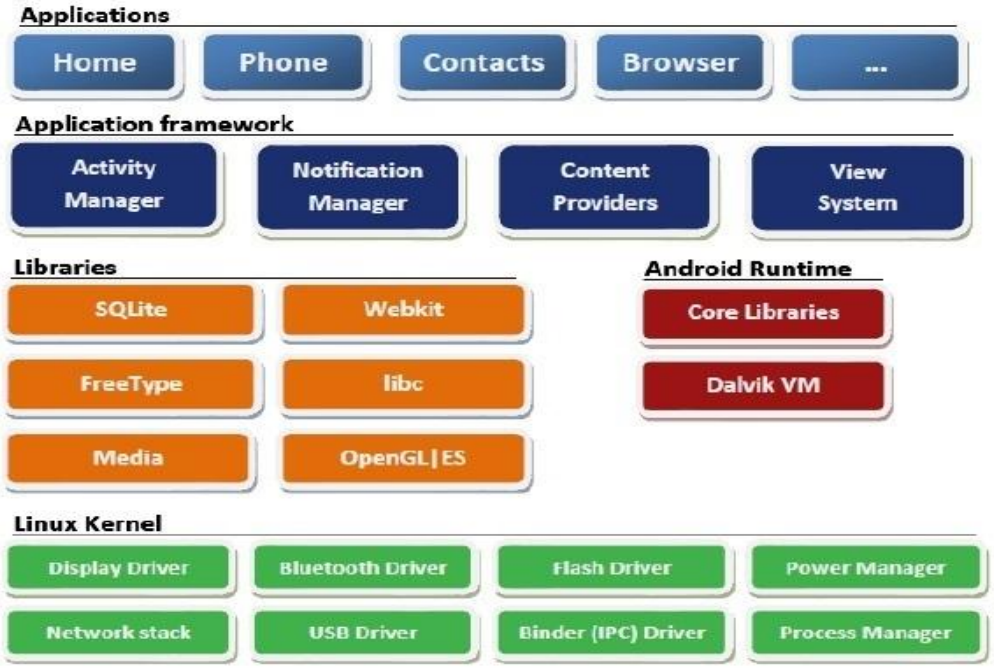



**Figure I.14 :** Architecture d'Android.

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.**[13]**.

### 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

### 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 manifest.

➢ **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 WIFI pour notre cas).

Une illustration explicative de ces concepts est représentée par le schéma de la figure I.16 :

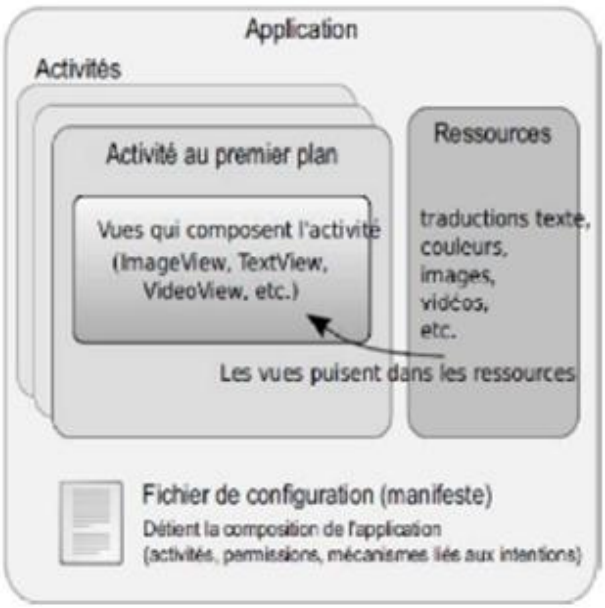



**Figure I.15** : Composants d'une application Android.

### 3.7 Gestion du Wifi sous Android

#### 3.7.1 Accéder à un réseau Wifi

Pour manipuler les données relatives aux réseaux Wifi, on doit faire appel à l'objet Wifi Manager. Cet objet sert d'interface entre le service système qui gère le Wifi et l'application. Il permet d'effectuer les opérations suivantes :

- ✓ Fournir des informations sur la connexion actuelle.
- ✓ Renvoyer la liste des points d'accès à portée.
- ✓ Manipuler les configurations enregistrées par l'utilisateur.
- ✓ Allumer ou éteindre la connexion Wifi.

#### 3.7.2 Activer ou désactiver le Wifi

Le premier point à aborder avec le Wifi, c'est bien évidemment de pouvoir obtenir son état (activé ou non) et au besoin de le modifier.

- ➢ **Obtenir l'état du Wifi**

```
private boolean estWifiActif(){
   WifiManager wifiManager = (WifiManager)getSystemService(Context.WIFI_SERVICE);
   if (!wifiManager .isWifiEnabled()){
      if (wifiManager.getWifiState() != WifiManager.WIFI_STATE_ENABLING) {
         return false;
      }
   }
   return true;
}
```

- ➢ **Activer /désactiver le wifi**

Pour activer/désactiver le wifi, il suffit d'instancier le Wifi Manager.
Cette instruction permet d'activer le wifi :

```
wifiManager.setWifiEnabled(false);
```

Cette instruction permet de désactiver le wifi :

```
wifiManager.setWifiEnabled(true);
```

### 3.7.3 Gérer les points d'accès

Avant de pouvoir rejoindre un réseau Wifi, on doit d'abord détecter les points d'accès environnants. Pour demander à l'appareil quels sont les points d'accès actifs, on utilise la méthode startScan. Le retour de cette méthode est immédiat : elle exécute une recherche en tâche de fond et les résultats seront connus plus tard grâce à la réception d'un événement synchrone. La valeur booléenne retournée par la méthode startScan indique uniquement si la détection des points d'accès a été initialisée.
Une fois la détection des réseaux Wifi lancée, nous pourrons récupérer le résultat en appelant la méthode getScanResults qui nous renverra une liste de résultats contenant les points d'accès disponibles dont :

- Leurs adresse Mac avec getAdresseMac(scanResult.BSSID).
- Leurs SSID avec getAPName(scanResult.SSID).
- La puissance du signal reçu avec getForceSignal(scanResult.level).

## Conclusion

Dans ce chapitre nous avons introduit les systèmes embarqués ainsi que les microcontrôleurs, nous avons aussi présenté la plateforme Android.
En ce qui concerne notre travail, nous allons réaliser un système embarqué basé sur la carte STM32F4-D pour le rover, pour le commander à distance nous allons utiliser un smartphone Android équipé d'une liaison wifi.
Dans le chapitre suivant, nous aborderons d'une part une généralité sur la robotique et d'autre part un aperçu sur les capteurs utilisés dans le domaine.

# CHAPITRE II

# Robotique
# &
# Capteurs

## Introduction

Les robots aujourd'hui 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. Demain, des robots seront aussi omniprésents et personnelle comme les ordinateurs personnels. Ces robots utilisent des capteurs pour bien mener les déférentes tâches et accomplir leurs missions.
Afin d'expliquer cela, ce chapitre est consacré d'une part à la robotique, et de l'autre aux capteurs.

## 1 La Robotique

### 1.1Historique de la robotique

La robotique est passée par plusieurs générations comme suit **[14]** :

- 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, est équipée 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 et 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. La planification de trajectoires, l'analyse et la modélisation de l'environnement de robot, appliqué sur des robots mobiles commerciaux. Également la recherche actuelle sur la conception mécanique des robots mobiles pour des applications hautement spécialisées, comme l'exploration sous-marine, les robots volants et le micro robot **[15]**.

### 1.2 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.

### 1.3 Les types des robots

Il existe deux grandes familles de robots :

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

#### 1.3.1 Les robots manipulateurs

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

##### 1.3.1.1 Les types des robots manipulateurs

Les robots manipulateurs viennent sous plusieurs formes.
Les formes se répartissent en en cinq grandes catégories :

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

**a) Robots cylindriques**

Le robot cylindrique, comme le montre la figure II.1, à deux axes de mouvement, un pour le mouvement en haut et en bas. La rotation se fait par la jonction à la base. De plus, le bras horizontal peut se déplacer à l'intérieur et à l'extérieur, ce qui donne un troisième axe de mouvement **[16].**

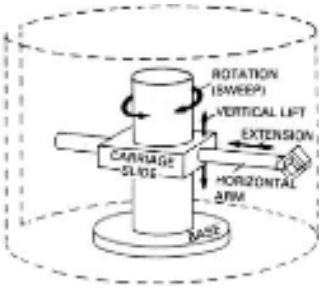



**Figure II.1** : Robot cylindrique.

**b) Robots rectilignes**

Les robots rectilignes (figure II.2) ont trois axes de mouvement (x, y, z). Pour cette raison, ils sont parfois appelés Robot cartésien. Ces robots sont exploités par vérin pneumatique **[16].**

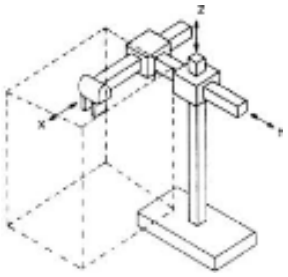


**Figure II.2** : Robot rectiligne.

**c) Robots sphériques**

Le robot sphérique (figure II.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 (à la base) et angulaires en haut ou en bas (sur le bras) **[16].**

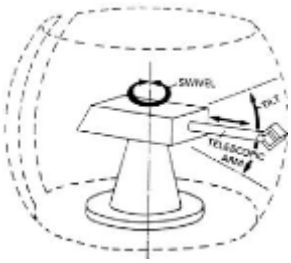



**Figure II.3** : Robot sphérique.

**d) Robots articulés**

Le bras articulé du robot ressemble à un bras humain. Il se compose de deux éléments, nommés l'avant-bras et le bras supérieur **[17].**

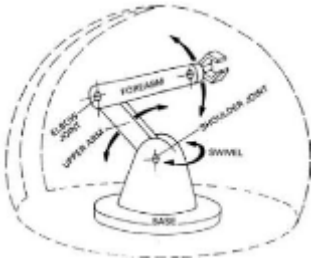



**Figure II.4** : Robot articulé.

**e) Robots SCARA**

Le terme «SCARA» signifie «Selective Compliance Arm for Robot Assembly»

Un robot SCARA est défini dans la norme ISO, en tant qu'un robot comportant deux liaisons pivots parallèles pour fournir conformément à un plan sélectionné comme le montre la figure II.5.et peut être considérée comme un cas particulier d'un robot cylindrique **[18].**

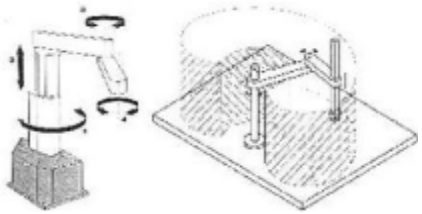


**Figure II.5** : Robot SCARA.

### 1.3.2 Les robots mobiles

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

#### 1.3.2.1 Architecture des robots mobiles

L'architecture des robots mobiles est présentée sur la figure II.6 se structure en quatre éléments :

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

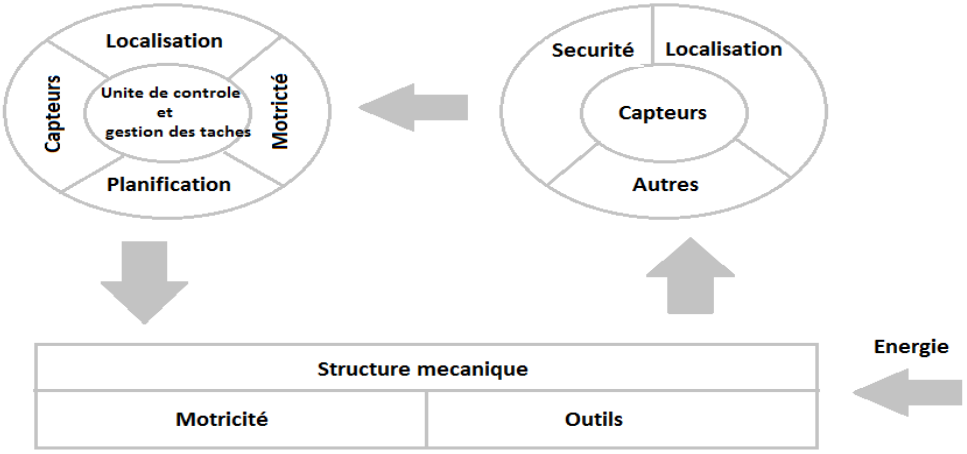



**Figure II.6** : Architecture d'un robot mobile.

#### 1.3.2.2 Classification des Robots Mobiles

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

##### 1.3.2.2.1 Les robots mobiles a 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 **[19]**.

**a) Robot unicycle**

Un robot de type unicycle est actionné par deux roues indépendantes comme le montre la figure II.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.

C'est un robot non-holonome, en effet il est impossible de le déplacer dans une direction perpendiculaire aux roues de locomotion.

Sa commande peut être très simple, il est en effet assez facile de le déplacer d'un point a un autre par une suite de rotations simples et de lignes droites **[20].**

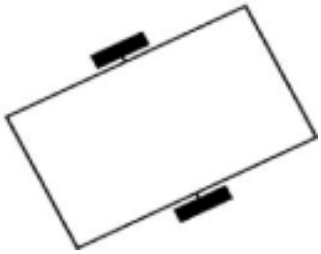


**Figure II.7** : Robot mobile unicycle.

**b) Robot tricycle**

Un robot de type tricycle (figure II.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.

C'est un robot non-holonome. En effet, 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 **[20].**

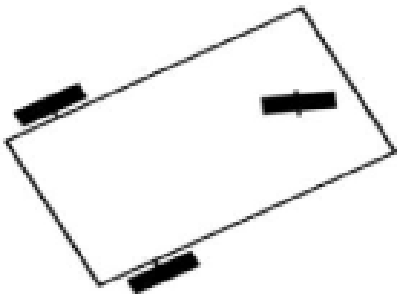


**Figure II.8** : Robot mobile tricycle.

**c) Robot voiture**

Un robot de type voiture 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, le deuxième pouvant être ramené au premier 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 **[20].**

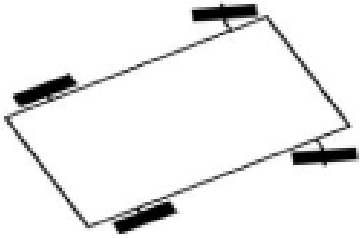


***Figure II.9** : Robot mobile voiture*

.

**e) Robot omnidirectionnel**

Un robot omnidirectionnel est un robot qui peut se déplacer librement dans toutes les directions. Il est en général constitué de trois roues décentrées orientables 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 **[20].**

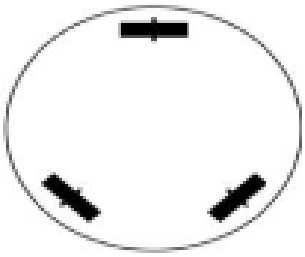


**Figure II.10** : Robot mobile omnidirectionnel.

- **Comparaison des différents types de robots mobilesà roues**

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

| | Avantage | Inconvénient |
|---|---|---|
| **Unicycle** | * Stable<br>* Rotation sur soi-même<br>* Complexité mécanique faible | * Non-holonome |
| **Tricycle** | * Complexité mécanique Modérée | * Non-holonome<br>* Peu stable<br>* Pas de rotation sur soi-même |
| **Voiture** | * Stable<br>* Complexité mécanique modérée | * Non-holonome<br>* Pas de rotation sur soi-même |
| **Omnidirectionnel** | * Holonome<br>* Stable<br>* Rotation sur soi-même | * Complexité mécanique importante |

***Tableau II.1 :*** *Les avantages et les inconvénients des robots mobiles à roues.*

### 1.3.2.2.2 Les robots mobiles à chenilles

L'utilisation des chenilles présentées l'avantage d'une bonne adhérence au sol. Et d'une faculté de franchissement d'obstacles. L'utilisation est orienté vers l'emploi sur sol accidenté ou de mauvaise qualité au niveau de l'adhérence (présence de boue, herbe,..).

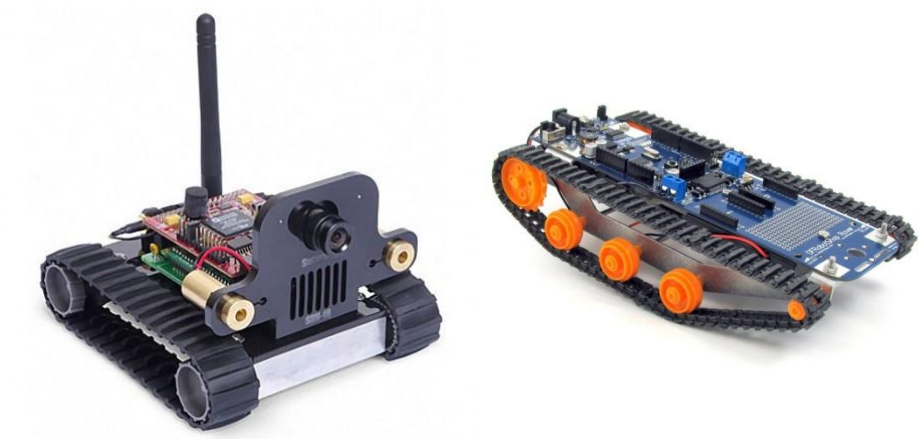


**Figure II.11** : Robots mobiles à chenilles.

#### 1.3.2.2.3 Les robots mobiles marcheurs

Les robots mobiles marcheurs sont destinés à réaliser des taches variées dont l'accès à un site difficile et dangereux à l'homme. Leur structure dans plusieurs degrés de liberté permet un rapprochement avec les robots manipulateurs. On distingue les robots marcheurs (figure II.12) à deux jambes (humanoïdes), à quatre pattes (type cheval), et a six pattes (type araignée).

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

#### 1.3.2.2.4 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 le montre la figure II.13.
Les applications de ce type de robots sont très spécialisées et les architectures des robots sont en général spécifiques à l'application visée **[21].**

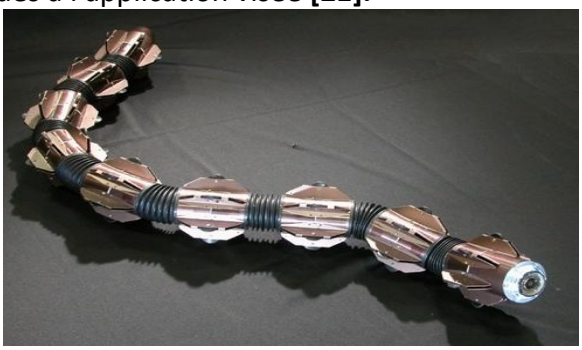


**Figure II.13** : Robot mobile rampant.

### 1.4 Domaine 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. Et le robot patient permet aux futurs chirurgiens-dentistes d'apprendre à soigner sans faire de dégâts.

### 1.5 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 **[22] :**

- 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 humains. Précision linéaire d'un robot typiquement est de 10 à 20 microns.

L'inconvénient des robots est qu'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 **[22]** :

- 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.

## 2 Les capteurs et éléments utilisés dans la robotique

### 2.1 Définition

Un capteur permet de convertir une grandeur physique en un signal électrique.
Ceci permettra un traitement du signal électrique par des structures électroniques à des fins de mesures et/ou de commande **[23]**.

### 2.2 Fonctionnement

Grâce à des lois élémentaires sur la physique, le capteur prélève une information physique (température, luminosité, humidité, débit, présence d'objet,…) et produit un signal électrique. Les caractéristiques de ce signal électrique (courant, tension, niveaux logiques, valeur moyenne, fréquence, amplitude, nombre binaire,…) dépendront directement de la grandeur physique à capter **[23].**

### 2.3 Architecteur d'un capteur

L'architecteur d'un capteur peut varier selon différents critères, la figure II.14 présente une architecteur générale d'un capteur :

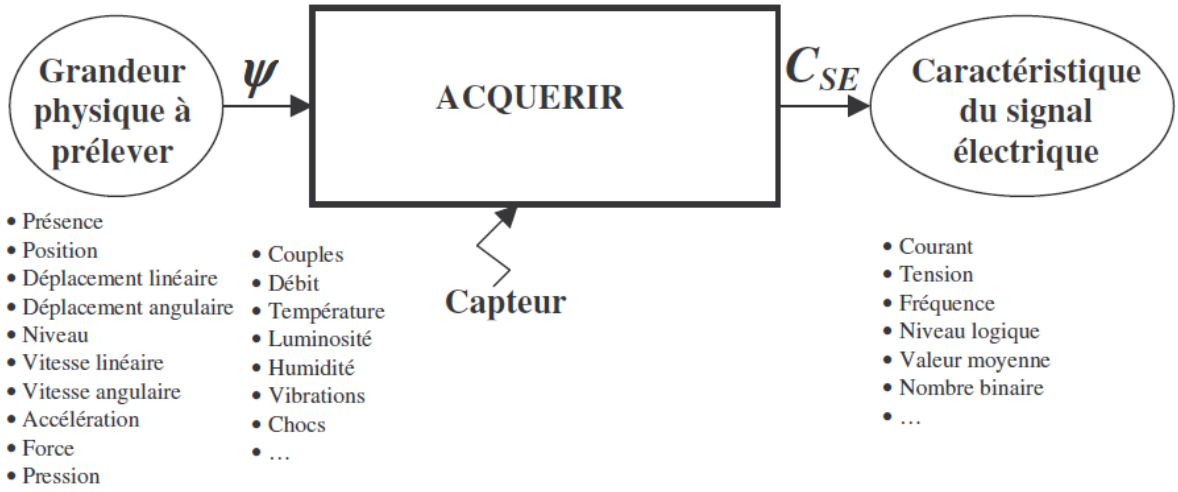



**Figure II.14** : Architecteur générale d'un capteur **[23]**.

### 2.4 Classification

#### 2.4.1 Apport énergétique

Selon leur mode de fonctionnement, on distingue deux catégories de capteurs :

##### 2.4.1.1 Les capteurs passifs

Ils ont besoin dans la plupart des cas d'apport d'énergie extérieure pour fonctionner (ex. : thermistance, photorésistance, potentiomètre). Ce sont des capteurs modélisables par une impédance. Une variation du phénomène physique étudié engendre une variation de l'impédance. Il faut leur appliquer une tension pour obtenir un signal de sortie.

##### 2.4.1.2 Les capteurs actifs

On parle de capteur actif lorsque qu'il effectue directement la transformation du phénomène physique en grandeur électrique. C'est la loi physique elle-même qui relie la mesure et la grandeur électrique de sortie.
Un capteur actif fonctionne assez souvent en électromoteur et dans ce cas, la grandeur de sortie est une différence de potentiel.

Le nombre des lois physiques permettant une telle transformation est évidemment limité, on peut donc recenser facilement les capteurs actifs (dont le nombre est fini). Toutefois, les domaines d'application sont eux très étendus.

### 2.4.2 Type de sortie

Les capteurs peuvent aussi faire l'objet d'une classification par leurs types de sortie :

#### 2.4.2.1 Capteurs analogiques

La sortie est une grandeur électrique dont la valeur est une fonction de la grandeur physique mesurée par le capteur. La sortie peut prendre une infinité de valeurs continues. Le signal des capteurs analogiques peut être du type :

- Sortie tension.
- Sortie courant.
- Règle graduée, cadran, jauge.

Exemple d'un capteur analogique typique :

- Capteur de température.
- Capteur d'alcoolémie.

#### 2.4.2.2 Capteurs numériques

La sortie est sous forme binaire. Elle peut prendre une infinité de valeurs discrètes. Le signal des capteurs numériques peut être du type :

- Des impulsions, avec un nombre précis d'impulsions ou avec une fréquence précise.
- Un signal carré.

Quelques capteurs numériques typiques :

- La boussole.
- Accéléromètres numériques.

#### 2.4.2.3 Capteurs logiques

Appelés aussi capteurs TOR (toute ou rien). Leurs sorties est un état logique 1 ou 0. Le signal des capteurs logiques peut être du type :

- Courant présent/absent dans un circuit.
- Potentiel, souvent 5 V/0 V.
- Led allumée/éteinte.
- Signal pneumatique (pression normale/forte pression).

Quelques capteurs logiques typiques :

- Capteurs de fin de course.
- Capteurs de rupture d'un faisceau lumineux.
- Divers capteurs de position.

### 2.4.3 Type de détection

- Détection avec contact (le capteur doit entrer en contact physique avec un phénomène pour le détecter).
- Détection sans contact (le capteur détecte le phénomène à proximité de celui-ci).

### 2.5 Caractéristiques

Un capteur est caractérisé selon plusieurs critères dont les plus courants sont :

- Etendue de mesure : Domaine de mesure pour lequel les indications du capteur ne doivent pas être entachées d'une erreur supérieure à l'erreur maximale tolérée. On appelle les valeurs limites du domaine, « portée minimale » et « portée maximale ».
- La sensibilité : C'est le rapport de la variation du signal de sortie à la variation correspondante de la grandeur à mesurer.
- La précision : C'est l'aptitude du capteur à donner des indications proches de la valeur vraie de la grandeur mesurée.
- Stabilité : La stabilité qualifie la capacité d'un capteur à conserver ses performances pendant une longue durée (problème de dérive du zéro par exemple).
- Rapidité : C'est l'aptitude du capteur à suivre dans le temps les variations de la grandeur à mesurer. Il faut donc tenir compte du temps de réponse, de la bande passante et la fréquence de coupure du capteur.
- Fidélité et justesse : La justesse est la qualité d'un capteur à fournir des indications précises. Tandis que la fidélité est la qualité d'un capteur à fournir des indications identiques pour une même valeur de la grandeur à mesurer.

## 3 Actionneurs

L'actionneur engendre un phénomène physique à partir de l'énergie qu'il reçoit. Par exemple :

- Une lumière (Diode LED, lampe …),
- Un rayonnement infrarouge (Diode infrarouge),
- Une chaleur (Résistance chauffante),
- Un mouvement (Moteur électrique).

## 4 Le Moteur DC

### 4.1 Définition

Un moteur à courant continu (figure II.15) est un moteur électrique. Il s'agit d'un convertisseur électromécanique permettant la conversion bidirectionnelle d'énergie électrique parcourue par un courant continu et un dispositif mécanique, selon la source d'énergie.

- En fonctionnement moteur, l'énergie électrique est transformée en énergie mécanique.

- En fonctionnement générateur, l'énergie mécanique est transformée en énergie électrique (elle peut se comporter comme un frein). Dans ce cas elle est aussi appelée dynamo.

**Figure II.15** : Moteur à courant continue.

### 4.2 fonctionnement du moteur DC

Le moteur à courant continu est composé de deux parties principales, comme le montre la figure II.16 :

- Rotor (partie qui tourne).
- Stator (partie fixe, statique).

En électrotechnique le stator s'appelle aussi inducteur et le rotor s'appelle l'induit. Sur la figure II.16, on peut observer au milieu – entouré par les aimants bleu et rouge qui constituent le stator – le rotor composé de fils de cuivre enroulés sur un support, lui-même monté sur un axe. Cet axe, est l'arbre de sortie du moteur. C'est lui qui va transmettre le mouvement à l'ensemble mécanique (pignons, chaîne, actionneur…) qui lui est associé en aval. Dans le cas d'un robot sur roues par exemple, on va mettre la roue sur cet axe, bien souvent par l'intermédiaire d'un réducteur qui diminue la vitesse de rotation tout en augmentant le couple **[25]**.

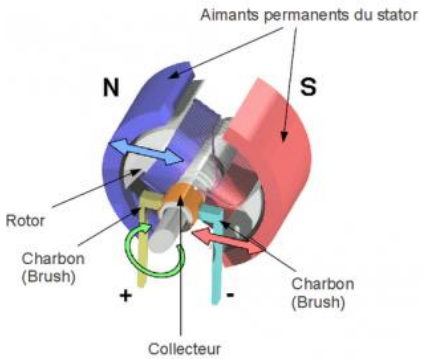



**Figure II.16** : Les parties d'un moteur a courant continue.

### 5 Le signal PWM

PWM (Pulse Width Modulation ou MLI Modulation à Largeur d'Impulsion) est un signal auquel la tension varier, sans modification ni d'amplitude ni de fréquence, mais en largeur de l'impulsion. En fait le signal PWM n'est pas continu, c'est un signal à impulsions (ressemble au signal carré). Il possède un état haut (un 1 logique) et un état bas (un 0 logique). La figure ci-dessous représentant un signal PWM :

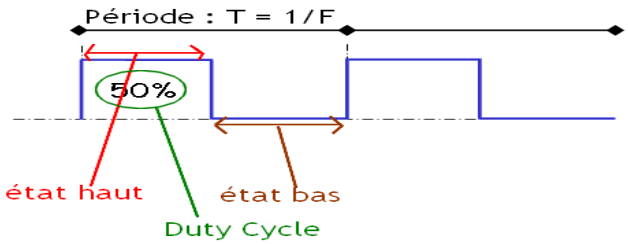



**Figure II.17** : Signal PWM à 50%.

**Conclusion**

A travers ce chapitre nous avons abordée les deux catégories principales des robots, ainsi que d'autres spécificités de la robotique, comme nous avons vu les capteurs et leurs classifications. Dans le chapitre suivant nous allons faire la conception de notre système.

# CHAPITRE III

# Analyse
# &
# Conception

## Introduction

Dans les chapitres précédents nous avons présenté une vue générale des systèmes embarqués ainsi que la robotique. Nous passons, à travers ce chapitre, à décrire la conception de notre système pour réaliser convenablement le travail demandé. Qui consiste a la réalisation d'un système embarqué, qui permet aux utilisateurs de commander un rover via une liaison wifi par l'utilisation d'un Smartphone Android.
Pour décrire notre conception, nous choisissons de donner en premier lieu une analyse du fonctionnement de notre système. Par la suite nous détaillons la conception de notre système au moyen des diagrammes d'appel de flux de données et de tâches.

## 1 Analyse et Conception

L'analyse et conception est primordiale dans toute système embarqué, elle doit être traitée avec rigueur et précision, car elle constitue la base du système à développer.
Avant de s'engager dans la conception, il est impératif de passer par la phase analyse. La conception s'appuie sur les résultats de la phase analyse.
Pour ce faire nous allons nous inspirer d'une méthode destinée spécialement aux systèmes embarques afin de mieux maitriser les caractéristiques structurelles et comportementales de notre système **[24]**.

Avant d'aborder les deux parties d'analyse et conception, la section suivante précise la notion de cycle de vie d'un système embarqué.

## 2 Notion de Cycle de vie d'un système embarqué

L'approche de base de processus de développement d'un system embarque suit en générale le cycle de vie suivant : phase d'analyse, de conception, développement à l'implémentation (mise en œuvre), test et déploiement. Et cela pour les systèmes complexes avec une longue durée de vie.
La figure III.1 présente le cycle de vie d'un système embarqué en générale :

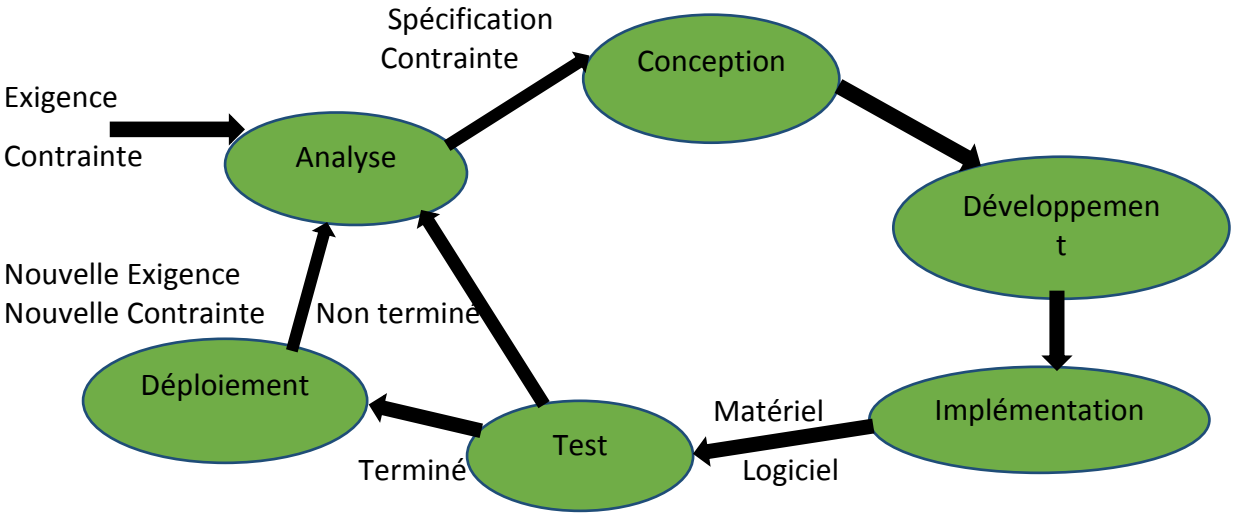



**Figure III.1** : Cycle de vie d'un système embarqué.

Nous aborderons chacune de ces étapes en rapport à notre travail dans la suite de ce chapitre et le chapitre suivant.

## 3 Analyse

L'analyse traite les exigences et les contraintes d'un système. Une exigence est un paramètre spécifique que le système doit satisfaire, par contre une contrainte est une limitation, dans laquelle le système doit fonctionner. En outre les spécifications sont des paramètres détaillés décrivant comment le système devrait fonctionner **[24]**.

Dans notre système on distingue deux parties bien différentes mais liées entre elles.
La partie s'exécutant sur un microcontrôleur (STM32F407-D) qui est implémenté sur le rover, et la partie qui s'exécute sur un smartphone Android, ces deux parties sont reliées par un module wifi ESP8266 qui permet d'instaurer la communication. De ce fait nous avons opté pour l'architecture client/ serveur.
La figure III.2 donne l'architecture de notre système.

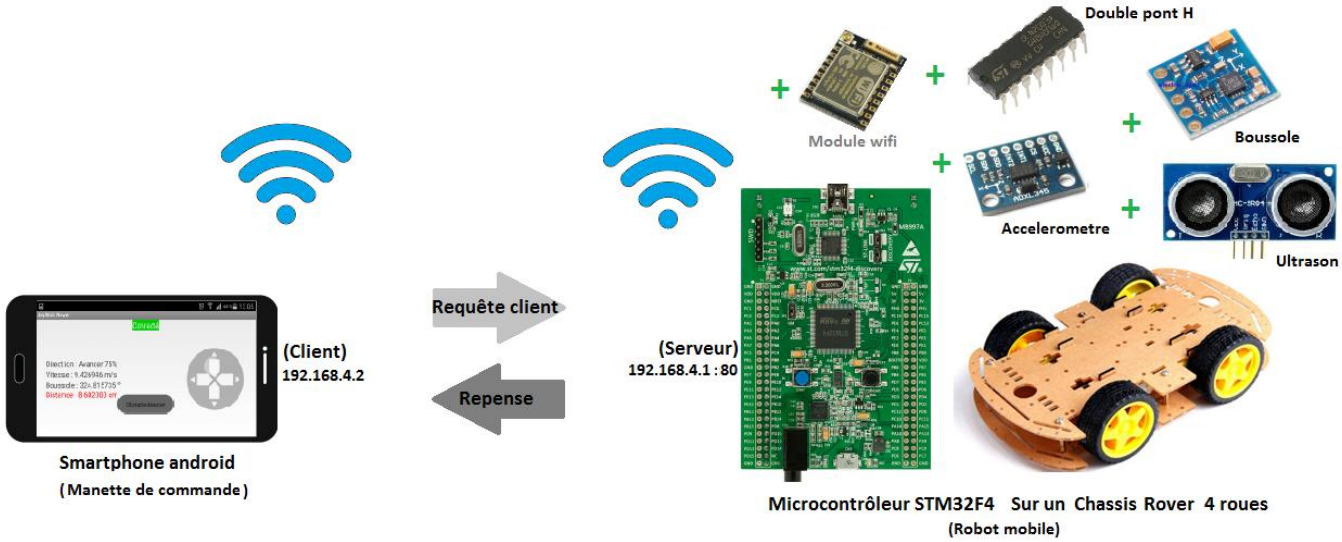



**Figure III.2 :** Architecture générale du système.

### 3.1 Fonctionnement

Chacune de ces entités (client/rover) effectue une tache bien précise :

- **Le client :** formé d'un Smartphone Android destiner à commander le rover via le Wifi, et visualise les différentes informations utiles (distance, orientation, vitesse), qui s'affichent en continue.

- **Le rover :** il est équipé d'un module Wifi (ESP8266) et d'un microcontrôleur (STM32F4-D). Ce dernier permet de jouer le rôle d'un serveur, cet ensemble forme le cerveau du rover, il s'occupe de l'interprétation et de l'exécution des requêtes (demandes) du client comme déplacer le Rover, récupéré des données (distance, orientation, vitesse) via des capteurs et de les renvoyer au client comme repense a sa demande.

  La communication entre ces deux entités est réalisée avec un protocole former de deux types de requêtes

- **Requête client :**
  C'est une requête de type **GET**, émanant du Smartphone vers le rover dont le corps contient une variable de la forme (nom, valeur), la valeur de cette variable précise la direction vers la quel le rover doit aller.

- **Repense :**
  Le contenu de cette réponse est de type texte, elle véhicule les données capturées par le Rover vers le Smartphone.

## 4 Conception

Au cours de cette partie nous construisons un modèle conceptuel du système (matériel et logiciel), par la présentation des trois diagrammes suivant :

### 4.1 Diagramme d'appel

Le diagramme d'appel représente la structure globale hiérarchique descendante du système, et le mécanisme d'accès aux données par l'interconnexion directe entre l'ensemble (matériel/logiciel).
Ces composants sont les suivants :

- **Logiciel** : ils apparaissent dans la partie supérieure du diagramme sous une forme ovale.
- **Matériel** : ils apparaissent dans la partie inférieure du diagramme sous une forme rectangulaire.
- **Flèche** : représente la direction de l'appel.

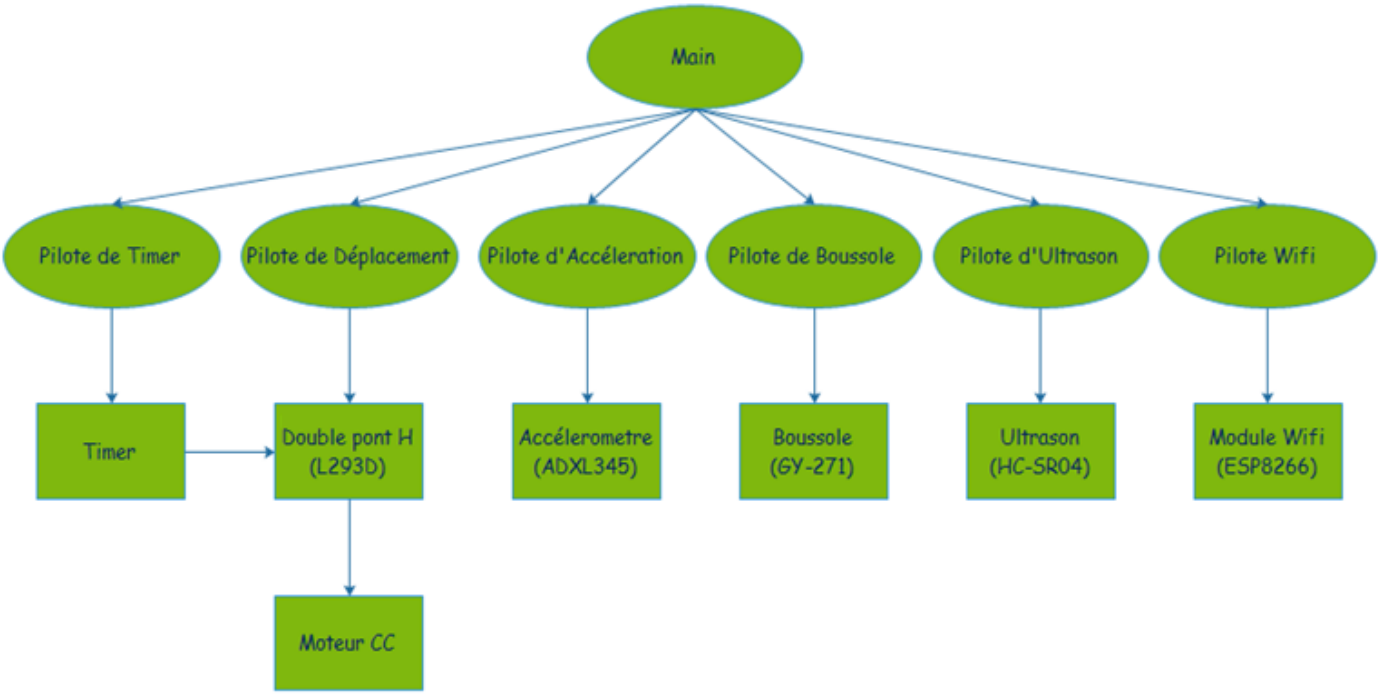



**Figure III.3** : Diagramme d'appel.

**Description :**
Tout d'abord le système commence par l'initialisation de la communication avec le module wifi par l'appel de la routine wifi, puis il attend les requêtes de déplacement (avant, gauche, arrière, droite) provenant du smartphone. Ces requêtes sont interprétées par la routine de

déplacement qui contrôle les quatre moteurs DC via le double pont H, grâce à des signaux PWM générés par le timer. Ce dernier est géré par la routine de timer. Une fois que la routine de déplacement est terminer le système appel respectivement les trois routines (routine d'ultrason, routine d'accélération, routine de boussole) pour récupérer les données, et les renvoie au Smartphone via le module wifi.

### 4.2 Diagramme de flux de données

Un graphe de flux de données illustre le fonctionnement global du système tout en cachant les détaille **[24],** il montre la circulation de flux internes des données de la source à la destination.

Les représentations graphiques logiciel/matériel du diagramme de flux de données sont pareilles avec le diagramme d'appel sauf qu'ici les flèches représentent le sens de circulation des données.

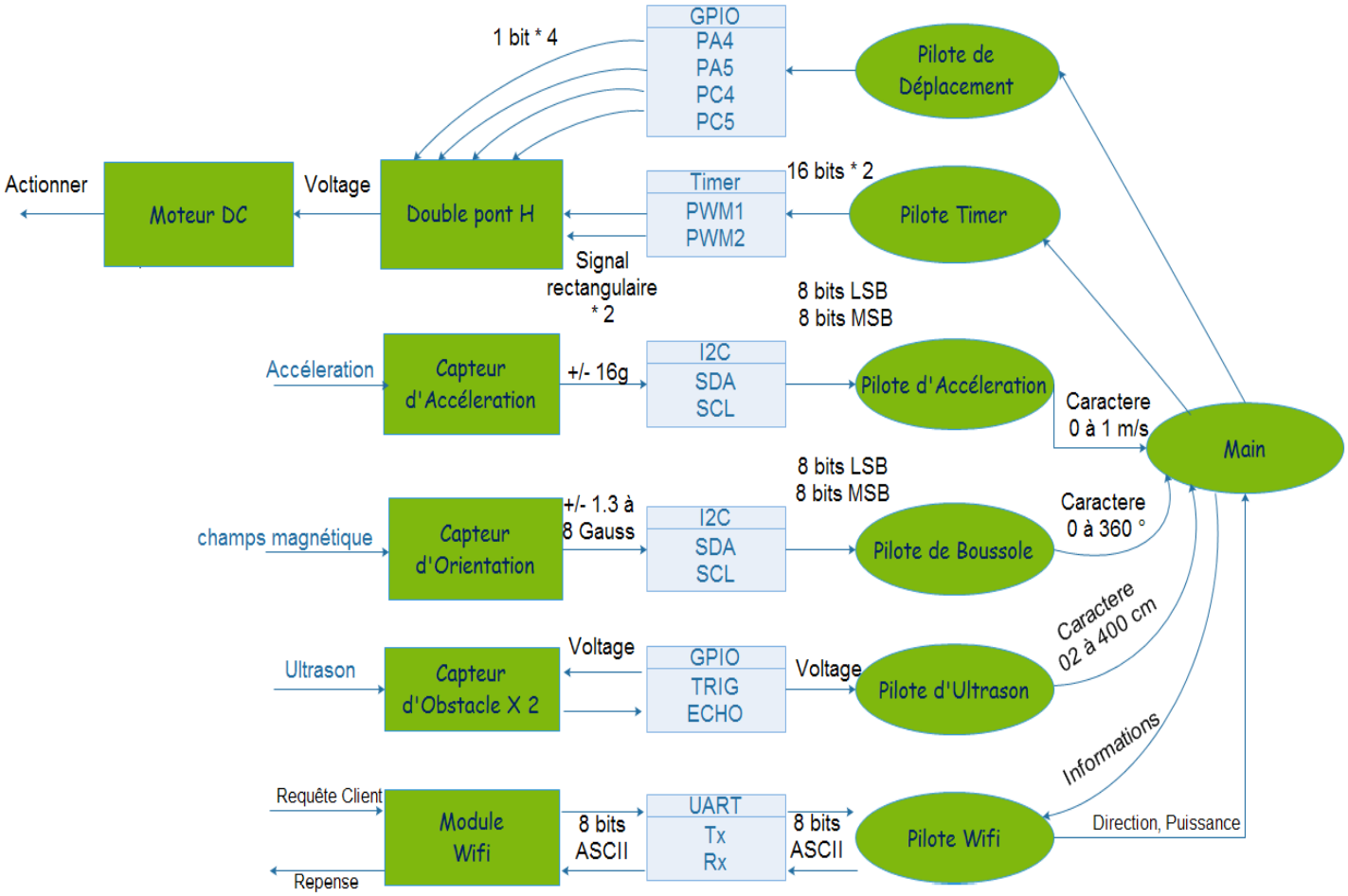



***Figure III.4 :*** *Diagramme du flux de données.*

**Description :**

- Le type des données qui circulent entre le mobile Android et le module Wifi sont de type texte (requête), donc elles sont des caractères ASCII.
  Au niveau de l'UART ces données sont envoyées caractère par caractère (8bits).

- La détection d'obstacle par l'ultrason se fait par l'envoid'une impulsion (5V) durant 10µs, à la pin trig.La durée de rebondissement de l'impulsion entre l'obstacle et l'ultrason est liée à la durée de l'impulsion renvoyée par la pin echo.Ensuite, on calcule la distance séparant le rover et l'obstacle déduite de cette impilsion.

- Après la détection de l'accélération par l'accéléromètre ou du champ mantique par boussole, ces derniers l'envoient aux interfaces de communication I2C sous format brute c'est-à-dire en 8 bits LSB (X0, Y0, Z0) et 8 MSB (X1, Y1, Z1) poids faible et poids fort respectivement des axes (X, Y, Z).Ensuite, cet ensemble de bits est reformulé en données claires qui peuvent être affichées sur l'écran du Smartphone.

- La routine de déplacement contrôle le sens de rotation de chacun des quatre moteurs du Rover grâce à 1 bits donc en tous 4 bits sont envoyés depuis cette routine au double pont-H L293D.

- Le Timer est géré par la routine du Timer qui lui envoie deux valeurs de 16 bits (0-65535), ces valeurs précisent la puissance des moteurs de chaque côté du Rover, en fonction de ces valeurs de 16 bits, Le Timer génère deux signaux PWM (une suite de 0 et de 1) au composant L293D.

- Le composant L293D alimente les moteurs avec une tension électrique (voltage).

### 4.3 Diagramme de tâches

Un diagramme de tâche est une représentation comportementale de la partie logicielle d'un système. Il présente le déroulement séquentiel et les déférentes routines qu'effectue le système pour réaliser ses tâches et cela par la décomposition successive d'une tâche en sous tâche pour atteindre une tâche plus simple. La décomposition d'une tâche se fait par l'une de ces méthodes soit séquentiel, conditionnel, itérative ou une interruption.

### 4.3.1 Diagramme de taches de l'entité Rover

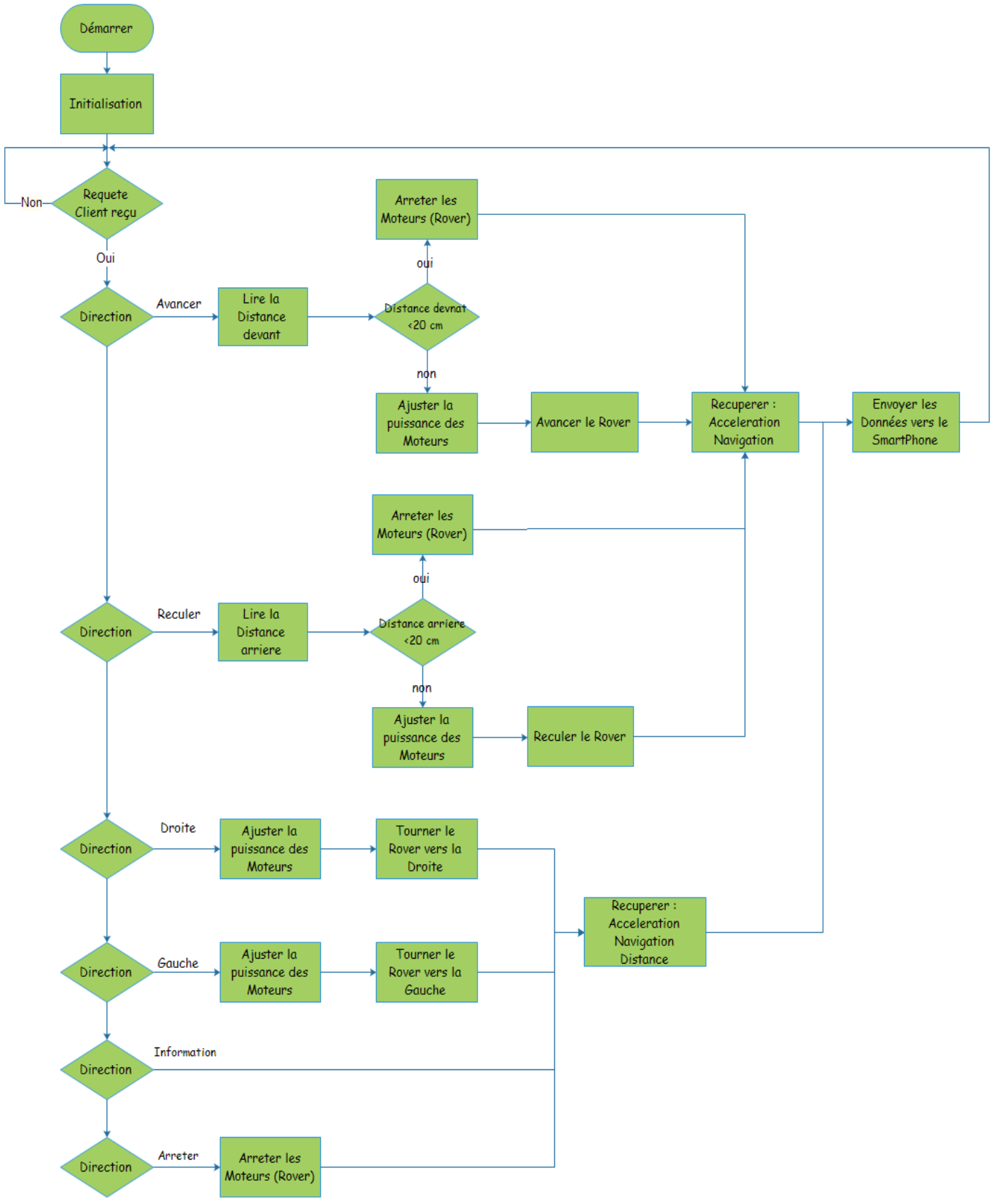



***Figure III.5 :*** *Diagramme de tâche de l'entité Rover.*

**Description :**
Apres l'initialisation des déférents composants matériels du système par le microcontrôleur, que ce soit interne (GPIO, Timer, USART, I2C…) ou externe (ultrason, boussole, accéléromètre, module wifi), il attend les requêtes client provenant de mobile Android.
Selon la valeur du paramètre direction véhiculé par la requête, on distingue six cas possibles, qui sont répartis comme suit :

- Avancer et reculer : d'abord, le microcontrôleur récupère la distance entre le rover et l'obstacle se trouvant devant ou derrière. Si cette distance est inférieure à vingt alors il arrête les moteurs sinon il ajuste la puissance des moteurs ainsi que le sens de rotation appropriés afin d'avancer ou de reculer, puis il récupère les autres informations (vitesse, orientations) et les envoient au client.
- Pour les deux autres directions (droite, gauche), c'est le même cas avec avancer et reculer juste qu'ici, il ne vérifie pas si y'a d'obstacle vue que les capteurs sont placés a l'avant et l'arrière du Rover.
- Arrêter : il arrêter les quatre moteurs, puis il récupère les données et les envoie au client comme repense.
- Information : dans ce cas il n'y a aucun ajustement à faire sur les moteurs, il suffit juste de récupéré les données et les envoyer au client.

À la fin d'exécution de chaque cas le microcontrôleur retourne à l'attente des requêtes éventuelle de client.

### 4.3.2 Diagramme de taches de l'entité client

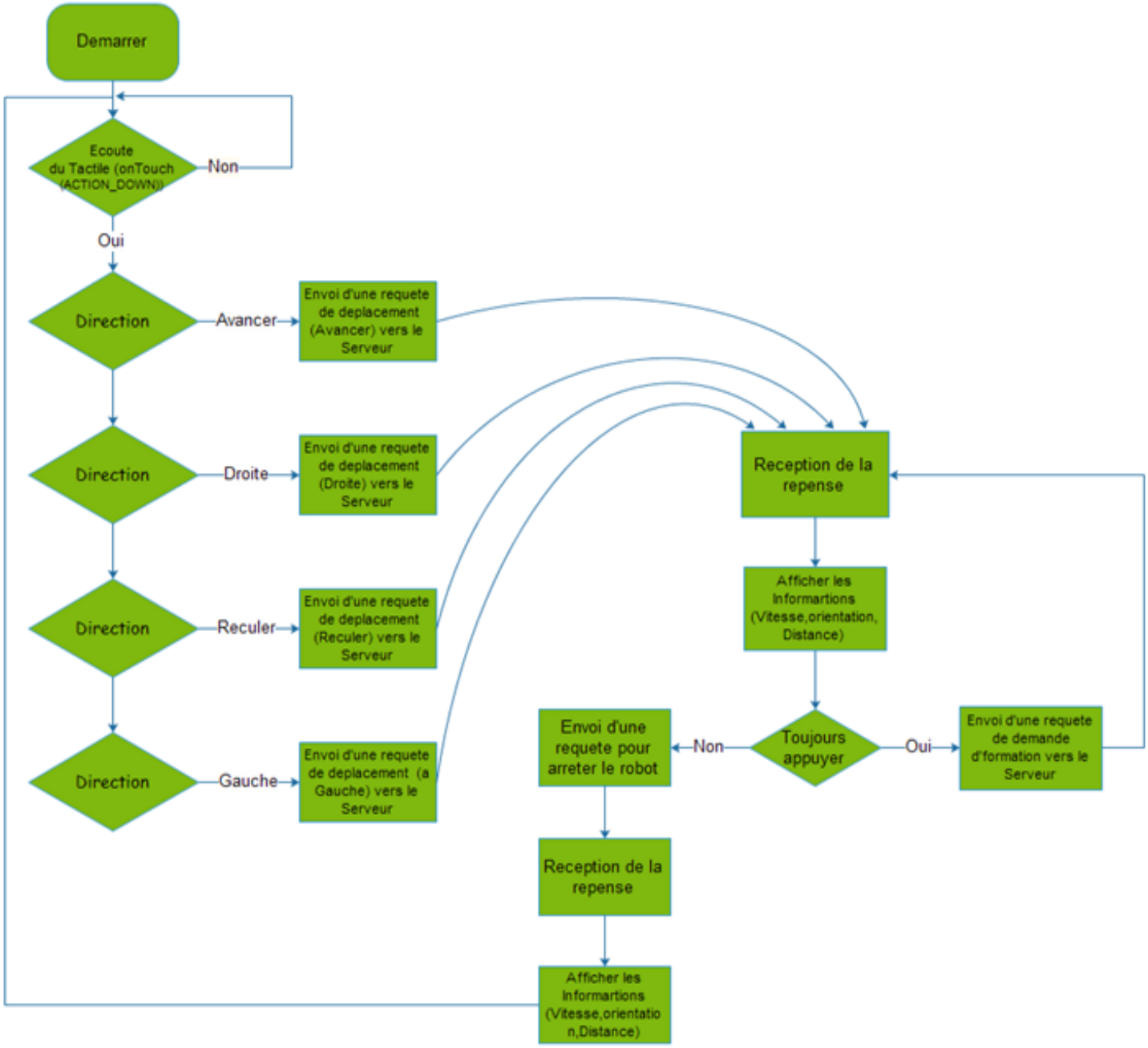



***Figure III.6 :*** *Diagramme de tache de l'entité Client.*

**Description :**

- L'appuie de l'utilisateur sur l'une des quatre directions figurant sur l'écran tactile du Smartphone d'éclanche l'envoie d'une requête vers le microcontrôleur précisant la direction du déplacement ainsi que la puissance du Rover.
- Au retour les informations (distance, vitesse, orientation) contenues dans la repense sont afficher sur l'écran.
- Tant que le doigt d'utilisateur reste appuyé sur l'écran, des requêtes sont envoyée au microcontrôleur chaque 200ms dans le but de demander des informations (distance, vitesse, orientation).

- Si l'utilisateur enlève son doigt de l'écran, cela engendre l'envoie d'une requête vers le microcontrôleur précisant l'arrêt du Rover, et de nouveau l'application Android attend l'appuie de l'utilisateur sur l'écran.

**Conclusion**

La phase de conception était importante pour pouvoir visualiser le fonctionnement de notre système d'une façon abstraite, et de voir de plus près les deux entités qui forme notre système, ainsi que les tâches effectue par chacun d'eux pour accomplir leur mission, et atteindre nos objectifs, et à partir de laquelle nous avons pu passer à la réalisation.

# CHAPITRE IV

# Réalisation
# &
# Mise en œuvre

## Introduction

Ce présent chapitre a pour objectif de présenter la réalisation de notre Rover commandé avec un Smartphone Android via le wifi. Nous entamons ce chapitre par la présentation de l'environnement de développement dont le matériel, et les outils logiciels utilisés. Ensuite, nous allons présenter les fonctions de nos applications.

## 1 Environnement de développement

### 1.1 Environnement matériel

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

- Une carte de développement (STM32F407 Discovery).
- Un double pont H (L293D).
- Un accéléromètre (ADXL345).
- Une boussole numérique (GY-271).
- Deux ultrasons (HC-SR04).
- Un module Wifi (ESP82666).
- Un châssis rover avec quatre moteurs CC.
- Une table d'essai.
- Des câbles métalliques.
- Deux batteries.
- Un Smartphone (SAMSUNG Galaxy S5).

#### 1.1.1 STM32F407 Discovery

C'est une carte de développement (figure IV.1) pour les systèmes embarqués très performante avec beaucoup de possibilités, contenant un processeur ARM Cortex-M4.

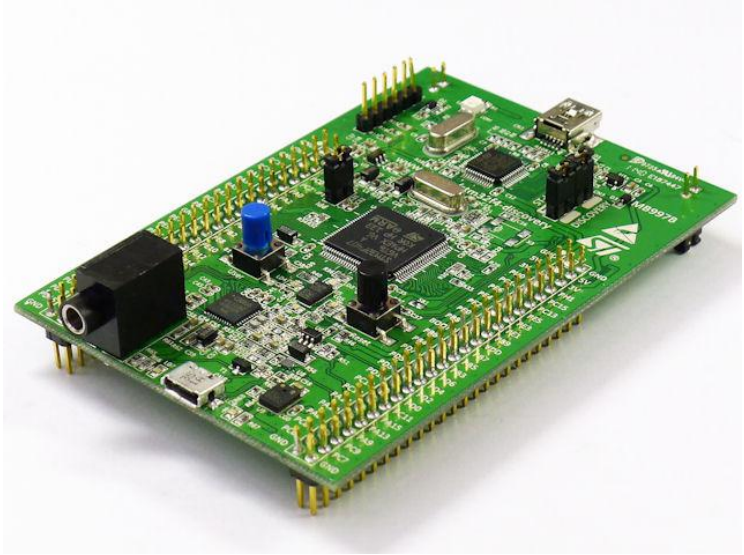


**Figure IV.1 :** STM32F407–Discovery.

##### 1.1.1.1 Caractéristiques et composants

La carte STM32F4 dispose, entre autres, de :

- Un Microcontrôleur stm32f407vgt6 32-bit ARM Cortex-M4F, 1 MB de Flash, 192 KB de RAM.
- Un connecteur ST-LINK/V2 qui permet d'utiliser le connecteur USB d'alimentation pour la programmation et le débogage.
- Un connecteur USB pour l'alimentation 5V.
- Des sorties 3V et 5V pour les applications externes.

- Un accéléromètre LIS302DL ou LIS3DSH ST MEMS 3-axis (sur les dernières versions c'est le LIS3DSH).
- Un micro MP45DT02 pour enregistrer des sons.
- Un audio DAC CS43L22, avec un pilote de haut-parleur class D.
- 4 LEDs, LED3 (orange), LED4 (vert), LED5 (rouge) and LED6 (bleu).
- 2 boutons poussoir (utilisateur et reset).
- Un connecteur USB OTG FS (micro-AB) pour s'interfacer avec des systèmes externes (clé USB, PC, …).
- 80 broches en E/S bidirectionnelle(GPIO), 14 Timers, 6 USART, 3 I2C ...

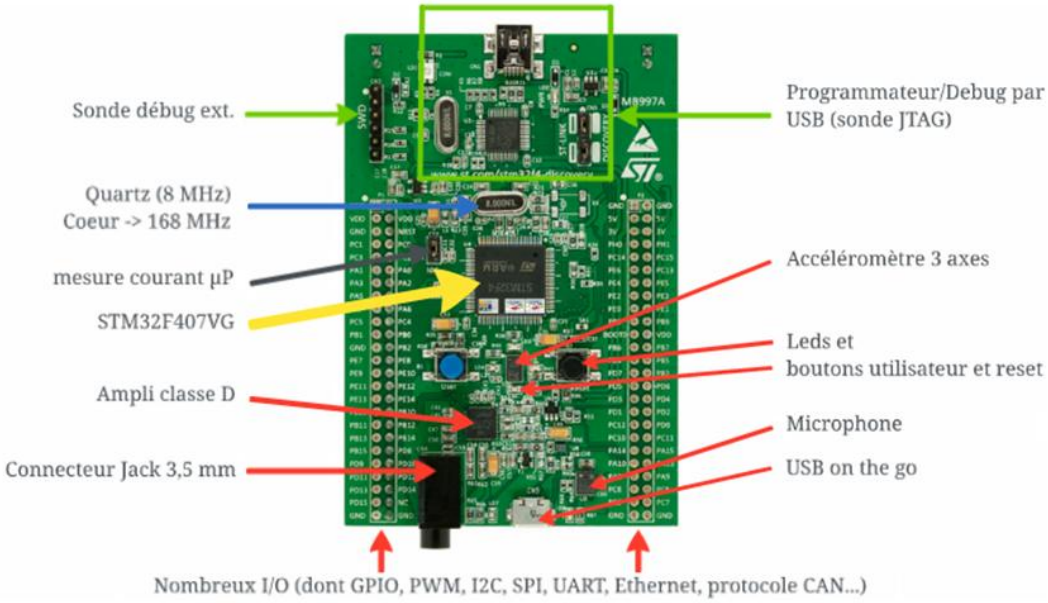



**Figure IV.2 :** Composants de la STM32F407 Discovery.

### 1.1.2 L293D

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.
En raccordant les sorties de façon appropriées, il est possible de constituer deux pont-h. Il est ainsi possible de commander deux moteurs distincts, dans les deux sens et indépendamment l'un de l'autre.
Ce circuit intégré peut être utilisé pour des moteurs CC alimentés jusqu'à 36 Volts, Le circuit peut fournir un maximum de 600mA par canal.

**Figure IV.3 :** Double pont-H L293D.

#### 1.1.2.1 Broches du L293D

La figure IV.4 montre la configuration des broches du L293D.

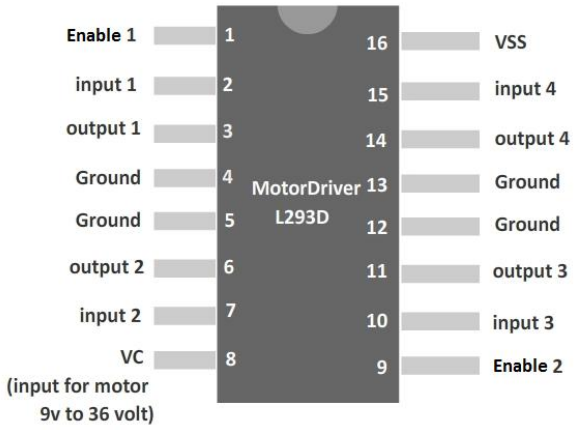



**Figure IV.4 :** Broches du L293D

.

### 1.1.2.2 Description des Broches

Le tableau *IV.1* résume la description des broches de la partie gauche du L293D, pareil pour la partie droite :

| Broche | Nom | Description |
|---|---|---|
| 1 | Enable 1 | Enable1 se raccorde a l'une des sortie de timer de la stm32f4 (le signal PWM_1) pour gérer la puissance des moteurs du coté gauche. |
| 2,7 | Input 1<br>Input 2 | Input1 et Input 2, sont les broches de commande du Pont-H Output1/Output2.<br>Se raccorde a la STM32F4, permet de commander le sens du courant entre Output 1 et Output 2. |
| 3,6 | Output 1<br>Output 2 | Output 1 et Output 2, seront les broches à raccorder à la charge (le moteur). |
| 4,5 | GND | Doit être raccorder à la masse (GND) de la source d'alimentation de puissance ''VS'' et à la masse de la source d'alimentation de la logique ''VSS'' |
| 8 | VS | Alimentation de puissance des moteurs. |
| 16 | VSS | Alimentation de la logique de commande (5V).<br>A raccorder à la borne +5V de la STM32F4. |

***Tableau IV.1** : Descriptions des broches du L293D.*

### 1.1.3 L'accéléromètre numérique ADXL345

L'ADXL345 est un accéléromètre à trois axes, avec une mesure de la résolution élevée (13 bits) jusqu'à ±16 g. Les données de la sortie numérique, sur 8 bits sont concaténées pour former des valeurs de 16 bits. Ces données sont accessibles via une interface SPI (3 ou 4 fils) ou une interface numérique I2C.


**Figure IV.5 :** Accéléromètre ADX345

. .

### 1.1.3.1 Caractéristiques de L'ADXL345

• Tension d'alimentation de 2 à 3,6 V en CC.
• Puissance très faible : 40 uA en mode Mesure, 0,1 uA en veille à 2,5 V.
• Interfaces SPI et I2C.

### 1.1.3.2 Configuration des broches de L'ADXL345

- GND: masse.
- VCC: 2V-3,6V CC.
- CS : relier a VCC pour faire fonctionner le capteur.
- SCL: (signal d'horloge).
- SDA: (signa de donnée).

### 1.1.4 la boussole numérique GY-271

C'est une boussole numérique trois axes capable de détecter des faibles champs magnétiques. Ce capteur peut être intégré dans de petits projets tels que les drones et les systèmes de navigation de robot.

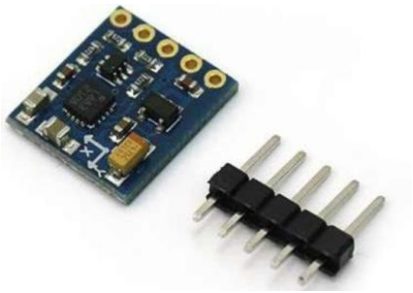


**Figure IV.6:** Boussole GY-271.

### 1.1.4.1 Caractéristiques du GY-271

- Alimentation 3V-5V CC.
- Chipset HMC5883L.
- Communication via le protocole I2C.
- Plage de mesure: ± 1.3-8 Gauss.
- Dimensions 14,8 x 13,5 x 3,5 mm.

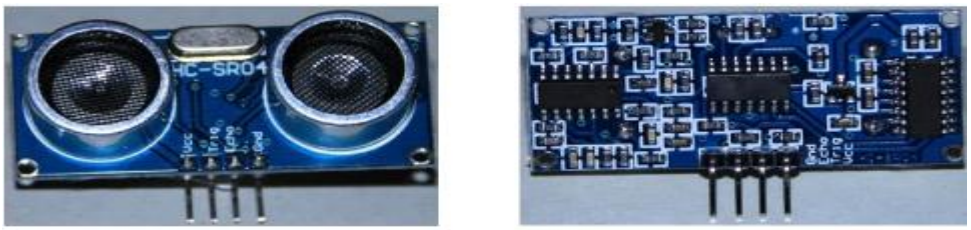
### 1.1.4.2 Configuration des broches du GY-271

- VCC: 3V-5V CC.
- GND: masse.

- SCL: (signale d'horloge).
- SDA: (signale de donnée).
- DRDY: non connecté.

### 1.1.5 Le capteur à ultrason HC-SR04

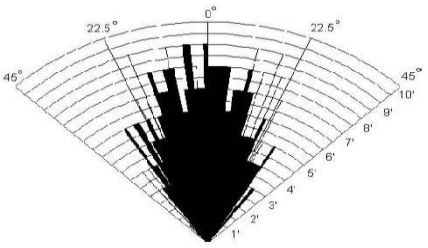
Le capteur HC-SR04 utilise les ultrasons pour déterminer la distance d'un objet. Il offre une excellente plage de détection sans contact, avec des mesures de haute précision et stables. Son fonctionnement n'est pas influencé par la lumière du soleil ou des matériaux sombres, bien que des matériaux comme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




Le capteur de face  le capteur de dos

***Figure IV.7** : Ultrason HC-SR04.*

#### 1.1.5.1 Caractéristique du HC-SR04




*Practical test of performance, Best in 30 degree angle*

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

#### 1.1.5.2 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.

### 1.1.6 Le module Wifi ESP8266

L'esp8266 est un module de communication permettant d'instaurer une liaison Wifi complète et autonome avec le microcontrôleur. Il utilise une liaison série TX/RX pour recevoir et envoyer des données, et se comporte donc comme un hôte pour les applications Wifi. Il peut également servir d'interface wifi (une carte réseau Wifi) pour un processeur ou microcontrôleur. Pour cela, le processeur lance des commandes spéciales via le port série dites commandes AT.

***Figure IV.8** : Module wifi ESP8266*

### 1.1.6.1 Spécifications techniques de L'ESP8266

- Protocole TCP/IP intégré
- Wifi direct (P2P), soft-AP
- 802.11 b/g/n
- Processeur 32-bit faible puissance intégré, pouvant être utilisé comme un processeur d'application
- Dimensions : 21,1 x 13,2 mm

### 1.1.6.2 Description des broches de L'ESP8266

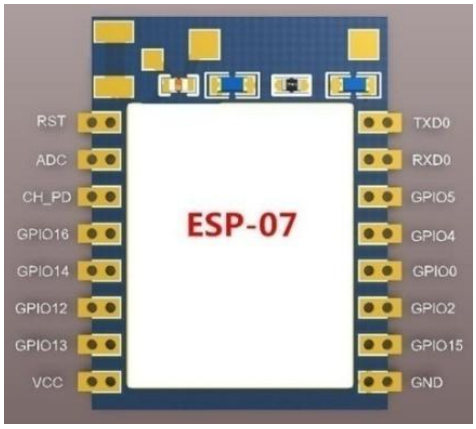



**Figure IV.9 :** Broches de l'ESP8266-07.

- TXD0 : C'est la sortie des données de l'ESP8266.
- RXD0 : entrée des données.
- CH_PD ("chip power down"): doit être à 3,3 volts pour permettre le fonctionnement du module.
- RST ("reset") : lorsque cette broche est reliée à la masse, il en résulte un reset (redémarrage) du module.
- Vcc : alimentation du module a 3,3 V.
- GND : la masse.
- GPIOs : peuvent être utilisées pour commander d'autres équipements.
- GPIO0 : mise à 3,3 volts en utilisation normale (branchée à GND lorsqu'on veut mettre à jour le firmware de l'ESP8266).

### 1.1.7 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 IV.10 :** Châssis rover 4WD.

### 1.1.8 Batteries

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

- Une batterie Power Bank d'une capacité de 8000 mAh, sortie (5v) pour alimenter l'ensemble de l'électronique du system embarquée.
- Une batterie de 9,6v pour alimenter les Moteurs.

Batterie Power Bank.

Huit batteries de 1.2 V monter en série.

**Figure IV.11 :** Batteries d'alimentation du système.

### 1.1.9 Table d'essai

Pour éviter de souder les différents composants on a utilisé une table d'essaie.

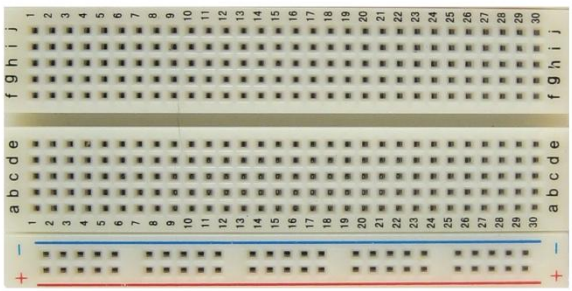


**Figure IV.12 :** tables d'essai.

### 1.1.10 Câbles métalliques

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

**Figure IV.13**: Câbles métalliques.

### 1.1.11 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
***Norme Wifi*** : 802.11n/ac

***Figure IV.14*** *: SAMSUNG Galaxy S5.*

## 1.2 Environnement cible

Le programme C du Rover est exécuté sur la STM32F407 Discovery mais peut être facilement porté sur d'autre microcontrôleur STM32 grâce à la STM32cube.
L'application Android est destinée pour tous les Smartphones équipés d'un système Android 4.0.3 Toutefois elle est utilisable sur les versions récentes d'android.

## 1.3 Environnement logiciel

Pour pouvoir réaliser notre projet dans de bonnes conditions, il est nécessaire de préparer l'environnement de travail. Nous avons effectué différentes tâches que nous décrivons dans ce qui suit :

- Installation de ***Keil uVision4*** (Pour programmer avec le langage C sur la stm32f4).
- Installation de la ***STM32Cube*** qui est une plate-forme de développement pour les microcontrôleurs STM32.
- Installation de la ***STM32CubeMX*** qui est un outil graphique qui permet de configurer le microcontrôleur.
- Installation de ***PuTTY*** (émulateur de terminal) pour des tests de débogage.

- Installation d'***Eclipse*** (comme pour le développement d'application java classique) pour le développement Android.
- Le Pulgin Android pour Eclipse ***ADT*** (Android Developer Tools), qui adapte Eclipse au développement d'application sous Android.
- Installation ***SDK*** (Software Development Kit) qui va contenir tous les outils nous permettant la construction de notre application sur Smartphone.
- La bibliothèque ***Client http Apache*** pour la communication (mode client-serveur) entre le Smartphone et le Rover.

### 1.3.1 Keil uVision

Keil - µvision est un environnement de développement pour systèmes embarqués fournissant un ensemble d'outils tels que:

- Un éditeur de code (C, C++, assembleur) avec coloration syntaxique.
- Un compilateur croisé pour architectures ARM7, ARM9, Cortex-M, C166, 8051 et 251.
- Un simulateur supportant plusieurs processeur et cartes SOC.
- Un débogueur intégré fournissant l'état du processeur et des périphériques des cartes supportées en mode simulation.
- Un ensemble de fichiers de démarrage (startup) pour accélérer le développement des applications

Son interface principale est donnée dans la figure *IV.15*:

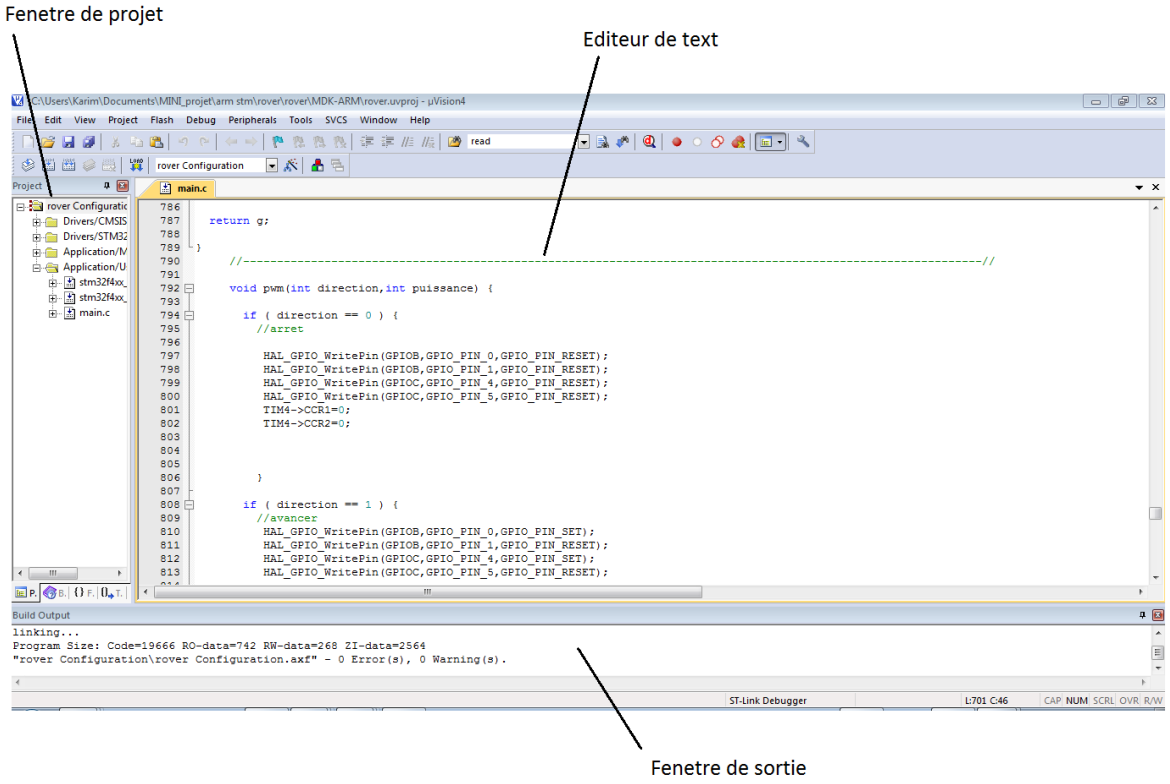



**Figure IV.15** : Interface principale de Keil.

### 1.3.2 STM32Cube & STM32CubeMX

C'est une nouvelle plate-forme de développement pour les microcontrôleurs STM32, STM32Cube comprend le configurateur graphique STM32CubeMX, un générateur de code C pour initialisation qui guide les utilisateurs étape par étape, et un riche ensemble de composants logiciels embarqués qui évitent d'intégrer des logiciels provenant de sources multiples. Le logiciel comprend une nouvelle couche d'abstraction matérielle (HAL) qui simplifie le portage entre deux microcontrôleurs STM32. En réunissant sur un seul boîtier tous les composants logiciels génériques nécessaires pour développer des applications sur microcontrôleurs STM32, cette plate-forme élimine la tâche complexe que représente l'édition des liens de dépendance entre les différents composants logiciels. L'outil STM32Cube fournit des centaines d'exemples d'utilisation, ainsi qu'un mécanisme de mise à jour qui assure un accès rapide et efficace aux versions les plus récentes du logiciel.

Son interface principale est donnée dans la figure *IV.16*:

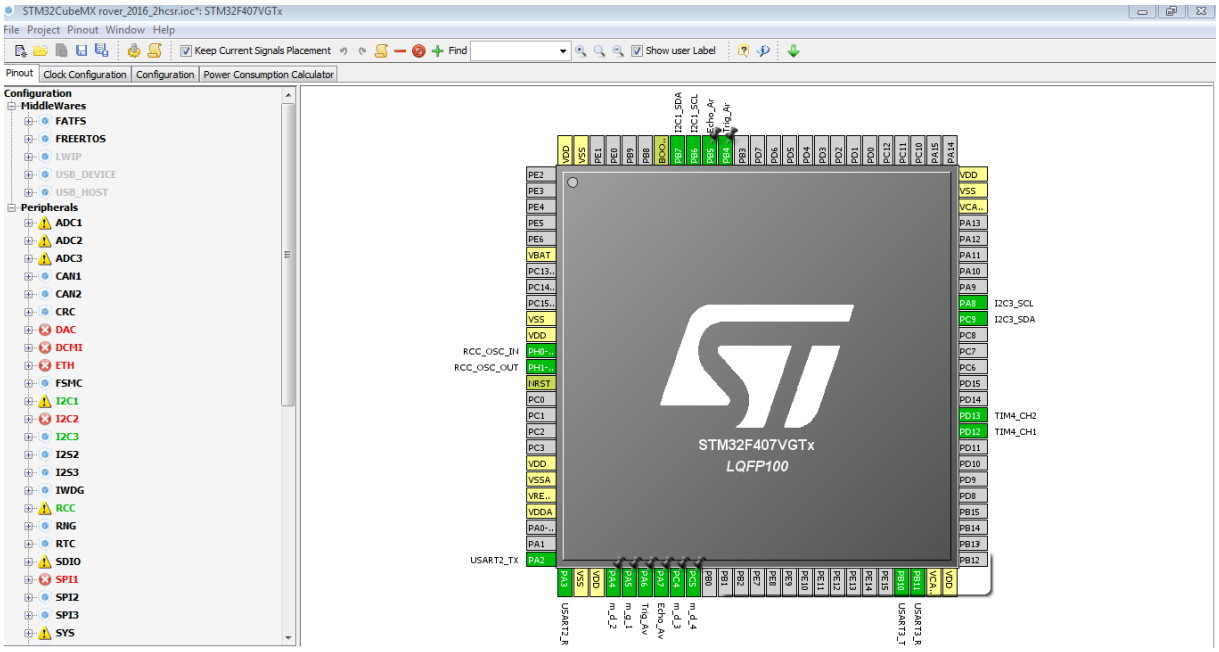



**Figure IV.16** : Configuration des broches de la STM32F4 avec l'outil STM32CubeMX.

### 1.3.3 Putty

Putty est un émulateur de terminal doublé d'un client pour les protocoles SSH, Telnet, rlogin, et TCP brut. Il permet également d'établir des connexions directes par liaison série.

Nous l'avons utilisé pour afficher les échanges de communication entre l'application Android et le Rover ainsi que les données capturées par les différents capteurs.

Son interface principale est donnée dans la figure IV.17 :

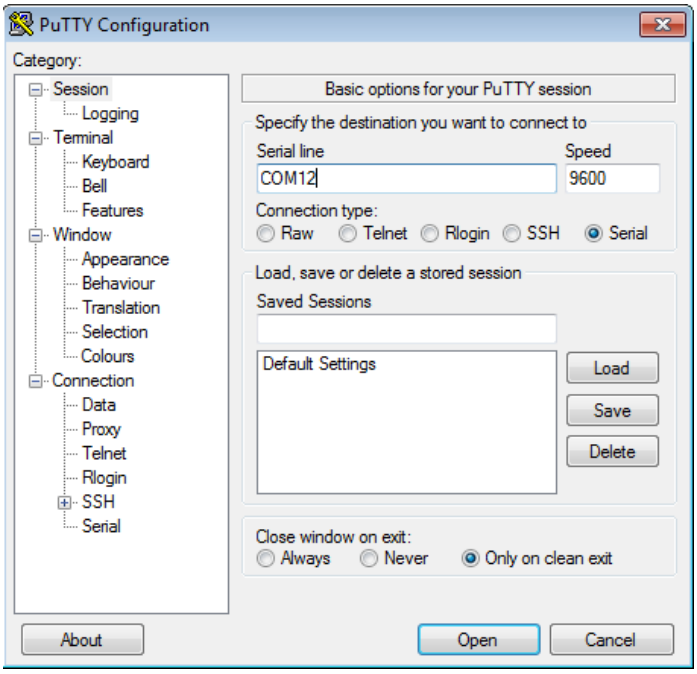



**Figure IV. 17** : Interface principale de Putty.

### 1.3.4 Eclipse

Eclipse est un IDE qui permet de programmer dans différents langages grâce à ses nombreux plug-ins et notamment le plug-in d'Android. Une interface spécifique permet de gérer des fichiers java et de compiler ses programmes. Les fichiers sont organisés selon une arborescence qui correspond aux paquetages java définis.
L'analyse syntaxique permet de mettre en valeur les mots clés dans les fichiers java. Eclipse dispose aussi d'un système d'auto complétion des fonctions, de détection des erreurs syntaxiques en temps réel sans oublier un système de débogage permettant d'exécuter ses programmes pas à pas.
Pour le développement de notre application, nous avons choisi Eclipse INDIGO, Son interface principale est donnée dans la figure IV.18 :

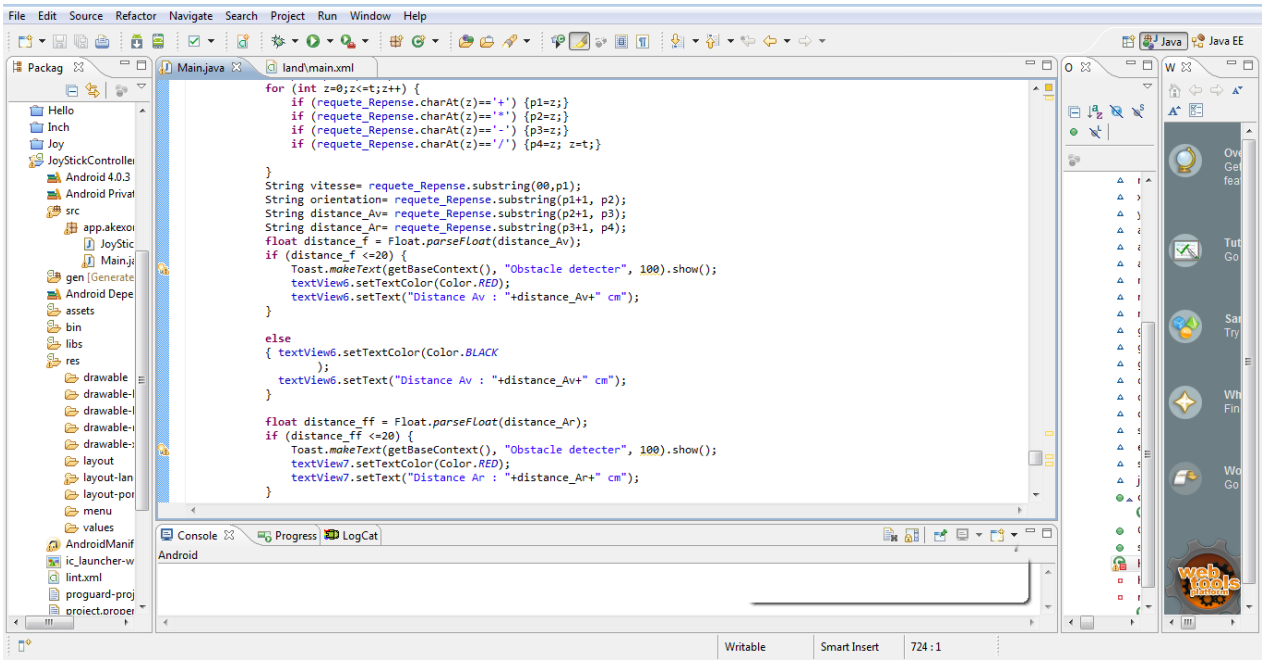



**Figure IV.18** : Interface principale d'Eclipse.

### 1.3.5 plugin ADT

Afin de développer sous Android, nous avons installé le plugin Android qui ajoutera à Eclipse les fonctionnalités spécialisées dans le développement sous Android.
Le plugin Android Développent Tools (ADT) est très complet et surtout très pratique il offre l'accès à de nombreuses fonctionnalités qui aident à développer des applications Android.

### 1.3.6 SDK Android

Le SDK (Software Developpement kit) est un ensemble d'outils de développement qui permet de créer de nouvelles applications. Son installation se fait en lançant le SDK-Manager.exe, téléchargé du site officiel d'Android.

### 1.3.7 Emulateur Android

Le SDK Android inclut un émulateur d'appareil mobile, c'est un dispositif mobile virtuel qui s'exécute sur l'ordinateur et permet de développer et de tester des applications Android sans l'aide d'un dispositif physique.
Lorsque l'émulateur est lancé, il nous permet d'interagir avec le périphérique mobile émulé. Nous utilisons également le pointeur de la souris pour "toucher" l'écran tactile et nous pouvons utiliser certaines touches du clavier pour invoquer certaines touches sur l'appareil.
L'émulateur Android imite toutes les fonctionnalités matérielles et logicielles d'un dispositif mobile typique, tel que la lecture des fichiers audio et vidéo, stockage des données. Sauf qu'on ne peut accéder à la carte réseau chose qui est indispensable pour le fonctionnement de notre application vu que le rover se commande via le wifi, pour cela nous n'avons pas utilisé cet outil. On est amené à exécuter l'application directement sur notre Smartphone physique en le reliant à la machine avec un câble USB et faire le choix lors du lancement de
L'exécution comme le montre la figure IV.19 :

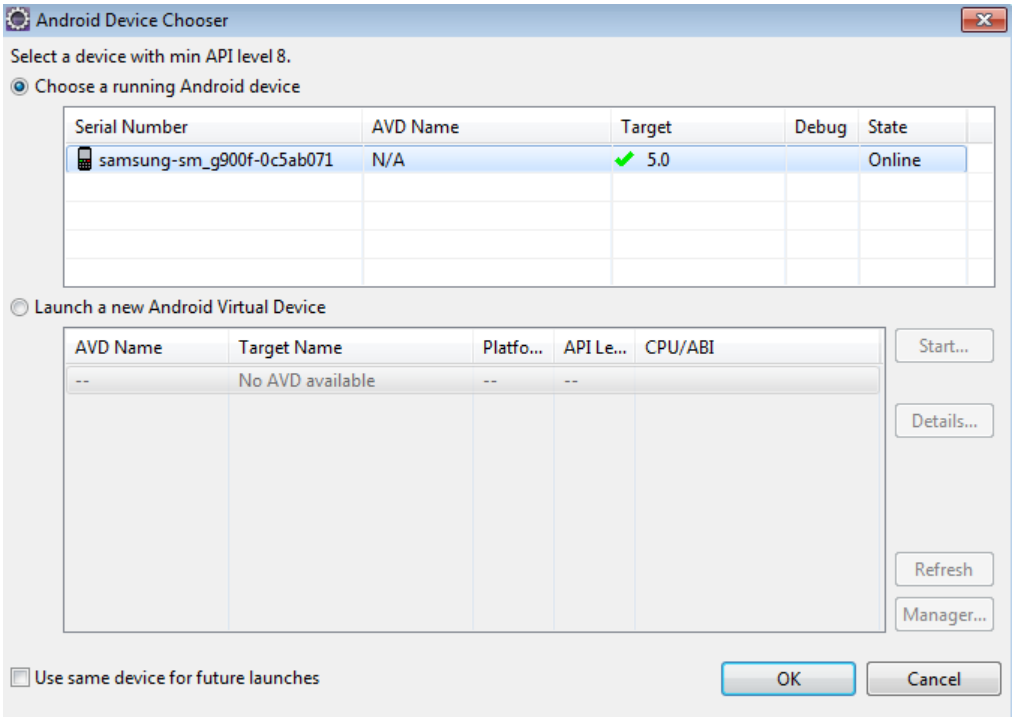



**Figure IV. 19**: Sélection du mobile lors de lu lancement de l'exécution.

### 1.3.8 La bibliothèques Client http de Apache

HttpClient est une bibliothèque Java qui peut être d'intérêt pour quiconque construire des applications clientes HTTP tels que les navigateurs Web, les clients de services Web ou des systèmes qui exploitent ou étendent le protocole HTTP pour la communication distribuée.

## 2 Fonctionnement

### 2.1 Application Android

Nous avons développée une application pour Smartphone Android qui comporte une activité principale nommée MainActivity composé d'un joystick virtuel pour commander le Rover et d'un écran pour afficher les différentes informations (vitesse, orientations, distance) capturées par le Rover.
Nous avons aussi ajouté deux permissions :

```
<uses-permissionandroid:name="android.permission.INTERNET"/>
<uses-permissionandroid:name="android.permission.ACCESS_NETWORK_STATE"/>
```

La première permission sert à accéder à internet pour permettre d'envoyer des requêtes.
La deuxième permission sert à lire l'état du Wifi et l'afficher sur l'activité principale.

L'interface de l'application est donnée dans la figure IV.20 :

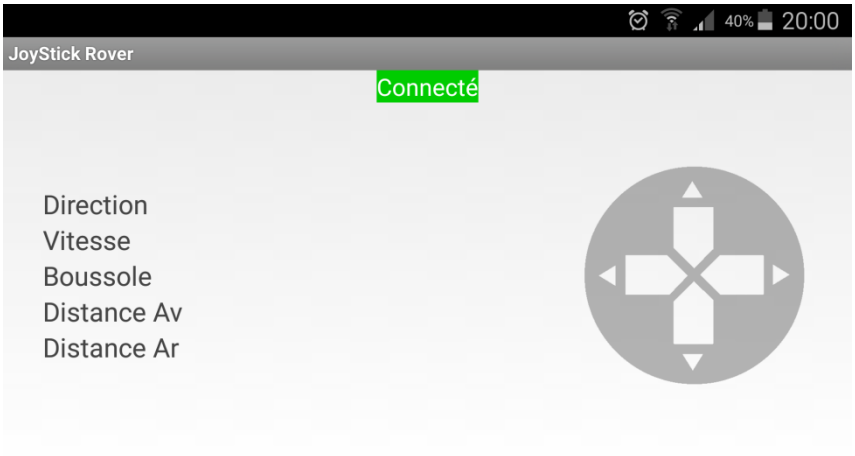



**Figure IV. 20**: Interface de l'application Android.

### 2.2 Programme C pour la STM 32F4

Le programme C a pour taches de :

1. initialiser les composants internes de la STM32F4 (Timer, USART, I2C,…) et les différent modules externes (ADXL345, GY-271, ESP8266,…).
2. Ecouter les messages du client (si l'utilisateur a manipulé le joystick sur son Smartphone).
3. exécuter la commande du client (déplacer le Rover).
4. Récupère les données capturées par les différents capteurs.
5. Envoyé ces données au client (Smartphone Android).
6. Répéter depuis 2 à 5.

### 3 Etablissement de la communication entre le Smartphone et le Rover

Pour établir cette communication, nous avons configuré l'ESP8266 en mode **AT commands**, en effectuant une mise à jour du firmware. La version du firmware choisie est « v0.9.5.2 AT Firmware.bin ».

La figure IV.21 montre le branchement de l'ESP8266 à la STM32F4:

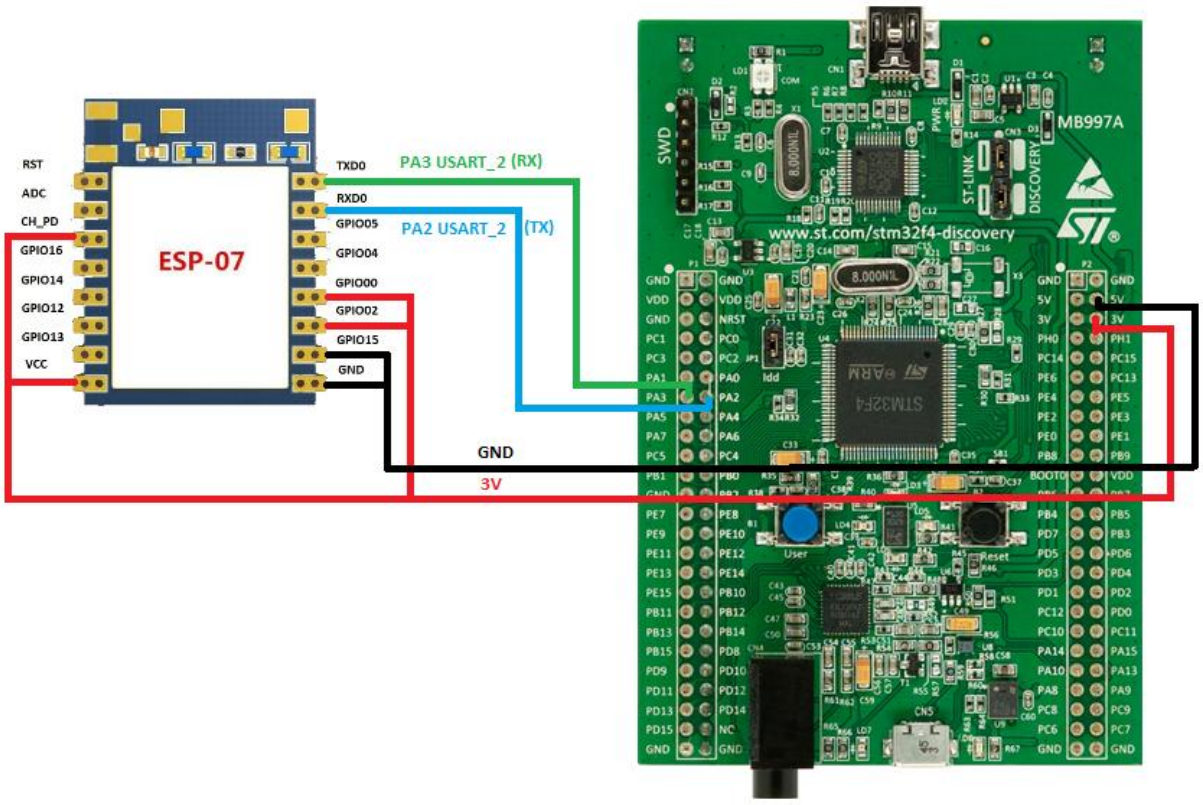



**Figure IV.21:** Branchement de L'ESP8266 au microcontrôleur STM32F4.

- **Initialisation du module**

L'initialisation du module ce fait en lui envoyant une suite de commandes sur la ligne TX l'une derrière l'autre dans un ordre précis, comme suite :

- AT+CWMODE=2 : Le module fonctionne selon le mode AP.
- AT+CIFSR : Obtenir l'adresse IP.
- AT+CIPMUX=1: TCP/UDP connexion multiple.
- AT+CIPSERVER=1,80 : Activer le serveur sur le port 80.

La figure IV.22 présente le résultat d'exécution du programme d'initialisation de l'ESP8266 sur le terminal putty :

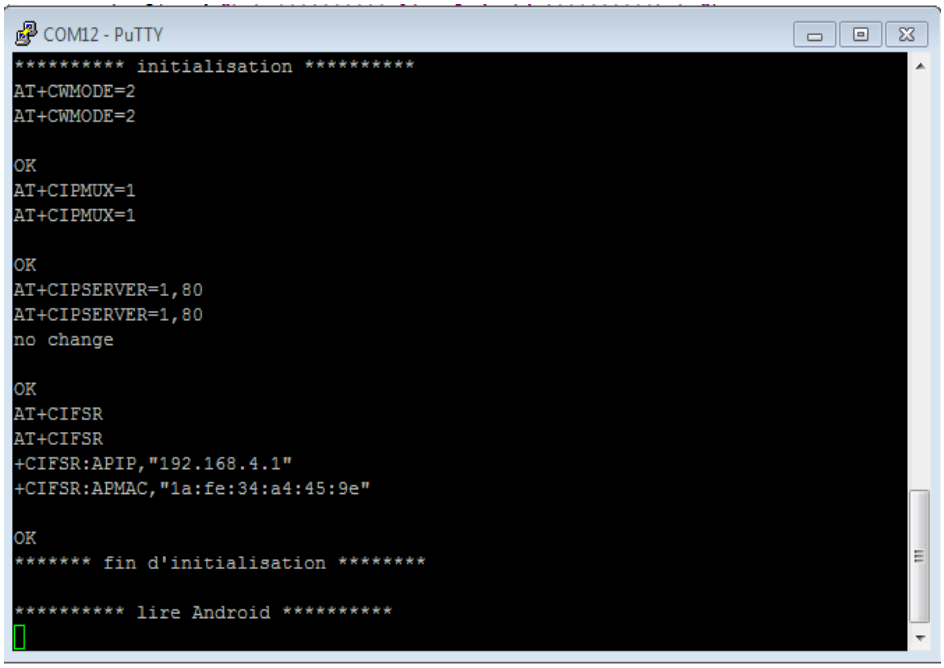



**Figure IV. 22** : Initialisation de l'ESP8266.

Une fois le module initialisé le nom du réseau est afficher sur la listes des réseaux wifi disponible du Smartphone, l'utilisateur doit saisir un mot de passe pour se connecter au Rover.

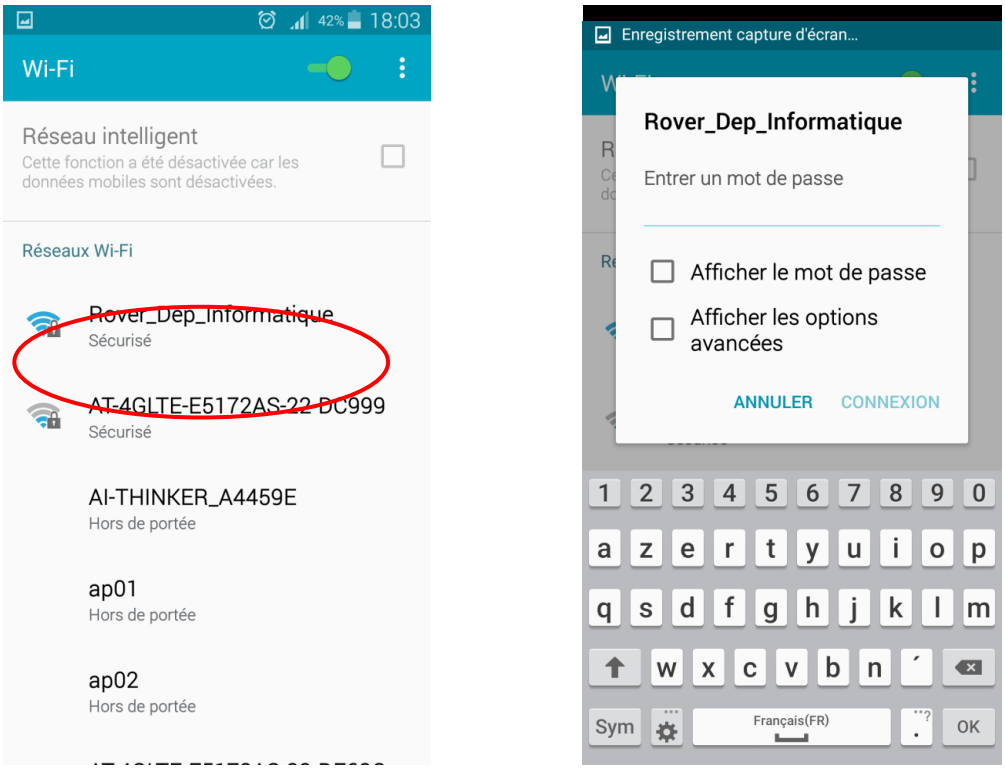



**Figure IV. 23** : Connexion au rover.

➢ **Quelques commandes utiles :**

- AT+CIPSEND=<id>, <t> : Cette commande permet d'envoyer une chaine de caractère d'une taille=t sur le canal numéro=id,
- AT+CIPCLOSE=<id> : Fermer le canal dont le numéro=id.

- AT+CIPSERVER=0 : Fermer la connexion.

### 4 Les Fonction de l'application Android

Au lancement de l'application l'interface montrée sur la figure IV.24 se présente à l'utilisateur :

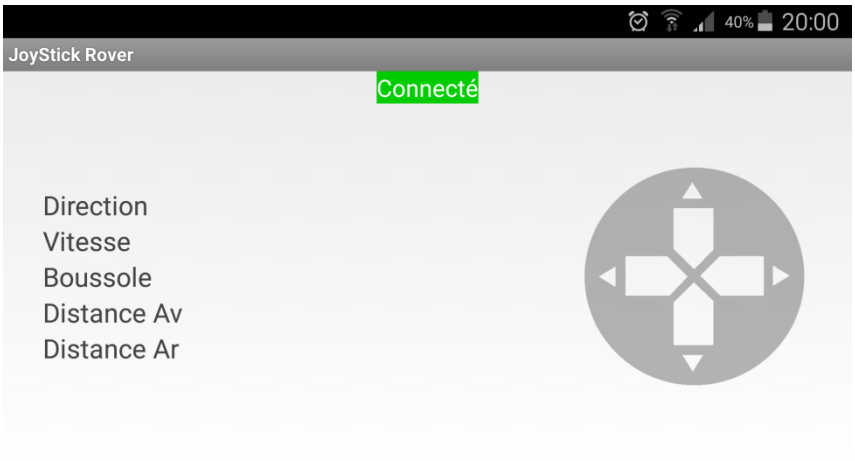



**Figure IV. 24**: Interface de l'activité principale.

Parmi les fonctions proposées par l'application :

1. ***Un joystick virtuel :***

   Ce joystick permet à l'utilisateur de commander le rover (avancer, tourner à droite, tourner à gaucher, reculer) toute en contrôlant la puissance des moteurs, son fonctionnement est décrit dans ce qui suit :

   - L'utilisateur doit toucher le joystick au centre avec son doigt, pour faire bouger le Rover, il doit glisser son doigt sur l'une des quatre directions possibles. Si l'appui est maintenu et, en fonction de la profondeur du glissement, Le client déclenche l'envoi d'une requête depuis le Smartphone. Le rover reçoit cette requête grâce à L' ESP8266 qui la retransmet au microcontrôleur sur la liaison série.
     La requête contient un couple (paramètre, valeur),la valeur est un entier composé de deux chiffres, le premier chiffre précise la direction (avancer, tourner à droite, tourner à gaucher, arrière), le deuxième précise la puissance (le rapport du signal PWM) reparti sur trois vitesse.
     Parmi les valeurs possibles on a :

?paramètre=chiffre1chiffre2

1 : Avancer
2 : Tourner à droite
3 : Reculer
4 : Tourner à gauche

1 : Puissance de 50%.
2 : Puissance de 75%.
3 : Puissance de 100%.

La figure IV.25 montre l'exemple d'une requête de déplacement envoyé depuis le Smartphone vers le Rover :

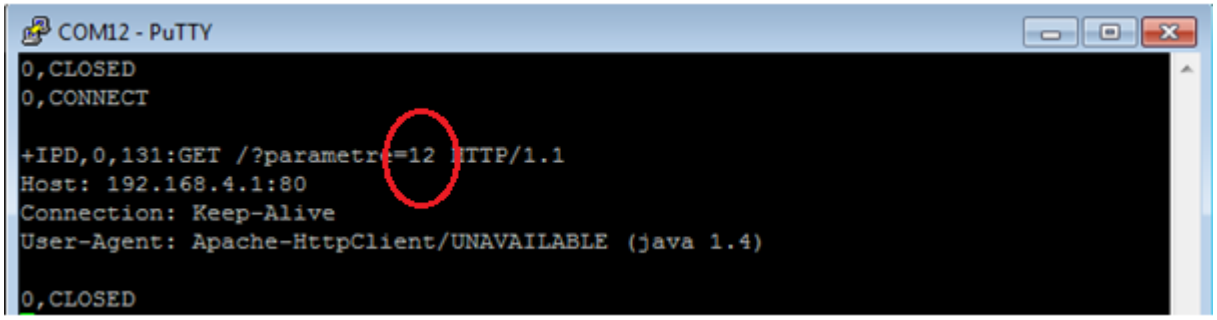



**Figure IV.25**: Exemple d'une requête de déplacement.

- Pour arrêter le rover l'utilisateur a deux choix :

  - Soit enlever son doit de l'écran tactile.
  - Ou bien glisser son doit vers le centre du joystick.

Cela est accompagné de l'envoi d'une requête stop avec un paramètre d'une valeur de 00

La figure IV.26 montre l'exemple d'une requête stop envoyé depuis le Smartphone vers le Rover :

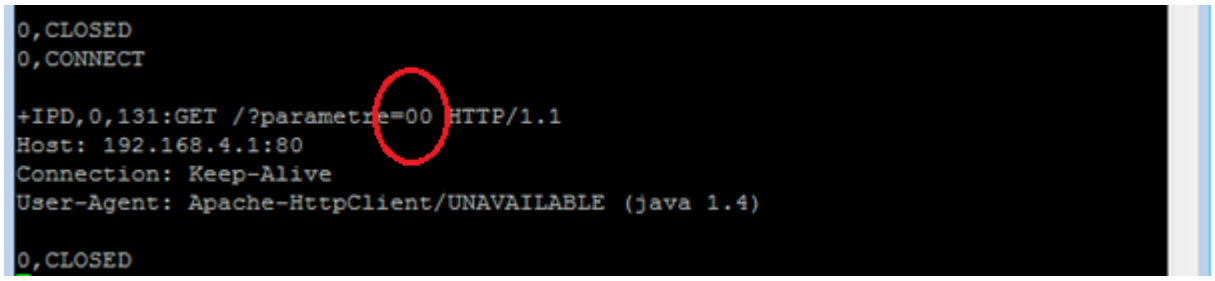



**Figure IV. 26** : Requête Stop.

- Tant que l'utilisateur reste appuyé sur le joystick, des requêtes dites requêtes d'informations sont envoyés vers le Rover chaque 250msCes requêtes ont pour but de demander au Rover de capturer des données (vitesse, orientation, distance) et les renvoyer ver le Smartphone afin de

les afficher sur l'activité principale. La requête d'envoi d'information contient un paramètre codifié par la valeur 99.

La figure IV.27 montre l'exemple d'une requête d'information envoyée depuis le Smartphone vers le Rover :


**Figure IV.27**: Requête d'information.

2. ***Affichage des informations capturées par le rover***

A chaque fois qu'une requête est envoyée depuis le Smartphone vers le Rover, celui ci répond avec une requête contenant des informations (vitesse, orientation, distance) vers le Smartphone afin de les afficher sur l'activité principale en utilisant des textView.

La figure IV.28 montre l'interface sur laquelle sont affiché les données capturées par le Rover :

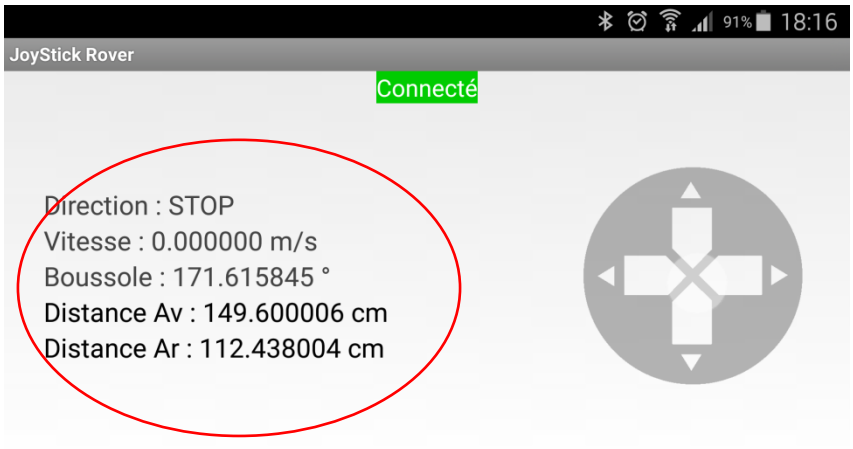



**Figure IV.28**: Interface d'affichage de données capturées par le Rover.

Si la distance captée par l'ultrason placé devant le robot est inférieur à 20 cm un message d'alerte s'affiche sur l'activité et la couleur d'affichage du champ distance deviens rouge (pareil pour l'ultrason placé à l'arrière du robot). Comme la montre la figure IV.29 :

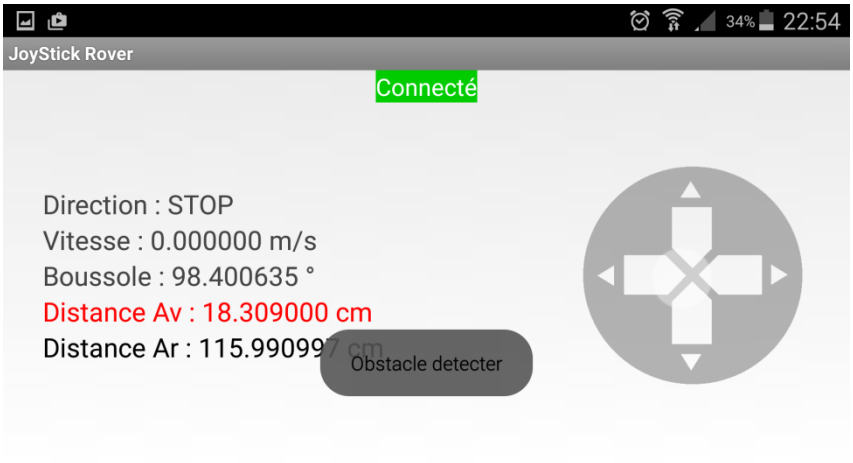



**Figure IV.29** : Affichage du message obstacle détecté.

## 5 Les fonctions du Rover

### 5.1 deplacement

Lors de la réception d'une requête par le microcontrôleur, la routine chargée d'analyser ces requêtes extrait le couple (paramètre, valeur) selon les deux chiffres qui composent la valeur. La routine de déplacement ajuste la puissance des moteurs et le sens de rotation comme suit :

- **Ajuster la puissance des moteurs :**
  Le Timer fourni deux signaux PWM (PWM1 et PWM2) avec le même rapport reliés respectivement aux entrées enable1 et Enable2 du composant L293D.

  La figure IV.30 et Le tableau IV.2 montrent le branchement des PWMs vers le L293D :

| STM32F4 | L293D |
|---|---|
| PD13 - Timer_4 PWM_1 | Enable 1 |
| PD12 - Timer_4 PWM_2 | Enable 2 |

***Tableau IV.2** : Branchement des PWMs vers le L293D.*

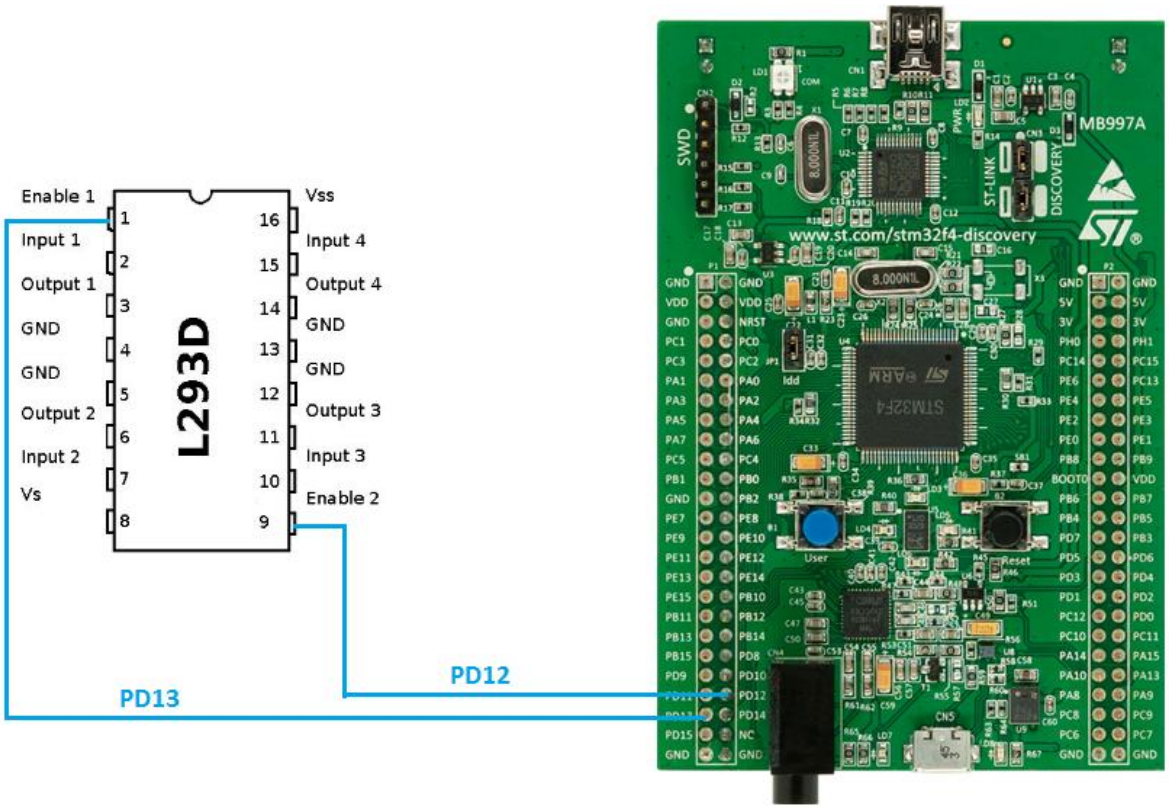



**Figure IV.30**: Branchement des PWMs vers le L293D.

- **Déterminer le sens de rotation des moteurs :**

  Notre Rover a le même comportement qu'un robot à chenilles, donc chaque deux roues du même coté (droite, gauche) ont le même comportement.

  Pour déterminer le sens de rotation des roues on relier quartes GPIOs programmé en sortie vers les quatre entré (Input1, Input2, Input3, Input4) du double pont L293D, ces quatre GPIOs sont divisé en deux groupe de deux, deux GPIOs pour le coté gauche et les deux autres pour le coté droite.

  Les tableaux et la figure ci-dessous montrent la logique de commande des moteurs du Rover en utilisant ces quartes GPIOs :

| Partie gauche | | |
|---|---|---|
| **PA5 - Input 1** | **PA4 - Input 2** | **Fonction** |
| 0 | 1 | Tourne dans le sens d'une horloge |
| 1 | 0 | Tourne dans le sens contraire |
| 0 | 0 | Stop |
| 1 | 1 | Stop |

***Tableau IV.3** : Logique de commande des moteurs de la partie gauche.*

| Partie Droite | | |
|---|---|---|
| **PC4 - Input 3** | **PC5 - Input 4** | **Fonction** |
| 0 | 1 | Tourne dans le sens d'une horloge |
| 1 | 0 | Tourne dans le sens contraire |
| 0 | 0 | Stop |
| 1 | 1 | Stop |

***Tableau IV.4*** *: Logique de commande des moteurs de la partie droite.*

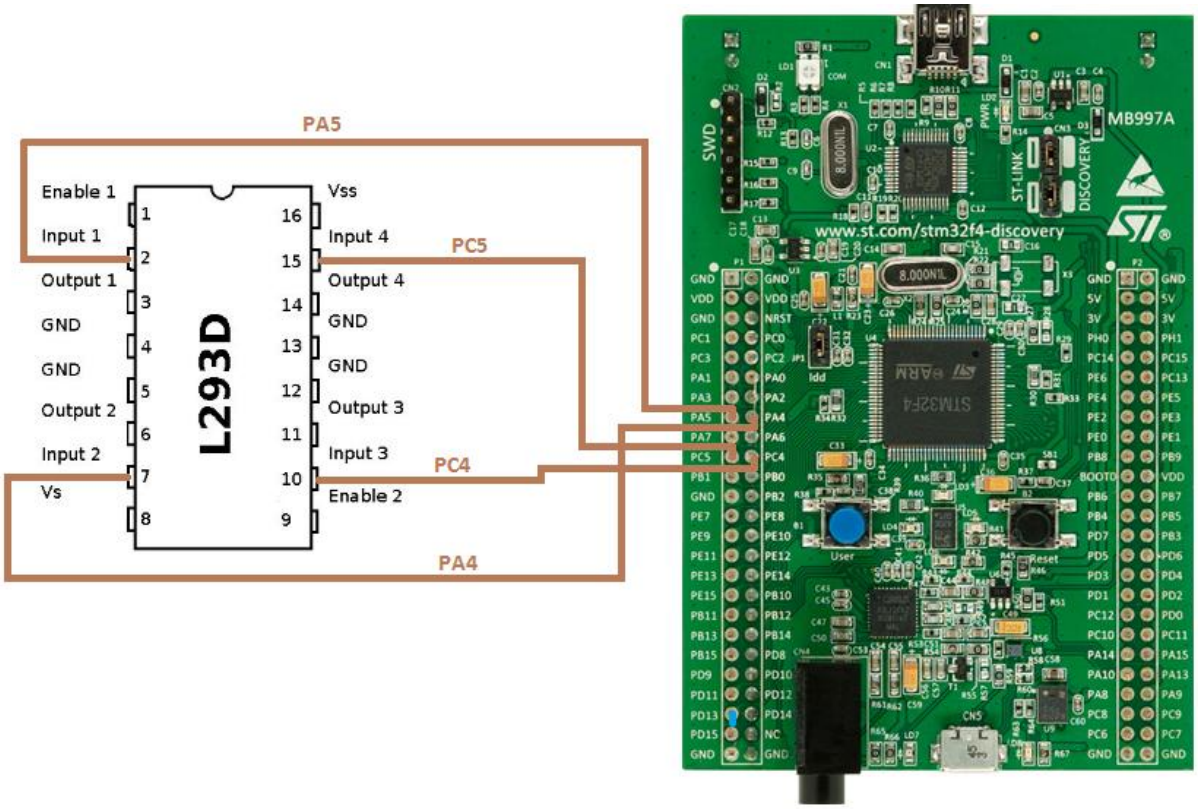



***Figure IV.31*** *: Logique de commande des moteurs.*

Il reste qu'a relier les sorties du L293D vers les quatre moteurs CC du Rover, alimenter le composant L293D et les moteurs, comme le montre la figure IV.32 :

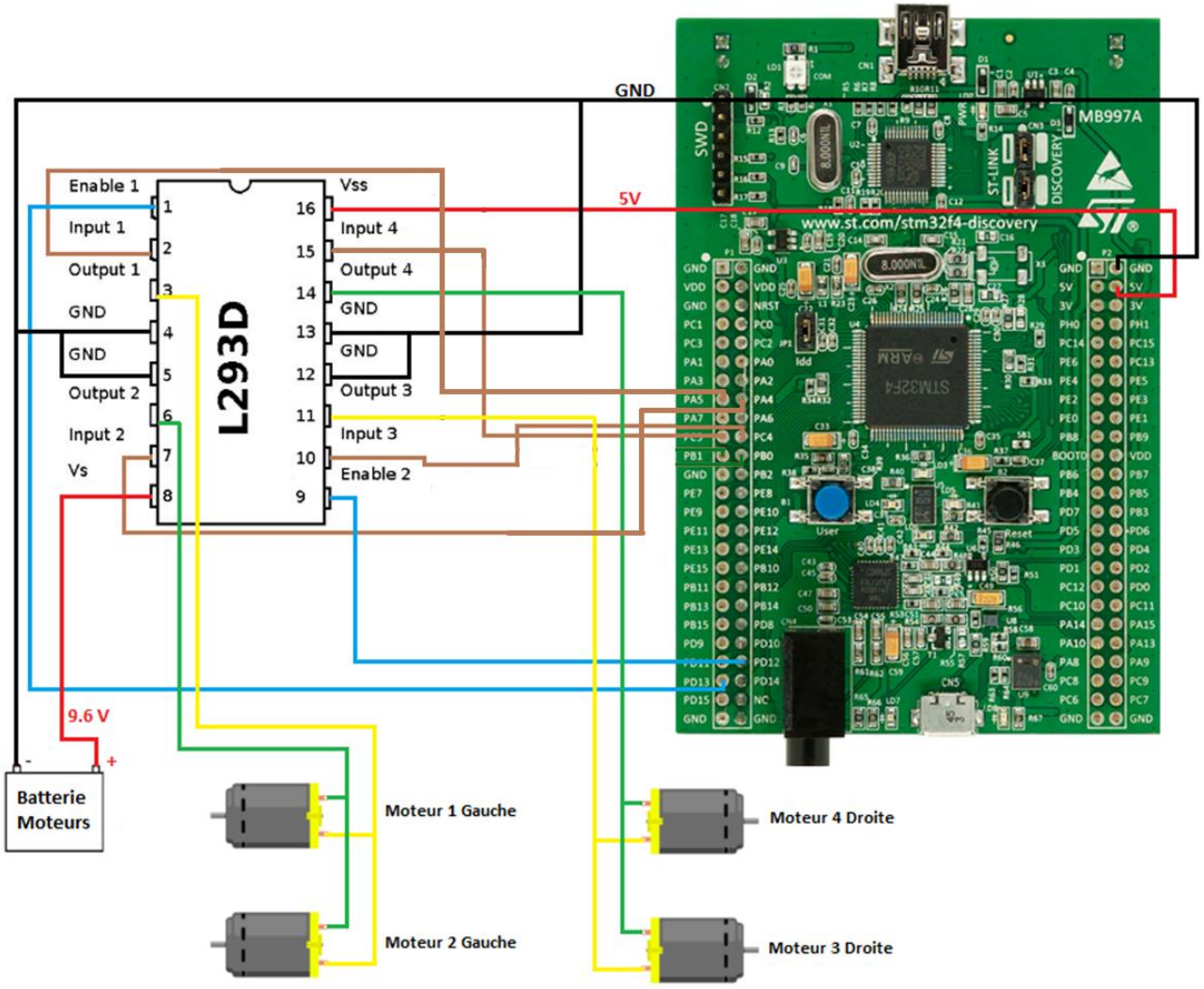



**Figure IV.32**: Schéma de câblage du L293D avec la STM32F4.

La figure IV.33 montre le comportement du rover selon le sens de rotation des roues :

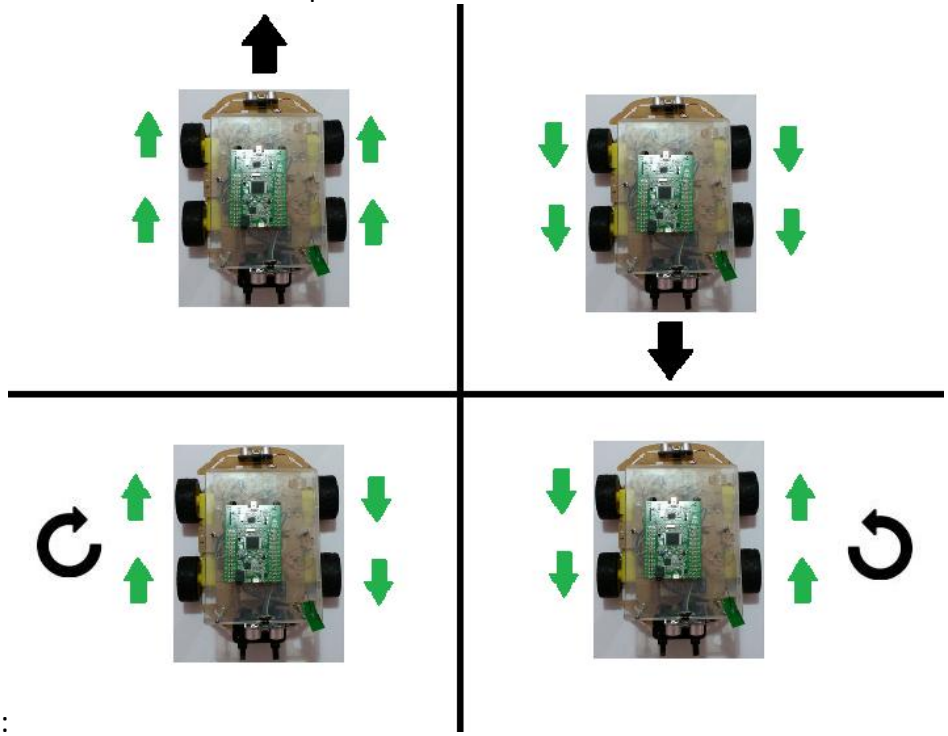


**Figure IV.33** : Direction de rover selon la rotation des roues

## 5.2 Capture de données

### 5.2.1 Capture de l'accelerartion

Pour cela on a relié l'accéléromètre ADXL345 au microcontrôleur comme la montre La figure IV.34 et le tableau IV.5 :

| STM32F4 | ADXL345 |
|---|---|
| PC9 I2C_3 | SDA |
| PA8 I2C_3 | SCL |
| VCC 3V | VCC, CS |
| GND | GND |

***Tableau IV.5 :*** *Branchement de l'ADXL345.*

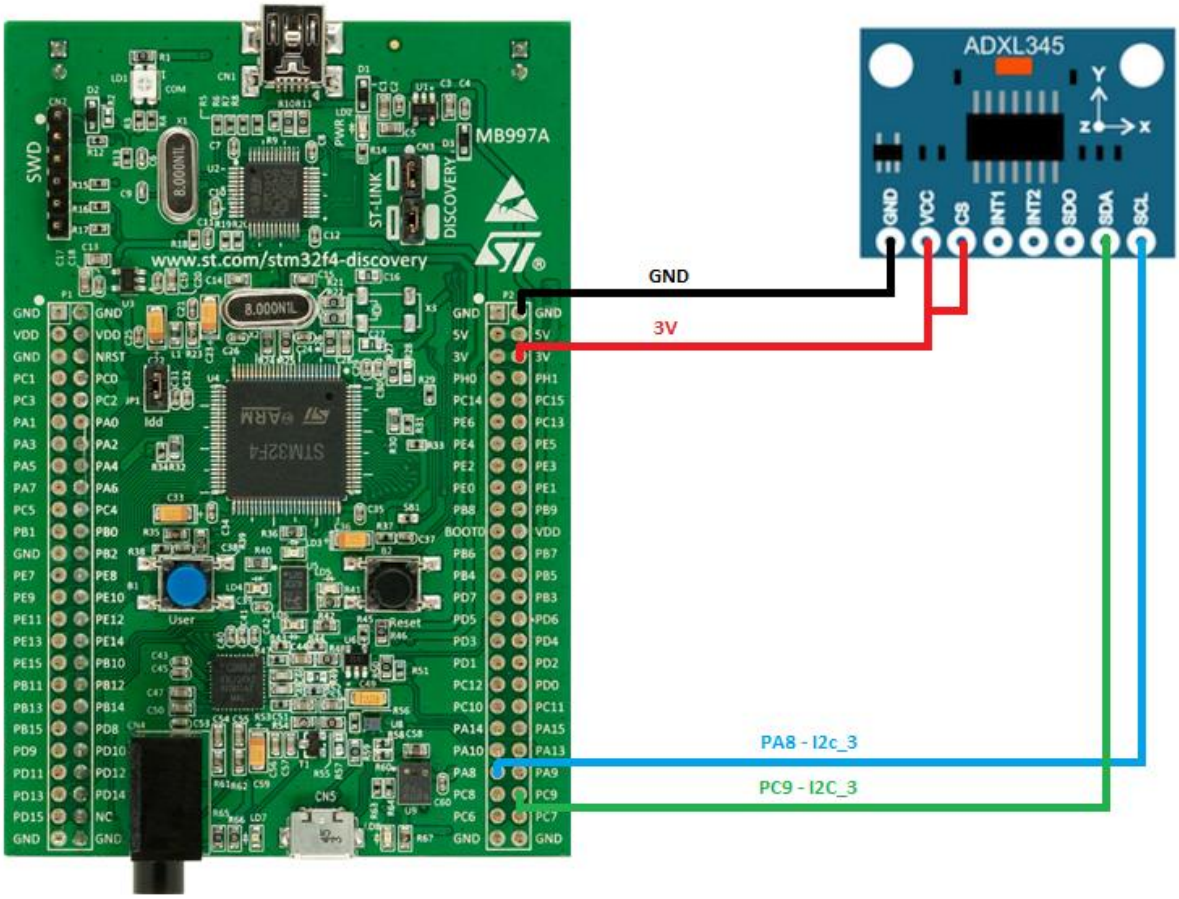



**Figure IV.34**: Branchement de l'ADXL345.

### 5.2.2 Capture de l'orientation

Afin de savoir dans quelle direction se dirige le Rover par rapport au nord on a relié la boussole numérique GY-271 au microcontrôleur comme la montre La figure IV.35 et Le tableau IV.6 :

| STM32F4 | GY-271 |
|---|---|
| PB7 I2C_1 | SDA |
| PB6 I2C_1 | SCL |
| VCC 3V | VCC |
| GND | GND |

***Tableau IV.6 :*** *Branchement du GY-271*

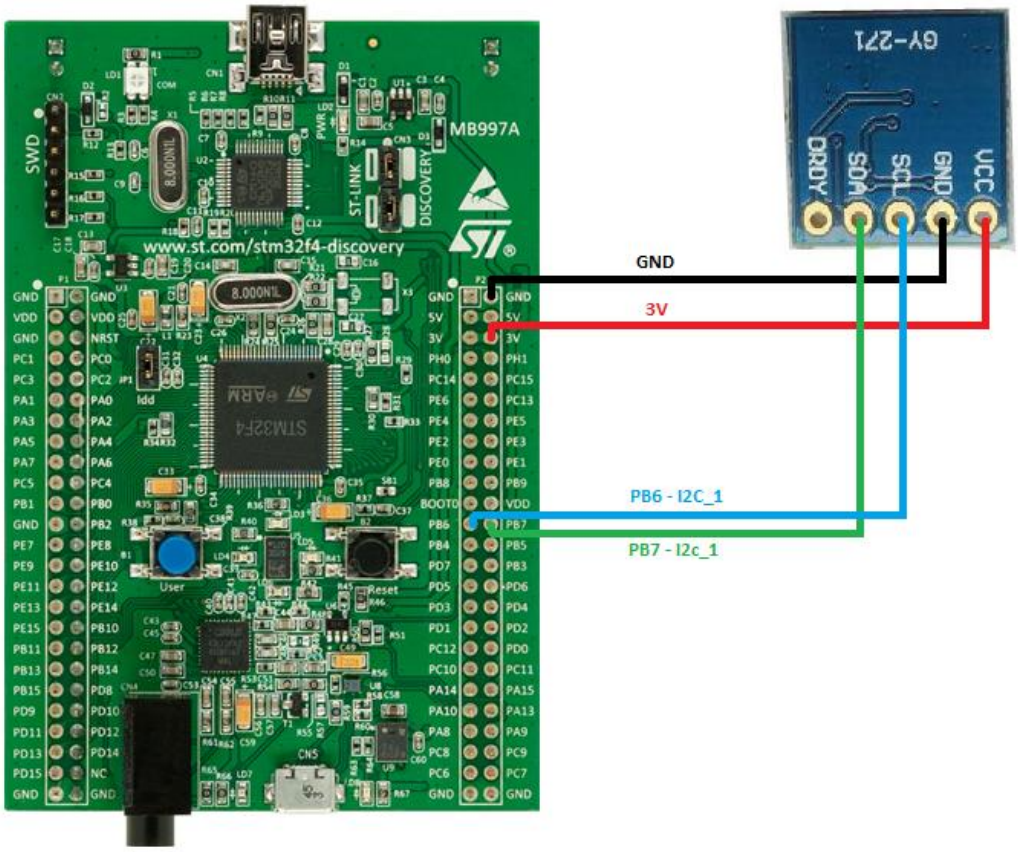



**Figure IV.35** : Branchement du GY-271.

### 5.2.3 Capture de la distance.

La figure IV.36 et Les tableaux ci-dessous montrent le branchement des HC-SR04s à la STM32F4 pour la détection d'obstacles :

- HC-SR04 de la partie avant du rover :

| STM32F4 | HC-SR04 |
|---|---|
| PA6 | TRIG |
| PA7 | ECHO |
| VCC 5V | VCC |
| GND | GND |

***Tableau IV.7** : Branchement du HC-SR04 (avant).*

- HC-SR04 de la partie arrière du rover :

| STM32F4 | HC-SR04 |
|---|---|
| PB4 | TRIG |
| PB5 | ECHO |
| VCC 5V | VCC |
| GND | GND |

***Tableau IV.8** : Branchement du HC-SR04 (arrière).*

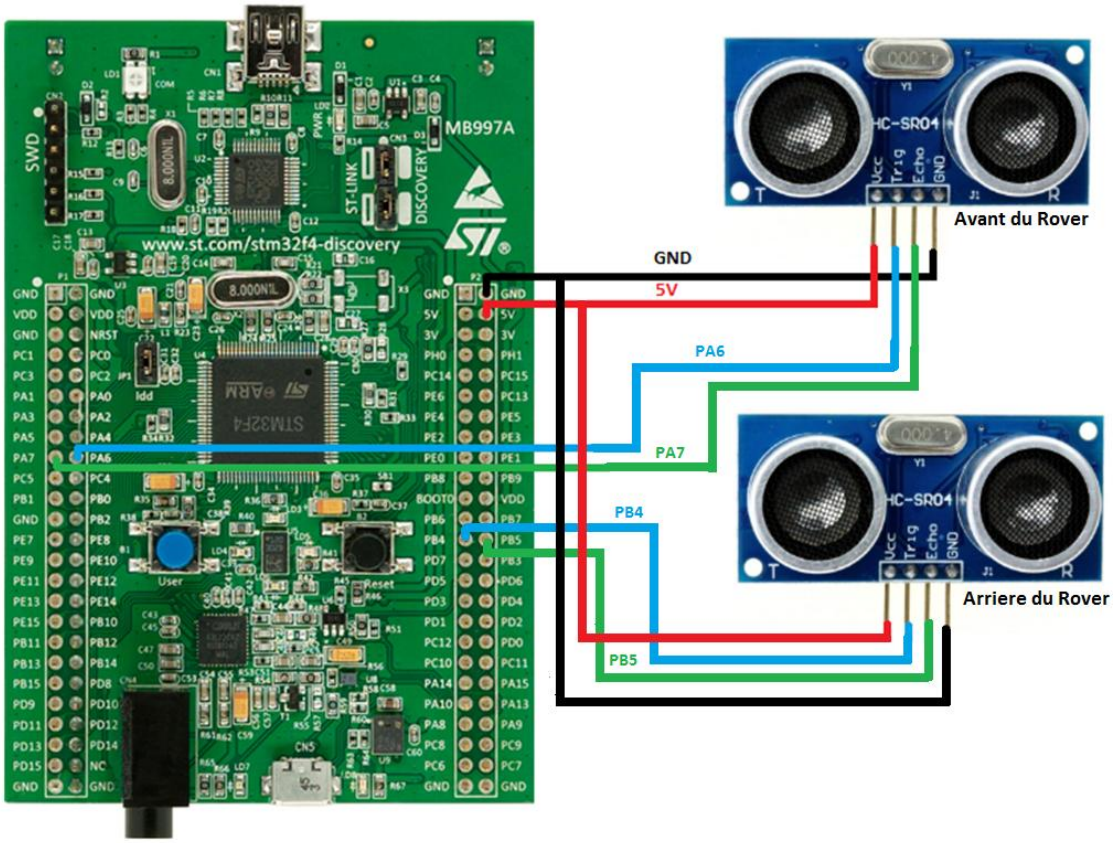



**Figure IV.36** : Branchement des HC-SR04s.

Le calcul de la distance est réalisé comme suit :

- **Initialisation du capteur :**
  Mettre le pin "TRIG" à une impulsion de niveau haut (5V) durant au moins 10us. Le module démarre ainsi sa lecture
- **Lecture des données :**
  Si le module détecte un objet devant lui, le pin "ECHO" passe au niveau haut (5V) et la distance de l'obstacle est proportionnelle à la durée de cette impulsion. Il est donc nécessaire d'effectuer le calcule de cette distance avec la formule suivante :

$$Distanc(cm) = (Echo\_Dist\_Time(\mu s)\ 10^{-4} x\, 340(ms^{-1}))/2$$

Avec :
340(m/s), la vitesse moyenne du son dans l'air
echo_dist_time = durée de l'impulsion en us
En divise par 2 parce que l'onde sonore fait un aller-retour.

Si la distance est inferieur à 10 cm le Rover est arrêté automatiquement. Et il ne peut plus avancer. Par contre, il peut reculer, tourner à gauche ou à droite.

Ces trois données capturées sont enveloppées dans une même requête, celle-ci est envoyée comme repense au client.
La figure IV.37 montre l'exemple d'une requête repense provenant du Rover vers le Smartphone.

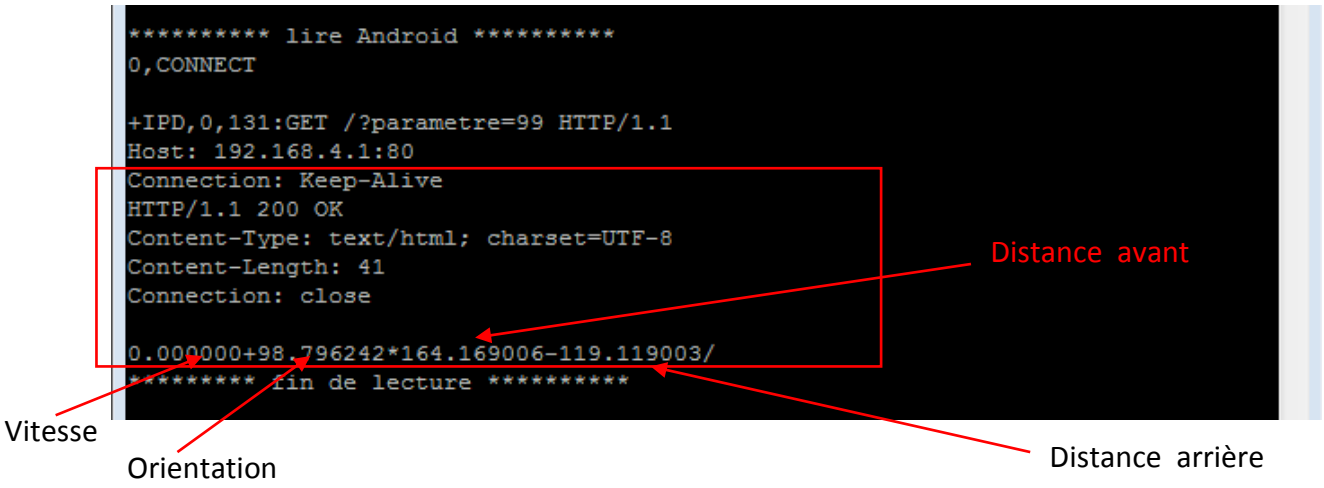



**Figure IV.37** : Requête Repense.

### 6 Assemblage du Rover

On présente ci-dessous l'assemblage final du Rover :

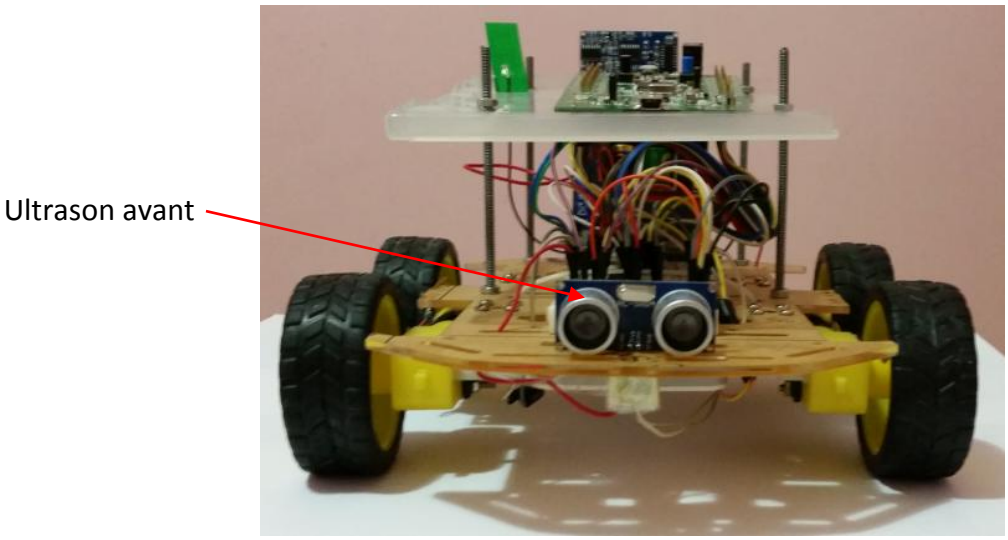



**Figure IV.38** : Le Rover vue de fasse.

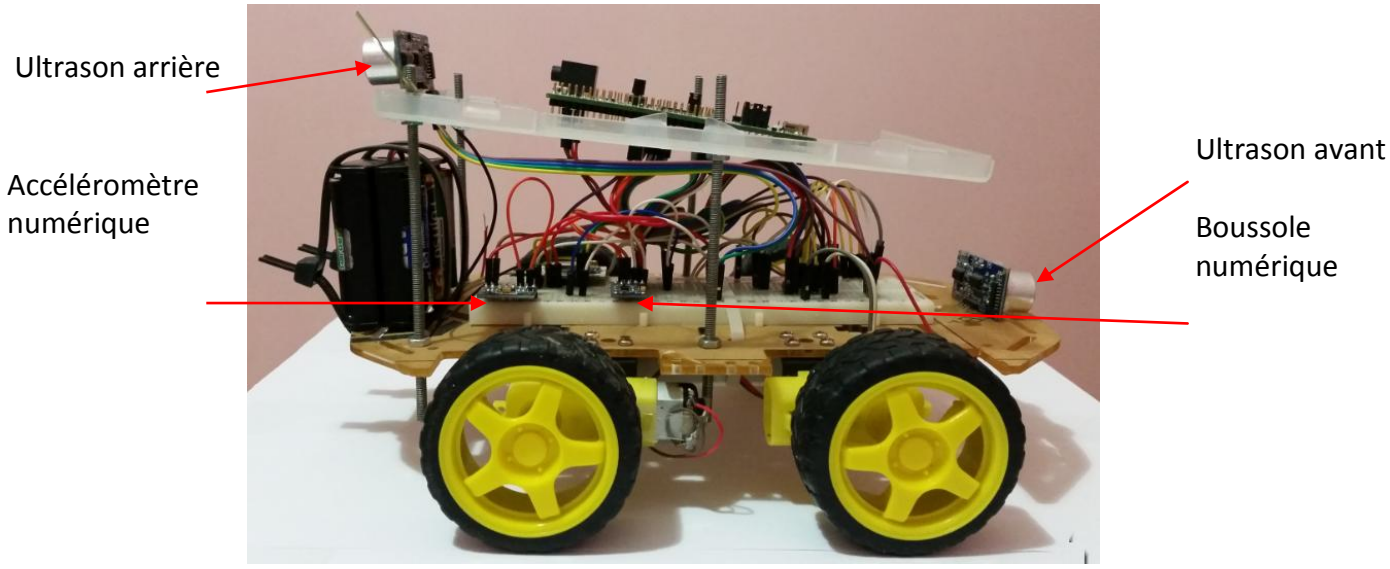



**Figure IV.39** : Le Rover vue de coté droit.

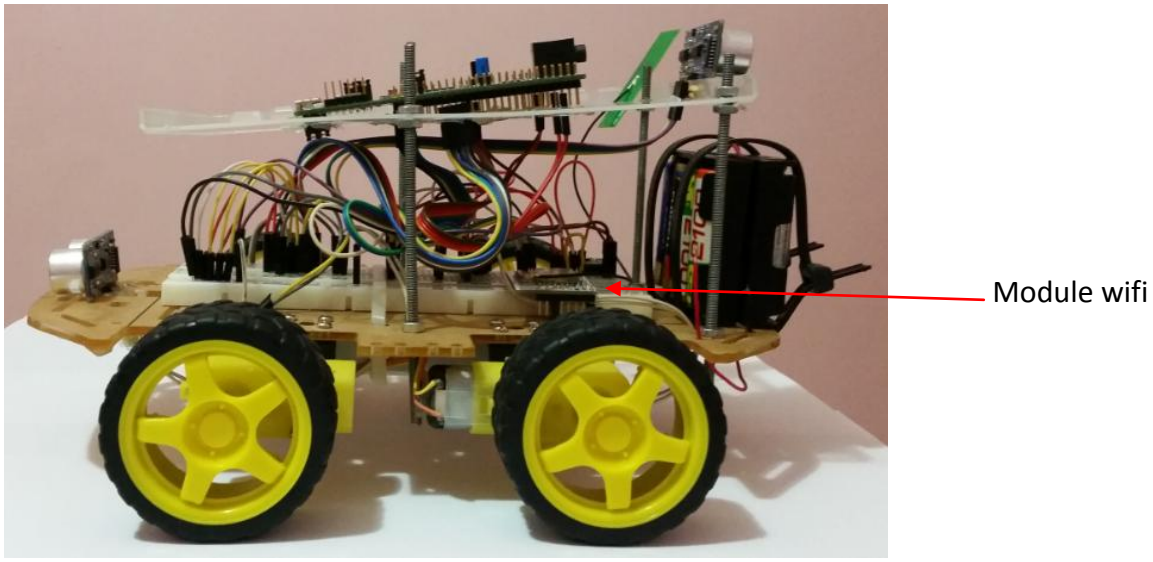



**Figure IV.40** : Le Rover vue de coté gauche.

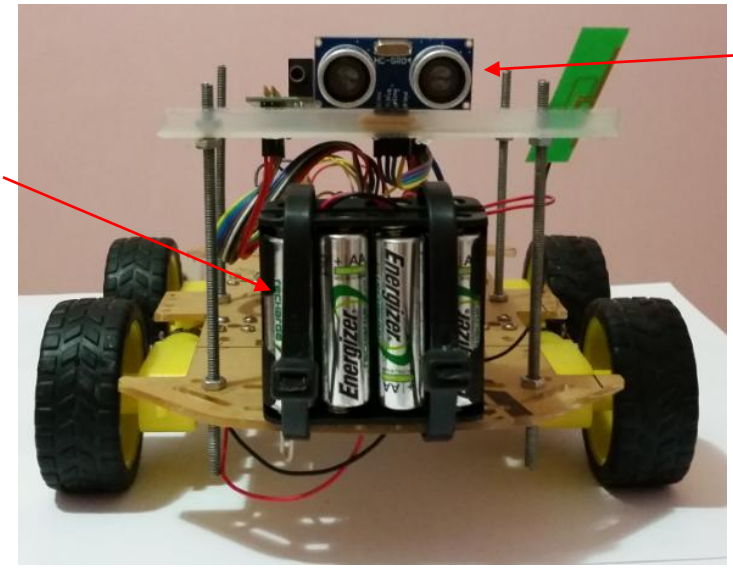



**Figure IV.41** : Le Rover vue de derrière.

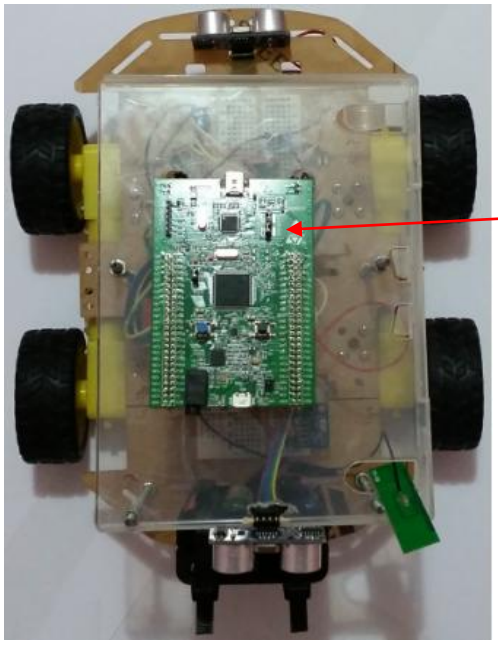



**Figure IV.42** : Le Rover vue de dessus.

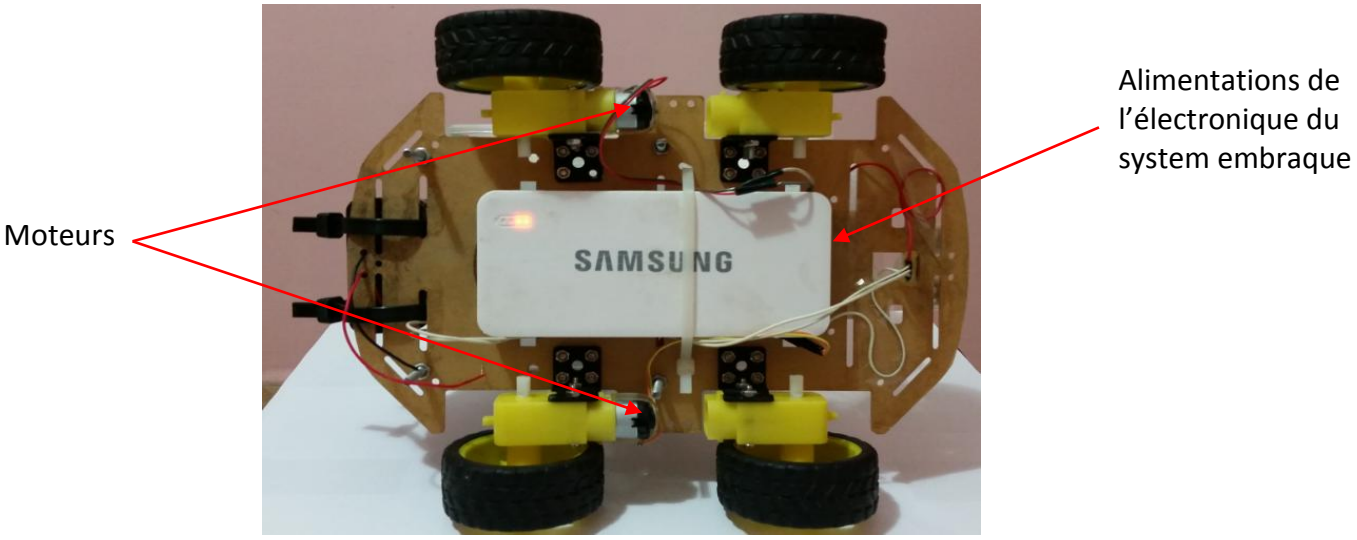



**Figure IV.43** : Le Rover vue de dessous.

**Conclusion**

Dans ce dernier chapitre, nous avons explicité les différentes étapes qui nous ont permis de réaliser ce Rover, ces composants utilisés et leurs tâches, on a présenté aussi une description générale du programme C implémenté sur la carte STM32F407 Discovery et l'application Android qui permet de Controller le Rover.

# CONCLUSION GENERALE

## Conclusion Générale

Nés dans les années 70, les systèmes embarqués font aujourd'hui partie intégrante de notre vie et tendent 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 embarqué, 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 et en hardware, ainsi qu'une bonne maitrise de l'environnement Android.

En arrivant au terme de notre travail, on peut dire que nous avons atteint un double objectif, le premier est la réalisation de notre système très complexe, et le second est enrichissement de nos connaissances en la matière, ainsi que la finalisation de nos études par un stage au niveau de laboratoire LABO-DZ, ce stage nous permettras d'avantage d'intégrer le domaine professionnel.

Ce travail ne constitue qu'un début dans le domaine de la robotique. 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] Ramzi BOULKROUNE, « mémoire : Les systèmes embarqués », Université de Annaba , 2009.
- [02] Jalil Boukhobza, «cours : systèmes d'exploitation embarqués »,Université de Bretagne Occidentale.
- [03] A .B Benyamina "cours : Ordonnancement Hiérarchique Multi-Objectif D'Application. Embarquées Intensives", thèse de doctorat D'Etat, 2008.
- [04] http://www.techniques-ingenieur.fr/
- [05] http://www-igm.univ-mlv.fr/~dr/XPOSE2002/SE/architecture.html
- [06] Audrey Marchand 2005-2006 « Linux pour l'embarqué ».
- [07] Jlassi Khaled ,« cours : Microprocesseurs et Microcontrôleurs » Université Virtuelle de Tunis 2008.
- [08] Sylvain MONTAGNY, « cours :Microprocesseurs et Microcontrôleurs » Université de Savoie.
- [09](http://www.reality.be/elo/labos2/uart.html
- [10] P. DUMARESQ et F. SIMARD, » Conception d'un système d'analyse de mouvement avec capteurs inertiels », avril 2012.
- [11] Jérôme VICENTE,« Les Microcontrôleurs » Ecole polytechnique universitaire de Marseille 2005-2006 .
- [12]https://fr.scribd.com/doc/109899931/Cours-Systeme-d-exploitation-Android.
- [13] Nouha Khyar, « Locate my car », Ecole nationale des sciences de l'informatique, Tunis.
- [14] : A. ALLOUI, A. HAJ 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.
- [15] : M. GHAOUI. « Planification d'un mouvement pour un robot mobile» Thèse de magister université de Batna année 1997.
- [16] Mechatronics; Auteur: V.S.Bagad, ISBN9788184314908, Edition 2008.
- [17] Computer Aided Manufacturing; C. Elanchezhian,G.ShanmugaSundar; First Edition 2005, Second Edition 2007.
- [18] Robotics for Electronics Manufacturing: Principles and Applications in cleanroom automation; Auteur: Karl Mathia, ISBN978-0-521-87652-0Hardback.
- [19] Robotics and control; Auteur: Mittal &Nagrath, ISBN 0-07-048293-4.
- [20] 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.
- [21] : A. PRUSKI. «Robotique générale» Edition Ellipase 1988.
- [22] Robotics (par AppuuKuttan) ; Auteur : AppuuKuttan, ISBN978-81-89866-38-9

- [23] (G BERTHOME – synthese capeturs Lycée Mireille GRENET - COMPIEGNE).
- [24] Jonathan W. Valvano (livre: “Embadded systems” Introduction to ARM®CORTEX - M MICROCONTROLLERS),JUIN 2014.