



Université Sultan Moulay Slimane

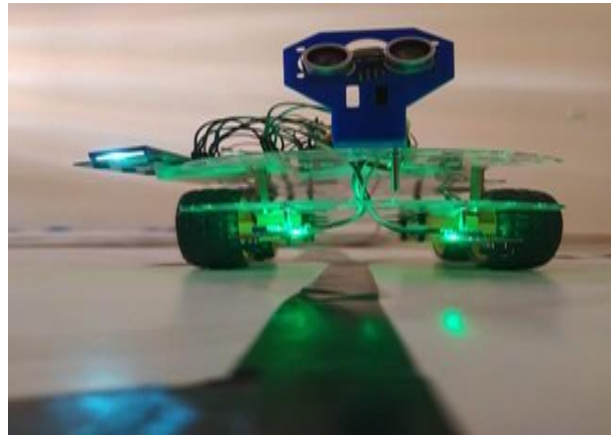
Faculté Polydisciplinaire

Département de Physique

- Béni Mellal -

Licence d'Études Fondamentales : Sciences de la Matière Physique

Parcours : Électronique



Projet de fin d'études

ROBOT SUIVEUR DE LIGNE ET DETECTEUR D'OBSTACLE

..... PFE réalisé par

EL KHABLI OMAR  
KASSIMI AMINE

..... Soutenu le 29 juillet 2020

Prof. A.MALAOUI, Encadrement

Année Universitaire : 2019-2020

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

قَالُوا سُبْحَانَكَ

لَا عِلْمَ لَنَا إِلَّا مَا عَلَّمْتَنَا

إِنَّكَ أَنْتَ الْعَلِيمُ الْحَكِيمُ

صَدَقَ اللَّهُ الْعَظِيمُ

# SOMMAIRE

## Table of Contents

Avant-propos.....	6
Dédicaces .....	7
Remerciements.....	8
Liste des figures .....	9
Liste des tableaux .....	11
Résumé.....	12
 <i>INTRODUCTION GENERALE.....</i>	 13
<i>CHAPITRE I : GENERALITES SUR LES SYSTEMES EMBARQUES.....</i>	 15
<i>I.1. INTRODUCTION .....</i>	 15
<i>I.2. DEFINITION D'UN SYSTEME EMBARQUE .....</i>	 15
<i>I.3. DEVELOPPEMENT DES SYSTEMES EMBARQUES.....</i>	 15
<i>I.4. DOMAINES D'APPLICATION.....</i>	 16
<i>I.5. CARACTERISTIQUES D'UN SYSTEME EMBARQUE.....</i>	 17
<i>I.6. COMPLEXITE D'UN SYSTEME EMBARQUE .....</i>	 17
<i>I.7. SYSTEME DE CONTROLE ET SYSTEME CONTROLE.....</i>	 18
<i>I.8. FIABILITE ET SURETE DES SYSTEMES EMBARQUES .....</i>	 19
<i>I.9. SECURITE DES SYSTEMES EMBARQUES .....</i>	 19
<i>I.10. COMPARAISON AUX SYSTEMES INFORMATIQUES STANDARDS .....</i>	 20
<i>I.11. CLASSIFICATION DES SYSTEMES EMBARQUES.....</i>	 20
<i>I.12. ARCHITECTURE GENERALE D'UN SYSTEME EMBARQUE.....</i>	 21
<i>I.13. APPLICATION DU SYSTEME EMBARQUE : ROBOT.....</i>	 22
<i>I.13.1.DEFINITION SUR D'UN ROBOT .....</i>	 22
<i>I.13.2.CARACTERISTIQUES D'UN ROBOT.....</i>	 22
<i>I.13.3.NOMBRE DES SERVICES DES ROBOTS DANS LE MONDE.....</i>	 23
<i>I.14. MICROCONTROLEURS DANS LES SYSTEME EMBARQUE.....</i>	 23

<b>I.14.1 DEFINITION .....</b>	<b>23</b>
<b>I.14.2. TYPES DES MICROCONTROLEURS .....</b>	<b>23</b>
<b>I.14.3. CARACTERISTIQUES PRINCIPALE .....</b>	<b>24</b>
<b>I.14.4.ARCHITECTURE DE MICROCONTROLEUR.....</b>	<b>24</b>
<b>I.15. CONCLUSION .....</b>	<b>24</b>
<b>CHAPITRE II : OUTILS NECESSAIRES POUR LA REALISATION .....</b>	<b>25</b>
<b>II.1. INTRODUCTION .....</b>	<b>25</b>
<b>II.2. PRESENTATION DU KIT D'ARDUINO.....</b>	<b>25</b>
<b>II.2.1. TYPES D'ARDUINO.....</b>	<b>26</b>
<b>II.2.2. TYPE D'ARDUINO : CARTE UNO.....</b>	<b>26</b>
<b>II.2.3. MICROCONTROLEUR D'ARDUINO UNO : ATMEGA328P .....</b>	<b>28</b>
<b>II.2.4. LOGICIEL DE PROGRAMMATION ARDUINO.....</b>	<b>29</b>
<b>II.3. CAPTEURS UTILISES.....</b>	<b>30</b>
<b>II .3.1 .CAPTEUR INFRAROUGE.....</b>	<b>30</b>
<b>II .3.2. CAPTEUR ULTRASON .....</b>	<b>31</b>
<b>II.4. ACTIONNEUR UTILISES .....</b>	<b>33</b>
<b>II.4.1. MOTEUR A COURANT CONTINU .....</b>	<b>33</b>
<b>II.4.2. PILOTE DE MOTEUR : L293D (PONT H) .....</b>	<b>34</b>
<b>II.4.3. PRESENTATION DU CIRCUIT L293D.....</b>	<b>35</b>
<b>II.5. AFFICHEUR LCD.....</b>	<b>35</b>
<b>II.6. CONCLUSION .....</b>	<b>37</b>
<b>CHAPITRE III : RESULTATS DE LA REALISATION ELECTRONIQUE ET MECANIQUE .....</b>	<b>38</b>
<b>III.1. INTRODUCTION .....</b>	<b>38</b>
<b>III.2. PRESENTATION DU PROJET .....</b>	<b>38</b>
<b>III.3. CAHIER DES CHARGES FONCTIONNEL.....</b>	<b>39</b>
<b>III-4-PARTIE ELECTRONIQUE .....</b>	<b>39</b>
<b>III.4.1 FONCTIONNEMENT D'ARDUINO DANS LE ROBOT.....</b>	<b>39</b>
<b>III.4.2. MODE DE FONCTIONNEMENT DES CAPTEURS DANS LE ROBOT.....</b>	<b>41</b>
<b>III.4.2.1. CAPTEUR INFRAROUGE .....</b>	<b>41</b>
<b>III.4.2.2. CAPTEUR D'OBSTACLE : ULTRASON.....</b>	<b>42</b>
<b>III.4.3. MOTEURS A COURANT CONTINU ET PONT H L293D .....</b>	<b>43</b>

<b>III.4.3.1. MOTEURS A COURANT CONTINU LIE A UNE ROUE .....</b>	<b>43</b>
<b>III.4.3.2. PILOTE DE MOTEUR : L293D (PONT H) .....</b>	<b>44</b>
<b>III.4.4. ALIMENTATION DU ROBOT : PILES .....</b>	<b>45</b>
<b>III.4.5. AFFICHEUR LCD .....</b>	<b>45</b>
<b>III.4.6. CONNEXION ENTRE LES COMPOSANTS DECRITS AU-DESSUS .....</b>	<b>45</b>
<b>III.4.7. SIMULATION DES MOTEURS AVEC LE CIRCUIT INTEGRE L293D SOUS LE             LOGICIEL ISIS PROTEUS .....</b>	<b>46</b>
<b>III-5-PARTIE MECANIQUE .....</b>	<b>47</b>
<b>III.5.1. CONCEPTION DU ROBOT.....</b>	<b>47</b>
<b>III.5.2. CHASSIS DU ROBOT.....</b>	<b>48</b>
<b>III.5.3. BRANCHEMENT DU ROBOT.....</b>	<b>48</b>
<b>III.5.4. MARCHE DU ROBOT .....</b>	<b>51</b>
<b>III.5.4.1. BLOC MOTEUR.....</b>	<b>51</b>
<b>III.5.4.2. DESCRIPTION DE LA CARTE DES CAPTEURS DE DETECTION DE LIGNE.....</b>	<b>52</b>
<b>III.5.4.3. SYSTEME DES CAPTEURS ALIGNES .....</b>	<b>52</b>
<b>III-5-PARTIE INFORMATIQUE .....</b>	<b>53</b>
<b>III.5.1. ALGORITHME .....</b>	<b>53</b>
<b>III.5.2. DESCRIPTION DES FONCTIONS DU PROGRAMME .....</b>	<b>53</b>
<b>III.5.3. DESCRIPTION DE L'ALGORITHME GLOBAL DU PROGRAMME .....</b>	<b>56</b>
<b>III.6. CONCLUSION.....</b>	<b>57</b>
<b>CONCLUSION GENERALE .....</b>	<b>58</b>
<b>ANNEXE.....</b>	<b>59</b>
<b>BIBLIOGRAPHIE .....</b>	<b>65</b>

## AVANT-PROPOS

### ❖ Nom (s) et prénom (s) des candidats

El KHABLI Omar

KASSIMI Amine

### ❖ Intitulé du travail

Robot suiveur de ligne et détecteur d'obstacle.

### ❖ Nom et prénom de l'encadrant

Pr. MALAOUI Abdessamad , Professeur à la Faculté Polydisciplinaire de Béni Mellal.

### ❖ Date de commencement du travail

05/07/2020

## ***DEDICACE***

Dédicace En premier lieu, à nos parents qui nous ont soutenus moralement et matériellement tout au long de notre vie. Nous espérons avant tout que nous serons toujours votre fierté. Que DIEU vous bénisse.

A nos frères et sœurs

A nos collègues à la Faculté Polydisciplinaire de Béni Mellal qui sont devenus de véritables amis pour les moments agréables qu'on a partagés tout au long de ces années, vous étiez toujours là à nous soutenir, à nous aider, et à nous encourager pour aller de l'avant dans notre travail.

A nos professeurs, qui ont remplacé nos parents pendant ces trois dernières années et ont pris en charge notre enseignement afin que nous puissions enrichir nos connaissances, nous espérons que nous serons le fruit mur de vos efforts et que nous ferons l'objet de votre fierté aussi.

A tous ceux qui sont loin de nos yeux mais toujours dans nos cœurs, merci pour votre amitié et votre amour.

## **REMERCIEMENTS**

Avant tous, nous tenons à remercier **Allah** le tout puissant, qui nous a donné la force et le courage afin d'élaborer ce modeste travail. Merci de nous avoir éclairé le chemin de la réussite.

Nous tenons tout d'abord à remercier Monsieur le Président de l'Université Sultan Moulay Slimane et Monsieur le Doyen de la Faculté Polydisciplinaire de Béni Mellal pour l'intérêt et le soutien qu'ils accordent à la recherche scientifique au sein de l'établissement.

Nous remercierons profondément, notre encadrant **Monsieur MALAOUI** Abdessamad, pour le privilège et la confiance qu'elle nous a accordé durant la réalisation de ce travail, pour son aide, sa patience, ainsi que pour ses précieux conseils. « Sincèrement monsieur, nous ne pouvons que vous exprimons notre respect et notre gratitude, grâce à vos conseils et à votre supervision on est entré dans un nouveau monde de sciences et de savoir ».

Nous exprimons notre profonde reconnaissance aux membres de jury pour l'honneur qu'ils nous font en acceptant de juger ce travail.

Nous exprimons nos remerciements à **EI-KHABLI** Ahmed Ingénieur en Télécommunications et systèmes d'information lauréat de l'INPT [Rabat] et le responsable de l'équipe recherche et développement embarqués chez Groupeer Technologies [France].





## ***LISTE DES FIGURES***

<b>FIGURE 1 : SCHEMA DU SYSTEME DE CONTROLE.....</b>	<b>18</b>
<b>FIGURE 2 : SCHEMA D'ARCHITECTURE D'UN SYSTEME EMBARQUE.....</b>	<b>21</b>
<b>FIGURE 3 : STOCK MONDIAL DES ROBOTS INDUSTRIELS DE 2008J JUSQU'A 2020 .....</b>	<b>23</b>
<b>FIGURE 4 : INTERACTION DE LA CARTE ARDUINO AVEC LES PERIPHERIQUES .....</b>	<b>26</b>
<b>FIGURE 5: CATRE ARDUINO UNO .....</b>	<b>27</b>
<b>FIGURE 6: STRUCTURE DE BASE D'UN PROGRAMME DANS LE LOGICIEL ARDUINO.....</b>	<b>29</b>
<b>FIGURE 7: IMAGE D'UN CAPTEUR INFRAROUGE.....</b>	<b>30</b>
<b>FIGURE 8: IMAGE D'UN CAPTEUR ULTRASON.....</b>	<b>31</b>
<b>FIGURE 9: IMAGE D'UN ROUE ET MOTEUR A COURANT CONTINU .....</b>	<b>33</b>
<b>FIGURE 10: IMAGE REPRESENTE LA CONSTITUTION DU MOTEUR A COURANT CONTINU .....</b>	<b>33</b>
<b>FIGURE 11 : IMAGE D'UN CIRCUIT L293D .....</b>	<b>34</b>
<b>FIGURE 12: IMAGE D'UN SHEILD.....</b>	<b>35</b>
<b>FIGURE 13: IMAGE D'UN ECRAN D'AFFICHEUR LCD .....</b>	<b>35</b>
<b>FIGURE 14: SCHEMA AFFICHEUR LCD .....</b>	<b>36</b>
<b>FIGURE 15: IMAGE DU ROBOT SUIT UNE LIGNE NOIRE .....</b>	<b>38</b>
<b>FIGURE 16: CODE C EXEMPLE BLINK.....</b>	<b>39</b>
<b>FIGURE 17 : CAS DE PRESENCE DE LIGNE NOIRE      FIGURE 18: CAS D'ABSENCE DE LIGNE NOIRE .....</b>	<b>42</b>
<b>FIGURE 19: SCHEMA ILLUSTRE LE FONCTIONNEMENT DU CAPTEUR ULTRASON .....</b>	<b>42</b>
<b>FIGURE 20: IMAGE REPRESENTE UN ENSEMBLE MOTEUR, REDUCTEUR ET ROUE .....</b>	<b>43</b>
<b>FIGURE 21: SCHEMA D'MOTEURS A COURANT CONTINU A CONNECTER AVEC LE CIRCUIT L293D .....</b>	<b>44</b>
<b>FIGURE 22: SIMULATION REPRESENTE LA CONNEXION DES COMPOSANTS DE NOTRE ROBOT SOUS ISIS.....</b>	<b>47</b>
<b>FIGURE 23: CONNEXION ENTRE LE MOTEUR A COURANT CONTINU ET LA ROUE .....</b>	<b>48</b>
<b>FIGURE 24: CONNEXION ENTRE LES MOTEURS ET LE DRIVER .....</b>	<b>48</b>
<b>FIGURE 25: FIXATION DES TROIS CAPTEURS OPTIQUES SUR LE CHASSIS.....</b>	<b>49</b>
<b>FIGURE 26: FIXATION DU CAPTEUR ULTRASON SUR LE CHASSIS.....</b>	<b>49</b>
<b>FIGURE 27: POSITION DE L'ARDUINO, LE MANIPULATEUR ET LE BOITIER.....</b>	<b>49</b>
<b>FIGURE 28: POSITION DE L'ARDUINO, LE MANIPULATEUR ET LE BOITIER.....</b>	<b>50</b>
<b>FIGURE 29: CONNEXION ENTRE LES PIECES DU ROBOT .....</b>	<b>50</b>

<b>FIGURE 30: CONNEXION DE L’AFFICHEUR ET LA CONNEXION DE LA VOITURE.....</b>	<b>51</b>
<b>FIGURE 31: DIRECTIONS POSSIBLES EN FONCTIONS DE TENSIONS MOTRICES.....</b>	<b>51</b>
<b>FIGURE 32: DESSIN DE LA CARTE DETECTION DE LIGNE .....</b>	<b>52</b>
<b>FIGURE 33: DEFERENT SITUATION DU ROBOT.....</b>	<b>52</b>
<b>FIGURE 34 : ORDINOGRAMME DE NOTRE ROBOT SUIVEUR DE LIGNE .....</b>	<b>53</b>
<b>FIGURE 35: SITUATION DU ROBOT POUR LA LIGNE DROITE .....</b>	<b>53</b>
<b>FIGURE 36: SITUATION DU ROBOT POUR VIRAGE GAUCHE .....</b>	<b>54</b>
<b>FIGURE 37: SITUATION DU ROBOT POUR VIRAGE DROITE.....</b>	<b>54</b>
<b>FIGURE 38: DISTANCE ENTRE LE ROBOT ET L’OBSTACLE.....</b>	<b>55</b>
<b>FIGURE 39: DIFFERENTES SITUATIONS DU ROBOT DURANT LA SUITE DE LA LIGNE .....</b>	<b>56</b>

## ***LISTEDESTABLEAUX***

<b>TABLE 1 : COMPARAISON ENTRE UN SYSTEME INFORMATIQUE ET UN SYSTEME EMBARQUE .....</b>	<b>20</b>
<b>TABLE 2 : CARACTERISTIQUES D'ARDUINO UNO .....</b>	<b>28</b>
<b>TABLE 3: PARAMETRES DU CAPTEUR ULTRASON .....</b>	<b>32</b>

## ***RESUME***

Au cours de notre étude en physique électronique, nous avons manipulé un robot qui marche dans un environnement autour des orientations optiques dont le but de se déplacer en ligne droite et courbe avec ses lumières vertes allumées. Il doit s'arrêter et s'allumer en rouge quand il arrive à 17 cm d'un obstacle. Si nous éliminons l'obstacle, il repart. L'objectif principal était de développer une série de travaux pratiques afin de montrer les avantages des systèmes embarqués. Pour cela, nous avons programmé ce robot à l'aide d'un type d'outil de développement : une carte ARDUINO qui contient un microcontrôleur ATMEL AVR (ATmega328P) dispose de 32ko de mémoire Flash permettant de stocker le programme à exécuter. Et nous avons utilisé des dispositifs disponibles comme : Capteurs, Moteur à courant continu, L293D, Afficheur LCD, Pile. A la fin de cette réalisation on peut ajouter d'autre capteurs pour développer ce mobile, les objectifs qui nous somme été proposés, sont été atteints.

Mot clé : robot, arduino, capteur ultrason, capteur infrarouge, L293D.

# Introduction générale

Durant notre parcours pour la spécialité électronique issu de la filière science de la matière physique(SMP), nous avons réalisé notre Projet de Fin d'étude (PFE) afin de mettre en œuvre les différents enseignements reçus dans cette section. Ce projet était une occasion pour mettre en pratique nos connaissances et compétences acquises dans notre formation. Pour cela, nous avons décidé de réaliser un robot suiveur de ligne détecteur d'obstacle toutes en mettant en défi nos connaissances en informatique, en électronique numérique, analogique et physique théorique vue dans notre parcours académique.

Notre objectif technique au terme de ce projet est de réaliser un robot éducatif, facile à monter et bon marché, capable de suivre une ligne qu'elle soit droite ou courbe, il détecte des obstacles. Vue l'importance des robots actuels dans notre vie quotidienne, et son réaction avec l'environnement. Le premier but des robots est de remplacer l'homme dans des activités difficiles et fatigantes pour l'employeur, aussi dans des tâches qui demandent une étroite collaboration avec lui, et en cas d'exploration des milieux qui présentent des dangers pour l'homme. Par exemple nous pouvons les utiliser dans plusieurs domaines : L'industrie, Le domaine militaire, La santé, Utilisation civile, L'espace, L'usage domestique...

Pour notre démarche nous avons effectué une étude préliminaire pour voir les besoins et les disponibilités des robots. Dans notre travail nous avons choisi des composants électroniques qui contiennent des éléments numériques, analogiques et nombre de porte. Pour cela, nous avons utilisé l'arduino puis ce qu'il contient ces descriptions, de plus il est disponible et moins cher. Aussi dans notre démarche nous avons besoin de trois parties mécanique, électronique et informatique avec une simulation dans le logiciel ISIS.

Pour la partie électronique, nous avons préalablement étudié l'impact des capteurs optique (infrarouge) qui détecte au sol la direction du parcours (ligne noire), ultrason qui est un module permet de voir les obstacles à l'avant et d'en connaître la distance. Ainsi le rôle de l'arduino qui est une plateforme matérielle et logicielle de développement d'applications embarquées. Et le moteur à courant continu qui permet la rotation de roue lié au shield qui contrôle les moteurs. La batterie alimente le robot par un courant continu. Enfin un afficheur LCD qui affiche la distance de détection des obstacles.

Pour la partie mécanique, nous avons choisi d'abord de discuter tous ensemble de la forme générale du robot. Notamment au niveau de la place nécessaire pour positionner les capteurs et autres composants électroniques et leurs adaptations. Nous avons dessiné sur une feuille les composants et les pièces qui nous

étions fournies, telles que les moteurs, les roues, les piles et la plateforme robotique. Mais il est rapidement apparu qu'il serait plus simple d'utiliser un logiciel ISIS était installé sur l'ordinateur nous donnant des valeurs précises.

Pour la partie informatique nous avons travaillé par un logiciel pour programmer la carte arduino qui permet de développer des programmes de langage peuvent être transférés sur la carte en branchant celle-ci sur un port USB.

Grosso modo nous avons réalisé réellement un prototype opérationnel se déplacera sur une moquette blanche et devra suivre une ligne noir droite ou courbe d'une largeur de 19 mm, il devra détecte des obstacles et affiche la distance de détection sur un afficheur LCD. Il pourra ensuite effectuer plusieurs figures et respecter les consignes indiquées dans le cahier des charges pour marquer des points et ainsi se qualifier pour les tours suivants. Cette piste sera étendue sur environ 10m.

Finalement ce projet sera exploité par la suite dans le cadre du master, pour le rendre améliorer par l'ajout ou le remplacement des composants plus évolués.

Ce travail est organisé en trois chapitres :

Le premier chapitre contient certaines définitions générales sur les systèmes embarqués et ses domaines d'applications, caractéristiques, architecture. Nous avons parlé sur les robots, ses caractéristiques et leurs nombre services. Encore nous avons défini les microcontrôleurs, ses types et ses caractéristiques.

Le deuxième chapitre consiste des informations sur l'Arduino, Capteurs infrarouge et ultrason, moteurs à courant continu, L293D et afficheur LCD.

Le troisième chapitre traite la partie de réalisation qui consiste la méthode branchement des composant électronique, également simulation dans ISIS et L'algorithme de programmation.

# Chapitre I : Généralités sur les systèmes embarqués

## I.1. Introduction

Les systèmes embarqués sont des systèmes électroniques qui sont complètement intégré au système qu'ils contrôlent. Nous avons les utilise dans différents domaines d'application telle que dans la robotique, le transport, l'astronautique, le militaire les télécommunications, l'électroménager, les guichets bancaires automatiques... Son utilisation dans ces multiples domaines traduit bien l'importance du marché de l'embarquer de nos jours. Il est un système qui contient des composants suivant :

Processeur, microcontrôleur, mémoire...

## I.2. Définition d'un système embarqué

Un **système embarqué** est un système complexe qui intègre du logiciel et du matériel conçus ensemble afin de fournir des fonctionnalités données. Il contient généralement un ou plusieurs microprocesseurs destinés à exécuter un ensemble de programmes définis lors de la conception et stockés dans des mémoires. Le système matériel et l'application (logiciel) sont intimement liés et immergés dans le matériel et ne sont pas aussi facilement discernables comme dans un environnement de travail classique de type ordinateur de bureau PC (*Personal Computer*).

Un système embarqué est autonome et ne possède pas des entrées/sorties standards tels qu'un clavier ou un écran d'ordinateur. Contrairement à un PC, l'interface IHM (Interface Homme machine) d'un système embarqué peut être aussi simple qu'une diode électroluminescente LED (*Light Emitter Diode*) qui clignote ou aussi complexe qu'un système de vision de nuit en temps réel, les afficheurs à cristaux liquides LCD (*LiquidCrystal Display*) de structure généralement simple sont couramment utilisés.

Afin d'optimiser les performances et la fiabilité de ces systèmes, des circuits numériques programmables FPGA (*Fild Programmable GateArray*), des circuits dédiés à des applications spécifiques ASIC (*Application Specific Integrated Circuits*) ou des modules analogiques sont en plus utilisés. Le logiciel a une fonctionnalité fixe à exécuter qui est spécifique à une application, l'utilisateur n'a pas la possibilité de modifier les programmes. Bien souvent, il n'a pas conscience d'utiliser un système à base des microprocesseurs.

## I.3. Développement des systèmes embarqués

Le développement des systèmes embarqués nécessite des connaissances à la fois en électronique et en informatique. Parmi le matériel nécessaire pour réaliser un système embarqué nous avons trouvé :

- La documentation (*datasheet*) sur les composants utilisés, c'est la première source d'informations pour le développement.
- L'outillage de base de l'électronicien (fer à souder, insoleuse...).
- Les outils d'analyses temporelles : oscilloscope, analyseur logique...
- Des composants de base (résistances, condensateurs...).
- Un microprocesseur ou un microcontrôleur.
- Un compilateur croisé (dit aussi en anglais cross-compiler).
- Un programmeur de microcontrôleur ou un programmeur in-situ.
- Un émulateur in-circuit ou ICE (*In Circuit Emulateur*). Cet équipement permet le débogage matériel et logiciel (possibilité de déverminer au niveau de la source du logiciel), cependant il reste coûteux.
- Une sonde JTAG. Peu coûteuse et très répandue, la sonde JTAG permet non seulement le débogage du logiciel in situ (lecture/modification de registres, mémoires, périphériques...) mais aussi la programmation de la mémoire FLASH des microcontrôleurs, que celle-ci soit interne ou externe comme dans le cas de certaines puces du fabricant NXP.
- Ingénierie des systèmes: approches multidisciplinaires pour définir, développer et déployer des systèmes embarquant des technologies numériques.

## I.4. Domaines d'application

Les domaines dans lesquels on trouve des systèmes embarqués sont de plus en plus nombreux :

- Astronautique : fusée, satellite artificiel, sonde spatiale, etc.
- Automate programmable industriel, contrôle-commande.
- Électroménager : télévision, four à micro-ondes.
- Environnement [archive].
- Équipement médical.
- Guichet automatique bancaire (GAB).
- impression : imprimante multifonctions, photocopieur, etc.
- Informatique : disque dur, Lecteur de disquette, etc.
- Métrologie.
- Militaire : missile.
- Multimédia : console de jeux vidéo, assistant personnel.
- Télécommunication : Set-topbox, téléphonie, routeur, pare-feu, serveur de temps, Téléphone portable.



- Transport : Automobile, Aéronautique (avionique), Ferroviaire.
- Robotique.

## I.5. Caractéristiques d'un système embarqué

- Temps réel : Les systèmes embarqués fonctionnent généralement en Temps Réel (TR): les opérations de calcul sont alors faites en réponse à un événement extérieur (interruption matérielle). La validité et la pertinence d'un résultat dépend du moment où il est délivré. Une échéance manquée induit une erreur de fonctionnement qui peut entraîner soit une panne du système (plantage), soit une dégradation non dramatique de ses performances.
- Faible coût : Lorsque les systèmes embarqués sont utilisés dans les produits de grande consommation, ils sont fabriqués en grande série. Les exigences de coût se traduisent alors en contraintes sur les différentes composantes du système : utilisation de faibles capacités mémoires et de petits processeurs (4 bits ou 8 bits), mais en grand nombre. Ainsi, les systèmes embarqués sont particulièrement sensibles au coût de production. Il existe des applications dans lesquelles les contraintes de coût de production et de maintenance ont une importance de même niveau que les performances envisagées.
- Consommation d'énergie : Dans les systèmes embarqués autonomes, la consommation d'énergie est un point critique pour le coût. En effet, une consommation excessive augmente le prix de revient du système embarqué, car il faut alors des batteries de forte capacité.
- Poids et volume
- Mobilité
- Autonomie
- Sécurité
- Fiabilité

## I.6. Complexité d'un système embarqué

Les systèmes embarqués requièrent souvent un faible encombrement (faible poids) PDA (*Personal Digital Assistant*), Internet et téléphone mobiles, ...). Leur technologie fait alors appel à une électronique et à des applications portables où l'on doit minimiser aussi bien l'encombrement que la consommation électrique. 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 conception actuelles, un système embarqué est généralement intégré sur un support silicium unique constituant ainsi un système complet intégré sur une puce SoC (*System on a Chip*).

Les systèmes sur puce contiennent généralement une grande variété de dispositifs programmables tels que des microcontrôleurs, des processeurs de traitement de signaux DSP (*Digital-Signal Processor*) et des ASIC qui sont développés pour des applications complexes nécessitant une production en grande série.

Les mémoires (ROM et RAM) y sont intégrés pour le stockage des données et des programmes. Ces composants digitaux cohabitent généralement sur le même support de silicium avec des composants analogiques et mixtes divers tels que des composants radiofréquences (RF) comme moyen de communication, des composants optiques pour le transfert de données à haut débit, des MEMS (*Micro Electro Mechanical System*) pour l'interfaçage avec le monde externe, des convertisseurs analogiques/numérique et numérique/analogique requis pour le dialogue interne. L'objectif est d'obtenir une coopération harmonieuse entre composants embarqués afin de garantir des services globaux.

Les contraintes d'implémentation physique sont liées à la consommation de ressource et au contexte de déploiement tel que le poids, la taille physique, la résistance aux vibrations, ou aux irradiations, ..., etc.

## I.7. Système de contrôle et système contrôlé

Quelle que soit la nature et la complexité du système, le système embarqué est composé en deux types :

- **Le système contrôlé** : environnement (procédé) équipé d'une instrumentation qui réalise l'interface avec le système de contrôle.
- **Le système de contrôle** : éléments matériels (microprocesseurs...) et logiciels dont la mission est d'agir sur le procédé via les actionneurs en fonction de l'état de ce procédé indiqué par les capteurs de manière maintenir ou conduire le procédé dans un état donné.

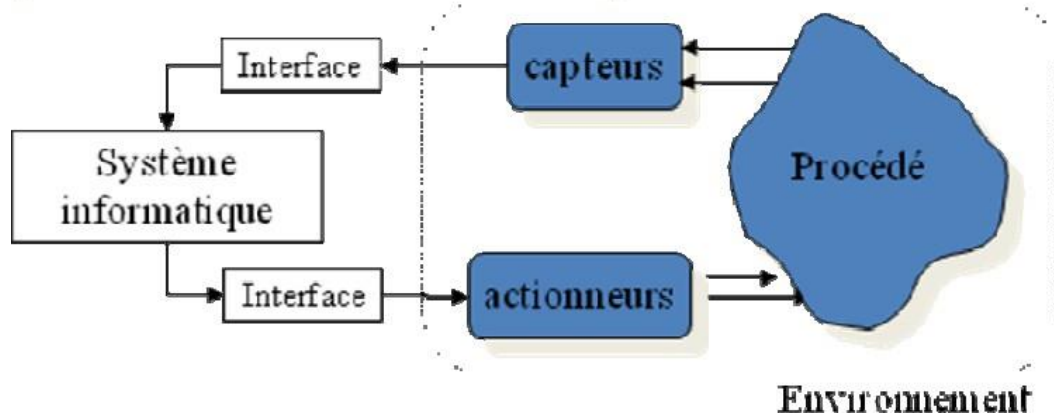


Figure 1 : Schéma du système de contrôle

Un système électronique embarqué ou enfoui est un élément constitutif d'un système plus complexe pour lequel il rend des services bien précis (contrôle, surveillance, communication...). Il est constitué de parties matérielles et logicielles qui sont conçues spécifiquement pour réaliser une fonction dédiée.

Système embarqué = Système électronique/informatique conçu pour réaliser une ou plusieurs tâches précises.

## **I.8. Fiabilité et sûreté des systèmes embarqués**

Du fait de leur portabilité et de la mobilité des produits dans lesquels ils sont incorporés, les systèmes embarqués évoluent généralement dans de conditions environnementales non déterministes et souvent non maîtrisées. Ils sont exposés à des variations et autres contraintes environnementales susceptibles d'induire des défaillances : vibrations, chocs, variation de température, variations d'alimentation, interférences RF, corrosion, humidité, radiations, ...

D'où la nécessité de prendre en compte des évolutions des caractéristiques des composants en fonction des conditions environnementales.

En même temps que s'accroît leur sophistication, les systèmes embarqués sont utilisés dans des applications de plus en plus critiques dans lesquels leur dysfonctionnement peut générer des nuisances, des pertes économiques ou des conséquences inacceptables pouvant aller jusqu'à la perte de vies humaines. C'est le cas, par exemple, des applications médicales ou celles de transports pour lesquelles une défaillance peut avoir un impact direct sur la vie d'êtres humains. C'est aussi le cas des applications spatiales, souterraines ou sous-marines où la défaillance peut entraîner des conséquences redoutables aussi bien en termes de sécurité qu'au niveau économique. Ce type de systèmes doit garantir une très haute fiabilité et doit pouvoir réagir en cas de panne de l'un de ses composants.

## **I.9. Sécurité des systèmes embarqués**

Les systèmes embarqués mettant en œuvre la connectivité IP sont aujourd'hui potentiellement vulnérables à une attaque par le réseau. Les attaques concernent actuellement les routeurs, les imprimantes réseau... mais rien n'empêche une attaque d'une maison individuelle avec son réseau domotique ou d'une voiture connectées à Internet !

L'aspect sécurité d'un système embarqué doit être maintenant pris en compte lors de sa conception, la sécurité des systèmes embarqués concerne essentiellement les points suivants :

- Le matériel.
- Le logiciel embarqué.
- Les communications avec le monde extérieur.

Un système embarqué doit garantir la confidentialité et l'intégrité des données. Parmi les types d'attaques sur un SE on peut citer :

- Les attaques matérielles : accéder aux composants, Attaques en courant, électromagnétisme (il faut utiliser la cage de Faraday), utiliser un faisceau laser ou faisceau d'ions lourds, le probing qui consiste à une descente de sondes pour agir sur les composants (il faut faire le durcissement : technologies SOS (Silicon On Sapphire) et SOI (Silicon On Insulator – isolant)).
- Les attaques logicielles : virus, cheval de Troie, deny of service, modification d'adresse IP, modification d'horloge, fermer un port socket.
- Détection et piratage des communications.

## I.10. Comparaison aux systèmes informatiques standards

La différence entre un système embarqué et un système informatique générale de but est un but bien précis et dans une bien moindre mesure, de la conception. Même si un système polyvalent peut être utilisé pour beaucoup de choses, un système embarqué est uniquement destiné à un seul but. Systèmes d'usage général :

*Table 1 : Comparaison entre un système informatique et un système embarqué*

<b>Informatique :</b>	<b>Embarqué :</b>
<ul style="list-style-type: none"> <li>• Processeur standard</li> <li>– Multiples unités fonctionnelles (flottant)</li> <li>– Vitesse élevée (&gt; GHz)</li> <li>– Consommation électrique élevée</li> <li>– Chaleur</li> <li>– Taille</li> <li>• MMU (Memory Management Unit) (mémoire virtuelle)</li> <li>• OS</li> <li>• Cache</li> <li>• Grand nombre de périphériques</li> </ul>	<ul style="list-style-type: none"> <li>• Processeur dédié (contrôleur)</li> <li>– Architecture adaptée</li> <li>– Vitesse faible (~200 MHz)</li> <li>– 8-32bits : mémoire limitée</li> <li>– Basse consommation</li> <li>– Petite taille, grand volume =&gt; faible coût</li> <li>• Processeur DSP (Digital signal processor) (traitements)</li> <li>– Très puissants</li> <li>• Quelques Mo de mémoire</li> <li>• RTOS</li> </ul>

## I.11. Classification des systèmes embarqués

### *a. Système Transformationnel :*

Activité de calcul, qui lit ses données et ses entrées lors de son démarrage, qui fournit ses sorties, puis meurt.

### *b. 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.

### *c. 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.

## **I.12. Architecture générale d'un système embarqué**

L'architecture d'un système embarqué se définit par le schéma suivant:

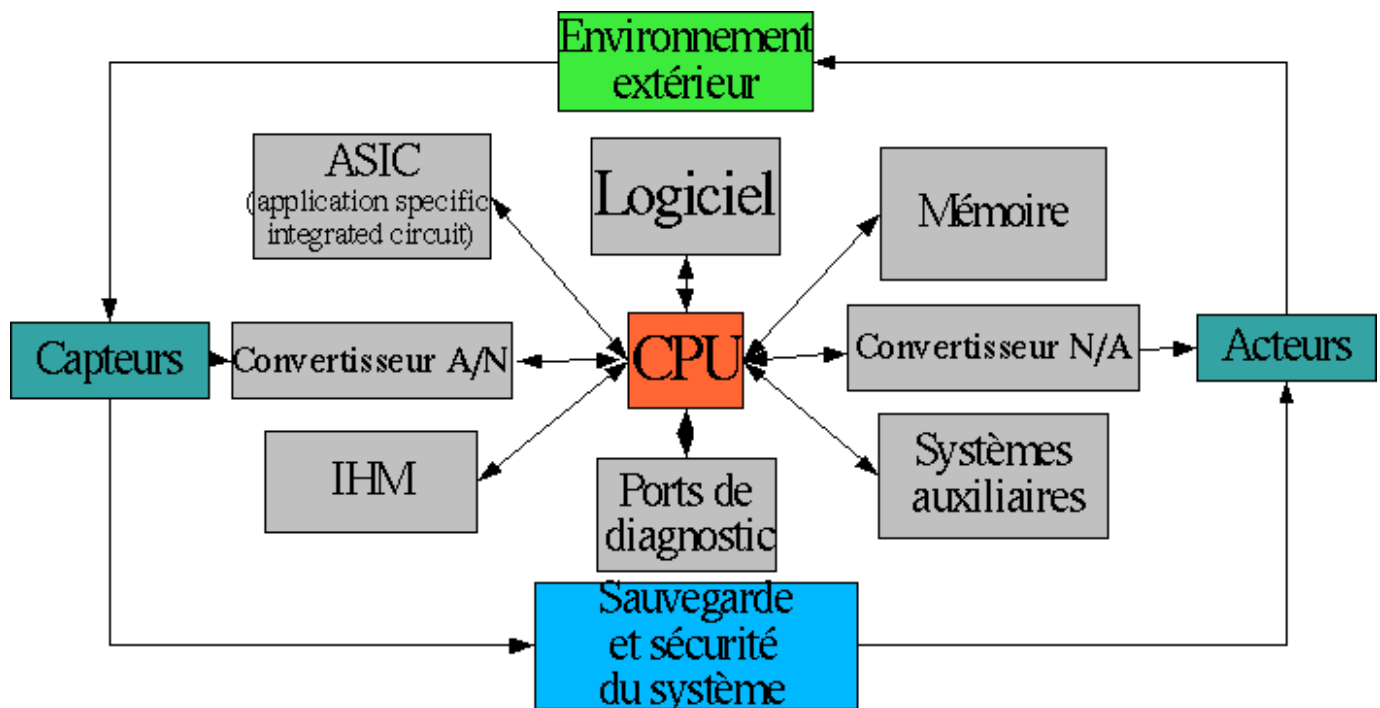


Figure 2 : Schéma d'architecture d'un système embarqué

- IHM : Interface Homme Machine
- A/N : Convertisseur Analogique Numérique
- N/A : Convertisseur Numérique Analogique
- CPU : Central Processing Unit (Processeur)
- ASIC: Application Specific Integrated Circuit

Cette architecture peut varier selon les systèmes: 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 réside parfois uniquement en un logiciel spécifique (ex: routeur), ou une boucle d'exécution (ex: ABS).

De même l'interface IHM 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, de l'ASIC et éventuellement de système externes traite l'information.
- Le traitement génère éventuellement une sortie qui est envoyée vers la sortie, les systèmes auxiliaires, les ports de monitoring ou l'IHM.

## I.13. Application du système embarqué : Robot

### I.13.1. Définition sur d'un robot

Il a été utilisé pour la première fois en 1921 par Karel Capek dans sa pièce R.U.R. : Rossums Universal Robots. Il provient du tchèque "robot signifie *travail forcé* en Tchèque" qui signifie corvée, travail obligatoire.

Est un automate doté des capteurs et d'effecteurs lui donnant une capacité d'adaptation et de déplacement proche de l'autonomie. De plus est un agent physique réalisant des tâches dans l'environnement dans lequel il évolue.

Machine qui imite une créature intelligente, peut être totalement mécanique (inintéressant ici), il est composé d'un ou plusieurs systèmes embarqués.

### I.13.2. Caractéristiques d'un robot

- L'**exactitude de positionnement**, définie par une position et une orientation dans l'espace cartésien
- La **vitesse** de déplacement (vitesse maximale en élongation maximale), l'**accélération**
- La **masse** du robot (de quelques centaines de kilos à quelques tonnes)
- Le **coût** du robot (pour petits robots avec charge maximale de quelques kilogrammes,
- La **maintenance** du robot (difficile pour les robots qui travaillent dans des environnements dangereux. Exemple : chambre froide)
- La **répétabilité** caractérise la capacité que le robot à retourner vers un point (position, orientation) donné.

### I.13.3. Nombre des services des robots dans le monde

Durant notre année 2020, tous secteurs industriels confondus, plus de 1,7 million de nouveaux **robots industriels** seront installés dans des usines à travers le monde, selon une étude de la Fédération internationale de la robotique (IFR). Aujourd'hui, la croissance la plus forte dans l'industrie de la robotique se trouve en Asie, la Chine étant le premier marché mondial.

En 2017, les installations de robots devraient augmenter de 21% dans la région Asie-Australie, de 16% sur le Continent américain et de seulement 8% en Europe. L'installation de robots dans les usines répond à la demande de cycles économiques plus rapides et à l'exigence de produire avec une plus grande flexibilité adaptée à la demande des clients dans tous les secteurs de fabrication.

1, 7 million new **industrial robots** by 2020

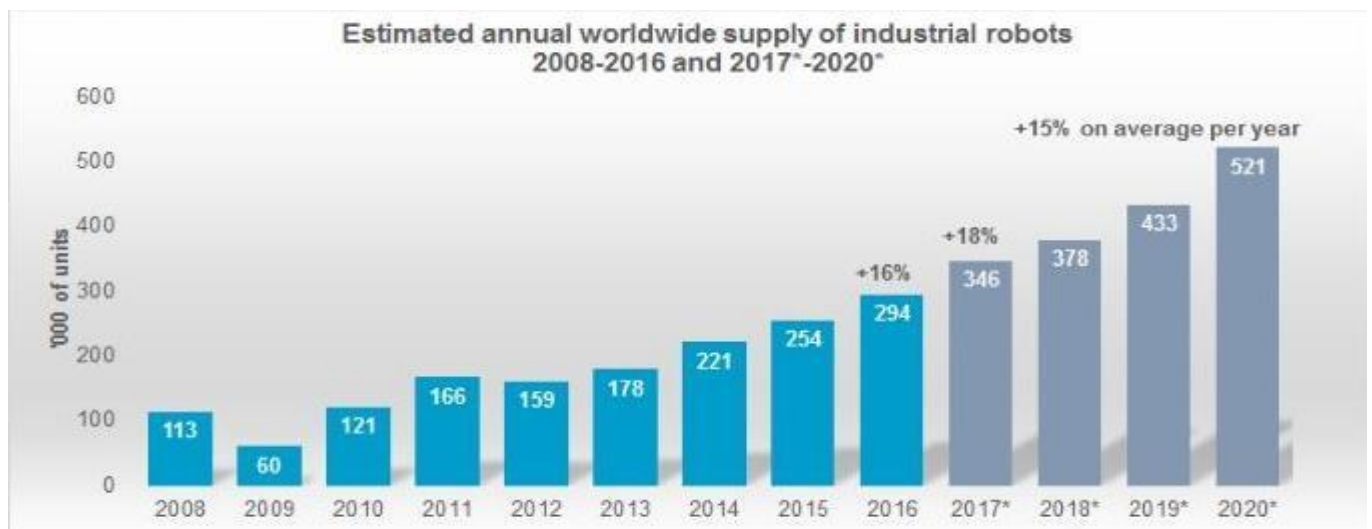


Figure 3 : Stock mondial des robots industriels de 2008j jusqu'à 2020

## I.14. microcontrôleurs dans les systèmes embarqués

### I.14.1. Définition

Le microcontrôleur est fondamentalement un ordinateur qui est placé sur une simple puce de circuit intégré. Il est composé d'une mémoire, d'un processeur aussi bien que des interfaces d'entrée/sortie. Le microcontrôleur est programmé pour exécuter certaines tâches, ce qui signifie que s'il y a un besoin de changer ou améliorer cette fonctionnalité, il suffit d'installer un nouveau programme sur la puce.

### I.14.2. Types des microcontrôleurs

Actuellement, il existe plusieurs types des microcontrôleurs, nous citons quelques un afin d'éclaircir l'évaluation de ce produit scientifique et académique:

- Les microcontrôleurs Atmel AVR (ATméga, ATtiny, etc.).
- Les microcontrôleurs PIC de technologie micro-puce (PIC16, PIC24, etc.).



- Les microcontrôleurs basés sur la technologie ARM.

### **I.14.3. Caractéristiques principale**

- Capacité de la mémoire : qui détermine la taille maximale du programme qui peut être intégré à la puce.
- Fréquence d'opération du processeur : qui détermine la vitesse d'opération de la puce.
- Capacité de mémorisation des données : combien de données peuvent être utilisées dans le programme.
- Nombre de broches entrée/sortie et leurs fonctions : différentes broches ont différentes fonctions
- Nombre de timer : important pour le critère de poursuite du temps.
- Consommation d'énergie : important pour les applications portables.

### **I.14.4. Architecture de microcontrôleur**

Il existe deux formats différents pour stocker des données en mémoire dans un circuit intégré. Ils sont appelés "Princeton" (qui est également connu comme "Von-Neuman" et "Harvard."), Architecture Princeton simplifie l'utilisation et l'accès à la mémoire en effectuant deux instructions et de données le long du même bus et de les stocker dans la même mémoire unité. Une architecture Harvard conserve des instructions et des données, excepté dans les unités de mémoire séparées chacune reliée par des bus distincts. L'architecture Princeton est plus simple et permet une utilisation plus efficace de la mémoire. Cependant l'architecture Harvard est plus rapide et peut se déplacer instructions et de données simultanément.

- Architecture Von Neumann : instructions et données dans une seule mémoire architecture dépréciée, lent.
- Architecture Harvard : instructions et données dans deux mémoires séparées architecture très utilisée car plus rapide (l'UC a accès simultanément à l'instruction et aux données associées), structure interne plus complexe.

## **I.15.Conclusion**

Ce premier chapitre présente des notions sur les systèmes embarqués parce qu'ils représentent le cœur de notre projet. Il est divisé en sept catégories : caractéristiques, complexité, contrôle, sécurité, architecture, classification. Ainsi nous a permis de connaître les caractéristiques du robot et son stock mondial industriel. De plus, il nous donne une occasion pour voir l'importance des microcontrôleurs dans les cartes électroniques, leurs types et caractéristiques.

Dans le chapitre suivant nous présentons tout ce qui est matériel pour réaliser notre plateforme robotique.



## Chapitre II : Outils nécessaires pour la réalisation

### II.1. Introduction

Le système embarqué est appliqué dans le temps réel de contrôle-commande d'un robot, réalisant une activité spécifique, dans notre cas un robot qui suit une ligne noire et détecte des obstacles. Pour réaliser ce mobile il faut présenter plusieurs composants essentiels cité par la suite :

- Carte Arduino
- Capteur ultrason
- Capteur infrarouge
- Moteur à courant continu et L293D
- Afficheur LCD

### II.2. Présentation du Kit d'Arduino

Le projet Arduino est créé en 2005 par *Massimo Banzì* avec ses étudiants *Hernando Barragan* et *Casey Reas* en Italie, ils ont pensé à créer un outil électronique simple et non couteux destiné aux gens non spécialisés au domaine, ceci était une base pour la création de la carte Arduino.

Le Kit Arduino est une carte électronique, d'une taille légèrement inférieure à celle d'une carte de crédit et disponible à l'achat. Elle est contrôlée par un microcontrôleur et d'autres composants électroniques : Les LEDs, les entrées/sorties analogiques, numériques, un quartz de 16MHz, les blocks d'alimentations, la masse, les mémoires régulateur de voltage, une broche raccordée à son entrée, un bouton reset, une sortie et une entée série ainsi d'un port USB.

Le branchement de notre réalisation électronique est schématisé comme suit. Le Kit Arduino est relié avec tous les dispositifs nécessaires à savoir les organes mécaniques (moteurs, actionneurs, servomoteurs...), numériques (afficheur graphiques couleur, afficheurs alphanumériques...), analogiques (LED, transistor, diode ...).

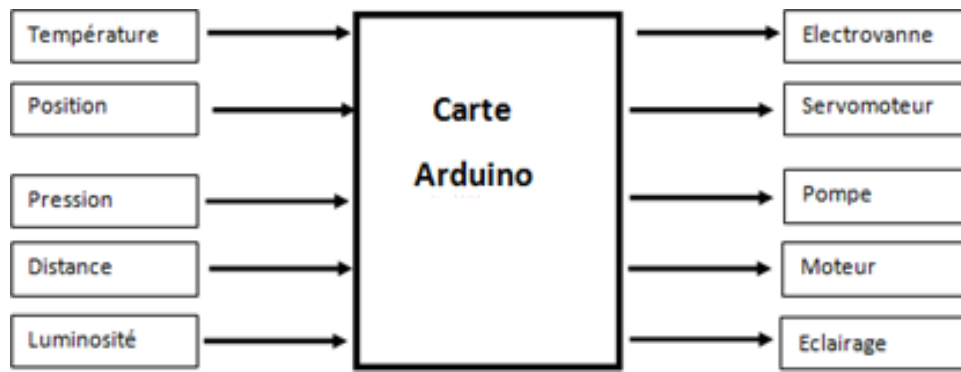


Figure 4 : Interaction de la carte arduino avec les périphériques

### II.2.1. Types d'Arduino

Actuellement, il existe plus de 20 versions de module Arduino, nous citons quelques un afin d'éclaircir l'évaluation de ce produit scientifique et académique:

- ❖ Le NG d'Arduino, avec une interface d'USB pour programmer et usage d'un ATmega8.
- ❖ L'Arduino Mini, une version miniature de l'Arduino en utilisant un microcontrôleur ATmega168.
- ❖ L'Arduino Nano, une petite carte programme à l'aide porte USB cette version utilisant un microcontrôleur ATmega168 (ATmega328 pour une plus nouvelle version).
- ❖ L'Arduino Bluetooth, avec une interface de Bluetooth pour programmer en utilisant un microcontrôleur ATmega168.
- ❖ L'Arduino Diecimila, avec une interface d'USB et utilise un microcontrôleur ATmega168.
- ❖ L'Arduino Mega, en utilisant un microcontrôleur ATmega1280 pour I/O additionnel et mémoire.
- ❖ L'Arduino UNO, utilisations microcontrôleur ATmega328.
- ❖ L'Arduino Mega2560, utilisations un microcontrôleur ATmega2560, et possède toute la mémoire à 256 KBS. Elle incorpore également le nouvel ATmega8U2.
- ❖ L'Arduino Leonardo, avec un morceau ATmega32U4 qui élimine le besoin de raccordement d'USB et peut être employé comme clavier.
- ❖ L'Arduino Esplora : ressemblant à un contrôleur visuel de jeu, avec un manche et son des intégrées pour le bruit, la lumière, la température, et l'accélération.

### II.2.2. Type d'Arduino : Carte Uno

Dans notre projet nous avons utilisé la carte Arduino Uno à cause de sa simplicité ainsi qu'il ne demande pas une solide connaissance dans le domaine d'électronique.

Le modèle UNO de la société ARDUINO est une carte électronique dont le cœur est un microcontrôleur ATMEL de référence ATmega328P. Il possède 6 entrées analogiques et 14 entrées/sorties numériques, un quartz de 16MHz, une alimentation de 9V, une sortie d'alimentation 3V et une autre 5V, la masse, un régulateur 5V et une broche raccordée à son entrée, un bouton reset, une sortie et une entrée série ainsi d'un port USB (voir figure 5 et table 2).

L'intérêt principal des cartes Arduino (d'autres modèles existent : Méga, Nano...) est leur facilité de mise en œuvre. Un environnement de développement (IDE), s'appuyant sur des outils open-source, est fourni. En outre, charger le programme compilé dans la mémoire du microcontrôleur se fait très simplement (via par port USB) dans cet IDE.

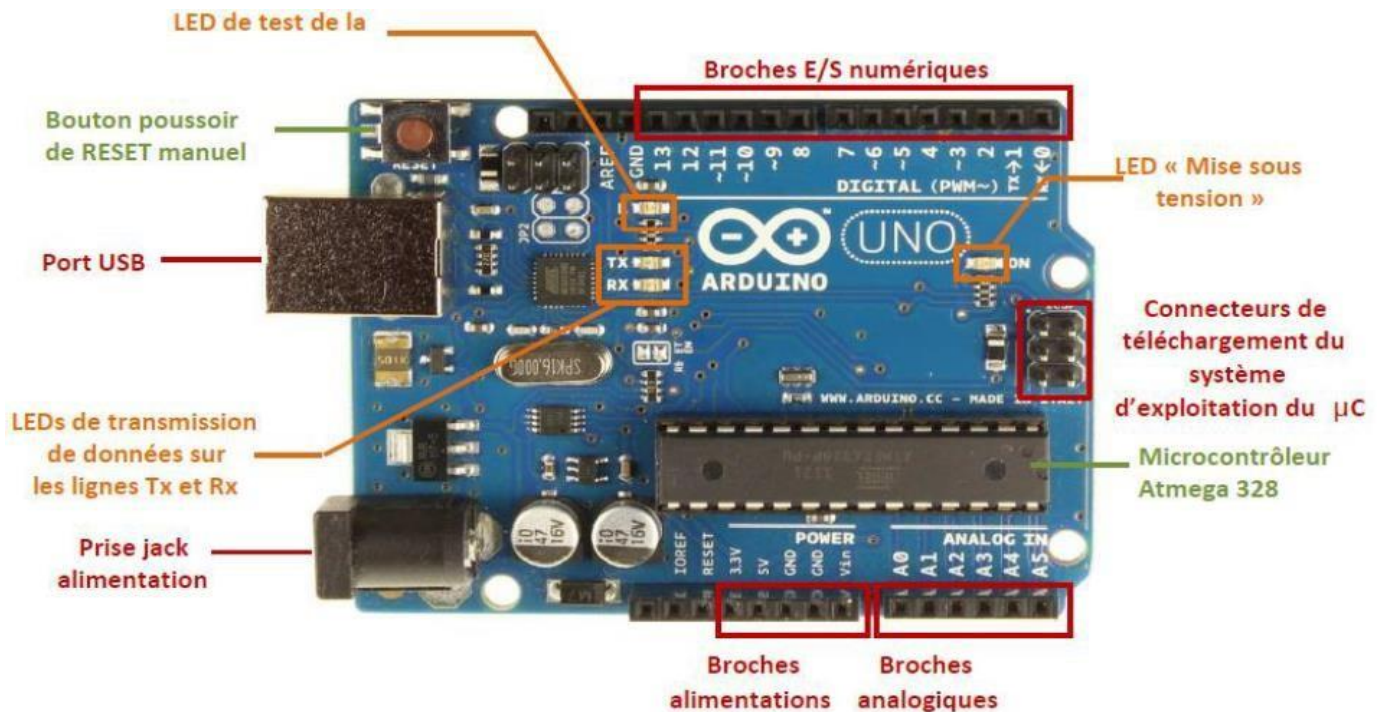


Figure 5: Carte Arduino Uno

Table 2 : Caractéristiques d'arduino uno

Microcontrôleur	ATmega328P
Tension de fonctionnement	5V
Tension d'entrée recommandée	7-12V
Limite de tension d'entrée	6-20V
Broches numériques	14
Broches analogiques	6
Mémoire flash	16KB

### II.2.3. Microcontrôleur d'Arduino uno : ATmega328P

L'ATMega328 est un microcontrôleur 8bits de la famille AVR dont la programmation peut réaliser en langage C/C++. Il est cadencé à 16Mhz et possédant 32ko, de mémoire programme et 2ko de mémoire RAM est un circuit intégré qui rassemble sur une puce plusieurs éléments complexes dans un espace réduit au temps des pionniers de l'électronique.



Figure 6 : Microcontrôleur ATMega328

Le microcontrôleur ATMega328 est constitué par un ensemble d'éléments qui ont chacun une fonction bien déterminée. Il est en fait constitué des mêmes éléments que sur la carte mère d'un ordinateur. Globalement, l'architecture interne de ce circuit programmable se compose essentiellement sur :

- **La mémoire Flash** : C'est celle qui contiendra le programme à exécuter. Cette mémoire est effaçable et réinscriptible mémoire programme de 32Ko.
- **RAM** : c'est la mémoire dite "vive", elle va contenir les variables du programme. Sa capacité est 2 ko.

- **EEPROM** : C'est le disque dur du microcontrôleur. On y enregistre des informations qui ont besoin de survivre dans le temps, même si la carte doit être arrêtée. Cette mémoire ne s'efface pas lorsque l'on éteint le microcontrôleur ou lorsqu'on le reprogramme.

## II.2.4. Logiciel de programmation Arduino

Pour programmer la carte Arduino, nous avons utilisé le logiciel Arduino prévu à cet effet. Il est développé pour Windows, Mac et Linux.

Ce logiciel de programmation des modules ARDUINO est une application Java, libre et multi-plateformes. Il est fourni un environnement de développement (IDE) avec un éditeur de source, les opérations de compilation et téléchargement dans la mémoire du microcontrôleur étant ramenées à des clicks sur des boutons dans IDE (très simple), qui peut transférer le programme au travers de la liaison USB. Le langage de programmation utilisé est un **mélange de C et de C++**, restreint et adapté aux possibilités de la carte.

Voici maintenant l'interface du logiciel, disponible gratuitement sur internet :

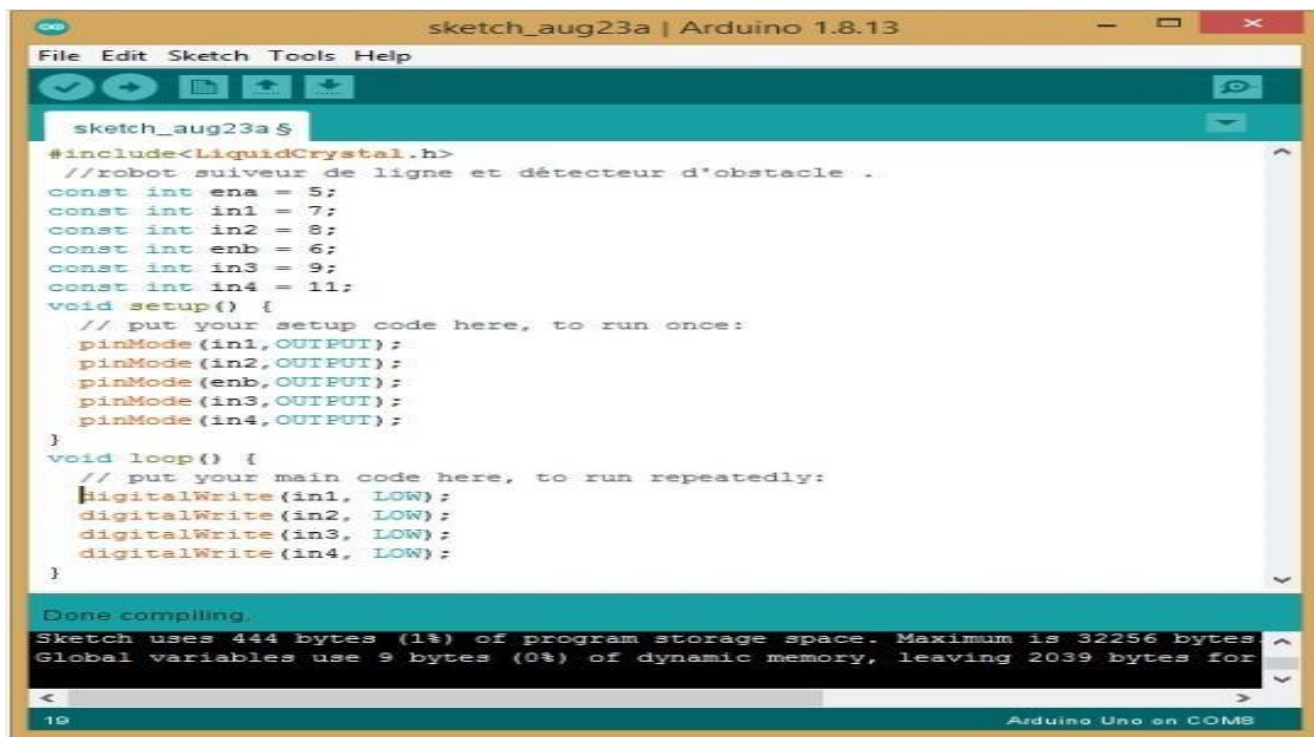


Figure 6: Structure de base d'un programme dans le logiciel Arduino

- Structure d'un projet ARDUINO

L'outil impose de structurer l'application de façon spécifique. Le compilateur utilisé est AVR GCC (compilateur C/C++ pour processeur AVR). Le programme principal (fonction main) est imposé, non modifiable, et décrit ci-dessous.

Les seules parties que nous avons développées spécifiquement sont :

- **la fonction setup ()** : constitue la partie initialisation, elle doit contenir les initialisations (times, interrupts ...).
- **la fonction loop ()** : constitue la partie exécution du code, elle est une fonction répétée indéfiniment.

## II.3. Capteurs utilisés

### II .3.1. Capteur infrarouge

#### *a) Descriptions du capteur infrarouge :*

Un capteur infrarouge (optique) se présente comme un équipement électronique capable de réagir avec les rayonnements infrarouges renvoyé par les objets qui lui font face. Ces rayonnements se situent en dehors du spectre de la lumière visible : ils sont donc invisibles à l'œil nu.



Figure 7: Image d'un capteur infrarouge

La ligne noire correspond à la sortie haute (HIGH) tandis que la ligne blanche correspond à la sortie basse (LOW). Ces deux signaux vont nous permettre de savoir si le robot est écarté de la ligne dessinée afin de corriger sa trajectoire à l'aide des 2 moteurs utilisés.

Le réglage de la sensibilité de chaque détecteur est fait à l'aide de deux potentiomètres :

- Dans le sens horaire pour augmenter la distance de détection.
- Dans le sens antihoraire la distance de détection diminue.

La distance de détection de ce module est de 2 à 30 cm (ou plus selon les modèles), l'angle de détection est de 35°. Cette distance de détection peut être contrôlée par le potentiomètre. Ce module contient deux LED infrarouge qui vont éclairer le sol et deux détecteurs de la lumière réfléchi pour distinguer le noir du blanc.

*b)* Les broches du capteur infrarouge :

- Vcc : connecté avec 5V ou 3,3V de la carte arduino.
- GND : connecté avec la masse de la carte arduino.
- OUT1 : connecté avec un pin de la carte arduino.
- OUT2 : connecté avec un pin de la carte arduino.

## II .3.2. Capteur ultrason

*a)* Descriptions du capteur ultrason :

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.



Figure 8: image d'un capteur ultrason

*b)* Broches du capteur ultrason :

- **Vcc** : pour l'alimentation (à connecter avec le 5v de la carte arduino).
- **Trig** : pour l'émission (à connecter dans un pin de la carte arduino).
- **Echo** : pour la réception (à connecter dans un pin de la carte arduino).
- **GND** : la masse de l'alimentation (à connecter avec le GND de la carte arduino).



### c) Caractéristiques du capteur ultrason :

Le capteur à ultrasons HC-SR04 est capable de mesurer la distance des objets situés de 2cm à 400cm du capteur avec une précision de 3mm. Le capteur est composé d'un émetteur d'ultrasons, d'un récepteur et du circuit de commande, nous avons d'autres caractéristiques citées par la suite (voir la table 3) :

- Dimensions : 45 mm x 20 mm x 15 mm
- Plage de mesure : 2 cm à 400 cm
- Résolution de la mesure : 0.3 cm
- Angle de mesure efficace : 15 °
- Largeur d'impulsion sur l'entrée de déclenchement : 10  $\mu$ s (Trigger Input Pulse width)

Table 3: Paramètres du capteur ultrason

Paramètre	Min	Type	Max	Unité
Tension d'alimentation	4.5	5.0	5.5	V
Courant de repos	1.5	2.0	2.5	mA
Courant de fonctionnement	10	15	20	mA
Fréquence des ultrasons	-	40	-	kHz

NB : la borne GND doit être connectée en premier, avant l'alimentation sur Vcc.

### d) Fonctionnement du capteur ultrason :

Déclenchement d'une mesure par envoi sur l'entrée Trigger d'une impulsion de 5V durant 10  $\mu$ s. Le capteur émet alors une série de 8 impulsions ultrasoniques à 40 kHz Puis il attend le signal réfléchi, lorsque celui-ci est détecté, une impulsion de durée proportionnelle à la distance mesurée est envoyée sur la sortie "Echo".

### e) Distance de détection du capteur ultrason :

Ce capteur se dispose d'un émetteur et d'un récepteur situés dans le même boîtier, l'émetteur envoie des ondes acoustiques dont la fréquence est grande et qui se propage à la vitesse du son ( $v=380 \text{ m.s}^{-1}$ ), ces ondes vont se réfléchir sur l'obstacle et revenir au récepteur. Le temps d'un aller-retour permet de connaître la distance entre l'obstacle et la source à l'aide de la relation :  $v=d/t$  donc  $d=v.t$  avec  $t$  : est le temps d'un aller-retour d'où :

$$D=d/2$$

D : est la distance séparant l'objet de l'obstacle.



## II.4. Actionneur utilises

### II.4.1. Moteur à courant continu

#### a) Description du moteur :

Le moteur à courant continu (MCC) permet de convertir l'énergie électrique en mouvement mécanique. Il est un élément majeur de la robotique puisqu'il permet le robot de faire un mouvement ou un déplacement. C'est pour cela nous avons la retrouve dans tous les robots.

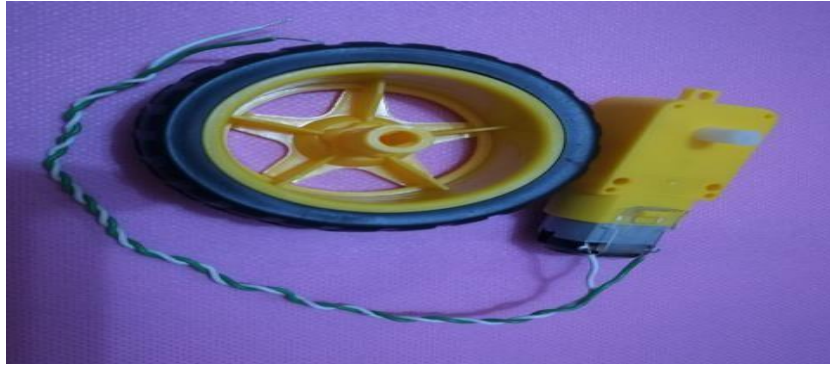


Figure 9: Image d'un roue et Moteur à courant continu

Ce moteur est constitué de trois parties principales : le stator (l'inducteur), le rotor (l'induit), et les dispositifs : armature, collecteur, Balais. L'inducteur est formé d'aimants permanents, seul le rotor est alimenté par 2 fils. Le balais (ou charbons montés, « brush » en Anglais) est un système de frottement permet d'alimenter le rotor frotté sur les contacts en rotation (le collecteur). Il possède deux bornes, l'application d'une tension continue à ces bornes conduit à la rotation du moteur dans un sens, l'inversion de la polarité de la tension le conduit à tourner dans le sens contraire. Aussi il s'appelle, un moteur à courant continu transforme une tension continue en un couple. Nous avons utilisé en général des moteurs à excitation constante.



Figure 10: image représente la constitution du moteur à courant continu

#### b) Caractéristiques du moteur :

- Alimentation : de 3V à 6V

- Consommation : environ 120mA
- Diamètre de la roue : 56mm
- Vitesse de rotation: 240tr/min sous 6V
- Vitesse de mouvement : 489m/min sous 6V
- Taille de la roue : 65mm\*26mm
- Axe de roue: 5,3mm\*6,66mm

#### II.4.2. Pilote de moteur : L293D (pont H)

Le L293D est le pilote de moteur le plus utilisé pour les petits robots. Il possède quatre demi- ponts en H, soit deux ponts complets permettant de commander 2 moteurs. Cette version L293D intègre les diodes de protection, inutile de les ajouter à l'extérieur. Il convient pour des courants d'environ 1A. Dans son principe de base, le pont H est un assemblage de 4 transistors (2PNP et 2 NPN) monté de telle que le courant puisse passer soit dans un sens, soit dans l'autre au travers de la charge (un moteur à courant continu). Il est utilisable pour la commande des moteurs, c'est-à-dire l'ajustement sur la vitesse.



Figure 11 : Image d'un circuit L293D

Notre robot est constitué de quatre moteurs courant continus, pour cela nous avons choisi ce shield qui contient le circuit L293D, ce contrôle deux moteurs gauches et le deuxième est deux moteurs droits.

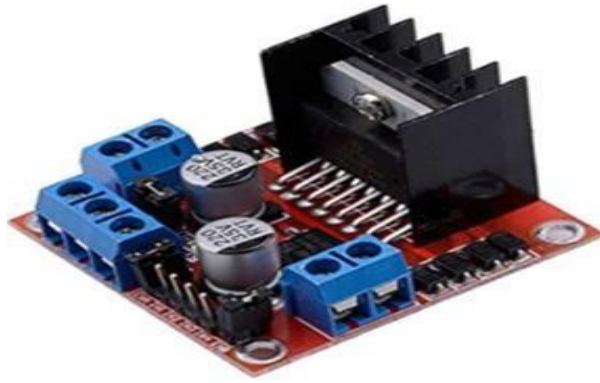


Figure 12: Image d'un sheild

### II.4.3. Présentation du circuit L293D

Ce circuit permet de réaliser 4 ponts en H donc d'alimenter 2 moteurs (moteurs gauche, moteurs droit) qui peuvent tourner dans les 2 sens. Il caractérisé par une tension d'alimentation des moteurs est comprise entre 4.3 et 36 V, un courant d'alimentation du moteur est de 1 A (600 mA pour le L293D) mais le circuit supporte des piques de 2 A (1.2 A pour le L293D), des entrées « Enable » permettent d'activer la commande des moteurs, il y a une alimentation Vss pour la partie logique et les transistors et une alimentation Vs Pour le moteur.

Le pont en H se compose de deux transistors NPN et de deux transistors PNP sui fonctionnent en saturé/bloqué.

## II.5. Afficheur LCD

### a) Description d'un afficheur LCD :

L'afficheur LCD est en particulier une interface visuelle entre un système (projet) et l'homme utilisateur.

« LCD est l'abréviation du terme anglais "*Liquid Crystal Display*" qui signifie en français " Écran à cristaux liquides". D'où afficheur LCD ».



Figure 13: Image d'un écran d'afficheur LCD

## b) Rôle d'un afficheur LCD :

Les afficheurs LCD transmettent les informations utiles d'un système à l'utilisateur. Il affichera donc des données susceptibles d'être exploitées par l'utilisateur d'un système. Il est devenu indispensable dans les systèmes techniques qui nécessitent l'affichage de paramètres de fonctionnement. Grâce à la commande par un microcontrôleur, ces afficheurs permettent de réaliser un affichage des messages aisés. Ils permettent également de créer ses propres caractères.

## c) Broches de l'afficheur LCD

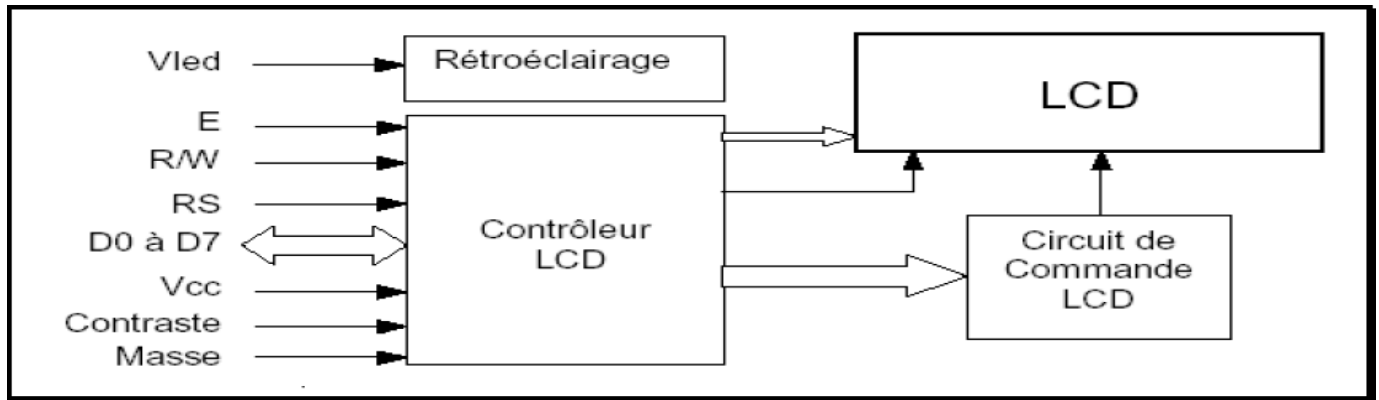


Figure 14: Schéma Afficheur LCD

- **VCC, Masse** : alimentation de l'afficheur LCD. Un afficheur LCD s'alimente en 0V-5V.
- **Contraste** : entrée permettant de régler le contraste de l'afficheur LCD. Il faut appliquer une tension continue réglable (entre 0V et 5V) à l'aide d'un potentiomètre.
- **V led** : différence de potentiel permettant de commander le rétro éclairage.
- **E** : entrée de validation (ENABLE), elle permet de valider les données sur un front descendant. Lorsque E=0 alors le bus de données est à l'état haute impédance.
- **RS** : Register Select cette entrée permet d'indiquer à l'afficheur si l'on souhaite réaliser une commande (RS=0) par des instructions spécifiques ou écrire une donnée (envoi du code du caractère à afficher) sur le bus (RS=1).
- **R/W** : entrée de lecture (R/W=1) et d'écriture (R/W=0). Lorsqu'on commande l'afficheur LCD il faut se placer en écriture.
- **D7...D0** : bus de données bidirectionnel, il permet de transférer les instructions ou les données à l'afficheur LCD.

## II.6. CONCLUSION

Dans ce chapitre, nous avons présenté les notions de fonctionnement du système embarqué utilisé, quelques informations sur le kit arduino, leurs composants, types et caractéristiques. Aussi la description des capteurs infrarouge et ultrason, les moteurs à courant continu et L293D et le rôle de l'afficheur LCD. Ce chapitre nous a permis de maîtriser les options de notre matériel utilisé. Encore nous avons vu le logiciel pour programmer l'arduino et le type de programmation utilisé dans notre projet.

Dans le chapitre suivant nous allons présenter réalisation d'un robot selon trois parties : électronique, mécanique, informatique.

# Chapitre III : Résultats de la réalisation électronique et mécanique

## III.1. Introduction

Ce chapitre présente les différentes étapes utilisées qui concerne notre réalisation pratique d'un robot qui doit suivre une ligne noire tracée sur le sol peu importe un sens à l'aide d'un capteur optique et détecte des obstacles par un capteur ultrason qui travaille avec les ondes ultrasonores. Pour réaliser ce prototype nous avons traité trois parties :

- Parties électronique : consiste les matérielles utilisés, leurs rôles et leurs fonctionnements, simulation du montage des composants dans ISIS.

- Partie mécanique : traite la connexion des matérielles et des composants électroniques utilisés et leurs positions dans le robot. Aussi le comportement du robot pour suivre une ligne.

- Partie informatique : contient la description des fonctions de programmation et l'algorithme global du robot.

## III.2. Présentation du projet

L'objectif de notre projet est de réaliser une application dans le domaine du système embarqué, il s'agit d'un système robot capable de suivre une ligne qu'elle soit droite ou courbe. S'il rencontre un obstacle ce dernier doit arrêter en attendant l'élimination de l'obstacle dans une durée limitée, quelque secondes par exemple. Sinon le robot va réagi avec une action d'avertissement sert à clignoter une LED, suivi d'une rotation total et avancement vers le sens opposé.

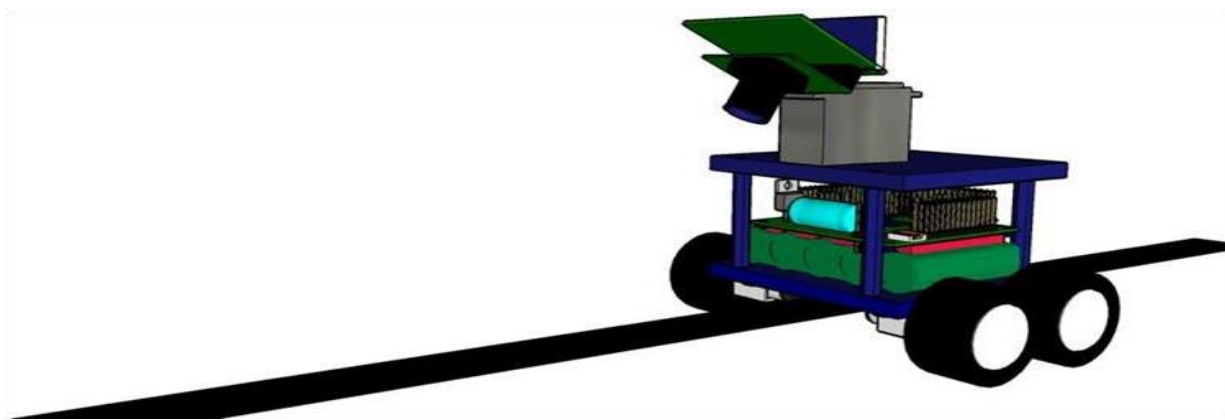


Figure 15: image du robot suit une ligne noire

### III.3. Cahier des charges fonctionnel

- ✓ Fonctions principales:
  - Démarrer
  - S'arrêter
  - Suivre une ligne noire (scotch ou peinture) de largeur 10 mm sur un sol blanc
  - Afficher la distance de robot avec l'obstacle.
- ✓ Fonctions liées aux règlements proposés :
  - Détecter la présence d'un obstacle → Action : Arrêter (10 s) + un demi-tour
  - Élimination d'obstacle avant (10s) → Action : Avancer suivant la ligne
  - Si le robot sort de la ligne → Action : Retour à son trajet

### III-4-Partie électronique

#### III.4.1 Fonctionnement d'Arduino dans le robot

Pour réaliser notre commande robotique, nous avons choisi le circuit programmable "Arduino Uno " qui doit contenir un programme écrit en langage C, ce programme constitué de plusieurs outils et fonctions spéciales à chaque carte électronique liée à l'arduino. Ces cartes sont : Moteur à courant continu, capteur Ultrason, capteur infrarouge, circuit L293 D, afficheur LCD. Pour programmer cette carte électronique par son logiciel arduino IDE nous avons dû relier au PC.

Après avoir relié l'Arduino au PC et installé le logiciel, Nous avons un exemple **blink** qui est le plus utile pour vérifier l'installation du logiciel et la connexion entre le PC et la carte. Ce code permet de faire clignoter une LED sur la broche 13, une LED est déjà présente sur l'Arduino.

```
Blink $
1 // the setup function runs once when you press reset or power the board
2 void setup() {
3   // initialize digital pin 13 as an output.
4   pinMode(13, OUTPUT);
5 }
6
7 // the loop function runs over and over again forever
8 void loop() {
9   digitalWrite(13, HIGH); // turn the LED on (HIGH is the voltage level)
10  delay(1000);             // wait for a second
11  digitalWrite(13, LOW);   // turn the LED off by making the voltage LOW
12  delay(1000);             // wait for a second
13 }
```

Figure 16: Code C exemple blink

Après l'écriture du programme en C qui commence par l'initialisation des variables, la liaison série ou encore la direction des ports dans la fonction **setup** (), et on utilise la fonction **loop** () (qui est une boucle infinie), pour la réalisation de notre objectif. Ensuite nous avons compilé ce programme pour le convertir en langage machine qui vérifie les erreurs, par la suite le charger vers la mémoire programme du microcontrôleur qui contient 2ko. Le programme de commande insérer dans la mémoire d'arduino est constitué de plusieurs outils :

➤ Outils informatiques nécessaires du capteur ultrason :

Nous pouvons voir ici que nous affichons deux valeurs la distance et la durée. La durée représente le temps du parcours de notre signal entre le moment où celui-ci est émis par « trigPin » et celui où il est reçu par « echoPin ».

➤ Outils informatiques nécessaires du capteur infrarouge :

Le programme qui permet de lire sur la console série la valeur de retour du capteur est basé sur une fonction `serial.begin()`, cette fonction s'utilise pour initialiser la liaison Série. Elle configure la vitesse de la communication série entre l'ordinateur et la carte Arduino. Dans la fonction `loop()`, la broche d'entrée est lue et sa valeur est stockée dans la variable `digitalWrite`. Cette valeur vaut HIGH lorsqu'aucun mouvement n'est détecté, car la résistance pull-up reçoit la tension de la broche 5 V. Dès qu'un mouvement survient, le collecteur ouvert fait chuter la tension et indique LOW.

➤ Outils informatiques nécessaires du moteur à courant continu

Pour faire varier la vitesse d'un moteur à l'aide d'Arduino nous avons utilisé le principe de la Modulation de largeur d'impulsion MLI(PWM), déjà configurée, de l'Arduino.

La fonction à utiliser est `analogWrite(broche,valeur)` où le paramètre broche (entier) correspond à la broche sur laquelle nous voulons appliquer le signal PWM et le paramètre valeur correspond la valeur de cette

PWM. Aussi pour récupérer la valeur dans le code, il faut utiliser la fonction `analogRead(broche)` qui renvoie la valeur lue sur 8 bits, la broche est ici A0 : `analogRead(A0)`. De plus nous avons utilisé la fonction `digitalWrite(Numéro broche, Niveau logique)`. Cette fonction permet d'imposer un niveau logique haut ou bas sur la sortie numérique sélectionnée.

➤ Outils informatiques nécessaires pour faire varier des trois LEDs

Pour faire varier l'état de trois LEDs, vous devez utiliser les valeurs de la fonction créée. Cela suffit de lire les entrées analogiques correspondantes aux capteurs, et de les comparer à ce seuil. Si la valeur est inférieure, la variable état de la LED respective est mis à 0, et l'on éteint la LED en mettant la broche à l'état bas (LOW).

➤ Outils informatiques nécessaires pour l'afficheur LCD



Nous avons déclaré la bibliothèque "LiquidCrystal.h" qui se chargera de gérer les timings et l'ensemble du protocole. Elle permet de faire communiquer l'Arduino avec la plupart des écrans à cristaux liquides. Cette bibliothèque est basée sur le pilote Hitachi HD44780 qui gère des écrans facilement identifiables à leur interface 16 broches. En effet, une fois que notre écran LCD est bien paramétré, il nous suffira d'utiliser qu'une seule fonction print(), pour afficher le texte sur l'écran.

### **III.4.2. Mode de fonctionnement des capteurs dans le robot**

Nous avons utilisé dans notre réalisation deux types de capteurs qui sont liés à l'Arduino :

- Un capteur qui permet la détection d'un marquage au sol (capteur de ligne /capteur infrarouge).
- Un capteur qui permet la détection d'un obstacle (capteur ultrason).

#### **III.4.2.1. Capteur infrarouge**

Le capteur de ligne permet en fait de détecter un marquage sombre tracé sur le sol. Il est composé d'un phototransistor et d'une diode infrarouge dirigée vers le sol, la diode produit un rayon infrarouge qui sera absorbé par la ligne noire au sol ou au contraire, ce rayon sera réfléchi par les zones plus claires. Le phototransistor adapté à chaque diode détecte ou non le rayonnement infrarouge. La diode et le phototransistor sont indépendants et permettent de trouver avec précision la position du Micro-robot par une ligne noire tracée au sol.

La diode est sous tension émet toujours un signal. Cas de présence de ligne noire nous n'aurons pas de réflexion, alors le phototransistor sera bloqué : La sortie  $S = 0$ .

-Voir la Figure 18 : Cas de présence de ligne noire

Cas d'absence de ligne noire nous aurons une réflexion, alors le phototransistor sera saturé : La sortie  $S = 1$ .

-Voir la Figure 19 : Cas d'absence de ligne noire

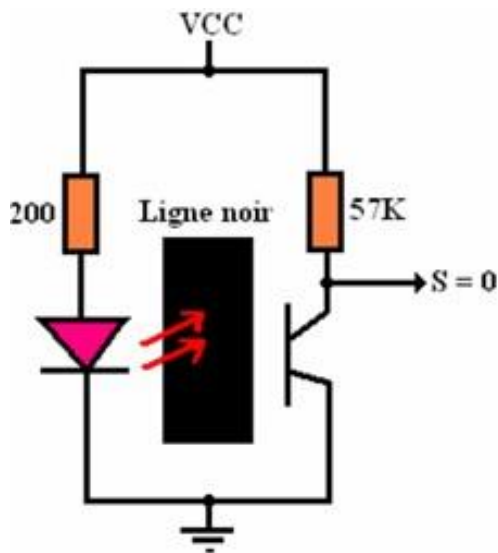


Figure 17 : Cas de présence de ligne noire

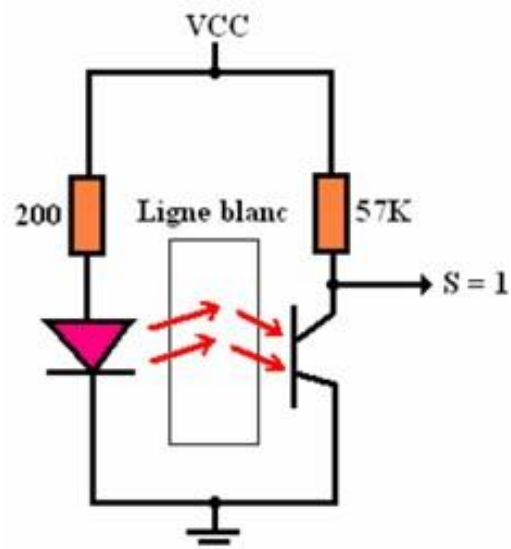


Figure 18: Cas d'absence de ligne noire

### III.4.2.2. Capteur d'obstacle : Ultrason

Pour mesurer la distance entre notre robot et un obstacle situé à l'avant, nous avons utilisé le capteur de détection HC-SR04, sa technique consiste à utiliser les ondes ultrason. Il émet à intervalles réguliers de courtes impulsions sonores à haute fréquence par le chirp. Ces impulsions se propagent dans l'air à la vitesse du son. Lorsqu'elles rencontrent un objet, elles se réfléchissent et reviennent sous forme d'écho au capteur. Celui-ci calcule alors la distance le séparant de la cible sur la base du temps écoulé entre l'émission du signal et la réception de l'écho. Un obstacle peut être détecté par : un capteur ultrason, IR distant, une caméra, ...

Dans notre projet nous avons utilisé le capteur ultrason de type HC-SR04.

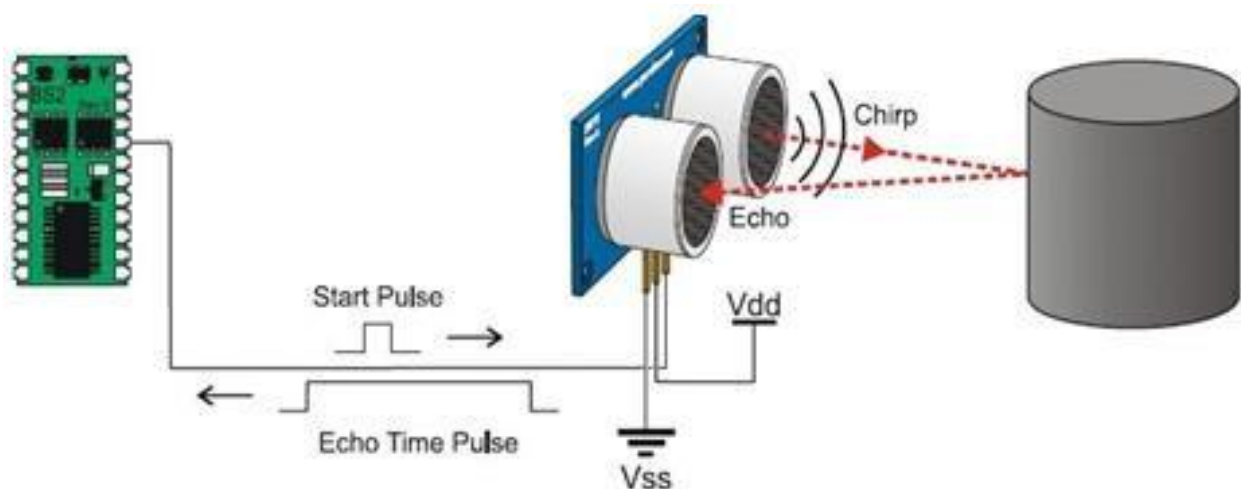


Figure 19: schéma illustre le fonctionnement du capteur ultrason

La distance des capteurs à ultrasons Micro-Sonic étant déterminée par le temps de propagation des ultrasons et non par leur intensité. Ils sont disponibles pour des portées de 20 mm à 10 m. Ce module est caractérisé par le fonctionnement suivant :

- Angle de mesure : 15 degrés.
- Signal d'entrée–trig: déclenchement de la mesure : impulsion 10 $\mu$ STTL.
- Signal de sortie – echo : impulsion lorsque l'écho est reçu: Signal TTL.

### III.4.3. Moteurs à courant continu et pont H L293D

#### III.4.3.1. Moteurs à courant continu lié à une roue

Nous avons choisi deux moteurs à courant continu standards pour actionner les roues du robot. Cependant, leurs vitesses maximales sont trop élevées pour un mouvement stable. C'est pourquoi nous avons été obligées de designer un réducteur de vitesse à base d'engrenages afin de réduire la vitesse de rotation de ces moteurs. Par la suite, nous allons spécifier les pièces de modélisation du réducteur. Il y a deux supports extérieurs le premier avec protection pour le moteur à courant continu, le deuxième avec troue pour la roue. Il y a les engrenages du réducteur, l'engrenage du moteur, les axes de rotation et les connecteurs et la roue.

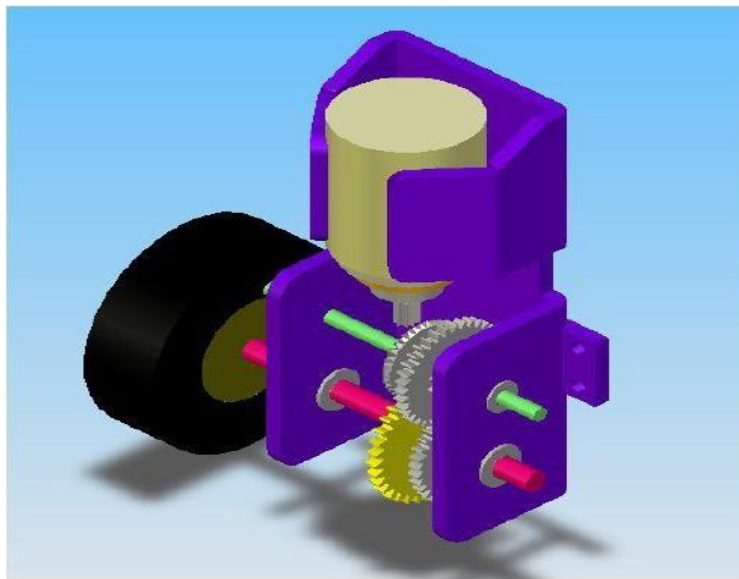


Figure 20: Image représente un ensemble moteur, réducteur et roue

Nous avons utilisé 4 moteurs à courant continu pour le déplacement du robot dans les deux sens. Lorsqu'il n'y a pas d'obstacle et il y a détection de la ligne noire, les 4 moteurs vont tourner dans le même sens, si ce n'est pas le cas (présence d'obstacle), les moteurs vont s'arrêter.

Lorsqu'il y a un virage, la solution qu'on a choisie pour faire tourner le robot soit à gauche ou bien à droite est la suivante :

Pour un virage à droite : le moteur gauche va tourner vers l'avant tandis que le moteur droit va tourner vers l'arrière.

Pour un virage à gauche : le moteur gauche va tourner vers l'arrière tandis que le moteur droit va tourner vers l'avant.

Nous utilisons quatre motoréducteurs à courant continu. Ils doivent être alimentés entre 12V et 36V, nous utiliserons donc l'alimentation 18V (2\*9V) pour alimenter les moteurs. Car chaque moteur à courant continu alimente entre 3V et 6V.

Après la connexion entre la roue et motoréducteur, le dispositif qui nous a permis d'inverser le sens de rotation est le circuit intégré L293D.

### III.4.3.2. Pilote de moteur : L293D (pont H)

Pour contrôler les moteurs à courant continu, nous avons utilisé le circuit intégré L293D permettant de piloter deux moteurs à courant continu dans les deux sens de rotation ou pour réduire ou accélérer la vitesse du moteur. Le montage de connexion correspond est :

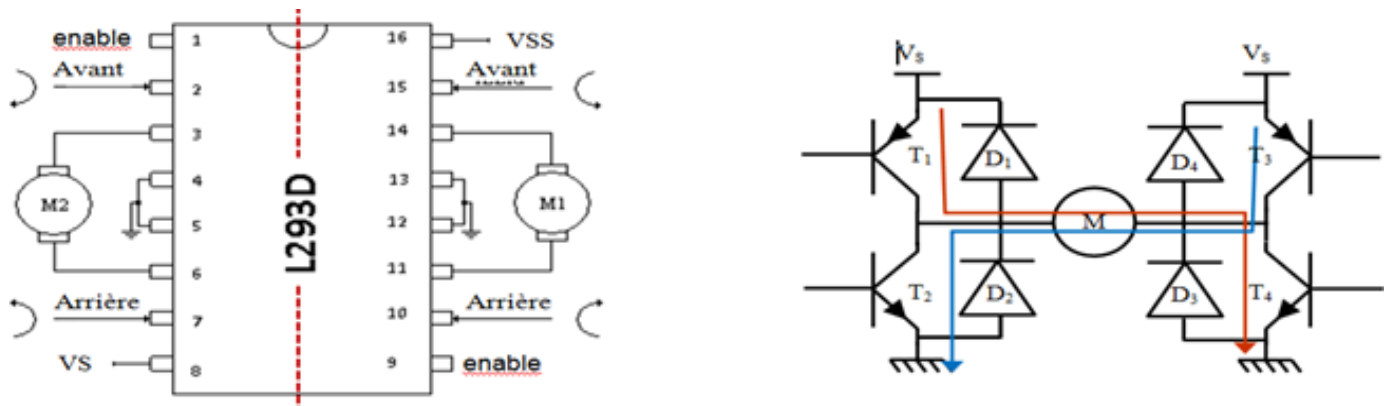


Figure 21: Schéma d'moteurs à courant continu à connecter avec le circuit L293D

- Fonctionnement du montage

-Si on applique un 0 logique sur l'entrée « Enable » (n°1), les 4 transistors sont bloqués et donc équivalents à des interrupteurs ouverts quelles que soient les valeurs (X) données aux entrées n°2 et n°7.

On en conclut qu'il n'y pas de courant qui circule dans le moteur et qu'il ne tourne pas.

-Les sorties n°3 et n°6 sont en haute impédance que l'on note Z.

-On applique un 1 logique sur l'entrée « Enable » et sur l'entrée n°2 et un 0 logique sur l'entrée n°7.

-Les transistors T2 et T3 sont bloqués alors que T1 et T4 sont saturés et se comportent comme des courts-circuits.

-T1 et T4 amène la tension Vs et la masse aux bornes du moteur qui se met à tourner.

-Si on met l'entrée n°2 à 0 et l'entrée n°7 à 1, alors le courant change de sens dans le moteur.

-L'entrée enable peut être utilisée en PWM ce qui permet d'avoir une variation allant de 0 (rotation nulle) à 255 (rotation maximale) pour le moteur dans les deux sens de rotation.

#### III.4.4. Alimentation du robot : Piles

Pour rendre notre robot autonome, nous avons besoin des piles pour alimenter tous ces composants telles que : les moteurs à courant continu, schield, arduino, capteurs et afficheur LCD. Ces piles ont l'avantage de fournir la puissance nécessaire à cette voiture mobile pour déplacer de façon adéquate pendant la durée du parcours de la ligne. Pour cela, nous avons utilisé des piles alcalines de 9V qui ont une énergie fournie

$E_{fournie}$  égale 3815 mWh, une puissance moyenne  $P_{charge}$  égale 250 mW et la durée de consommation environ 15 h pour l'Arduino Uno R3.

#### III.4.5. Afficheur LCD

Nous avons décidé d'ajouter un écran LCD au projet car, bien qu'inutile au bon fonctionnement du robot, il nous permettra de faciliter le débogage de la carte en cas de problèmes. En effet nous pourrons y afficher différentes valeurs : comme la distance détection entre le robot et l'obstacle, vitesse de déplacement et les tensions sur les endroits souhaites.

#### III.4.6 Connexion entre les composants décrits au-dessus

La connexion entre les composants électroniques du robot se fait comme suit :

➤ Connexion entre le capteur infrarouge et l'Arduino

❖ Capteur infrarouge => Arduino

- Vcc                      => 5V de l'arduino
- GND                    => GND de l'arduino
- OUT                    => A3 et A4 de l'arduino

➤ Connexion entre le capteur ultrason et l'Arduino

❖ Capteur ultrason => Arduino

- Vcc                      => 5V de l'arduino
- GND                    => GND de l'arduino
- Trig                    => A1 de l'arduino
- Echo                   => A2 de l'arduino

➤ Connexion entre le Scheild et l'Arduino

❖ Scheild => Arduino

-Pour le moteur gauche :

- ENA    => D3 de l'arduino
- IN1    => D4 de l'arduino

- IN2 => D5 de l'arduino

-Pour le moteur droit :

- IN3 => D6 de l'arduino
- IN4 => D7 de l'arduino
- ENB => D8 de l'arduino
- (+12V) => (+du pile) + VIN Power de l'arduino
- GND => (-du pile) + GND Power de l'arduino
- (+5V) => 5V de l'arduino

➤ Connexion entre afficheur LCD et l'Arduino

❖ Afficheur LCD => Arduino

- ENA => D9 de l'arduino
- D4 => D10 de l'arduino
- D5 => D11 de l'arduino
- D6 => D12 de l'arduino
- D7 => D13 de l'arduino
- E => D2 de l'arduino
- D12 => RS de l'arduino
- Vdd => GND Power de l'arduino
- Vss+RW => 5V Power de l'arduino

### III.4.7 Simulation des moteurs avec le circuit intégré L293D sous le logiciel ISIS

#### PROTEUS

Pour mieux prédire le comportement de notre robot, nous avons réalisé une simulation qui exigée le schéma suivant pour la version basic (détecteur d'obstacle):

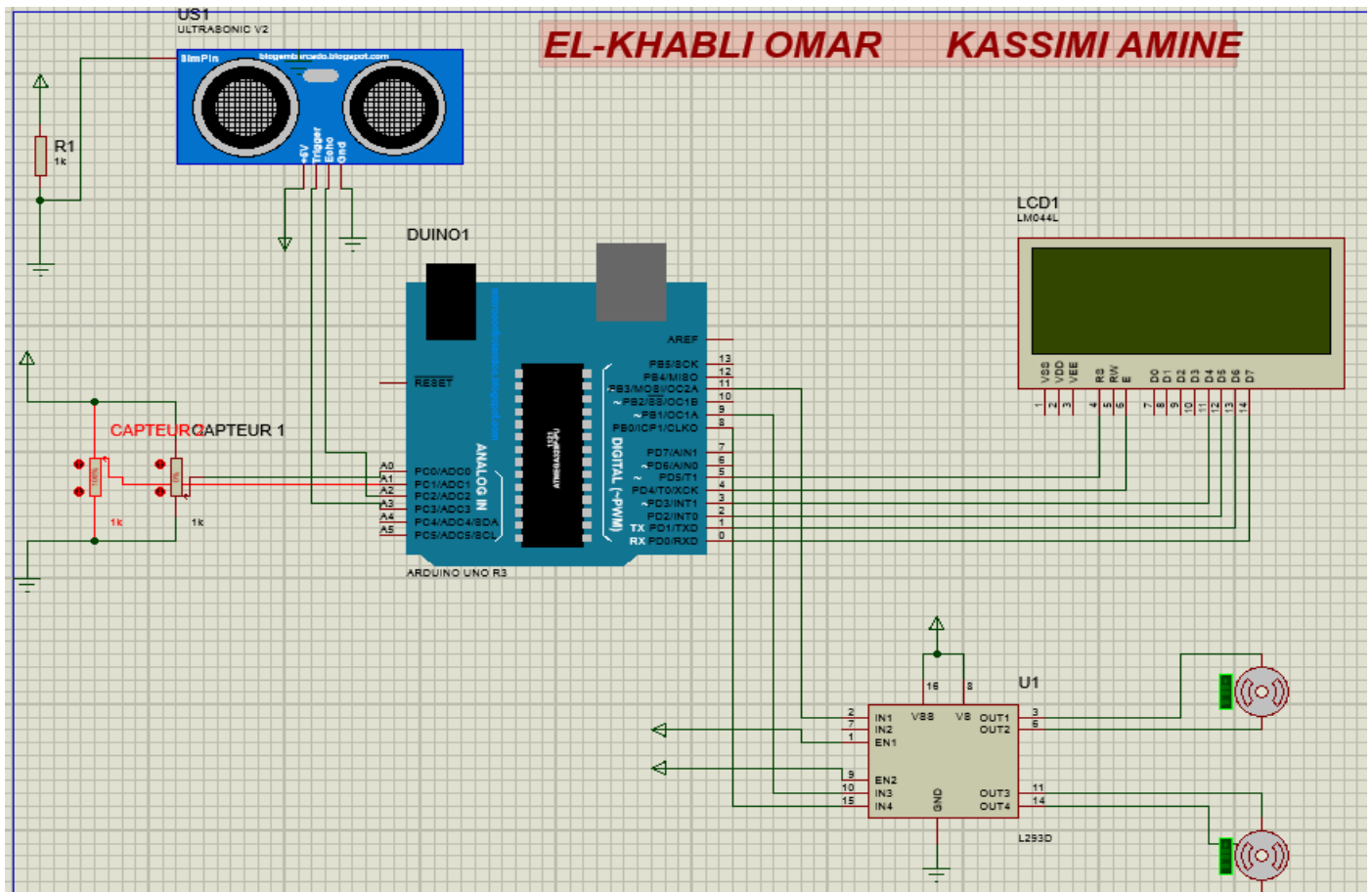


Figure 22: Simulation représente la connexion des composants de notre robot sous ISIS

N.B : Notre robot contient 4 moteurs branché avec un driver contient un circuit L293D, mais dans ce branchement nous avons travaillé seulement sur 2 moteurs.

### III-5-Partie mécanique

La partie mécanique est une des parties qui nous a pris le moins de temps puisque le châssis ainsi que la mécanique (moteur, roue, batterie), nous ont été fournis par les organisateurs du challenge.

#### III.5.1. Conception du robot

Ce robot est équipé des yeux électroniques qui observent une ligne noire et détectent les obstacles, chaque œil électronique est constitué d'un capteur infrarouge de type capteur de réflexion optique (IR) TCRT5000. Les moteurs assurent la propulsion et la direction du robot. Les capteurs optiques situés à l'avant permettent la détection de la ligne.

### III.5.2. Châssis du robot

Le châssis est en plastique de couleur noire, les roues sont également en plastique et positionnées au côté du robot afin d'équilibrer celui-ci, nous a fourni des pions en plastiques positionnés à l'avant ainsi qu'à l'arrière. Le branchement des pièces du robot à quatre roue demande à nous de faire la connexion à l'aide des vis, les écrous hexagonaux, les supports de métal, les entretoises de cuivre hexagonal, les fils. Ce branchement se fait comme suit :

### III.5.3. Branchement du robot

- Position des pièces
  - Branchement entre le moteur, la roue et son fixation sur le châssis



Figure 23: connexion entre le moteur à courant continu et la roue

- Position du driver, les moteurs et son fixation sur le châssis

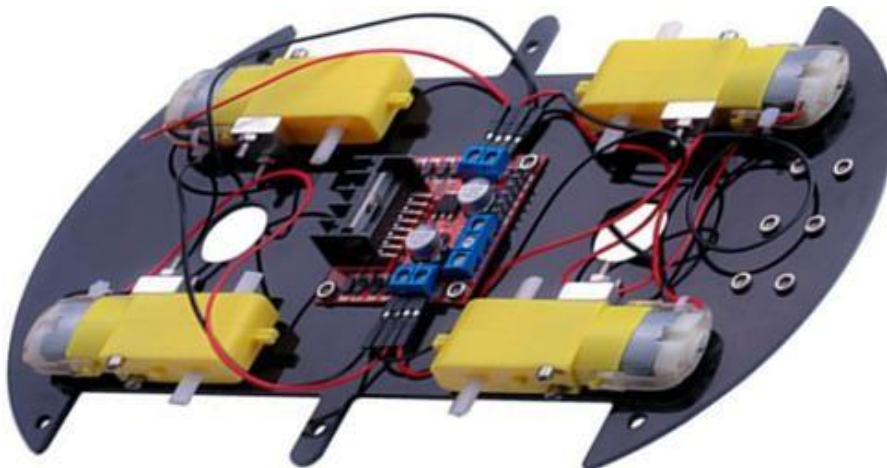


Figure 24: connexion entre les moteurs et le driver



- Position du capteur infrarouge et son fixation sur le châssis



Figure 25: Fixation des trois capteurs optiques sur le châssis

- Position du capteur ultrason et son fixation sur le châssis



Figure 26: Fixation du capteur ultrason sur le châssis

- Position d'arduino, carte de manipulation, boîtier des piles et son fixation sur le châssis

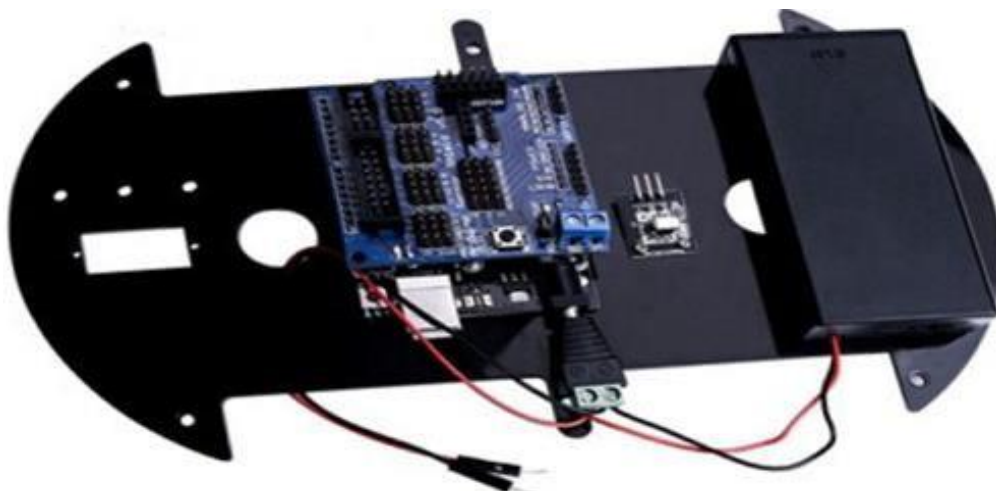


Figure 27: Position de l'arduino, le manipulateur et le boîtier

- Connexion entre les pièces du robot
  - Connexion entre le manipulateur, cadre de réception infrarouge, capteur ultrason

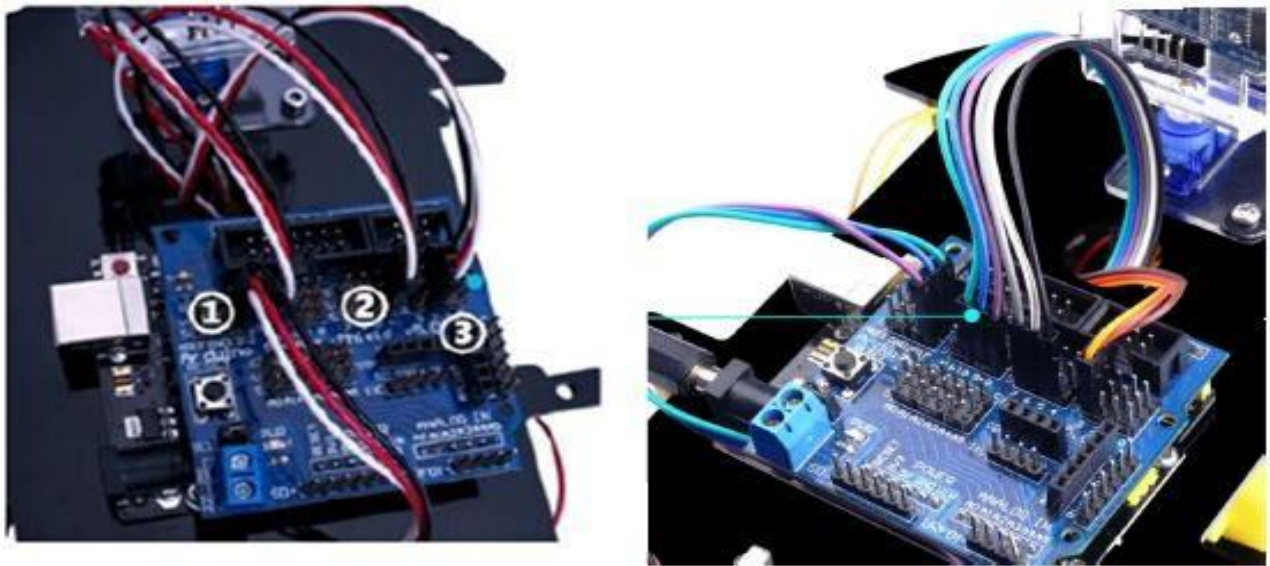


Figure 28: Position de l'arduino, le manipulateur et le boîtier

- Connexion entre les pièces , châssis supérieur et châssis inférieur

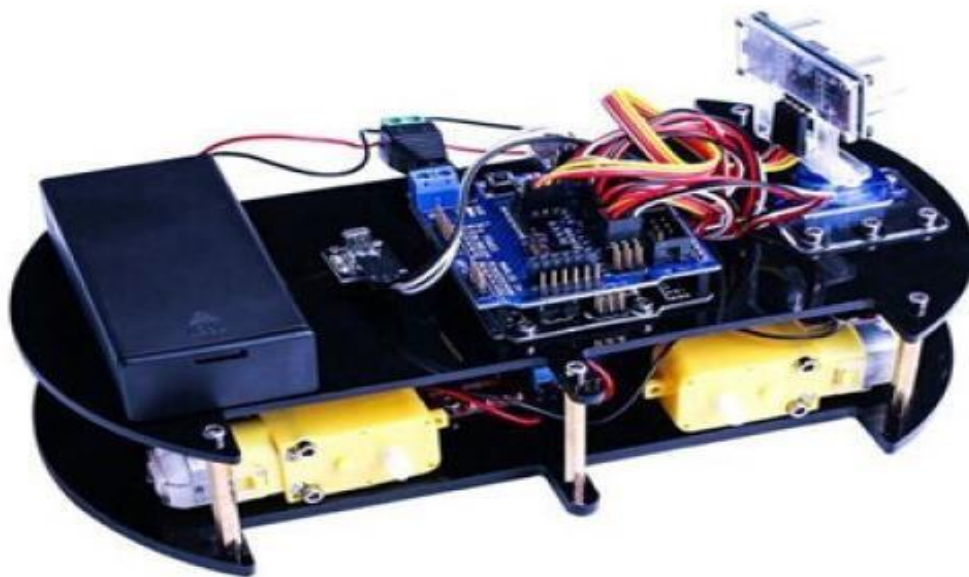


Figure 29: connexion entre les pièces du robot

- Connexion entre le manipulateur et l'afficheur LCD et la forme de la voiture

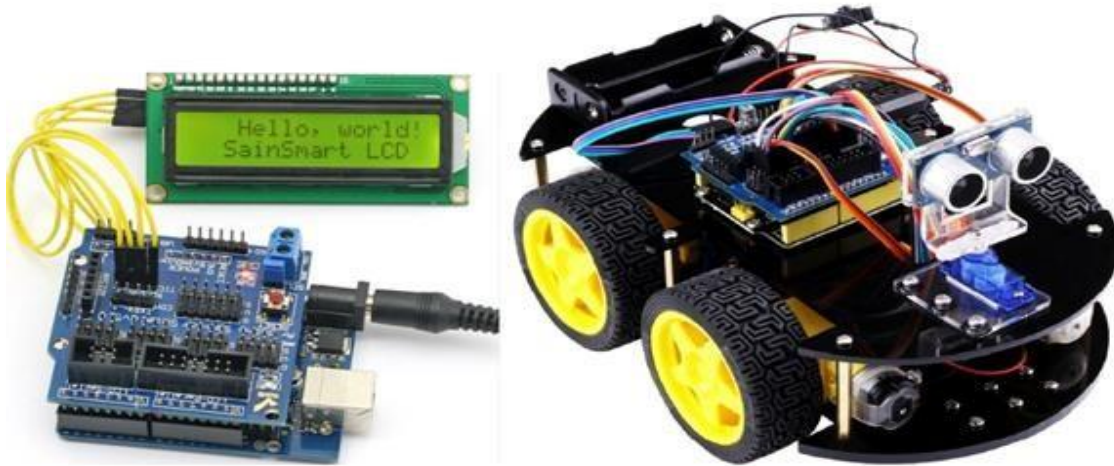


Figure 30: connexion de l'afficheur et la connexion de la voiture

### III.5.4. Marche du robot

La marche du robot se fait selon le fonctionnement des moteurs et la détection de la ligne à l'aide des capteurs infrarouges.

#### III.5.4.1. Bloc moteur

Le moteur fonctionne en 5 volts au maximum de leur puissance, après quelques tests nous avons remarqué que le moteur avait besoin de 1 A en charge pour fonctionner. Donc pour avoir une vitesse max nous appliquons une tension de 5V et pour arrêter le moteur, nous appliquons une tension de 0V.

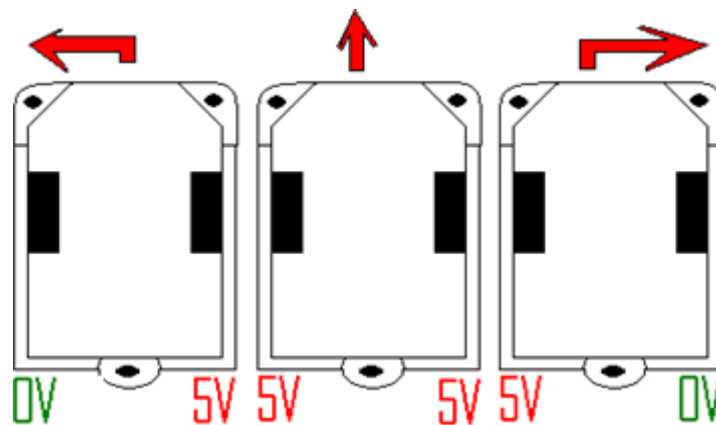


Figure 31: Directions possibles en fonctions de tensions motrices

### III.5.4.2. Description de la carte des capteurs de détection de ligne

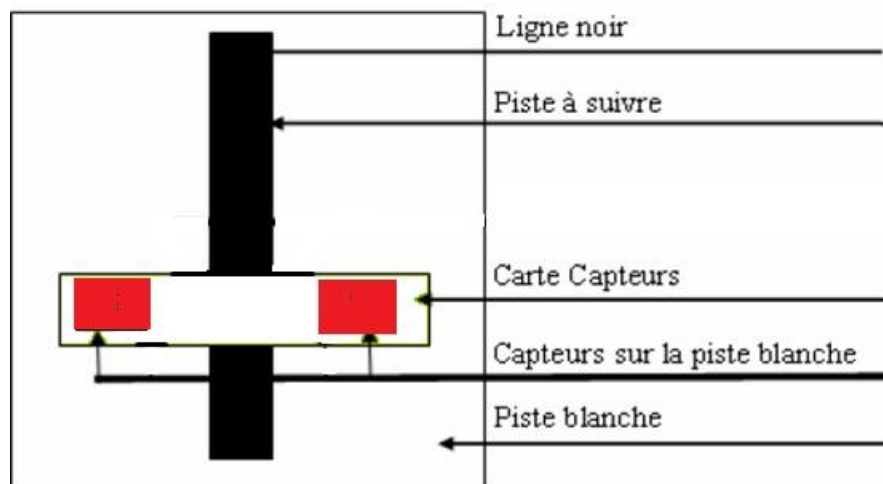


Figure 32: Dessin de la carte détection de ligne

### III.5.4.3. Système des capteurs alignés

Le robot suit une ligne noire sur un arrière-plan blanc tracé au sol qui représente le chemin à suivre, et pour faire cela le robot a besoin de deux capteurs infrarouges qui distingues la ligne noire de l'arrière-plan blanc, Tant que les deux capteurs ne détectent pas la ligne, le robot avance (situation A). Lorsqu'un des deux capteurs détecte la ligne, le robot doit tourner dans la direction de ce dernier pour se mettre au milieu de la ligne (situation B et C), si les deux capteurs détectent la ligne noire, le robot avance (situation D).

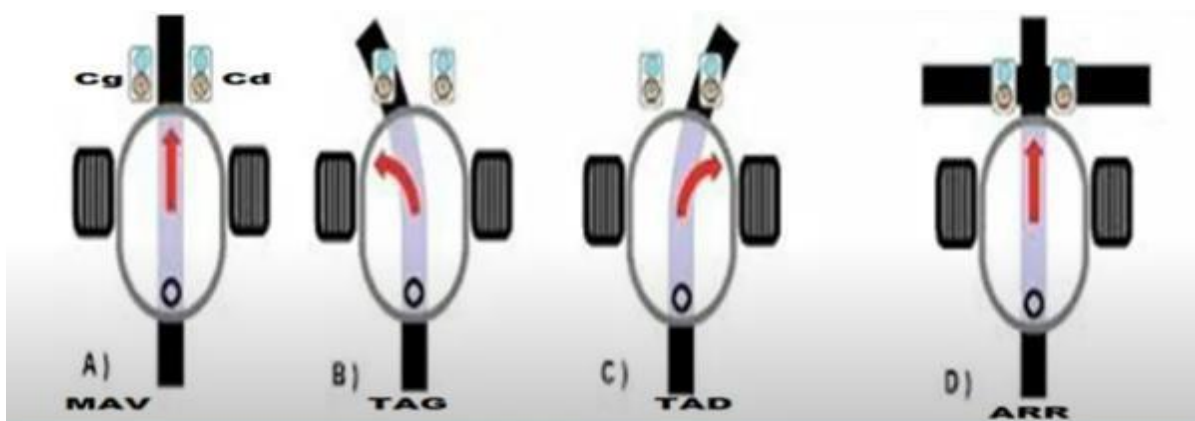


Figure 33: Déférent situation du robot

## III-5-Partie informatique

### III.5.1Algorithme

Notre programme inséré dans l'arduino est schématisé comme suit:

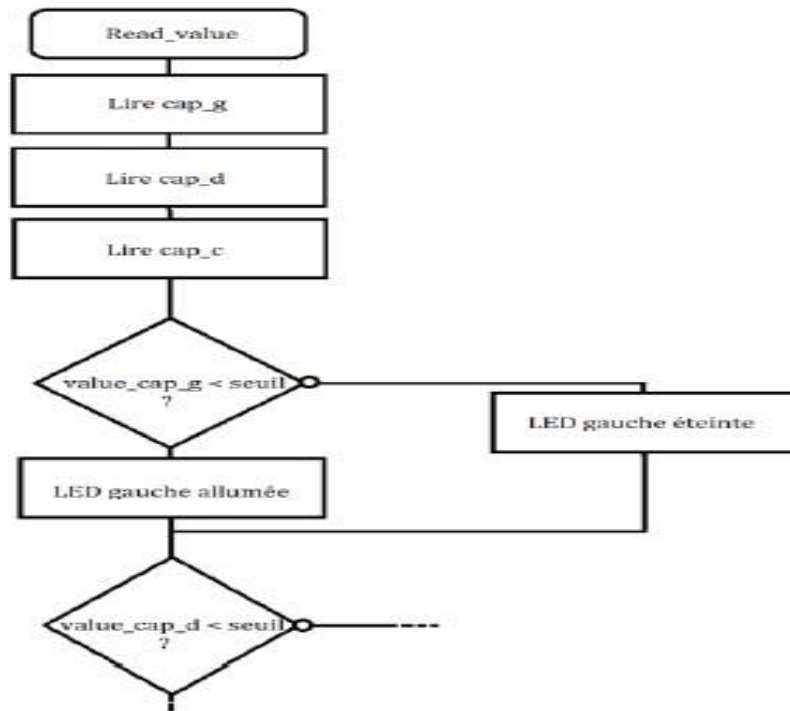


Figure 34 : Ordinoigramme de notre robot suiveur de ligne

### III.5.2. Description des fonctions du programme

Ce programme se découpe en 3 fonctions, incluant loop et setup :

- Fonction ligne Droite () :

La fonction ligne Droite se charge de la gestion de la ligne droite et de la rectification de trajectoire en ligne droite.

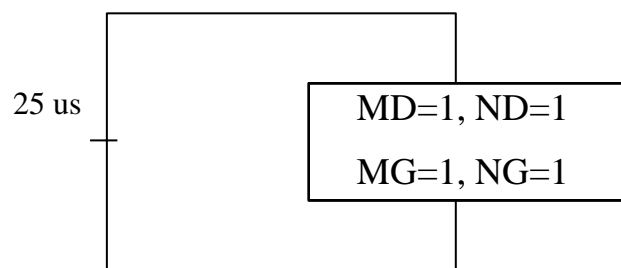
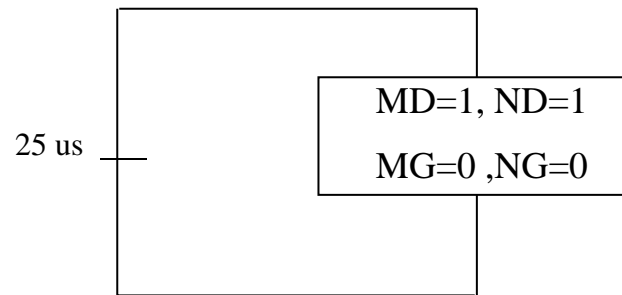


Figure 35: Situation du robot pour la ligne droite

- Fonction virage Gauche ():

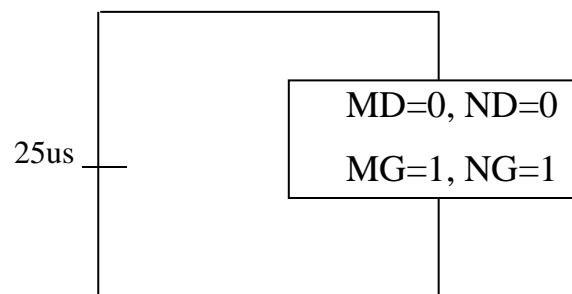
La fonction virage Gauche se charge du retour à la trajectoire et de la rectification d'en ligne droite.



*Figure 36: Situation du robot pour virage gauche*

- Fonction virage Droite () :

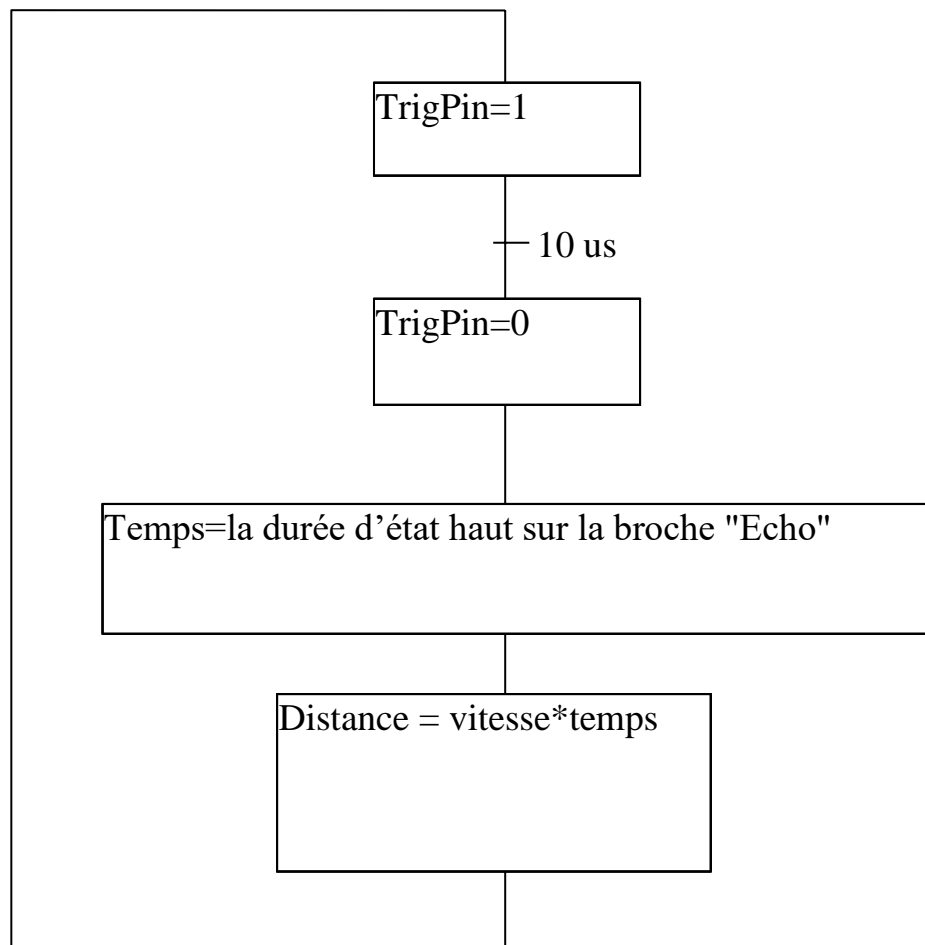
La fonction virage Droite se charge du retour à la trajectoire et de la rectification d'en ligne droite.



*Figure 37: Situation du robot pour virage droite*

- Fonction void look ():

La fonction void look permet de déterminer la distance entre l'obstacle et le robot.



*Figure 38:* Distance entre le robot et l'obstacle

Ces fonctions sont des fonctions non bloquantes, c'est à dire qu'elles s'exécutent sur une durée finie, et elles ne peuvent pas bloquer le programme.

### III.5.3. Description de l'algorithme global du programme

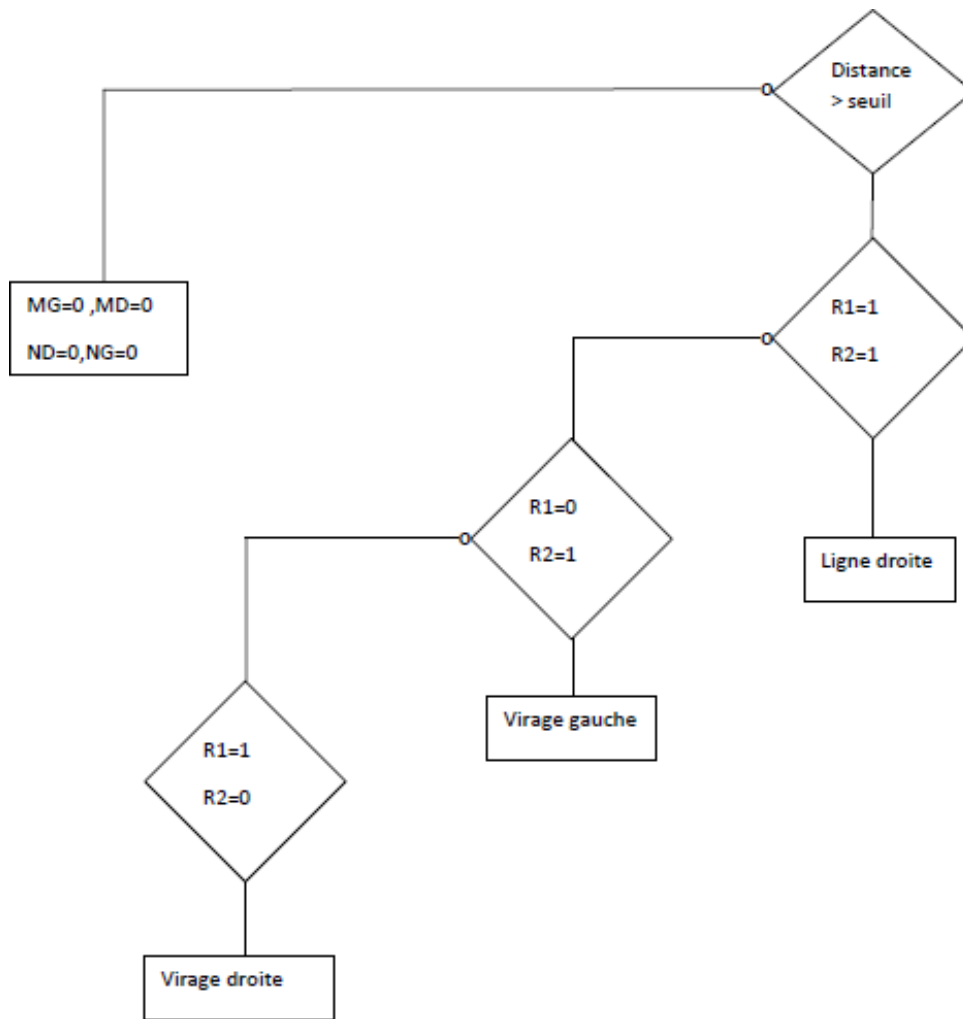


Figure 39: Différentes situations du robot durant la suite de la ligne

- Description du diagramme :

- MG=0, MD=0, ND=0, NG=0 : Le robot est en repos.
- MG=1, MD=1, ND=1, NG=1 : Le robot est en mouvement.
- R1=1, R2=0 : Le robot tourné vers la droite.
- R1=0, R2=1 : Le robot tourné vers la gauche.



### **III.6. Conclusion:**

Dans ce chapitre qui traite la réalisation d'un robot sous forme d'une voiture mobile, nous avons présenté une partie électronique comporte des informations sur les composants utilisés, ainsi leurs simulations dans ISIS avant les connecté. Une partie mécanique traite le branchement et la fixation des pièces sur les châssis, aussi la forme finale de la voiture programmée et la méthode de suivre une ligne noir tracé sur le sol. Une partie Informatique contient l'ordinogramme général de l'algorithme, la description des fonctions virage et l'algorithme global du robot.

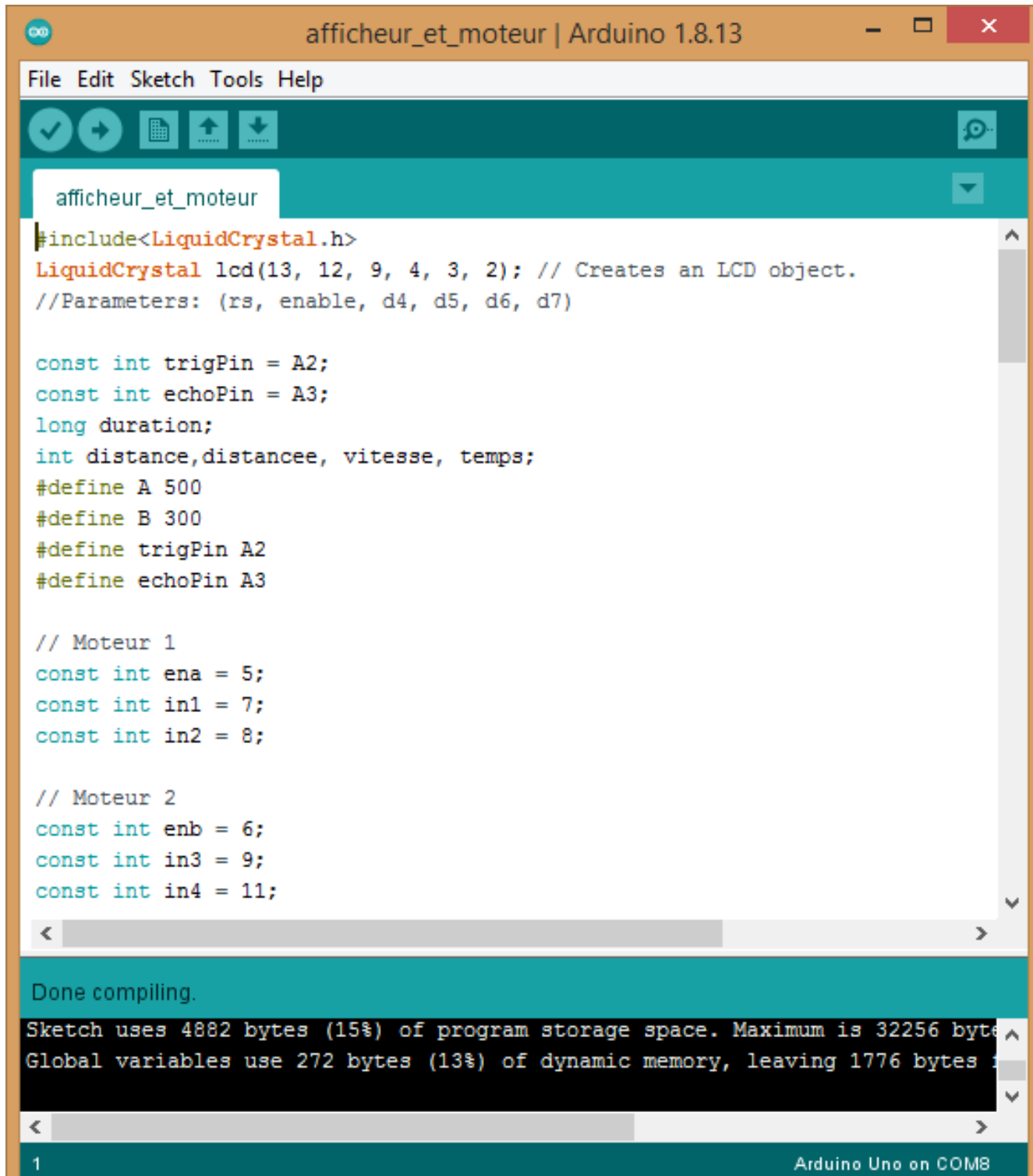
.

# Conclusion générale

Le projet que nous avons réalisé rentre dans le cadre de projet de fin d'étude de licence parcours électronique. Pour cela, nous avons exploité notre connaissance et compétence en électronique et informatique industrielle pour concevoir et réaliser un robot suiveur de ligne été comme application du système embarqué. En effet ce robot constitué de trois arguments : le coté électronique, le coté informatique et le coté mécanique. Pour le coté électronique nous avons choisi de travaillé avec le microcontrôleur ou bien la carte arduino vue son prix très réduit, sa disponibilité, sa spécification électronique très vaste, en plus il dispose un très grande nombre entrées-sorties et la programmation simple. Encore les moteurs à courant continu et les capteurs à cause du prix moins chère, disponible et s'adapte avec le microcontrôleur, il nous permet de réaliser la détection des lignes et la mesure de la distance entre le robot et l'obstacle. Pour le coté mécanique nous avons travaillé sur le placement, l'adaptation des pièces notamment les moteurs et les capteurs et le passage de cette voiture par certaines orientations sa ligné. Pour le coté informatique nous avons travaillé sur la programmation C distingué au l'arduino. Au cours de ce travaille nous avons trouvé quelque problème au niveau d'écriture du programme et le type du shield correspond au moteurs. Nous proposons comme perspective la réalisation d'un robot qui fonctionne avec la carte wifi pour lire afficher les données comme la vitesse, la distance de détection sur un téléphone mobile ou bien une tablette en éliminant l'afficheur LCD. Ainsi il est une occasion pour découvrir d'autres composants pour l'améliorer.

## Annexe :

Code global du robot

The image is a screenshot of the Arduino IDE interface. The title bar at the top reads 'afficheur\_et\_moteur | Arduino 1.8.13'. Below the title bar is a menu bar with 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. Underneath the menu bar is a toolbar with icons for checking, running, saving, and other functions. The main text area contains the following C++ code:

```
#include<LiquidCrystal.h>
LiquidCrystal lcd(13, 12, 9, 4, 3, 2); // Creates an LCD object.
//Parameters: (rs, enable, d4, d5, d6, d7)

const int trigPin = A2;
const int echoPin = A3;
long duration;
int distance,distancee, vitesse, temps;
#define A 500
#define B 300
#define trigPin A2
#define echoPin A3

// Moteur 1
const int ena = 5;
const int in1 = 7;
const int in2 = 8;

// Moteur 2
const int enb = 6;
const int in3 = 9;
const int in4 = 11;
```

The code is color-coded: keywords in blue, constants in cyan, and comments in grey. Below the code editor is a status bar that says 'Done compiling.' followed by memory usage information: 'Sketch uses 4882 bytes (15%) of program storage space. Maximum is 32256 bytes' and 'Global variables use 272 bytes (13%) of dynamic memory, leaving 1776 bytes'. At the bottom of the window, it says '1' on the left and 'Arduino Uno on COM8' on the right.



```
afficheur_et_moteur | Arduino 1.8.13
File Edit Sketch Tools Help

afficheur_et_moteur
const int in4 = 11;

// Vitesse du moteur
int motorSpeed = 0;

const int ir1 = A0;
const int ir2 = A1;

int ir1Va1 = analogRead( ir1 );
int ir2Va1 = analogRead( ir2 );

void setup() {
  lcd.begin(16,2);
  pinMode(trigPin,OUTPUT);
  pinMode(echoPin,INPUT);

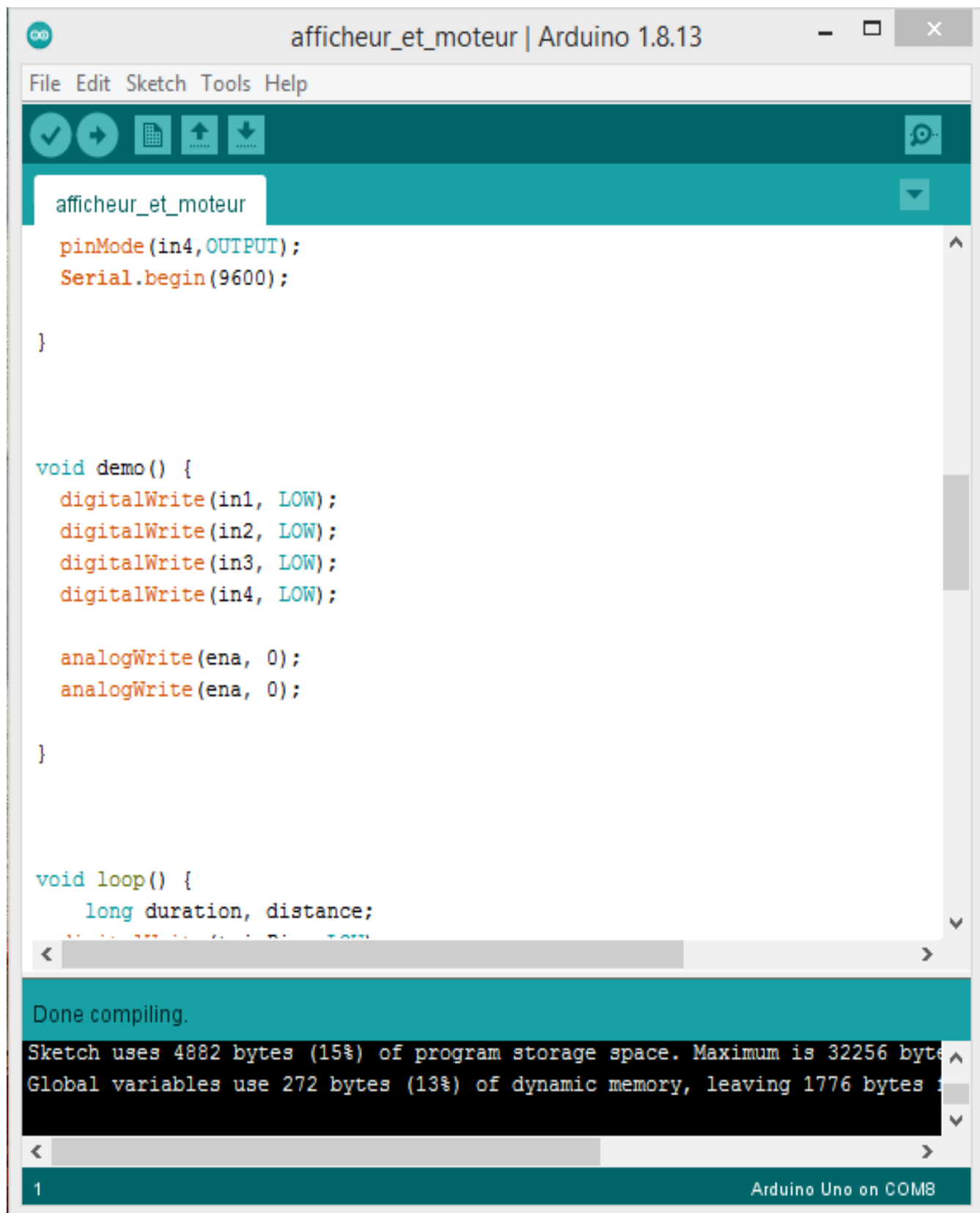
  pinMode(ena,OUTPUT);
  pinMode(in1,OUTPUT);
  pinMode(in2,OUTPUT);
  pinMode(enb,OUTPUT);
  pinMode(in3,OUTPUT);
  pinMode(in4,OUTPUT);
  digitalWrite(in4,HIGH);
}

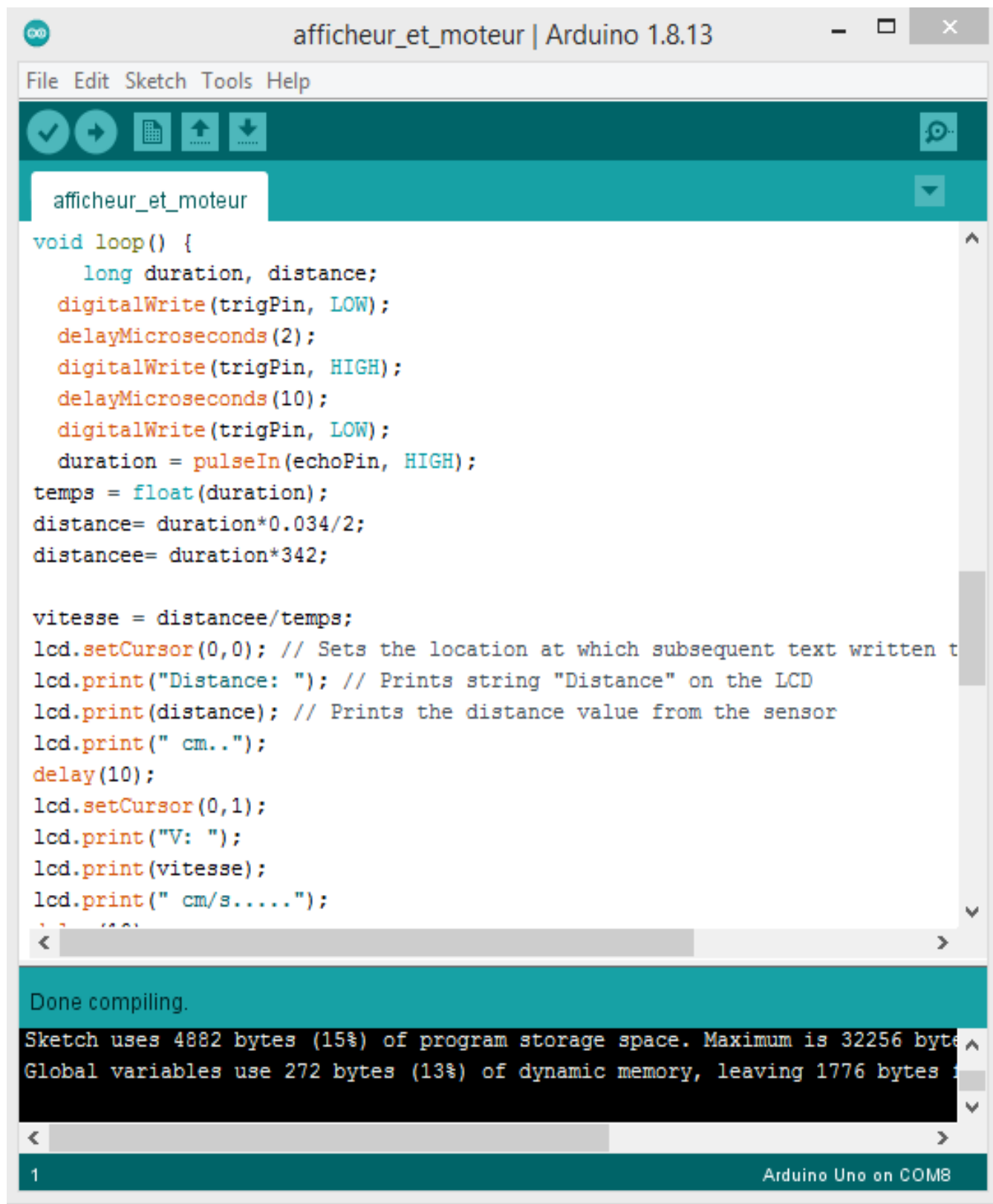
void loop() {
  // ...
}
```

Done compiling.

Sketch uses 4882 bytes (15%) of program storage space. Maximum is 32256 bytes.  
Global variables use 272 bytes (13%) of dynamic memory, leaving 1776 bytes free.

1 Arduino Uno on COM8





afficheur\_et\_moteur | Arduino 1.8.13

File Edit Sketch Tools Help

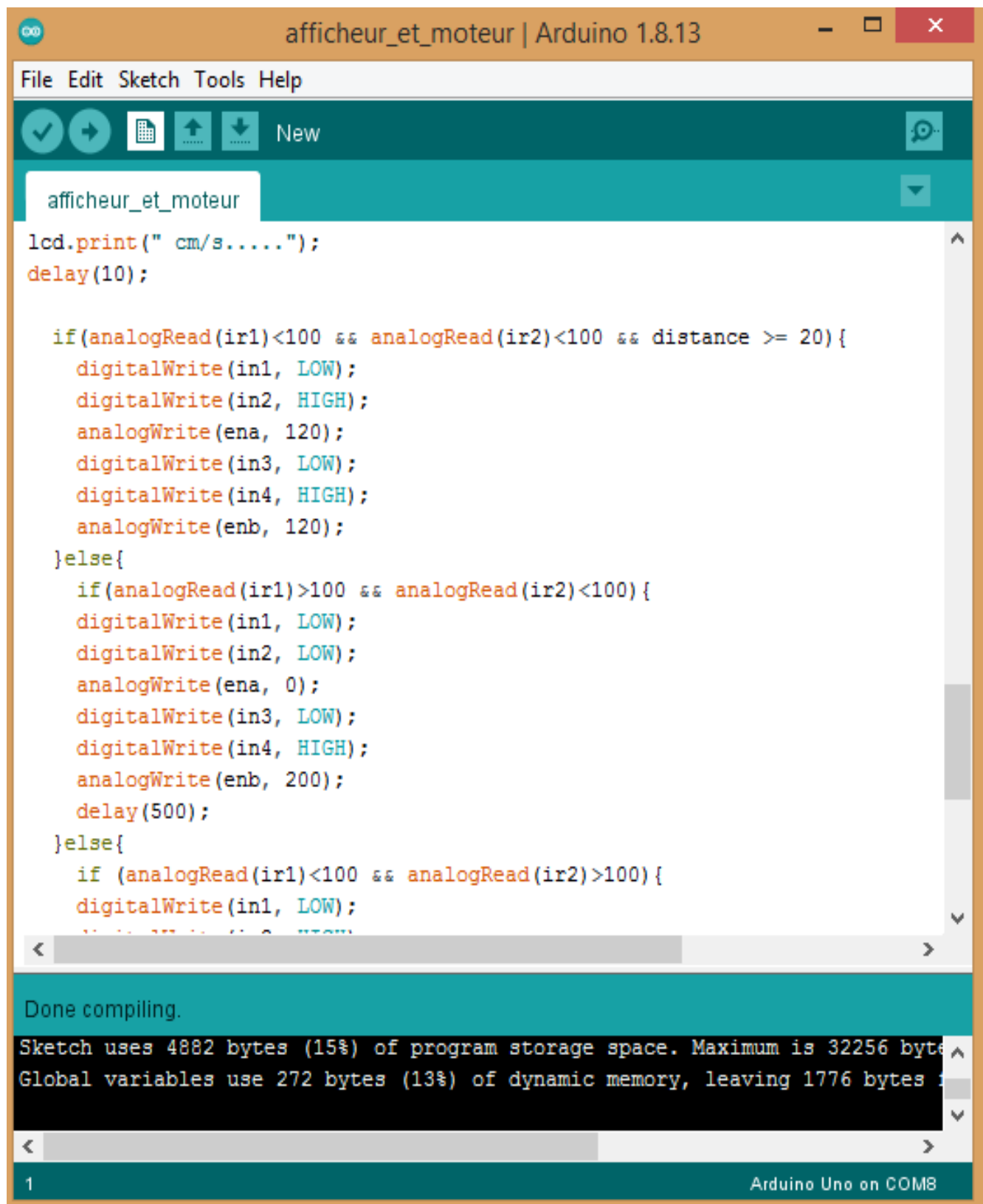
afficheur\_et\_moteur

```
void loop() {  
    long duration, distance;  
    digitalWrite(trigPin, LOW);  
    delayMicroseconds(2);  
    digitalWrite(trigPin, HIGH);  
    delayMicroseconds(10);  
    digitalWrite(trigPin, LOW);  
    duration = pulseIn(echoPin, HIGH);  
    temps = float(duration);  
    distance= duration*0.034/2;  
    distancee= duration*342;  
  
    vitesse = distancee/temps;  
    lcd.setCursor(0,0); // Sets the location at which subsequent text written to  
    lcd.print("Distance: "); // Prints string "Distance" on the LCD  
    lcd.print(distance); // Prints the distance value from the sensor  
    lcd.print(" cm..");  
    delay(10);  
    lcd.setCursor(0,1);  
    lcd.print("V: ");  
    lcd.print(vitesse);  
    lcd.print(" cm/s.....");  
    // ...  
}
```

Done compiling.

Sketch uses 4882 bytes (15%) of program storage space. Maximum is 32256 bytes.  
Global variables use 272 bytes (13%) of dynamic memory, leaving 1776 bytes free.

1 Arduino Uno on COM8



afficheur\_et\_moteur | Arduino 1.8.13

File Edit Sketch Tools Help

✓ ↻ 📄 ⬆ ⬇ New 🔍

afficheur\_et\_moteur

```
lcd.print(" cm/s.....");
delay(10);

if(analogRead(ir1)<100 && analogRead(ir2)<100 && distance >= 20){
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  analogWrite(ena, 120);
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
  analogWrite(enb, 120);
}else{
  if(analogRead(ir1)>100 && analogRead(ir2)<100){
    digitalWrite(in1, LOW);
    digitalWrite(in2, LOW);
    analogWrite(ena, 0);
    digitalWrite(in3, LOW);
    digitalWrite(in4, HIGH);
    analogWrite(enb, 200);
    delay(500);
  }else{
    if (analogRead(ir1)<100 && analogRead(ir2)>100){
      digitalWrite(in1, LOW);
      // ...
    }
  }
}
```

Done compiling.

Sketch uses 4882 bytes (15%) of program storage space. Maximum is 32256 bytes.  
Global variables use 272 bytes (13%) of dynamic memory, leaving 1776 bytes free.

1 Arduino Uno on COM8

```
afficheur_et_moteur | Arduino 1.8.13
File Edit Sketch Tools Help

afficheur_et_moteur
    delay(500);
}else{
    if (analogRead(ir1)<100 && analogRead(ir2)>100){
        digitalWrite(in1, LOW);
        digitalWrite(in2, HIGH);
        analogWrite(ena, 200);
        digitalWrite(in3, LOW);
        digitalWrite(in4, LOW);
        analogWrite(enb, 0);
        delay(500);

    }else{
        lcd.setCursor(0,1);
        lcd.print("detect obstacle!");
        digitalWrite(in1, LOW);
        digitalWrite(in2, LOW);
        analogWrite(ena, 0);
        digitalWrite(in3, LOW);
        digitalWrite(in4, LOW);
        analogWrite(enb, 0);
    }
}
}

Done compiling.
Sketch uses 4882 bytes (15%) of program storage space. Maximum is 32256 bytes
Global variables use 272 bytes (13%) of dynamic memory, leaving 1776 bytes
1 Arduino Uno on COM8
```



## Bibliographie

- [1]: P. Ficheur, Linux embarqué, 2<sup>nd</sup> édition, 2005.
- [2]: F. Boudoin, M. Lavabre, Capteurs : principales utilisations, *Edition Casteilla*, 2007.
- [3]: Paret, Dominique, Rebaine, Hassina, Réseaux de communication pour systèmes embarqués, 2<sup>nd</sup> édition, Dunod, 2018.
- [4]: J. Nssey, Arduino pour les Nuls, 2<sup>nd</sup> édition, 2017.
- [5]: R. Siegwart, I.R. Nourbakhch, D. Scaramuzza, Introduction to Autonomous Mobile Robots, 2<sup>nd</sup> Edition, 2011.
- [6]: L. Jaulin, *La robotique mobile*, Editions ISTE, 2015.
- [7]: V. Maille, C. Accard, B. Breton, *Les robots : apprendre la robotique par l'exemple*, Editions Ellipse, 2016.
- [8]: R. Frank, *Understanding Smart Sensors (Artech House sensors library)*, 2<sup>nd</sup> edition, 2016.
- [9]: B. Hollembeak, Today's Technician: Automotive Electricity and Electronics Classroom and Shop Manual Pack, 5th edition, Delmar, 2010.

