REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
الجمهورية الجزائرية الديمقراطية الشعبية

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

HIGHER SCHOOL IN APPLIED SCIENCES
--T L E M C E N--

وزارة التعليم العالي والبحث العلمي

المدرسة العليا في العلوم التطبيقية
-تلمسان-

Mémoire de fin d'étude

Pour l'obtention du diplôme d'Ingénieur

Filière : Automatique
Spécialité : Automatique

Présenté par : TELDJOUNE Zakaria
GUENNICHE Ismail

Thème

Réalisation d'un Véhicule Auto Guidé (AGV)

Soutenu publiquement, le 06 / 07 / 2022, devant le jury composé de :

M BRAHAMI Mustapha Anwar	MCA	ESSA. Tlemcen	Président
M ABDELLAOUI Ghouti	MCB	ESSA. Tlemcen	Directeur de mémoire
Mme SEBBAGH Hafidha	MCA	ESSA. Tlemcen	Co- Directeur de mémoire
M ABDI Sidi Mohammed	MCB	ESSA. Tlemcen	Examinateur 1
M M'HAMEDI Mohammed	MAA	ESSA. Tlemcen	Examinateur 2

Année universitaire : 2021 /2022


بسم الله الرحمن الرحيم

Dédicaces

Je dédie cet ouvrage

À mes chers parents, pour tous leurs sacrifices, leur amour, leur tendresse, leur soutien et leurs prières tout au long de mes études,

À mes chers frères ZOHEIR, ZINNE EDDINE et MOHAMED AMINE pour leurs encouragements permanents, et leur soutien moral,

À ma grand-mère et toute ma famille pour leur soutien et leur encouragement,

À mon ami Younes et tous mes amis qui m'ont toujours encouragé, et à qui je souhaite plus de succès,

Sans oublier mon binôme pour son soutien moral, sa patience et sa compréhension tout au long de ce projet.

Que ce travail soit l'accomplissement de vos vœux tant allégués, et le fuit de votre soutien infaillible,

Merci d'être toujours là pour moi.

ISMAIL.GUENNICHE

Dédicaces

Je dédie ce travail

A mes parents qui m'a soutenu et encouragé ces années d'études.

Qu'ils trouvent ici le témoignage de ma profonde reconnaissance.

A mes sœurs, ma grand-mère et ceux qui ont partagé avec moi tous les moments d'émotion lors de la réalisation de ce travail. Ils m'ont chaleureusement supporté et encouragé tout au long de mon parcours.

A mon âme sœur, et ma famille qui m'a donné de l'amour et de la vivacité.

A tous mes amis qui m'ont toujours encouragé, et à qui je souhaite plus de succès.

TELDJOUNE.Zakaria

Remerciements

Nous rendons nos profondes gratitudes à Dieu tout puissant qui nous a aidés à réaliser ce modeste travail.

Nous exprimons nos profondes gratitudes à nos parents pour leurs encouragements, leurs soutiens et pour les sacrifices qu'ils ont enduré.

Nous remercions, notre encadreur et co-encadreur M ABDELLAOUI Ghouti et Mme SEBBAGH Hafidha pour les efforts qu'ils ont déployé, pour nous aider, conseiller, encourager et corriger.

Nous sommes sensibles à l'honneur que nous a fait monsieur Mustapha Anwar BRAHAMI, pour avoir accepté de présider notre mémoire et de nous honorer de sa présence au sein du jury.

Nous tenons egalement à adresser nos vifs remerciements à monsieur Sidi Mohamed ABDI, et à monsieur mohammed MHAMEDI, nous les remercions chaleureusement pour avoir accepté d'examiner le present memoire et pour ses observations et remarques pertinentes et constructives.

Nous remercions vivement M ADJIM RAMZ-EDDINE Abderrezak, l'ingénieur de FABLAB à l'Ecole supérieur en science appliquée de Tlemcen, pour nous avoir aidé dans la réalisation de notre travail. Sans oublier tous nos amis qui ont contribué de près ou de loin à la réalisation de ce travail, trouvent ici notre sincère reconnaissance.

Résumé

Notre projet consiste à réaliser un Robot mobile Automatique basé sur les robots auto guidés (AGV), avec certains ensemble d'améliorations et de performances tels que :

— La fonction suiveur de ligne.

— La fonction de détection et d'évitement d'obstacles.

— La fonction de mémorisation du chemin.

— La fonction de la recherche du plus court chemin (chemin le plus intelligent).

— La fonction de guidage manuel.

— La fonction de mesure de poids.

Le travail abordé dans ce mémoire comporte trois parties :

— L'objet de la première partie est une présentation des robots AGV

— La deuxième est une étude des algorithmes de la recherche du plus court chemin

— La troisième partie présente la réalisation pratique et résultat.

Abstract

Our project consists in realizing an Automatic mobile Robot based on the automated guided vehicle (AGV), with some set of improvements and performances such as :

— The line follower function.

— The obstacle detection and avoidance function.

— Path memorization function.

— The function of finding the shortest path (smartest path).

— Manual guidance function.

— The weight measurement function.

The work covered in this thesis has three parts :

— The subject of the first part is a presentation of AGV robots

— The second is a study of shortest path search algorithms

— The third part presents the practical realization and result.

ملخص

يتمثل مشروعنا في صناعة روبوت آلي متنقل يعتمد على الروبوتات ذاتية التوجه (AGV) ، مع بعض التحسينات والعروض مثل :

- وظيفة الكشف عن العوائق وتجنبها.
- وظيفة حفظ المسار.
- وظيفة إيجاد أقصر طريق (أذكى طريق).
- وظيفة التوجيه اليدوي.
- وظيفة قياس الوزن.
- وظيفة تابع الخط.

يحتوي العمل في هذه المذكرة من ثلاثة أجزاء :

- موضوع الجزء الأول هو عرض تقديمي لروبوتات AGV
- والثاني دراسة خوارزميات البحث عن أقصر طريق
- الجزء الثالث يعرض الانجاز العملي والنتيجة.

Table des matières

Table des figures

Liste des tableaux

Table des Acronymes

AGV	Automated Guided Vehicle
AMR	Autonomous Mobile Robot
API	Application Programming Interface
ARM	Advanced RISC Machines
CNC	Computer Numerical Control
DC	Direct Current
IA	Artificial intelligence
IDE	Integrated Development Environment
IoT	Internet of Things
ISIS	Integrated Software for Imagers and Spectrometers
LCD	Liquid Crystal Display
LiDAR	Light Detection and Ranging
MCC	moteur à courant continu
MIMO	Multiple Input, Multiple Output
MOSFET	Metal-Oxide Semiconductor Field-Effect Transistor
PID	Proportional-Integral-Derivative (controller)
PWM	Pulse Width Modulation
R.U.R	Rossum's Universal Robots
RISC	Reduced Instruction Set Computer
ROS	Robot Operating System
WLAN	Wireless Local Area Network

TABLE 1: Caption

Introduction générale

La fusion entre l'activité des industries actuelle et le monde de la robotique et de l'automatisation a donné naissance à des produits précis et de meilleures qualités qui donnent un sens à l'utilisation des nouvelles technologies. [18]

Le mot < robot > a été introduit pour la première fois en 1921 dans une pièce de théâtre écrite par le tchèque Karel Capek, et intitulée R.U.R (Rossum's Universal Robots).

En 1942, Issac Asimove formule, dans sa nouvelle < Runaround > les trois lois de la robotique censées protéger l'être humain :

-Un robot ne peut pas blesser un être humain ou, par inaction, ne permet pas à un être humain de se blesser.

-Un robot doit obéir aux ordres qui lui sont donnés par les êtres humains, sauf lorsque de tels ordres entreraient en conflit avec la première loi.

-Un robot doit protéger sa propre existence tant qu'une telle protection n'est pas en conflit avec la première ou la deuxième loi.

En 1953, le premier AGV au monde a été introduit au Royaume-Uni pour le transport, qui a été modifié à partir d'un tracteur et peut être guidé par un câble aérien.

En 1961, le premier robot industriel commence à travailler sur une ligne d'assemblage dans l'usine du General Motors, pour soulever et tordre des pièces chaudes de métal.

En 1979, Le premier bras articulé fait son apparition sur une chaine de production.

En 1986, Honda a créé le premier robot capable de marcher sur deux pieds comme

un humain.

En 1988, le premier robot de service fait son apparition à l'hôpital de Danbury (Etats-Unis).

En 1997, Le NASA envoie son premier robot explorateur sur Mars.

Le 3 juin 2014, La Commission Européenne et 180 entreprises de recherche ont lancé le programme SPARC, programme civil de recherche et d'innovation en robotique. [3]

Parmi tous les types de robots, le robot AGV à une place particulière parmi les autres. L'amélioration de la technologie permet à cette conception de se développer et devenir plus utile dans diverses applications. Le robot AGV est un robot mobile programmable intégrant un capteur qui peut automatiquement percevoir et se déplacer le long de la trajectoire planifié. Ce système se compose de plusieurs parties comme les installations de guidage, le central, le système de charge et le système de communication.

L'utilisation initiale et l'invention de l'AGV n'est pas claire exactement et a été mentionné dans différents articles et référence à plusieurs reprises.

Les AGV sont largement appliqués dans différents types d'industries, y compris la fabrication des usines et les dépôts pour la manutention des matériaux. Après des décennies de développement, ils sont largement utilisés en raison de leur efficacité, flexibilité, fiabilité, sécurité et de l'évolutivité de leur système dans diverses tâches et situations.

L'AGV fonctionne toute la journée en continu, ce que l'être humaine ne peut pas le faire. Par conséquent, l'efficacité de la manutention peut être stimulée en ayant la tâche de collaboration avec le nombre d'AGV. Dans ce cas, l'administrateur peut utiliser plus d'AGVs car le système est extensible.

L'AGV à la capacité d'éviter les collisions et les freinages d'urgence. Généralement l'état de fonctionnement est surveillé par le système de contrôle, afin que la fiabilité et la sécurité soient assurées. [18]

L'objectif principal de notre travail est d'implémenter un robot AGV suiveur de

ligne utilisant Une carte électronique Arduino.

Cette thèse est divisée en trois chapitres :

Le premier chapitre présente quelques informations générales sur les robots AGV, les différents types et leurs applications.

Le deuxième chapitre présente l'étude des algorithmes de la recherche du plus court chemin comme Algorithme de Bellman-Ford, Algorithme de Dijkstra, Algorithme de Kruskal, Algorithme de Prim et algorithme de colonie de fourmis.

Le troisième chapitre est consacré à la réalisation pratique et aux résultats obtenus.

Chapitre 1

Présentation des AGV

1.1 Introduction

Véhicule à guidage automatique - AGV (auto guided vehicle) est un véhicule de transport équipé de dispositifs de guidage robotiques électromagnétiques ou optiques qui peuvent se déplacer le long d'un chemin de guidage spécifié, avec une protection de sécurité et diverses fonctions de transmission. Généralement, il peut être contrôlé par un ordinateur qui contrôle le chemin de marche et son comportement, ou utilisé le système de suivi électromagnétique pour configurer le chemin de déplacement.

1.2 Histoire des AGVs

En 1953, Le premier système AGV à induction électromagnétique a été construit et présenté par les États-Unis au monde. Il s'agit d'un tracteur de remorquage modifié qui était utilisé pour tirer une remorque et suivre un câble aérien dans un entrepôt d'épicerie. Pour la ligne de production réelle, une étape clé de l'innovation théorique à la production réelle a été franchie. À la fin des années 50 et au début des années 60, les États-Unis ont appliqué pour la première fois l'AGV aux entrepôts automatisés. En 1973, Volvo en Suède a entrepris de développer des équipements d'assemblage non synchrones comme alternative à la chaîne d'assemblage de convoyeurs conventionnelle. Le résultat était 280

AGV d'assemblages contrôlés par ordinateur (voir la figure 1.1).

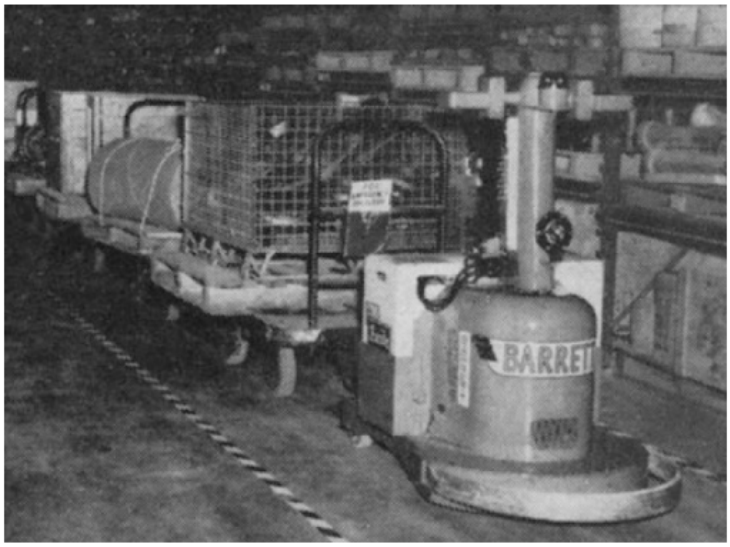

FIGURE 1.1: Les plus anciens AGV.

Au milieu des années 70, en raison de l'introduction d'un véhicule à chargement unitaire pouvant fournir des services multifonctionnels pour le domaine de la manutention, l'industrie des AGV a connu son premier développement majeur. En 1976, l'Institut de recherche chinoise sur les machines de levage et de transport de Pékin a développé le premier prototype d'AGV (voir la figure 1.2).

FIGURE 1.2: L'ancien AGV avec treillis métallique comme capteur de pare-chocs.

En 2010, la recherche technologique et le développement d'applications sur les AGVs

dans l'industrie est arrivés à un certain niveau de maturité et l'accent est principalement mis sur l'intelligence. Cela est principalement dû à la décroissance démographique mondial ces dernières années et à l'augmentation continue des coûts de main-d'œuvre des entreprises, qui a incité un grand nombre d'industries manufacturières haut de gamme nationales et étrangères à mettre en œuvre des méthodes de production sans personnel. [19]

2013-jusqu'à présent : Utilisation intensive de logiciels open source tels que ROS (par exemple, Robotnik), roscpp (ROS basé sur c++) et rospy (basé sur Phython). Implémentations d'IA open source pour AGV. L'utilisation des sources d'énergie améliorées/alternatives (Gaussin) et la fusion des technologies AMR et AGV (Otto 1500, Oppent). Recherche sur le système sans fil 5G pour la navigation AGV (ASTI, Espagne). L'émergence des manipulateurs mobiles (bras robots sur socles mobiles) (voir la figure 1.3).

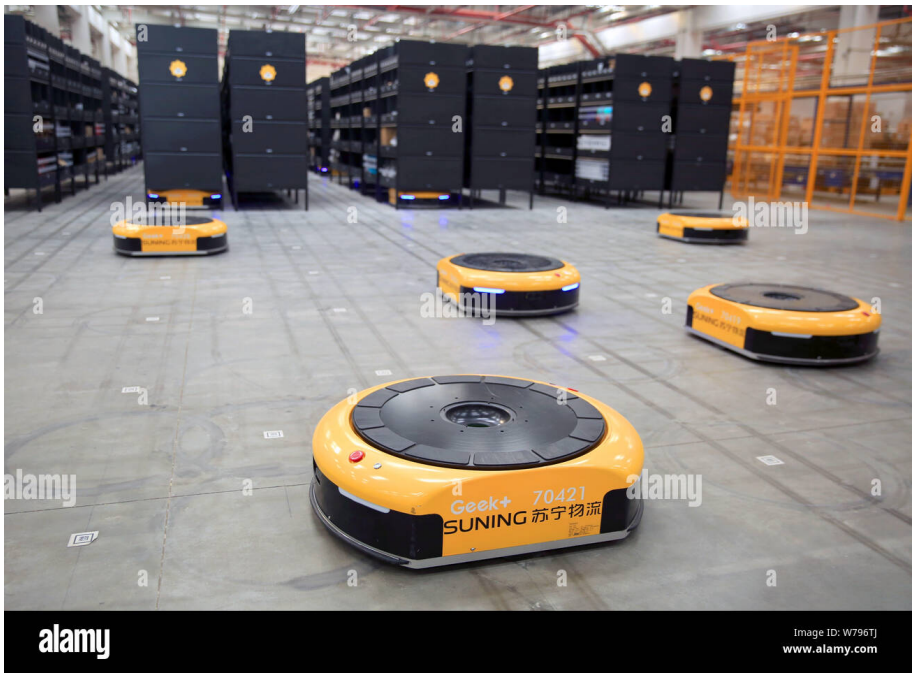

FIGURE 1.3: Les AGV chargés de la livraison et du tri des colis.

1.3 Les composantes d'un AGV

Un véhicule AGV est généralement composé d'un châssis de véhicule, un contrôleur intégré, des moteurs, des conducteurs, des systèmes de navigation et de collision, des capteurs d'évitement, un dispositif de communication et une batterie, certains d'entre eux ont un dispositif de transfert de charge. Diverses conceptions de véhicules AGV ont été réalisées par des instituts de recherche et des sociétés. Mais principalement tous les AGV sont constitués de pièces illustrées par la figure 1.4 :

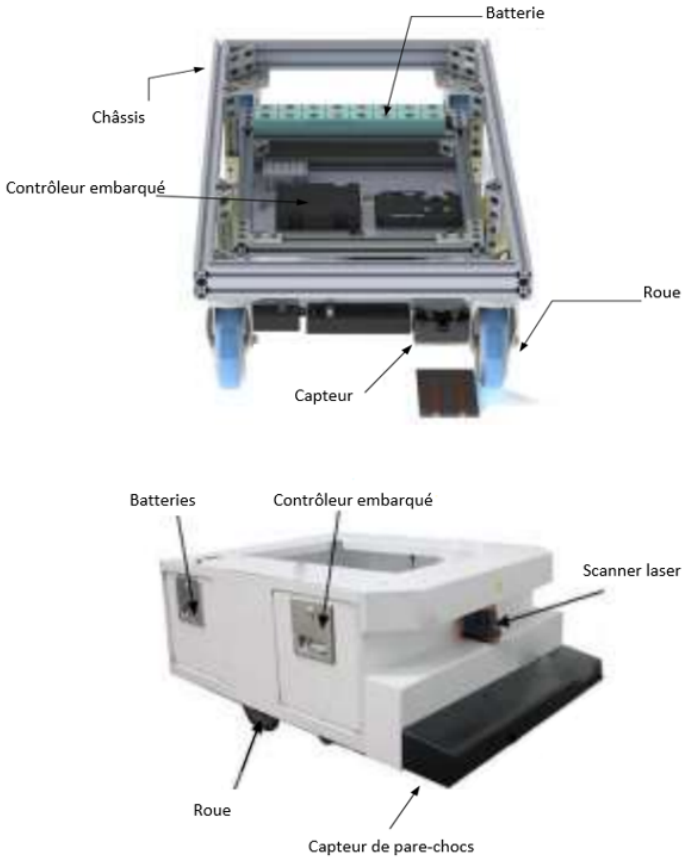

FIGURE 1.4: Les composantes d'un AGV.

1.3.1 Châssis

Le châssis est la structure rigide qui contient toutes les pièces mécaniques de tous les appareils AGV. Il est essentiel pour le maintien et la sécurité. Il doit être solide pour

supporter le poids qu'il supporte (voir la figure 1.5).

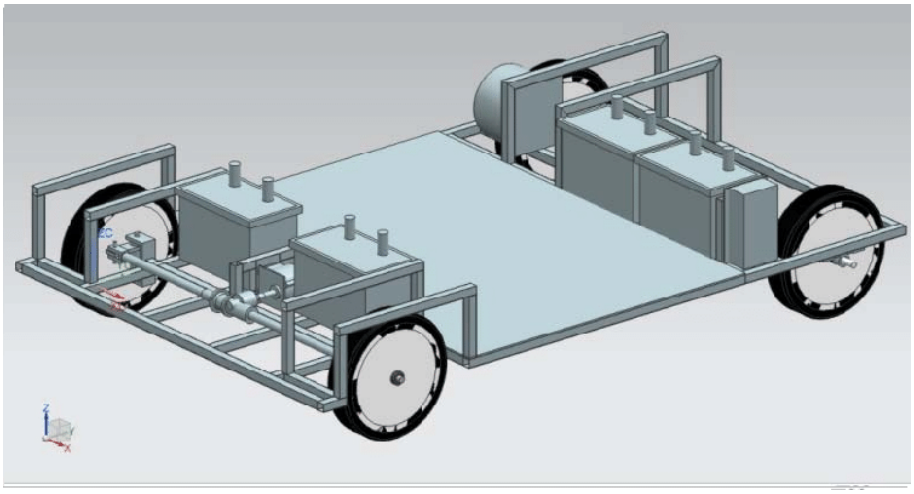

FIGURE 1.5: Chassis Prototype d'un AGV.

1.3.2 Contrôleur embarqué

Le niveau d'intelligence de l'AGV dépend relativement des performances du contrôleur embarqué. Le schéma AGV de base utilise un microcontrôleur ou un API (Application Programming Interface) pour gérer les tâches de capture de données à partir de capteurs et les communiquer au système de contrôle central et du mouvement de véhicule. Certains schémas avancés utilisent une puissante puce ARM (Advanced RISC Machines) alimentée par le ROS (Robot Operating System) pour gérer ces tâches (voir la figure 1.6 et 1.7).

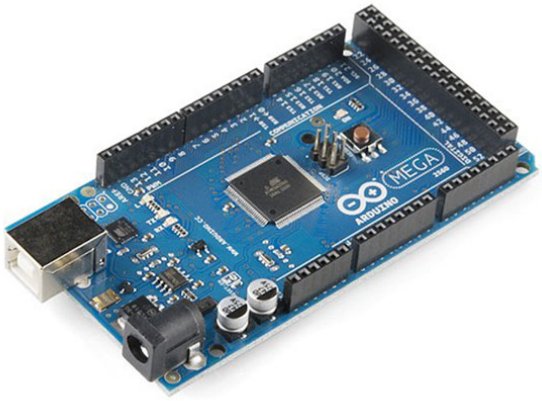

FIGURE 1.6: Microcontrolleur Arduino MEGA 2560

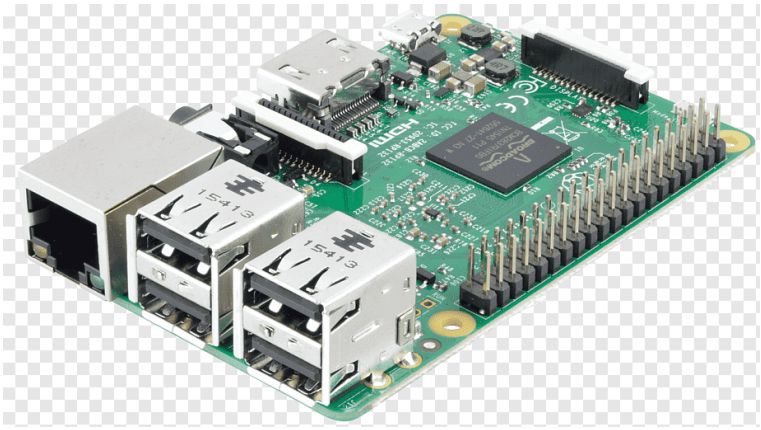

FIGURE 1.7: Carte raspberry pi 4 model B.

1.3.3 Capteurs

Les capteurs permettent de guider sans problème les robots et leur fournir les informations nécessaires pour exécuter correctement leur programme/tâche. Le type de capteur utilisé dépend des robots et de leur tâche. Il existe trois types de capteurs disponibles : un capteur de guidage pour guider l'AGV pour son déplacement le long d'une bande magnétique (bande de guidage), un capteur d'adresse pour obtenir les informations de localisation de l'AGV et un capteur d'arrêt pour détecter l'emplacement d'arrêt. Ce genre de capteurs exigent la présence des bandes magnétiques (bande de guidage) fixées au sol

et des tiges magnétiques pour leurs bon fonctionnement. [16]

FIGURE 1.8: Capteur de guidage.

FIGURE 1.9: Capteur d'adresse.

1.3.4 Dispositif de communication

Contrairement à d'autres équipements d'automatisation industrielle qui communiquent généralement sur un bus filaire, l'AGV nécessite une communication sans fil. Récemment, pour garantir la fiabilité, les produits utilisent généralement le WLAN (Wireless Local Area Network) avec une antenne MIMO (multiple-input and multiple-output)

doté d'un niveau élevé de conception de compatibilité électromagnétique. Actuellement, la technologie 5G est appliquée sur les AGV produits par Ericsson et China Mobile, offrant une fiabilité et une sécurité développée. Le dispositif de communication permet la connectivité avec le réseau local de l'atelier pour collecter des informations à partir des appareils/machines de l'atelier (par exemple, des bras de robot, des automates programmables, des moniteurs d'énergie, des appareils IoT). Les données collectées sont ensuite converties au format de données lisible par Smart AGV Manager via ce module. [28]

1.3.5 Batterie

Les AGV fonctionnent généralement à l'électricité avec une batterie qui doit être chargée. Les AGV extérieurs et plus grands qui peuvent transporter des charges de plusieurs tonnes fonctionnent parfois au diesel. Il existe différents types de batteries qui peuvent être utilisées dans les AGV, les plus courants sont :

— Batterie plomb-acide ouverte (Meilleur rapport qualité-prix, le plus courant).

— Batterie sans entretien (Le moins de maintenance).

— Batterie à rechargement rapide (Chargement rapide).

— Batterie nickel-cadmium (Chargement rapide, respect de l'environnement). [14]

1.3.6 Moteurs

Le moteur qui nous pouvons utilisé dans les AGVs est généralement monté avec un encodeur pour mesurer la distance de parcours. Le type de moteur utilisé dans AGV est nommé moteur à engrenages à courant continu, moteur à courant continu sans balai ou servomoteur. Ces moteurs sont sélectionnés en fonction de leurs impacts sur la flexibilité et la précision du mouvement de l'AGV. Le mode de conduite est divisé en trois types de transmission :

— Une seule roue motrice signifie qu'une roue motrice a pour fonction de marcher et de diriger et que deux roues motrices sont fixes.

— L'entraînement différentiel a deux roues motrices qui utilisent la différence de vitesse pour réaliser la rotation.

— L'entraînement omnidirectionnel est beaucoup plus flexible. Avec deux roues motrices et rotatives, le mouvement parallèle et la fonction d'entraînement différentiel sont disponibles (voir la figure 1.10).

FIGURE 1.10: Moteurs et entraînements des AGV.

1.4 Types d'AGVs

Les AGV se présentent sous de nombreuses formes avec différents attributs et domaines d'application. Ce qu'ils ont en commun a essayé d'être décrit par beaucoup, mais il n'y a pas de définition officielle. Selon Müller (1983), un AGV est un système de transport sans conducteur utilisé pour le mouvement horizontal de matériaux. Ganesharajah, Hall et Sriskandarajah (1998) définissent les AGV comme des véhicules sans conducteur alimentés par batterie, contrôlés par ordinateur de manière centralisée et adressables indépendamment, qui sont utilisés pour déplacer les travaux entre les postes de travail dans un atelier. Il existe de nombreuses autres définitions, mais ce qu'elles ont toutes en commun, c'est qu'un AGV est un véhicule qui déplace des matériaux horizontalement sans conducteur humain. Cela peut être dans un environnement de production, un entrepôt, un hôpital ou d'autres endroits où le matériel est transporté. Comme les AGV sont utilisés dans de

nombreux domaines d'application, ils peuvent avoir une apparence très différente et avoir des attributs différents. Ullrich (2015) est d'avis que la meilleure façon de catégoriser les AGV est d'examiner les charges qu'ils transportent (voir la figure 1.11). [14]

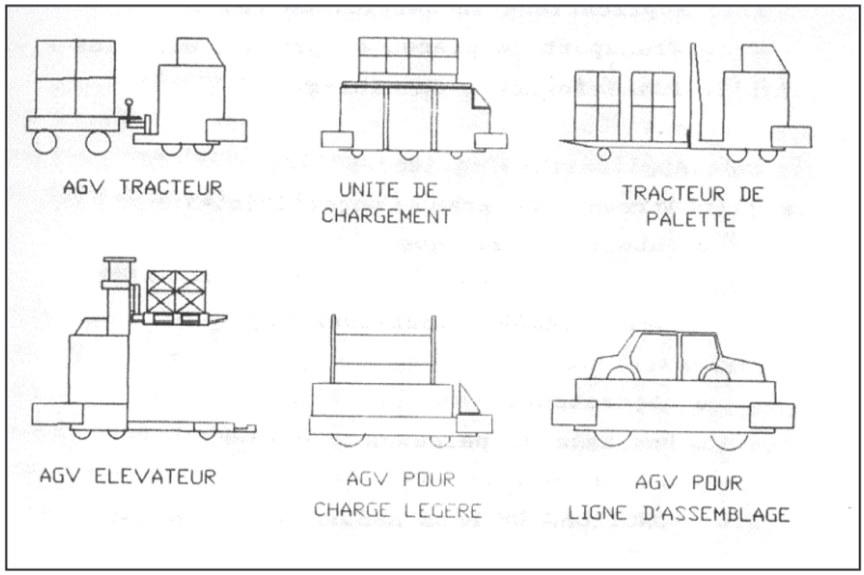

FIGURE 1.11: Différents types d'AGV. [6]

1.4.1 AGV tracteurs

Les AGV tracteurs remorquent les chariots non motorisés transportant des charges diverses, manuellement ou automatiquement par l'intermédiaire d'un attelage actionné. La version autonome est la forme standard des chariots tracteurs. Cependant, en cas d'intervention humaine fréquente, la version commandée manuellement peut également être nécessaire. Lorsqu'un opérateur prend le relai, le chariot passe immédiatement en mode manuel et désactive la guidance automatique. Les chariots AGV tracteurs sont également utilisés entre les différents entrepôts d'un même site industriel (voir la figure 1.12).

FIGURE 1.12: Véhicules à guidage laser types AGV séries tracteurs.

1.4.2 AGV pour charges unitaires

Parfois appelés "top carriers", les véhicules guidés automatisés (AGV) à charge unitaire remontent aux premières itérations des AGV et peuvent transporter différents types de charge. En termes plus simples, ces robots sont des systèmes de livraison de fret portables et autonomes capables de se déplacer dans un entrepôt ou des installations utilisant différentes technologies de navigation AGV. Ces robots fascinants sont principalement utilisés pour le transport industriel de marchandises et de matériaux lourds autour des entrepôts ou des installations de stockage (voir la figure 1.13). [20]

FIGURE 1.13: Unité de charge AGV par Muratec.

1.4.3 AGV tracteur de palettes

Le transpalette AGV combine une ingénierie de pointe et une technologie de navigation. Le tracteur à palettes AGV peut fonctionner en toute sécurité dans un environnement habité car il est doté de lasers, de capteurs et de pare-chocs électromécaniques pour fournir un certain niveau de système de sécurité. La navigation se fait via la navigation laser, donc aucun travail au sol n'est requis, ce qui facilite l'intégration de système d'entrepôt existant (voir la figure 1.14). [2]

FIGURE 1.14: AGV tracteur de palette électrique.

1.4.4 AGV élévateur à fourche

Le chariot à fourches est l'un des appareils de manutention les plus utilisés dans le monde de la logistique, en particulier dans les grands entrepôts de stockage. Le chariot à fourches est un véhicule de manutention incontournable. Il permet, entre autres, de soulever des palettes pour leur stockage en hauteur et de déplacer des conteneurs en toute simplicité (voir la figure 1.15). [5]

FIGURE 1.15: Chariot à fourches frontiales.

1.4.5 AGV pour charge légère

C'est un véhicule autonome compact pouvant porter des charges légères. Ce chariot autonome a été spécialement conçu pour les espaces confinés. Ces véhicules autoguidés sont créés pour répondre à des demandes logistiques ou de convoyage dans des milieux hospitaliers ou industriels (voir la figure 1.16).

FIGURE 1.16: AGV pour charge légère.

1.4.6 AGV pour ligne d'assemblage

L'idée est d'utiliser des AGV à la place des structures physiques particulières telles que la chaîne de plancher, le tapis mobile, les convoyeurs, les ponts roulants... toutes sont des lignes qui nécessitent d'importants travaux de génie civil pour l'installation. Les AGV transportent le produit initial (non fini) à produire ou à assembler le long de la ligne de production et les opérateurs ou robots vont pas à pas modifier, ajouter, transformer, ajoutant ainsi de la valeur du produit initial de base au produit fini. Ces AGVs augmentent la productivité, réduisent les coûts d'exploitation et améliorent l'ergonomie pour de nombreuses industries et sont idéaux pour les chaînes de montage. [29]

1.5 Domaines d'application d'AGV

L'AGV est utilisée dans des différents domaines. Citons quelques-uns :

1.5.1 L'industrie

Les technologies d'automatisation traditionnelles comprennent le Filoguidage, le guidage laser et le guidage automatique. Cependant, d'autres techniques tout aussi efficaces existent, telles que le géo-guidage et le guidage par ultrasons. Les AGV sont le plus souvent utilisés dans les applications industrielles pour déplacer automatiquement les marchandises dans les usines, les entrepôts ou les ateliers (voir la figure 1.17). [20]

FIGURE 1.17: AGV dans le domaine Industriel.

1.5.2 Le domaine militaire

Les AGVs sont de plus en plus utilisés dans le domaine militaire. En effet, la miniaturisation permet désormais de construire des robots discrets mais chargés en capteurs idéaux pour les missions d'espionnage ou de renseignement. Ces robots posent des problèmes éthiques et légaux. Cela a inspiré des associations à diriger des actions de sensibilisation à ces problèmes pour encadrer l'utilisation de ces robots militaires (voir la figure

1.18).

FIGURE 1.18: AGV de service militaire.

1.5.3 La santé

Dans les hôpitaux et cliniques, les AGVs (voir la figure 1.19) assurent un transport automatique des denrées et des produits (repas, linge, déchets, produits pharmaceutiques et stérilisés,etc.).

FIGURE 1.19: AGV médical.

1.5.4 Utilisation civile

En raison de leur polyvalence, de plus en plus de tâches sont confiées aux AGVs. C'est Par exemple Yukitaro, un robot chasse-neige japonais : Il est livré avec deux caméras et un GPS. Il monte et descend la rue de manière autonome utilisez la "bouche" de la pelle pour engloutir la neige, puis renvoyez la neige sous forme de glaçons (voir la figure 1.20).

FIGURE 1.20: Robot de Yukitaro.

1.5.5 L'usage domestique

La démocratisation des robots ces dernières années a conduit de nombreux robots à s'installer chez les particuliers pour effectuer des tâches pour leurs propriétaires. En effet, ils sont capables de nettoyer et de tondre la pelouse, de nettoyer les piscines... ce qui a poussé certains clients (riches) à acquérir ces domestiques modernes. Parmi les types des robots domestiques les plus populaires utilisés sont : le robot aspirateur (voir la figure 1.21), le robot tondeuse à gazon, le robot laveur de vitres et le robot assistant.

FIGURE 1.21: Robot aspirator.

1.6 Avantages d'AGV

Ainsi, les AGV, véhicules autonomes ou robots autonomes sont les nouveaux compagnons d'affaires dans un grand nombre de secteurs d'activité. Vont-ils remplacer l'humain et faire de nous ce que nous étions, bien sûr que non ! L'homme et sa capacité de réflexion et d'organisation seront toujours utiles. Cependant, les tâches les plus ingrates et sans haute valeur ajoutée seront de plus en plus confiées à des AGV. Par exemple, déplacer une palette d'un point A à un point B, ou amener des raisins fraîchement cueillis au bout d'un champ, sont les types de tâches qui peuvent être déléguées.

Bien sûr, la logique économique joue son rôle. AGV peut fonctionner 24 heures sur 24 et ne jamais manquer. Il réduit également les erreurs d'oubli ou d'inattention. Certes, l'investissement sous-jacent est important, mais une utilisation appropriée de cette nouvelle ressource peut rapidement s'avérer rentable.

Les AGV peuvent également limiter les dommages causés par les opérateurs. En effet, ils commettent parfois des erreurs qui peuvent endommager des infrastructures ou des véhicules. Grâce à l'utilisation de plusieurs capteurs, les AGV ne peuvent pas faire d'erreurs ou détecter de graves dommages avant qu'ils ne surviennent. Nous verrons comment cela fonctionne dans un prochain article.

Les AGV permettent également de sécuriser différentes zones de travail. En effet, grâce aux AGV, il est possible d'isoler des tâches dangereuses dans certaines zones interdites au public. Cela évite de placer les opérateurs dans des zones dangereuses. De plus, les tâches effectuées par les AGV, et non par les opérateurs, sont souvent des tâches de traitement lourdes qui altèrent le corps.

Les AGV ont également l'avantage de rendre certaines tâches plus productives. Sa rapidité, sa réactivité et surtout le fait qu'il soit connecté à une base de données d'entrepôts lui permet d'être particulièrement efficace. [26]

1.7 Les dangers des AGVs

L'utilisation de ces véhicules dans l'entreprise peut présenter un danger pour les personnes. En effet, la coexistence ou la proximité de ces véhicules robotisés et des travailleurs est source de risque dans des situations telles que :

Les AGV effectuent différents mouvements dans leurs vols, tels qu'avancer, reculer, tourner, monter et descendre, charger et décharger. L'utilisation de ces véhicules dans l'entreprise provoque principalement des phénomènes mécaniques dangereux tels que des collisions avec des personnes ou des écrasements entre des AGV et des éléments mobiles (autres véhicules motorisés) ou fixes.

Il est utilisé pour le transport de marchandises et est équipé de systèmes spécifiques pour le ramassage de marchandises, tels que des palettes, des bobines, des conteneurs, etc. En conséquence, les zones dangereuses des dispositifs de support de charge sont variées. La chute d'objets n'est pas à exclure.

Les raisons peuvent être variées, du comportement inapproprié des personnes à proximité, de la négligence, de la défaillance technique du système ou de l'état du sol (humide, rugueux, glissant, sablonneux, etc.). D'autres phénomènes dangereux sont liés aux travaux d'entretien des véhicules motorisés ou d'entretien des infrastructures situées sur les voies de circulation des AGV. [14]

1.8 Quelles sont les nouvelles tendances d'AGV ?

Ces dernières années, les capacités des systèmes AGV ont monté en flèche à mesure que la technologie des logiciels et des capteurs s'est améliorée. Les constructeurs proposent désormais des voitures plus précises, plus sûres et plus performantes. Plusieurs technologies pourraient avoir un impact considérable sur l'industrie des AGV dans les années à venir.

1.8.1 Le LiDAR

Le capteur LiDAR transmet des impulsions laser qui mesurent la distance entre l'objet et l'AGV dont il est équipé. Ces données combinées permettent la création d'une carte à 360 degrés de la zone d'exploitation, permettant à l'AGV de naviguer sans aucune infrastructure supplémentaire.

1.8.2 Les systèmes de vision par caméra

La caméra capture des informations en temps réel, ce qui aide l'AGV à "voir" les obstacles et l'infrastructure de l'installation. Lorsque nous couplons ces informations aux données fournies par les capteurs LiDAR, nous obtenons une image 3D dynamique et complète de la zone opératoire.

1.8.3 Les nouveaux logiciels

Le logiciel est l'épine dorsale du système AGV. Leurs capacités peuvent résoudre les défis uniques de chaque installation et ainsi mettre en œuvre des solutions spécifiques pour répondre au mieux à des applications spécifiques.

1.9 Conclusion

Les véhicules à guidage automatique (AGV) sont définis comme un groupe de véhicules collaboratifs sans pilote opérant sur un site de fabrication et coordonnés par un système de contrôle informatique centralisé ou distribué. Comme mentionné, leur utilisation principale est d'aider à automatiser le processus de création du produit.

Par conséquent, nous avons conclu que les AGV jouent un rôle important dans l'industrie de l'ingénierie et dans les usines de production de grande surface pour améliorer plus rapidement la technique de manutention, augmenter l'efficacité et minimiser les coûts et le temps de transport.

Chapitre 2

Étude des algorithmes de la recherche du plus court chemin

2.1 Introduction

Le problème du chemin le plus court est l'un des sujets les plus étudiés en informatique, en particulier en théorie des graphes. Un chemin le plus court optimal est celui qui respecte les critères de longueur minimale d'une source à une destination.

Il y a eu une augmentation des recherches sur les algorithmes du plus court chemin en raison des nombreuses et diverses applications du problème. Ces applications incluent les protocoles de routage réseau, la planification d'itinéraire, le contrôle du trafic, la recherche de chemin dans les réseaux sociaux, les jeux informatiques et les systèmes de transport, etc.

Il existe également divers paramètres dans lesquels un chemin le plus court peut être identifié. Par exemple, le graphe peut être statique, où les sommets et les arêtes ne changent pas dans le temps. En revanche, un graphe peut être dynamique, où les sommets et les arêtes peuvent être introduits, mis à jour ou supprimés au fil du temps. Le graphe contient des arêtes dirigées ou non dirigées. Les poids sur les arêtes peuvent être des poids négatifs ou non négatifs. Les valeurs peuvent être des nombres réels ou entiers, cela dépend du type de problème émis.

2.2 Définitions

Soit $G = (S, A)$ un graphe orienté valué tel que la fonction coût : $A \to R$ associée à chaque arc (s_i, s_j) de A un coût réel.

Le coût d'un chemin $p =< s_0, s_1, s_2, ..., s_k >$ est égal à la somme des coûts des arcs composant le chemin, c'est à dire,

$$cot(p) = \sum_{i=1}^{k} cot(s_{i-1}, s_i) \tag{2.1}$$

Le coût d'un chemin sera aussi appelé poid du chemin.

Le coût (ou poid) d'un plus court chemin entre deux sommets s_i et s_j est noté

$\delta(s_i, s_j)$ est défini par

$$\begin{cases} \delta(s_i, s_j) = +\infty & \text{s'il n'existe pas de chemin entre } s_i \text{ et } s_j \\ \delta(s_i, s_j) = \min\{cot(p)/p \text{ chemin de } s_i \text{ à } s_j\} & \text{sinon} \end{cases}$$

Considérons par exemple le graphe valué orienté dans la figure 2.1 :

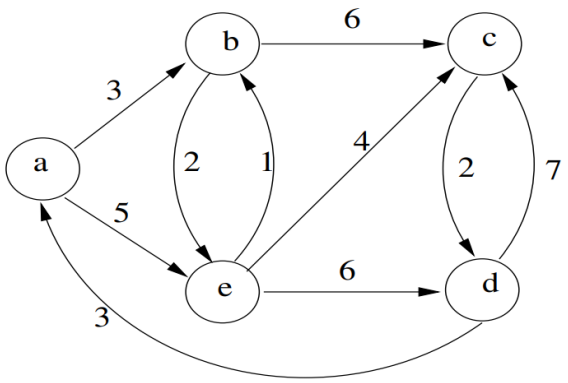

FIGURE 2.1: Exemple d'un graphe valué orienté

Nous avons

$$\delta(a, b) = 3, \delta(a, e) = 5, \delta(a, c) = 9 \text{ et } \delta(a, d) = 11$$

Condition d'existence du chemin le plus court : s'il existe un chemin entre les deux sommets u et v contiennent un circuit de coût négatif, alors $\delta(u, v) = -\infty$, et il n'y a pas de plus court chemin entre u et v. Le circuit négatif est appelé circuit de absorbant.

Par exemple, considérons le graphe orienté valué dans la figure 2.2 :

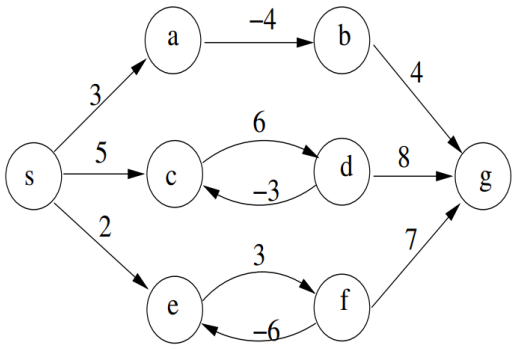

FIGURE 2.2: Exemple d'un graphe contient un circuit absorbant

Le chemin $< s, e, f, e, f, g >$ contient le circuit $< e, f, e >$ de coût négatif -3. Autrement dit, à chaque fois que le nous passons dans ce circuit, nous diminuons de 3 le coût total du chemin. Par conséquent, $\delta(s, g) = -\infty$ et il n'existe pas de plus court chemin entre s et g.

Définition du problème de plus court chemin à origine unique : étant donné un graphe orienté G = (S, A), une fonction coût : $A \to R$ et un origine $s_0 \in S$, nous voulons calculer pour chaque sommet $s_j \in S$ le coût $\delta(s_0, s_j)$ du plus court chemin de s_0 à s_j . Nous supposerons que le graphe G ne comporte pas de circuit absorbant.

Remarque : Nous avons

$$\delta(s_0, s_i) = \delta(s_0, \pi[s_i]) + cot(\pi[s_i], s_i)$$

Arborescence des plus courts chemins : Nous allons en fait calculer non seulement les coûts des plus courts chemins, mais aussi les sommets présents sur ces plus courts chemins. Cette arborescence est mémorisée dans un tableau π tel que

$$\pi[s_0] = Null$$

,

$$\pi[s_j] = s_i \text{ si } s_i \to s_j \text{ est un arc de l'arborescence ,}$$

Pour connaitre le plus court chemin entre deux sommets s_0 et s_k donné, il faudra alors remonter de s_k jusqu'à s_0 en utilisant π.

Considérons par exemple le graphe valué orienté dans la figure 2.3 :

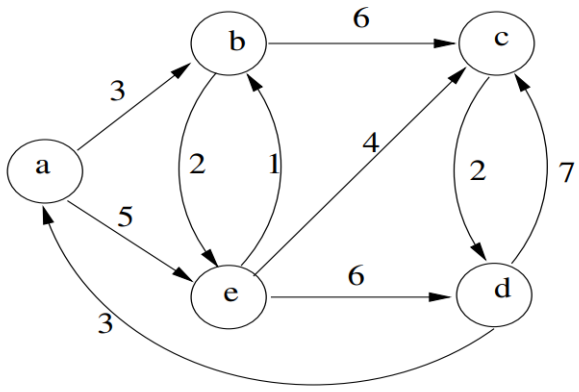

FIGURE 2.3: Exemple d'un graphe avec plusieurs arborescences des plus courts chemins

Le graphe de la figure 2.4 possède des arborescences des plus courts chemins dont l'origine est a, par exemple

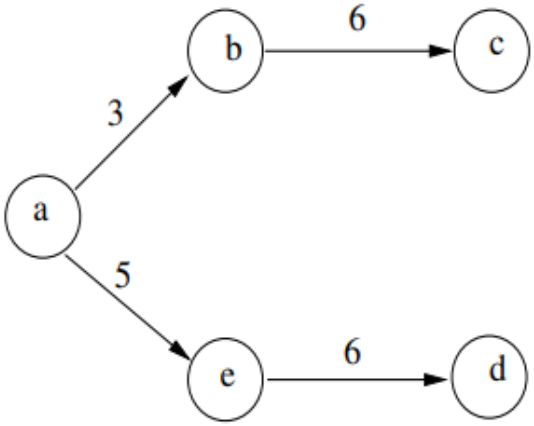

FIGURE 2.4: Exemple d'un graphe avec des arborescences simples des plus courts chemins

L'arborescence de la figure 2.4 est représentée par le tableau π tel que $\pi[a] =$

$Null, \pi[b] = a, \pi[c] = b, \pi[d] = e$ et $\pi[e] = a$.

Nous allons maintenant étudier 4 algorithmes qui permettent de résoudre des problèmes de recherche de plus courts chemins à origine unique : l'algorithme de Dijkstra, l'algorithme de Bellman-Ford, l'algorithme de colonie de fourmis et l'algorithme de Floyd-Warshall.

Les deux premiers algorithmes Dijkstra et Bellman-Ford procèdent de la même manière, selon une stratégie dite "gloutonne". L'idée est d'associer à chaque sommet $s_i \in S$ une valeur $d[s_i]$ qui représente une borne maximale du coût du plus court chemin entre s_0 et s_i (c'est-à-dire $\delta(s_0, s_i)$). Ainsi, au départ,

$$\begin{cases} d[s_0] = \delta(s_0, s_0) = 0 \\ d[s_i] = +\infty \geq \delta(s_0, s_i) \text{ pour tout sommet } s_i \neq s_0 \end{cases}$$

L'algorithme diminue alors progressivement les valeurs $d[s_i]$ associées aux différents sommets, jusqu'à ce que nous ne puissions plus les diminuer, autrement dit, jusqu'à ce que $d[s_i] = \delta(s_0, s_i)$.

Pour diminuer les valeurs de d, nous allons examiner itérativement chaque arc $s_i \to s_j$ du graphe, et regarder si nous ne pouvons pas nous améliorer la valeur de $d[s_j]$ en passant par s_i. Cette opération de diminution est appelée «relâchement de l'arc $si \to s_j$», et s'écrit :

Algorithm 1 Relâchement de l'arc $si \to s_j$

proc relacher (s_i, s_j)

si $d[s_j] > d[s_i] + cot(s_i, s_j)$ alors

/* il vaut mieux passer par s_i pour aller à s_j */

$d[s_j] \leftarrow d[s_i] + cot(s_i, s_j)$

$\pi[s_j] \leftarrow s_i$

fin si

fin relacher. [21]

2.3 Les différents algorithmes de plus courts chemins

2.3.1 Algorithme de Bellman-Ford

L'algorithme de Bellman-Ford résout le problème du plus court chemin à origine unique dans le cas le plus général où les poids des arcs peuvent avoir des valeurs négatives. Étant donné un graphe orienté pondéré G = (S, A), d'un *cot* et une origine s, l'algorithme retourne une valeur booléenne indiquant s'il existe un circuit de poids négatif accessible depuis s. S'il n'en existe pas, l'algorithme donne les plus courts chemins et leurs poids.

2.3.1.1 Algorithme

Algorithm 2 Bellman-ford

fonc Bellman-Ford(G = (S, A), coût : S→ $\Re$, $s_0 \in S$)

/* S est l'ensemble des sommets.*/

/* A est un ensemble de couples de sommets $(s_i, s_j) \in S^2$.*/

retourne une arborescence des plus courts chemins d'origine s_0

pour chaque sommet $s_i \in S$ faire

$$d[s_i] \leftarrow +\infty$$

/* $d[s_i]$ représente une borne maximale du coût du plus court chemin entre s_0 et s_i.*/

$$\pi[s_i] \leftarrow Null$$

/*L'arborescence associée à un parcours de graphe sera mémorisée dans un tableau π tel que $\pi[s_j] = s_i$ si s_j a été découvert à partir de s_i, et $\pi[s_0] = Null$ si s_0 est la racine, ou s'il n'existe pas de chemin de la racine vers s_0.*/

fin pour

$$d[s_0] \leftarrow 0$$

pour k variant de 1 a $|S|-1$ faire

pour chaque arc $(s_i, s_j) \in$ A faire relacher(s_i, s_j) fin pour

fin pour

pour chaque arc $(s_i, s_j) \in$ A faire

si $d[s_j] > d[s_i] + cot(s_i, s_j)$ alors afficher("circuit absorbant") finsi

fin pour

retourner (π)

fin Bellman−ford [21]

2.3.1.2 Exemple de l'algorithme de Bellman-ford

Considérons l'exemple du graphe dans la figure 2.5 pour aplliquer l'algorithme de Bellman-ford :

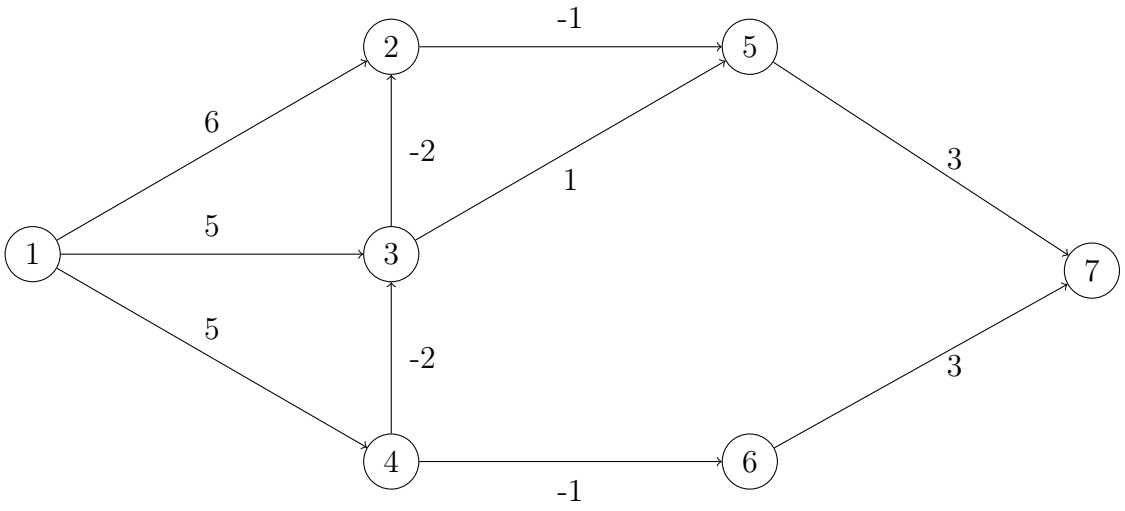

FIGURE 2.5: Exemple d'un graphique aléatoire

Ordre de relâchement des arcs :

$$[(1,2),(1,3),(1,4),(3,2),(2,5),(3,5),(4,3),(4,6),(6,7),(5,7)]$$

Etape 1 :

Nous initialisons les distances de la source à tous les sommets en tant qu'infini et la distance à la source elle-même en tant que 0. (voir la figure 2.6)

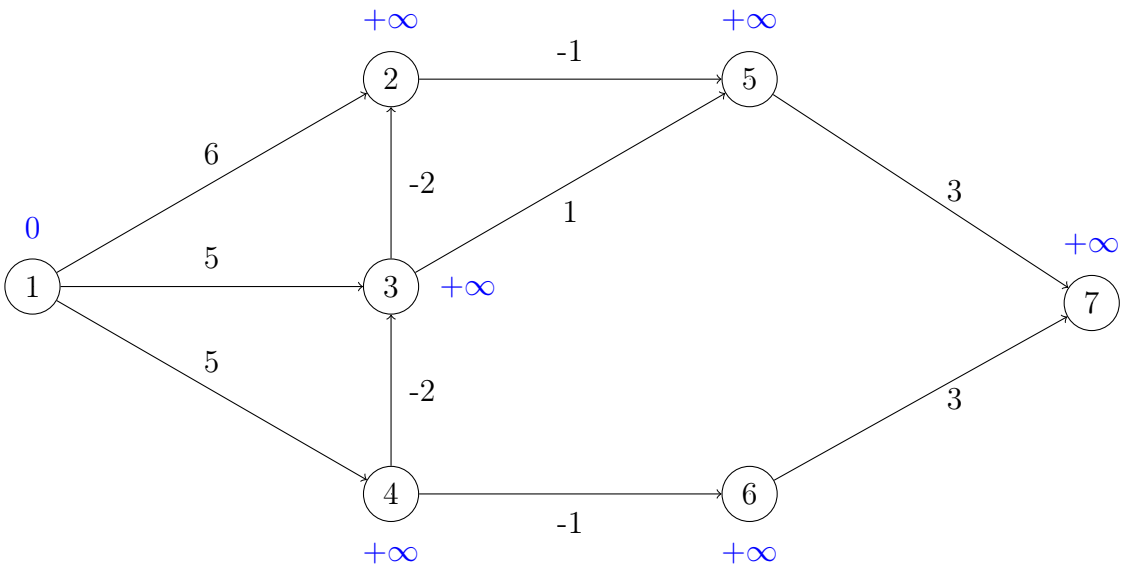

FIGURE 2.6: Graphe après l'initialisation des distances

Le tableau 2.1 indique la distance de chemin des sommets à partir de chaque arc en gardant la valeur de plus court chemin trouvé à chaque sommet :

Arête	relaxation (mise à jour)
$(1,2)$	Distance $[2] = 6$
$(1,3)$	Distance $[3] = 5$
$(1,4)$	Distance $[4] = 5$
$(3,2)$	Distance $[2] = 3$
$(2,5)$	Distance $[5] = 2$
$(3,5)$	Rien à faire $(2 < 4)$
$(4,3)$	Distance $[3] = 3$
$(4,6)$	Distance $[6] = 4$
$(6,7)$	Distance $[7] = 7$
$(5,7)$	Distance $[7] = 5$

TABLE 2.1: La distance des sommets de chaque arc

Etape 2

Considérons l'exemple du graphe dans la figure 2.7 pour aplliquer l'algorithme de Bellman-ford :

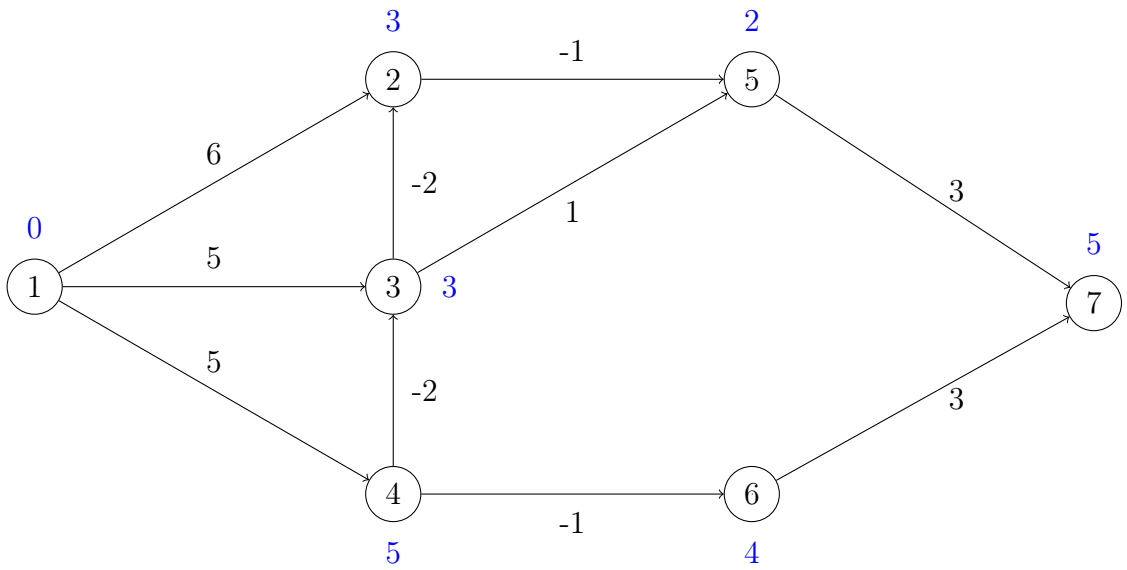

FIGURE 2.7: Graphe de la deuxième itération

Le tableau 2.2 indique la distance de chemin des sommets à partir de chaque arc dans la deuxième itéraion en gardant la valeur de plus court chemin trouvé à chaque sommet :

Arête	relaxation (mise à jour)
$(1, 2)$	Rien à faire
$(1, 3)$	Rien à faire
$(1, 4)$	Rien à faire
$(3, 2)$	Distance $[2] = 1$
$(2, 5)$	Distance[5]=0
$(3, 5)$	Rien à faire
$(4, 3)$	Rien à faire
$(4, 6)$	Rien à faire
$(6, 7)$	Rien à faire
$(5, 7)$	Distance $[7] = 3$

TABLE 2.2: La distance des sommets de chaque arc d'itération deuxième

Etape 3

Dans la figure 2.8 nous avons les valeurs du plus courts chemins :

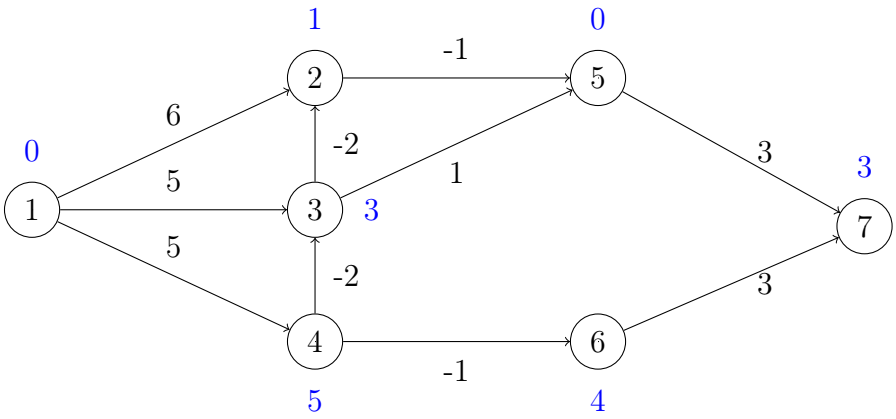

FIGURE 2.8: Graphe de la troisième itération

Le tableau 2.3 indique qu'il n'y a aucun changement dans les coûts des arcs dans la troisième itéraion en gardant les valeurs de plus court chemin précédant :

Arête	relaxation (mise à jour)
$(1, 2)$	Rien à faire
$(1, 3)$	Rien à faire
$(1, 4)$	Rien à faire
$(3, 2)$	Rien à faire
$(2, 5)$	Rien à faire
$(3, 5)$	Rien à faire
$(4, 3)$	Rien à faire
$(4, 6)$	Rien à faire
$(6, 7)$	Rien à faire
$(5, 7)$	Rien à faire

TABLE 2.3: Tableau d'itération troisième

Il n'ya pas un circle absorbant négatif donc nous avons garantit les distances les plus courtes de chaque sommet.

L'algorithme s'arrête.

2.3.1.3 Avantages

Bellman-Ford apporte de nombreux avantages aux utilisateurs. Premièrement, nous pouvons minimiser nos coûts lors de la construction du réseau. En effet, l'algorithme Bellman-ford trouve les poids des chemins les plus courts soumis d'un nœud source donné à un autre nœud. Nous n'avons donc pas besoin de construire de nombreux routeurs pour construire un chemin d'un nœud à un autre. En plus de cela, l'algorithme Bellman Ford maximise les performances de notre système. L'algorithme trouvera le poids de chemin minimum. Les poids des chemins correspondent aux délais de propagation du système. Dans Bellman-Ford, toutes les connexions suivront le chemin le moins lourd. Cette action

augmente la latence globale du système. Cependant, rien ne garantit que le choix du chemin le plus court maximisera les performances du système. [13]

2.3.1.4 Inconvénients

A partir du nœud de départ, cet algorithme se réitère jusqu'à ce qu'il ne trouve plus de nœud dont la distance (ou étiquette) peut être améliorée. Dans de nombreux cas, il examine plusieurs fois un nœud visité. Ceci pose aussi un problème de performance non négligeable. De plus, l'inconvénient d'un tel algorithme est le fait de trouver un cycle dont la somme des distances (ou étiquettes) est négative. Puisque l'algorithme compare la distance de chaque nœud, cette dernière se trouve être améliorée continuellement dans un tel cycle. [13]

2.3.2 Algorithme de Dijkstra

L'algorithme de Dijkstra est un excellent moyen classique de calculer le chemin le plus court dans un graphe à partir d'une seule origine. Pour corriger cet algorithme, tous les poids doivent être positifs ou nuls. De plus, cela est illustré à l'aide d'une boucle statique non triviale. L'algorithme de Dijkstra nécessite l'implémentation d'une file d'attente prioritaire (qui n'est pas une simple structure de données). Nous voyons alors l'émergence de toutes les structures de données dans le calcul de la complexité de l'algorithme. [17]

2.3.2.1 Algorithme

Algorithm 3 Dijkstra

fonc Dijkstra(G = (S, A), coût : S $\rightarrow \Re^+$, $s_0 \in S$)

retourne une arborescence des plus courts chemins d'origine s_0

pour chaque sommet $s_i \in S$ faire

$$d[s_i] \leftarrow +\infty$$

$$\pi[s_i] \leftarrow Null$$

fin pour

$$d[s_0] \leftarrow 0$$

$$E \leftarrow \emptyset$$

/* L'ensemble E contient chaque sommet s_i pour lequel nous connaissons un plus court chemin depuis s_0. */

$$F \leftarrow S$$

/* L'ensemble F contient tous les autres sommets. Nous avons $E \cup F = S$ */

tant que $F \neq \emptyset$ faire

soit s_i le sommet de F tel que $d[s_i]$ soit minimal

$$/^* d[s_i] = \delta(s_0, s_i)^* /$$

/* $\delta(s_i, s_j)$ est le coût (ou poid) d'un plus court chemin entre deux sommets s_i et s_j */

$$F \leftarrow F - \{s_i\}$$

$$E \leftarrow E \cup \{s_i\}$$

pour tout sommet $s_j \in \text{succ}(s_i)$ faire : relacher$(s_i, s_j\ )$

$$/^* succ(s_i) = \{s_j / (s_i, s_j) \in A\}^* /$$

fin tant

retourner (π)

fin Dijkstra. [21]

2.3.2.2 Exemple de l'algorithme de Dijkstra

Nous allons avoir ici un exemple de l'algorithme de Dijkstra appliqué sur le graphe suivante pour trouver le chemin le plus court. (voir la figure 2.9)

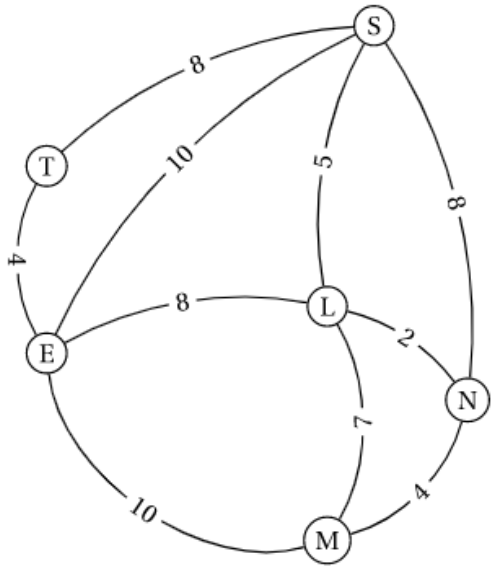

FIGURE 2.9: Exemple d'un graphe

Initialisation

Nous allons commencer d'abord par construire un tableau ayant comme colonnes chacun des sommets du graphe. Nous ajoutons une colonne à gauche qui listera les sommets choisis à chaque étape (cette colonne est facultative mais facilitera la compréhension de l'algorithme).

Puisque nous partons du sommet M, nous nous inscrivons, sur la première ligne intitulée « Départ », 0_M dans la colonne M et ∞ dans les autres colonnes. (voir le tableau 2.4)

Cela signifie qu'à ce stade, nous pouvons atteindre M en 0 minute et que nous n'avons atteint aucun autre sommet puisque nous n'avons pas encore emprunté le chemin.

	E	L	M	N	S	T
Départ	∞	∞	0_M	∞	∞	∞

TABLE 2.4: Départ

Étape 1 :

Nous sélectionnons le plus petit résultat de la dernière ligne. Ici, c'est « 0_M » qui correspond au chemin menant au sommet M en 0 minute.

— Nous mettons en évidence cette sélection (nous l'écrirons en rouge mais il est également possible de la souligner, de l'entourer, etc.).

— Nous écrivons le sommet sélectionné et la durée correspondante dans la première colonne (ici nous écrivons M(0)).

— Nous désactivons les cases situées en dessous de notre sélection en les grisant par exemple. En effet, nous avons trouvé le chemin le plus court menant à M ; il sera inutile d'en chercher d'autres. (voir le tableau 2.5)

	E	L	M	N	S	T
Départ	∞	∞	0_M	∞	∞	∞
M(0)						

TABLE 2.5: Tableau d'étape 1

À partir de sommet M, nous voyons sur le graphique que nous pouvons atteindre E, L et N respectivement en 10, 7 et 4 minutes. Ces durées sont les durées les plus courtes ; elles sont inférieures aux durées inscrites sur la ligne précédente qui étaient « ∞ ».

Nous écrivons donc 10_M, 7_M et 4_M dans les colonnes E, L et N. Le M situé en indice signifie que l'on vient du sommet M.

Enfin nous complètons la ligne en recopiant dans les cellules vides les valeurs de la ligne précédente. (voir le tableau 2.6)

	E	L	M	N	S	T
Départ	∞	∞	0_M	∞	∞	∞
M(0)	10_M	7_M		4_M	∞	∞

TABLE 2.6: Les sommets que nous pouvons atteindre à partir M

Étape 2 :

Nous sélectionnons le plus petit résultat de la dernière ligne. Ici, c'est «4_M » qui correspond au chemin menant au sommet N en 4 minutes.

— Nous mettons cette sélection en évidence.

— Nous traçons le sommet retenu et la durée correspondante dans la première colonne : N (4).

— Nous désactivons les cases situées en dessous de notre sélection. Nous avons trouvé le trajet le plus court menant à N ; il dure 4 minutes. (voir tableau 2.7)

	E	L	M	N	S	T
Départ	∞	∞	0_M	∞	∞	∞
M(0)	10_M	7_M		4_M	∞	∞
N(4)						

TABLE 2.7: Le chemin le plus court vers N

— À partir de N, nous pouvons rejoindre L et S (nous ne nous préoccupons plus de M qui a été « désactivé »).

— Si l'on rejoint L : Nous mettons 2 minutes pour aller de N à L et 4 minutes pour aller de M à N (ces 4 minutes sont inscrites dans la première colonne) soit au total 6 minutes. Ce trajet est plus rapide que le précédent qui durait 7 minutes. Nous indiquons donc 6_N dans la colonne L. Le N situé en indice signifie que le nous venons du sommet N.

— Si l'on rejoint S : Nous mettons 8 minutes pour aller de N à S et 4 minutes pour aller de M à N soit au total 12 minutes. Ce trajet est plus rapide que le précédent qui était ∞. Nous indiquons donc 12_N dans la colonne S. Puis nous complètons la ligne en recopiant dans les cellules vides les valeurs de la ligne précédente. (voir le tableau 2.8)

	E	L	M	N	S	T
Départ	∞	∞	0_M	∞	∞	∞
M(0)	10_M	7_M		4_M	∞	∞
N(4)	10_M	6_N			12_N	∞

TABLE 2.8: Les chemins les plus courts qu'on peut atteindre à partir M et N

Étape 3 :

— Nous sélectionnons le plus petit résultat de la dernière ligne. Ici, c'est « 6_N » qui correspond au chemin menant au sommet L en 6 minutes.

— Nous mettons en évidence cette sélection.

— Nous écrivons le sommet retenu et la durée correspondante dans la première colonne : L (6).

— Nous désactivons les cases situées en dessous de notre sélection. Nous avons trouvé le trajet le plus court menant à L ; il dure 6 minutes. (voir le tableau 2.9)

	E	L	M	N	S	T
Départ	∞	∞	0_M	∞	∞	∞
M(0)	10_M	7_M		4_M	∞	∞
N(4)	10_M	6_N			12_N	∞
L(6)						

TABLE 2.9: Le chemin le plus court vers L

— À partir de L, nous pouvons rejoindre E et S (nous ne nous préoccupons plus de M ni de N qui ont été « désactivés »).

— Si nous rejoins E : nous prenons 8 minutes pour aller de L à E et 6 minutes pour aller de M à L soit, au total, 14 minutes. Ce trajet n'est pas plus rapide que le précédent qui durait 10 minutes.

— Nous copions donc simplement le contenu précédent 10_M dans la colonne E.

— Si nous rejoins S : nous prenons 5 minutes pour aller de L à S et 6 minutes pour aller de M à L soit au total 11 minutes. Ce trajet est plus rapide que le précédent qui durait 12 minutes. Nous indiquons donc 11_L dans la colonne S. (voir le tableau 2.10)

IMPORTANT !

Nous nous inscrivons la durée d'un trajet dans le tableau uniquement si elle est inférieure à la durée indiquant sur la ligne précédente. Sinon, la valeur précédente est copiée. Puis nous complètons la ligne en recopiant dans les cellules vides les valeurs de la ligne précédente.

	E	L	M	N	S	T
Départ	∞	∞	0_M	∞	∞	∞
M(0)	10_M	7_M		4_M	∞	∞
N(4)	10_M	6_N			12_N	∞
L(6)	10_M				11_L	∞

TABLE 2.10: Les chemins les plus courts qu'on peut atteindre à partir M , N et L

Étape 4 :

— Nous sélectionnons le plus petit résultat. C'est «10_M » qui correspond au chemin menant au sommet E en 10 minutes.

— Nous mettons en évidence cette sélection.

— Nous écrivons le sommet retenu et la durée correspondante dans la première colonne : E (10).

— Nous désactivons les cases situées en dessous de notre sélection. Nous avons trouvé le trajet le plus court menant à E ; il dure 10 minutes. (voir le tableau 2.11)

	E	L	M	N	S	T
Départ	∞	∞	0_M	∞	∞	∞
M(0)	10_M	7_M		4_M	∞	∞
N(4)	10_M	6_N			12_N	∞
L(6)	10_M				11_L	∞
E(10)						

TABLE 2.11: Le chemin le plus court vers E

À partir de E, nous pouvons rejoindre S et T (nous ne nous préoccupons plus des autres sommets qui ont été « désactivés »).

— Si nous rejoinions S : nous mettons 10 minutes pour aller de E à S et 10 minutes pour aller de M à E (ces 10 minutes sont inscrites dans la première colonne) soit au total 20 minutes. Ce trajet n'est pas plus rapide que le précédent qui durait 11 minutes. Donc, nous copions simplement le contenu précédent 11_L dans la colonne S.

— Si nous rejoinions T : nous mettons 4 minutes pour aller de E à T et 10 minutes pour aller de M à E soit au total 14 minutes. Ce trajet est plus rapide que le précédent qui était ∞. Nous indiquons donc 14_E dans la colonne T. (voir le tableau 2.12)

	E	L	M	N	S	T
Départ	∞	∞	0_M	∞	∞	∞
M(0)	10_M	7_M		4_M	∞	∞
N(4)	10_M	6_N			12_N	∞
L(6)	10_M				11_L	∞
E(10)					11_L	14_E

TABLE 2.12: Les chemins les plus courts que nous puissions atteindre à partir de M , N, L et E

Étape 5 :

Nous sélectionnons le plus petit résultat. C'est « 11_L » qui correspond au chemin menant au sommet S en 11 minutes.

Nous avons trouvé le trajet le plus court menant à S, il dure 11 minutes. Comme S est le sommet cible, donc il est inutile d'aller plus loin et le tableau est terminé (voir le tableau 2.13).

	E	L	M	N	S	T
Départ	∞	∞	0_M	∞	∞	∞
M(0)	10_M	7_M		4_M	∞	∞
N(4)	10_M	6_N			12_N	∞
L(6)	10_M				11_L	∞
E(10)					11_L	14_E

TABLE 2.13: Tableau final

Il reste toutefois à reconstituer le trajet qui correspond à cette durée de 11 minutes. En pratique, il est plus facile de trouver le trajet en sens inverse en « remontant » dans le tableau de la façon suivante :

— Nous partons de notre point d'arrivée S.

— Nous recherchons la cellule marquée en rouge de la colonne S ; elle contient 11_{L}. Nous notons la lettre écrite en indice L.

— Nous recherchons la cellule marquée en rouge de la colonne L ; elle contient 6_{N}. Nous notons la lettre écrite en indice N.

— Nous recherchons la cellule marquée en rouge de la colonne N ; elle contient 4_{M}. Nous notons la lettre écrite en indice M.

Nous avons arrivé à notre point de départ M après être passé par N et L et S (liste obtenue en listant les sommets en ordre inverse).

Le trajet optimal est donc **M - N - L - S**.

Enfin, nous pouvons vérifier sur le graphe 2.9 que ce trajet est correct et dure 11 minutes.

2.3.2.3 Avantages

Le principal avantage de l'algorithme de Dijkstra est sa complexité considérablement faible, qui est presque linéaire. Avec cet algorithme, nous pouvons nous arrêter dès que le point d'arrivé devient la case à traiter puisque le chemin que nous avons construit jusque-là est un chemin minimal. [13]

2.3.2.4 Inconvénients

Cet algorithme est de type glouton. C'est pourquoi dans un certain nombre de cas ou les cases à traiter ont un même niveau de trafic l'algorithme n'est pas très performant car il les parcourt toutes. C'est pareil si nous prenons un autre critère de tri comme la distance entre la case à traiter et la case d'arrivée. Puisque chaque nœud traité est marqué, nous ne pouvons pas revenir en arrière. De ce fait, l'algorithme de Dijkstra est inadapté quand nous avons des poids négatifs. [13]

2.3.3 Algorithme de colonie de fourmis

Les fourmis se caractérisent par un comportement collectif où la priorité de chaque fourmi est le bien-être de la communauté. Chaque fourmi de la colonie fonctionne de manière indépendante et n'est pas supervisée d'une manière ou d'une autre. Ce concept est appelé hétérarchie (par opposition à hiérarchie) car chaque fourmi contribue au bon fonctionnement de communauté et en retour est aidée par la communauté dans son développement. Ainsi, la colonie se surveille par des mécanismes relativement faciles à étudier.

En observant une colonie de fourmis en quête de nourriture près du nid, nous réalisons que cela résout des problèmes comme celui de trouver le chemin le plus court. Les fourmis résolvent des problèmes complexes avec des mécanismes faciles à modéliser. Par conséquent, leur comportement est facile à simuler par des algorithmes.

Pour chaque fourmi, le trajet entre un nœud i et un nœud j dépend de :

— La liste des nœuds déjà visitées, qui définit les mouvements possibles à chaque pas, quand la fourmi k est sur les nœuds i : J_i^k ;

— La visibilité qui est l'inverse de la distance entre deux nœuds

$$\eta_{ij} = \frac{1}{d_{ij}}$$

Avec d_{ij} est la distance entre i et j.

— L'intensité de la piste τ_{ij} représente la quantité de la phéromone déposée sur l'arrêt reliant les deux nœuds.

— La loi du déplacement est la suivante :

$$p_{ij}^k(t) = \begin{cases} \frac{(\tau_{ij}(t))^\alpha . (\eta_{ij})^\beta}{\sum_{l \in J_i^k} (\tau_{il}(t))^\alpha . (\eta_{il})^\beta} & si : j \in J_i^k \\ 0 & si : j \notin J_i^k \end{cases} \tag{2.2}$$

Les paramètres α et β contrôlent l'importance relative de l'intensité de la piste « $\tau_{ij}(t)$ » et de la visibilité « η_{ij} ».

— Après un tour complet, chaque fourmi laisse une certaine quantité de phéromone $\Delta\tau_{ij}^k(t)$ sur l'ensemble de son parcours, la quantité compte sur la qualité de la solution trouvée :

$$\Delta\tau_{ij}^k(t) = \begin{cases} \frac{Q}{L^k(t)} & si : (i;j) \in T^k(t) \\ 0 & si : (i;j) \in T^k(t) \end{cases} \tag{2.3}$$

où $T^k(t)$ est le trajet effectué par la fourmi k à l'itération t, $L^k(t)$ la longueur de la tournée et Q un paramètre fixé.

— L'algorithme ne serait pas complet sans le processus d'évaporation des pistes de phéromone. La règle de mise à jour des pistes est la suivante :

$$\tau_{ij}(t+1) = (1-\rho).\tau_{ij}(t) + \sum_{k=1}^{m} \Delta\tau_{ij}^k(t) \tag{2.4}$$

où m est le nombre de fourmis et ρ le taux d'évaporation. [12]

2.3.3.1 Algorithme

Algorithm 4 Colonie de fourmis

Pour $t = 1, ..., tmax$

Pour chaque fourmi $k = 1, ..., m$

choisir une ville au hasard

Pour chaque ville non visitée i

choisir une ville j, dans la liste J_i^k des villes restantes, selon la formule 2.2

Fin Pour

Déposer une piste $\Delta\tau_{ij}^k(t)$ sur le trajet $T^k(t)$ conformément à l'équation 2.3

Fin Pour

Evaporer les pistes selon la formule 2.4

Fin Pour [25]

2.3.3.2 Exemple

- Les fourmis se déplacent entre le nid(E) et la source de la nourriture (A) sur le chemin A-E. (voir la figure 2.10)

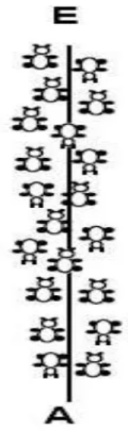

FIGURE 2.10: Déplacement des fourmis

- Un obstacle qui coupe le chemin.
- La fourmi qui se déplace de A vers E et se trouve en B, a 2 choix (voir la figure 2.11) :
  1. Le chemin B-C-D
  2. Le chemin B-H-D

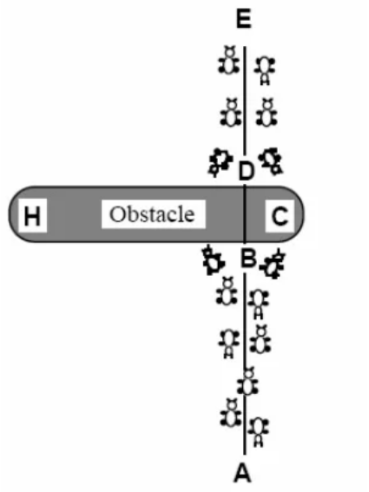

FIGURE 2.11: Chemin avec obstacle

- Le choix = f (intensité de phéromone)
- La première fourmi a des probabilités égale de suivre les deux chemins.
- Celle qui suit le chemin B-C-D arrive en premier en D illustrée dans la figure 2.12

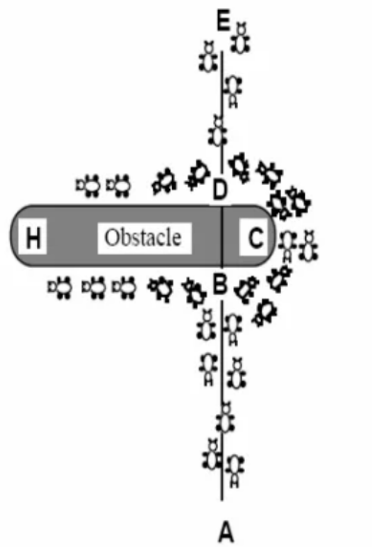

FIGURE 2.12: Déplacement des fourmis dans un chemin avec obstacle

- Les fourmis qui se retournent de E en D ont deux choix :
  1. Le chemin D-C-B

2. Le chemin D-H-B

- Le chemin D-C-B aura une plus forte intensité de phéromone causée par :

1. La moitié des fourmis qui prennent ce chemin de retour.
2. Le nombre supérieur des fourmis qui ont suivi le chemin B-C-D et qui se retournent. (voir la figure 2.13)

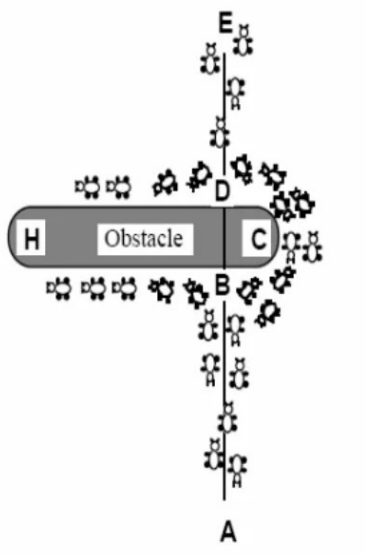

FIGURE 2.13: Déplacement des fourmis dans le meilleur chemin D-C-B

2.3.3.3 Avantages

L'algorithme des colonies des fourmis est une heuristique avec une très grande adaptabilité aux différents problèmes et elle est utilisée pour les problèmes basés sur des graphes. [22]

2.3.3.4 Inconvénients

L'algorithme des colonies des fourmis a un temps d'exécution parfois long et peut arriver à un état bloquant. [22]

2.3.4 Algorithme de Floyd-Warshall

L'algorithme Floyd-Warshall, également connu sous le nom d'algorithme Roy-Warshall, algorithme Roy-Floyd ou algorithme Floyd, est un algorithme permettant de trouver efficacement et simultanément les plus courts chemins entre tout couple de sommets dans un graphique pondéré. L'algorithme de Floyd-Warshall utilise une approche de programmation dynamique. C'est à dire chaque problème est découpé en sous-problèmes indépendants puis résolus et stockés pour les calculs ultérieurs. Il permet de travailler avec des arcs de poids négatifs mais non pas des cycles de poids négatifs. [8]

2.3.4.1 Algorithme

Algorithm 5 Floyd

Entrées : Un graphe $G = (S, A)$, une application $w : A \leftarrow R$

Sorties : Une application $\delta : X \text{ x } X \rightarrow R$

//Init

pour tous les $x, y \in X$ faire

$$\delta^0(x,y) \leftarrow \begin{cases} \infty & \text{si } (x,y) \notin A \\ 0 & \text{si } \quad x = y \\ w(x,y) & \text{si } (x,y) \in A \end{cases}$$

Fin pour

pour K de 1 à n faire

pour tous les $x, y \in A$ faire

$$\delta^k(x,y) \leftarrow min(\delta^{k-1}(x,y), (\delta^{k-1}(x,x_k) + \delta^{k-1}(x_k,y)))$$

Fin pour

Fin pour

si Il existe $x \in X$ tel que $\delta^n(x,x) < 0$ alors

retourner circuit de coût négatif

sinon

retourner δ^n (matrice n x n) [15]

2.3.4.2 Exemple

Nous allons avoir ici un exemple de l'algorithme de Floyd-Warshall appliqué sur le graphe suivante pour trouver le chemin le plus court. (voir la figure 2.14)

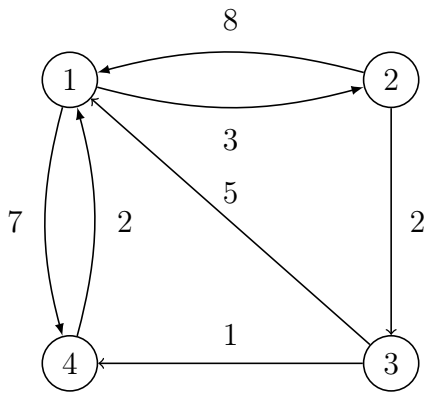

FIGURE 2.14: Exemple d'un graphe pour appliquer l'algorithme de Floyd

Dans un premier temps, nous initialisons la matrice de solution A^0, identique à celle du graphe d'entrée.

\	1	2	3	4
1	0	3	∞	7
2	8	0	2	∞
3	5	∞	0	1
4	2	∞	∞	0

On regarde dans la matrice A^0 les chemins où le sommet 1 peut être un sommet intermédiaire

\	1	2	3	4
1	0	3	∞	7
2	8	0	2	∞
3	5	∞	0	1
4	2	∞	∞	0

Le tableau 2.14 indique la distance de chemin des arcs sans intermédiaire et avec le sommet intermédiaire 1 en comparant les deux en gardant la valeur de plus court chemin des arcs :

u	intermédiaire	v	[u,v]	[u,x]+[x,v]	Coût
2	[2,1]+[1,3]	3	2	$8+\infty$	2
2	[2,1]+[1,4]	4	∞	8+7	15
3	[3,1]+[1,2]	2	∞	5+3	8
3	[3,1]+[1,4]	4	1	5+7	1
4	[4,1]+[1,2]	2	∞	2+3	5
4	[4,1]+[1,3]	3	∞	$2+\infty$	∞

TABLE 2.14: Coût minimale des arcs à partir du sommet 1

La nouvelle matrice A^1

\	1	2	3	4
1	0	3	∞	7
2	8	0	2	15
3	5	8	0	1
4	2	5	∞	0

Maintenant, le sommet 2 peut être un sommet intermédiaire.

\	1	2	3	4
1	0	3	∞	7
2	8	0	2	15
3	5	8	0	1
4	2	5	∞	0

Le tableau 2.15 indique la distance entre le chemin des arcs avec le sommet intermédiaire 2 et sans cette intermédiare en comparant les deux en gardant la valeur du plus court chemin des arcs.

u	intermédiaire	v	[u,v]	[u,x]+[x,v]	Coût
1	[1,2]+[2,3]	3	∞	3+2	5
1	[1,2]+[2,4]	4	7	3+15	7
3	[3,2]+[2,1]	1	5	8+8	5
3	[3,2]+[2,4]	4	1	8+15	1
4	[4,2]+[2,1]	1	2	5+8	2
4	[4,2]+[2,3]	3	∞	5+2	7

TABLE 2.15: Coût minimale des arcs à partir du sommet 2

La nouvelle matrice A^2

\	1	2	3	4
1	0	3	5	7
2	8	0	2	15
3	5	8	0	1
4	2	5	7	0

Considérons le sommet 3 comme intermédiaire :

\	1	2	3	4
1	0	3	5	7
2	8	0	2	15
3	5	8	0	1
4	2	5	7	0

Le tableau 2.16 indique la distance entre le chemin des arcs avec le sommet intermédiaire 3 et sans cette intermédiare en comparant les deux en gardant la valeur du plus court chemin des arcs :

u	intermédiaire	v	[u,v]	[u,x]+[x,v]	Coût
1	[1,3]+[3,2]	2	3	5+8	3
1	[1,3]+[3,4]	4	7	5+1	6
2	[2,3]+[3,1]	1	8	2+5	7
2	[2,3]+[3,4]	4	15	2+1	3
4	[4,3]+[3,1]	1	2	1+5	2
4	[4,3]+[3,2]	2	5	7+8	5

TABLE 2.16: Coût minimale des arcs à partir du sommet 3

La nouvelle matrice A^3

\	1	2	3	4
1	0	3	5	6
2	7	0	2	3
3	5	8	0	1
4	2	5	7	0

Considérons le sommet 4 comme intermédiaire :

\	1	2	3	4
1	0	3	5	6
2	7	0	2	3
3	5	8	0	1
4	2	5	7	0

Le tableau 2.17 indique la distance entre le chemin des arcs avec le sommet intermédiaire 4 et sans cette intermédiare en comparant les deux en gardant la valeur du plus court chemin des arcs :

u	intermédiaire	v	[u,v]	[u,x]+[x,v]	Coût
1	[1,4]+[4,2]	2	3	6+5	3
1	[1,4]+[4,3]	3	5	6+7	5
2	[2,4]+[4,1]	1	7	3+2	5
2	[2,4]+[4,3]	3	2	3+7	2
3	[3,4]+[4,1]	1	5	1+2	3
3	[3,4]+[4,2]	2	8	1+5	6

TABLE 2.17: Coût minimale des arcs à partir du sommet 4

La nouvelle matrice A^4

\	1	2	3	4
1	0	3	5	6
2	5	0	2	3
3	3	6	0	1
4	2	5	7	0

La matrice du plus court chemin A^4 du graphe 2.14.

\	1	2	3	4	
1	0	3	5	6	⟵ plus court chemin à partir du sommet 1
2	5	0	2	3	⟵ plus court chemin à partir du sommet 2
3	3	6	0	1	⟵ plus court chemin à partir du sommet 3
4	2	5	7	0	⟵ plus court chemin à partir du sommet 4

2.3.4.3 Avantages

L'algorithme de Floyd-Warshall est un algorithme permettant de trouver le chemin le plus court dans un graphe pondéré avec des poids de bord positifs ou négatifs (mais pas de cycles négatifs). Pour ce faire, il compare tous les chemins possibles à travers le graphe entre chaque paire de sommets et une comparaison O(V3) dans le graphe.

2.3.4.4 Inconvénients

L'algorithme de Floyd-Warshall fonctionne sur tous les nœuds qui sont liés sans exception et renvoie une matrice contenant tous les chemins les plus courts entre tous les nœuds. Par exemple, l'algorithme Floyd-Warshall calcule le passage à travers plus de 50 pays en un détour ou deux, l'itinéraire le moins taxé sera généralement obtenu. En termes de complexité temporelle, il est donc plus facile d'utiliser l'algorithme de Dijkstra. [11]

2.4 Conclusion

Dans ce chapitre, nous avons introduit des définitions et des termes sur les graphes utilisés dans les algorithmes de plus court chemin. Ensuite, nous avons présenté quatre algorithmes de plus court chemin en commençant par l'algorithme de Bellman-ford par suite algorithme de Dijkstra après l'algorithme de colonie de fourmis en finissant par l'algorithme de Floyd.

L'algorithme de Dijkstra résout le problème de la recherche du plus court chemin lorsque tous les coûts sont positifs ou nuls. Ce dernier est le meilleur algorithme utilisé d'après les quatre définis pour construire le plus court chemin entre les positions où le

robot se trouve et la position où il veut se rendre. Nous préférons l'algorithme de Dijkstra par rapport aux autres algorithmes car il est suffisamment rapide et il donne exactement le plus court chemin dans toutes les situations et nous n'avons pas des couts négatifs dans les données collectées par le robot. Nous avons choisi cet algorithme en faisant attention au résultat que nous voulons. Un autre avantage de l'algorithme est sa large gamme d'application. En principe, les algorithmes peuvent être appliqués à différents problèmes pour lesquels des mécanismes de construction de solutions peuvent être conçus.

Chapitre 3

Réalisation pratique du robot

3.1 Introduction

Dans ce chapitre, nous allons passer à la réalisation pratique du robot. Nous allons d'abord commencer par détailler le fonctionnement du Robot, ensuite nous réaliserons une conception pour notre Robot afin de pouvoir simuler le projet sur un logiciel dédié. Après la simulation, nous testerons le projet si tout marche comme prévu, nous procéderons avec les tests finaux et la validation.

3.2 Le fonctionnement du Robot

Notre robot peut porter des charges jusqu'à 20 kg et le transporter à travers le suit d'une ligne noire à l'aide des captures infrarouges il peut également résoudre une labyrinthe et éviter tous les obstacles sur son chemin pour éviter les chocs de collusion et peut il donner l'état de batterie et le poids porté sur le robot sur un écran LCD.

Nous devons d'abord construire un prototype de ce robot (suiveur de ligne).

3.3 Robot suiveur de ligne

Un robot suiveur de ligne est un robot qui a pour but de suivre sa ligne à l'aide des capteurs infrarouges. Il est très utilisé dans les différents applications dans le domaine industriel .Par exemple nous pouvons le trouver dans les stocks pour déplacer des produits qui ont un grand poids (dans notre cas).

Parmi les avantages d'un robot suiveur de ligne.

— Une bonne fiabilité mécanique.

— Un faible cout de fabrication.

— Simple pour la réalisation.

3.4 Schéma électronique du prototype

Nous allons réaliser le schéma dans la figure 3.1 pour la version basique (Suiveur de ligne et détecteur d'obstacle) avec logiciel ISIS proteus :

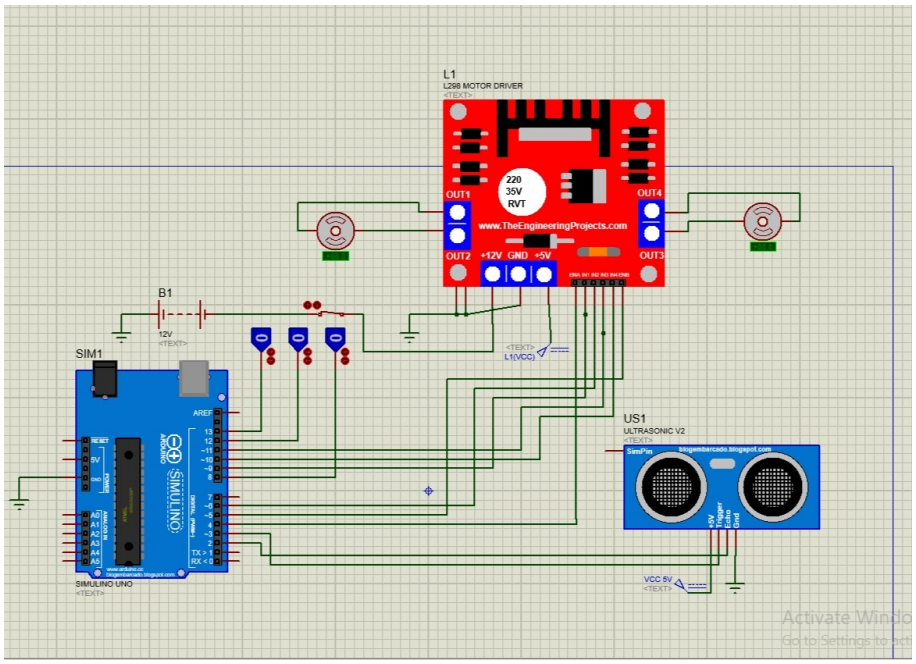

FIGURE 3.1: Simulation avec logiciel ISIS proteus.

Dans ce schéma, nous avons simuler le comportement basique d'un robot suiveur de ligne, pour cette raison la version final comportera certains composants qui ne figure pas dans ce montage. Nous allons simuler seulement les composants disponibles dans la librairie du logiciel ISIS proteus.

La figure 3.2 montre l'organigramme principal du robot suiveur de ligne

- ▷ **G** : le robot marche à gauche.
- ▷ **A** : le robot marche en avant.
- ▷ **D** : le robot marche à droite.

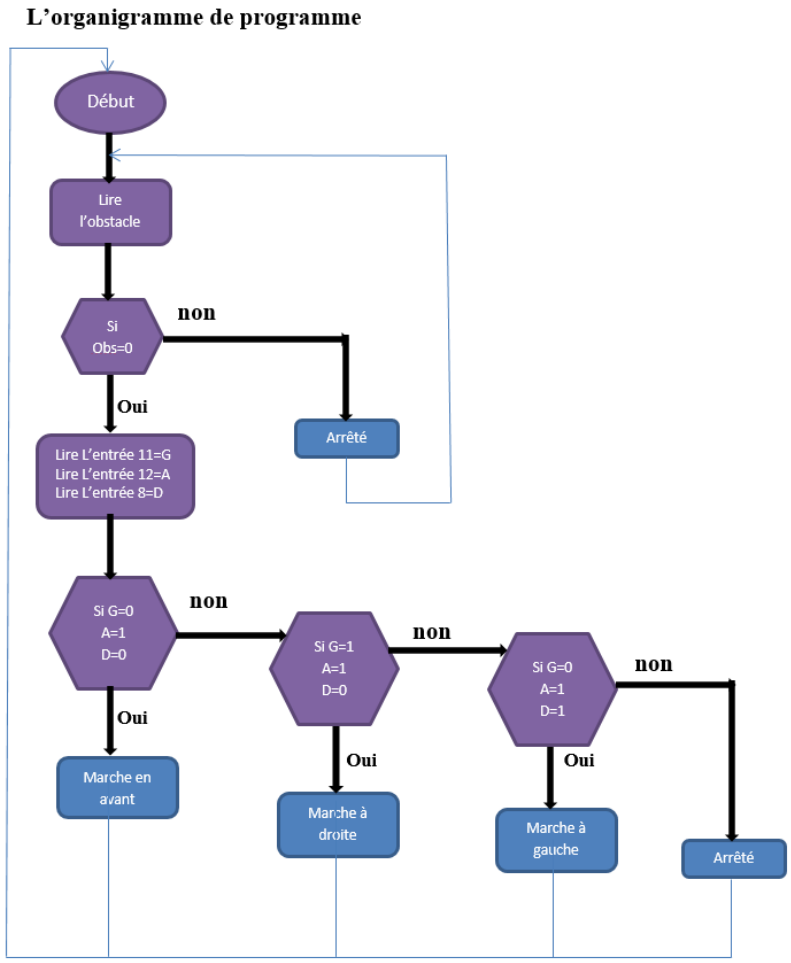

FIGURE 3.2: L'organigramme de mouvement d'un suiveur de ligne basic.

3.5 La détection de ligne et de mouvement de robot

Le tableau 3.5 montre les valeurs possibles des détecteurs de bande et le mouvement du robot

État capteur à gauche	État capteur Centre	État capteur à droit	État capteur Obstacle	État du robot
0	0	0	0	Robot arrête
0	0	1	0	Marche à gauche
0	1	0	0	Marche en avant
0	1	1	0	Marche à gauche
1	0	0	0	Marche à droit
1	0	1	0	Robot arrête
1	1	0	0	Marche à droit
1	1	1	0	Robot arrête
X	X	X	1	Robot arrête

TABLE 3.1: Les différents états des capteurs.

— Le capteur suiveur de ligne permet à détecter la couleur noire ce qui implique :

- L'état « 1 » du capteur indique la détection de la couleur à la bande « noire ».
- L'état « 0 » du capteur indique la non détection de la couleur à la bande « noire ».

— Le capteur ultrason permet à détecter l'obstacle ce qui implique :

- L'état « 1 » du capteur indique la détection d'obstacle.
- L'état « 0 » du capteur indique la non détection d'obstacle.

3.6 La régulation PID sur Matlab

Dans cette partie nous allons essayer de faire une régulation PID de vitesse mais nous n'avons pas atteint à attacher un encodeur au moteur alors nous allons faire une simulation de régulation PID sur Matlab.

Nous allons choisir la méthode de Ziegler-Nichols.

La méthode de Ziegler-Nichols est une méthode heuristique de réglage d'un régulateur PID. Elle a été développée par John G.Ziegler et Nathaniel B.Nichols.[10]

3.6.1 Présentation de la méthode

D'abord la première méthode de Ziegler-Nichols est appliquée seulement pour les systèmes qui en une réponse indicielle proche d'un système de premier ordre.

Ça veut dire que le Gp(s) c'est un système qui a une réponse indicielle proche d'un système de premier ordre. Si nous avons à l'entrée de ce système une fonction échelon unitaire ou une autre valeur, la sortie de ce système est la réponse suivante (voir la figure 3.3).

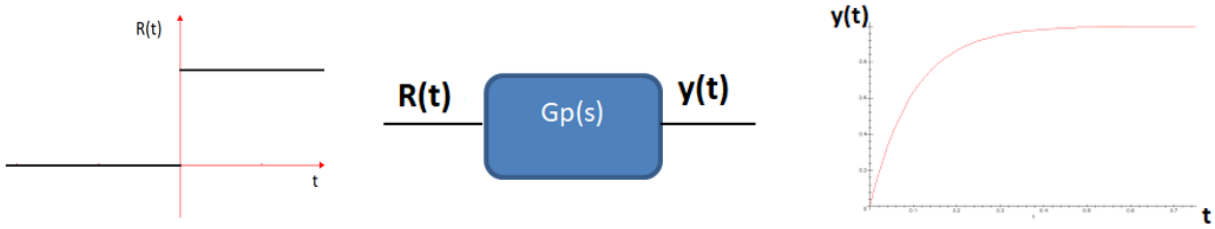

FIGURE 3.3: La fonction de transfert avec sa consigne et sa réponse.

Dans ce cas nous pouvons appliquer la première méthode de Ziegler-Nichols sur le système Gp(s).

Maintenant l'objectif de cette méthode est de calculer les paramètres de correcteur Gc(s) pour notre système Gp(s) (voir la figure 3.4).

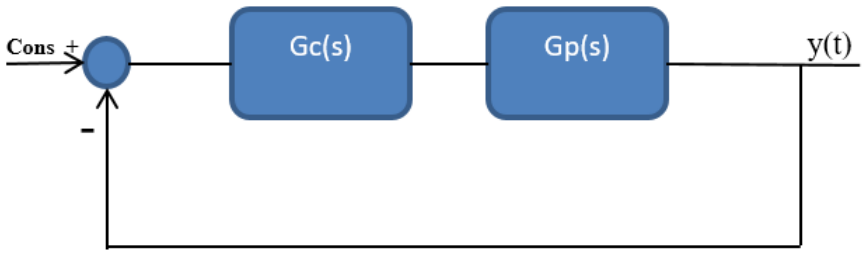

FIGURE 3.4: Système corrigée en boucle fermée.

- **Gc(s)** : Le correcteur PID
- **Gp(s)** : FTBO du notre système
- **Cons** : La consigne
- **y(t)** : La sortie

3.6.2 Procédure

1er étape

Nous allons appliquer une entrée échelon d'amplitude K sur notre système et nous allons tracer la sortie de ce système y(t) (voir la figure 3.5).

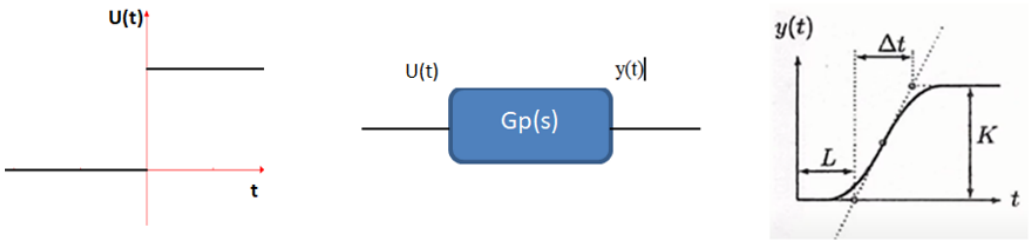

FIGURE 3.5: Notre système avec sa consigne et réponse

2ème étape

Il faut déterminer le point d'inflexion de la courbe de la sortie et nous traçons la tangente.

3ème étape

Déterminer le point d'intersection entre la tangente avec l'axe du temps et la valeur finale de la sortie

4ème étape

Déterminer les valeurs de $\Delta t, L$ et K et calculer $R = \frac{K}{\Delta t}$ avec :

Δt : la durée entre les deux intersections de la tangent

L : le retard

K : la valeur finale de la sortie

R : la tangente.

5ème étape

Nous Déterminons les paramètres dans le tableau 3.2

Gc(s)	Fonction	Paramètre
P	Kp	$Kp = \frac{1}{R*L}$
PI	$Kp(1 + \frac{1}{Ti*s})$	$Kp = \frac{0.9}{R*L}$ et $Ti = \frac{L}{0.3}$
PID	$Kp(1 + \frac{1}{Ti*s} + Td * s)$	$Kp = \frac{1.2}{R*L}$; $Ti = \frac{L}{0.5}$ et $Td = 0.5L$

TABLE 3.2: Les paramètres de correcteur PID

3.7 Modélisation du moteur à courant continu (MCC)

L'induit d'un moteur à courant continu peut être représenté par le schéma 3.6 :

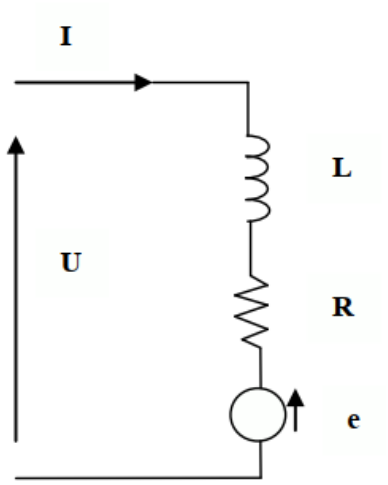

FIGURE 3.6: Induit moteur à courant continu

Avec :

R : La résistance interne de l'induit du moteur à courant continu.

L : L'inductance interne de l'induit du moteur à courant continu.

e(t) = Force contre électromotrice du moteur $[V]$

$u(t)$ = Tension du moteur

$i(t)$ = Intensité dans le moteur

3.7.1 Équations électromécaniques du MCC

3.7.1.1 Équations électriques

La loi des mailles permet d'écrire :

$$u(t) = R.i(t) + L\frac{di}{dt} + e(t) \tag{3.1}$$

Avec $e(t) = K.\Omega(t)$

K = Coefficient de la force contre électromotrice $[V/(rad/s)]$

$\Omega(t)$ = Vitesse angulaire du moteur $[rad/s]$

3.7.1.2 Équations mécaniques

Le principe fondamental de la dynamique permet d'écrire :

$$J\frac{d\Omega}{t} + f\Omega = C_{em} - Cr \tag{3.2}$$

Avec : $C_{em} = K.I$

C_{em}= Couple exercé par le moteur $[N.m]$

Cr= Couple résistant sur l'axe moteur $[N.m]$

J = Inertie équivalente ramenée sur l'arbre moteur $[kg.m]$

f : coefficient de frottement visqueux $[N.m.s]$

K = Constante de couple

3.7.1.3 Équations électromécaniques du MCC dans le domaine de Laplace

La transformée de Laplace de l'équation électrique :

$$u(s) = R.i(s) + sL.i(s) + K\Omega(s) \tag{3.3}$$

Soit :

$$i(s) = \frac{u(s) - K\Omega(s)}{Ls + R} \tag{3.4}$$

La transformée de Laplace de l'équation mécanique :

$$Js\Omega(s) + f\Omega(s) = Ki(s) - Cr \tag{3.5}$$

Soit :

$$\Omega(s) = \frac{ki(s) - Cr}{Js + f} \tag{3.6}$$

À partir des relations 3.4 et 3.6 nous pouvons établir le schéma fonctionnel de la machine à courant continu illustrée dans la figure 3.7 :

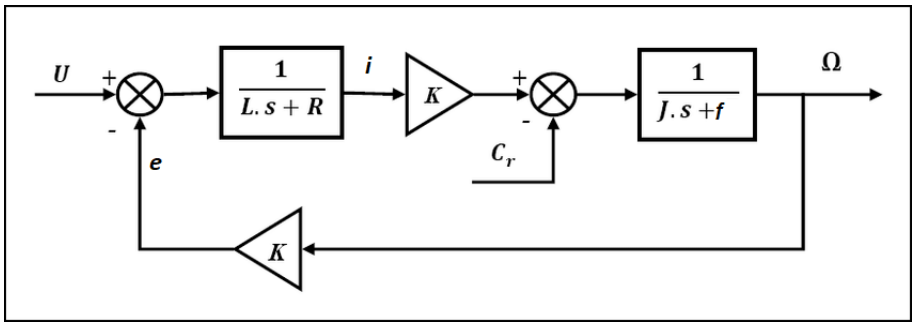

FIGURE 3.7: Schéma fonctionnel du moteur à courant continu

3.7.2 Fonction de transfert du MCC

En négligent le couple résistant devant le couple électromagnétique nous aurons :

$$\Omega(s) = \frac{Ki(s)}{Js + f} \tag{3.7}$$

Le courant s'écrit alors :

$$i(s) = \frac{(Js + f).\Omega(s)}{K} \tag{3.8}$$

En remplaçant le courant dans l'équation électrique nous obtenons :

$$u(s) = R.\frac{(Js + f).\Omega(s)}{K} + sL.\frac{(Js + f).\Omega(s)}{K} + K\Omega \tag{3.9}$$

Après développement nous obtenons la fonction de transfert de la vitesse par rap-

port à la tension d'entrée :

$$\frac{\Omega}{u} = \frac{K}{LJs^2 + (RJ + f)s + Rf + K^2} \tag{3.10}$$

Donc

$$Gp(s) = \frac{\Omega}{u} = \frac{K}{(Js + f) * (Ls + R)s + K^2} \tag{3.11}$$

3.8 La simulation sur Matlab

Notre fonction de transfert du moteur est :

$$Gp(s) = \frac{K}{(Js + f) * (Ls + R)s + K^2} \tag{3.12}$$

Avec

R : résistance aux bornes de l'induit. $R = 1W$

L : inductance aux bornes de l'induit. $L = 0,5mH$

K : constante de f.é.m. $K = 0,01V.s/rad$

j : moment d'inertie. $J = 0.01kg.m$

f : coefficient de frottement visqueux. $f = 0.1N.m.s$

s : la transformée de la place.

Le script de matlab dans la figure 3.8 :

```
clear all; close all; clc
%Les parametres
J = 0.01;
f = 0.1;
K = 0.01;
R = 1;
L = 0.5;
%FTBO de moteur dc
s = tf('s');
P_motor = K/((J*s+f)*(L*s+R)+K^2);
[ys,ts]=step(P_motor,2);

plot(ts,ys,'r','LineWidth',2)
xlabel('ts')
ylabel('P-motor')
title ('Réponse indicielle du moteur')
grid
```

FIGURE 3.8: Le script FTBO de moteur dc

La réponse indicielle du moteur est représenté dans la figure 3.9 :

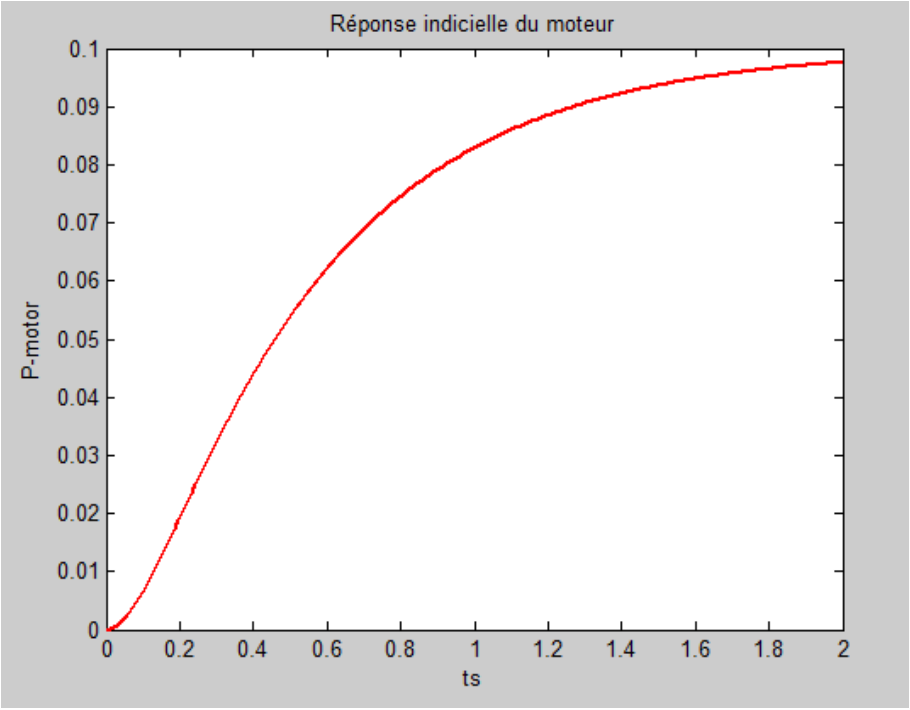

FIGURE 3.9: La réponse indicielle du moteur

Interprétation

Nous remarquons que notre moteur nous donne une réponse indicielle de premier ordre d'amplitude $0.1 rad/s$.

Ensuite nous calculons les paramètres de PID par la méthode de Ziegler-Nichols comme nous avons expliquer précédemment et appliquer un correcteur PID sur notre système pour avoir le résultat illsutré par la figure 3.11.

Le script dans la figure 3.10 représente la méthode de Ziegler-Nichols :

```
%Méthode de Zigler-Nickols
L=0.05;dt=3.95;H=0.1; R=H /dt; %par identification sur la courbe de FTBO de moteur dc
%%correcteur pid
Kp=1.2/(R*L)
Ti=L/0.5
Td=0.5*L
Gc_PID=Kp*(1+1/(Ti*s)+Td*s); %FTBO de correcteur PID
H_PID=feedback(Gc_PID*P_motor,1); %FTBF de systeme corrigé
[y,t]=step(10*H_PID,1);

plot(t,y,'k','LineWidth',2)
xlabel('ts')
ylabel('H-PID')
title ('La réponse en boucle fermée')
grid
```

FIGURE 3.10: Le script de la méthode de Ziegler-Nichols

La réponse en boucle fermée obtenu est illustrée dans la figure 3.11 :

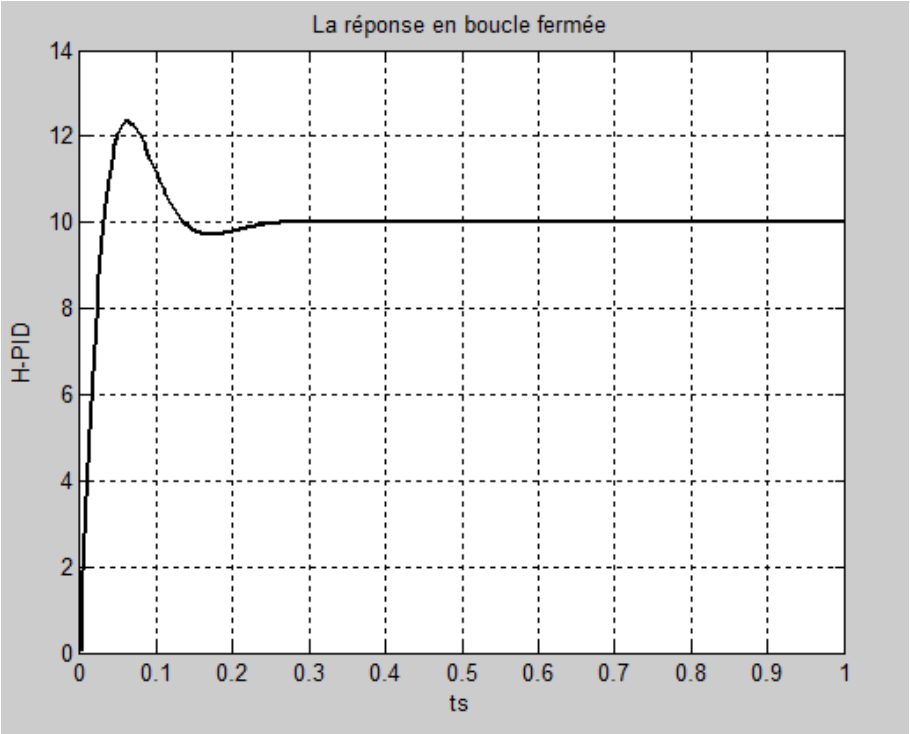

FIGURE 3.11: La réponse en boucle fermée

3.8.1 Interprétation

Nous remarquons que le dépassement et de 2.2 rad/s, le temps de réponse 0.22s et l'erreur statique et nulle.

3.8.2 L'amélioration de la méthode

Nous allons voir l'effet lorsque nous augmentons les paramètres PID sur les performances du système dans le tableau 3.3

Contrôleur	Ts	Mp	Tr	Erreur statique
Kp	Diminue	Augmente	Effet faible	Diminue
Ki	Diminue	Augmente	Augmente	Elimine
Kd	Effet faible	Diminue	Diminue	Effet faible

TABLE 3.3: L'effet de chaque paramètre PID

Avec

- Ts : le temps de monté.

- Mp : le dépassement.
- Tr : le temps de réponse.

Donc pour améliorer cette réponse nous allons agir sur l'action dérivé et intégrale, donc nous avons multiplié par 4 pour obtenir le résultat illustré par la figure 3.13.

Le script de Matlab est représenté dans la figure 3.12) :

```
19      %%L'amélioration de la méthode
20 -    Kp=1.2/(R*L)
21 -    Ti=4*L/0.5
22 -    Td=4*0.5*L
23 -    Gc_PID=Kp*(1+1/(Ti*s)+Td*s); %FTBO de correcteur PID
24 -    H_PID=feedback(Gc_PID*P_motor,1); %FTBF de systeme Amélioré
25 -    [y,t]=step(10*H_PID,1);
26
27 -    plot(t,y,'k','LineWidth',2)
28 -    xlabel('ts')
29 -    ylabel('H-PID')
30 -    title ('La réponse en boucle fermée')
31 -    grid
```

FIGURE 3.12: Amélioration de la méthode

La réponse améliorée en boucle fermée obtenu est dans la figure 3.13 :

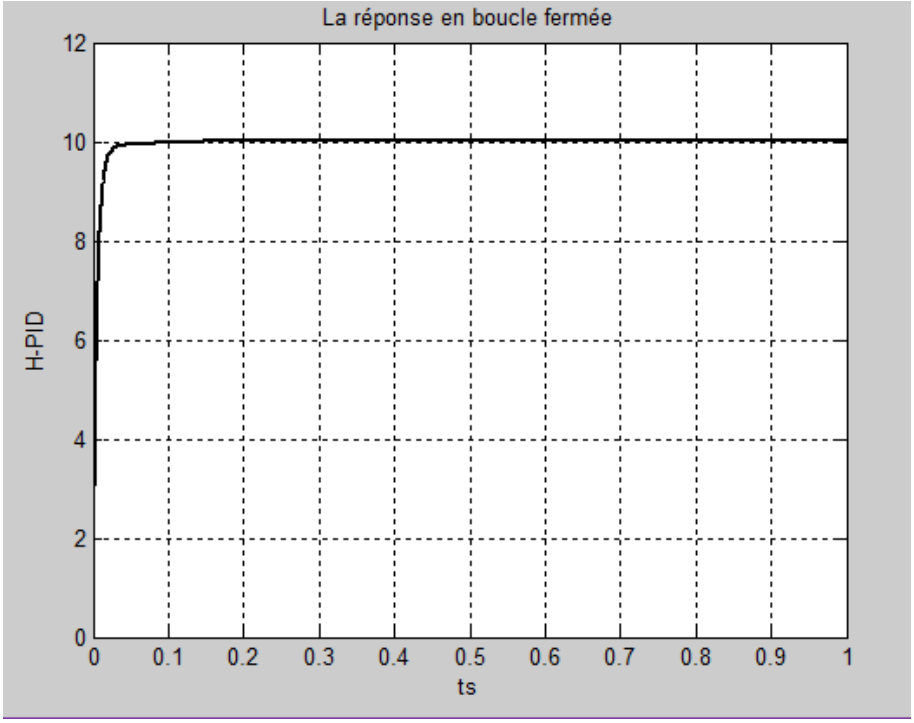

FIGURE 3.13: La réponse améliorée en boucle fermée

Interprétation

Nous remarquons que lorsque nous allons agir sur l'action dérivée, le dépassement disparu et le temps de réponse diminuent jusqu'à $0.04s$.

3.9 Organigramme du notre projet

Le schéma bloc explicatif du notre robot AGV final est présenté sur la figure 3.14

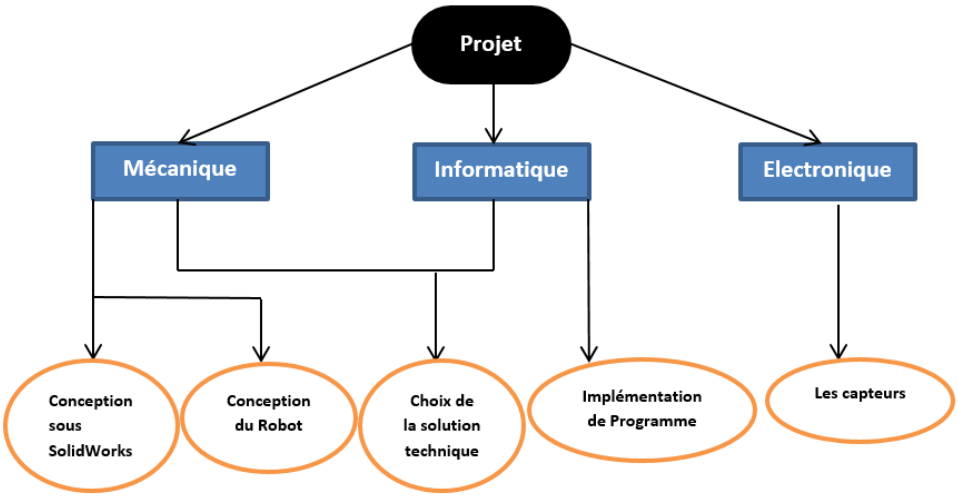

FIGURE 3.14: Organigramme du notre projet.

3.9.1 La partie mécanique

3.9.1.1 La plaque de châssis

Le châssis est une base sur laquelle sont disposés les éléments de notre robot, il doit être solide et léger pour obtenir une forme adaptée pour notre application (déplacer des produits qui ont poids important). Pour notre application nous avons choisi au début un châssis en contreplaqué découpé par une machine CNC laser ayant les dimensions représentées dans la figure 3.15.

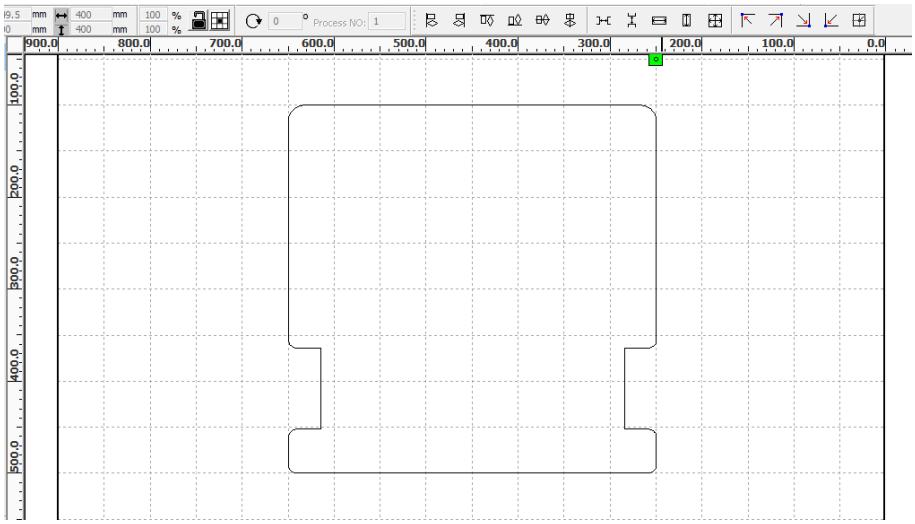

FIGURE 3.15: Les dimensions de la plaque de châssis

Mais ce châssis n'a pas donné des résultats positifs lors de son essai sur le terrain en raison de sa nature extensible. Le châssis final représenté à la figure 3.16.

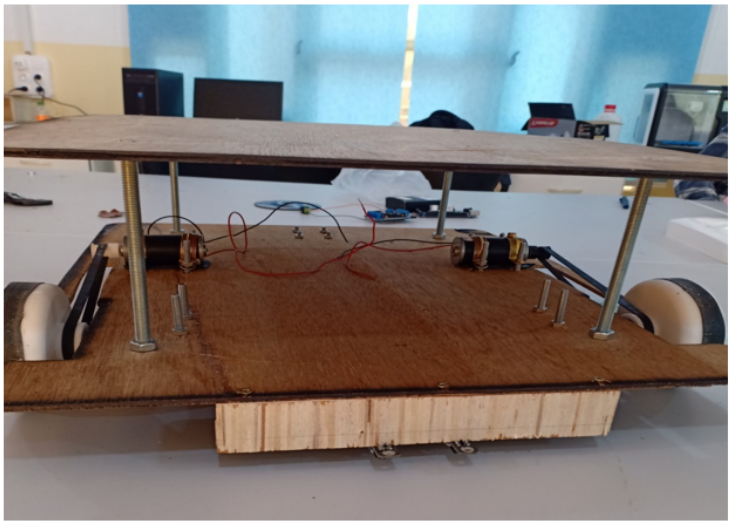

FIGURE 3.16: Châssis final en contreplaqué

Alors après nous avons choisi un châssis en résine ayant la forme représentée à la figure 3.17.

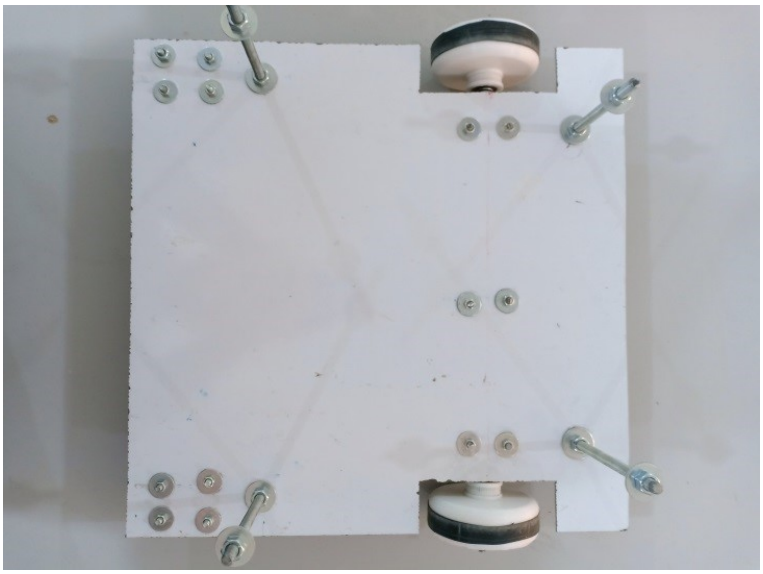

FIGURE 3.17: Châssis en résine

3.9.1.2 Les roues

Les roues représentent une partie très importante pour le déplacement de l'AGV et l'excès de poids. Il y a deux types des roues utilisés dans notre projet (deux roues motrices et deux roues libres). Concernant les roues motrices nous allons désigner des roues avec une poulie intégrée sur Solidworks ayant les dimensions représentées à la figure 3.18 et 3.19.

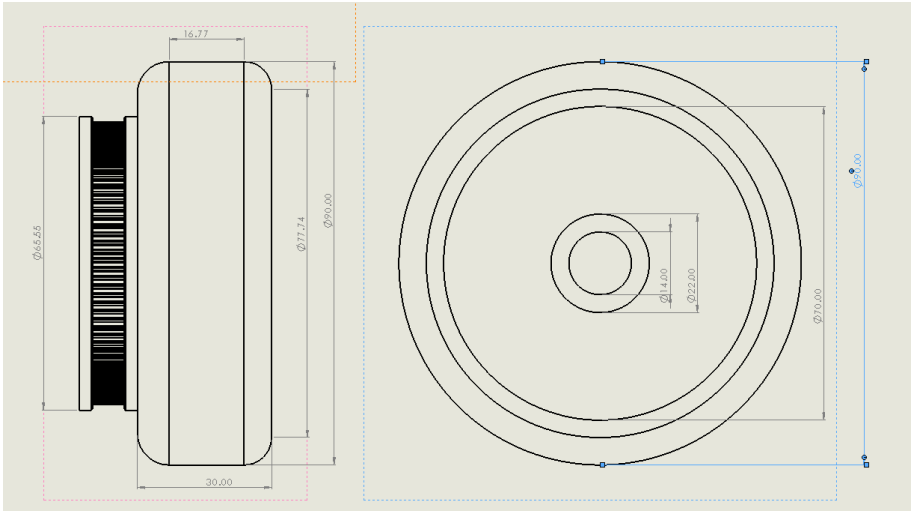

FIGURE 3.18: Dimension de la roue motrice

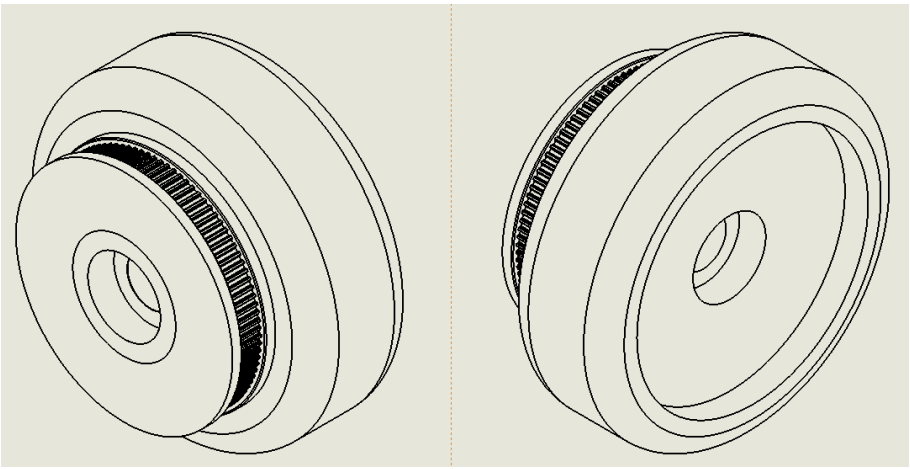

FIGURE 3.19: La roue motrice

La figure 3.20 représente l'impression 3D d'une roue motrice

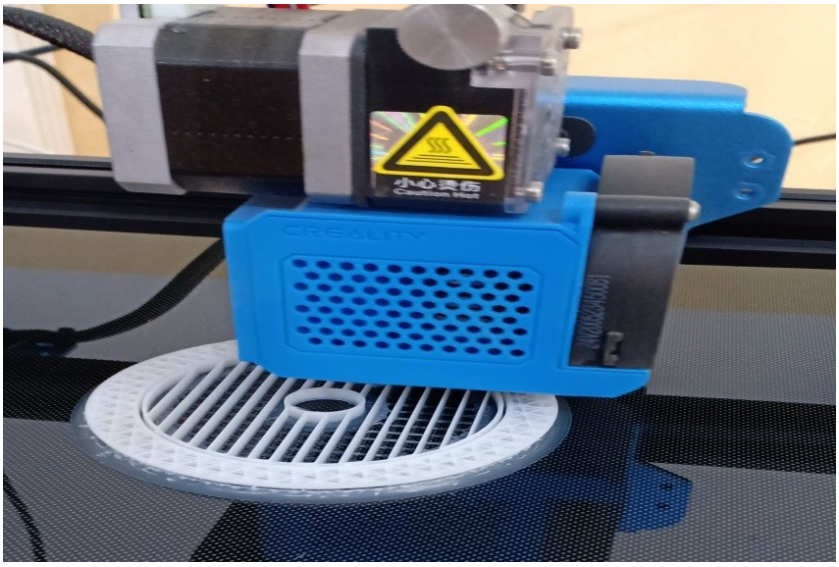

FIGURE 3.20: Impression d'une roue motrice

Les roues motrices du robot suiveur doivent être couverte en matière de caoutchouc pour éviter le glissement et les pertes du mouvement, elles sont fixées sur un axe fixe et des roulements pour une meilleure transmission mécanique telle que la poulie fixée sur l'arbre du moteur qui contienne 20 dents et la poulie réceptrice de la roue motrice contient 100 dents, donc un rapport de 5 tel que la vitesse angulaire de sortie doit être divisée par 5 et

le couple multiplié par 5.

$$r = \frac{Dm}{Dr} = \frac{\omega s}{\omega e} \tag{3.13}$$

r : le rapport.

Dm : les dents de la poulie motrice $Dm = 20$.

Dr : les dents de la poulie réceptrice Dr =100.

Tel que : Dr=5.Dm.

ωe : la vitesse angulaire de poulie motrice.

ωs : la vitesse angulaire de la poulie réceptrice.

$$\omega s = \omega e.\frac{Dm}{Dr} = \frac{\omega e}{5} \tag{3.14}$$

Et la puissance de l'entrée est :

$$Pa = Ce.\omega e \tag{3.15}$$

Et la puissance de sortie est :

$$Pu = Cu.\omega s \tag{3.16}$$

Tel que : $Pa = Pu$

$$Cu = \frac{Pu}{\omega s} = 5.\frac{Pu}{\omega e} \tag{3.17}$$

$$\Longrightarrow Cu = 5.Ce$$

Avec :

Ce : le couple d'entré.

Cu : le couple de sortie.

Et les deux roues libres sont utilisées pour la stabilité du robot pour les différents mouvements possibles. La figure 3.21 représente le système des roues.

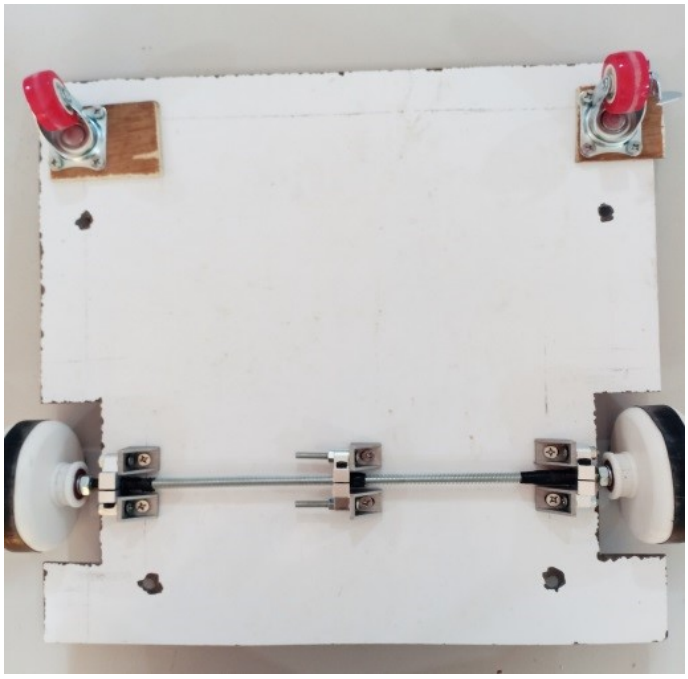

FIGURE 3.21: Système des roues.

3.9.1.3 Les moteurs électriques

Pour que le robot se déplace, il est nécessaire d'utiliser des moteurs, et parce que nous avons décidé de fabriquer un robot AGV, nous utilisons des puissants moteurs à courant continu (deux moteur A et B) pour contrôler les deux roues avant.

FIGURE 3.22: Moteur à courant continu à balais

Caractéristiques du moteur utilisé dans ce projet

- Type de produit : moteur à courant continu à balais.
- Modèle : XD-3420.
- Puissance nominale : $30W$.
- Tension nominale : 12 V.
- Vitesse nominale : 3500 tr/min.
- Courant : 0.5 A.
- Longueur de l'arbre de sortie : $26mm/1.02inch$.
- Couple : $1kg.f.cm$.
- Poids : $460g$. [27]

3.9.2 La partie électronique

Après avoir décrit le principe et la structure de la partie mécanique, nous allons commencer maintenant à travailler sur la partie électronique de l'ensemble du système. Commençons par une description des différents composants (voir la figure 3.23).

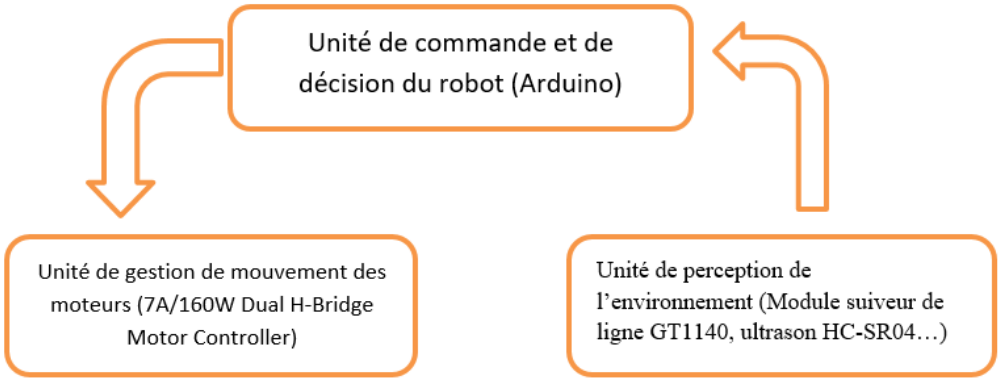

FIGURE 3.23: Structure de la partie électronique.

La figure 3.23 représente une description du système composée de trois unités. Nous utilisons un module suiveur de ligne GT1140 et un ultrason HC-SR04 (Unité de Perception de l'environnement) pour recevoir les informations de l'environnement, qui seront transmises à une carte Arduino Mega (unité de commande et de décision) pour les traiter. Après le traitement, la carte Arduino transmet la décision à l'unité de gestion du mouvement des moteurs (contrôleur de moteur à double pont en H $7A/160W$). Pour déplacer le robot, un moteur à courant continu haute puissance est utilisée.

3.9.2.1 Unité de commande

Le logiciel Arduino

Il est entièrement gratuit et peut-être téléchargé gratuitement sur Arduino. Ainsi que les codes source les plus simples sont largement annoté. Il possède une interface facile à manipuler. Pour programmer une carte Arduino il suffit de connecté le câble USB du module Arduino à l'ordinateur, puis nous ouvrons l'un des croquis sous le logiciel Arduino et commencons à compiler, après le chargement du code compilé sur Arduino, ce dernier exécutera automatiquement le programme lorsque l'opération est effectuée correctement

par conséquent l'ordinateur n'est plus nécessaire, sauf pour l'alimentation. Le projet fonctionne directement (voir la figure 3.24 et 3.25).

FIGURE 3.24: Logiciel arduino IDE sous Windows

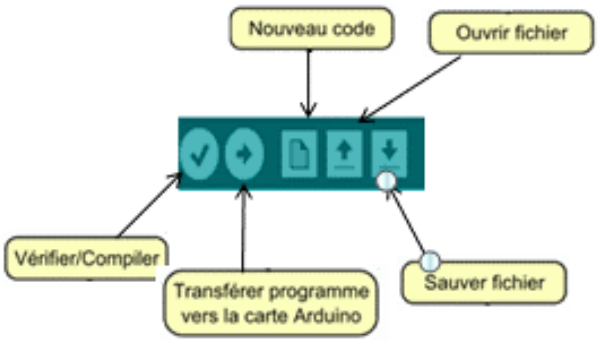

FIGURE 3.25: La barre des icones

Les avantages

1. Economique : les cartes microcontrôleurs son bon marché et l'éditeur est gratuit.
2. Facile à programmer : le langage arduino est très simple (C et C++).

3. Multiplateforme (Windows, Mac, Linux).
4. Logiciel et matériel open-source : le schéma de la carte et disponible pour tous.
5. Nombreuses librairies : ce qui représente un grand nombre de fonctions facile à programmées.
6. Nombreuse extension matérielles : facilement en brochable sur la carte à broche microcontrôleur.
7. Communauté très active : nombreux blog et site web de passionné permettrons de retrouvé de nombreuse ressources online.

Alors dans notre travaille nous avons utilisé un arduino Méga (voir la figure 3.26).

FIGURE 3.26: La carte Arduino Méga

Le signal PWM

Le PWM (Pulse Width Modulation) est un signal très utilisé en électronique. Il permet de commander la vitesse d'un moteur à courant continu (DC) selon notre besoin. Le principe est de faire varier (moduler) la durée /la largeur d'une impulsion de 5v.

PWM : Conversion numérique /analogique

Le signal PWM peut commander un composant électronique :

- Avec un rapport cyclique de 100%, le composant voit une tension en entrée de5v.
- Avec un rapport cyclique de 50%, le composant voit une tension en entrée de 2.5v. [23]

Le microcontrôleur de l'Arduino peut générer un signale PWM, Mais, nous ne pouvons utiliser que certaines broches, la figure 3.27 représente les pines de signal PWM sur l'Arduino Méga. Le rapport cyclique est exprimé en un chiffre entre 0 et 255 (50%correspond à 127) : codage sur 8 bits.

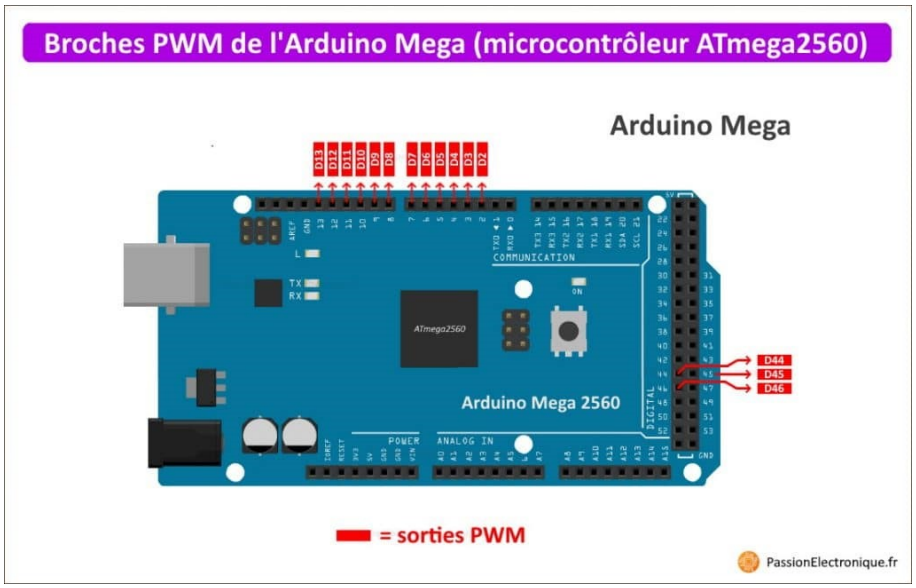

FIGURE 3.27: Les pines de signal PWM sur l'Arduino Méga

Notre Arduino Mega est capable de générer du PWM sur 15 broches de sortie via la fonction analogWrite, qui utilise les timers pour générer un signal d'onde carrée à fréquence fixe et largeur variable à la sortie pour les impulsions d'état élevé.

3.9.2.2 Unité de perception de l'environnement

Module suiveur de ligne GT1140

Premièrement, puisque la tâche principale de ce robot est de suivre une trajectoire, il a besoin d'un capteur de détection de bande de piste, la question est de savoir comment ça marche pour reconaitre l'existence de cette bande.

Parmi les nombreux capteurs qui peuvent effectuer la tâche, nous avons sélectionné un capteur infrarouge GT1140, dont le schéma et la mise en œuvre ne sont pas compliqués du fait de sa disponibilité, et leur coût n'est pas cher.

Dans notre projet nous avons utilisé sept capteurs infrarouges (GT1140)(voir la figure 3.28), Ce Module suiveur de ligne basé sur un réflecteur optique et un amplificateur. Le seuil de déclenchement du signal digital est réglable via un potentiomètre ajustable.

Le raccordement est sur une entrée digitale dans notre carte Arduino Méga (voir la figure 3.29).

FIGURE 3.28: Module suiveur de ligne GT1140

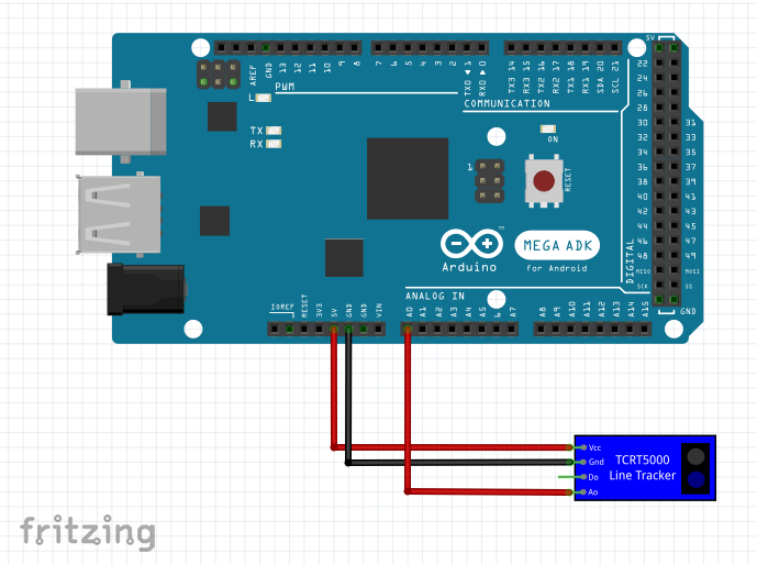

FIGURE 3.29: Connexion du module suiveur de ligne GT1140

Caractéristique suiveur de ligne GT1140

- Alimentation : 3,3 à 5 Vcc.
- Consommation : 20 mA sous 5 V.
- Sortie :
  - état bas : ligne noire.
  - état haut : ligne blanche.
- T° de service : 0 à +50 °C.
- Dimensions : 42 x 11 x 12 mm. [24]

Ultrason HC-SR04

Pour que le robot AGV doit être capable d'évité les obstacles au cours de son chemin nous avons besoin d'un capteur à ultrasons HC-SR04(voir la figure 3.30) ce capteur est capable de mesurer des distances d'objets de 2 cm à 400 cm, La précision du capteur est de 3mm. Le capteur se compose d'un émetteur à ultrasons, d'un récepteur et un circuit de commande.

Comment ça fonctionne :

1. Envoyer un signal numérique haut à l'émetteur pendant 10 μs.
2. Le capteur envoie automatiquement 8 impulsions ultrasonores de 40 kHz et détecte le signal qui revient.
3. Si le signal revient, la durée du niveau haut du signal reçu correspond au temps entre transmission et réception d'ultrasons.

   Calcul de la distance : distance = (durée du signal reçu élevé * vitesse du son) / 2 (vitesse du son dans l'air : 340 m/s).[9]

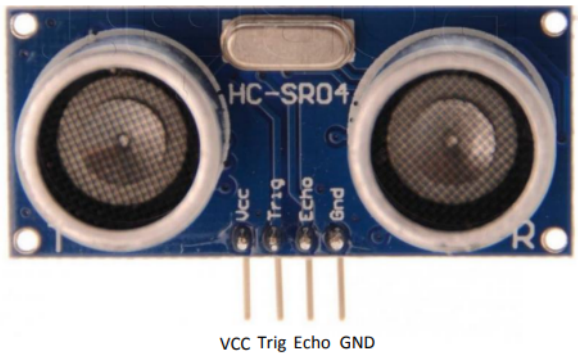

FIGURE 3.30: Capteur Ultrason HC-SR04

La figure 3.31 représente la connexion de capteur Ultrason dans notre projet :

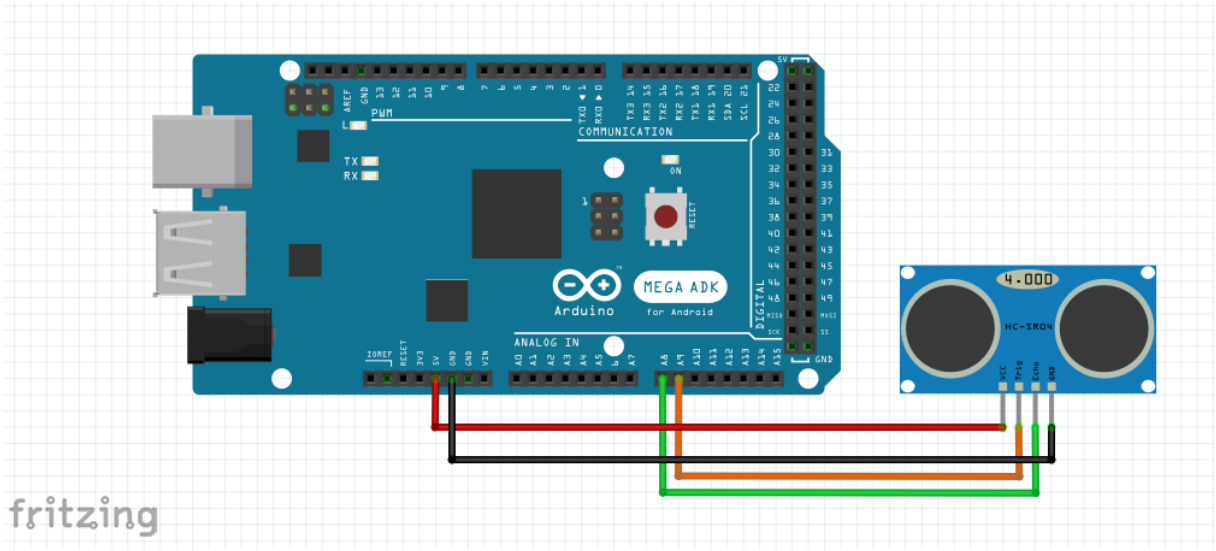

FIGURE 3.31: Connexion du capteur Ultrason HC-SR04

Afficheur LCD

Les écrans à cristaux liquides sont des modules intelligents et compacts qui nécessitent peu de composants externes pour fonctionner correctement. Ils consomment relativement peu (de 1 à 5 mA), sont relativement bon marché et sont très simples d'utilisation.

Il existe plusieurs types d'afficheurs sur le marché, qui se différencient non seule-

ment par leur taille (de 1 à 4 lignes, 6 à 80 caractères par ligne), mais aussi par leurs caractéristiques techniques et leur tension de fonctionnement. Ils sont largement utilisés dans les composants de microcontrôleurs et présentent une excellente convivialité. Ils peuvent également être utilisés pendant la phase de développement d'un programme, car nous pouvons facilement afficher les valeurs de différentes variables.

Dans notre cas nous allons afficher l'état de batterie et la charge placée sur notre AGV (voir la figure 3.32). [7]

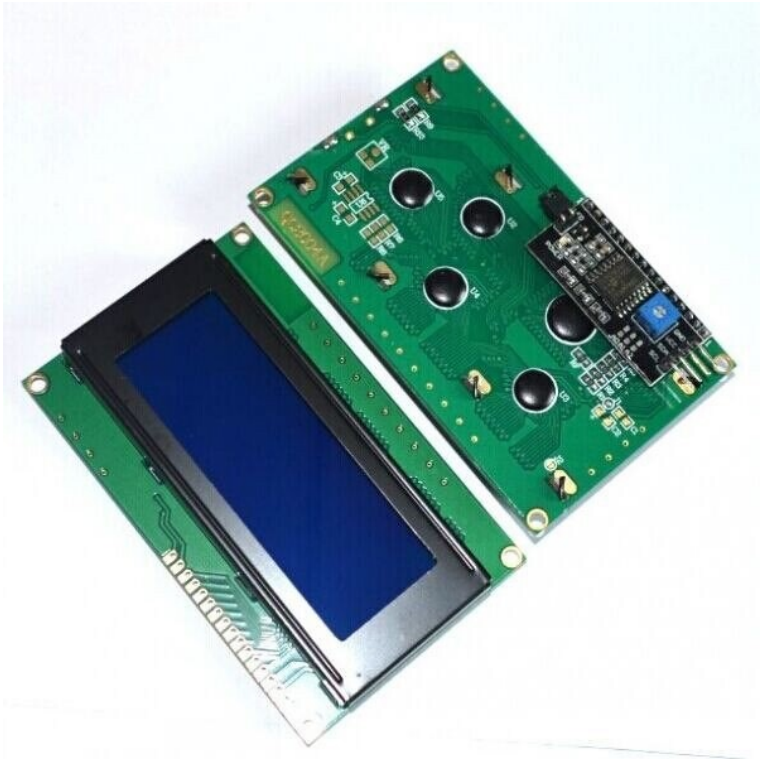

FIGURE 3.32: Ecran LCD 20x4

La figure 3.33 représente la connexion d'écran LCD avec Arduino méga dans notre projet :

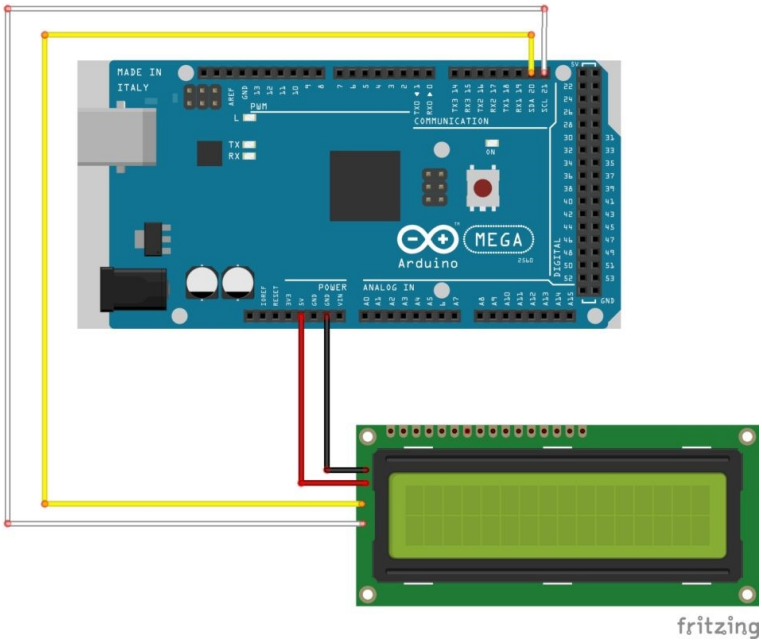

FIGURE 3.33: Connexion d'écran LCD avec Arduino méga

Capteur de poids

Un capteur de force (ou cellule d'effort) est un dispositif utilisé pour convertir une force (par exemple un poids) appliquée sur un objet en signal électrique.

Le capteur est généralement construit en utilisant des jauges de déformation (Strain gauge) connectée en un pont approprié.

Un amplificateur HX711 est nécessaire pour lire le signal délivré par le transducteur. Le but des extensomètres à fils résistants ou jauges de déformation (jauges de contrainte) est de traduire la déformation d'une pièce en variation de résistance électrique (plus les extensomètres s'étirent, plus leurs résistances augmentent).

Montage

Nous souhaiterons accrocher la cellule de charge entre deux plaques en forme de "Z", avec des vis et des entretoises de montage afin que la contrainte puisse être correctement mesurée comme indiqué dans la figure 3.34 :

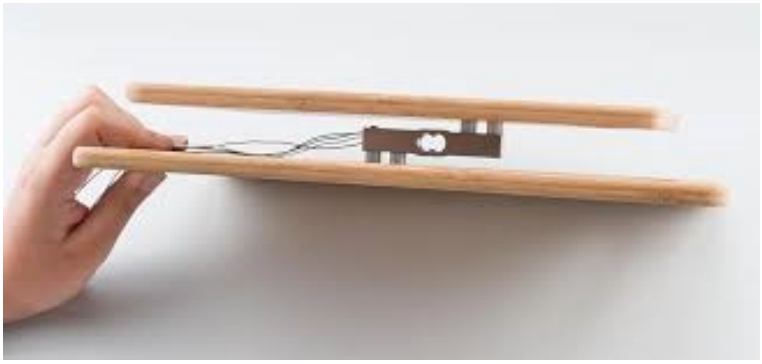

FIGURE 3.34: Montage de capteur de poids HX711

Notons qu'un seul côté de la cellule de charge est vissé dans chaque carte. Cela fournit un moment de force, ou couple, sur la jauge de contrainte plutôt qu'une simple force de compression, ce qui est plus facile à mesurer et beaucoup plus précis. [4]

Caractéristique

- Alimentation : 5 Vcc (via le module HX711).
- Plage de mesure : 0 à 20 kg.
- Précision : 0,5 % de la pleine échelle
- Brochage :
  - Rouge : E+
  - Noir : E-
  - Vert : A+
  - Blanc : A-
- Température de fonctionnement : -10 à +40 °C
- Dimensions : 56 x 13 x 13 mm

Pont diviseur de tension

Pour afficher l'état de notre batterie il faut le brancher dans un port analogique pour récupérer la tension de sortie, mais le problème c'est que l'arduino ne peux pas

supporter une tension plus que 5v et notre batterie est de tension de 12v, pour cela le pont diviseur est la méthode idéale pour récupérer la tension souhaité.

Le pont diviseur est formé de deux résistances dont les valeurs déterminent la tension de sortie (voir la figure 3.35).

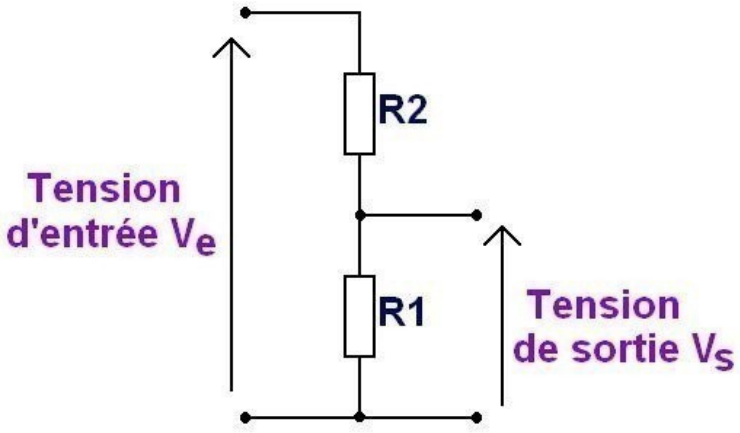

FIGURE 3.35: Montage pont diviseur de tension

Ve = tension d'entrée

Vs = tension de sortie

La tension de sortie Vs vaut :

$$Vs = Ve.\frac{R1}{R1 + R2} \tag{3.18}$$

La tension de sortie ne peut jamais être supérieure à la tension d'entrée (en valeur absolue) : le montage atténue toujours et n'amplifie jamais.

3.9.2.3 Unité de gestion de mouvement des moteurs

Pour déplacer le robot, deux moteurs à courant continu sont utilisés. Dans notre projet, Nous sommes limités au contrôle des moteurs à courant continu les plus courants.

Choisissons le contrôleur de moteur à double pont en H 7A/160W pour contrôler le moteur.

Pour cela, il existe deux techniques utilisé :

- Sens de rotation contrôlée via pont en H.
- Contrôler la puissance (et donc la vitesse) via la modulation d'amplitude du signal (PWM ou modulation de largeur d'impulsion)

Drive moteur $7A/160W$

Il s'agit d'un pilote de moteur CC double à profil bas ultra-compact pour les projets à contraintes d'espace, capable de fournir une puissance jusqu'à $7A$ par canal de sortie. Il utilise une logique similaire à celle du pilote de moteur L298, où nous contrôlons le pilote avec 3 broches de signal (IN1, IN2, ENABLE) (voir la figure 3.36 et 3.37).

Ce pilote de moteur est entraîné par un MOSFET haute puissance, avec l'optocoupleur des signaux de commande est isolé pour protéger les circuits délicats et les problèmes de boucle de masse. Nous pouvons piloter ce drive avec une logique $3.3V$ et $5V$. [1]

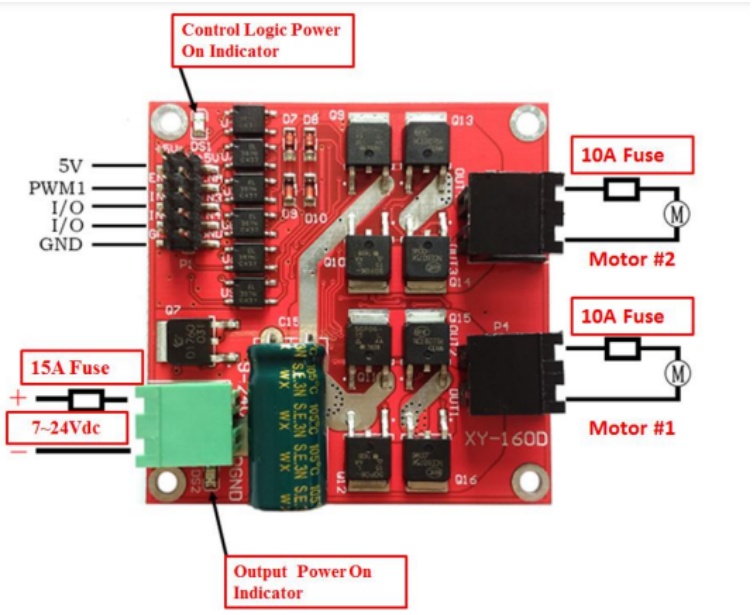

FIGURE 3.36: Contrôleur de moteur à double pont en H 7A/160W

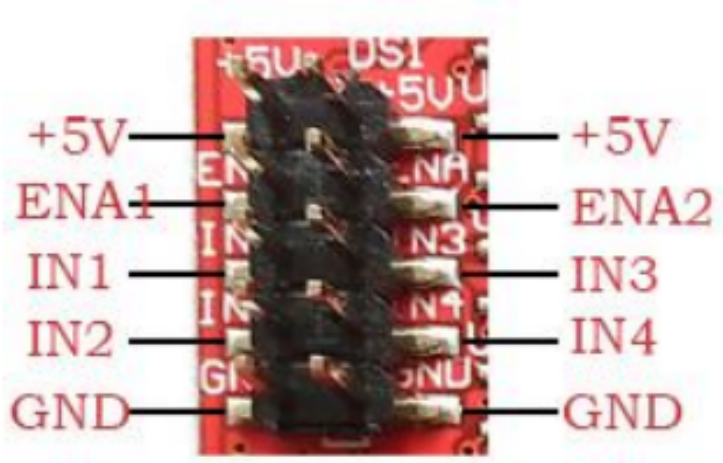

FIGURE 3.37: Fonction logique de contrôle

Tableau logique de contrôle

Le tableau 3.4 représente le tableau logique de contrôle :

IN1	IN2	ENA	OUT1-OUT2
0	0	X	Freinage moteur
1	1	X	Flottant
1	0	PWM	avant + contrôle de vitesse
0	1	PWM	inverse + contrôle de vitesse
1	0	1	Avance à pleine vitesse
0	1	1	Marche arrière à pleine vitesse

TABLE 3.4: Tableau logique de contrôle

Caractéristiques

- Tension d'alimentation : 7 $\sim$ 24 VCC. (Limite : 6,5 $\sim$ 27 VDC).
- Niveau du signal de contrôle (Compatible $3.3V/5V$).
  - Logique élevée (H) : CC 3,0 $\sim$ 6,5 V

— Logique bas (L) : DC 0 $\sim$ 0,8 V

- Canal de sortie : 2.
- Courant du signal de commande : 3 $\sim$ 11 mA (chaque route).
- Courant de fonctionnement continu maximum : 7 A.
- Courant de crête : 50 A.
- Contrôle de vitesse PWM : 0 $\sim$ 10 KHz.
- Largeur d'impulsion minimale valide : 5 us.
- Température de fonctionnement : -25 $\sim$ 85 °C. [1]

Pont en H

Un pont en H est une structure électronique utilisée pour contrôler la polarité d'un terminal dipôle, généralement composé de quatre éléments de commutation. Un schéma en forme de H comme illustré à la figure 3.38 Les commutateurs peuvent être des relais, des transistors ou d'autres éléments de commutation, selon l'application prévue.

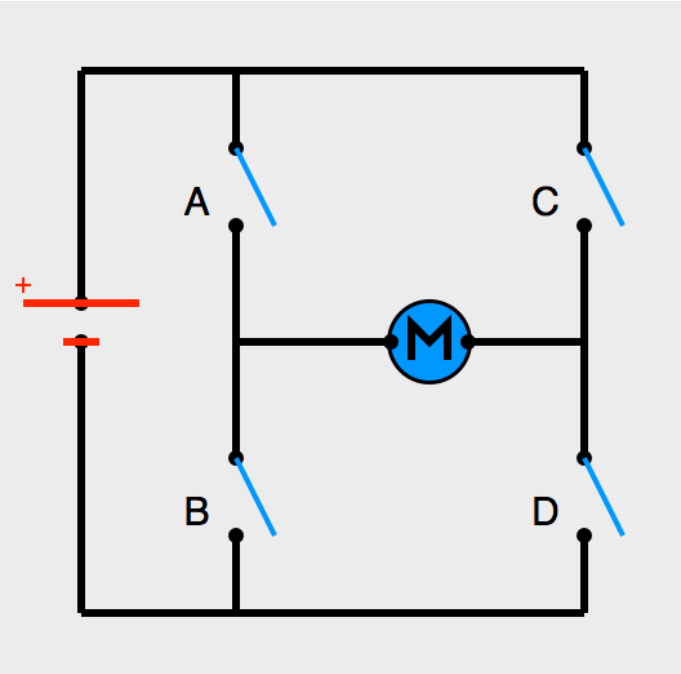

FIGURE 3.38: Le pont en H

Voyons maintenant ce qui se passe lorsque nous activons l'interrupteur A et D en même temps (Schéma gauche), ou les interrupteurs B et C (schéma droite) selon la Figure 3.39.

Dans l'image de gauche, les interrupteurs A et D sont fermés, donc le courant entre par la jambe gauche du moteur et sort par la jambe droite, de sorte que le moteur tourne. Sur la photo de droite, les interrupteurs B et C sont fermés, donc le courant entre par la jambe droite du moteur et passe à travers son côté gauche fait tourner le moteur dans le sens opposé.

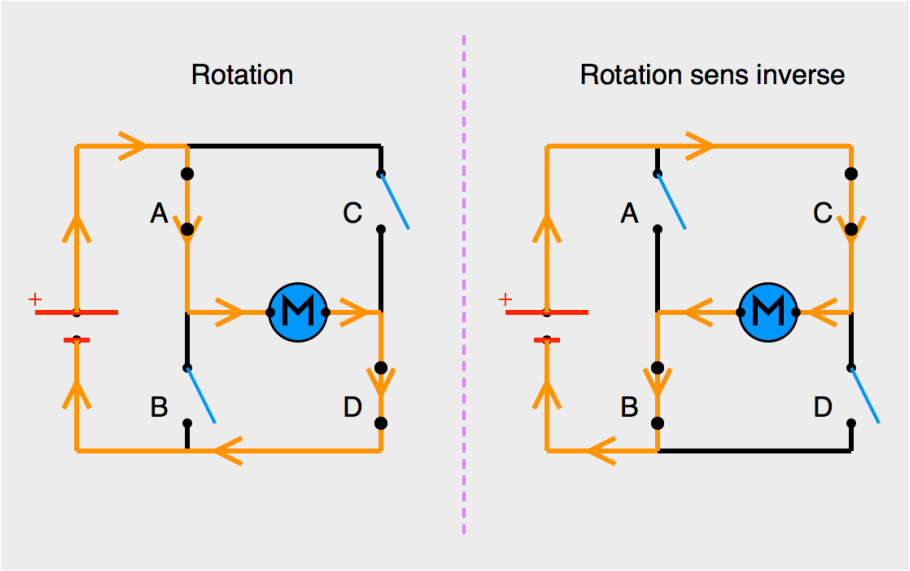

FIGURE 3.39: Sens du courant en fonction de l'état des interrupteurs d'un pont en H.

3.10 La programmation

3.10.1 Définition du programme

Le robot reçoit la commande et travaille automatiquement en fonction de ces dernières. Il vérifie s'il y a une action pour la trajectoire, s'il y en a, il se déplacera dans cette direction Il avance jusqu'à ce qu'il trouve un obstacle ou reçoive une autre commande pour marcher vers la droite ou à gauche, le logiciel qui permet de programmer la carte Arduino

s'appelle Arduino IDE.

3.10.2 Le montage de notre travail

Le schéma éléctrique globale de notre AGV est illustrée dans la figure 3.40 :

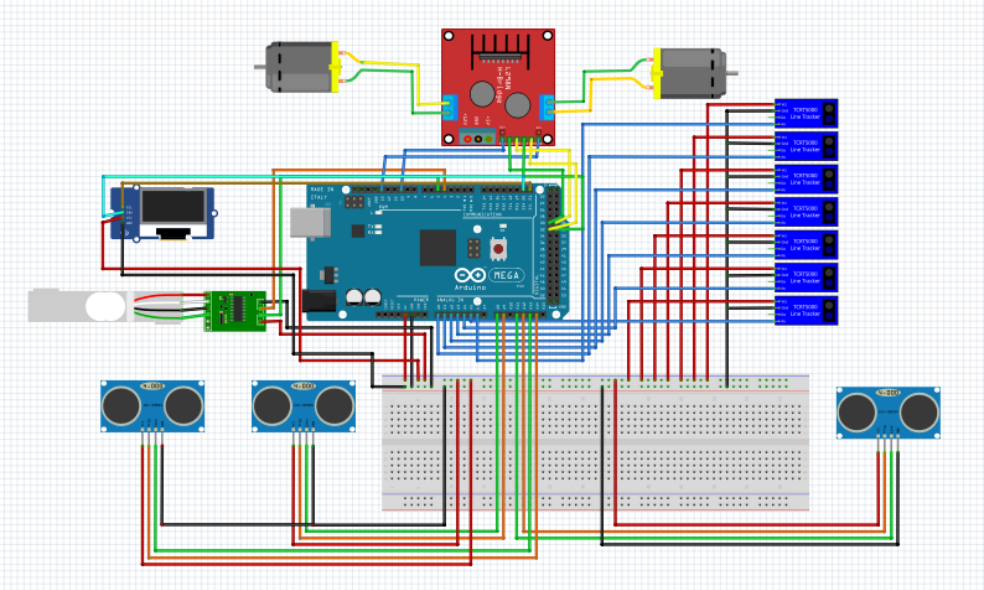

FIGURE 3.40: Schéma éléctrique globale de notre AGV.

Dans la figure 3.41 montre le montage de notre robot AGV suiveur de ligne.

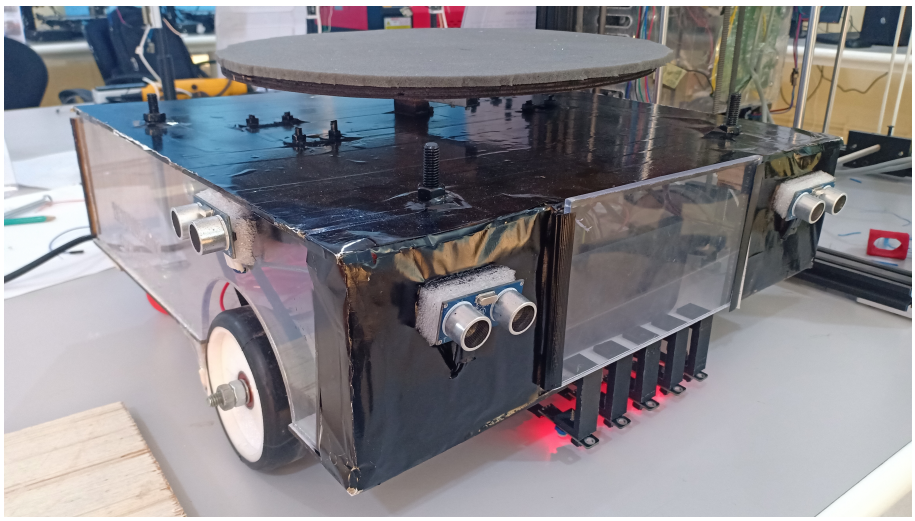

FIGURE 3.41: Montage de notre robot AGV suiveur de ligne.

3.10.3 Le teste de robot AGV suiveur de ligne

Après la finalisation du montage, nous avons testé notre robot dans le FABLAB (voir les figures 3.42 et 3.43 et 3.44 et 3.45 et 3.46 et 3.47 et 3.48 et 3.49).

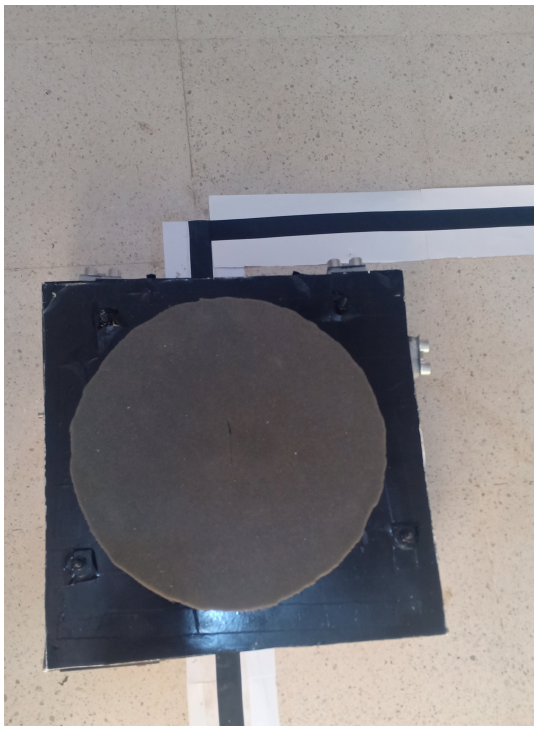

FIGURE 3.42: Rotation droite 90°.

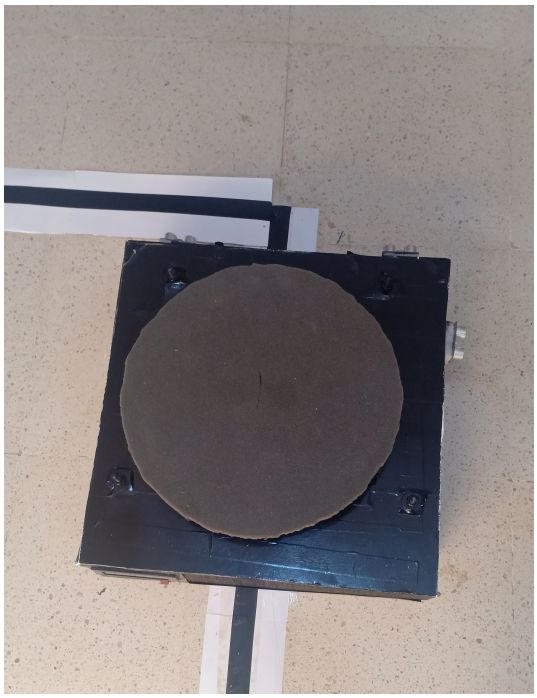

FIGURE 3.43: Rotation gauche 90°.

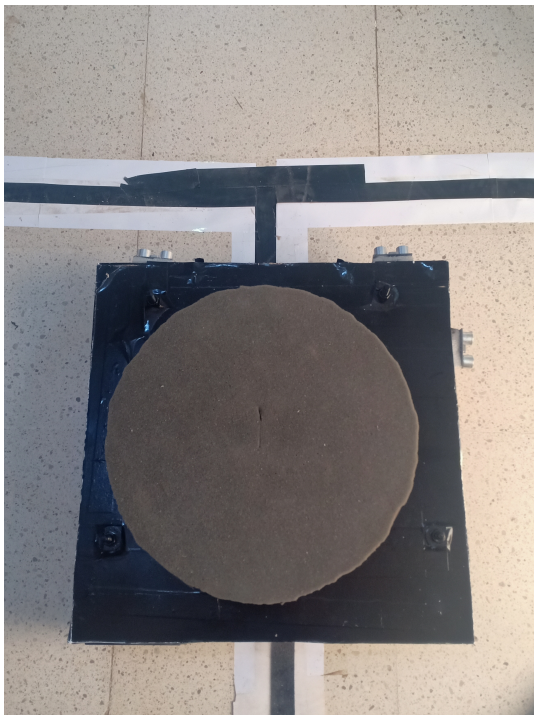

FIGURE 3.44: Intersection T, Rotation gauche 90°.

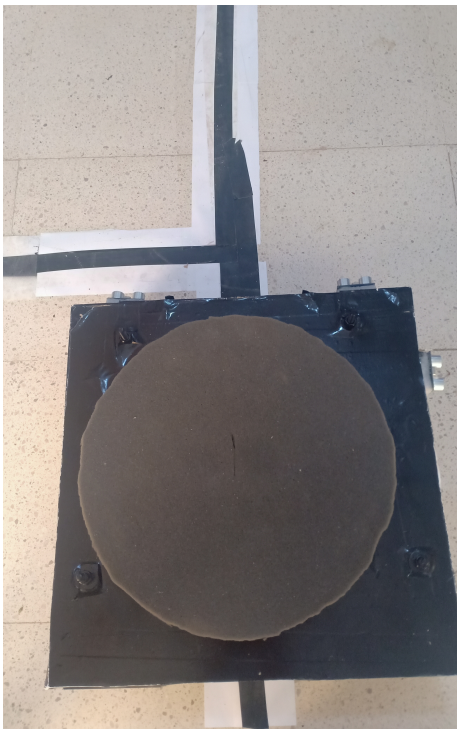

FIGURE 3.45: Rotation gauche 90°

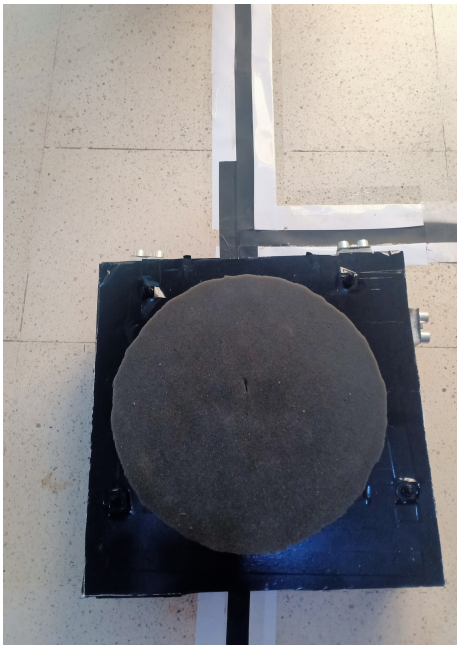

FIGURE 3.46: Marche en avant

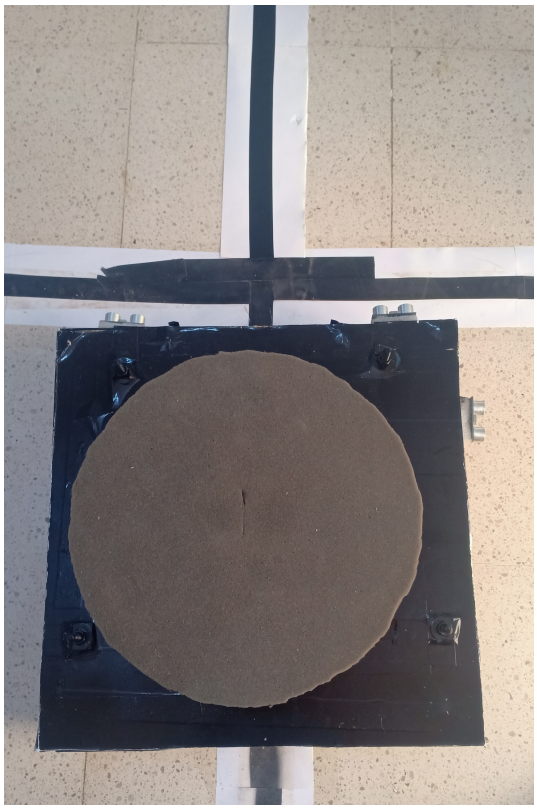

FIGURE 3.47: Intersection +, Rotation gauche 90°

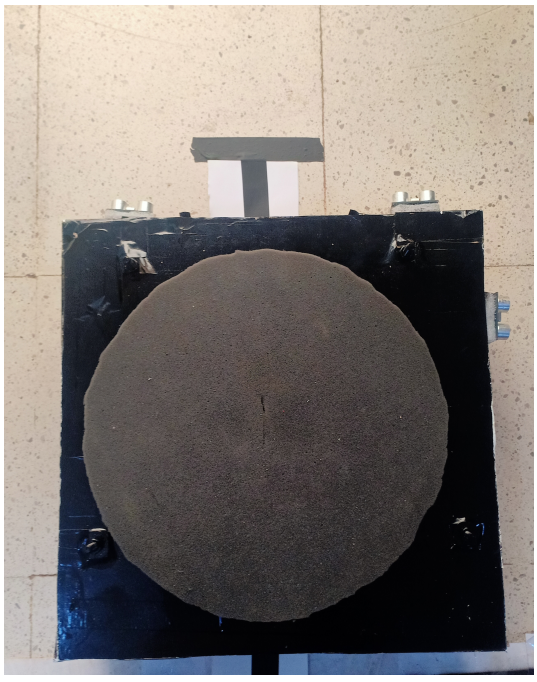

FIGURE 3.48: Demi-tour

FIGURE 3.49: Affichage d'état de batterie et le poids

C1	C2	C3	C4	C5	C6	C7	État du robot
0	0	0	0	0	0	0	Marche en avant
0	0	1	1	1	0	0	Rotation 180°
1	0	0	0	0	0	0	Rotation gauche 90°
0	1	0	0	0	0	0	Marche à gauche +
0	1	1	0	0	0	0	Marche à gauche +
0	0	1	0	0	0	0	Marche à gauche
0	0	1	1	0	0	0	Marche à gauche
0	0	0	1	0	0	0	Marche en avant
0	0	0	1	1	0	0	Marche à droit
0	0	0	0	1	0	0	Marche à droit
0	0	0	0	1	1	0	Marche à droit +
0	0	0	0	0	1	0	Marche à droit +
0	0	0	0	0	0	1	Rotation droit 90°
0	1	1	1	0	0	0	Rotation gauche 90°
0	0	0	1	1	1	0	Marche en avant
0	1	1	1	1	1	0	Rotation gauche 90°

TABLE 3.5: Les différents états des capteurs.

3.11 Conclusion

Dans ce chapitre nous avons présenté les étapes de la réalisation de notre robot. Cette réalisation est constituée en trois étapes :

La première étape consiste à construire une structure mécanique qui est composée d'un châssis en résine, conception et impression des roues et des poulies sous SolidWorks, et les moteurs.

La deuxième partie nous avons identifié les structures de la partie électronique,

La troisième partie est la programmation.

D'après le test, nous pouvons conclure que le robot exécute notre commande avec succès.

Conclusion générale

La robotique peut être définie comme l'ensemble des techniques et études tendant à concevoir des systèmes mécaniques, informatiques, électroniques ou mixtes, capables de se substituer à l'homme dans ses fonctions motrices, sensorielles et intellectuelles.

Ce projet nous a permis de découvrir le monde de la robotique : la mécanique, systèmes embarqués et la programmation. De plus, le travail effectué a été une occasion pour nous d'appliquer le langage C pour la programmation des microcontrôleurs. Avec la conception et la réalisation pratique, l'apprentissage était très intéressant.

Nous avons donc pu réaliser l'objectif de notre projet qui est l'utilisation du logiciel et la carte électronique Arduino pour manipuler un robot AGV à quatre roues en utilisant des capteurs de type infrarouge (GT1140), des capteurs de distance ultason (HC-SR04) et capteur de poids (HX711). Nous avons étudié tout d'abord les principales caractéristiques d'un robot et en particulier le robot AGV. Ensuite nous avons présenté les différents composants utilisés pour construire ce robot ainsi que les étapes de fabriquation suivies pour concevoir, réaliser et tester chaque partie du robot.

Après la finalisation du montage, nous avons testé notre robot dans le FABLAB à l'ESSAT, et d'après le test on peut conclure que notre robot AGV suit la trajectoire avec succès.

Pour conclure, ce projet nous a permis, d'une part, d'améliorer nos compétences scientifiques en électronique, capteurs, régulation et du côté programmation en langage

C pour l'Arduino. D'autre part ce projet nous a permis de travailler en groupe que ce soit pour la répartition du travail, la programmation du robot ou résoudre les nombreux problèmes rencontrés lors du procerssus de fabriquation du robot, parmis ces problèmes le retour d'information sur les positions du moteur afin d'effectuer une régulation sur la vitesse de déplacement du robot et l'instabilité de l'allimentation due à la qualité de la batterie.

Pour les perspectives de ce projet, nous pouvons changé les moteurs utilisés par des moteurs dotés d'un encodeur pour pouvoir faire la régulation de la vitesse en fonction de la variation du poids et la tension de la batterie, aussi nous proposons d'améliorer le type de la batterie et intégrer un sous système électronique pour la régulation de l'alimentation du robot. Nous prévoyons aussi d'autonomiser notre réalisation en passant d'un robot AGV à un robot AMR (Autonomous Mobile Robot) par l'intégration de l'intelligence artificielle, ceci exige le changement de la partie processeur en intégrant un processeur doté d'une capacité élevé pour voir exécuter des programmes plus performent comme les algorithmes d'intelligence artificielle et gérer plus de capteurs tel que les caméras. Les possibilités d'optimisation du robot sont nombreuses pour cela nous restons optimistes à une eventuelle amélioration de notre projet.

L'avenir de ce projet représente l'avenir de notre pays dans le domaine de la robotique et l'automobile électrique, car nous prévoyons de continuer dans ce domaine afin de développer tout une industrie robotique en améliorants ces fonctionnalités et faire quelques transformations pour s'ouvrir à d'autre domaines tels que :

— Le spatial : Les robots d'exploration spatial.

— Protection civile : les robots Sentinelle d'assistance technique et de lutte contre l'incendie.

Bibliographie

[1] *7A/160W Dual H Bridge Motor Controller - Handson Tec*. URL : https://www.handsontec.com/dataspecs/module/7A-160W%5C%20motor%5C%20control.pdf.

[2] Mohamed Nezar Abourraja. "Gestion multi-agents d'un terminal à conteneurs". Thèse de doct. Normandie Université, 2018.

[3] Ahmed Akili. "etat de l'art sur la robotique et solution adoptée". In : *Conception et réalisation d'un robot mobile sans Fil À base d'arduino*. Édition universitaires européennes, 2017, p. 5-7.

[4] Alldatasheet.com. *HX711 PDF, HX711 description, HX711 datasheet, HX711 View ::: Alldatasheet :::* URL : https://pdf1.alldatasheet.com/datasheet-pdf/view/1371088/SPARKFUN/HX711.html.

[5] Moustapha Ahmed Bouh. *Conception d'entrepôt: Sélection des équipements de manutention et d'entreposage*. Ecole Polytechnique, Montreal (Canada), 2017.

[6] Samir BOUKHARI. "REALISATION ET GUIDAGE D'UN AGV". Thèse de doct. Université Mohamed Boudiaf-M'Sila, 2007.

[7] Cyril BRUNAUD, Alexis FOURRE et Thierry LEQUEU. "Carte électronique avec afficheur LCD". In : ().

[8] Noel Camara, Mamadou Saliou Dile et M Nakechbandi. “Implémentation en Java et Simulation via Anylogic, d’algorithmes de re-routage de camions pour aller chercher des conteneurs frigorifiques vides”. In : ().

[9] *Capteur à ultrasons HC-SR04 HCSR04*. URL : http://www.robot-maker.com/shop/img/cms/datasheet-capteur-ultrasons-hc-sr04.pdf.

[10] *cours régulation industrielle*. 2021.

[11] Vincent Delhaye et Marcel Gérard. “Le plus court chemin d’imposition des multinationales: application de l’algorithme de Dijkstra”. Thèse de doct. Master’s thesis, Louvain School of Management, Université catholique de . . ., 2015.

[12] Johann Dréo. “Adaptation de la métaheuristique des colonies de fourmis pour l’optimisation difficile en variables continues. Application en génie biologique et médical.” Thèse de doct. Université Paris XII Val de Marne, 2003.

[13] Julien POLO Duong-Khang NGUYEN. “Recherche du plus court chemin”. In : ().

[14] Camilla Feledy et Mark Schiller Luttenberger. “A State of the Art Map of the AGVS Technology and a Guideline for How and Where to Use It”. In : (2017).

[15] Zoltán Kása. “Warshall’s algorithm–survey and applications”. In : *arXiv preprint arXiv:1905.00276* (2019).

[16] Liam Lynch et al. “Automated ground vehicle (agv) and sensor technologies-a review”. In : *2018 12th International Conference on Sensing Technology (ICST)*. IEEE. 2018, p. 347-352.

[17] Pape Adama Mboup et al. “Optimisation de l’utilisation de l’algorithme de Dijkstra pour un simulateur multi-agents spatialisé”. In : *2015 World Congress on Information Technology and Computer Applications (WCITCA)*. IEEE. 2015, p. 1-6.

[18] Ata Jahangir MOSHAYEDI, Li JINSONG et Liefa LIAO. "AGV (automated guided vehicle) robot: Mission and obstacles in design and performance". In : *Journal of Simulation and Analysis of Novel Technologies in Mechanical Engineering* 12.4 (2019), p. 5-18.

[19] Sandra PERMANN. "Automated Guided Vehicles and Autonomous Mobile Robots in Hospitals". Thèse de doct. Wien, 2021.

[20] Z SARI. "Modélisation, analyse et évaluation des performances d'un AS/RS à convoyeur gravitationnel". Thèse de doct. Thèse de Doctorat d'état, Université de Tlemcen, Algérie, 2003.

[21] Christine SOLNON. "Théorie des graphes et optimisation dans les graphes". In : *INSA de Lyon* ().

[22] DJELLAT SOULIMANE. "Optimisation Par Colonie de Fourmies". In : (2012-2013).

[23] Chahinaz et TAHANI. "COMMANDE D'UN CONVERTISSEUR DC/DC PAR UN MICRO CONTROLEUR". Thèse de doct. 2021.

[24] Go TRONIC. *Module suiveur de ligne GT1140*. URL : https://www.gotronic.fr/art-module-suiveur-de-ligne-gt1140-26142.htm.

[25] Examinateur Mr BELDJILALI Bouziane Professeur UO. "Application des algorithmes de colonies de fourmis pour l'optimisation et la classification des images". Thèse de doct. UNIVERSITE DES SCIENCES ET DE LA TECHNOLOGIE d'Oran, 2013.

[26] Chen WANG et Jian MAO. "Summary of agv path planning". In : *2019 3rd International Conference on Electronic Information Technology and Computer Engineering (EITCE)*. IEEE. 2019, p. 332-335.

[27] *XD-3420 12V/24V 30W CW/CCW moteur electrique réversible moteur à courant continu d'aimant permanent de Haute Vitesse et bruit faible (12V) : Amazon.fr: Commerce, industrie et science*. URL : https://www.amazon.fr/XD-3420-Electrique-R%5C%C3%5C%A9versible-Courant-Permanent/dp/B0787XGGM7.

[28] Fengjia YAO et al. "Optimizing the scheduling of autonomous guided vehicle in a manufacturing process". In : *2018 IEEE 16th International Conference on Industrial Informatics (INDIN)*. IEEE. 2018, p. 264-269.

[29] Lixiang ZHANG, Yaoguang HU et Yu GUAN. "Research on hybrid-load AGV dispatching problem for mixed-model automobile assembly line". In : *Procedia CIRP* 81 (2019), p. 1059-1064.