

PROJET

Présenté à

L'Institut International de Technologie de Sfax

Intitulé

**Navigation autonome de robot avec évitement
d'obstacle toute en détectant le gaz et la fumée**

Réalisé par :

Slim Yaich

Mr. TAREK FRIKHA

Président

Table des matières

1. Généralité sur la robotique mobile	11
1.1 Introduction	11
1.2 Historique des robots	11
1.3 Domaine D'application	12
1.3.1 Les robots industriels	12
1.3.2 Les robots explorateur	12
1.3.2 Les robots chirurgicaux	13
1.3.2 Les robots militaires	13
1.4 Les robots mobiles	14
1.4.1 Les robots mobiles à roues	14
1.4.2 Les robots mobiles à chenilles	16
1.4.3 Les robot mobile à pattes.....	16
1.5 Conclusion.....	17
2. navigation d'un robot autonome.....	18
2.1 Introduction	18
2.2 Cartographie	18
2.3 Planification	18
2.4 Navigation	19
2.5 Les stratégies de navigation :	19
2.6 Conclusion.....	21
3. Détection et Evitement d'obstacles	22
3.1 Introduction	22
3.2 Les télémètres	22
3.2.1 Télémètre ultrason.....	22
3.2.2 Télémètre laser	23
3.2.3 Telemetre infrarouge	24

3.3	Caméra	24
3.3.1	Caméra thermique	24
3.3.2	Caméra analogique.....	25
3.3.3	Les caméra stéréoscopiques.....	25
3.4	Evitement d'obstacle.....	25
3.5	Conclusion.....	26
4.	CONCEPTION DU ROBOT	27
4.1	Introduction	27
4.2	Conception mécanique	27
4.2.1	Servo moteur SG90.....	27
4.2.1.1	Rôle et choix du servo moteur.....	27
4.2.1.2	Principe de fonctionnement.....	29
4.2.1.3	Branchement du servo moteur.....	29
4.2.2	Châssis à quatre roues motrices	30
4.2.2.1	Moteur à courant continu.....	30
4.2.2.2	Réducteur.....	31
4.2.2.3	Châssis.....	32
4.3	Conception électrique	32
4.3.1	Carte arduino UNO	32
4.3.2	Description matérielle de la carte Arduino.....	33
4.3.2.1	Le microcontrôleur ATmega328.....	33
4.3.2.2	Les entrées / sorties numériques	34
4.3.3	La carte de puissance L298N.....	34
4.3.3.1	Caractéristique.....	35
4.3.4	Le capteur ultrason SRF04	36
4.3.4.1	Présentation.....	36
4.3.4.2	Principe des ultrasons.....	36
4.3.4.3	Connexion du capteur ultrason SRF04 à une carte arduino.....	37
4.3.5	Détecteur de gaz (GPL, Méthane, Butane, hydrogène, fumée) MQ-5.....	37
4.3.6	Buzzer.....	39
4.3.7	W1209 Contrôle de la température du thermostat Commutateur.....	40

4.4 Conception informatique.....	41
4.4.1 Logiciel de la carte Arduino	41
4.4.1.1 Description.....	41
5. REALISATION DU ROBOT	43
5.1 Introduction	43
5.2 L'environnement de travail :	43
5.3 Schéma électrique	44
5.4 Assemblage des composants	45
5.5 Programmation du robot	47
5.5.1 Algorithme de détection d'obstacle	47
5.5.2 Algorithme principale de mouvement de notre robot	49
5.5.3 les bibliothèques utilisées	49
5.5.4 Les constantes	49
5.6 Conclusion.....	50
6 Conclusion	51
6.1 Circuits additionnels de l'arduino	54

Table des figures

Figure 1 : historique des robots	11
Figure 2 : robot industriel	12
Figure 3 : Robot Explorateur	12
Figure 4 : Robot Chirurgicale	13
Figure 5 : robot militaire.....	13
Figure 6 : drone militaire.....	13
Figure 7 : Robot Mobile à roues	15
Figure 8 : Robot Mobile à chenilles	16
Figure 9 : Robot Mobile à pattes	16
Figure 10 : Action associé à un lieu	20
Figure 11 : Navigation topologique.....	20
Figure 12 : Navigation métrique	21
Figure 13 : Télémètre ultrason	22
Figure 14 : Principe de capteur ultrason	23
Figure 15 : Télémètre laser	23
Figure 16 : Télémètre infrarouge	24
Figure 17 : caméra thermique	24
Figure 18 : Servo moteur SG90	27
Figure 19 : Dimensions du servo moteur SG90	28
Figure 20 : Branchement du servo moteur avec la carte Arduino.....	29
Figure 21 : Roue motrice à réducteur	30
Figure 22 : Moteur à courant continu F130	30
Figure 23 : un exemple d'un motoréducteur.....	31
Figure 24 : Réducteur de vitesse.....	31
Figure 25: Châssis du robot	32
Figure 26 : Carte Arduino UNO.....	32
Figure 27 : Microcontrôleur ATmega328	33
Figure 28 : Les différents entrée/ sortie de la carte arduino	34
Figure 29: Carte de puissance L298N	35
Figure 30 : Schéma de branchement de la carte L298N avec l'arduino et le moteur	35
Figure 31 : Capteur HC-SR04.....	36

Figure 32 : Connexion du SFR04 à la carte Arduino	37
Figure 33 : Détecteur de Gaz	38
Figure 34 : Buzzer piézo-électrique	39
Figure 35 : Capteur de température W1209.....	40
Figure 36 : Interface logicielle Arduino	41
Figure 37 : Réalisation du circuit	44
Figure 38 : Figure assemblage châssis-moteur-roues.....	45
Figure 39 : Capteur de température W1209-montage de carte de puissance-Arduino.....	46
Figure 40 : Capteur Ultra son - servomoteur	47

Liste des tableaux

Tableau 1 : Type de robot mobile	14-15
--	-------

Acronymes :

AVR : Alf and Vegard's RISC

EEPROM : Electrically Erasable Programmable Read-Only Memory

GND : Ground

IDE : Integrated development environment

LED : Light-emitting diode

PWM : Pulse width modulation

SRAM : Static random-access memory

TTL : Transistor-transistor logic

INTRODUCTION GENERALE

L'Homme a toujours souhaité se libérer du travail dans ce qu'il a de fatiguant, d'inintéressant physiquement ou mentalement. Le progrès technique qu'on a connu depuis un demi-siècle répond en bonne parti à son désir : l'invention des robots qui sont des machines programmables destinées à remplacer l'Homme en assurant la qualité des résultats obtenus. L'origine du mot robot est issue du tchèque "robota" qui signifie travail forcé, besogne, corvée. La conception de ces systèmes est l'objet d'une discipline scientifique, branche de l'automatisme nommé robotique.

La robotique touche aujourd'hui de nombreux secteurs de la vie, elle a extrêmement progressé durant le siècle dernier. Les chercheurs parviennent peu à peu à donner à des machines une intelligence artificielle. Les robots envahissent littéralement notre vie, on les trouve dans les films de science-fiction et dans tous les autres domaines tels que l'industrie, la santé, le nettoyage...aussi, ils se trouvent dans toutes les entreprises pour accélérer la production ou pour agir dans les environnements inaccessibles et dangereux pour l'Homme, à la maison pour aider à la cuisine et aux tâches ménagères, dans les services publics : hôpitaux, casernes de pompiers, la police... Les machines sont amènes à résoudre les problèmes les plus complexes de la science et de l'ingénierie d'une manière intelligente comme par exemple la conduite avec les voitures autonomes, ou trop dangereuses, comme un site radioactif...

Dans ce logique, notre travail consistera à concevoir un robot mobile permettant d'effectuer plusieurs missions autonomes dans des conditions extrêmes.

Ainsi notre travail est subdivisé en quatre principales parties :

Partie 1 : Recherche bibliographique

Partie 2 : Conception d'un mode de navigation autonome et choix des capteurs

Partie 3 : Elaboration d'un algorithme autonome qui inclue le mode de navigation dégradées

Partie 4 : Test de validation

1. GENERALITE SUR LA ROBOTIQUE MOBILE

1.1 Introduction

Les robots mobiles sont particulièrement demandés pour réaliser des tâches considérées comme pénibles, dangereuses ou impossible pour l'être humain. C'est pour cela le développement des robots mobiles est motivé par la nécessité et le souhait de mettre des robots en œuvre pour assister l'homme dans ses travaux et dans son environnement quotidien. Bien souvent les robots mobiles à roues sont les systèmes les plus étudiés, parce qu'ils sont plus simples à réaliser, plus que d'autres types de robots mobiles. Ce qui permet d'en venir plus rapidement à l'étude de leur navigation.

1.2 Historique des robots

Vers la fin des années 1970, la robotique devient un thème de recherche à part entière stimulé par la robotique industrielle et les études sur les bras manipulateurs. Mais, c'est surtout à partir des années 1980 qu'un intérêt croissant a été accordé à la recherche en robotique mobile motivé d'une part par l'augmentation de la puissance de calcul embarquée et la présence des nouvelles technologies de communication sans fils, l'amélioration des capteurs d'autre part.

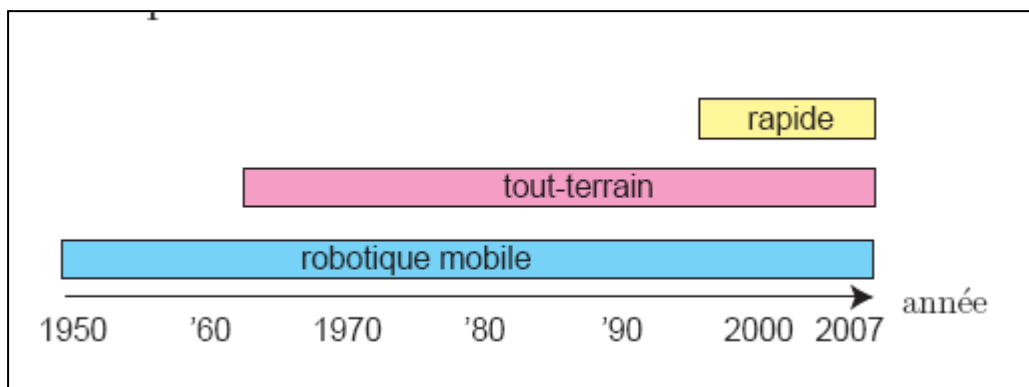


Figure 1 : historique des robots

1.3 Domaine D'application

1.3.1 Les robots industriels

Cette catégorie est intensément utilisée dans le domaine de l'industrie. Ils sont maintenant utilisés pour fabriquer à peu près tous les types de produits et surtout les tâches itératives et dangereuses, tels que la manipulation d'objets, la soudure et l'assemblage de pièces. Notons toutefois que l'industrie automobile utilise plus de 50% de l'ensemble des robots industriels



Figure 2 : robot industriel

1.3.2 Les robots explorateurs

De plus qu'elle dépend des moyens importants, l'exploration de l'espace est très dangereuse pour l'être humain, d'où l'utilisation des robots est la solution la plus efficace pour remplacer l'homme dans les environnements difficiles et inaccessibles.



Figure 3 : Robot Explorateur

1.3.3 Les robots chirurgicaux

Les robots chirurgicaux sont à la base des robots industriels qui permettent de fournir un aide en simplifiant les tâches du chirurgien pour rendre les opérations chirurgicales plus confortables pour le chirurgien ainsi que pour le patient.

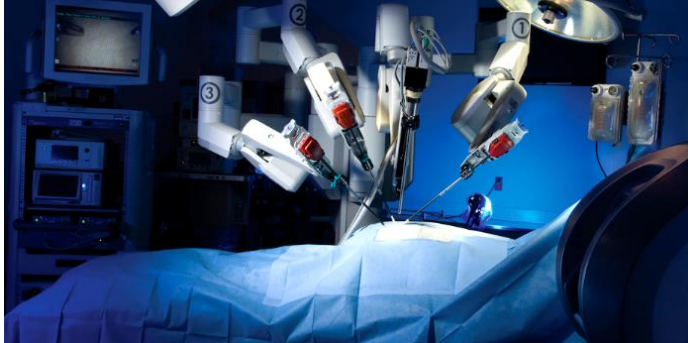


Figure 4 : Robot Chirurgicale

1.3.4 Les robots militaires

Les robots militaires sont utilisés pour simplifier les tâches des soldats et limiter les pertes humaines. Ils sont principalement utilisés pour la surveillance aussi bien dans les airs que dans la mer.

Des systèmes sont déjà actuellement en service dans un certain nombre de forces armées, avec des succès remarquables, tel que le drone “Predator“, qui est capable de prendre des photographies de surveillance, et même lancer des missiles air-sol.



Figure 5 : robot militaire



Figure 6 : drone militaire

1.4 Les robots mobiles

Un robot mobile est un véhicule doté de moyens de locomotion qui lui permettent de se déplacer.

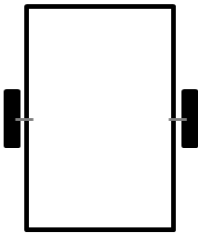
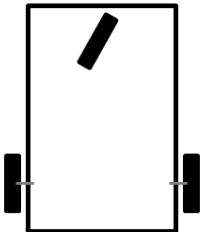
L'objectif principal d'un robot mobile consiste à réaliser un mouvement en reliant un point source à un point destination. Suivant son degré d'autonomie il peut être doté de moyens de perception et de raisonnement.

Il existe plusieurs types de robot mobile :

1.4.1 Les robots mobiles à roues

La mobilité par roues est la structure mécanique la plus communément appliquée. Cette technique assure selon l'agencement et les dimensions des roues un déplacement dans toutes les directions avec une accélération et une vitesse importante. Le franchissement d'obstacles ou l'escalade de marches d'escaliers est possible dans une certaine mesure. Toutes les configurations (nombre, agencement, fonction) des roues sont appliquées.

Nous allons effectuer notre projet sur ce type de robots mobiles qui engendre essentiellement quatre types présentés dans le tableau suivant :

Type de robots	Image	Avantages	Inconvénients
Robot unicycle		Stable Rotation sur soi-même Complexité mécanique faible	Non holonome
Robot tricycle		Complexité mécanique modérée	Non holonome Peu stable Pas de rotation sur soi-même

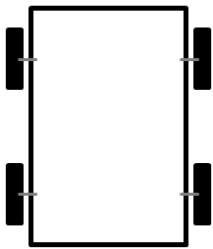
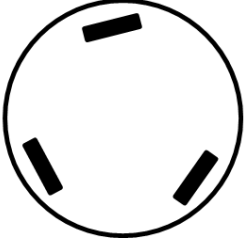
Robot voiture		Complexité mécanique modérée Stable Complexité mécanique modérée	Non holonome Pas de rotation sur soi- même
Robot omnidirectionnel		Holonome Stable Rotation sur soi- même	Complexité mécanique importante

Tableau 1 : Type de robot mobile

En ce qui nous concerne, nous nous pencherons sur le robot mobile à quatre roues.

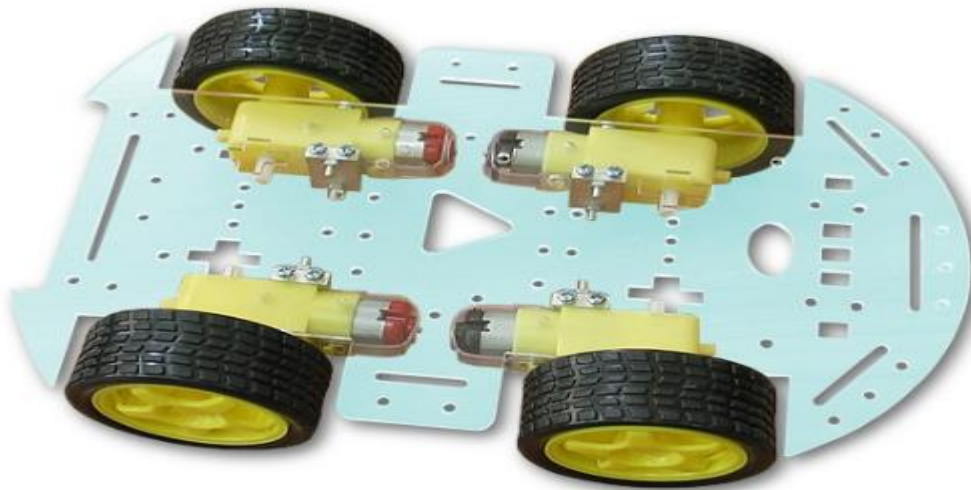


Figure 7 : Robot Mobile à roues

1.4.2 Les robots mobiles à chenilles

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

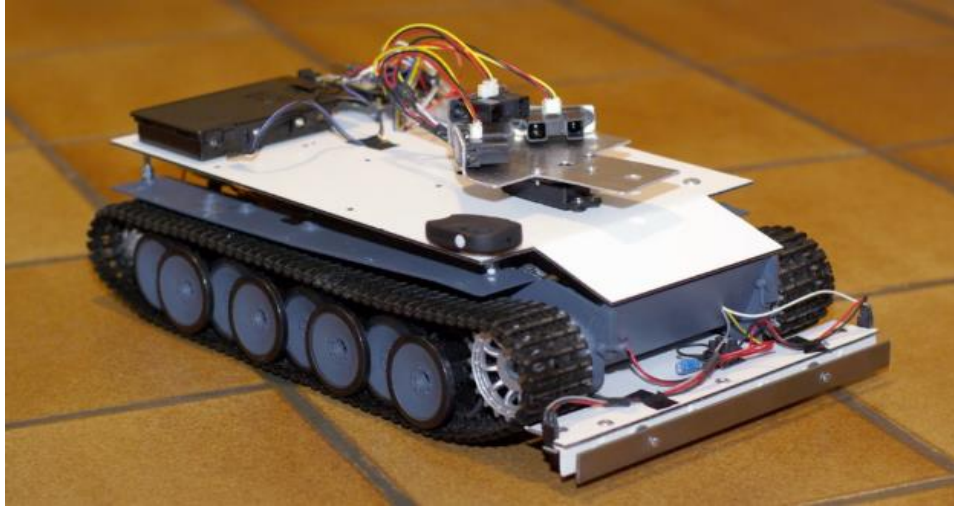


Figure 8 : Robot Mobile à chenilles

1.4.3 Les robots mobiles à pattes

Des plates-formes à deux, quatre ou six pattes peuvent également être utilisées. Les plates-formes à six pattes sont relativement pratiques car le robot est en équilibre permanent, ce qui facilite le contrôle. Ils sont mieux adaptés au franchissement d'obstacles et au travail dans un environnement construit pour la morphologie humaine. Ils peuvent franchir des obstacles que les roues ne peuvent surmonter, monter et descendre des escaliers, circuler sur des terrains très accidentés, et finalement, ils ont moins tendance à s'enfoncer dans un sol instable.



Figure 9 : Robot Mobile à pattes

1.5 Conclusion

Ce chapitre présente également des généralités importantes sur la robotique mobile. Nous avons mentionné tout d'abord une partie de l'histoire, ensuite nous avons passé à leurs domaines d'applications, ainsi leurs classifications par types. L'appellation Robot mobile regroupe tous les types de robots qui ont la capacité de déplacement qui est la caractéristique commune entre eux, la différence réside dans la manière, qui dépend du domaine d'utilisation de robot, par laquelle le robot va atteindre cette faculté de mouvement. La mobilité par les roues est la structure mécanique la plus communément appliquée. Cette technique assure selon l'agencement et les dimensions des roues un déplacement dans toutes les directions avec une accélération et une vitesse importante.

2. NAVIGATION D'UN ROBOT AUTONOME

2.1 Introduction

La navigation du robot mobile est une fonction qui lui permet de se diriger dans son environnement et d'attendre une cible. Il est nécessaire de pouvoir préparer un chemin permettant de rejoindre une position précise à partir de n'importe quelle autre position. Cependant l'environnement du robot est généralement occupé d'obstacles, plusieurs techniques sont utilisées dans la navigation autonome. Ces techniques seront présentées dans ce qui suit :

2.2 Cartographie

La cartographie permet de modifier les données requises par les capteurs en un environnement (une carte). Lorsque la carte est disponible La localisation détermine l'emplacement du robot qui correspond à son emplacement réel dans son environnement. Cette définition pose un problème de dépendance entre confection de la carte et de la localisation. En effet la localisation et la cartographie sont fortement liées car un robot ne peut pas se localiser tant qu'il n'y a pas de carte et la conversion des données des capteurs ne peut pas avoir lieu si le robot ne sait pas son emplacement et où il doit organiser les autres éléments

2.3 Planification

Une fois que le robot dispose d'une estimation de sa position, il convient de s'en servir pour générer les tâches qui lui permettent d'atteindre son objectif. Pour cela plusieurs méthodes sont possibles. Le choix d'une méthode de planification est guidé par deux questions :

1. Quel type d'espace utilise : l'espace de travail ou l'espace des configurations ?
2. Quel type de méthodes : des méthodes exactes ou des méthodes approchées ?

Les méthodes exactes sont basées sur une exploitation complète de la description de l'environnement. Par opposition, les méthodes approchées réalisent tout d'abord une discrétisation de l'environnement sous forme de grilles régulières ou irrégulières. L'espace libre ainsi représenté est un sous-ensemble de l'espace libre réel. Alors que les méthodes exactes sont susceptibles d'être complètes, les méthodes approchées ne le sont jamais [4].

2.4 Navigation

La navigation autonome d'un robot mobile est une procédure qui permet généralement de trouver un mouvement libre dans l'espace de configuration en présence d'obstacle proche du robot. Cet espace est un ensemble des paramètres qui caractérise la position et l'orientation du robot.

la navigation du robot se fait d'une configuration initiale $q_0=(x_0,y_0,\theta_0)$ à une configuration finale $q_f=(x_f,y_f,\theta_f)$.

Le robot partir son mouvement d'une situation initiale $s(t)$ suivant la vitesse et l'angle de braquage il doit se mouvoir vers une situation $s(t+1)$.

La procédure de navigation se fait d'une façon itérative comme suit :

- A chaque instant t_i , le robot calcule les distances de son position aux obstacles d_i et la position courante et finale.
- Déterminer les variables de commande adéquats par le système de contrôle $v_r(t + 1)$ et $\alpha(t+1)$.
- Exécuter ces étapes en remplaçant vers une nouvelle position.
- Répéter les étapes 1,2et3 jusqu'à la destination finale.

2.5 Les stratégies de navigation :

Il existe plusieurs stratégies de navigation permettant un robot mobile de se déplacer pour atteindre un but :

-Action associée à un lieu : cette stratégie permet d'atteindre un but en décomposant l'environnement en plusieurs positions pour laquelle son déplacement est invisible. Cette stratégie consiste donc à mémoriser des positions et définir une action entre d'eux. En faisant l'enchaînement de ces actions, une route sera construite qui permet d'atteindre notre destination. Une fois on a choisi une route ce dernier ne pourra pas être utilisés pour atteindre un autre but, dans ce cas cette stratégie permet de chercher une autre route selon le changement de but. La figure ci-dessous montre l'action associée à un lieu :

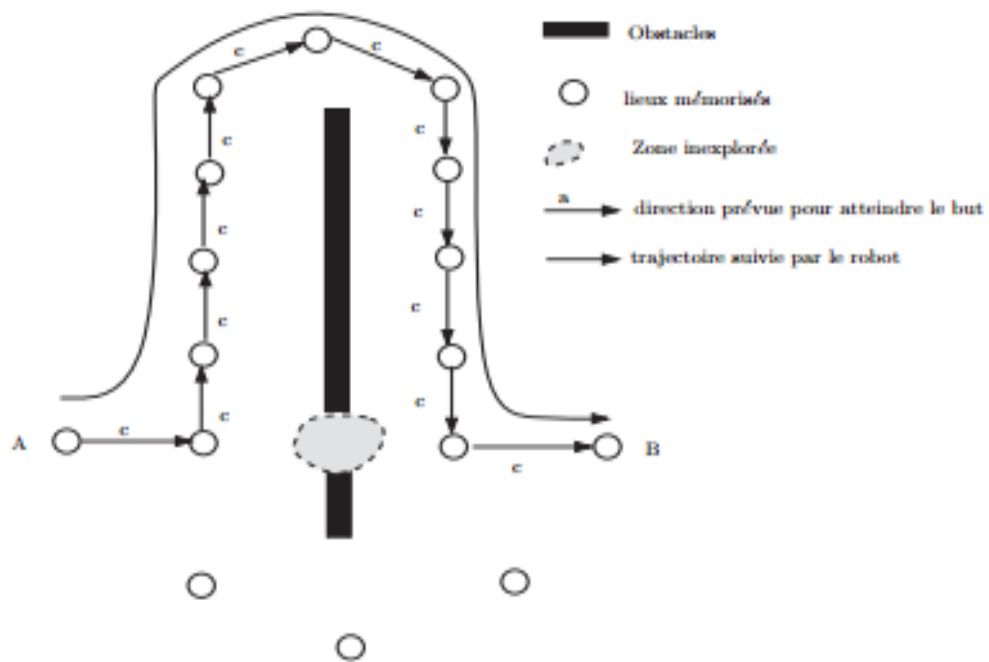


Figure 10 : Action associé à un lieu

-**Navigation topologique** : le principe de cette navigation est le même que celle précédente mais en mémorisant les différentes relations entre les positions spatiales comme représentés dans la figure ci-dessous. Ces relations nous permettent de se diriger vers n'importe quelle position mais ne sont plus associées à un but particulier. L'avantage de cette stratégie c'est qu'on peut calculer les différents chemins entre deux position arbitraires mais elle présente un inconvénient c'est qu'il est incapable de planifier un robot à se déplacer vers des positions ou suivant des chemins connus. Un exemple de navigation topologique :

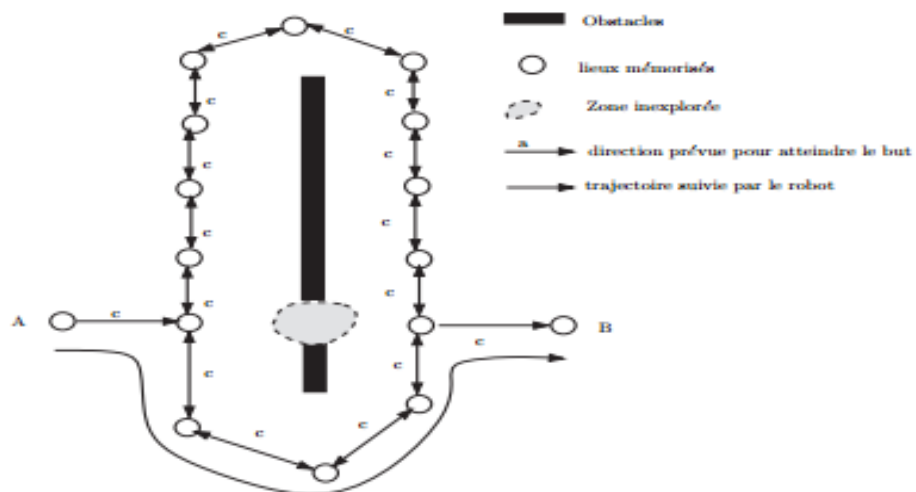


Figure 11 : Navigation topologique

-Navigation métrique : c'est une extension de la stratégie précédente mais cette stratégie permet la planification des routes non exploités dans l'environnement connu représente la figure ci-dessous. Cette stratégie permet de calculer le plus court chemin entre les différentes positions mémorisées pour cela elle mémorise les positions métriques pour chaque lieu visité par le robot, ainsi grâce à cette navigation on peut se déplacer vers des lieux non enregistrés par exemple la figure ci-dessous montre le déplacement du robot qui permet d'atteindre la position D à partir de la position A en passant par la zone inexploitable.

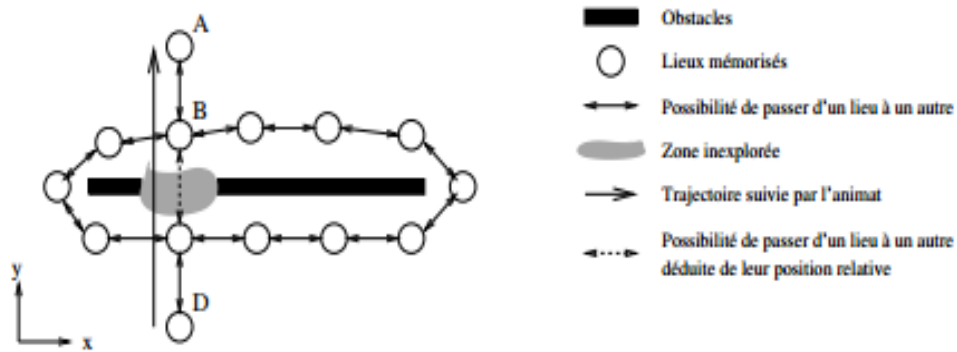


Figure 12 : Navigation métrique

2.6 Conclusion

Dans ce chapitre nous avons donné l'état de l'art de la navigation autonome ainsi que leurs techniques et leurs méthodes de navigation autonome.

3. DETECTION ET EVITEMENT D'OBSTACLES

3.1 Introduction

Dans le domaine robotique, un objet est considéré comme un obstacle lorsque l'intersection de sa trajectoire avec la ligne de direction du robot est non vide à instant donné. Plusieurs capteurs sont développés et utilisés par les chercheurs afin d'accomplir la tâche de la détection et l'évitement des obstacles.

3.2 Les télémètres

C'est une catégorie de capteurs qui permet de détecter les obstacles à partir d'une émission d'une onde qui se réfléchit à la surface de l'obstacle situé dans la direction de propagation. et en faisant une comparaison entre cette onde émise avec son écho on peut estimer la distance entre le capteur et la surface de l'objet.

On distingue alors trois types de télémètre :

3.2.1 Télémètre ultrason



Figure 13 : Télémètre ultrason

Les télémètres à ultrason sont les capteurs les plus utilisés grâce à leurs faibles couts et leurs simplicités. L'information particuliers la plus intéressante dans le domaine robotique est la distance séparant le robot aux obstacles pour cela on utilise le capteur à ultrason vue de leur disponibilité et leur capacité de mesure qui varie entre 3cm et 3m.

Figure 10 : Télémètre ultrason

Le principe de ce télémètre consiste à envoyer une onde puis il mesure le temps après lequel l'écho revient, par la suite connaissant la vitesse de vol dans le milieu considéré et le temps de vol on peut calculer la distance par la formule suivante :

$$D = \frac{\text{Vitesse de propagation} * \text{durée de vol}}{2}$$

La figure suivante montre le principe du télémètre à ultrason :

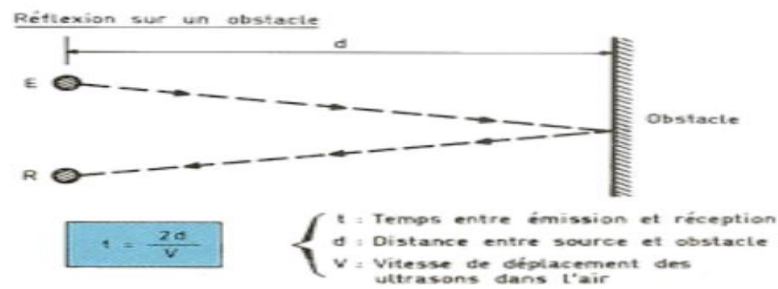


Figure 14 : Principe de capteur ultrason

3.2.2 Télémètre laser



Figure 15 : Télémètre laser

Le principe de fonctionnement du télémètre laser se repose sur l'envoi d'un faisceau laser vers un obstacle qui renvoie à son tour un faisceau lumineux. Puis en calculant le déphasage entre l'émission et la réception on peut calculer la distance en séparant le robot aux obstacles. Les télémètres laser sont très utilisés de nos jours-là dans les applications de cartographie et de localisation.

3.2.3 Télémètre infrarouge

On peut utiliser ces capteurs soit pour le suivi ou pour la détection d'obstacles. Leur utilisation est similaire à celle des lasers. Les mesures de distance peuvent être faites par triangulation, par mesure d'intensité et par mesure de temps de vol. Nous avons aussi parfois des angles d'ouvertures importantes à l'émission.

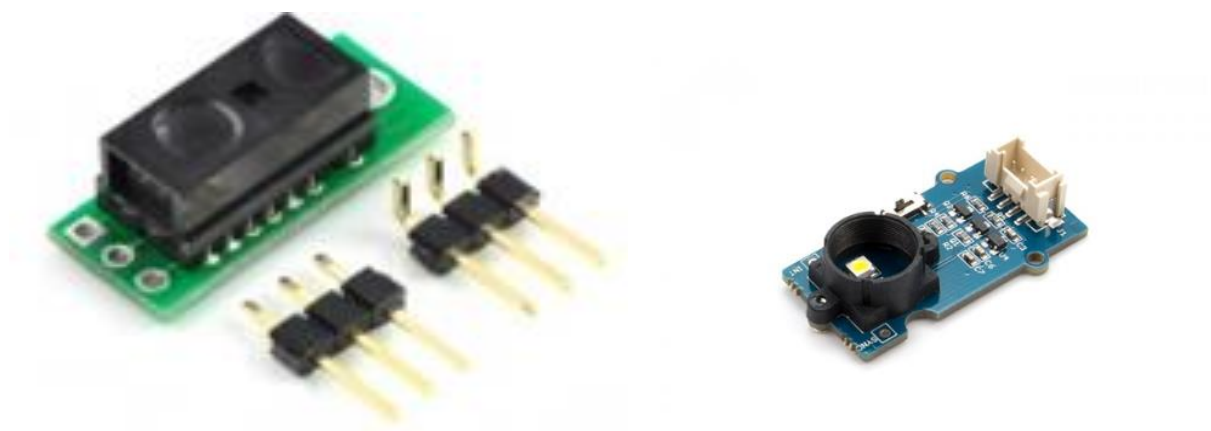


Figure 16 : Télémètre infrarouge

3.3 Caméra

La caméra semble proche des méthodes utilisées par l'homme pour avoir des informations sur l'environnement qui l'entoure.

3.3.1 Caméra thermique



La caméra thermique est une caméra de surveillance qui filme le jour et la nuit même en absence totale de la lumière. La caméra thermique enregistre les infrarouges émis par des corps à des longueurs d'onde de l'ordre de 10 microns. Cela sert par exemple à détecter la présence d'un corps humain.

Figure 17 : caméra thermique

3.3.2 Caméra analogique

Ces caméras sont reliées par un câble coaxial à un téléviseur ou un écran, où les images s'affichent. Elles envoient des flux continus de données vers un dispositif de stockage. Ce type de capteur est impuissant de percevoir dans un milieu sombre.

3.3.3 Les caméra stéréoscopiques¹

Lorsque nous avons deux caméras observant la même partie de l'environnement à partir de deux points de vue différentes, il est possible d'estimer la distance des objets et d'avoir ainsi une image de profondeur, qui peut être utilisée pour l'évitement d'obstacles ou la cartographie. Cette méthode suppose toutefois un minimum d'éléments saillants dans l'environnement (ou un minimum de texture) et peut être limitée, par exemple dans un environnement dont les murs sont peints de couleurs uniformes.

3.4 Evitement d'obstacle

L'évitement d'obstacles est nécessaire et représente l'une des problématiques dans le domaine de la robotique mobile. Pour assurer la sécurité du robot, il est nécessaire qu'il se déplace sans collision avec des obstacles. Pour cela, le robot mobile doit être autonome et intelligent, il doit être doté d'un comportement d'évitement d'obstacles qui lui permet de se déplacer dans son environnement.

En effet les méthodes d'évitement d'obstacles sont des méthodes locales qui permettent d'assurer la sécurité du robot dans un temps très court. On peut diviser ces méthodes en deux catégories :

- La première catégorie consiste à calculer un ensemble de solutions potentielles en se basant sur des informations sur l'environnement puis choisir une solution afin de satisfaire des contraintes de tâches. Les solutions peuvent prendre une forme de direction privilégiée du robot ou d'une consigne en vitesse. Ces méthodes appelées méthodes délibératives.
- La deuxième catégorie consiste à calculer une commande en se basant sur des informations disponibles sur l'environnement. Ce sont des approches de l'intelligence artificielle inspirées des réseaux de neurones artificiels, zone de déformation virtuelles ZDF, logique floue ou sur les approches basées sur des phénomènes physiques nature comme les champs de potentiels. Ces méthodes sont appelées des méthodes réactives.

3.5 Conclusion

Nous venons de présenter les capteurs généralement utilisés pour le suivi et pour la détection d'obstacles, ainsi que les méthodes de traitement qui y sont associées. Une large gamme d'applications suppose donc l'emploi de plusieurs capteurs, mais les impératifs économiques recommandent plutôt pour un nombre minimal de capteurs embarqués. Il semble donc nécessaire de bâtir une architecture de suivi et de détection qui couvre une plage de fonctionnement maximale, avec un nombre minimum de capteurs, tout en assurant une très grande robustesse de fonctionnement.

4. CONCEPTION DU ROBOT

4.1 Introduction

La mission du robot doit accomplir les tâches de la navigation autonome avec la détection et l'évitement des obstacles imprévus.

Notre conception s'est articulée sur trois grande parties :

- Une première partie présente une description mécanique du robot mobile.
- Une deuxième partie définit la partie électronique.
- Une troisième partie présente les différentes procédures et programmes utilisés pour la commande du robot.

4.2 Conception mécanique

4.2.1 Servo moteur SG90

Les servos sont des moteurs à courant continu avec des circuits intégrés et des boucles de contrôle de retour. Et aucun pilote de moteur requis. Ils sont extrêmement populaires avec les robots, les avions radio commandés. La plupart des servomoteurs peuvent tourner d'environ 90 à 180 degrés. Certains tournent à 360 degrés ou plus.



Figure 18 : Servo moteur SG90

4.2.1.1 Rôle et choix du servo moteur

Notre robot a besoin de faire un balayage à gauche et à droite pour voir s'il y a des obstacles et choisir le chemin le plus ouvert et le plus propre pour le traverser. Pour effectuer cette tâche le robot besoin d'une partie mobile comme le cou humain pour voir à gauche et à droite.

Nous trouvons la solution dans un composant électromécanique appelé servo moteur qui peut tourner en un angle bien déterminé selon la commande donnée par notre carte Arduino.

Après une observation sur les marques des servo moteur disponibles au marché et suite d'une étude des performances, de la compatibilité avec la carte et le prix nous avons choisi le servo moteur SG90 qui a les caractéristiques suivantes :

- Dimension : 22 mm x 11.5 mm x 22.5 mm
- Poids net : 9 g
- Vitesse de fonctionnement : **0,12 secondes / 60 degrés** (4.8V à vide)
- Amplitude de rotation **180°**
- Couple de blocage (4.8V) : **1 kg/cm**
- Plage de température : -30 à +60°C
- Tension de fonctionnement : entre 3.0V et 7.2V
- Compatible avec tout type des cartes Arduino
- Fil de connexion 150 mm, connecteur 3 fil standard.
- Excellent rapport qualité / prix (10 Dinars environ).

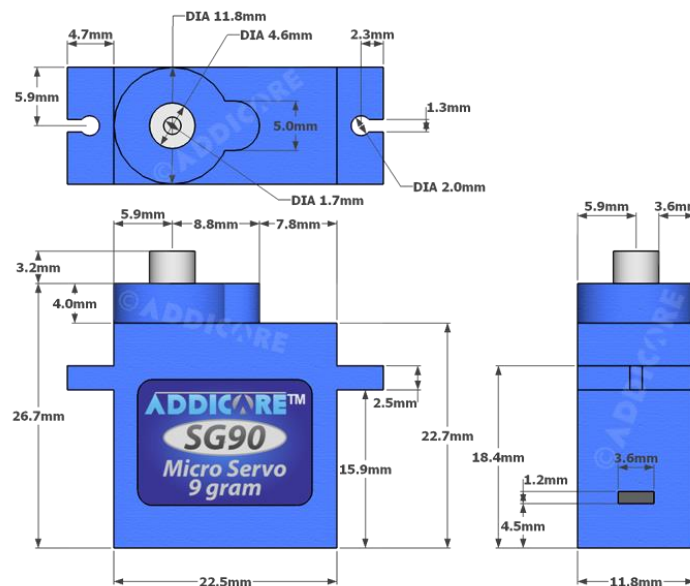
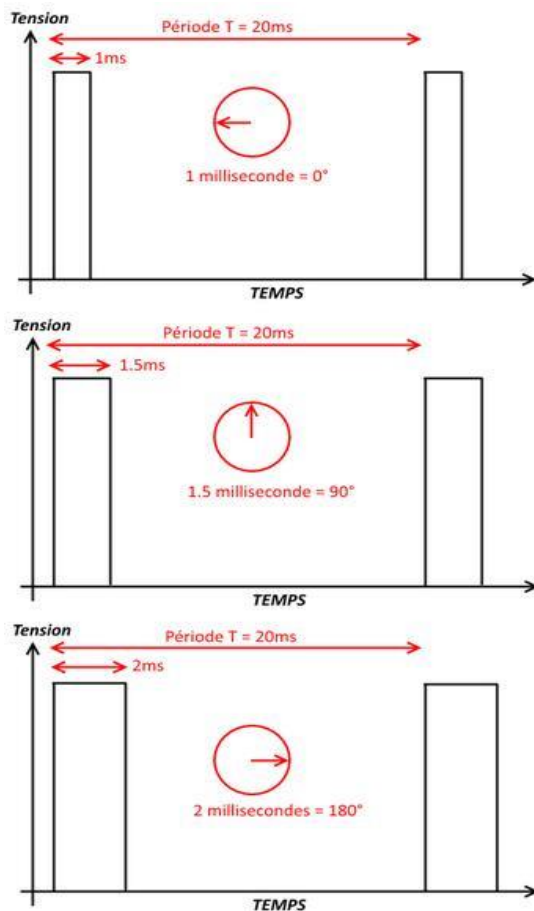


Figure 19 : Dimensions du servo moteur SG90

Remarquons que les bonnes caractéristiques de ce composant comme le poids léger, le coût faible, le couple élevé (par rapport à nos besoins) et la consommation réduite, représentent un fort argument qui nous encourage à choisir ce type de servo moteur.

4.2.1.2 Principe de fonctionnement :



Lorsque notre carte donne une consigne au servo moteur, cette consigne est transmise au moyen d'un signal numérique, d'une impulsion pour être précis (PWM).

Pour que le servomoteur reste à une position donnée, il faut transmettre toutes les 20 millisecondes (soit à une fréquence de 50Hz) une impulsion d'une longueur comprise entre 1 et 2 millisecondes :

- Une impulsion de 1 milliseconde correspond à un angle de 0° .
- Une impulsion de 2 millisecondes correspond à un angle de 180° .
- En envoyant une impulsion d'une longueur intermédiaire, on obtient des angles différents, par exemple 90° est obtenu avec une impulsion de 1.5 milliseconde.

4.2.1.3 Branchement du servo moteur

Le servo SG90 contient trois broches : deux broches d'alimentation (Vcc et GND) et une broche pour recevoir les impulsions des consignes à partir de la carte.

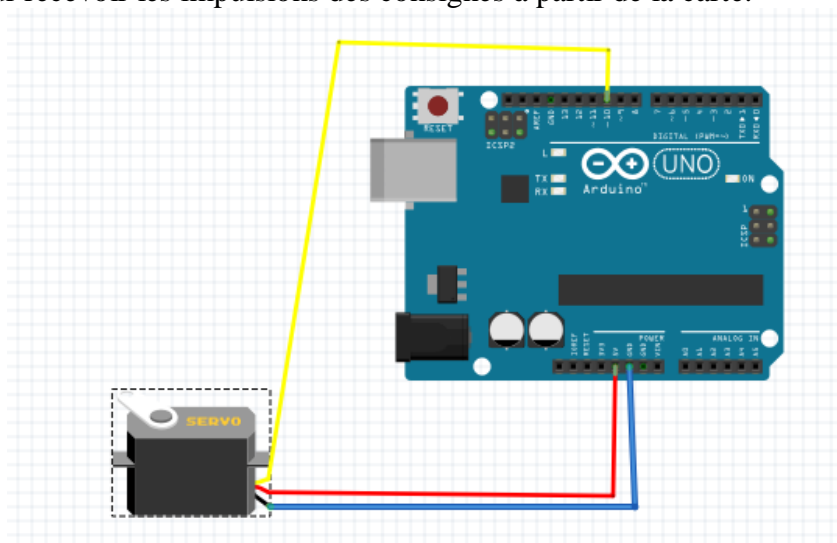


Figure 20 : Branchement du servo moteur avec la carte Arduino

4.2.2 Châssis à quatre roues motrices

Le corps principal de ce robot est son châssis, cette dernière porte les différentes composantes électroniques et mécaniques. Ce châssis contient principalement quatre roues entraînées par quatre moteurs à courant continu à travers de quatre réducteurs

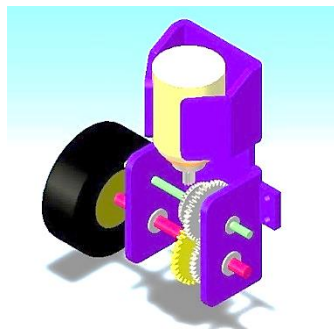


Figure 21 : Roue motrice à réducteur

4.2.2.1 Moteur à courant continu

Selon nos besoins nous avons choisi le moteur courant continu F130 pour mettre ce robot en mouvement :



Figure 22 : Moteur à courant continu F130

Les caractéristiques de ce moteur sont :

- Puissance utile : 4,3W
- Vitesse de rotation : 11600 tr/min
- Couple maxi : 1,47 N.cm
- Longueur arbre : 9,7 mm
- Tension nominale : 12V/DC
- Courant à vide : 0,08 A
- Longueur : 30,6 mm
- Largeur : 18,3 mm
- Hauteur : 24,2 mm
- Courant max : 0,143
- Poids : 38 g

4.2.2.2 Réducteur

Le couple fourni directement par notre moteur est assez faible pour entrainer les roues ce qui nous pousse à utiliser des réducteurs de vitesse. Le réducteur est composé d'un boîtier des engrenages couplés de telles sortes qu'ils puissent réduire la vitesse et augmenter le couple puisque la puissance fournie par le moteur est constante et elle est le produit du couple et de vitesse :

$$P = C . V$$

Ce réducteur nous donne en sortie :

Rapport d'engrenage : 1 / 120

Vitesse hors charge : 100 tr/min



Figure 23 : un exemple d'un motoréducteur

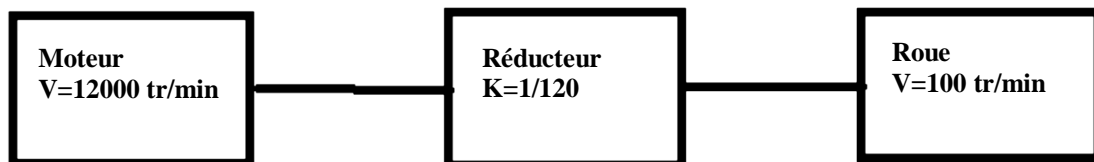


Figure 24 : Réduction de vitesse

4.2.2.3 Châssis

Le corps principal de ce robot est son châssis, ce dernier doit être suffisamment large de telle sorte qu'il puisse embarquer les différentes composantes électroniques et mécaniques du robot.

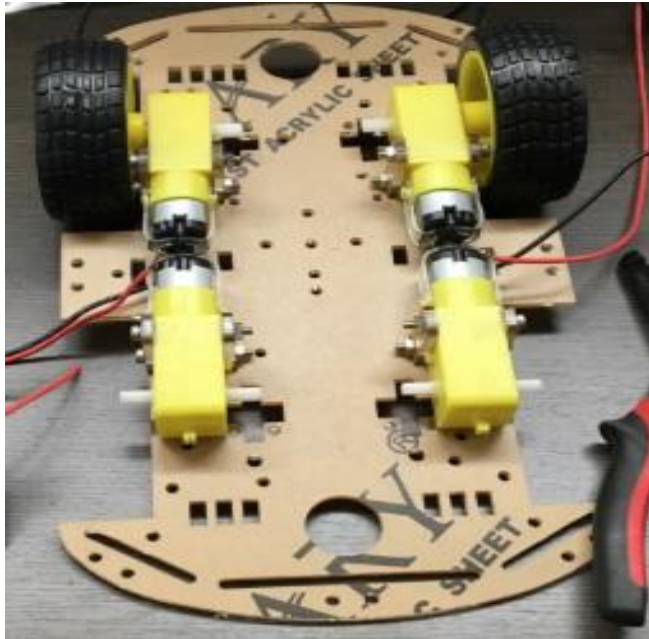


Figure 25 : Châssis du robot

4.3 Conception électrique

4.3.1 Carte Arduino UNO²

Arduino est une plate-forme de prototypage d'objets interactifs à usage créatif constituée d'une carte électronique et d'un environnement de programmation. Elle est basée sur un microcontrôleur qui peut être programmé pour analyser et produire des signaux électriques, de manière à effectuer des tâches très diverses comme la domotique (le contrôle des appareils domestiques : éclairage, chauffage...), le pilotage d'un robot, de l'informatique embarquée, ...etc.

En effet, cet environnement matériel et logiciel permet à l'utilisateur de formuler ses projets par l'expérimentation directe avec l'aide de nombreuses ressources disponibles en ligne sans tout ne connaître ni tout comprendre de l'électronique.

Pont tendu entre le monde réel et le monde numérique, Arduino permet d'étendre les capacités des relations humain/machine ou environnement/machine.

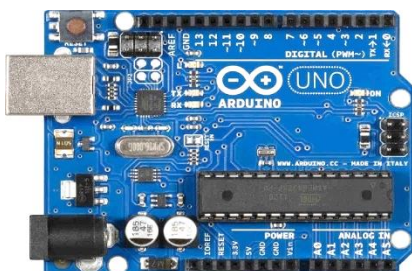


Figure 26 : Carte Arduino UNO

4.3.2 Description matérielle de la carte Arduino

Le modèle UNO présenté sur la figure 1 de la société Arduino est une carte électronique dont le cœur est un microcontrôleur ATMEL de référence ATmega328. Le microcontrôleur ATmega328 est un microcontrôleur 8bits de la famille AVR dont la programmation peut être réalisée en langage C. Cette carte admet un résonateur céramique 16 MHz.

4.3.2.1 Le microcontrôleur ATmega328

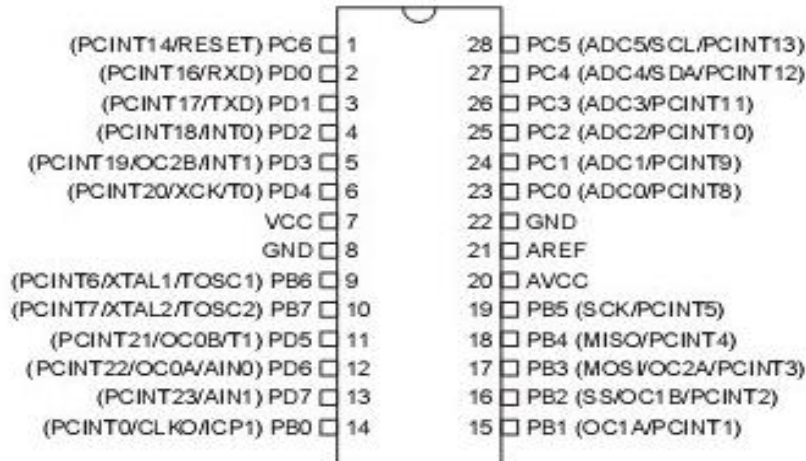


Figure 27 : Microcontrôleur ATmega328

Les caractéristiques principales de ce microcontrôleur sont :

- **FLASH** : mémoire programme de 32Ko (dont 0.5 Ko sont utilisés pour le programme de démarrage). Cette mémoire est l'équivalent du disque dur pour l'ordinateur. C'est la place que nous avons pour stocker le programme.
- **SRAM** : données volatiles 2Ko. Elle sert à stocker le résultat des variables
- **EEPROM** : données non volatiles 1Ko. Elle permet de sauvegarder des valeurs de variables et ceci même à l'extinction de la carte, donc le nombre des réécritures est limité.
- **Digital I/O** (entrées-sorties Tout Ou Rien) : 3 ports Port B, Port C, Port D (soit 23 broches en tout I/O)
- **PWM**: 6 broches OC0A(PD6), OC0B(PD5), OC1A(PB1), OC1B(PB3), OC2A(PB3), OC2B(PD3)
- **Port série** (USART) : émission/réception série via les broches TXD(PD1) /RXD(PD0)

4.3.2.2 Les entrées / sorties numériques

C'est par ces connexions que le microcontrôleur est relié au monde extérieur, une carte Arduino standard est dotée de :

- 6 entrées analogiques.
- 14 entrées/sorties numériques dont 6 peuvent assurer une sortie PWM.

Les entrées analogiques lui permettent de mesurer une tension variable (entre 0 et 5 V) qui peut provenir de capteurs ou d'interfaces divers (potentiomètres, etc.). Les entrées/sorties numériques reçoivent ou envoient des signaux « 0 » ou « 1 » traduits par 0 ou 5 V. On décide du comportement de ces connecteurs (entrée ou sortie) en général dans l'initialisation du programme mais il peut être aussi changé dans le corps du programme.

Les sorties numériques peuvent actionner de nombreux composants (LED, transistor, etc.), mais elles ne peuvent pas fournir beaucoup de courant (40 mA pour une carte Arduino UNO). Pour piloter des circuits de plus forte puissance, il faut passer par des transistors ou des relais. La puce ATMEGA n'est pas capable de sortir des tensions variables. Heureusement, 6 des sorties numériques (N° 3, 5, 6, 9, 10, 11) peuvent produire un signal PWM. Ce sigle signifie « Pulse Width modulation » en anglais ; en français l'on parle de MLI : "« Modulation de largeur d'impulsion ». Il s'agit d'un artifice permettant de produire une tension variable à partir d'une tension fixe.

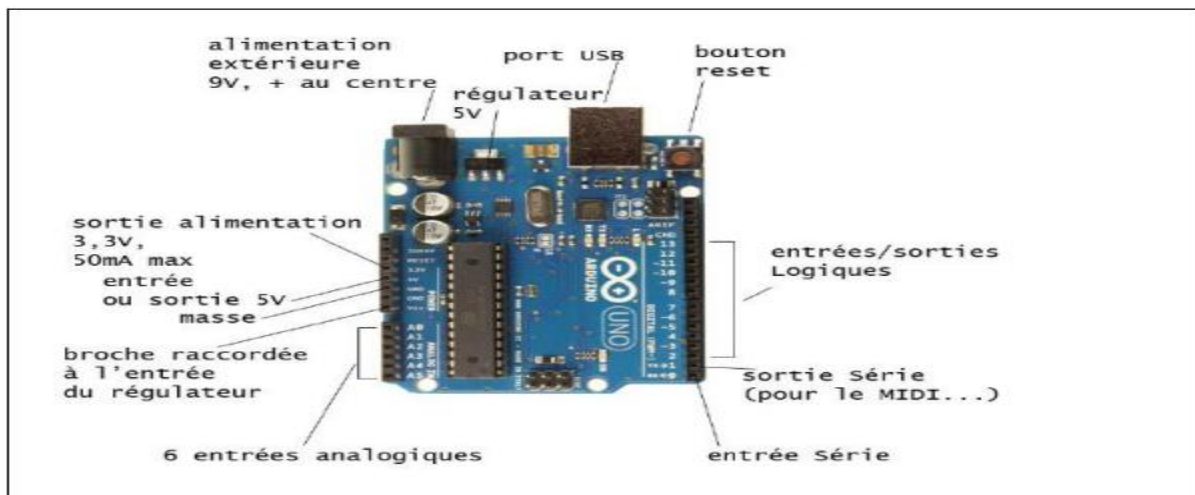


Figure 28 : Les différents entrée/ sortie de la carte Arduino

4.3.3 La carte de puissance L298N

La carte de puissance est un circuit qui se connecte avec la carte Arduino pour fournir la puissance nécessaire au fonctionnement de deux moteurs courants continue ou un moteur pas à pas. Il est conçu pour supporter des tensions plus élevées. Des courants importants tout en proposant une commande logique TTL (basse tension, courant faible, idéal donc pour un microcontrôleur).

4.3.3.1 Caractéristique

- Pont H double : L298N
- Tension : 5V
- Courant : 0-36mA (Courant maximal : 2A (Dans une seule branche))
- Puissance Maximale : 25W
- Dimensions : 43 * 43 * 26mm
- Poids : 26g

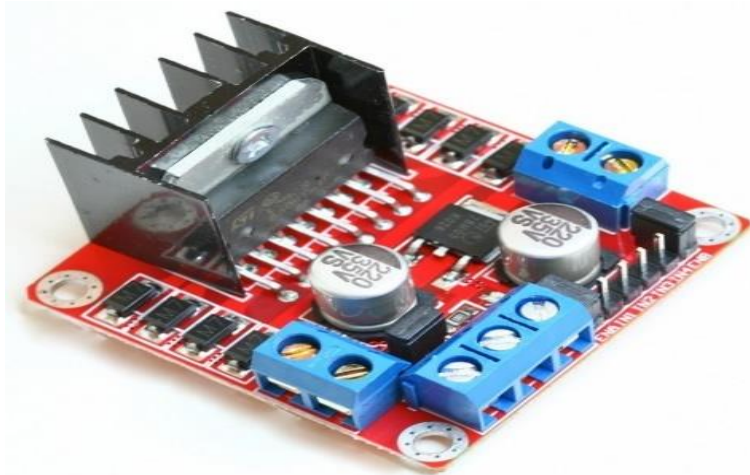


Figure 29 : Carte de puissance L298N

Schéma de branchement :

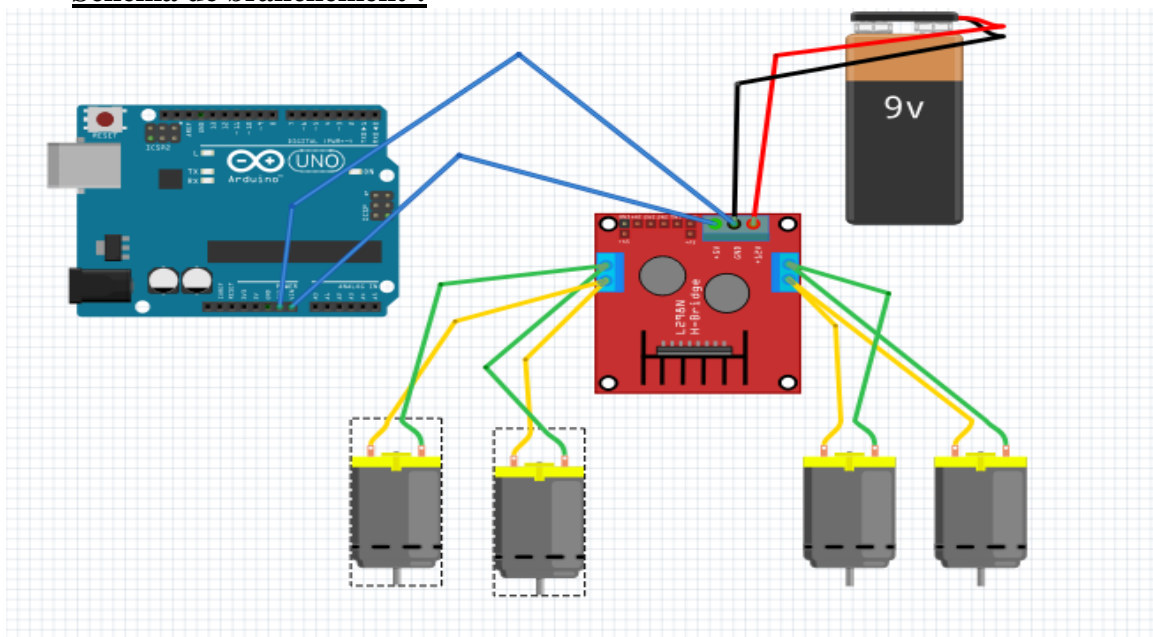


Figure 30 : Schéma de branchement de la carte L298N avec l'Arduino et le moteur

4.3.4 Le capteur ultrason SRF04

4.3.4.1 Présentation

Le capteur SRF04 est un capteur qui utilise les ultrasons pour détecter les obstacles. Il est relativement compact (taille équivalente à un SHARP IR). Il permet de détecter les obstacles de 3 cm à 3 m. Il est très simple d'utilisation : il est alimenté en 5 V et possède une entrée Trigger et une sortie Echo.



Figure 31 : Capteur HC-SR04

4.3.4.2 Principe des ultrasons

Le son est une onde mécanique qui se propage sur un support matériel (ici l'air). Les ultrasons sont des ondes ayant une fréquence inaudible pour l'homme (supérieur à 20kHz). La vitesse de propagation des ondes dépend de leur milieu d'évolution (300m/s dans l'air, 1500m/s dans l'eau). Lorsqu'une onde rencontre un obstacle, une partie de cette onde est réfléchi (elle rebondit sur cet obstacle) alors que l'autre partie est transmise (absorbée par l'obstacle). C'est cette onde réfléchi qui nous intéresse (on l'appelle également echo). Le capteur SRF04 est composé d'un émetteur et d'un récepteur. En mesurant le temps entre l'émission et la réception de l'onde et la vitesse de propagation de l'onde dans le milieu, on peut en déduire la distance de l'obstacle.

4.3.4.3 Connexions du capteur ultrason SRF04 à une carte Arduino

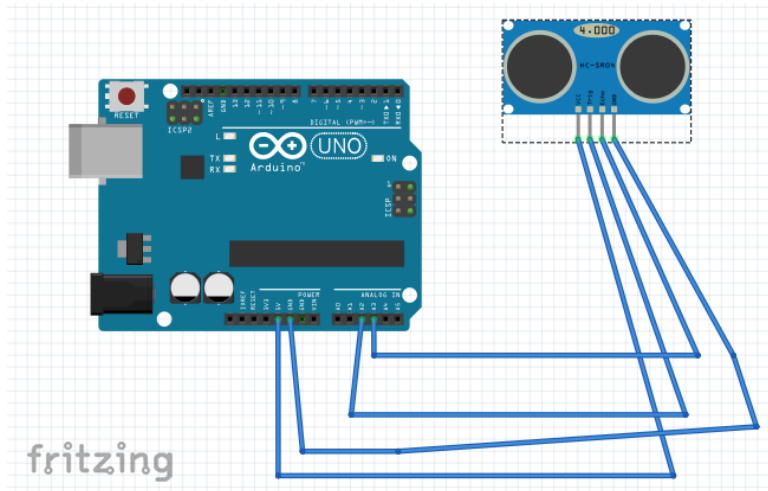


Figure 32 : connexion du SFR04 à la carte Arduino

Le capteur Ultrason doit être branché comme suit :

- Le VCC se connecte à l'alimentation 5V.
- Le GND se connecte au ground.
- Le Trig et l'écho se connectent à n'importe quels pin (dans notre cas A2 et A3)

4.3.5 Détecteur de gaz (GPL, Méthane, Butane, hydrogène, fumée) MQ-5

Le MQ-5 est un capteur qui permet de détecter du gaz ou de fumée à des concentrations de 300 ppm à 10000 ppm. Après calibration, le MQ-5 peut détecter différents gaz comme le GPL (LPG), l'i-butane, le propane, le méthane, l'alcool, l'hydrogène ainsi que les fumées. Il est conçu pour un usage intérieur à température ambiante. Le MQ5 doit être alimenté en 5V pour le capteur physico-chimique puisse atteindre sa température de fonctionnement. Il dispose d'une sortie analogique et d'un réglage de la sensibilité par potentiomètre.



Figure 33 : Détecteur de Gaz

Le senseur MQ-5 est un senseur avec une sortie analogique (AOut) qui signale la présence de fumée en élevant la tension en sortie. Plus il y a de fumée et plus la tension monte. Il est possible de régler la sensibilité du module à l'aide du potentiomètre se trouvant à l'arrière du module, ce dernier permet d'ajuster un seuil d'activation pour le signal digital (DOut) qui change lorsque le seuil est atteint. Ce qui permet de déclencher une action (effet sonore d'un buzzer).

Les Caractéristiques du MQ5 sont :

- Puce principale : LM393, ZYMQ-2 détecteur de gaz
- Haute sensibilité et bonne sélectivité
- Tension de fonctionnement : 5V DC
- Tension de sortie analogique : 0 ~ 5V (plus la concentration est élevée, plus la tension est élevée)
- Plage de détection : 200 à 10000ppm
- Longue durée de vie et stabilité fiable

Les différents pins d'un capteur MQ5 sont :

- VCC : alimentation positive (5V)
- GND : alimentation négative
- DO : sortie du signal du commutateur TTL
- AO : sortie du signal analogique

Quatre trous de vis pour un positionnement facile

Dimensions : 32 x 22 x 27mm

4.3.6 Buzzer

Un buzzer est un dispositif de signalisation audio, qui produit un effet sonore lorsqu'il est excité.

Il existe deux types de buzzers : buzzers électromécaniques et buzzers piézo-électriques.

Les buzzers électromécaniques sont représentés sous la forme d'un petit boîtier rectangulaire ou cylindrique, avec connexion électrique rigide pour la fixation directe sur circuit imprimé ou avec des connexions électriques constituées du fil électrique souple. Ils fonctionnent sous une tension continue généralement comprise entre 3 V et 28 V.

Les buzzers piézo-électriques sont constitués d'un diaphragme piézoélectrique, d'une cavité avec un orifice et de connexions pour les bornes électriques.

Pour fonctionner, ce type du buzzer nécessite une tension alternative de 3 V à 30 V avec une fréquence qui varie entre 2 KHz et 10 KHz.

Les buzzers piézo-électriques sont utilisés pour émettre des bips, des tonalités et des alertes.



Figure 34 : Buzzer piézo-électrique.

4.3.7 W1209 Contrôle de la température du thermostat Commutateur :

Cet appareil est idéal pour contrôler des lampes ou 12 volts réchauffeurs / refroidisseurs que vous pourriez avoir. Il suffit de brancher dans une source 12VDC dans le V12 et GND pour alimenter le système, il est livré avec une sonde 52cm qui va commencer à mesurer la température, vous pouvez modifier les paramètres dans le menu pour avoir un départ différé pour mesurer la température. L'utilisation d'un relais on peut faire office de refroidisseur lorsque la température augmente le ventilateur se met en marche ou en tant que dispositif de chauffage lorsque la température diminue tourner sur le système. Choisir un ou l'autre peut être fait dans les réglages de menu. Il est un dispositif simple « plug and Play » qui devrait vous permettre de démarrer avec vos projets que le relais agira comme un interrupteur normalement ouvert, lorsque le thermostat atteint la température cible programmée il activera le relais pour créer un circuit fermé. Livré avec une alarme pour indiquer si la température monte passer la température de votre réglage. Cette fonctionnalité est désactivée par défaut dans les paramètres du menu.

Caractéristiques :

Tension de fonctionnement : 12VDC

Courant de veille : 35mA

Courant Actif : 65mA

Max tension sur le relais : 12VDC

Max actuel sur le relais : 10A

Température de fonctionnement : -10 ~ 60 °C

Humidité de fonctionnement : 20% ~ 85%

Dimension : 48 x 40 mm (W x L)

Longueur du câble : 52cm

Plage de température : -50 à 110 ° C

Fréquence de rafraîchissement : 0.5 sec

La précision d'hystérésis : 0.1 ° C

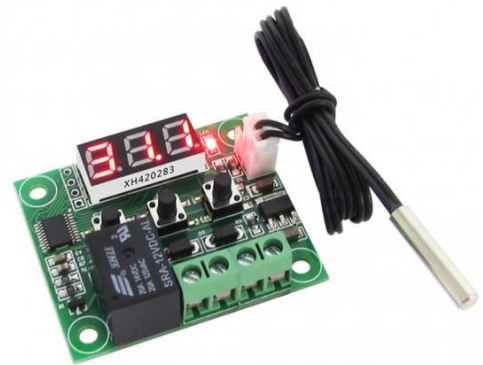


Figure 35 : capteur de température w1209

4.4 Conception informatique

4.4.1 Logiciel de la carte Arduino

4.4.1.1 Description

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

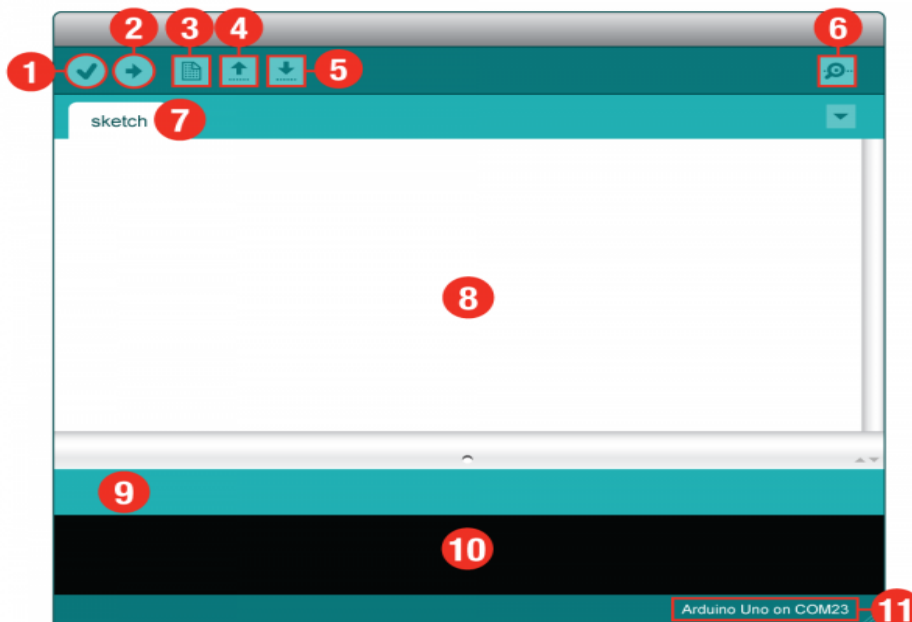


Figure 36 : Interface logicielle Arduino

1-Vérifier : Compile et approuve le code. Il va signaler des erreurs dans la syntaxe (comme des points-virgules manquants ou des parenthèses).

2-Téléverser : envoie le code au tableau 101.

3-Nouveau : ce bouton ouvre un nouvel onglet de fenêtre de code.

4-Ouvrir : cette fonction permet d'ouvrir un croquis existant.

5-Enregistrer : ceci enregistre l'esquisse actuellement active.

6-Moniteur série : Cela ouvrira une fenêtre qui affiche toutes les informations transmises en série. Il est très utile pour le débogage.

7-Nom de l'esquisse : montre le nom de l'esquisse sur laquelle nous travaillons actuellement.

8-Zone de code : c'est la zone où nous composons le code de notre croquis.

9-Zone de message : c'est là que l'IDE nous dit s'il y a des erreurs dans notre code.

10-Console de texte : elle affiche des messages d'erreur complets. Lors du débogage, la console de texte est très utile.

11-Port et série de carte : nous montre quelles carte et sélections de port série.

Chaque code écrit en IDE comporte deux parties :

- Une fonction setup où nous mettons les configurations des broches et initialisons les différentes variables. Cette fonction s'exécute une seule fois au début du programme :

```
void setup() {  
    // put your setup code here, to run once  
}
```

- Une fonction loop où nous mettons le corps du programme et les étapes qui seraient traitées selon l'ordre chronologique infiniment :

```
void loop() {  
    // put your main code here, to run repeatedly  
}
```

5. REALISATION DU ROBOT

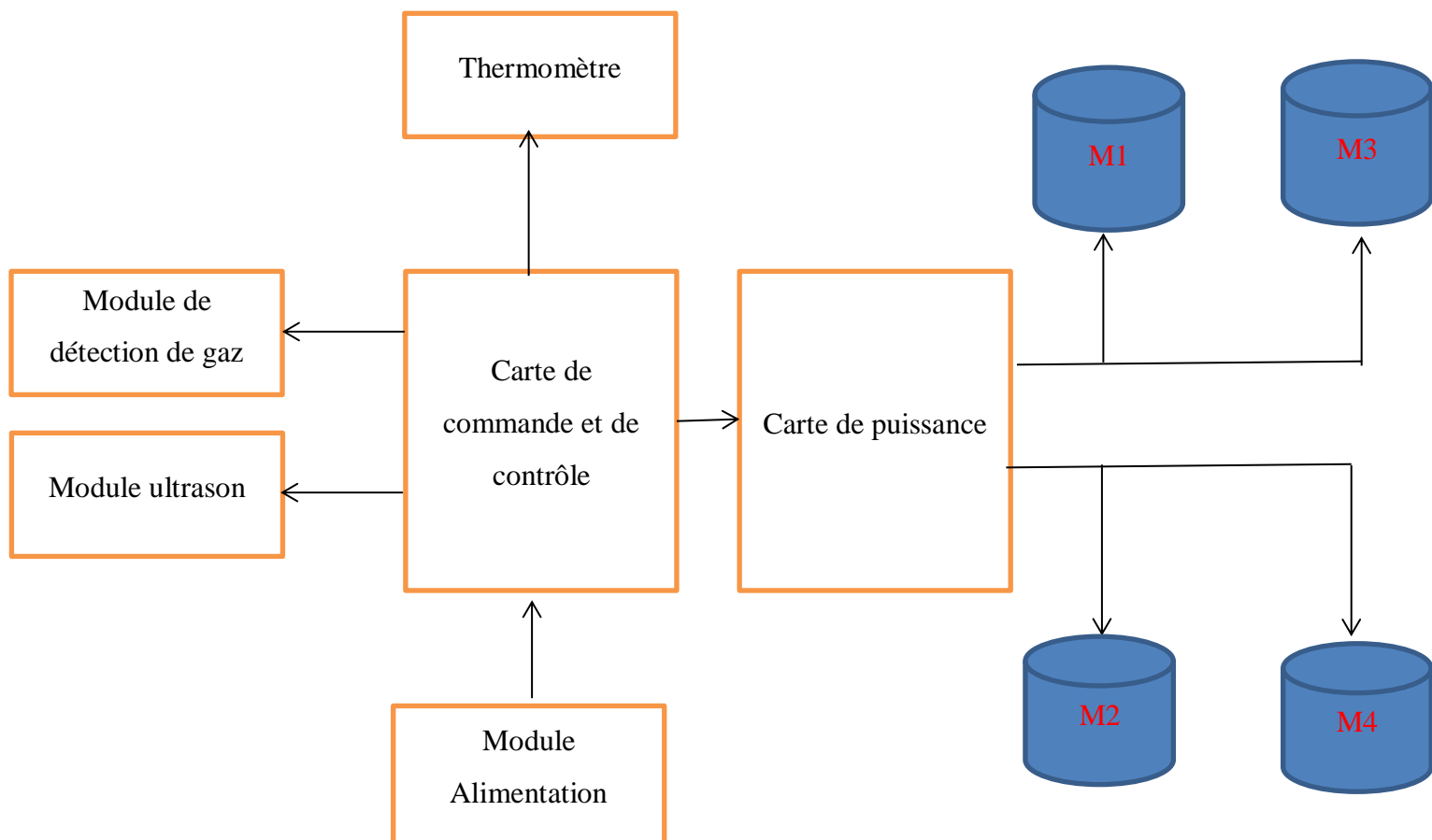
5.1 Introduction

Après avoir achevé la partie de conception et la partie de définition des matérielles utilisés, on va entamer l'étape de réalisation qui représente une partie très importante du temps consacré à ce projet.

L'objectif de ce chapitre est de présenter l'environnement de travail, les étapes de construction et de montrer les tests d'évaluation du produit finale

5.2 L'environnement de travail :

La figure ci-dessous montre le schéma synoptique du robot :



5.3 Schéma électrique

Pour faire le schéma électrique de notre robot, nous avons utilisés le logiciel Fritzing qui est une application de conception de circuit imprimé qui permet de concevoir de façon entièrement graphique le circuit :

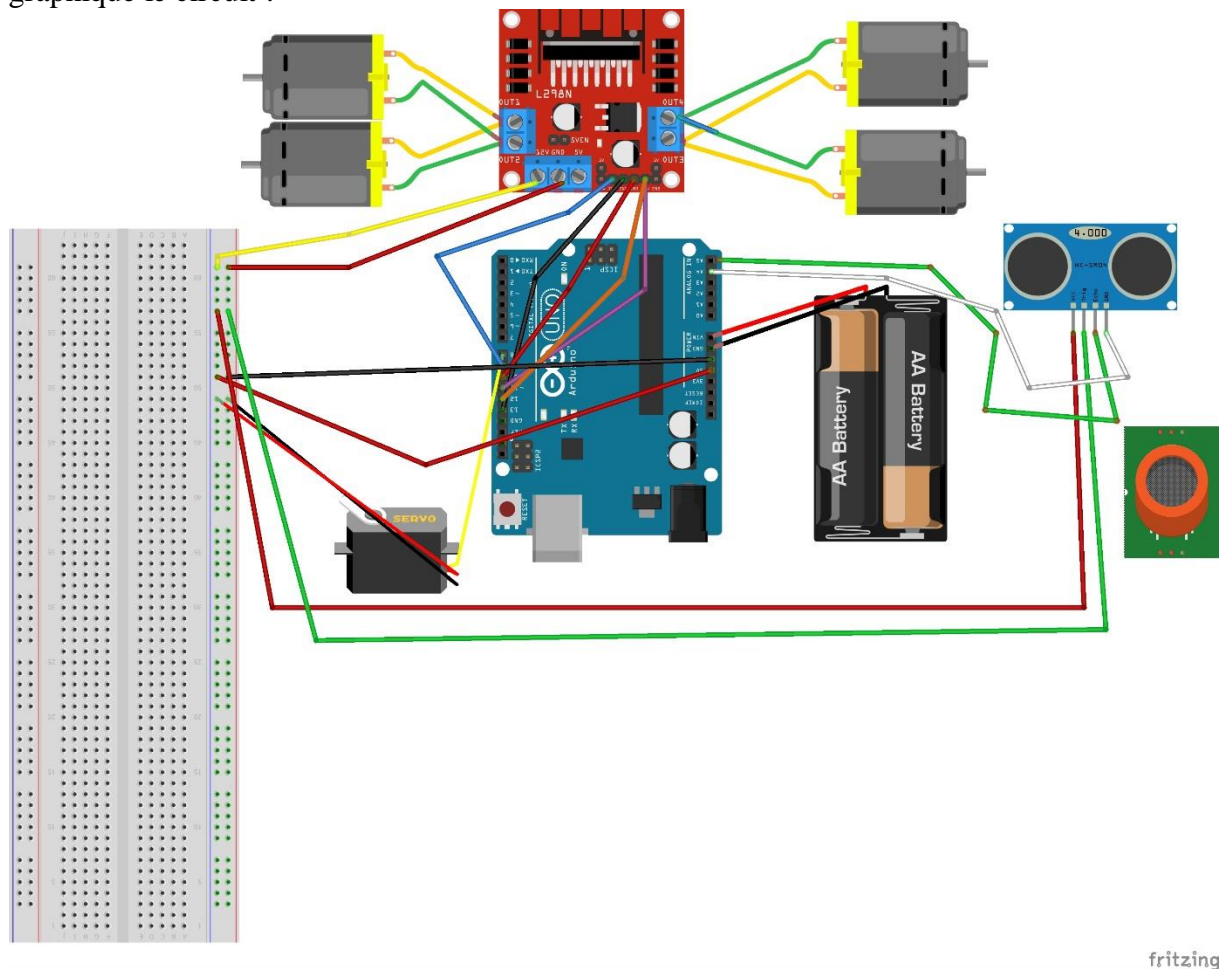


Figure 37 : réalisation du circuit

5.4 Assemblage des composants

Le châssis est un composant existant sur lequel on aura à monter les différents composants dont les moteurs et les roues. Ainsi nous allons commencer d'abord par câbler les moteurs hors du châssis puis les monter sur le châssis et enfin relier ces moteurs aux roues libres comme le montre la figure 33.

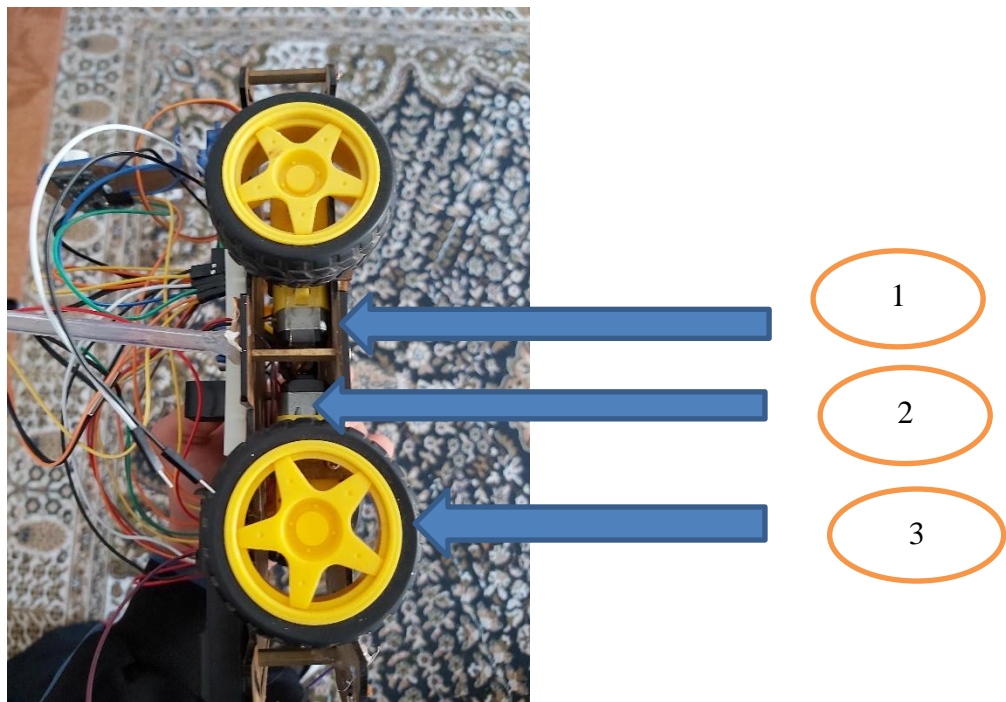


Figure 38 : figure assemblage châssis (1) - moteur (2) - roues (3)

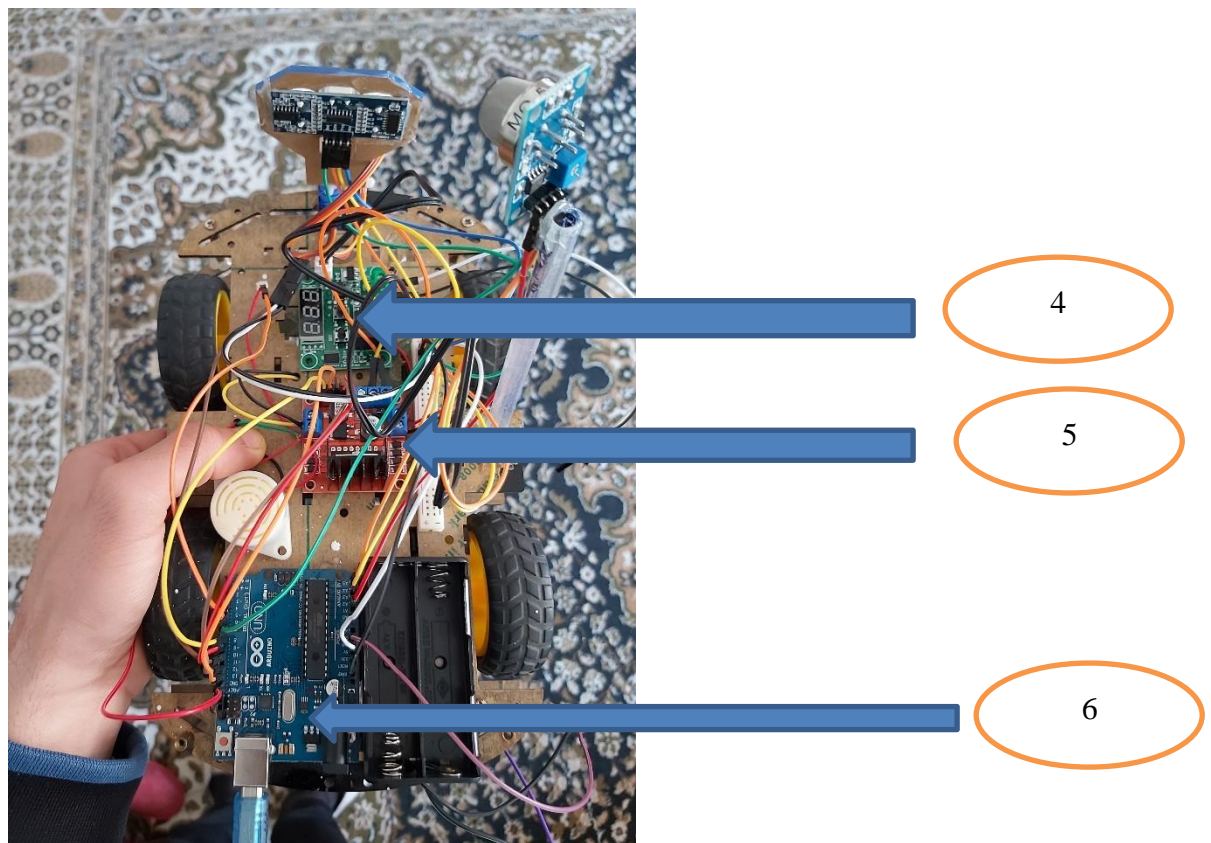


Figure 39 : capteur de température w1209 (4) - montage de carte de puissance (5) – Arduino (6)

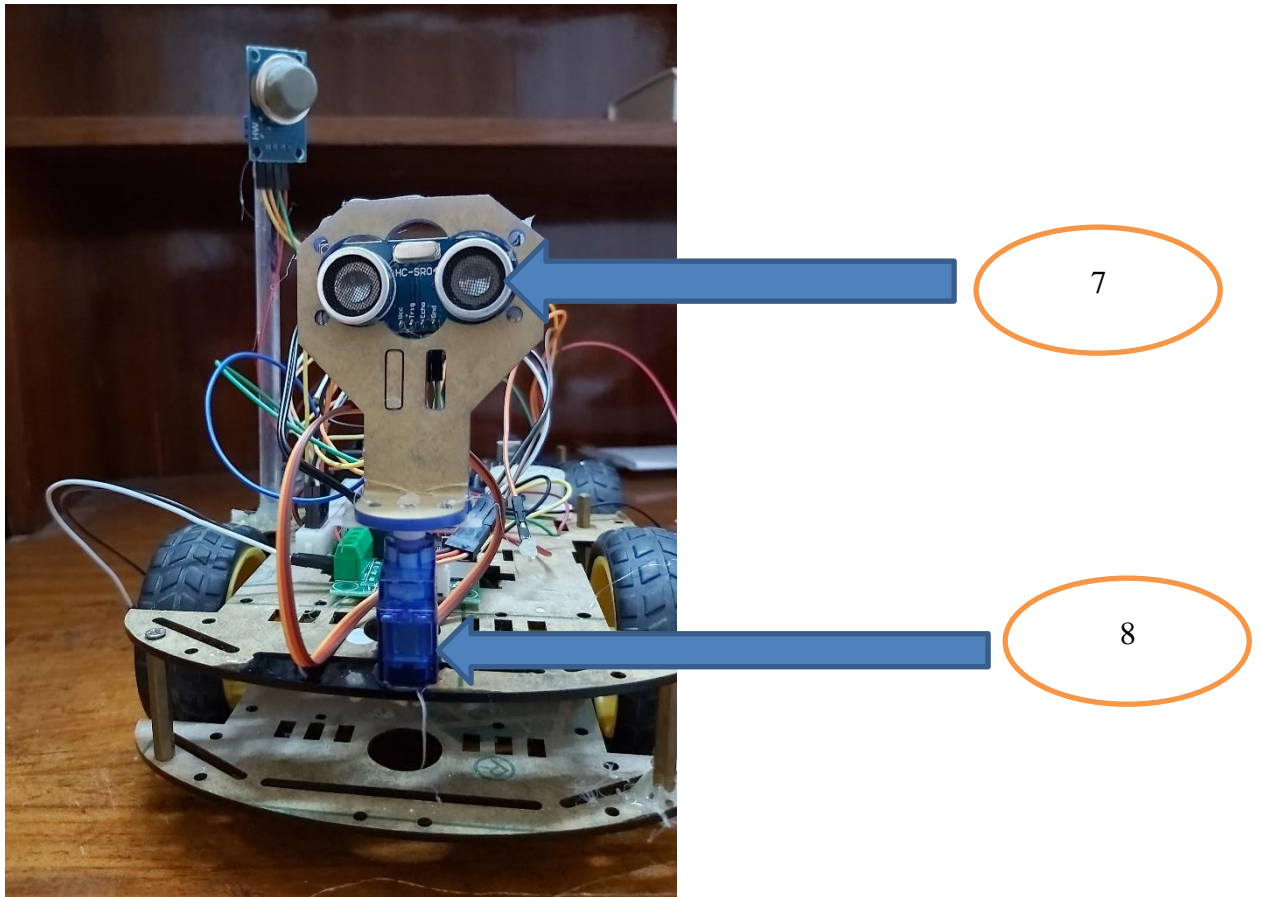


Figure 40 : capteur ultra son (7) - servomoteur (8)

La fin de la mise en place de la carte de puissance- sera suivie du collage de la pile rechargeable et du servomoteur sur le châssis ; le capteur ultra son, lui est monté par collage sur un support au-dessus du servomoteur dans la mesure où celui-ci balaye de gauche à droite pour retrouver un chemin libre lorsqu'il est confronté à un obstacle. Ainsi il faudra faciliter son mouvement en le câblant avec des fils souples pour éviter le retardement du servomoteur.

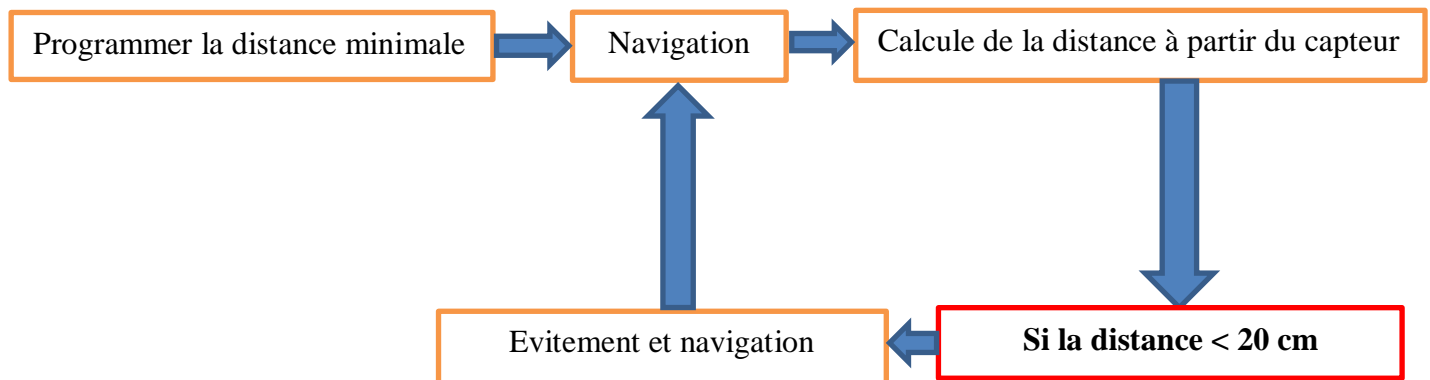
Câblage d'ensemble :

Lorsque ces différents composants cités ci-dessous sont assemblés sur le châssis, il faudra maintenant les relier par des fils électriques adéquats pour assurer leur bon fonctionnement. Ainsi le tout est fait et notre robot pourra commencer à fonctionner comme il le faut sans dérive.

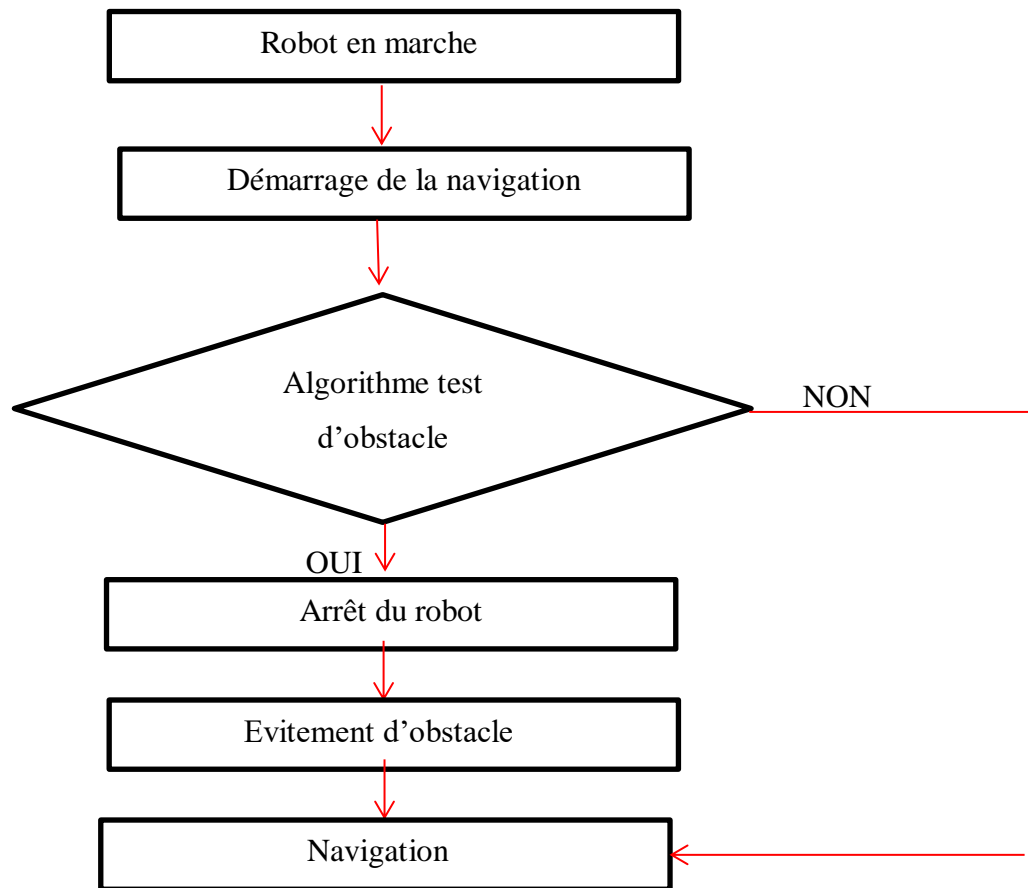
5.5 Programmation du robot

Après avoir effectué les connexions de notre circuit, du côté matériel, notre robot est Terminé. Tout ce que nous devons faire ensuite est de brancher notre arduino sur notre ordinateur et téléverser le programme.

5.5.1 Algorithme de détection d'obstacle



5.5.2 Algorithme principale de mouvement de notre robot :



5.5.3 Les bibliothèques utilisées :

On a utilisé deux bibliothèques

- La bibliothèque <Servo.h> qui permet de commander et de positionner le servomoteur à n'importe quelle position entre 0 et 180 degrés.
- La bibliothèque <NewPing.h> qui permet au capteur ultrason de déterminer la distance aux obstacles.

5.5.4 Les constantes

Pour notre programme, nous avons utilisé de nombreuses constantes globales, définies en tout début de programme. Toutes ces constantes sont de type entier. Elles représentent deux choses : ou bien les valeurs des broches et leur nom associé, ou bien des valeurs de constantes utilisés dans le programme.

5.6 Conclusion

La réalisation du robot était divisée en deux grandes parties :

- Une partie hardware qui est la réalisation électronique et mécanique du robot avec l'assemblage des composants
- Une partie software qui est basé essentiellement sur la programmation en langage C/Arduino.

6. CONCLUSION

Ce travail est d'une importance capitale dans la mesure où il nous permet d'acquérir plus de connaissances théoriques et pratiques dans divers domaines tant technique qu'informatique. En effet par le biais de ce projet nous avons conçu un système de navigation autonome pour un robot mobile avec détection et évitement d'obstacles.

Ainsi, nous avons commencé d'abord par présenter un aperçu général sur la robotique mobile.

Ensuite, nous avons fait une analyse des problèmes de détection et de déviation d'obstacles auxquels sont confrontés les robots. En réalité, la déviation d'obstacles est une étape nécessaire pour naviguer et atteindre la destination désirée dans un milieu qui contient un ou plusieurs obstacles. Des résultats issus de notre travail, nous ont permis d'équiper notre robot d'un système permettant de naviguer d'un point à un autre tout en assurant la détection et la déviation des différents obstacles, ces résultats ont été des lors validés non seulement dans les environnements intérieur et extérieurs mais également par la plate de simulation V rep.

Afin de perfectionner le fonctionnement du robot, d'autre capteurs peuvent être embarqués pour améliorer la navigation dans des milieux complexes présentant des conditions plus sévères. Ce qui permet d'évaluer l'environnement dans lequel peut évoluer le robot et sa capacité à résister aux contraintes extérieures auxquels il est soumis.

Bibliographie

[1] : P. REIGNIER. Pilotage réactif d'un robot mobile 'étude du lien entre la perception et l'action. Thèse de doctorat, Institut national polytechnique de Grenoble, 1994.

[2] : R. I. Hartley and A. Zisserman. Multiple View Geometry in Computer Vision. Cambridge University Press, ISBN: 0521540518, second edition, 2004.

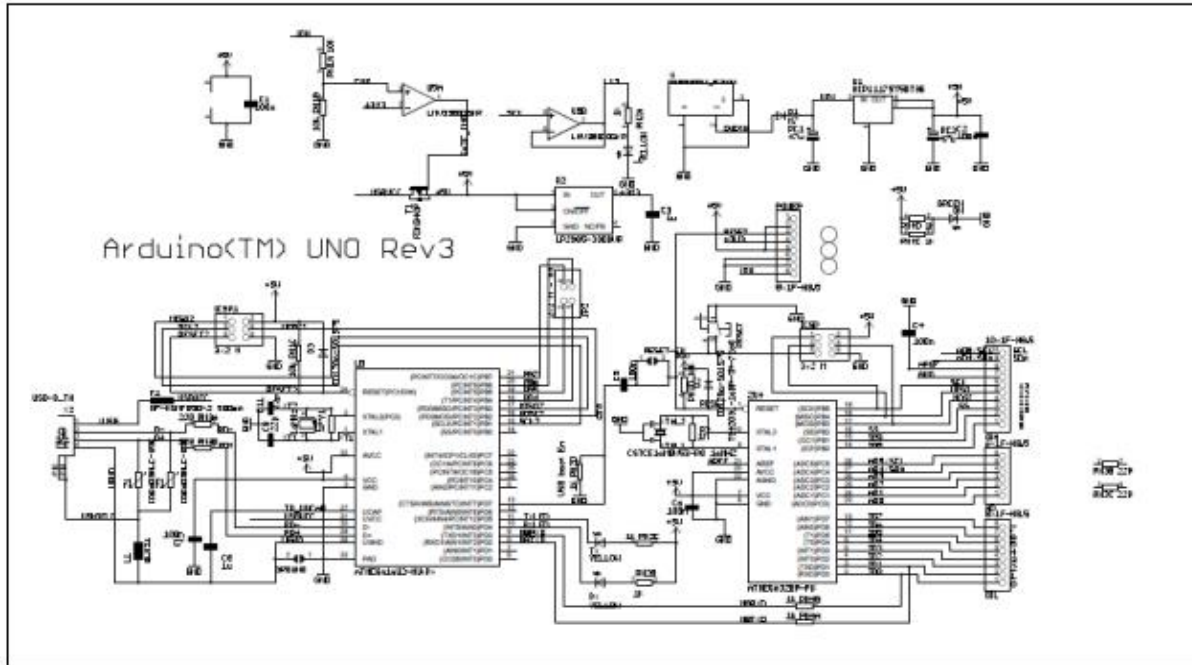
[3]: *Contrôler un servomoteur avec une carte Arduino*, Carnet du maker. [**en ligne**]. Disponible sur : <https://www.carnetdumaker.net/articles/controler-un-servomoteur-avec-une-carte-arduino-genuino/> (consulté le 02 /06/2019)

[4]: *Introduction à Arduino*, Université de Lomoges. [**En ligne**]. Disponible sur : http://www.unilim.fr/pages_perso/deneuille/docs/Info2PC/Cours4handout.pdf (consulté le 05 /06/2019)

[5]: <https://boutique.semageek.com/fr/594-accelerometre-3-axes-et-magnetometre-lsm303.html> (consulté le 05 /04/2019)

[6]: <https://chene-bleu.info/i/ladifference/vweb2.asp?quelarticle=103060>

Annexe1 : Fiche technique (Datasheet) de la carte Arduino UNO



Les caractéristiques de carte Arduino uno :

- Microcontrôleur : ATmega328.
- Tension d'alimentation interne = 5V.
- Tension d'alimentation (recommandée)= 7 à 12V, limites =6 à 20 V.
- Entrées/sorties numériques : 14 dont 6 sorties PWM.
- Entrées analogiques = 6.
- Courant max par broches E/S = 40 mA.
- Courant max sur sortie 3,3V = 50mA.
- Mémoire Flash 32 KB dont 0.5 KB utilisée par le bootloader.
- Mémoire SRAM 2 KB.
- Mémoire EEPROM 1 KB.
- Fréquence horloge = 16 MHz.
- Dimensions = 68.6mm x 53.3mm.

6.1 Circuits additionnels de l'Arduino

Il est possible de spécialiser la carte Arduino en l'associant avec des circuits additionnels que l'on peut fabriquer soi-même ou acheter déjà montés. Lorsqu'ils se branchent directement sur la carte, ces circuits s'appellent des « Shields » ou cartes d'extension. Ces circuits spécialisés apportent au système des fonctionnalités diverses et étendues dont voici quelques exemples :

- Ethernet : communication réseau.
- Bluetooth : communication sans fil.
- Pilotage de moteurs (pas à pas ou à courant continu).
- Pilotage de matrices de LEDs : pour piloter de nombreuses LEDs avec peu de sorties.
- Ecran LCD : pour afficher des informations.
- Lecteur de carte mémoire : lire ou stocker des données.
- Lecteur de MP3.
- GPS : pour avoir une information de position géographique.
- Joystick.

Annexe 2 : Le code

```
#include <Servo.h>                                //Servo motor library. This is
standard library
#include <NewPing.h>                                //Ultrasonic sensor function
library. You must install this library
//les pin réservés au shield L298
const int M1_Direction = 12; // précise la direction du moteur 1 ( avance si HIGH)
const int M1_Frein = 9; // freine mot 1 si HIGH
const int M2_Direction = 13; // précise la direction du moteur 2 ( avance si HIGH)
const int M2_Frein = 8; // freine mot 2 si HIGH
const int buzzer = 6;

int pin_d = 7; // Senseur DOUT (digitale)
int pin_a = A2; // Senseur AOUT (analogique)

int niveau_senseur = 300;

//les pins du capteur ultrasons
#define trig_pin A4                                //le pin trig(sortie) du capteur ultrasons est
attaché à A4
#define echo_pin A5                                //le pin echo (entrée) du capteur ultrasons est
attaché à A5
#define distance_maximal 200                        // définir un constante de distance maximal
dont le capteur le prend en compte*
boolean goesForward = false;                        // l'état du robot
int distance = 100;

NewPing sonar (trig_pin, echo_pin, distance_maximal); // initialisation de la
fonction newPing par (A4,A5,200cm distance maximale*)
Servo servo_moteur;                                //nommer le servo moteur
```

```

////////////////////////////////////VOID
SETUP////////////////////////////////////

void setup(){

                                //configuration des pin dédiés au shield L298

comme sorties
    pinMode(M1_Direction, OUTPUT);
    pinMode(A3, OUTPUT);
    pinMode(M1_Frein, OUTPUT);
    pinMode(M2_Direction, OUTPUT);
    pinMode(M2_Frein, OUTPUT);
    pinMode(pin_d, INPUT);
    pinMode(pin_a, INPUT);

    // Définir le buzzer et LEDs comme sortie

    pinMode(buzzer, OUTPUT);

    servo_moteur.attach(10);                                //attacher le servomoteur sur la pin 10
    digitalWrite(A3, HIGH);
    servo_moteur.write(90);                                //deplacer le servo moteur 45° de 90°
vers 115°
    delay(2000);                                            //attendre 2 sec
    distance = readPing();                                //lire le distance en cm
    delay(100);                                            //attendre 0.1 sec
    distance = readPing();                                //lire le distance en cm
    delay(100);                                            //attendre 0.1 sec
    distance = readPing();                                //lire le distance en cm
    delay(100);                                            //attendre 0.1 sec
    distance = readPing();                                //lire le distance en cm
    delay(100);                                            //attendre 0.1 sec
}

```

///VOID

LOOP///

```
void loop(){
digitalWrite(buzzer, LOW);
  int distanceRight = 0;
  int distanceLeft = 0;
  delay(50);
  // Lecture de DOUT du senseur sur l'entree digital
  int valeur_digital = digitalRead(pin_d);
  // Lecture de AOUT du senseur sur l'entree analogique
  int valeur_analogique = analogRead(pin_a);
  if (valeur_analogique > niveau_senseur)
  {

digitalWrite(buzzer, HIGH);
  }
else {
  if ((distance <= 40)&&(distance >= 7 )) {                                //si il y a un obstacle
    moveStop();                                                            //alors freine le robot
    delay(300);                                                            //attendre 0.3 sec
    moveBackward();                                                        // et recule le robot
    delay(500);                                                            //attendre 0.4 sec
    moveStop();                                                            // freine le robot
    delay(300);                                                            //attendre 0.3 sec
    distanceRight = lookRight();                                           // voir à droite et lire la distance
    delay(900);                                                            // attendre 0.3 sec
    distanceLeft = lookLeft();                                             // voir à gauche et lire la distance
    delay(900);                                                            //attendre 0.3 sec
```



```

        if (distance >= distanceLeft){
            gauche
            turnRight();
            moveStop();
        }
        else{
            turnLeft();
            moveStop();
        }
    }
    else
    {
        moveForward();
        marche avant
    }
    distance = readPing();
    // tout en lisant la distance
}

```

```

//////////////////////////////////////////////////////////////////LA
LOOKRIGHT//////////////////////////////////////////////////////////////////

```

```

int lookRight(){
    //cette fonction retourne la distance
    à droite du robot
    servo_moteur.write(10);
    delay(800);
    int distance = readPing();
    delay(100);
    servo_moteur.write(90);
    return distance;
}

```

```

//////////////////////////////////////////////////////////////////LA
LOOKLEFT//////////////////////////////////////////////////////////////////

```

```

int lookLeft() {
    //cette fonction retourne la distance à
    gauche du robot
    servo_moteur.write(170);
    delay(800);
    int distance = readPing();
    delay(100);
    servo_moteur.write(90);
    return distance;
    delay(100);
}

//////////////////////////////////////////////////////////////////LA
READPING//////////////////////////////////////////////////////////////////FONCTION

int readPing() {
    //cette fonction retourne la distance
    en cm
    delay(70);
    int cm = sonar.ping_cm();
    if (cm==0){
        cm=250;
    }
    return cm;
}

//////////////////////////////////////////////////////////////////LA
MOVESTOP//////////////////////////////////////////////////////////////////FONCTION

void moveStop() {
    // cette fonction permet le freinage
    du robot

    digitalWrite(M1_Frein, LOW);
    digitalWrite(M2_Frein, LOW);
    digitalWrite(M1_Direction, LOW);
    digitalWrite(M2_Direction, LOW);

}

```

```

//////////////////////////////////////////LA
MOVEFORWARD//////////////////////////////////////////
void moveForward(){
//cette fonction faire tourner les
deux moteurs dans la même sens en avant

    if(!goesForward){
//condition si le moteur est en
arret alors...

        goesForward=true;
        digitalWrite(M1_Direction, HIGH);
        digitalWrite(M1_Frein, LOW);
        digitalWrite(M2_Direction, HIGH);
        digitalWrite(M2_Frein, LOW);

    }
}

```

```

//////////////////////////////////////////LA
MOVEBACKWARD//////////////////////////////////////////
void moveBackward(){
//cette fonction faire tourner
les deux moteurs dans la même sens en arrière

    goesForward=false;
    digitalWrite(M1_Direction, LOW);
    digitalWrite(M1_Frein, HIGH);
    digitalWrite(M2_Direction, LOW);
    digitalWrite(M2_Frein, HIGH);

}

```

```

//////////////////////////////////////////LA
TURNRIGHT//////////////////////////////////////////
void turnRight(){
// cette fonction permet au moteur
de tourner à droite

```

```

digitalWrite(M1_Direction, HIGH);
digitalWrite(M1_Frein, LOW);
digitalWrite(M2_Direction, LOW);
digitalWrite(M2_Frein, LOW);
delay(400);
digitalWrite(M1_Direction, HIGH);
digitalWrite(M1_Frein, LOW);
digitalWrite(M2_Direction, HIGH);
digitalWrite(M2_Frein, LOW);

}

//////////////////////LA      FONCTION      TURN
LEFT//////////////////////

void turnLeft(){                                // cette fonction permet au moteur de
tourner à gauche

digitalWrite(M1_Direction, LOW);
digitalWrite(M1_Frein, LOW);
digitalWrite(M2_Direction, HIGH);
digitalWrite(M2_Frein, LOW);
delay(400);
digitalWrite(M1_Direction, HIGH);
digitalWrite(M1_Frein, LOW);
digitalWrite(M2_Direction, HIGH);
digitalWrite(M2_Frein, LOW);

}

```