République **A**lgérienne **D**émocratique et **P**opulaire
Ministère de l'**E**nseignement **S**upérieur et de la **R**echerche **S**cientifique

UNIVERSITE **M**OULOUD **M**AMMERI DE **T**IZI-**O**UZOU

FACULTE DU **G**ENIE **E**LECTRIQUE ET D' **I**NFORMATIQUE
DEPARTEMENT D'AUTOMATIQUE

Mémoire de Fin d'Etudes de MASTER ACADÉMIQUE

Domaine : **Sciences et Technologies**
Filière : **Automatique**
Spécialité : **Automatique et Système**

Présenté par
Youcef BOUGHAZI

Thème

Implémentation d'une commande floue à une architecture de télé opération position-position en présence d'un retard fixe de transmission

Mémoire soutenu publiquement le 01/10/ 2024 devant le jury composé de :

Mme Nadia DJEGHALI
Professeur, Université Mouloud MAMMERI de Tizi-Ouzou, Président

M Rabah MELLAH
Professeur, Université Mouloud MAMMERI de Tizi-Ouzou, Encadrant

Mme Nacera ARAR
Professeur, Université Mouloud MAMMERI de Tizi-Ouzou, Examinateur

Mme Karima AMOURA
Professeur, Université Mouloud MAMMERI de Tizi-Ouzou, Examinateur

République **A**lgérienne **D**émocratique et **P**opulaire
Ministère de l'**E**nseignement **S**upérieur et de la **R**echerche **S**cientifique

UNIVERSITE **M**OULOUD **M**AMMERI DE **T**IZI-**O**UZOU

FACULTE DU **G**ENIE **E**LECTRIQUE ET D' **I**NFORMATIQUE
DEPARTEMENT D'AUTOMATIQUE

Mémoire de Fin d'Etudes de MASTER ACADÉMIQUE

Domaine : **Sciences et Technologies**
Filière : **Automatique**
Spécialité : **Automatique et Informatique Industrielle**

Présenté par
Anis MERABET

Thème

Implémentation d'une commande floue à une architecture de télé opération position-position en présence d'un retard fixe de transmission

Mémoire soutenu publiquement le 01/10/ 2024 devant le jury composé de :

Mme Nadia DJEGHALI
Professeur, Université Mouloud MAMMERI de Tizi-Ouzou, Président

M Rabah MELLAH
Professeur, Université Mouloud MAMMERI de Tizi-Ouzou, Encadrant

Mme Nacera ARAR
Professeur, Université Mouloud MAMMERI de Tizi-Ouzou, Examinateur

Mme Karima AMOURA
Professeur, Université Mouloud MAMMERI de Tizi-Ouzou, Examinateur

Remerciements

Nous remercions tout d'abord Dieu Tout-Puissant de nous avoir donné la santé et le courage nécessaires pour mener à bien ce projet de fin d'études dans les meilleures conditions.

Nous tenons également à exprimer nos sincères remerciements à notre promoteur, Monsieur Rabah MELLAH, pour ses précieux conseils, son soutien constant, ainsi que son expertise et sa disponibilité, qui ont été essentiels à la réussite de ce travail.

Nos remerciements s'adressent également aux membres du jury, pour l'honneur qu'ils nous ont fait en acceptant d'examiner et d'évaluer notre travail.

Nous remercions chaleureusement tous ceux et celles qui ont contribué à l'accomplissement de ce modeste projet.

Merci

Dédicaces :

Je dédie ce travail :

.À ma mère, qui a toujours été à mes côtés depuis le début de mes études. Son soutien inconditionnel, son amour et sa patience ont été une source inestimable de motivation et de force.

.A mon père pour ses conseils, sa sagesse et ses encouragements, dont la présence discrète mais essentielle a grandement contribué à ma réussite.

.A mon frère, mon bras droit, pour son soutien continu, qui m'a toujours apporté réconfort et motivation.

.À la mémoire de mes grands-parents que j'aime et qui ont toujours occupé une place spéciale dans mon cœur.

.À mes précieux amis et camarades,

.À tous ceux qui m'ont aidé, encouragé et accompagné tout au long de mes études,

.À mon ami et binôme Anis, sans qui ce mémoire n'aurait pu être réalisé.

BOUGHAZI Youcef

Dédicaces :

Je dédie ce travail :

.À ma mère, pour son amour et son soutien indéfectible,
.À la mémoire de mon père, dont les valeurs continuent de me guider,
.À ma sœur et à mon frère, pour leur présence constante à mes côtés,
.À mes chers grands-parents, pour leurs encouragements et leurs prières,
.À mes petites nièces, qui apportent joie et motivation à ma vie,
.À mes précieux amis et camarades,
.À tous ceux qui m'ont aidé, encouragé et accompagné tout au long de mes études,
.À mon ami et binôme Youcef, sans qui ce mémoire n'aurait pu être réalisé.

MERABET Anis

Notations

Acronymes

MIT :	Massachusetts Institute of Technology
NASA :	National Aeronautics and Space Administration.
AVR :	Automatic Voltage Regulator
IDE :	Integrated Development Environment
ATMEL :	Advanced Technology for Memory and Logic
PWM :	Pulse Width Modulation
ARM :	Advanced RISC Machines
UARTs :	Universal Asynchronous Receiver / Transmitter
USBOTG	:Universal Serial Bus On-The-Go
DAC :	Digital to Analogue Converter,
TWI :	Two-Wire Interface
SPI :	Serial Peripheral Interface
JTAG :	Joint Test Action Group
USB :	Universal Serial Bus
AC :	Alternating Current
DC :	Direct Current
GND :	Ground
IOREF :	Input-Output voltage REFerence
ROM :	Read-Only Memory
SRAM :	Static Random Access Memory
RAM :	Random Access Memory
MCU :	Microcontroller Unit
SPL :	Serial Peripheral Interface
CAN :	Controller Area Network
SDA :	Serial Data Pin

SCL : Serial Clock Pin

ADC : Analog-to-Digital Converter

AREF : Analogue REFerence

BR1 : Bridge Recitifier

TTL : Transistor-Transistor Logic family

CDC : Communications Device Class

Table des matières

Table des figures

Liste des tableaux

Introduction générale

La manipulation des produits dangereux a été à l'origine de la téléopération. En 1949, **Goertz** a déposé un brevet pour un manipulateur maître-esclave, qui est considéré comme l'un des premiers systèmes de téléopération[20]. Ce dispositif était conçu pour manipuler des matériaux radioactifs en toute sécurité, éloignant ainsi les opérateurs des dangers potentiels.

Ces dernières années, avec le développement rapide des technologies informatiques, des technologies de communication électrique et des technologies de contrôle, le système de téléopération intégrant ces technologies a attiré de plus en plus d'attention . Le système de téléopération permet aux opérateurs humains d'intervenir dans des environnements dangereux ou hostiles, et permet d'étendre les capacités de perception et de manipulation humaines à un environnement distant[1]. De manière générale, un système de téléopération typique est composé d'un opérateur humain, d'un robot maître, d'un canal de communication, d'un robot esclave et d'un environnement.

Dans un système de téléopération, le contrôle bilatéral fournit des informations de retour de force à l'opérateur humain via un robot maître. D'autre part, un robot esclave interagit avec des environnements inconnus. Les robots maître et esclave sont couplés par un canal de communication, où les informations de position, de vitesse et de force sont transférées. Sur commande de l'opérateur humain, le robot esclave situé à un emplacement distant suit le mouvement du robot maître. Afin d'améliorer les performances de la tâche, un retour de force du robot esclave vers le robot maître est nécessaire. Si les informations circulent dans les deux sens entre le maître et l'esclave, le système de téléopération est dit contrôlé de manière bilatérale. La téléopération bilatérale est un sujet difficile des technologies de contrôle actuelles avec un certain nombre d'applications traditionnelles et potentielles, telles que l'exploration spatiale et sous-marine, la manipulation d'objets dangereux .

En présence de retards, il y a un délai entre le moment où l'opérateur réalise une action (par exemple un déplacement du robot maître) et le moment où il observe l'effet de son action (par exemple, le moment où il voit le robot esclave bouger). La stratégie

généralement adoptée est la stratégie dite « action attente » (''move and wait strategy '') : l'opérateur réalise une succession de petits déplacements et attend d'observer le résultat de l'action précédente pour passer à l'action suivante. Il en résulte une augmentation très importante du temps de réalisation d'une tâche. Ce phénomène souligne l'un des principaux défis de la téléopération, à savoir le temps de retard dans les communications, qui tend à déstabiliser le système lorsque l'on vise un couplage fort entre le site maître et le site esclave. .

Dans ce travail, nous proposons un contrôle flou pour commander un système de téléopération bilatérale à un degré de liberté, en utilisant une carte Arduino. Ce système doit permettre une commande intuitive du robot téléopéré tout en reproduisant plus fidèlement possible les mouvements de robot maître. .

Le présent mémoire est structuré comme suit :

Le premier chapitre fournit une description générale des systèmes de téléopération, en exploitant leur analyse à travers l'évaluation des deux critères de performances (stabilité et transparence). Il présente également les architectures de contrôle les plus utilisées en pratique et leurs performances.

Le deuxième chapitre est consacré à l'étude de la logique floue. Nous y expliquerons les notions de base de la théorie des ensembles flous ainsi que les outils mathématiques nécessaires à leur manipulation. Nous présenterons également la structure générale d'un système flou et les différentes étapes du raisonnement flou.

Le troisième chapitre se concentrera sur la partie Hardware et Software de notre projet, principalement la carte Arduino, qui assure la communication entre la partie commande et la partie opérative.

Le quatrième chapitre sera dédié à l'application pratique d'une stratégie de commande floue pour contrôler le système de téléopération que nous avons développé. Nous discuterons des matériaux utilisés, des simulations réalisées et des résultats expérimentaux obtenus, en les interprétant pour évaluer les performances de poursuite et l'efficacité de la stratégie de commande floue mise en place.

Le mémoire se termine par une conclusion générale sur l'ensemble de l'étude présentée.

Chapitre 1

Description d'une architecture de télé opération position position en présence d'un retard fixe de transmission.

1.1 Introduction

Un système de téléopération est une machine qui permet à un opérateur humain de se déplacer, de percevoir et de manipuler mécaniquement des objets à distance[1].Ces systèmes ont permis de dispenser la présence de l'homme dans des milieux dangereux tels que les milieux nucléaires, ainsi que dans des missions d'exploration comme celles en milieu sous-marin ou spatial.

Un système de téléopération se compose d'un robot maître, qui interprète les commandes de l'opérateur humain, d'un robot esclave responsable de l'exécution de la tâche, et d'un canal de communication permettant l'échange d'informations entre les deux robots [1].

Idéalement, le système de téléopération doit être complètement transparent, en reflétant fidèlement les interactions entre l'esclave et l'environnement sur l'opérateur, créant ainsi une illusion de télé-présence où l'opérateur ressent les efforts comme s'il réalisait directement la tâche. Cependant, atteindre cette transparence tout en maintenant la stabilité est un défi, en raison des retards de transmission des données entre le robot maître et le robot esclave [2].

Dans ce chapitre, nous allons présenter une description générale sur les systèmes de téléopération, leur analyse à travers l'évaluation des deux critères de performances (stabilité et transparence), ainsi que les architectures de contrôle les plus utilisées en pratique et leurs performances.

1.2 Synthèse historique

Un des premiers objets téléopérés a été la torpille de **Brennan** (1877). Elle offrait la possibilité à un opérateur de contrôler, de façon pratique, sa direction, jusqu'à une distance de 1,8 km, avec une vitesse pouvant atteindre 27 nœuds (environ 50 km/h). Les bases de ce système étaient d'origine mécanique, les commandes de l'opérateur étant transmises par l'intermédiaire des fils[19]. Les origines de la téléopération moderne (qui utilise un milieu de transmission électromagnétique) peuvent êtres considérées comme datant de la fin du 19ème siècle, en étroite liaison avec le développement des communications radio. Dans son brevet n° 613809 du 8 novembre 1898, **Nikola Tesla** décrit la télécommande (sans fils) d'un bateau miniature. À cette époque (début du vingtième siècle), l'opérateur était en contact visuel direct .

En 1953, des recherches menées au MIT1 conduisent au développement d'une nouvelle technologie de machines numériques adaptées à des opérations précises et répétées. Le concept du robot industriel est breveté en 1954. Ce brevet décrit la réalisation d'un bras mécanique asservi capable d'effectuer des tâches à caractère industriel. En 1958, Planet Corporation (USA) combine avec succès cette technologie avec la technologie des télé-manipulateurs développée pour l'industrie nucléaire américaine et crée le premier robot manipulateur industriel. La société Unimation voit le jour peu après, et livre le premier « Unimate » à General Motors en 1961. Vers la fin des années 70, ces robots dits « de première génération » se généralisent à l'ensemble de la production industrielle. Pendant que se développe ce nouveau concept « d'usine automatique », la robotique aborde d'autres domaines.

Ainsi, l'Union Soviétique réussit à téléopérer un robot mobile sur la Lune en 1971. En 1976, la sonde « Viking I » de la NASA2 se pose sur la planète Mars, équipée d'un bras manipulateur téléopéré depuis la terre afin de prélever des échantillons de sol et de rochers, la mission Mars Pathfinder avec le robot « sojourner » en 1997 ou encore la mission Mars Exploration Rovers avec les deux robots jumeaux « Spirit et Opportunity » en 2004 [3].

1.3 Définition

1.3.1 Téléopération

La téléopération est un ensemble de principes et techniques permettant à un opérateur humain de contrôler, en temps réel, un système robotique à distance pour exécuter des tâches spécifiques. Ce système, contrôlé à partir d'une station de commande, est relié à l'opérateur par un canal de communication spécifique.

1.3.2 Système de Téléopération

C'est un système de contrôle à distance, permettant à un opérateur humain de contrôler un robot éloigné à travers un canal de communication. Il repose sur l'échange d'informations de position et de force entre le dispositif maître, manipulé par l'utilisateur, et le dispositif esclave, qui exécute la tâche.

Lorsque les données de position et de force sont transmises uniquement du dispositif maître au dispositif esclave, on parle de système de téléopération unilatéral. En revanche, si l'environnement distant fournit un retour d'effort à l'opérateur via une interface haptique, le système est qualifié de bilatéral.

Dans un système bilatéral, L'opérateur humain exerce une force sur le dispositif maître, tel qu'un gant haptique, pour définir une position désirée, qui est ensuite transmise au dispositif esclave par le canal de communication. Le robot exécute alors cette commande. Les forces d'interaction entre le robot et son environnement sont renvoyées par le canal de communication sous forme de retour haptique, visuel, ou sonore à l'opérateur humain. Cette rétroaction permet à l'opérateur d'évaluer la situation et d'ajuster la force appliquée en conséquence. Ce principe est illustré à la (Figure 1.1) .

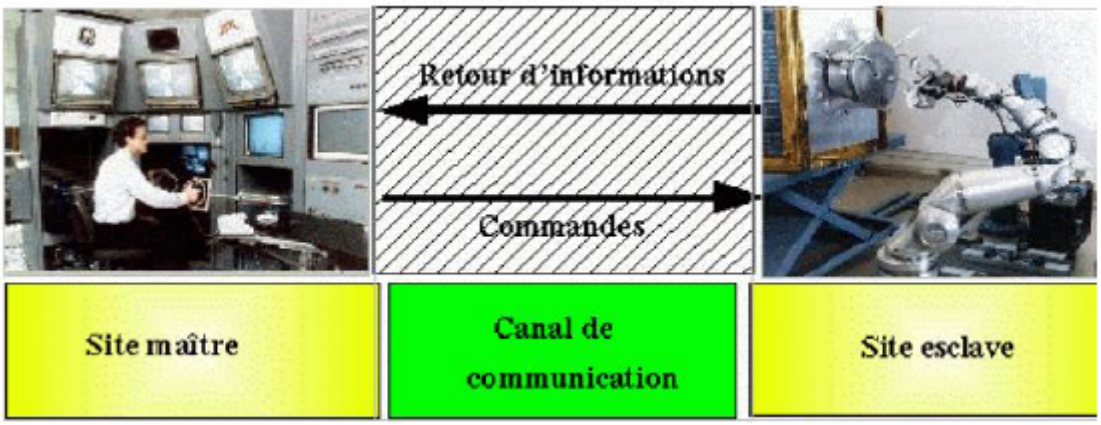


FIGURE 1.1 – Illustration de l'architecture générale d'un système de Télé-opération.

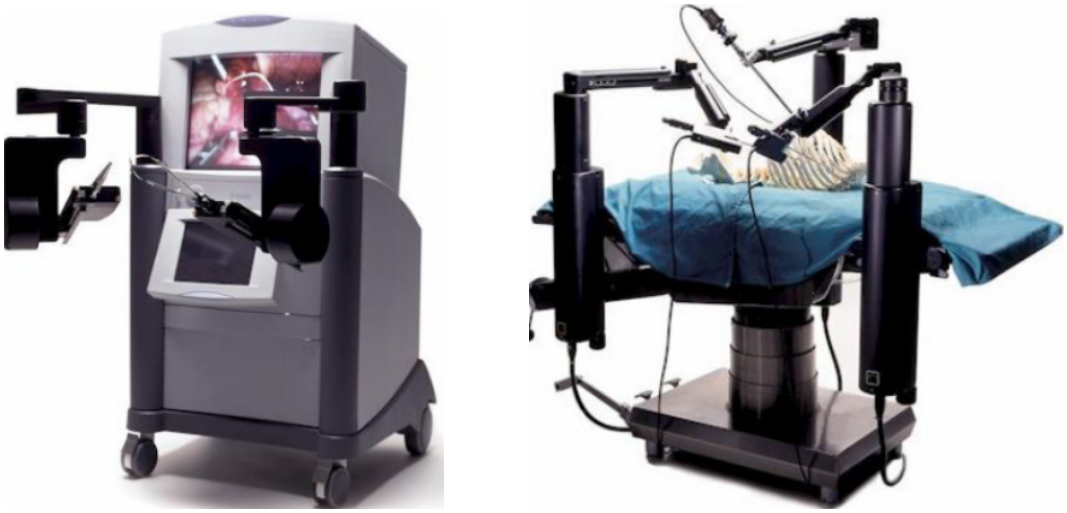
1.4 Quelques domaines d'application de la Téléopération

1.4.1 Domaine médicale

Les médecins et les chirurgiens utilisent de plus en plus de robots afin de les assister ou même de les remplacer dans certaines tâches. La téléopération médicale, ou téléchirurgie, permet aux chirurgiens de réaliser des opérations à distance via des robots [3]. Elle offre une précision accrue et l'accès à des soins spécialisés pour les patients isolés. Les systèmes avancés reproduisent fidèlement les gestes du chirurgien, réduisant ainsi les risques et la

fatigue opératoire. Cette technologie promet de transformer l'accès aux soins chirurgicaux dans le monde entier, en surmontant les barrières géographiques et en améliorant les résultats pour les patients.

Le concept de télé-opération mise en œuvre sur **Da Vinci** a été développé à l'extrême avec **Zeus** (Figure 1.2). En effet, en septembre 2001, le professeur **Marescaux** a téléopéré ce système depuis New York en réalisant avec succès une cholécystectomie (ablation de la vésicule biliaire) sur une patiente hospitalisée à Strasbourg. Cette première opération transatlantique baptisée « **Opération Lindbergh** », fut réalisable grâce à l'utilisation d'un réseau de communication à haut débit de France Télécom (10 mégabits/seconde). Le délai entre le geste du chirurgien sur la console maître et le retour visuel des mouvement des manipulateurs a pu être réduit à 150 ms [3].



(a) Console de commande avec les bras maître.

(b) Manipulateurs esclaves.

FIGURE 1.2 – Système Zeus.

1.4.2 Domaine spatial

L'utilisation de la télérobotique dans l'exploration spatiale révolutionne notre compréhension de l'univers. Des missions emblématiques telles que **Mars Pathfinder** avec le rover Sojourner en 1997 et les rovers jumeaux Spirit et Opportunity en 2004 démontrent son efficacité (Figure 1.3). Ces robots autonomes, équipés d'une multitude de capteurs, permettent une exploration approfondie de la surface martienne à la recherche de preuves de la présence passée d'eau liquide. En parallèle, la télérobotique facilite la maintenance des satellites et des stations spatiales, évitant ainsi le besoin d'envoyer des astronautes

pour des opérations de réparation, ce qui réduit les risques et les coûts associés.

(a) Mission «Mars Pathfinder» avec le robot sojourner en 1997.

(b) Mission «Mars ExplorationRovers» avec les deux robots jumeaux Spirit et Opportunity en 2004.

FIGURE 1.3 – Illustration de la télérobotique dans le domaine spatial.

1.4.3 Domaine militaire

La télérobotique joue un rôle crucial dans divers domaines militaires, notamment la désactivation de mines, la surveillance des territoires ennemis, et la commande à distance d'équipements tels que les drones, les véhicules terrestres et les robots de déminage (Figure 1.4). Bien que les robots autonomes ne soient pas encore largement déployés en raison de la complexité du terrain, la téléopération de plateformes robotisées offre déjà un soutien précieux aux soldats, notamment dans des environnements hostiles ou lors de tâches dangereuses. Cette évolution vers l'utilisation de robots dans les opérations militaires répond à la nécessité de réduire les risques pour les soldats tout en assurant l'efficacité des missions, ce qui pourrait potentiellement contribuer à la promotion de la paix en minimisant les dangers pour les êtres humains impliqués.

FIGURE 1.4 – Illustration de la télérobotique dans le domaine militaire.

1.5 Structure générale d'un système de Téléopération

La Figure 1.5 illustre les différents composants d'un système de téléopération à savoir [5] :

FIGURE 1.5 – Représentation d'un système de téléopération sous forme d'un réseau.

Opérateur humain : C'est la personne qui contrôle le système de téléopération. Il utilise le dispositif maître pour envoyer des commandes au dispositif esclave et surveiller l'environnement distant. L'opérateur interagit avec le dispositif maître et reçoit les retours d'information de l'environnement par le biais des interfaces haptiques, visuelles ou sonores.

Interface maître : il est utilisée par l'opérateur pour envoyer des commandes au manipulateur esclave. Il peut s'agir d'une télécommande, d'un joystick, d'un gant haptique, d'un contrôleur de mouvement, etc.

Canal de communication : C'est un moyen par lequel les commandes de l'opérateur sont transmises au dispositif esclave et les informations sur l'état du système sont renvoyées à l'opérateur. Il peut s'agir de connexions filaires ou sans fil telles que les réseaux informatiques, les liaisons radio ou les liaisons par satellite.

Manipulateur esclave : Il représente le robot ou dispositif contrôlé à distance par l'opérateur. Il reçoit les commandes du dispositif maître via le canal de communication et exécute les actions correspondantes dans l'environnement distant.

Environnement distant : Lieu où le dispositif esclave opère. Il peut s'agir d'un espace terrestre, sous-marin, aérien ou extraterrestre. L'environnement peut présenter des défis tels que des obstacles, des conditions météorologiques changeantes, ou des dangers potentiels.

1.6 Classification des systèmes de téléopération

1.6.1 Téléopération Unilatérale

La téléopération unilatérale implique la transmission unidirectionnelle de commandes de l'opérateur vers le robot (site esclave) (Figure 1.6), souvent limitée à des indications de

position ou de force, sans retour haptique pour l'utilisateur. Ce système fonctionne principalement sur la base d'une commande en boucle ouverte. Bien que moins complexe à mettre en œuvre, cette approche est limitée dans ses applications en raison du manque de feedback direct pour l'opérateur. Ce contexte spécifique peut rendre la téléopération unilatérale pertinente et efficace pour certaines tâches où la surveillance visuelle est adéquate pour guider les actions du robot.

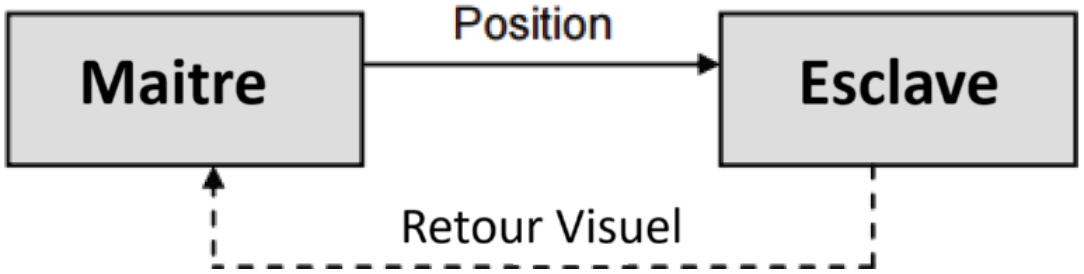


FIGURE 1.6 – Système de téléopération unilatérale.

1.6.2 Téléopération Bilatérale

La téléopération bilatérale est un système dans lequel un opérateur contrôle à distance un robot (le site esclave) tout en recevant un retour sensoriel de l'environnement à travers le robot (le site maître) (Figure 1.7). Cela permet une interaction bidirectionnelle entre l'opérateur et le robot, où les mouvements et les actions de l'opérateur sont répliqués par le robot, et où les sensations tactiles ou les forces rencontrées par le robot sont renvoyées à l'opérateur. Cette configuration offre un niveau élevé de feedback sensoriel, permettant à l'opérateur de ressentir directement les interactions avec l'environnement distant et de réaliser des tâches avec une précision et une sensibilité accrues. La téléopération bilatérale est largement utilisée dans des domaines tels que la chirurgie robotique, la maintenance à distance et les opérations dans des environnements dangereux ou inaccessibles.

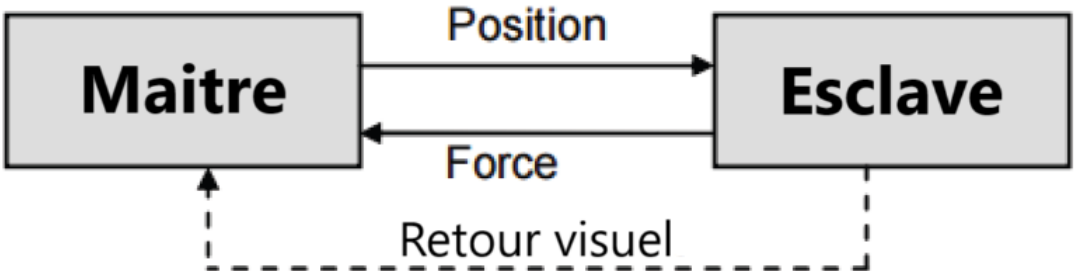


FIGURE 1.7 – Système de téléopération bilatérale.

1.7 Représentation mathématique d'un système de téléopération

1.7.1 Représentation d'un système maître-esclave à un degré de liberté

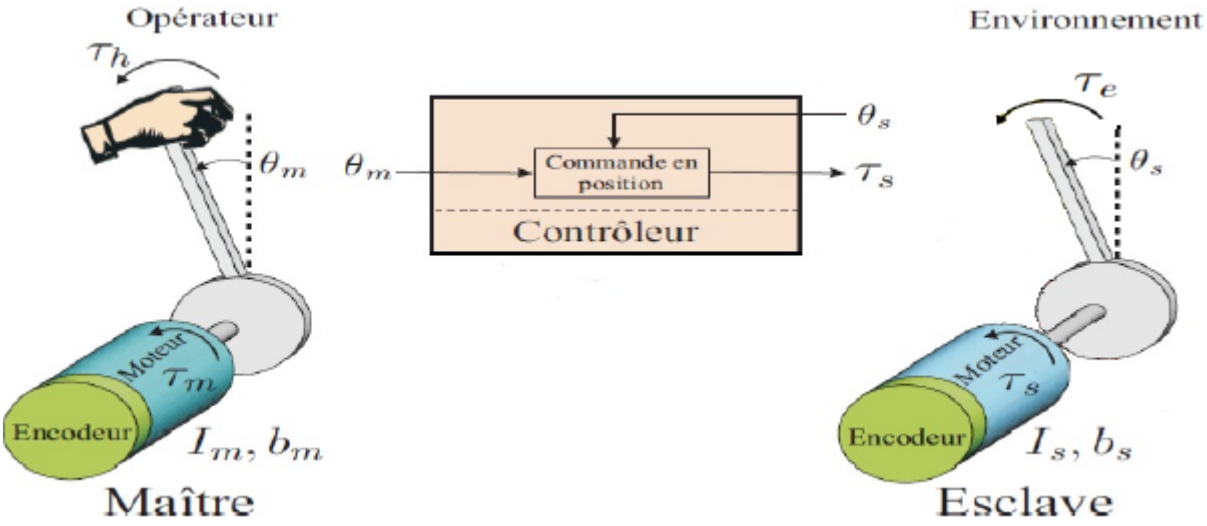


FIGURE 1.8 – Système de téléopération à 1ddl (mode de contrôle Position-Position).

La Figure 1.8 donne un système classique de téléopération haptique à 1 degré de liberté. représenté par les inerties Im et Is des masses en mouvement, et les frottements visqueux bm et bs (l'indice s est relatif au mot anglais slave). En considérant les différents couples appliqués à l'arbre de chaque système, et en négligeant les effets de gravité de la barre. La dynamique du dispositif maître et du dispositif esclave est donnée par les équations suivantes [2, 4] :

$$\begin{cases} I_m\ddot{\theta}_m + b_m\dot{\theta}_m = \tau_m + \tau_h \\ I_s\ddot{\theta}_s + b_s\dot{\theta}_s = \tau_s + \tau_e \end{cases} \tag{1.1}$$

Avec θ_m est la position du maître et θ_s est celle de l'esclave, τ_h la force (couple) produite par l'opérateur sur le maître, τ_e la force (couple) exercée par l'environnement sur l'esclave. τ_m et τ_s sont les forces (couples) délivrées par les actionneurs pour commander respectivement les mouvements du maître et de l'esclave. La dynamique de l'objet en interaction avec le robot esclave lors d'un contact rigide est donnée par l'équation dynamique suivante :

$$\tau_s = I_w\ddot{\theta}_s + b_w\dot{\theta}_s + C_w\theta_s \tag{1.2}$$

Où I_w, b_w et C_w représentent respectivement la masse, les frottements visqueux et la rigidité de l'objet.

La dynamique de l'opérateur humain est représentée également par l'équation suivante :

$$\tau_{op} - \tau_m = I_{op}\ddot{\theta}_m + b_{op}\dot{\theta}_m + C_{op}\theta_m \tag{1.3}$$

Où I_{op}, b_{op} et C_{op} représentent respectivement la masse, les frottements visqueux et la rigidité de l'opérateur, alors que τ_{op} désigne la force générée par les muscles de l'opérateur.

1.7.2 Représentation hybride

Un système de téléopération peut être modélisé par un réseau à deux ports, dérivé des réseaux électriques, reliant la force (couple) de l'opérateur τ_h et sa position θ_m aux variables de force τ_e et de position θ_s du manipulateur esclave.

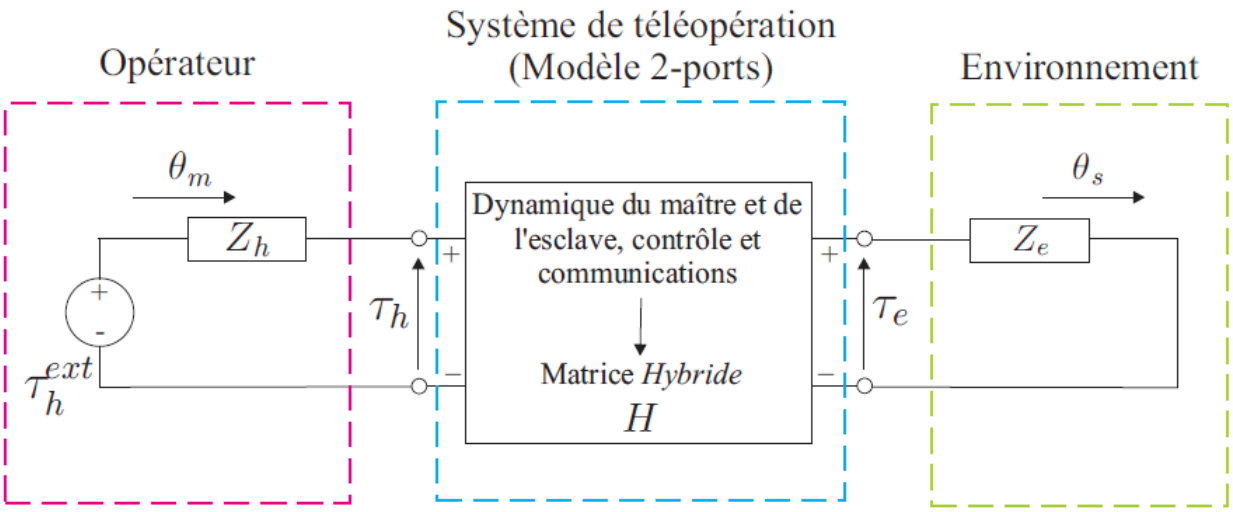


FIGURE 1.9 – Représentation d'un système de téléopération par modèle 2-ports.

La Figure 1.9 illustre un système de téléopération présenté par un réseau à deux ports sous une forme hybride, reliant l'opérateur humain caractérisé par une impédance Z_h à l'environnement distant d'impédance Z_e. Ce réseau est décrit par la relation suivante, proposée par (**Hannaford**, 1989)[2, 4] :

$$\begin{bmatrix} \tau_h \\ \theta_s \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} \\ h_{21} & h_{22} \end{bmatrix} \begin{bmatrix} \theta_m \\ \tau_e \end{bmatrix} \tag{1.4}$$

La matrice 2x2 est la matrice d'immitance hybride H. Elle caractérise complètement le système de téléopération (dynamique du maître et de l'esclave, contrôleurs, communications). Les composantes $h_{ij}(s)$ définissent les relations entre les positions et les forces

au maître et à l'esclave. Elles seront utilisées pour étudier les performances et la stabilité du système. Ces paramètres peuvent être interprétés physiquement :

Pour $\tau_e = 0$ (c'est-à-dire la force de l'interaction entre l'esclave et son environnement est nulle) :

— $h_{11} = \frac{\tau_h}{\theta_m}$: c'est **l'impédance ressentie par l'opérateur** lorsque l'esclave est en mouvement libre (aucune interaction avec l'environnement). Ce terme doit donc être minimisé et idéalement égal à zéro.

— $h_{21} = \frac{\theta_s}{\theta_m}$: est la fonction de transfert du **Tracking en Position**, en mouvement libre, qui doit tendre vers 1 sur la plage de fréquence la plus large possible.

Pour $\theta_m = 0$ (c'est-à-dire il n'a pas de transmission de position du maître vers l'esclave) :

— $h_{12} = \frac{\tau_h}{\tau_e}$: c'est le **Tracking en force**, lors d'un contact, lorsque la position du maître est maintenue constante.

— $h_{22} = \frac{\theta_s}{\tau_e}$: c'est **l'admittance de contact à l'esclave**. Pour une position du maître fixe, elle représente **le mouvement de l'esclave en présence d'une force de contact**. Elle aura pour effet de diminuer la raideur présentée à l'utilisateur par rapport à celle de l'environnement.

Les deux derniers termes sont exprimés dans la situation où la position du maître est maintenue constante. Cela n'est cependant pas facilement réalisable en pratique. Alors (**Aliaga** et **al.**, 2004) ont donc proposé de définir un set de quatre critères de performance facilement identifiable par l'expérience. Plus de détaille dans la référence.

1.8 Architectures de contrôle des systèmes de téléopération

1.8.1 Contrôle Position-Position

La méthode de contrôle Position-Position c'est la méthode utilisée dans notre projet. Elle se base exclusivement sur l'échange des mesures de positions entre le maître et l'esclave à travers deux canaux de communication $C1$ et $C4$. Le retour d'effort est généré à partir des erreurs de position entre le maître et l'esclave. Ces erreurs sont ensuite utilisées par les contrôleurs du maître C_m et de l'esclave C_s pour commander les mouvements de l'esclave respectivement. Historiquement, c'est la première méthode qui a été implémentée sur les systèmes de téléopération haptique[2, 5].

Le schéma de base du contrôle position-position est illustré à la Figure 1.10.

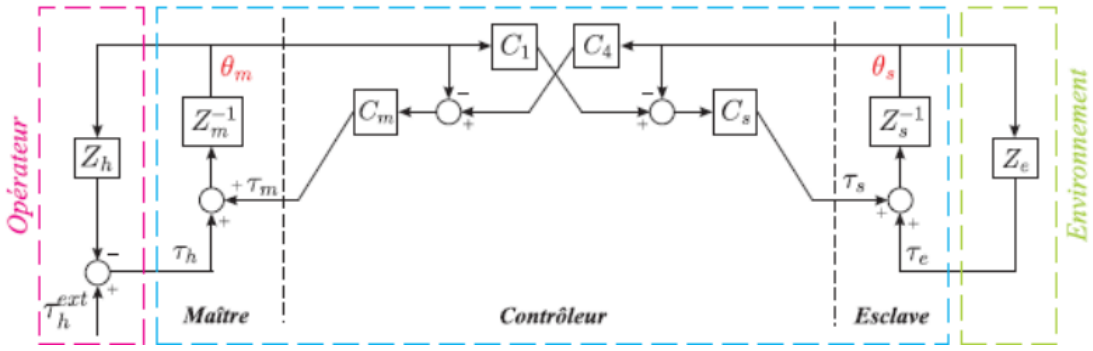


FIGURE 1.10 – Schéma de contrôle position-position.

la représentation hybride du système est alors fournie par[19] :

$$\begin{bmatrix} \tau_h \\ \theta_s \end{bmatrix} = \begin{bmatrix} \frac{(Z_m+C_m)(Z_s+C_s)-C_1C_4C_mC_s}{Z_s+C_s} & -\frac{C_4C_m}{Z_s+C_s} \\ \frac{C_1C_s}{Z_s+C_s} & \frac{1}{Z_s+C_s} \end{bmatrix} \begin{bmatrix} \theta_m \\ \tau_e \end{bmatrix} \tag{1.5}$$

l'avantage principal de cette stratégie de contrôle est sa facilité d'implémentation du point de vue hardware. Les systèmes dépourvus de capteur de force, peuvent être utilisés pour créer des effets haptiques seulement sur base de leurs capteurs de position. Cette méthode présente également en pratique une meilleure stabilité par rapport aux autres architectures de contrôle à deux canaux. Néanmoins, ces avantages sont au prix d'une dégradation des perceptions en mouvement libre[20].

1.8.2 Contrôle Force-Position

Cette stratégie de contrôle a été largement utilisée dans les systèmes de téléopération bilatérale. Son fonctionnement repose sur l'échange des signaux de position et de force à travers les canaux $C1$ et $C2$ respectivement, un capteur d'effort est donc nécessaire sur le site distant pour mesurer les forces d'interaction de l'esclave avec l'environnement. Dans cette méthode, la position du maître est transmise au robot esclave, tandis que ce dernier transmit à l'opérateur les efforts qu'il applique sur l'environnement [2, 5].

Le schéma de base de l'architecture de contrôle force-position est présenté dans la Figure 1.11.

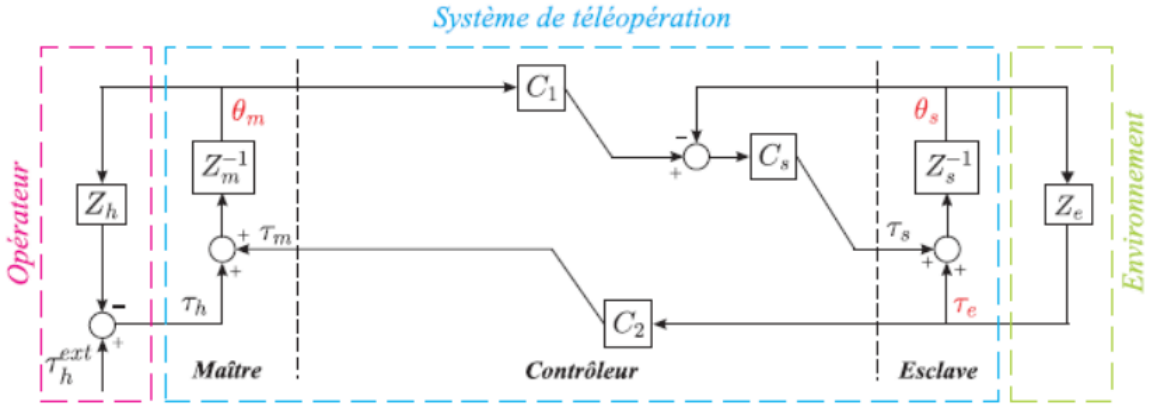


FIGURE 1.11 – Schéma de contrôle force-position.

la représentation hybride du système est alors fournie par [19] :

$$\begin{bmatrix} \tau_h \\ \theta_s \end{bmatrix} = \begin{bmatrix} Z_m & -C_2 \\ \frac{C_1 C_s}{Z_s + C_s} & \frac{1}{Z_s + C_s} \end{bmatrix} \begin{bmatrix} \theta_m \\ \tau_e \end{bmatrix} \tag{1.6}$$

Dans cette architecture de contrôle, l'opérateur ressent les effets dynamiques du maître. Donc pour avoir de meilleures performances, il faut d'une part utiliser une interface maître avec des caractéristiques dynamiques faibles, et d'autre part utiliser un contrôleur de position robuste pour l'esclave afin de maintenir la stabilité du système[20].

1.8.3 Contrôle à quatre canaux

Les deux méthodes de contrôle précédentes, Force-Position et Position-Position, sont des méthodes à deux canaux, avec l'échange de deux signaux entre le maître et l'esclave. Nous avons mis en évidence les limites de performance de ces stratégies. Dans l'objectif de les améliorer, le schéma à 4 canaux a été proposé par **(Lawrence, 1993)**. Il consiste à échanger simultanément les quatre informations de position et de force entre les deux sites à travers les quatre canaux $C1$, $C2$, $C3$ et $C4$. La présence d'un capteur de position et un autre de force est donc nécessaire dans chacun des robots maître et esclave[2, 5].

Le schéma de base de **Lawrence** est présenté dans la Figure 1.12.

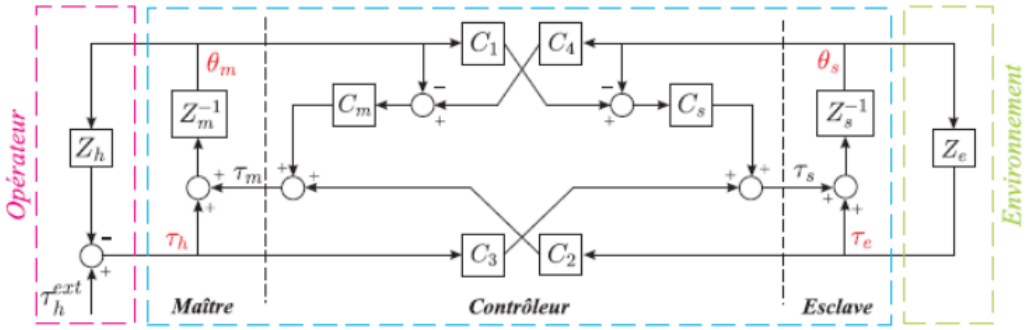


FIGURE 1.12 – Schéma de contrôle à quatre canaux.

La représentation hybride de l'équation dynamique du système est alors donnée par[19] :

$$\begin{bmatrix} \tau_h \\ \theta_s \end{bmatrix} = \begin{bmatrix} \frac{(Z_m+C_m)(Z_s+C_s)-C_1C_4C_mC_s}{Z_s+C_s+C_3C_4C_m} & -\frac{C_2(Z_s+C_s)+C_4C_m}{Z_s+C_s+C_3C_4C_m} \\ \frac{C_3(Z_m+C_m)+C_1C_s}{Z_s+C_s+C_3C_4C_m} & \frac{1-C_2C_3}{Z_s+C_s+C_3C_4C_m} \end{bmatrix} \begin{bmatrix} \theta_m \\ \tau_e \end{bmatrix} \tag{1.7}$$

Dans le contrôle à quatre canaux, la transparence idéale est vérifiée en ce qui concerne le tracking en force, le tracking en position et l'impédance maximum transmissible par le système lors d'un contact, néanmoins l'impédance minimum ressentie en mouvement libre présente un défaut, où l'opérateur ressent les effets dynamiques de l'interface maître[20].

1.9 Les caractéristique des systèmes de téléoperation

1.9.1 Stabilité

La stabilité est une propriété fondamentale des systèmes de téléopération [6]. Les travaux de Kosuge et son équipe ont montré que des variations, même relativement petites, du délai de transmission peuvent entraîner une instabilité [3]. L'opérateur et l'environnement distant apparaissent comme des systèmes incertains et variables. De plus, les systèmes de téléopération maître-esclave sont des systèmes non linéaires et complexes [7]. Ils présentent une dynamique qui peut être indisponible en raison des incertitudes du modèle. Ces incertitudes peuvent être particulièrement problématiques lorsque le robot esclave est en contact avec un environnement inconnu [8].

Pour faire face à ces défis, les lois de commande doivent être robustes face aux incertitudes dynamiques et cinématiques. L'objectif est de garantir la stabilité de la téléopération bilatérale, ce qui signifie que le système doit être capable de maintenir son état d'équilibre malgré les perturbations et les incertitudes [7].

Pour garantir la stabilité d'un système de téléopération à retour d'effort, l'analyse de la passivité est essentielle. Plus précisément, on cherche à ce que le port d'interaction (correspondant au point d'échange de puissance entre le système et la source d'énergie

extérieur) entre le robot et l'environnement soit passif (ne crée pas d'énergie). Une passivité de ce port vérifiée, le système interagira via ce port, de façon stable, avec n'importe quel environnement passif. Donc la stabilité est déduite de la passivité.

Pour étudier la stabilité et la transparence on utilise les vitesses du maître et de l'esclave représentées respectivement par $\dot{\theta}_m$ et $\dot{\theta}_s$.Pour les obtenir, il suffit de dériver leurs positions θ_m et θ_s.

Stabilité par la passivité :

Le système de communication peut être modélisé par un quadripôle à 2 ports caractérisé par sa matrice hybride $H(s)$ (Figure 1.13)[9].

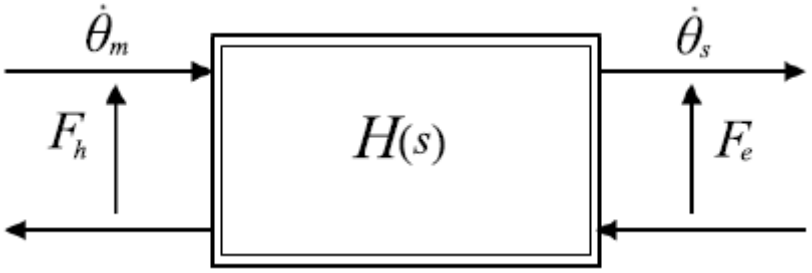


FIGURE 1.13 – Représentation d'un système de téléopération sous forme d'un quadripôle.

La référence est choisie de façon à ce que les vitesses dans le quadripôle soient positives. La matrice hybride de ce système est définie par la relation[9] :

$$\begin{bmatrix} F_h(s) \\ -\dot{\theta}_s \end{bmatrix} = \begin{bmatrix} h_{11}(s) & h_{12}(s) \\ h_{21}(s) & h_{22}(s) \end{bmatrix} \begin{bmatrix} \dot{\theta}_m(s) \\ F_e(s) \end{bmatrix} \tag{1.8}$$

s : est la variable de Laplace.

La condition nécessaire et suffisante pour la stabilité est extrêmement difficile à obtenir puisque le système est multi-variables (plusieurs entrées, plusieurs sorties). Cependant, une condition suffisante pour la stabilité basée sur la passivité est bien adaptée à cette application.

Cette méthode permet en plus de s'affranchir des contraintes imposées par les impédances de l'opérateur et de l'environnement éloigné, et ainsi de garder le système stable quels que soient ces deux derniers facteurs. On définit les vecteurs suivants [9] :

$$F(t) = \begin{bmatrix} T_h & T_e \end{bmatrix}^T \text{ et } \dot{\theta}(t) = \begin{bmatrix} \dot{\theta_m} & \dot{\theta}_s \end{bmatrix}^T$$

Qui sont des fonctions de temps de carré sommable (T dénote la transposition). Alors le système est passif si on a :

$$\int_0^t F^T(t)\dot{\theta}(t)dt = -S_0 \qquad \forall t \succ 0 \tag{1.9}$$

S_0 : Est un nombre fixé souvent associé avec une énergie initiale stockée à $t = 0$.

Si le système est initialement relaxé, c'est-à-dire si les sources du système sont nulles à $t = 0$, et aucune des inerties n'a une vitesse initiale, alors $S_0 = 0$. Et lorsque l'inégalité devient une égalité, le système est dite non dissipatif où sans pertes.

Plus concrètement, on voit que $F^T\dot{\theta}$ représente une puissance et que son intégration représente une énergie. Le système est passif si l'énergie qu'il voit entrer est plus importante que celle qu'il fournit à l'extérieur. On comprend ainsi qu'un système passif est nécessairement stable.
A partir de la matrice hybride du système on peut déterminer si celui-ci est passif ou non : On définit $S(s)$ matrice de dispersion du système définie par l'expression suivante :

$$F(s) - \dot{\theta}(s) = S(s)(F(s) + \dot{\theta}(s)) \tag{1.10}$$

Ce qui entraîne d'après (1.8) et (1.10) :

$$S(s) = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} (H(s) - I)(H(s) + I)^{-1} \tag{1.11}$$

Théorème [9] :

Un système est passif quelles que soient ses entrées F et $\dot{\theta}$ si et seulement si $|S| = 1$.

1.9.2 Transparence

Dans une téléopération bilatérale, on désire que le système soit totalement transparent[9]. La transparence parfaite est atteinte lorsque l'utilisateur a l'impression de manipuler directement les objets de l'environnement, sans ressentir ni les effets dynamiques du manipulateur maître, ni ceux du manipulateur esclave[10].

La Figure 1.14 représente un système de téléopération sous forme d'un réseau.

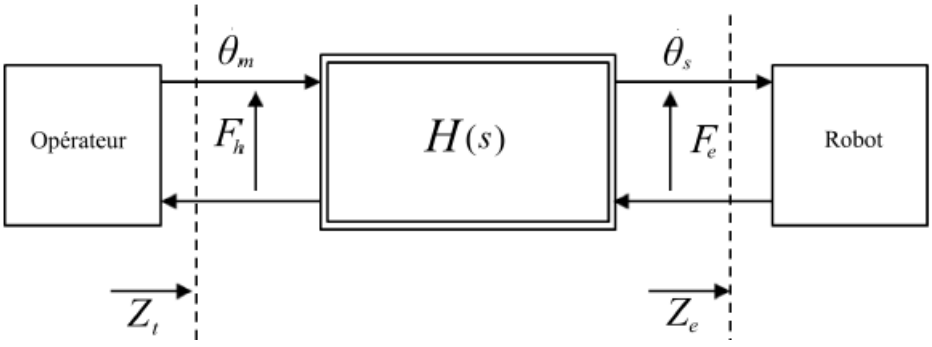


FIGURE 1.14 – Représentation d'un système de téléopération sous forme d'un réseau.

Les forces sont analogues à des tensions et les vitesses à des intensités. Cette analogie est possible si on considère que le système de téléopération est linéaire. Lorsque le robot esclave est en contact avec son environnement, la vitesse de l'effecteur de l'esclave $\dot{\theta}_s$ et la force F_e qu'il exerce sur son environnement ne sont plus indépendantes. Elles sont reliées par l'impédance Z_e transmise par l'environnement de l'esclave[9] :

$$F_e = Z_e(\dot{\theta}_s) \tag{1.12}$$

Si l'opérateur perçoit l'environnement distant comme s'il était sur place, alors la force exercée par l'opérateur sur le bras maître F_h et la vitesse du bras maître $\dot{\theta}_m$ devraient avoir la même relation : c'est-à-dire pour les mêmes forces $F_e = F_h$ on désire obtenir les mêmes vitesses, soit $\dot{\theta}_s = \dot{\theta}_m$. Ceci nécessite que l'impédance transmise Z_t ou ressentie par l'opérateur, soit définie telle que :

$$F_h = Z_t(\dot{\theta}_m) \tag{1.13}$$

Donc pour que la transparence soit parfaite , il faut satisfaire les conditions de transparence (i.e. $F_e = F_h$ et $\dot{\theta}_s = \dot{\theta}_m$).
Selon **Lawrence**, la transparence idéale d'un système de téléopération est atteinte lorsque le rapport Z_t/Z_e tend vers 1 ($(Z_t/Z_e) \to 1$)[9].
Où : Z_t est l'impédance ressentie par l'opérateur humain.

1.10 Le problème de délai de transmission

De nombreuses recherches ont été réalisées pour l'étude et l'analyse du problème de délai (retard) de transmission en téléopération. Deux approches ont été utilisées afin d'analyser ce problème **(Kheddar, 1997)**. La première est basée sur l'automatique classique, tandis que la seconde est basée sur la théorie des lignes et de la passivité.

La première approche est de nature compensatoire, l'idée était d'introduire des correcteurs dans la boucle d'asservissement afin de compenser les systèmes à retard **(De Larminat, 1993)**. La seconde approche, quant à elle, considère le retard comme subi, c'est à dire comme partie intégrante du système **(Niemeyer et Slotine, 1991)**, **(Leung et Francis, 1992)**, **(Spong, 1993)** et **(Lawrence, 1993)** et **(Andriot, 1997)**[3].

Malheureusement, ces deux approches exigent que le délai de transmission soit connu, alors que la majorité des protocoles publiques actuels de transmission de données informatiques ne garantissent pas la constance de ces délais. En effet, indépendamment de la taille des données transmises, le délai peut varier d'une manière très aléatoire pour certains protocoles de transmission (par exemple Internet). Les travaux de **Kosuge** et

son équipe ont montré **(Kosuge et al., 1996)** que ce problème entraîne une instabilité même si les variations de délai sont relativement petites.

Dans les systèmes de téléopération originaux, l'opérateur doit souvent passer par un long apprentissage avant d'obtenir une bonne adaptation et une maîtrise adéquate du système de téléopération **(Zak et Das, 1995)**. De plus, l'adaptation obtenue peut ne pas être adéquate pour d'autres architecture de téléopération. La sophistication et le nombre important d'applications téléopérées engendre une augmentation de la charge de travail de l'opérateur, sa fatigue et ce qui entraîne une diminution de ses performances de poursuite et augmente ainsi les risques d'erreurs de téléopération[3].

1.11 Conclusion

Dans ce chapitre, nous avons présenté un aperçu général sur les systèmes de téléopération, incluant une brève histoire et ses applications modernes. Nous avons exploré Les éléments essentiels de ces systèmes, indispensables pour comprendre le contrôle de la téléopération.

Nous avons examiné la classification des systèmes de téléopération, mettant en évidence les différences entre les systèmes unilatéraux et les systèmes bilatéraux. La représentation mathématique des systèmes, notamment celle d'un système maître-esclave à un degré de liberté et la représentation hybride, a également été abordée.

Les trois principales architectures de contrôle ont été présentées et comparées selon leurs performances. Les méthodes Force-Position et Position-Position, bien que simples et pratiques, présentent des limitations en termes de performance haptique et nécessitent un matériel spécifique. À l'inverse, l'architecture à quatre canaux de communication offre une transparence idéale et une meilleure qualité de transmission de la raideur, malgré une complexité accrue en termes de mise en œuvre.

L'analyse des systèmes de téléopération se fait par l'évaluation de la stabilité et de la transparence. Nous avons présenté la méthode de l'analyse de la passivité en ce qui concerne la stabilité, ainsi nous avons donné les critères qui définissent la transparence idéale.

Notre travail se concentre sur la mise en œuvre d'un contrôleur flou utilisant une carte Arduino pour gérer un système de téléopération à un degré de liberté. Dans ce système, le robot maître, contrôlé par l'utilisateur humain, commande le robot esclave situé à distance, lequel doit reproduire les actions effectuées par le robot maître. L'implémentation du contrôleur flou vise à ajuster avec précision la position du robot esclave tout en prenant

en compte le retard fixe de transmission des données entre les deux robots. Ce qui sera détaillé dans le chapitre suivant.

Chapitre 2

Synthèse d'un régulateur flou

2.1 Introduction

Les outils intelligents tels que la logique floue, les réseaux de neurones artificiels et les algorithmes génétiques, jouent un rôle de plus en plus crucial dans la conception, la modélisation et la commande des systèmes complexes, que ce soit des robots, des procédés biologiques, ou des véhicules routiers[11].
Ces méthodes d'intelligence artificielle proposent des solutions efficaces pour traiter des modèles dynamiques non linéaires et complexes, caractéristiques de nombreux systèmes robotiques, comme les systèmes de téléopération. En l'absence de modèles mathématiques précis, ces outils permettent de surmonter les défis liés au contrôle, où les erreurs de modélisation peuvent avoir un impact considérable sur les performances des contrôleurs.

Parmi ces outils, La logique floue introduite par **Zadeh** dans les années soixante constitue un outil très puissant pour la représentation des termes et des connaissances vagues. Elle est issue de la capacité de l'homme à décider et à agir d'une manière intelligente malgré l'imprécision et l'incertitude des données disponibles. Contrairement aux approches classiques basées sur des modèles mathématiques rigoureux, la commande par logique floue repose sur un ensemble de règles linguistiques du type "Si... Alors...", qui modélisent la stratégie de raisonnement d'un opérateur humain. Cette approche permet d'aborder le contrôle des systèmes complexes de manière flexible et adaptative, sans nécessiter une connaissance exhaustive du système.

Dans ce chapitre, nous présentons une synthèse sur la logique floue. Nous commencerons par expliquer les notions de base de la théorie des ensembles flous ainsi que les outils mathématiques nécessaires à leur manipulation. Ensuite, nous présentons la structure générale d'un système flou, ainsi que les différentes étapes du raisonnement flou. Enfin, nous discuterons des avantages et des inconvénients de la commande floue, en soulignant ses points forts et les défis qu'elle présente dans les applications pratiques.

2.2 Historique de la Logique Floue

En 1961, le professeur Lotfi A. Zadeh, de l'Université de Berkeley en Californie, pose les bases théoriques de la logique floue. Quelques années plus tard, en 1973, il propose d'appliquer cette logique aux problèmes de réglage, ouvrant ainsi la voie à de nouvelles approches en automatisation. En 1974, la première application pratique de la logique floue est réalisée sur une turbine à vapeur. D'autres applications industrielles suivent, notamment en 1980 sur un four à ciment et en 1983 sur un épurateur d'eau. En 1985, le Japon commercialise les premiers produits industriels utilisant la logique floue pour des tâches de réglage et de commande, avec le développement de processeurs dédiés à ces applications[12].

2.3 Domaines d'Application de la Logique Floue

La logique floue est présente presque dans tous les domaines, on cite quelques domaines d'applications de logique floue à savoir :

- Robotique.
- Systèmes experts.
- Reconnaissance des formes.
- Classification.
- Traitement d'images.

2.4 Généralités sur la logique floue

2.4.1 La Logique Floue

La logique est l'étude des méthodes de raisonnement, où le raisonnement signifie obtenir de nouvelles propositions à partir de propositions existantes. Dans le cadre de la logique classique, une proposition est soit vraie, soit fausse (1 ou 0). Bien que cette logique ait dominé le monde scientifique pendant plus d'un siècle, elle rencontre de nombreux problèmes liés à son utilisation et à son application en raison de la nature absolue des valeurs de vérité. La logique floue est alors apparue, offrant une transition de la vérité absolue à la vérité partielle, représentée par un nombre compris dans l'intervalle [0 1][13].

Fondée sur la théorie des ensembles, la logique floue permet de représenter les connaissances incertaines et imprécises, et fournit un moyen approximatif mais efficace pour décrire le comportement des systèmes qui sont trop complexes ou mal définis. La conception d'un système basé sur cette logique commence par le choix de variables linguistiques (entrées et sorties du contrôleur), le pas suivant consiste à créer l'univers du discours pour chaque variable linguistique, c'est-à-dire un ensemble de valeurs linguistiques que la va-

riable peut prendre. Ces valeurs sont représentées avec des ensembles qui peuvent être discrets ou continus.

2.4.2 Notion de sous-ensemble flou

Dans la théorie classique des ensembles, un sous ensemble A de X est défini par une fonction caractéristique $\mu_A(x)$ qui caractérise chaque élément x appartenant à U.

$$\mu_A(x) = \begin{cases} 1 & x \in A \\ 0 & x \notin A \end{cases} \tag{2.1}$$

Cette fonction d'appartenance ne peut prendre que deux valeurs possibles : la valeur 1 si x appartient à A ou la valeur 0 si x n'appartient pas à A.
Un sous ensemble flou A de X est défini par une fonction d'appartenance qui associe a chaque élément x de X, le degré $\mu_A(x)$, compris entre 0 et 1 avec lequel x appartient a A :

$$\mu_A(x) \in [0, 1] \tag{2.2}$$

2.4.3 Variables linguistique

Une variable linguistique « floue » décrit une situation imprécise par des mots ou des expressions tels que : (négatif, zéro, positif) ou (faible, moyenne, élevée).
Une variable linguistique prend une valeur linguistique qui est un ensemble flou défini sur l'univers de discours.

2.4.4 Fonctions d'Appartenance

Un ensemble flou est défini par sa « fonction d'appartenance », qui correspond à la notion de « fonction caractéristique » en logique classique[14], et fonction d'appartenance en logique floue. Il permet de définir le degré de vérité de cette variable floue en fonction de sa grandeur d'entrée.

Voici les fonctions d'appartenance les plus couramment utilisées :

Fonction Triangulaire : Elle est définie par trois paramètres (a,b et c), qui déterminent les coordonnées des trois sommets (Figure 2.1).

$$\mu(x) = max\left(min\left(\frac{x-a}{b-a}, \frac{c-x}{c-b}\right), 0\right) \tag{2.3}$$

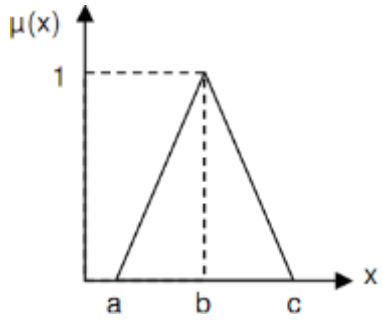


FIGURE 2.1 – Fonction Triangulaire.

Fonction Trapézoïdale : Elle est définie par quatre paramètres (a,b,c et d), qui déterminent les coordonnées des quatre sommets (Figure 2.2).

$$\mu(x) = max\left(min\left(\frac{x-a}{b-a}, 1, \frac{d-x}{d-c}\right), 0\right) \tag{2.4}$$

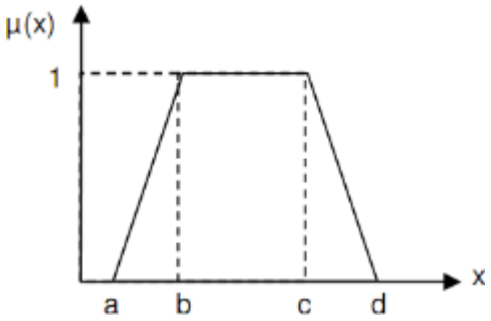


FIGURE 2.2 – Fonction Trapézoïdale.

Fonctions Gaussienne : Elle est caractérisée par deux paramètres (σ et m) (Figure 2.3).

$$\mu(x) = \exp\left(-\frac{(x-m)^2}{2\sigma^2}\right) \tag{2.5}$$

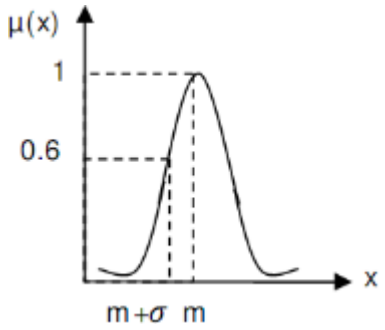


FIGURE 2.3 – Fonction Gaussienne.

Fonction Sigmoïde : Elle est définie par deux paramètres (a et c) (Figure 2.4).

$$\mu(x) = \left(\frac{1}{1 + \exp\left(-a\left(x - c\right)\right)}\right) \tag{2.6}$$

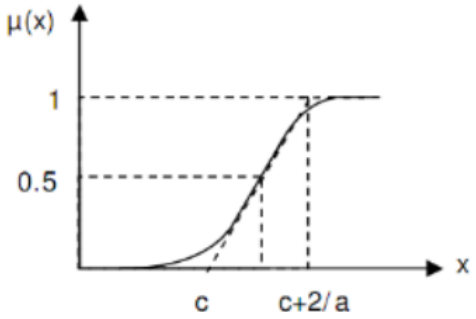


FIGURE 2.4 – Fonction Sigmoïde.

Il est à signaler que le choix de la forme de la fonction d'appartenance dépend de l'application et des connaissances de l'opérateur.

2.4.5 Univers de Discours

L'univers de discours représente le domaine de variation de la variable linguistique. Pratiquement, il représente le domaine du fonctionnement du système à commander, il faut donc le prendre en considération lors du réglage du régulateur. comme illustré sur la figure ci-dessous (Figure 2.5) dont plusieurs sous-ensembles sont décrits par des fonctions d'appartenances μ_A triangulaires et trapézoïdales.

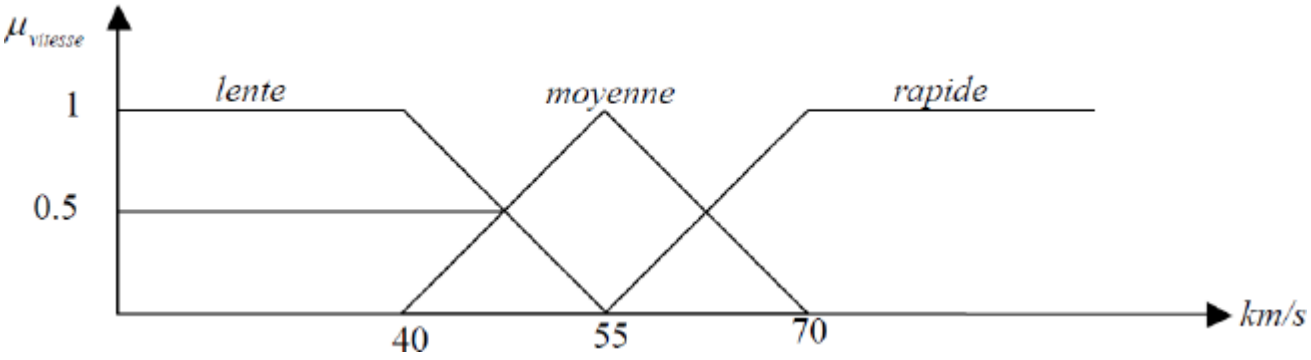


FIGURE 2.5 – Exemple de représentation graphique des termes linguistiques.

2.5 Opérations sur les ensembles flous

Soient A et B deux sous-ensembles flous définis dans un univers du discours U par les fonctions d'appartenance μ_A et μ_B (Figure 2.6). L'union, l'intersection et la complémentation des sous-ensembles flous, peuvent être définis à l'aide des opérations sur les fonctions d'appartenance [14, 11].

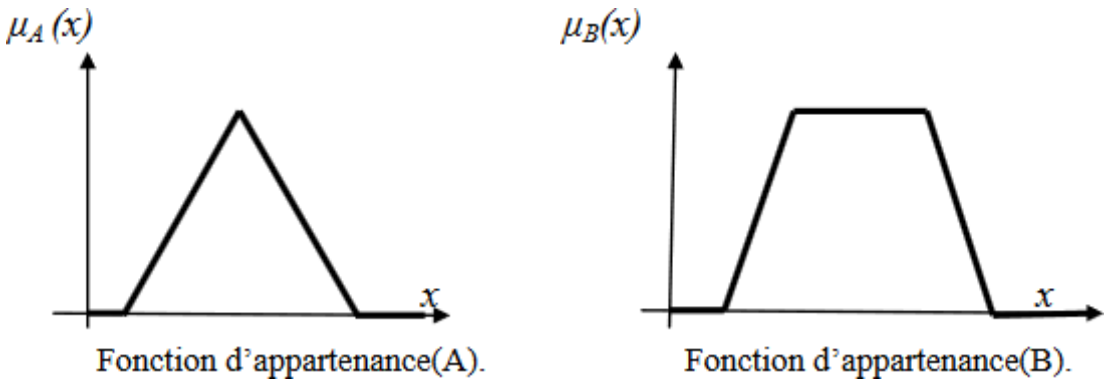


FIGURE 2.6 – Fonctions d'appartenance triangulaire (A), et trapézoïdale (B).

2.5.1 Intersection :

L'intersection de deux sous-ensembles flous A et B de U est un sous ensemble flou $A \bigcap B$ qui est défini par le plus petit sous ensemble contenu à la fois dans A et B (Figure 2.7), sa fonction d'appartenance est donnée par :

$$\mu_{A \bigcap B}(x) = min[\mu_A(x), \mu_B(x)]; \forall x \in U. \tag{2.7}$$

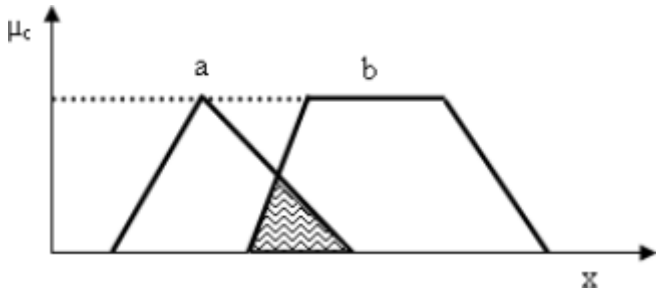


FIGURE 2.7 – Fonction d'appartenance

2.5.2 Union :

L'union de deux sous-ensembles flous A et B de U est un sous ensemble flou AUB qui est défini par le plus grand sous-ensemble flou qui contient A et qui contient B (Figure 2.8). Sa fonction d'appartenance est donnée par :

$$\mu_{A\cup B}(x) = max[\mu_A(x), \mu_B(x)]; \forall x \in U. \tag{2.8}$$

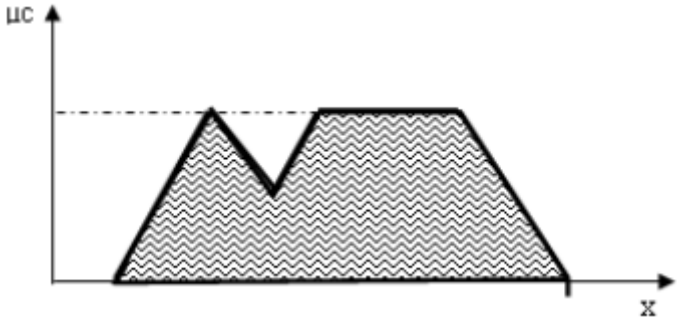


FIGURE 2.8 – Fonction d'appartenance AUB

2.5.3 Complément :

Le complément d'un sous-ensemble flou A est un ensemble flou dénoté par $\bar{\text{A}}$ dont la fonction d'appartenance est donnée par :

$$\mu_{\bar{\text{A}}}(x) = 1 - \mu_{\text{A}}(x); \forall x \in U. \tag{2.9}$$

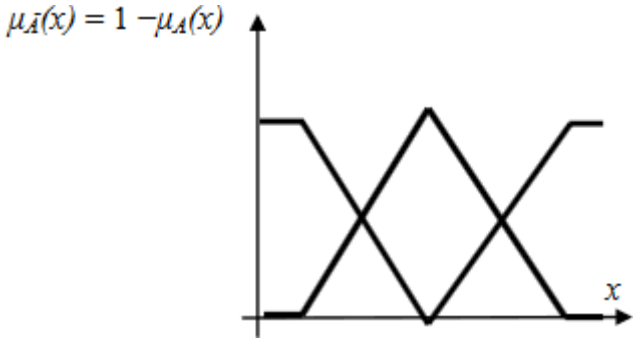


FIGURE 2.9 – Opérateur de complémentation de la fonction d'appartenance A.

2.6 Structure Générale d'un régulateur Flou

Un régulateur flou est un système flou conçu spécifiquement pour réguler un processus. Sa structure générale est illustrée par le schéma de la (Figure 2.10). La représentation des connaissances se base sur des ensembles flous, où les règles fournissent directement les conclusions[11].

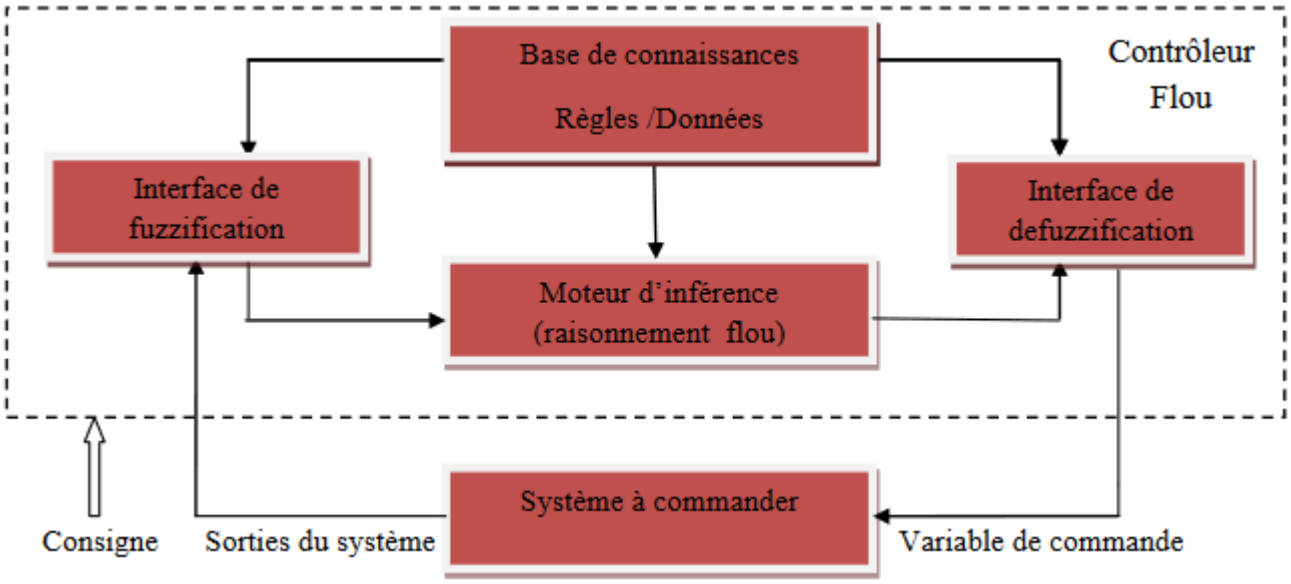


FIGURE 2.10 – Structure de base d'un régulateur flou.

Les variables caractéristiques du système à commander et les consignes définissent les variables d'entrée du contrôleur flou. Les variables caractéristiques sont, en général, les grandeurs de sortie du processus. Les variables de sortie du contrôleur flou sont les commandes appliquées au processus.

La configuration de base du contrôleur flou comprend quatre parties :

- La fuzzification des variables d'entrée, avec éventuellement un prétraitement de l'information.
- La Base de connaissances.
- Le Moteur d'inférence floue.
- La défuzzification, avec éventuellement un post-traitement de l'information.

2.6.1 Fuzzification

La fuzzification est la première étape d'un système de logique floue. Elle consiste à convertir des données d'entrée numériques en variables linguistiques floues, permettant au système de traiter des informations imprécises ou incertaines. Ce processus utilise des fonctions d'appartenance pour déterminer le degré d'appartenance de chaque donnée numérique à des ensembles flous prédéfinis, facilitant ainsi leur intégration dans le moteur d'inférence.

Exemple :

Considérons une entrée *Vitesse* V , définie par l'ensemble des variables linguistiques : Très lente , Lente, Moyenne , Rapide , Très Rapide.

Dans cette figure (2.11), les termes suivants sont définis :

- Univers du discours : $[0, 200]$
- Variable linguistique : La vitesse.
- Valeurs linguistiques : Très lente , Lente, Moyenne , Rapide , Très Rapide.

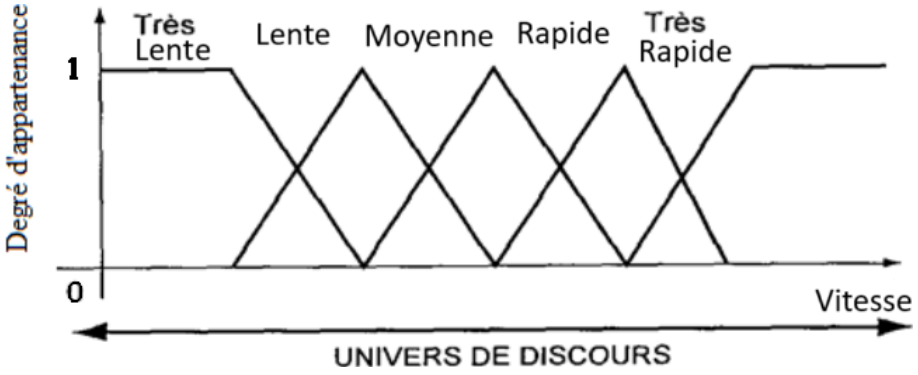


FIGURE 2.11 – Exemple de variation d'une variable linguistique.

2.6.2 Base de connaissance

La conception d'une base de connaissances représente la phase la plus importante dans la conception des systèmes experts. Elle comprend la base de données et la base des règles floues.

2.6.2.1 La base de données

Elle contient les informations nécessaires pour définir les variables linguistiques, les ensembles flous et les fonctions d'appartenance. Cette base permet au système d'avoir une compréhension claire des données d'entrée et de sortie en termes flous.

2.6.2.2 La Base de Règles Floues

La base de règles d'un système flou décrit le comportement des sorties en fonction des différentes entrées, en termes qualitatifs. Elle est constituée de règles floues de la forme suivante [12] :

- ***Si*** condition 1 ***ET/OU*** condition 2 ***(ET/OU ...)***, ***Alors*** action sur les sorties.
- ***Si*** condition 3 ***ET/OU*** condition 4 ***(ET/OU ...)***, ***Alors*** action sur les sorties.
- ...
- ***Si*** condition n ***ET/OU*** condition n + 1 ***(ET/OU ...)***, ***Alors*** action sur les sorties.

2.6.3 Moteur d'Inférences

Cette étape consiste à transformer la partie floue obtenue par la fuzzification en une nouvelle partie floue, afin de définir l'ensemble flou qui caractérisera la commande. Le mécanisme d'inférence combine les règles floues pour effectuer une transformation des ensembles flous de l'espace d'entrée vers ceux de l'espace de sortie.

Plusieurs méthodes d'inférence sont couramment utilisées, parmi lesquelles :

- Méthode d'inférence Max-Min
- Méthode d'inférence Max-Prod
- Méthode d'inférence Somme-Prod

1. **Méthode Max-Min (Mamdani) :** Le modèle flou de type **Mamdani** a d'abord été proposé pour tenter de contrôler une machine à vapeur et une chaudière en utilisant un ensemble de règles de contrôle linguistiques obtenues auprès d'un opérateur humain expérimenté **(Mamdani et Assilian, 1974)**[16]. Dans cette méthode, l'opérateur "**ET**" est réalisé par la fonction "**Min**", qui prend la valeur minimale des degrés d'appartenance. la conclusion "**ALORS**" de chaque règle est également calculée avec la fonction "**Min**" et la liaison entre toutes les règles (

opérateur "**OU**") par la fonction Max .La dénomination de cette méthode, dite Max-Min ou "implication de **Mamdani**", est due à la façon de réaliser les opérateurs ALORS et OU de l'inférence[15].

La figure 2.12. Représente graphiquement le principe de la méthode d'inférence max-min.

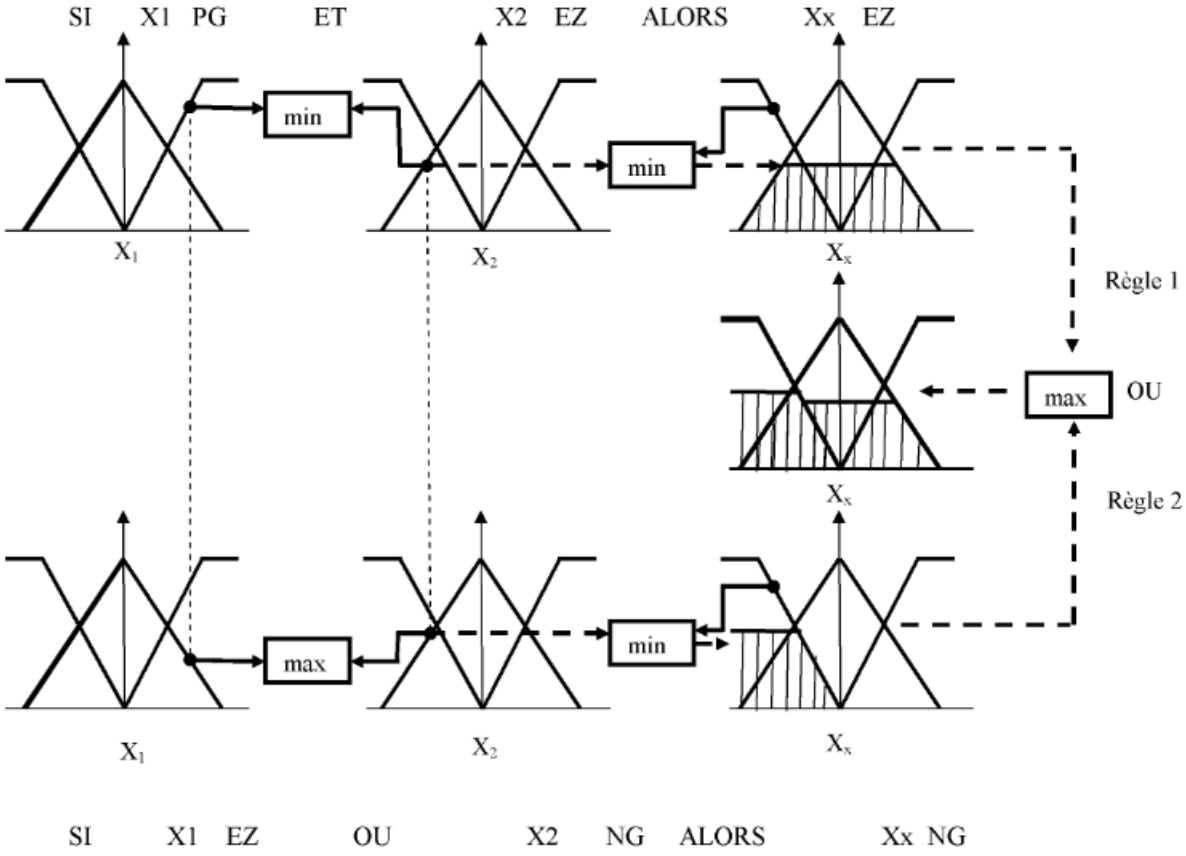


FIGURE 2.12 – Représentation graphique du principe de la méthode d'inférence max-min.

2. **Méthode Max-Prod (Larsen) :** La différence par rapport à la méthode précédente réside dans la manière de réaliser la conclusion "**ALORS**". Dans ce cas, on utilise le produit comme illustré par la (Figure 2.13)[15].

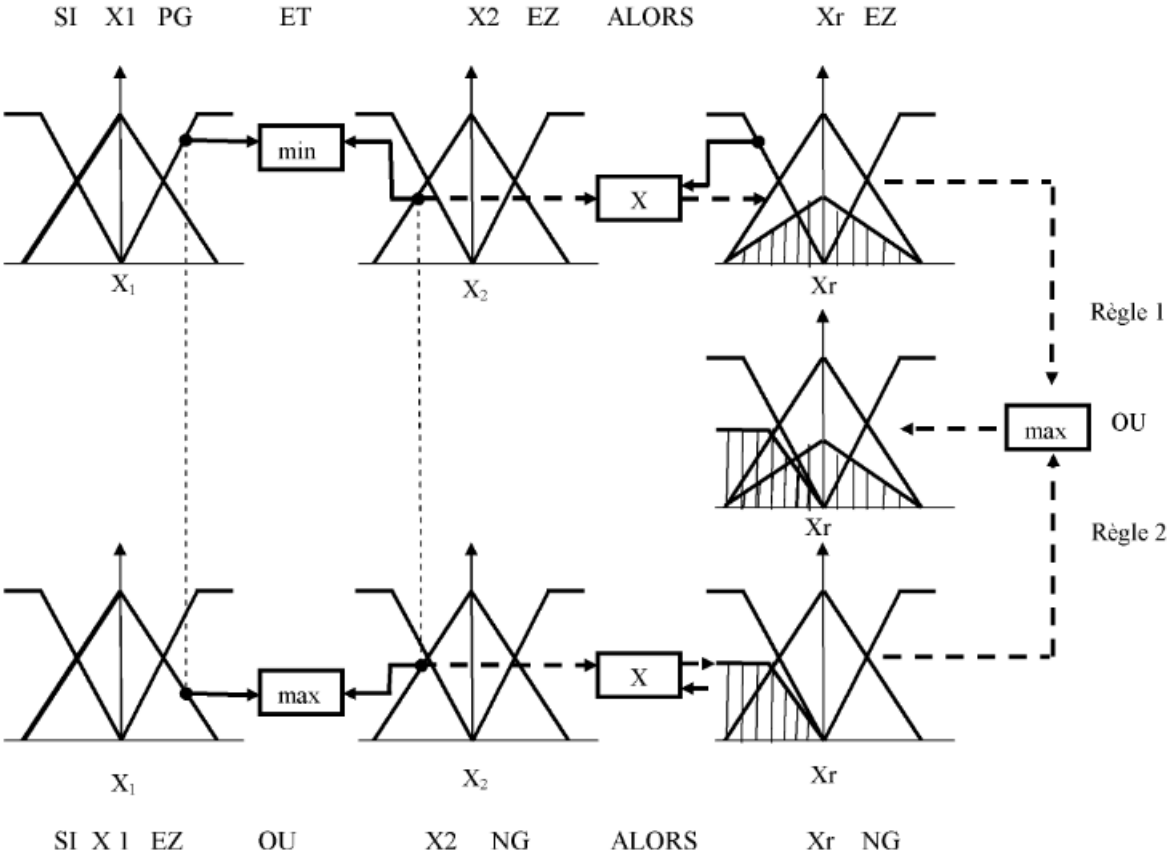


FIGURE 2.13 – Représentation graphique du principe de la méthode d'inférence max-prod.

3. **Méthode Somme-Prod (Sugeno) :** Dans ce cas, l'opérateur "**ET**" est réalisé par le produit, de même que la conclusion "**ALORS**". Cependant, l'opérateur "**OU**" est réalisé par la valeur moyenne des degrés d'appartenance intervenant dans l'inférence[15].
   La méthode d'inférence somme-prod est représentée graphiquement à la figure 2.14.

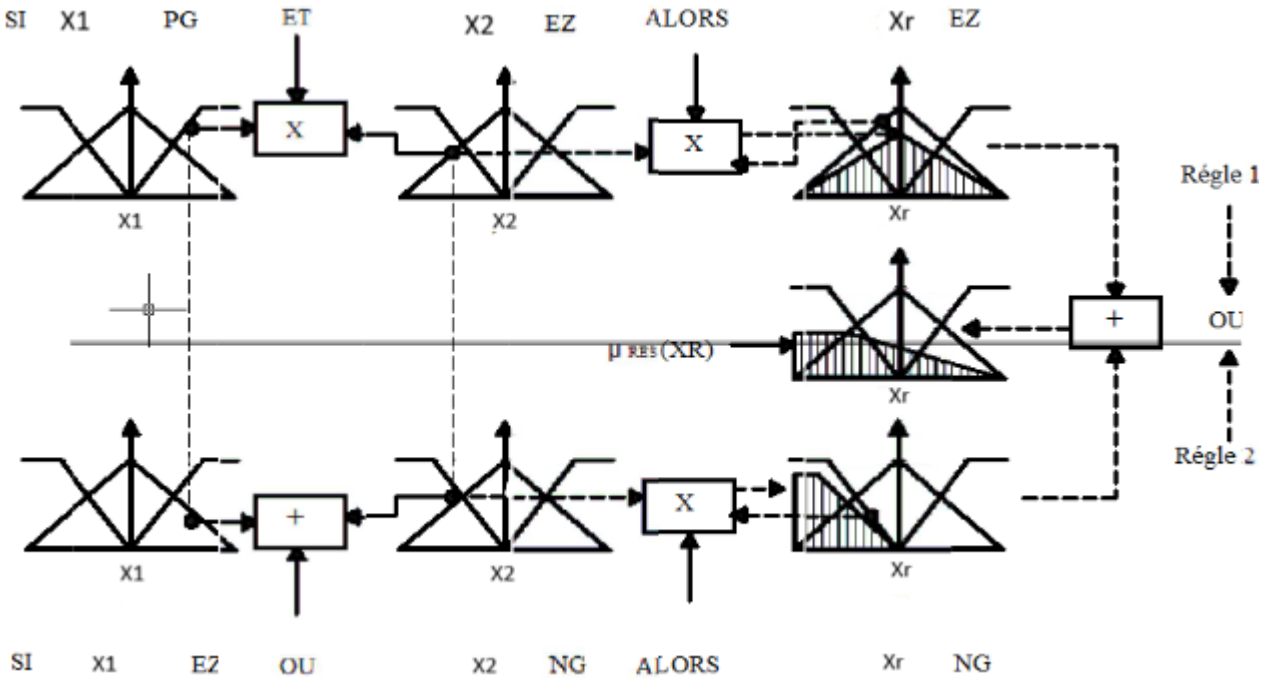


FIGURE 2.14 – Représentation graphique du principe de la méthode d'inférence Somme-Prod.

2.6.4 Défuzzification

L'étape de défuzzification consiste à transformer l'ensemble flou résultant de l'inférence de différentes règles en une valeur numérique unique, représentative de cet ensemble. Cette valeur numérique est ensuite appliquée pour commander un processus réel. En d'autres termes, il est nécessaire de passer du « monde flou » au « monde réel », et c'est précisément ce que fait la défuzzification[11].

Plusieurs stratégies de défuzzification existent, les plus utilisées sont :

— Méthode de centre de gravité.
— Méthode du maximum.
— Méthode de la moyenne des maximums.

1. **La Méthode de Centre de Gravité :** La défuzzification par la méthode de centre de gravité consiste à calculer l'abscisse du centre de gravité de la fonction d'appartenance résultante μ_r.

$$y^* = \frac{\int y.U_c(y)dy}{\int U_c dy} \tag{2.10}$$

Où :
y^* : est la valeur de sortie défuzzifiée.
$U_c(y)$: le degré d'appartenance.
y : une valeur de l'ensemble flou.

2. **Méthode de Moyenne des Maximums :** Cette méthode génère une commande précise en calculant la moyenne des valeurs pour lesquelles l'appartenance est maximale. La formule pour cette méthode est :

$$y^* = \frac{\sum_{i=1}^{n} y_i}{n} \tag{2.11}$$

Où :
y^* : est la valeur de sortie défuzzifiée.
y_i :sont les valeurs de sortie correspondantes aux points où la fonction d'appartenance atteint son maximum.
n : le nombre de ces valeurs maximales.

3. **Méthode de Moyenne Pondérée :** Cette méthode est convenable pour les ensembles flous avec des fonctions d'appartenance de sortie symétriques et produit des résultats très proches de la méthode **COG**. Cette méthode nécessite moins de calculs. La formule pour cette méthode est :

$$y^* = \frac{\sum y_i.U_c(y_i)}{\sum U_c(y_i)} \tag{2.12}$$

Où :
y^* : est la valeur de sortie défuzzifiée.
$U_c(y)$:C'est la valeur de la fonction d'appartenance U_c pour la sortie y_i, qui exprime le degré auquel chaque sortie y_i appartient à l'ensemble flou.
y_i : une valeur de l'ensemble flou.

Exemple :

Supposons que nous avons un système de contrôle flou pour ajuster la vitesse d'un ventilateur en fonction de la température ambiante.
Les ensembles flous pour la température sont :

Très Froid (TF) avec une valeur numérique de $10°C$.
Froid (F) avec une valeur numérique de $15°C$.
Tempéré (T) avec une valeur numérique de $20°C$.
Chaud (C) avec une valeur numérique de $25°C$.
Très Chaud (TC) avec une valeur numérique de $30°C$.

Imaginons maintenant que le système donne les degrés d'appartenance suivants pour une certaine température mesurée :

$$\mu 1 = 0.1 \text{ pour } TF$$
$$\mu 2 = 0.4 \text{ pour } F$$
$$\mu 3 = 0.7 \text{ pour } T$$
$$\mu 4 = 0.3 \text{ pour } C$$
$$\mu 5 = 0 \text{ pour } TC$$

En appliquant la méthode de la moyenne pondérée pour calculer la valeur défuzzifiée ,elle est calculée comme suit :

$$\mu_T = \frac{(0.1 \times 10) + (0.4 \times 15) + (0.7 \times 20) + (0.3 \times 25) + (0 \times 30) +}{0.1 + 0.4 + 0.7 + 0.3 + 0}$$

$$\mu_T = \frac{28.5}{1.5}$$

$$\mu_T = 19°C$$

La température défuzzifiée calculée est de $19°C$, ce qui signifie que le système détermine que la température ambiante est approximativement $Tempérée$, avec une légère influence des ensembles "$Froid$" et "$Chaud$". Cela montre comment le système combine les informations floues pour produire une sortie claire, en tenant compte des degrés d'appartenance des différents ensembles.

2.7 Avantages et Désavantages de la Commande Floue

Certainement, la commande par la logique floue comporte un certain nombre d'avantages et d'inconvénients.

Les avantages essentiels sont :

— La non-nécessité d'une modélisation détaillée du processus à commander ni une analyse mathématique approfondie.
— La possibilité de bénéficier et d'implémenter des connaissances et des expertises humaines sur le système à commander.
— La logique floue est particulièrement efficace pour gérer des systèmes complexes, fortement non linéaires et difficiles à modéliser.

En revanche, les inconvénients sont :

— le manque de méthodes systématiques précises pour la conception d'un réglage (choix des grandeurs à mesurer, détermination de la fuzzification , des inférences et de la défuzzification.
— L'impossibilité de démontrer la stabilité du circuit de commande en toute généralité.
— La cohérence des inférences n'est pas garantie à priori, ce qui peut entraîner l'apparition de règles d'inférence contradictoires.
— La possibilité d'apparition de cycles limites à cause de fonctionnement non-linéaire.

2.8 Conclusion

La logique floue nous a permis de contourner la difficulté du modèle mathématique en passant au modèle linguistique basé sur une expertise. L'architecture d'un système flou est déterminée par une meilleure compréhension des ensembles flous et des opérateurs flous. Nous avons constaté qu'il n'existe pas un seul type de système flou, mais il y en a plusieurs. Un utilisateur des systèmes flous doit décider sur le type des fonctions d'appartenance, le type des règles floues, la méthode du raisonnement flou et la stratégie de défuzzification. Vu la capacité de raisonnement humain et le traitement linguistique que présente la logique floue, son utilisation pour compenser les incertitudes s'avère très intéressante. C'est pour cette raison que nous avons choisi de l'intégrer dans la commande de notre architecture de téléopération position-position.

Chapitre 3

Description de la carte Arduino

3.1 Introduction

Dans ce troisième chapitre, nous nous concentrerons sur la présentation de la carte Arduino, qui joue un rôle central dans la réalisation de notre système de télé-opération. L'Arduino a été sélectionné pour sa capacité à assurer la communication efficace entre les différents composants du système, à savoir :

1. La partie commande (ordinateur).
2. Le site maître.
3. Le site esclave.

L'objectif était de pouvoir commander et piloter le système à distance, en permettant une communication bidirectionnelle entre ces différents éléments.

3.2 Historique

Arduino a commencé en Italie, chez Interaction Design Institute Ivera (IDII), une école spécialisée en design qui se focalise sur la manière dont nous interagissons avec les produits, les systèmes et les environnements numériques, et comment ces derniers nous influencent en retour.

Le terme de design d'interaction a été élaboré par **Bill Verplank** et **Bill Moggridge** vers le milieu des années 80. Le dessin de **Verkplank** illustre parfaitement le principe de base du design d'interaction : si vous faites quelque chose, vous ressentez un changement, et c'est à partir de là que vous pouvez savoir quelque chose du monde qui vous entoure.

C'est ainsi qu'en 2005, le projet Arduino a été lancé pour fournir un matériel bon

marché et facile à utiliser à destination des étudiants en design d'interaction. On dit que **Massimo Banzi** et **David Cuartielles** nommèrent leur projet d'après **Arduin d'Ivera**, un roi italien, mais des sources fiables indiquent que ce serait le nom d'un bar situé à proximité de l'université[17].

3.3 Définition de la carte Arduino

La carte Arduino est un circuit imprimé en matériel libre, les modules d'origine sont fabriqués par la société italienne SMART PROJECTS (dont les plans sont publiés en licence libre) sur lequel se trouve un microcontrôleur qui peut être programmer, afin d'analyser et de produire des signaux électriques.

Arduino est une plate-forme open source d'électronique programmée qui est basé sur une simple carte à microcontrôleur (de la famille AVR), et un logiciel, véritable environnement du développement intégré, pour écrire, compiler et transférer le programme vers la carte à microcontrôleur. Ces cartes sont basé sur interface entrée/sortie simple pouvant recevoir des entrées d'une grande variété d'interrupteurs ou de capteurs, et pouvant contrôler une grande variété de lumière, moteurs ou tout autre types d'actionneurs et tous cela se fait sur un environnement de développement proche du langage C.

3.4 Les principaux avantages du système Arduino

Le système Arduino offre plusieurs avantages significatifs qui le rendent accessible et attrayant pour les utilisateurs de tous niveaux. Ces avantages incluent [17] :

- **Un environnement de programmation clair et simple :** L'environnement de programmation Arduino (le logiciel Arduino IDE) est facile à utiliser pour les débutants, tout en étant assez flexible pour que les utilisateurs avancés puissent en tirer profit également.
- **Pas cher :** les cartes Arduino sont relativement peu coûteuses comparativement aux autres plateformes. La moins chère des versions du module Arduino peut être assemblée à la main,(les cartes Arduino pré-assemblées coûtent moins de 2500 DA).
- **Multi-plateforme :** Le logiciel Arduino, écrit en Java, tourne sous les systèmes d'exploitation Windows, Macintosh et Linux. La plupart des systèmes à microcontrôleurs sont limités à Windows.
- **Logiciel Open Source et extensible :** Le logiciel Arduino et le langage Arduino sont publiés sous licence open source, disponible pour être complétés par des programmateurs expérimentés. Le langage peut être aussi étendu à l'aide de librairies C++.
- **Matériel Open source et extensible :** on peut le copier, le fabriquer et le modifier librement. Les cartes Arduino sont basées sur les microcontrôleurs Atmel

ATMEGA8, ATMEGA168, ATMEGA 328, Les schémas des modules sont publiés sous une licence Creative Commons (CC). Les concepteurs de circuits expérimentés ou relativement inexpérimentés peuvent réaliser leur propre version des cartes Arduino, en les complétant et en les améliorant.

3.5 Réalisation du système de téléopération à base d'Arduino Due

Nous avons choisi la carte Arduino Due pour piloter notre système de téléopération en raison de ses capacités avancées, qui répondent parfaitement aux exigences de notre projet. Nous avons exploité 4 entrées numériques pour lire les signaux des encodeurs configurés en mode interruption, pour détecter les changements d'état, nous avons utilisé 3 sorties numériques PWM pour contrôler la vitesse des moteurs et gérer les signaux de commande via le module L298N.

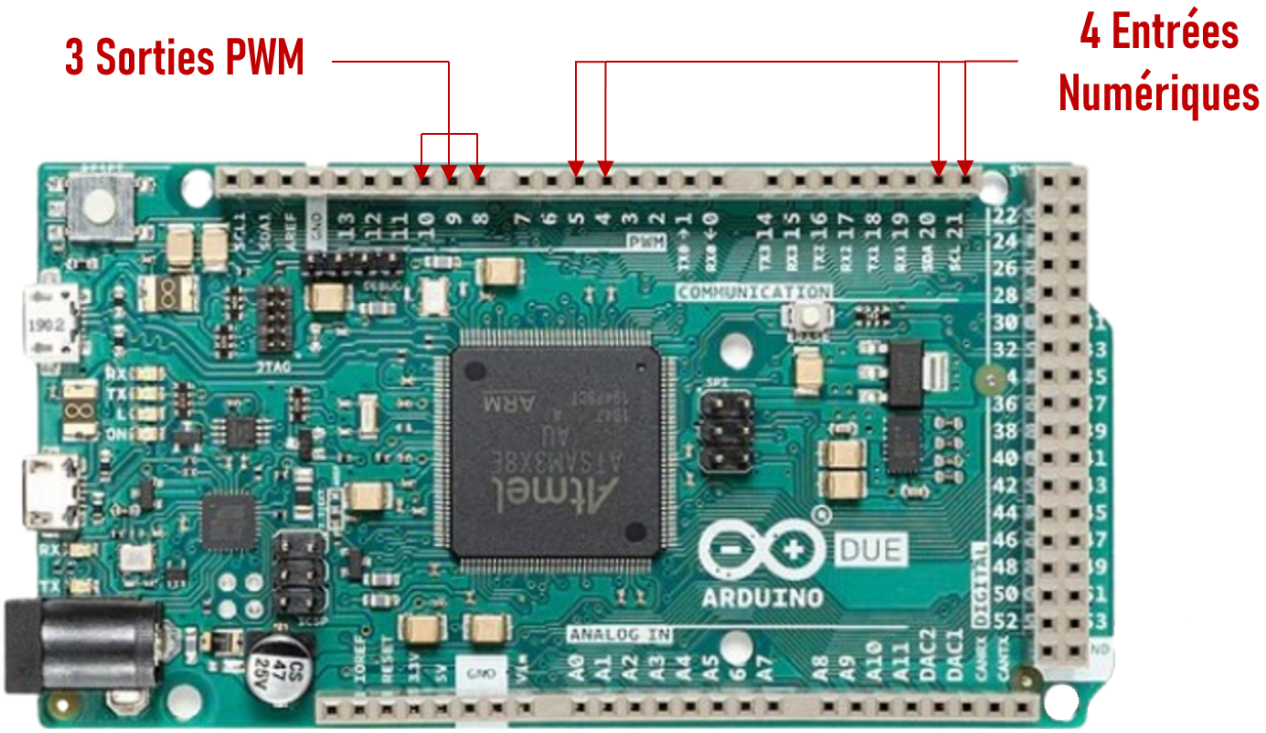


FIGURE 3.1 – Exploitation de notre carte.

3.6 Architecture de la carte Arduino Due

L'Arduino Due est une carte microcontrôleur basée sur le processeur Atmel SAM3X8E ARM Cortex-M3. Il s'agit de la première carte Arduino reposant sur un microcontrôleur à cœur ARM 32 bits. Elle dispose de 54 broches d'entrée/sortie numériques (dont 12 peuvent être utilisées comme sorties PWM), de 12 entrées analogiques, de 4 UARTs (ports série

matériels), d'une horloge de 84 MHz, d'une connexion USB OTG, de 2 DAC (convertisseurs numérique-analogique), de 2 interfaces TWI, d'une prise d'alimentation, d'un connecteur SPI, d'un connecteur JTAG, d'un bouton de réinitialisation et d'un bouton d'effacement.

Contrairement à la plupart des cartes Arduino, l'Arduino Due fonctionne à 3,3V. La tension maximale que les broches d'entrée/sortie peuvent supporter est de 3,3V. Appliquer une tension supérieure à 3,3V sur une quelconque broche d'entrée/sortie pourrait endommager la carte .

La figure 3.2 montre la carte Arduino Due que nous avons utilisée dans notre projet.

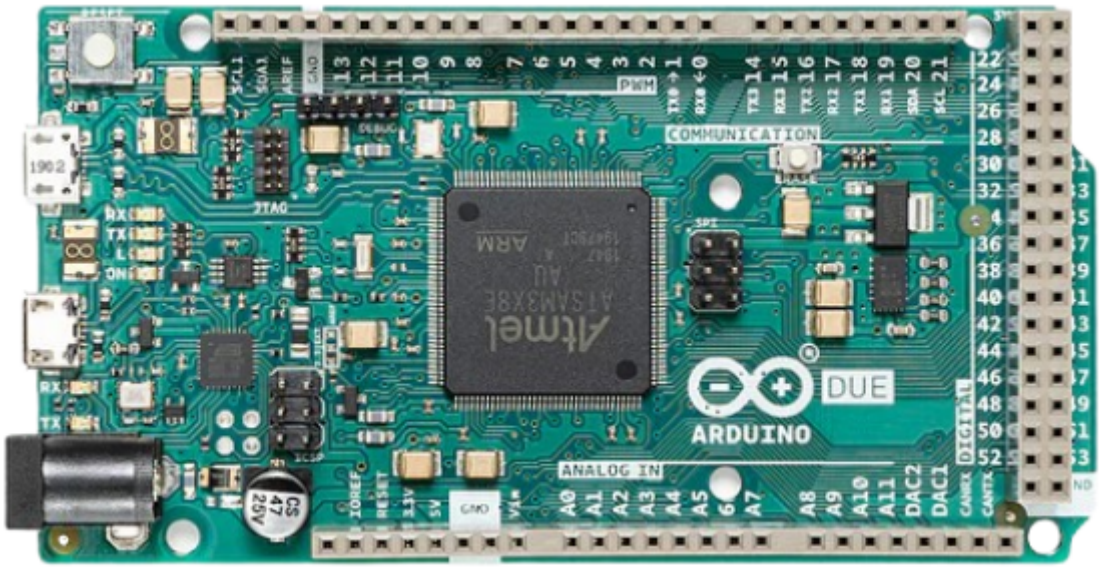


FIGURE 3.2 – Carte Arduino Due.

3.6.1 Alimentation :

L'Arduino Due peut être alimenté via le connecteur USB ou avec une alimentation externe. La source d'alimentation est sélectionnée automatiquement. L'alimentation externe (non-USB) peut provenir soit d'un adaptateur AC-DC (bloc secteur), soit d'une batterie. L'adaptateur peut être connecté en branchant une fiche positive de 2,1 mm au centre dans la prise d'alimentation de la carte. Les câbles d'une batterie peuvent être insérés dans les broches Gnd et Vin du connecteur POWER. La carte peut fonctionner avec une alimentation externe de 6 à 20 volts. Cependant, si elle reçoit moins de 7V, la broche 5V pourrait fournir moins de cinq volts, et la carte pourrait être instable. En utilisant plus de 12V, le régulateur de tension pourrait surchauffer et endommager la carte. La plage recommandée est de 7 à 12 volts[18].

Broches d'alimentation	Fonctions
Vin	La tension d'entrée de la carte Arduino lorsqu'elle utilise une source d'alimentation externe (par opposition aux 5 volts provenant de la connexion USB ou d'une autre source d'alimentation régulée). Vous pouvez fournir une tension via cette broche, ou si vous alimentez la carte via la prise d'alimentation.
5V	Cette broche fournit une tension régulée de 5V à partir du régulateur sur la carte. La carte peut être alimentée soit par la prise d'alimentation DC (7-12V), le connecteur USB (5V) ou la broche VIN de la carte (7-12V). Fournir une tension via les broches 5V ou 3,3V contourne le régulateur et peut endommager la carte
3.3V	Alimentation 3,3 volts générée par le régulateur intégré. Le courant maximal est de 800 mA. Ce régulateur fournit également l'alimentation au microcontrôleur SAM3X.
IOREF	Cette broche sur la carte Arduino fournit la référence de tension avec laquelle le microcontrôleur fonctionne. Un shield correctement configuré peut lire la tension de la broche IOREF et sélectionner la source d'alimentation appropriée ou activer des traducteurs de tension sur les sorties pour travailler avec du 5V ou du 3,3V
GND	Broches de masse (Ground).

TABLE 3.1 – Broches d'Alimentation et leurs Fonctions.

3.6.2 Mémoire

Le SAM3X dispose de 512 Ko (2 blocs de 256 Ko) de mémoire flash pour stocker le code. Le bootloader est préinstallé en usine par Atmel et est stocké dans une mémoire ROM dédiée. La SRAM disponible est de 96 Ko, répartie en deux banques contiguës de 64 Ko et 32 Ko. Toute la mémoire disponible (Flash, RAM et ROM) peut être accédée directement en tant qu'espace d'adressage plat.

Il est possible d'effacer la mémoire Flash du SAM3X à l'aide du bouton d'effacement intégré. Cela supprimera le sketch actuellement chargé dans le MCU. Pour effacer, maintenez enfoncé le bouton d'effacement pendant quelques secondes pendant que la carte est alimentée[18].

3.6.3 Broches d'entrée et de sortie :

I/O numériques : broches de 0 à 53

Chaque broche numérique de l'Arduino Due peut être utilisée comme entrée ou sortie via les fonctions pinMode(), digitalWrite() et digitalRead(). Elles fonctionnent à 3,3 volts. Chaque broche peut fournir (source) un courant de 3 mA ou 15 mA, selon la broche, ou

recevoir (sink) un courant de 6 mA ou 9 mA, selon la broche. Elles possèdent également une résistance pull-up interne de 100 KOhm (déconnectée par défaut). Certaines broches ont des fonctions spécialisées :

Séries : 0 (RX) et 1 (TX)

Série 1 : 19 (RX) et 18 (TX)

Série 2 : 17 (RX) et 16 (TX)

Série 3 : 15 (RX) et 14 (TX)

Utilisées pour recevoir (RX) et transmettre (TX) des données série TTL (avec un niveau de 3,3 V). Les broches 0 et 1 sont connectées aux broches correspondantes du circuit ATmega16U2 USB-to-TTL Serial.

PWM : broches de 2 à 13 Ces broches fournissent une sortie PWM 8 bits avec la fonction analogWrite(). La résolution du PWM peut être modifiée avec la fonction analogWriteResolution().

SPI : connecteur SPI (connecteur ICSP sur d'autres cartes Arduino) Ces broches prennent en charge la communication SPI à l'aide de la bibliothèque SPI. Elles sont exposées sur un connecteur central à 6 broches, compatibles physiquement avec les cartes Uno, Leonardo et Mega2560. Le connecteur SPI ne peut être utilisé que pour la communication avec d'autres périphériques SPI, pas pour programmer le SAM3X avec la technique In-Circuit-Serial-Programming (ICSP). Le SPI du Due dispose également de fonctionnalités avancées qui peuvent être utilisées avec les méthodes SPI étendues.

CAN : CANRX et CANTX

Ces broches prennent en charge le protocole de communication CAN, mais ne sont pas encore supportées par les APIs Arduino.

LED "L" : 13

Une LED intégrée est connectée à la broche numérique 13. Lorsque la broche est HIGH, la LED est allumée ; lorsqu'elle est LOW, la LED est éteinte. Il est également possible de moduler l'intensité lumineuse de la LED, car la broche 13 est une sortie PWM.

TWI 1 : 20 (SDA) et 21 (SCL).

TWI 2 : SDA1 et SCL1

Prend en charge la communication TWI avec la bibliothèque Wire. Les broches SDA1 et SCL1 peuvent être contrôlées à l'aide de la classe Wire1 de la bibliothèque Wire. Les broches SDA et SCL ont des résistances pull-up internes, mais pas SDA1 et SCL1. Il est nécessaire d'ajouter deux résistances pull-up pour utiliser Wire1.

Entrées analogiques : broches de A0 à A11

L'Arduino Due dispose de 12 entrées analogiques, chacune offrant une résolution de 12 bits (c'est-à-dire 4096 valeurs différentes). Par défaut, la résolution des lectures est définie sur 10 bits pour être compatible avec d'autres cartes Arduino. Il est possible de changer la résolution de l'ADC avec analogReadResolution(). Les broches analogiques du Due mesurent entre la masse et une valeur maximale de 3,3V. Appliquer plus de 3,3V sur ces broches peut endommager la puce SAM3X. La fonction analogReference() est ignorée sur le Due. La broche AREF est connectée à la référence analogique du SAM3X via un pont de résistance. Pour utiliser la broche AREF, la résistance BR1 doit être dessoudée de la carte.

DAC1 et DAC2 : Ces broches fournissent de véritables sorties analogiques avec une résolution de 12 bits (4096 niveaux) grâce à la fonction analogWrite(). Elles peuvent être utilisées pour générer une sortie audio à l'aide de la bibliothèque Audio.

Autres broches sur la carte :

AREF : Tension de référence pour les entrées analogiques. Utilisée avec analogReference().

Reset : Baissez cette ligne à LOW pour réinitialiser le microcontrôleur. Utilisée principalement pour ajouter un bouton de réinitialisation aux shields qui bloquent celui de la carte.

3.6.4 Communication :

L'Arduino Due dispose de plusieurs fonctionnalités pour communiquer avec un ordinateur, un autre Arduino ou d'autres microcontrôleurs, ainsi qu'avec divers dispositifs tels que téléphones, tablettes, caméras, etc. Le SAM3X offre un UART matériel et trois USARTs matériels pour la communication série TTL (3,3V)[18].

Port de programmation : Il est connecté à un ATMega16U2 qui fournit un port COM virtuel au logiciel sur un ordinateur connecté. (Pour reconnaître l'appareil, les machines Windows nécessitent un fichier .inf, mais les machines OSX et Linux reconnaissent automatiquement la carte en tant que port COM). Le 16U2 est également connecté à l'UART matériel du SAM3X. La communication série sur les broches RX0 et TX0 fournit la communication Serial-to-USB pour programmer la carte via le microcontrôleur ATmega16U2. Le logiciel Arduino inclut un moniteur série qui permet d'envoyer et de recevoir des données textuelles simples à la carte et depuis celle-ci. Les LED RX et TX sur la carte clignotent lors de la transmission de données via la puce ATmega16U2 et la connexion USB à l'ordinateur (mais pas pour la communication série sur les broches 0 et 1).

Port USB natif : Ce port est connecté directement au SAM3X et permet une communication série (CDC) via USB. Il fournit une connexion série au moniteur série ou à d'autres applications sur votre ordinateur. Il permet également à l'Arduino Due d'émuler une souris ou un clavier USB connectés à un ordinateur. Pour utiliser ces fonctionnalités, consultez les pages de référence des bibliothèques Mouse et Keyboard.

Le port USB natif peut également agir comme hôte USB pour des périphériques connectés, tels que des souris, des claviers et des smartphones.

Protection contre les surintensités USB :

L'Arduino Due est équipé d'un fusible réarmable (polyfuse) qui protège les ports USB de votre ordinateur contre les courts-circuits et les surintensités. Bien que la plupart des ordinateurs disposent de leur propre protection interne, ce fusible ajoute une couche supplémentaire de sécurité. Si plus de 500 mA sont appliqués au port USB, le fusible coupera automatiquement la connexion jusqu'à ce que le court-circuit ou la surcharge soit éliminé.

3.6.5 Les caractéristiques de microcontrôleur Atmel SAM3X

Voici ses principales caractéristiques [18] :

— 84 instructions puissantes, la plupart à exécution en un seul cycle.
— 32 registres d'usage général de 32 bits.
— Opération entièrement statique.
— Jusqu'à 84 MIPS de sorties à 84 MHz.
— Mémoire Flash non-volatile de programme jusqu'à 512 Ko.
— Résistance : 100.000 cycles d'écriture/effacement pour la mémoire flash.
— 96 Ko de SRAM répartis en deux banques contiguës.
— Serrure de programmation pour la sécurité du logiciel.
— 16 canaux ADC de 12 bits et 2 canaux DAC de 12 bits.
— Contrôle PWM sur plusieurs canaux.
— Modes basse consommation avec gestion flexible de l'énergie (Sleep, Wait, Backup).

Le Microcontrôleur Atmel SAM3X est donné par cette figure ci-dessous (Figure 3.3) :

FIGURE 3.3 – Le Microcontrôleur Atmel SAM3X.

3.7 Architecture SOFTWARE de la carte

L'Arduino IDE (Environnement de Développement Intégré) est le logiciel clé pour programmer les cartes Arduino. Cet outil open-source permet d'écrire et de télécharger du code sur votre carte Arduino. Il est compatible avec Windows, Mac OS, Linux, et même les Chromebooks via l'éditeur en ligne.

3.7.1 Présentation de l'Espace de développement Intégré (EDI) Arduino

La carte Arduino constitue le noyau de notre système. Elle est programmée à l'aide d'un logiciel compatible appelé Arduino IDE (Environnement de Développement Intégré), une application écrite en Java et inspirée du langage Processing. L'IDE permet d'écrire, de modifier un programme et de le convertir en une série d'instructions compréhensibles pour la carte.

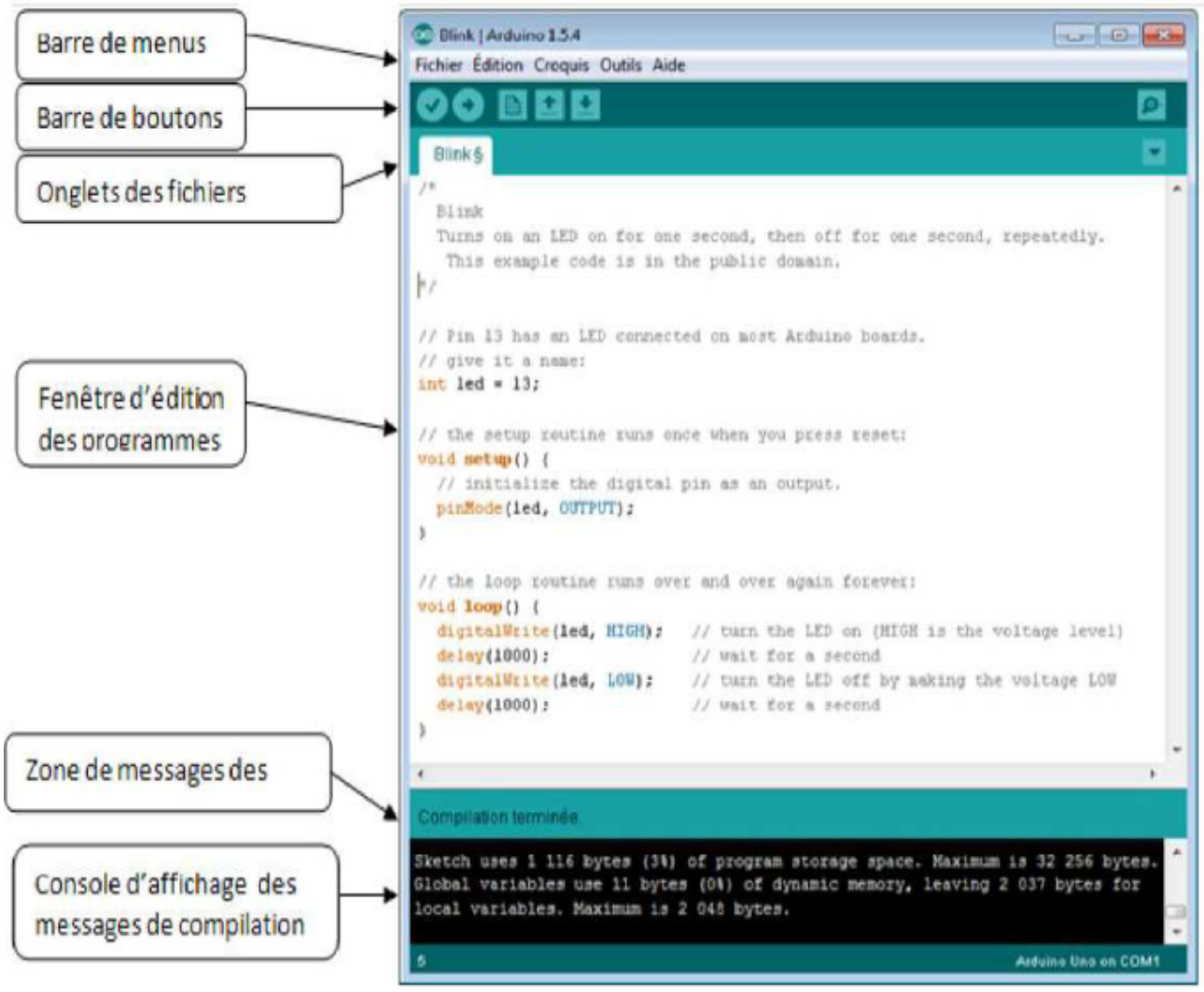
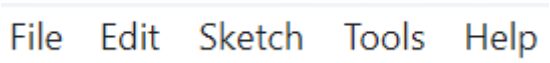


FIGURE 3.4 – Les différentes parties de la fenêtre principale du logiciel Arduino.

3.7.2 Différentes composition de l'espace de développement intégré(EDI)

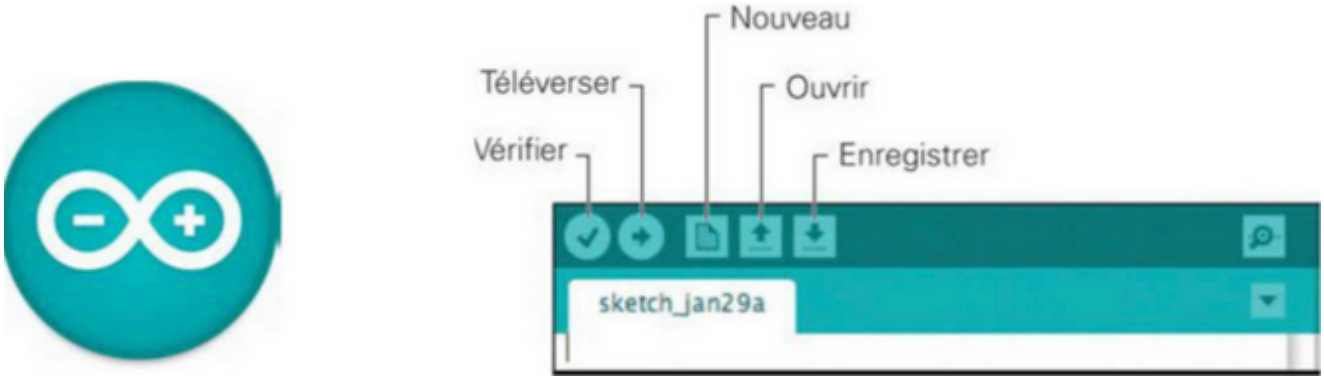
Une barre de menus :

Dans la barre de menus, vous trouverez les différents menus grâce auxquels vous pourrez appeler certaines fonctions de l'IDE.



- **File (Fichier) :** ce menu contient les différentes options de créations, d'ouverture de sauvegarde, d'impression du programme, ou l'ouverture d'un exemple parmi les exemples qui accompagnent le logiciel Arduino.
- **Edit (Éditer) :** ce menu contient les options de copier/coller, sélection, et les options de recherche.
- **Sketch (Programme ou séquence) :** ce menu contient les différentes fonctions de la barre des boutons, ainsi que les options d'ajouts de librairies ou de fichiers.
- **Tools (Outils) :** c'est dans ce menu qu'on sélectionne le type de carte à programmer,et le port série utilisé.
- **Help (Aide) :** ce menu est fait pour donner de l'aide concernant les différents problèmes rencontrés au niveau du logiciel Arduino.

Une barre des boutons :



- **Vérifier/compiler :** Vérifie le code à la recherche d'erreur.
- **Téléverser :** Compiler votre code et le transférer vers la carte Arduino.
- **Nouveau :** Crée un nouveau code (ouvre une nouvelle fenêtre)
- **Ouvrir :** Ouvre la liste de tous les programmes dans" le livre de programmes".
- **Enregistrer :** Enregistrer un programme.
- **Moniteur Série :** Ouvre la fenêtre du moniteur (ou terminal) série.

Structure d'un programme Arduino :

Éditeur : L'espace utilisé pour écrire le programme à réaliser, avec des onglets pour naviguer entre différents fichiers.

Zone de messages : Utilisée par l'IDE pour fournir à l'utilisateur des informations sur l'état du programme et les actions en cours.

Console texte : Affiche les messages relatifs au résultat de la compilation, indiquant si le programme contient des erreurs.

3.7.3 Le programme Arduino :

3.7.3.1 Description de la structure d'un programme arduino

Le langage Arduino est basé sur les langages C/C++, permet d'écrire des programmes sous forme de suites d'instructions textuelles, ligne par ligne. La carte Arduino lit et exécute ces instructions séquentiellement, dans l'ordre où elles sont écrites, comme dans une programmation classique. La structure d'un programme Arduino se divise en trois parties principales :

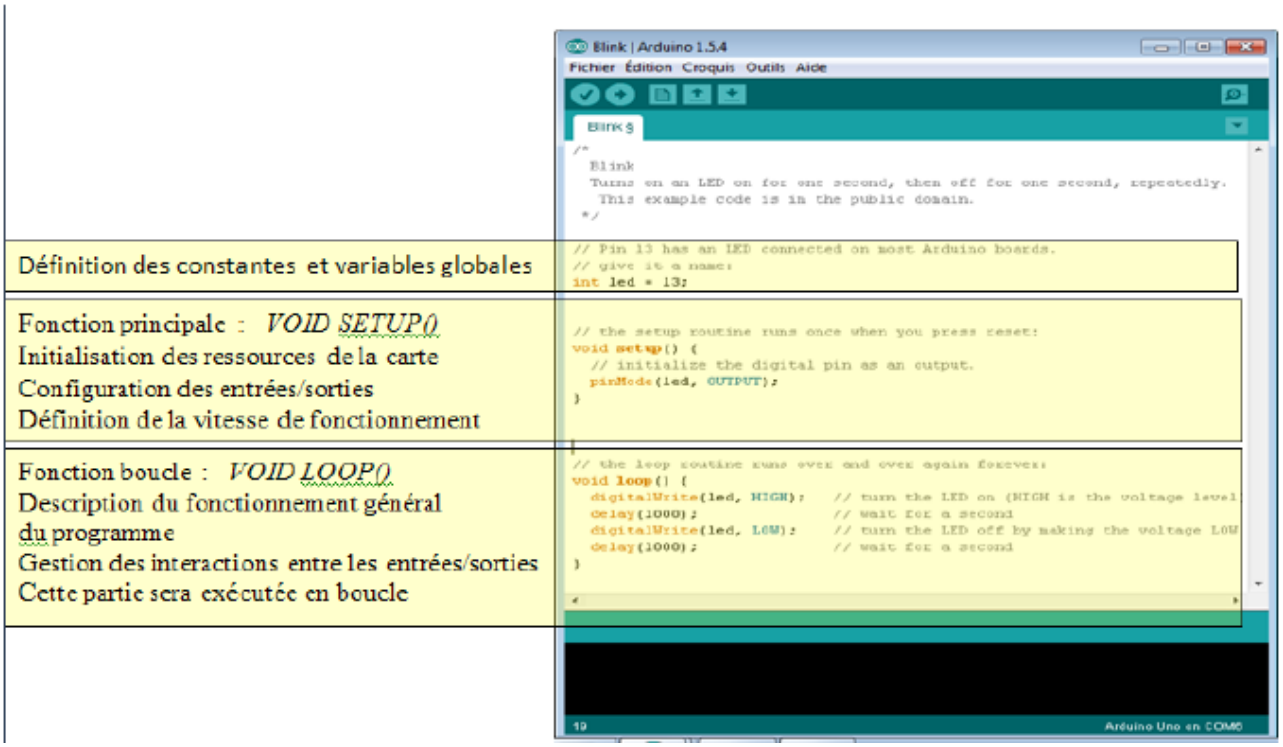


FIGURE 3.5 – Les différentes composantes du programme Arduino.

3.7.3.2 Le jeu d'instructions du langage Arduino :

Les syntaxes du programme :

— Chaque instruction se termine par un « ; ».
— Les accolades « { » et « } » sont les "conteneurs" du code du programme. Elles sont propres aux fonctions, aux conditions et aux boucles. Les instructions du programme sont écrites à l'intérieur de ces accolades.
— Les commentaires sont des lignes de texte incluses dans le programme et qui ont pour but de vous informer vous-même ou les autres de la façon dont le programme fonctionne. Ces lignes ajoutées sont ignorées par le compilateur. Les commentaires sont précédés des caractères « // » ou bien encadrés par « /* » et « */ » .
— Il est formellement interdit de mettre des accents en programmation, sauf dans les commentaires.
— Un nombre en binaire doit être précédé de la lettre « B ».
— Un nombre écrit en hexadécimal doit être précédé par les caractères « 0x ».

Commandes du programme Arduino :

Structure :

Sketch	void setup()
	void loop()
Contrôle et conditions	if
	if...else
	for
	switch case
	while
Opérateurs de comparaison	== (équivalent à)
	! = (différent de)
	< (Inférieur à)
	> (Supérieur à)
	<= (inférieur ou égal à)
	>= (supérieur ou égal à)
Opérateurs booléennes	&&(Et)
	|| (ou)
	! (Et pas)

Variables :

Constantes	HIGH (état 1)
	LOW (état 0)
	INPUT (configuré en entrée)
	OUTPUT (configuré en sortie)
	INPUT-PULLUP
Types de Données	char (variable 'caractère')
	int (variable 'nombre entier')
	long (variable 'nombre entier de très grande taille')
	string (variable 'chaine de caractères')
	array (tableau de variables)

Fonction :

Entrées-sorties numériques	pinMode (broche, état)	configuration des broches en entré ou en sortie
	digitalWrite (broche, état)	écrire un état sur une broche numérique
	digitalRead (broche)	lire un état sur une broche numérique
	unsigned long pulseIn (broche, état)	lire une impulsion sur une broche numérique
Entrées analogiques	int analogRead (broche)	lire la valeur d'une broche analogique
	analogWrite (broche, valeur)	PWM : écrire une valeur analogique sur les broches
Gestion du temps	unsigned long millis()	temps de fonctionnement du programme
	delay(ms)	attente, en millisecondes
	delayMicroseconds(us)	attente, en microsecondes

3.7.3.3 Les étapes à suivre pour programmer la carte

1. Écriture de programme :
   cette première étape consiste à saisir le programme

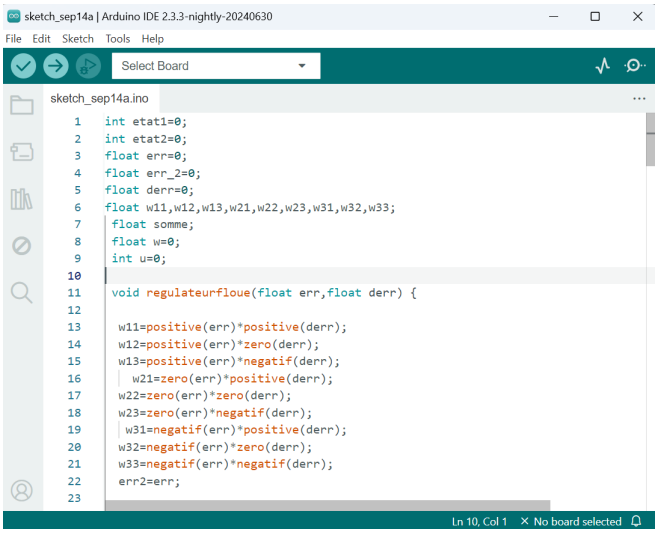


FIGURE 3.6 – l'écriture du programme

2. Compilation du programme :
   On vérifie ici les erreurs de syntaxe dans le code. Si le programme a des anomalies, le compilateur indique le type d'erreur et la ligne concernée. Pour les corriger, consultez l'aide dans la barre de menus.

FIGURE 3.7 – Compilation du Programme

Si aucune erreur n'est détectée, le programme est compilé et le message « Done compiling » apparaît, et affiche la taille du programme.

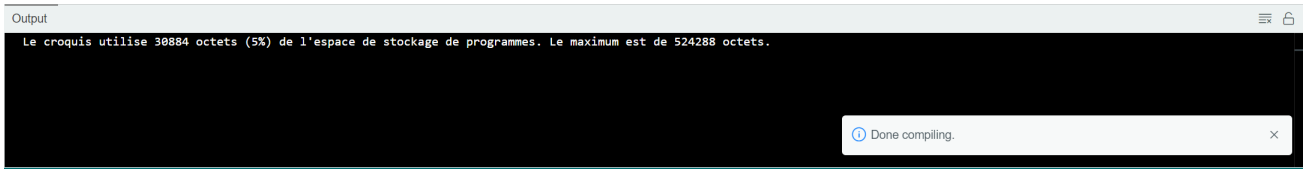


FIGURE 3.8 – console d'affichage

3. Sélection type de carte :
   Avant de transférer le programme vers la carte Arduino, il faut sélectionner le type carte Arduino (cible) depuis le menu Tools > Board. La carte doit évidemment être connectée à l'ordinateur via un câble USB. Vérifions que c'est bien le nom " Arduino Due (Programming Port) " qui est coché. Si ce n'est pas le cas, on le coche.

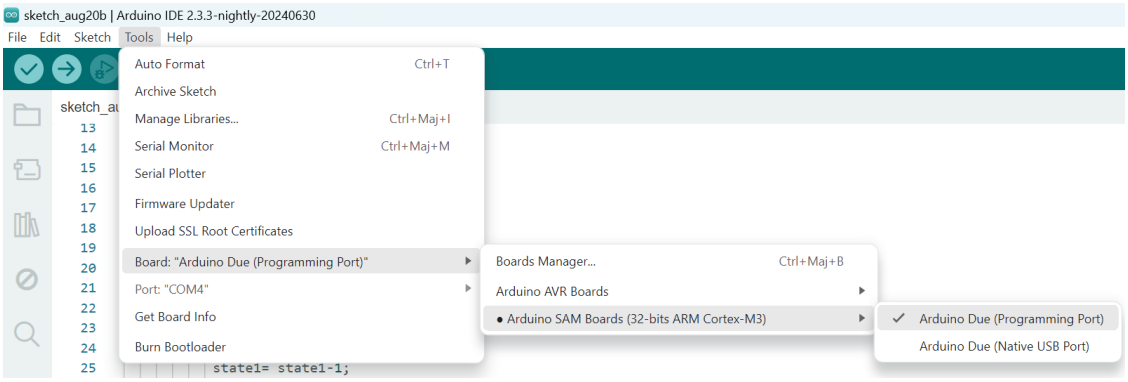


FIGURE 3.9 – Sélection de la cible

4. Sélection du port :
   Sélectionner le port série sur lequel est branché cette carte (depuis le menu Tools > Port)

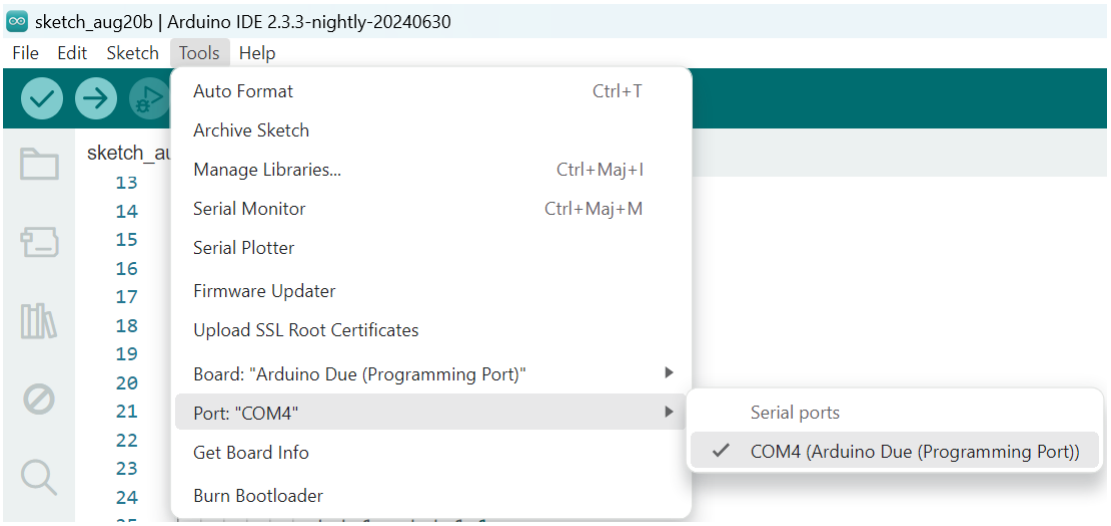


FIGURE 3.10 – Sélection du port

5. Transfert du programme :
   Cela se fait en cliquant sur le bouton.

FIGURE 3.11 – Transfert du programme

Le programme est téléchargé vers la carte, le message « Done uploading » s'affiche une fois l'opération terminée sur la plupart des cartes, nous devons voir les LEDs des lignes RX et TX clignoter rapidement, témoignant que le programme

est bien transféré. Durant le transfert, le bouton devient jaune et le logiciel Arduino affiche un message indiquant que le transfert est en cours.

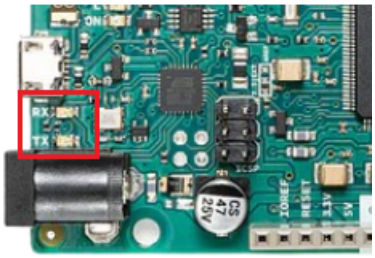


FIGURE 3.12 – Clignotement des deux LED du port série.

3.8 Conclusion

Ce chapitre a offert une vue d'ensemble de la carte Arduino en abordant son historique, ses aspects matériels, ainsi que le logiciel et le langage de programmation associés. Grâce à l'Arduino, nous avons pu développer un système de téléopération fonctionnel et flexible, parfaitement adapté aux besoins du projet. Sa polyvalence et sa simplicité de programmation en font un choix judicieux pour ce type d'application. La compréhension de ces éléments constitue une base solide pour exploiter efficacement la carte dans notre projet.

Chapitre 4

Application d'une stratégie de commande floue à l'architecture de télé opération développée

4.1 Introduction

Dans ce chapitre, nous nous concentrons sur l'application pratique d'une stratégie de commande floue pour contrôler le système de téléopération que nous avons développé. L'objectif est de mettre en œuvre un régulateur flou adapté à notre architecture, en tenant compte des spécificités du système et des défis rencontrés, notamment le retard fixe de transmission entre le robot maître et le robot esclave.

Nous commencerons par définir l'univers de discours, qui constitue le cadre dans lequel les variables du système sont exprimées. Nous examinerons ensuite les entrées du régulateur, en détaillant les choix que nous avons faits pour l'erreur actuelle et sa dérivée. La sortie du régulateur sera également décrite, ainsi que les fonctions d'appartenance associées à chaque variable.

Les règles d'inférence qui déterminent le comportement du contrôleur flou en fonction des différentes combinaisons d'entrées seront détaillées. Par la suite, Nous aborderons ensuite la méthode de défuzzification choisie, en expliquant comment elle permet de convertir les résultats flous en une commande précise et exploitable.

Après avoir décrit la conception et la mise en œuvre du régulateur flou, nous discuterons des matériaux utilisés dans notre application. Enfin, nous présenterons les simulations réalisées et les résultats expérimentaux obtenus, en les interprétant pour évaluer les performances de poursuite et l'efficacité de la stratégie de commande floue mise en œuvre.

4.2 Structure du régulateur flou

La structure du régulateur flou est composée de quatre étapes essentielles : la fuzzification, l'établissement des règles floues, l'inférence et la défuzzification. Dans le cadre de notre projet, nous allons explorer chacune de ces étapes afin de développer un contrôleur flou capable de contrôler la position du robot esclave tout en prenant en compte les retards fixes de transmission de données entre le robot maître et le robot esclave.

4.2.1 Univers de discours

Comme mentionné dans le deuxième chapitre, l'univers de discours représente l'intervalle sur lequel s'étend la variable linguistique. Dans notre cas, cet intervalle est défini de [−1,1].

4.2.2 Entrées du régulateur

Pour concevoir notre régulateur, nous commençons par sélectionner les entrées appropriées. Généralement , dans un système de régulation de ce type, l'erreur de position e est l'entrée principale du régulateur, tandis que la commande u est la sortie. Cependant, pour améliorer les performances et obtenir des résultats plus précis, nous ajoutons la dérivée de l'erreur *de* comme deuxième entrée. Cette approche permet au régulateur de disposer de plus d'informations, ce qui améliore le contrôle de la commande et augmente la précision du système.

— **Erreur de position e :**

La différence entre la position souhaitée (référence) et la position actuelle du moteur. Cette variable indique si la position actuelle du moteur esclave est en retard ou en avance par rapport à la position souhaitée du moteur maître.

Une erreur positive signifie que la position actuelle est inférieure à la position de consigne, ce qui nécessite un ajustement pour rattraper le retard et aligner la position de l'esclave avec celle du maître.

Une erreur négative indique que la position actuelle est supérieure à la position de consigne, ce qui implique une correction pour réduire l'avance et synchroniser la position du moteur esclave avec celle du maître.

— **Variation de l'Erreur de position de :**

La dérivée de l'erreur représente la variation de l'erreur de position au fil du temps, c'est-à-dire à quelle vitesse l'écart entre la position actuelle du moteur esclave et la position de consigne du moteur maître change. Cette variable indique si l'erreur de position augmente, diminue, ou reste constante.

Une variation positive de l'erreur de position signifie que l'erreur de position augmente avec le temps. Cela indique que l'écart entre la position actuelle du moteur esclave et la position de consigne du moteur maître s'aggrave, ce qui signifie que le moteur esclave est en retard et que ce retard continue de croître.
Une variation négative de l'erreur de position indique que l'erreur de position diminue, ce qui signifie que le moteur esclave se rapproche de la position de consigne du moteur maître et que l'écart entre les deux positions se réduit.
En résumé, la dérivée de l'erreur de position indique si l'écart entre les positions du maître et de l'esclave s'améliore ou s'aggrave avec le temps. Elle est cruciale pour ajuster les commandes et améliorer la synchronisation entre les deux moteurs.

4.2.3 Sortie du régulateur

Dans notre système de régulation de position basé sur la logique floue, la sortie du régulateur est déterminée par un processus de défuzzification. Ce processus permet de transformer les résultats flous obtenus des inférences en une seule valeur précise qui guidera l'ajustement de la position du robot esclave.

Pour notre application, nous avons défini cinq constantes pour couvrir toutes les situations possibles (position positive, négative ou nulle), ainsi que deux valeurs intermédiaires pour une meilleure précision.

Les différentes constantes de sortie sont définies comme suit :

.GN(Grand Négatif) = -90
.N (Négatif) = -75
.Z (Zero) = 0
.P (Positif) = 75
.GP (Grand Positif) = 90

Après avoir calculé la valeur de la sortie, elle est ajoutée à la commande de l'itération précédente pour déterminer la nouvelle commande. Ainsi :

— Une sortie positive peut indiquer qu'il est nécessaire d'ajuster la position pour se déplacer davantage dans une direction spécifique.
— Une sortie négative peut indiquer qu'il est nécessaire de réduire la position ou de corriger le mouvement pour se rapprocher de la position cible.

Cette approche permet de moduler finement le contrôle de la position du robot esclave en fonction des écarts détectés entre la position actuelle et la position cible, tout en prenant en compte les changements dynamiques nécessaires pour maintenir la précision et la stabilité du système.

4.2.4 Fonctions d'appartenance

Pour définir les fonctions d'appartenance des variables d'entrée appropriées à notre système, nous nous basons sur l'expertise pratique. Nous ajustons ensuite ces fonctions en fonction des résultats expérimentaux. Pour chaque variable, nous créons plusieurs ensembles flous pour couvrir toutes les situations possibles. Ces ensembles sont affinés pour s'assurer qu'ils reflètent correctement les conditions du système et optimisent ainsi son contrôle et ses performances.

— **Erreur de position e :**

N (Négative)

$$\mu_N(x) = \begin{cases} 1 & si \quad x \leq -1 \\ -x & si \quad -1 \prec x \prec 0 \\ 0 & si \quad x \geq 0 \end{cases} \tag{4.1}$$

Z (Zéro)

$$\mu_Z(x) = \begin{cases} 0 & si \quad x \leq -1 \; ou \; x \geq 1 \\ 1+x & si \quad -1 \prec x \leq 0 \\ 1-x & si \quad 0 \prec x \prec 1 \end{cases} \tag{4.2}$$

P (Positive)

$$\mu_P(x) = \begin{cases} 0 & si \quad x \leq 0 \\ x & si \quad 0 \prec x \prec 1 \\ 1 & si \quad x \geq 1 \end{cases} \tag{4.3}$$

— **Variation de l'Erreur de position de :**

Nous appliquons le même raisonnement pour le choix des fonctions pour de.
N (Négative)

$$\mu_N(x) = \begin{cases} 1 & si \quad x \leq -1 \\ -x & si \quad -1 \prec x \prec 0 \\ 0 & si \quad x \geq 0 \end{cases} \tag{4.4}$$

Z (Zéro)

$$\mu_Z(x) = \begin{cases} 0 & si \quad x \leq -1 \; ou \; x \geq 1 \\ 1+x & si \quad -1 \prec x \leq 0 \\ 1-x & si \quad 0 \prec x \prec 1 \end{cases} \tag{4.5}$$

P (Positive)

$$\mu_P(x) = \begin{cases} 0 & si \quad x \leq 0 \\ x & si \quad 0 \prec x \prec 1 \\ 1 & si \quad x \geq 1 \end{cases} \tag{4.6}$$

Le chevauchement des fonctions d'appartenance permet une commande plus fluide et robuste. Les fonctions N et P sont représentées par des fonctions trapézoïdales, tandis que la fonction Z est triangulaire.
La figure ci-dessous illustre les fonctions d'appartenance des variables e et de.

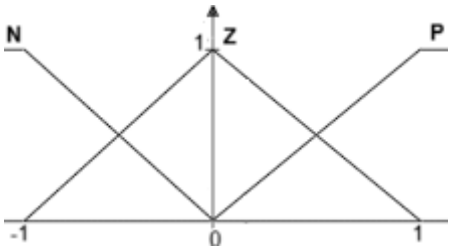


FIGURE 4.1 – fonctions d'appartenance e et de.

4.2.5 Règles d'inférence

Dans le système de régulation floue présenté, deux variables d'entrée sont considérées : l'erreur actuelle e et la variation de l'erreur de. Chacune de ces variables est associée à trois fonctions d'appartenance, à savoir positive (P), zéro (Z), et négative (N). Les règles d'inférence de notre système sont conçues pour couvrir toutes les combinaisons possibles de ces états d'appartenance. En conséquence, un total de neuf règles est établi pour décrire le comportement du contrôleur flou.

Ces règles sont formulées comme suit :

Si (e est N) ET (de est N) *Alors* (u est GN)
Si (e est N) ET (de est Z) *Alors* (u est N)
Si (e est N) ET (de est P) *Alors* (u est Z)

Si (e est Z) ET (de est N) *Alors* (u est N)
Si (e est Z) ET (de est Z) *Alors* (u est Z)
Si (e est Z) ET (de est P) *Alors* (u est P)

Si (e est P) ET (de est N) *Alors* (u est Z)
Si (e est P) ET (de est Z) *Alors* (u est P)

$$Si \quad \text{(e est P) ET (de est P)} \quad Alors \quad \text{(u est GP)}$$

La matrice d'inférence est la suivante :

U		dE		
		P	Z	N
E	P	GP	P	Z
	Z	P	Z	N
	N	Z	N	GN

TABLE 4.1 – Matrice d'inférence

4.2.6 Défuzzification :

La défuzzification peut se faire par diverses méthodes, mais nous allons nous concentrer ici sur la méthode de la moyenne pondérée, largement utilisée pour sa simplicité et son efficacité.

Voici donc sa formule mathématique :

$$y^* = \frac{\sum y_i . U_c(y_i)}{\sum U_c(y_i)} \tag{4.7}$$

Où :
y^* : est la valeur de sortie défuzzifiée.
$U_c(y)$: le degré auquel chaque y_i appartient à l'ensemble flou.
y_i : une valeur de l'ensemble flou.

Cette méthode consiste à pondérer chaque valeur de sortie des fonctions d'appartenance par son degré d'appartenance respectif. Ensuite, une moyenne pondérée est calculée pour obtenir la valeur de sortie défuzzifiée.

4.3 Description des Composants Matériels

4.3.1 Circuit L298N

Ce pilote de moteur bidirectionnel est basé sur le circuit intégré très populaire de moteur à double pont en H L298. Le circuit vous permet de contrôler facilement et indépendamment des moteurs jusqu'à 2A chacun dans les deux sens. Il est idéal pour les applications robotiques et bien adapté pour se connecter à un microcontrôleur qui ne nécessite que quelques lignes de commande du moteur. Il est conçu pour supporter des tensions et des courants plus élevés tout en fournissant une commande logique (tension plus faible, courant plus faible, idéal pour un microcontrôleur)

La Figure 4.2 illustre le circuit L298N.

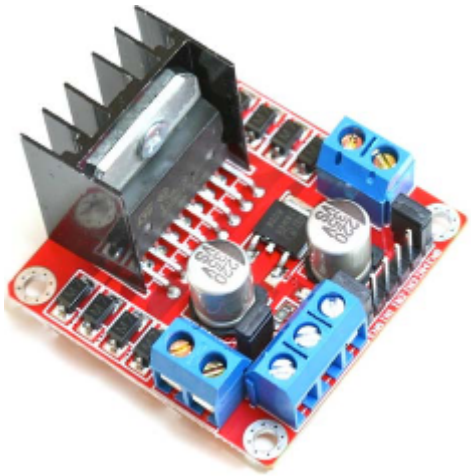


FIGURE 4.2 – Circuit L298N.

Caractéristiques Techniques :

- Tension de fonctionnement : 4V à 46V.
- Courant total (DC) : jusqu'à 4 A (2A par canal).
- Basse tension de saturation.
- Protection contre les surcharges et courts-circuits..
- Régulateur de tension intégré : Régulateur embarqué 5V à faible chute de tension, accessible par l'utilisateur.
- Diodes de protection.
- Alimentation logique : 5V.
- Puissance maximale : 25W.

4.3.2 Moteur à courant continu JGA25-370

Le moteur que nous avons utilisé est un moteur à courant continu de 12V équipé d'un encodeur intégré (Figure 4.3), ce qui permet de mesurer la position angulaire et la vitesse de rotation du moteur en temps réel. Ce type de moteur est particulièrement adapté aux applications nécessitant un contrôle précis de la vitesse et de la direction, comme les systèmes de téléopération ou les robots mobiles. Grâce à l'encodeur, il est possible de réguler finement les mouvements et de corriger les erreurs de positionnement.

FIGURE 4.3 – Moteur à courant continu avec encodeur intégré.

Caractéristiques Techniques :

- Tension de fonctionnement : entre 6V et 18V
- Tension nominale du moteur : 12V
- Vitesse à vide à 12V : 915 tr/min
- Courant à vide à 12V : 50 mA
- Courant de décrochage à 12V : 1200 mA
- Couple de décrochage à 12V : 1 kg.cm
- Rapport de réduction : 1 :9,6
- Poids : 82 g

Branchement du Moteur :

Le moteur dispose de six fils pour les connexions du moteur et de l'encodeur (Figure 4.4) :

1. **Rouge :** Moteur -
2. **Noir :** Masse du capteur à effet Hall (GND).
3. **Jaune :** Phase A de l'encodeur.
4. **Vert :** Phase B de l'encodeur.
5. **Bleu :** Vcc du capteur à effet Hall (5V).
6. **Blanc :** Moteur +

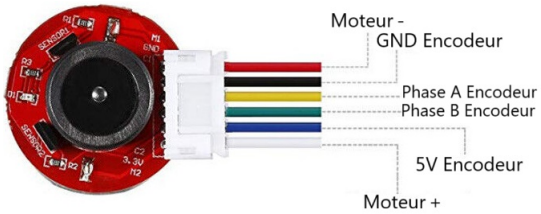


FIGURE 4.4 – Schéma de câblage du Moteur.

4.4 Résultats et discussions

Dans cette section, nous présentons et discutons les résultats obtenus à partir de deux approches : d'abord par l'implémentation sur Arduino, puis par la simulation sous Simulink. L'objectif est d'analyser le comportement du système maître-esclave en présence d'un retard de transmission, et d'évaluer l'impact du régulateur flou sur les performances du système, notamment la précision de la position et l'atténuation de l'erreur.

4.4.1 Implémentation sur Arduino

Dans le cadre de notre projet, un système maître-esclave de téléopération a été conçu et piloté par un régulateur flou implémenté sur Arduino. Ce régulateur analyse l'erreur de position ainsi que sa variation pour ajuster la commande moteur de l'esclave. Les résultats expérimentaux obtenus à partir de l'Arduino montrent les courbes suivantes :

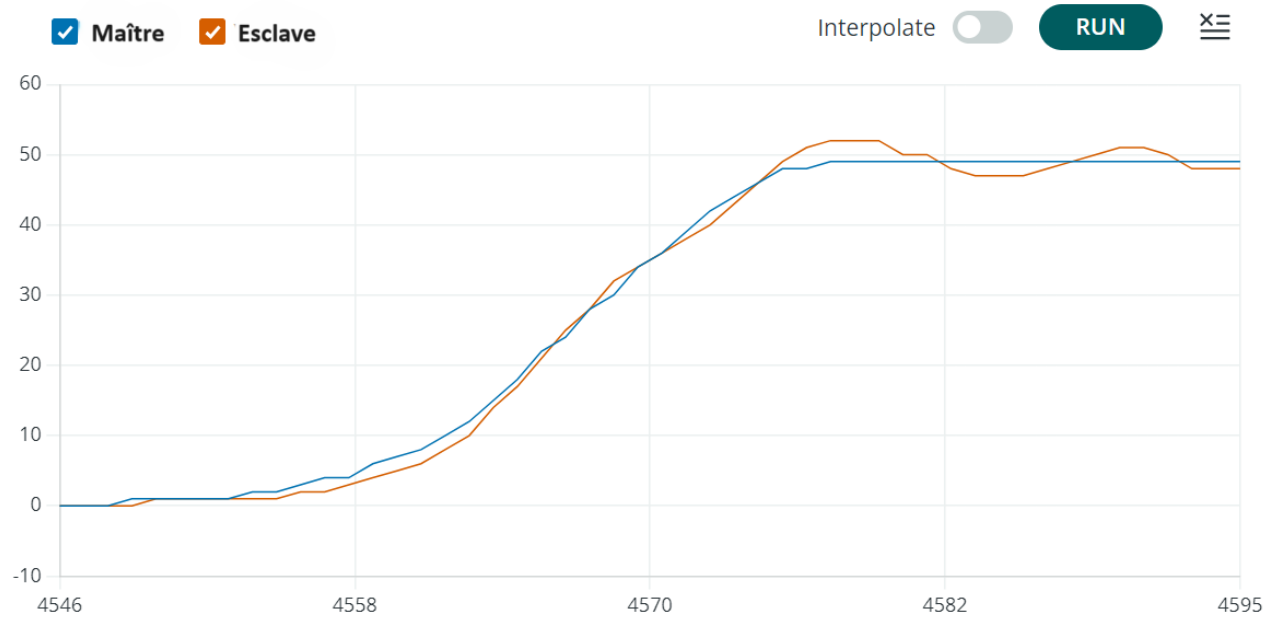


FIGURE 4.5 – Résultats de simulation des deux positions Maître/Esclave.

La figure 4.5 présente la position du maître et de l'esclave en fonction du temps. La position du maître est commandée par un opérateur, et l'esclave suit cette position. En raison du retard de transmission introduit dans le système, la position de l'esclave présente un léger décalage par rapport à celle du maître. Cependant, grâce à l'ajustement en temps réel de la commande via le régulateur flou, l'esclave parvient à suivre de près la trajectoire imposée par le maître.

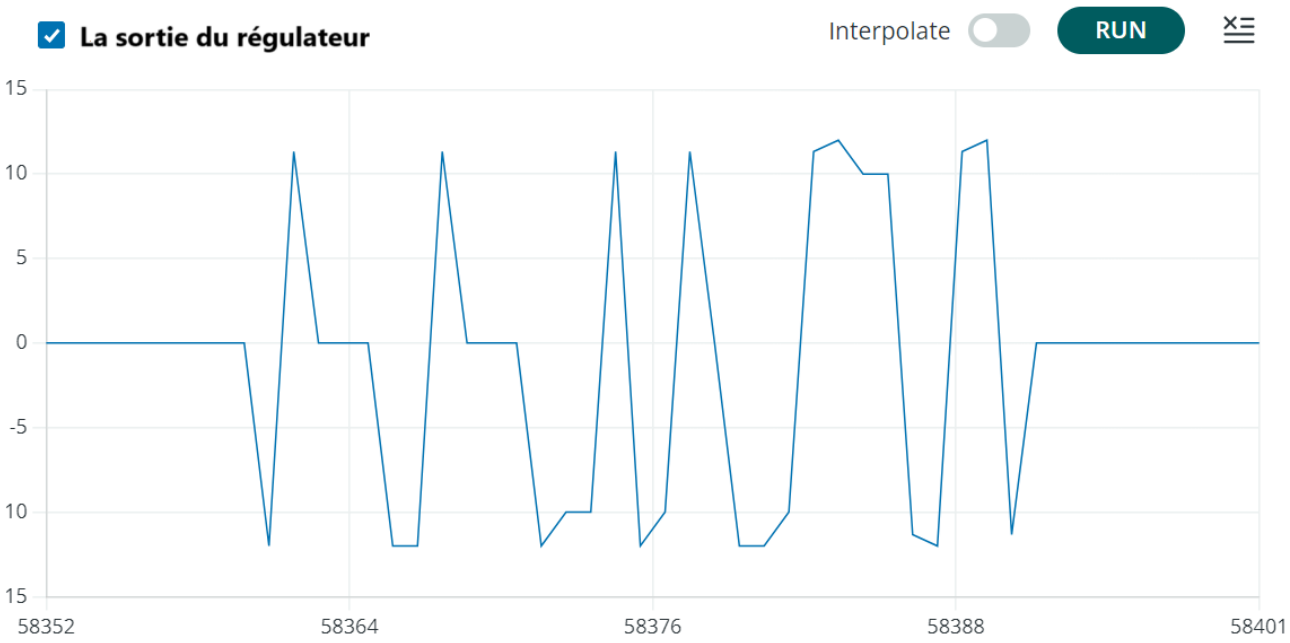


FIGURE 4.6 – Résultats de simulation de la commande.

La figure 4.6 montre l'évolution de la commande générée par le régulateur flou. Cette commande ajuste la vitesse et la position du moteur esclave pour réduire l'erreur entre le maître et l'esclave. Les variations du graphe reflètent les corrections appliquées par le régulateur pour minimiser cet écart.

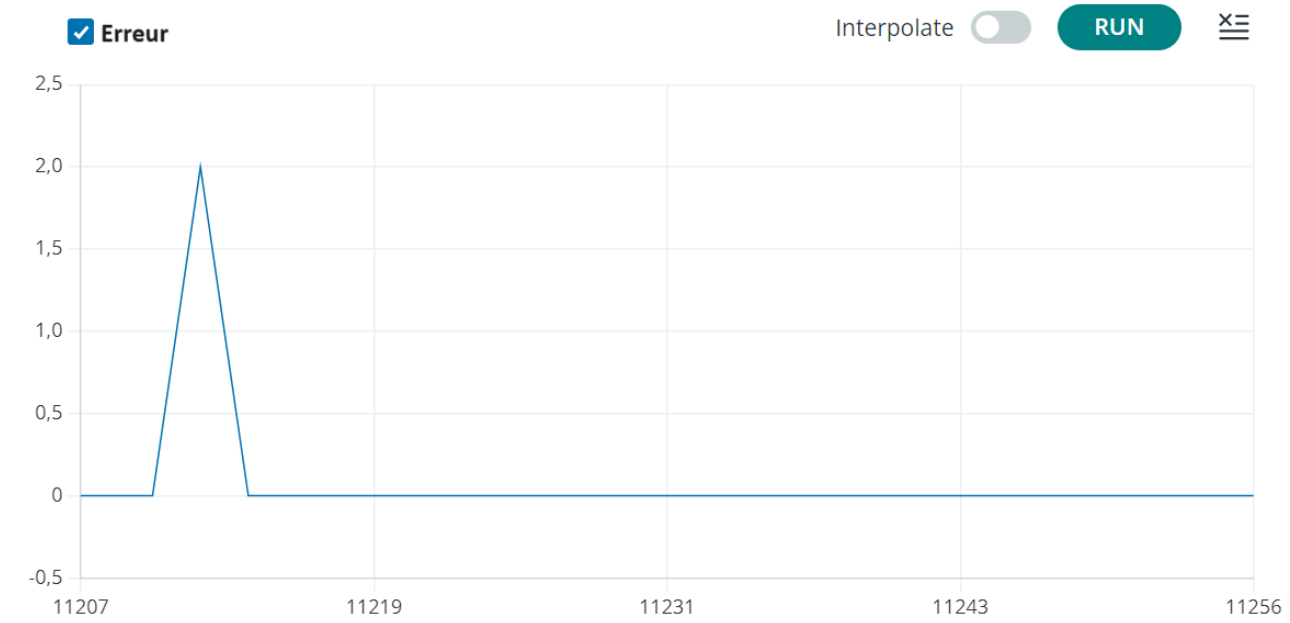


FIGURE 4.7 – Erreur de position.

La figure 4.7 met en évidence l'erreur de position entre les deux mouvements. On y observe que l'écart diminue rapidement pour se stabiliser à un niveau minimal en phase finale, ce qui illustre l'efficacité du régulateur flou dans la réduction des erreurs. Ce résultat

est particulièrement notable car il est obtenu avec un temps de réaction réduit du moteur esclave, montrant que le régulateur flou non seulement améliore la précision, mais aussi assure une réponse rapide et stable du système.

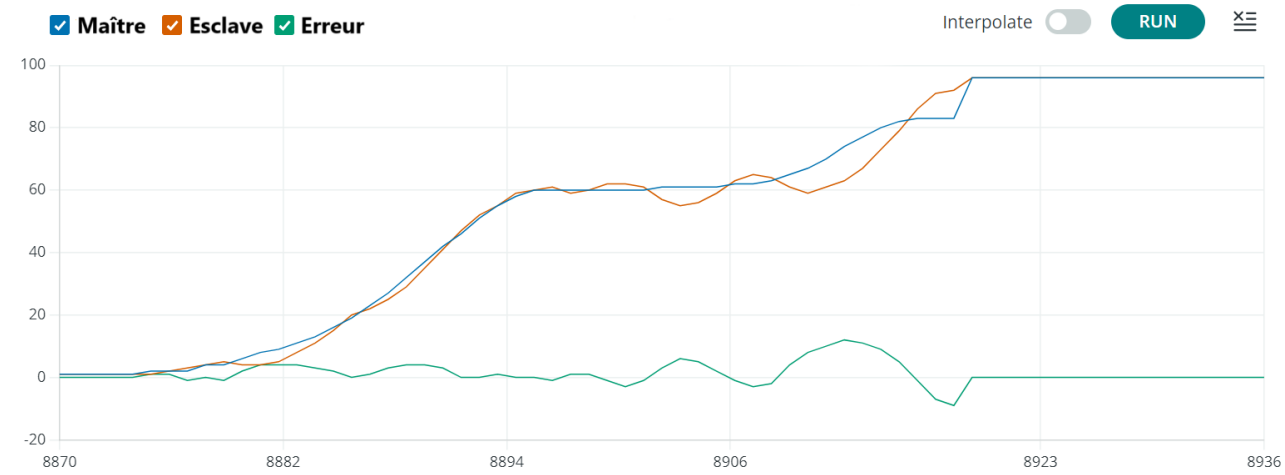


FIGURE 4.8 – Résultats de simulation des deux positions Maître/Esclave et l'erreur de position.

La figure 4.8 combine les résultats pratiques obtenus pour les positions du maître et de l'esclave, ainsi que l'erreur de position correspondante. Cette figure montre que l'esclave suit étroitement la position du maître, malgré le retard de transmission fixe. L'erreur de position, bien qu'initialement élevée, est réduite au fil du temps grâce à l'action du régulateur flou. Les résultats expérimentaux valident ainsi l'efficacité du régulateur flou pour maintenir une précision raisonnable même en présence de retard de transmission.

4.4.2 Simulation sous Simulink

Dans un système maître-esclave modélisé sous Simulink, il est possible d'afficher les résultats expérimentaux, notamment le délai de transmission, en suivant une approche structurée. Le maître envoie des commandes aux esclaves, qui exécutent ces commandes avant de renvoyer une réponse. Le schéma bloc représentant cette modélisation est illustré dans la figure 4.9 ci-dessous.

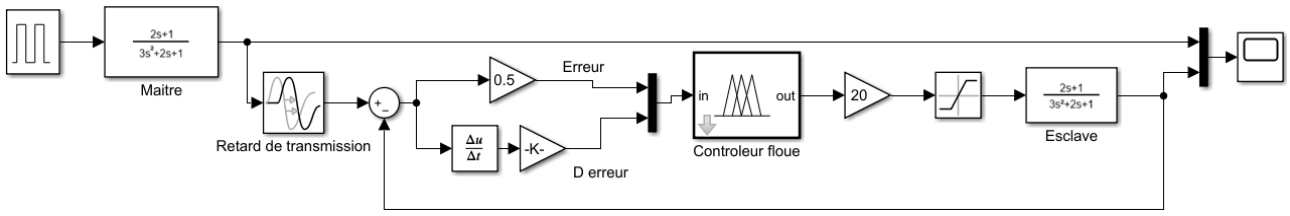


FIGURE 4.9 – Modèle de simulation du système Maître/Esclave.

Les simulations ont été réalisées dans deux scénarios :

Cas avec un retard de transmission de 0.5 s :

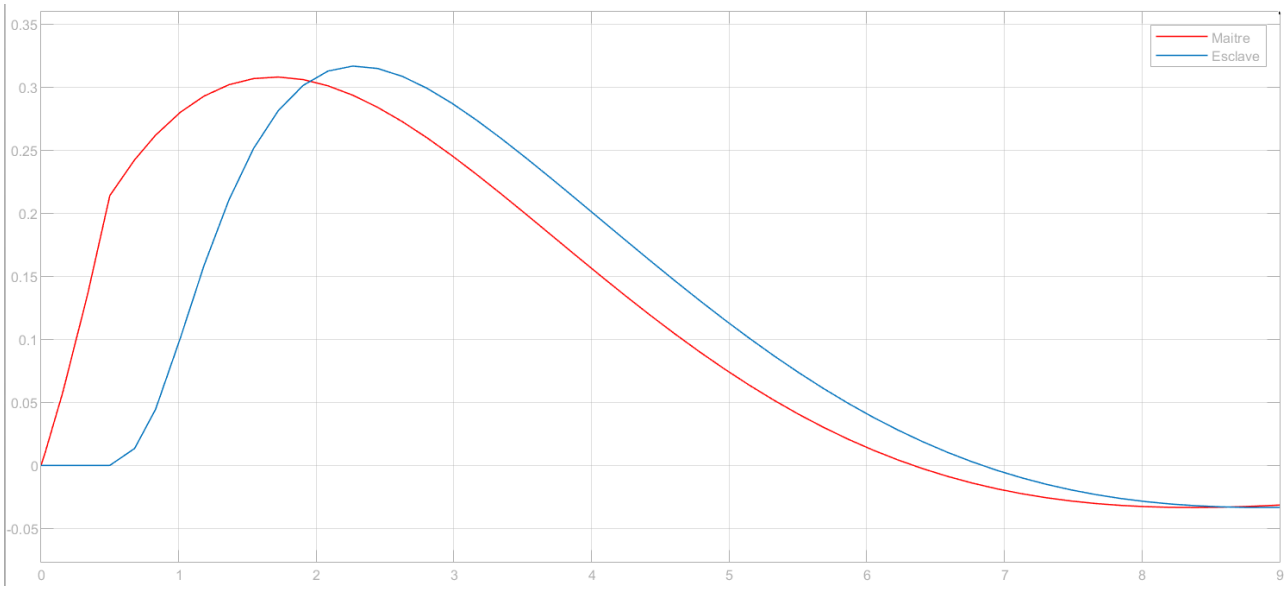


FIGURE 4.10 – Position du maître et de l'esclave avec un retard de 0.5 s.

Cette figure 4.10 montre comment la position de l'esclave suit celle du maître, avec un retard de 0,5 s. Ce délai affecte la synchronisation du système, révélant l'impact du retard sur la capacité de l'esclave à reproduire les mouvements du maître de manière fluide et précise.

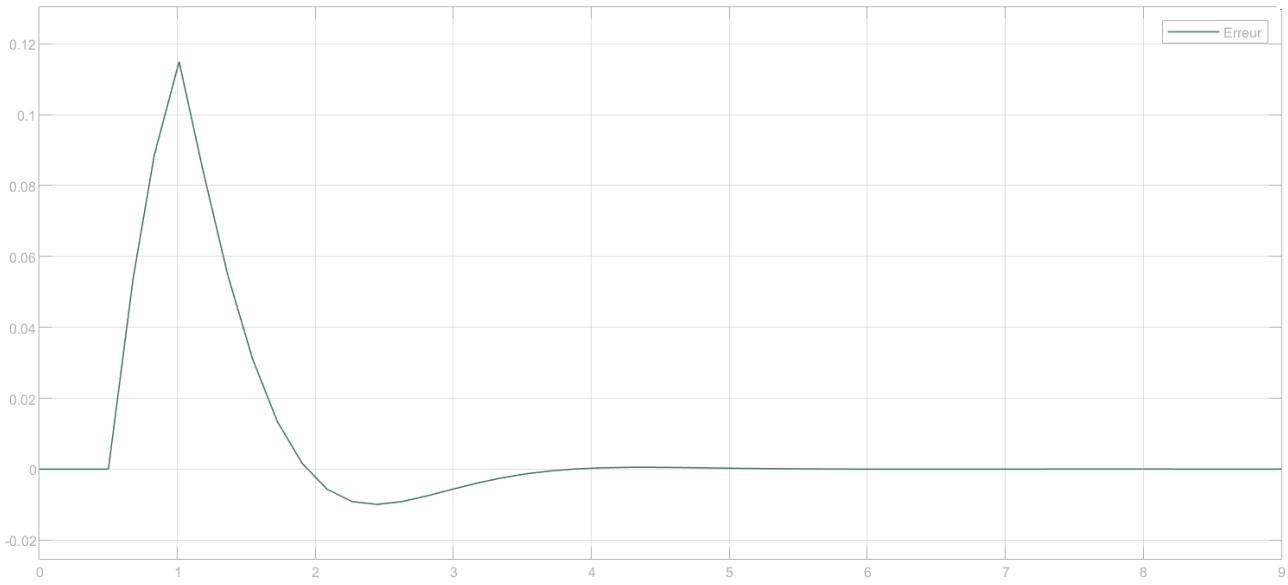


FIGURE 4.11 – Erreur de position avec un retard de 0.5 s.

Cette courbe (Figure 4.11) représente l'erreur de position entre le maître et l'esclave, en tenant compte d'un retard de transmission de 0,5 s. Avec la présence d'un délai de 4 s dans l'échange de communication entre les deux sites. L'erreur initialement importante

diminue progressivement au fil du temps, jusqu'à s'annuler complètement.

Cas sans retard de transmission : Ce cas théorique nous permet d'évaluer les performances idéales du système lorsque le retard est nul.

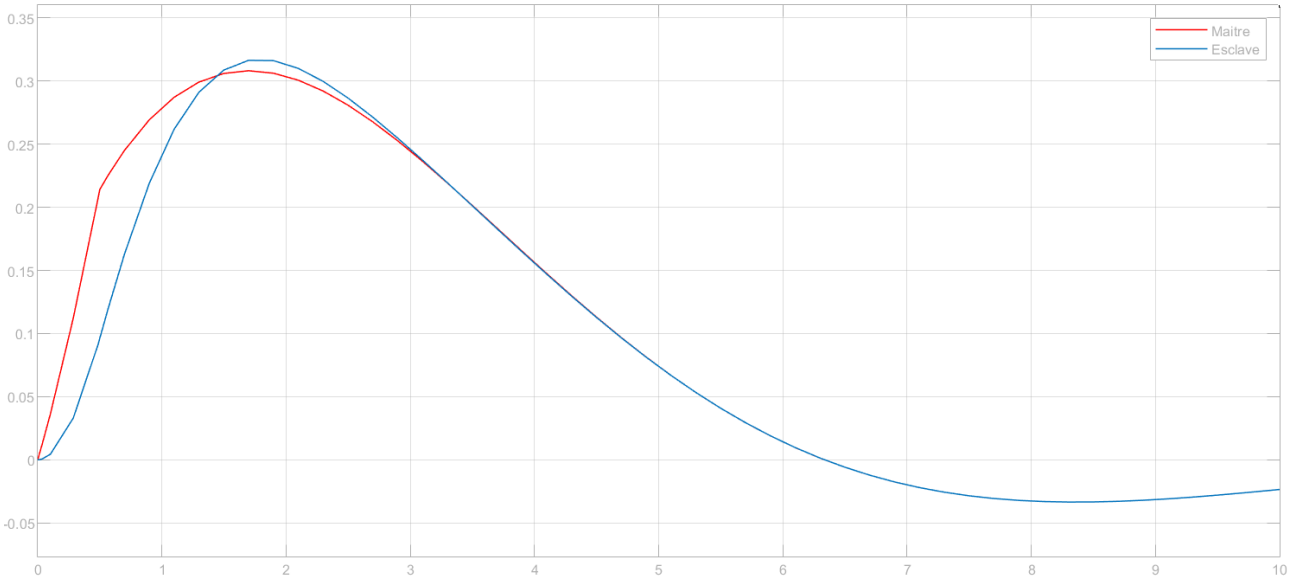


FIGURE 4.12 – Position du maître et de l'esclave sans retard.

Cette figure 4.12 montre la position du maître et celle de l'esclave. L'esclave suit de près le maître, avec un léger décalage visible. Cependant, ce décalage reste faible, jusqu'à ce que l'esclave corresponde exactement à la position du maître.

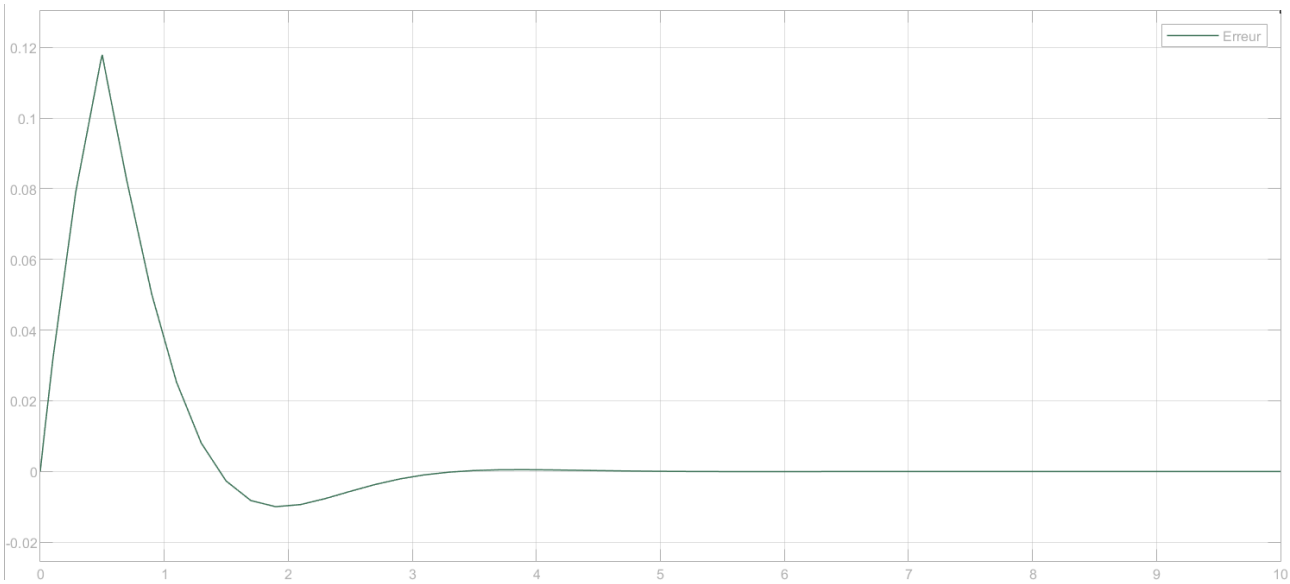


FIGURE 4.13 – Erreur de position sans retard.

Cette figure 4.13 montre l'erreur de position entre le maître et l'esclave, en tenant compte de la présence d'un délai de 3,5 s dans l'échange de communication entre les deux

sites. Au début, une erreur est visible, mais elle diminue progressivement à mesure que le système s'ajuste.

4.4.3 Discussion des résultats

Les résultats obtenus à partir des deux approches, Arduino et Simulink, montrent clairement l'impact significatif du régulateur flou sur la performance du système maître-esclave. Dans l'implémentation sur Arduino, le régulateur flou a permis une réduction rapide de l'erreur de position, même en présence de retards de transmission. Ce comportement est essentiel dans des applications où la réactivité est cruciale, comme dans les systèmes de téléopération.

La simulation sous Simulink a complété ces observations en fournissant une vue théorique des performances idéales du système. La comparaison entre les cas avec et sans retard de transmission a mis en évidence comment les retards peuvent influencer la précision et la fluidité des mouvements, soulignant l'importance d'un algorithme de contrôle performant.

Globalement, ces résultats illustrent la capacité du régulateur flou à compenser efficacement les erreurs dues aux retards de transmission, assurant ainsi une synchronisation et une performance améliorées dans les systèmes de téléopération.

4.5 La maquette réalisée

Notre maquette réalisée est donné par cette figure ci-dessous :

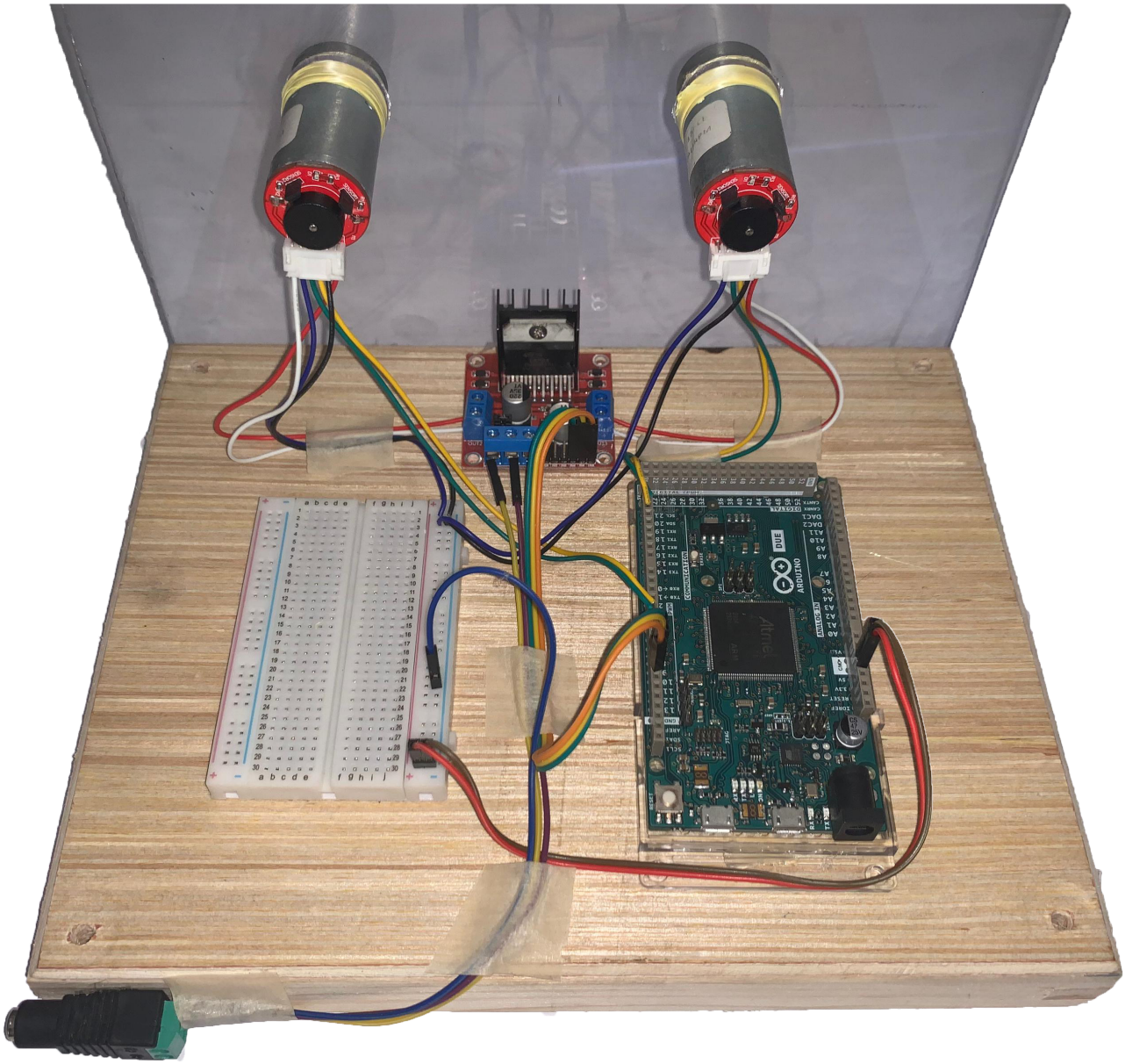


FIGURE 4.14 – La maquette réalisée.

4.6 Conclusion

Ce chapitre a illustré l'application d'une stratégie de commande floue à notre système de téléopération. Nous avons détaillé la conception du régulateur flou ainsi que les différents composants matériels utilisés. Les simulations et les résultats expérimentaux ont démontré l'efficacité de ce régulateur.

Les analyses montrent que le régulateur flou joue un rôle crucial dans l'amélioration des performances du système. Il réduit l'écart de position entre le maître et l'esclave tout en assurant une réponse rapide et stable. Ce régulateur contribue ainsi à la stabilité et à la précision du système, le rendant fiable dans diverses conditions. Ces résultats soulignent l'importance d'intégrer un régulateur flou dans les systèmes de téléopération pour optimiser leur performances.

Conclusion générale

Le travail présenté dans ce mémoire a pour objectif d'aborder les techniques de l'intelligence artificielle, plus particulièrement les techniques à base de la logique floue en vue de commander des systèmes de téléopération.

Pour exposé notre démarche, nous avons, dans la première étape étudié une architecture de contrôle des systèmes de téléopération à deux canaux de communication qui utilise les signaux de position, relevés à travers un encodeur mais sans recourir à l'utilisation d'un capteur de force. Cette dernière est très souhaitable afin de permettre une commande intuitive du robot télé opéré tout en reproduisant le plus fidèlement possible les efforts. Par ailleurs, en vue d'améliorer les performances de la commande à distance, des solutions ont été envisagées dans le but de la stabilisation du système, à savoir les conditions sur la passivité et l'élimination du retard de transmission.

L'implémentation du contrôleur flou sur un système de téléopération à un degré de liberté, à travers en utilisant la carte Arduino Due, a permis de réduire le temps de calcul des algorithmes et à minimiser les retards de communication entre le maître et l'esclave. Ceci se traduit par la transmission des informations de position de manière précise et rapide.

L'analyse des résultats obtenus à partir des simulations, ainsi que des expériences pratiques effectuées via la carte Arduino, a permis d'évaluer l'efficacité du contrôleur flou dans un système de téléopération en présence du retard de transmission. Les simulations détaillées réalisées sur Simulink ont été comparées aux résultats obtenus lors des essais pratiques sur la carte Arduino Due. Il en ressort que les performances de poursuite obtenus dans les deux contextes sont remarquablement similaires, ce qui confirme ainsi la robustesse du contrôleur flou.

En effet, les simulations ont permis de mieux comprendre l'impact du délai de transmission sur la précision des mouvements entre le maître et l'esclave, tandis que l'implémentation pratique a démontré que le contrôleur flou peut être exécuté en temps réel, tout en minimisant l'impact les délais de communication de telle manière à garantir les performances de poursuite avec une bonne précision entre des signaux de position. Les résultats obtenus, tant dans les simulations que dans les expériences pratiques avec Arduino, montrent que l'intégration du contrôleur flou améliore la stabilité du système tout en maintenant une bonne précision, même en présence de retards de transmission. Cela

confirme que ce type de contrôle est particulièrement adapté aux systèmes de téléopération où les délais peuvent affecter les performances de poursuite.

En conclusion, nous avons voulu, au terme de ce mémoire, tenter de dégager que le contrôleur flou est essentiel pour compenser l'effet des erreurs de position dues aux variations du délai, en ajustant dynamiquement la commande en fonction de l'erreur de position et de sa dérivée.

Bibliographie

[1] **Claudio Pacchierotti**, Cutaneous Haptic Feedback in Robotic Teleoperation, Springer Series on Touch and Haptic Systems, 2015.

[2] **Pierre LETIER** Bras Exosquelette Haptique : Conception et Contrôle (Laboratoire des Structure Actives Service des Constructions Mécaniques et Robotique) juillet 2010.

[3] **OTMANE**, **Samir**. Téléopération, télérobotique et Internet : Techniques & applications. Université d'Evry Val d'Essonne CNRS-FR, 2010, vol. 2873, p. 4-56.

[4] **KHATI**, **Hocine**. Commande d'une architecture de téléopération par la carte FPGA. 2020. Thèse de doctorat. UNIVERSITE MOULOUD MAMMERI TIZI-OUZOU.

[5] **Zarrad**, **Walid**. Télé-opération avec retour d'effort pour la chirugie mini-invasive. Diss. Université Montpellier II-Sciences et Techniques du Languedoc, 2007.

[6] **Bolopion, Aude**. Couplages haptiques pour la téléopération à l'échelle nanoscopique. Diss. Université Pierre et Marie Curie-Paris VI, 2010.

[7] **Launay, F.** (n.d.). Régulation et Commande Automatique - Cours Master. LIAS.

[8] **Marczyk, J.** "L'incertitude en Conception : Formalisation, Estimation."

[9] **Courreges, Fabien.** "Détermination d'une Architecture de Téléopération Bilatérale en Présence de Retards." Laboratoire Vision et Robotique (2000).

[10] **Barbé, Laurent.** Téléopération avec retour d'efforts pour les interventions percutanées. Diss. Université Louis Pasteur-Strasbourg I, 2007.

[11] **Guenounou, Ouabib.** Méthodologie de conception de contrôleurs intelligents par l'approche génétique. Application à un bioprocédé. Diss. Université Paul Sabatier-Toulouse III, 2009.

[12] **Huguenin, G**. (n.d.). Introduction à la logique floue.

[13] **Siddique, Nazmul**. Intelligent control : a hybrid approach based on fuzzy logic, neural networks and genetic algorithms. Vol. 517. Springer, 2013.

[14] **Chevrie, François**, and **François Guély**. "La logique floue." Cahier technique 191 (1998) : 1-28.

[15] **Baghli, Lotfi.** Contribution à la commande de la machine asynchrone, utilisation de la logique floue, des réseaux de neurones et des algorithmes génétiques. Diss. Université Henri Poincaré-Nancy I, 1999.

[16] **Mamdani, E. H., & Assilian, S.** (1975). An experiment in linguistic synthesis with a fuzzy logic controller. International journal of man-machine studies, 7(1), 1-13.

[17] **Grisolia, J.** (n.d.). Microcontrôleurs & open source hardware. Département de Génie Physique – INSA Toulouse.

[18] **Mukherjee, Arunabho**, et al. "Function generator using Arduino DUE." (2016).

[19] **Gray, Edwyn** (2004). Nineteenth-Century Torpedoes and Their Inventors. Naval Institute Press. ISBN 1-59114-341-1.

[20] **Goertz**, R. C. (1952). Mechanical master-slave manipulator. U.S. Patent 2,632,574.

Annexes

L298N :

La Figure 4.15 présente le module L298N

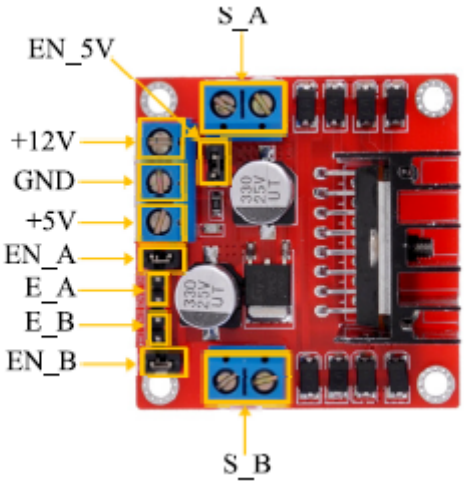


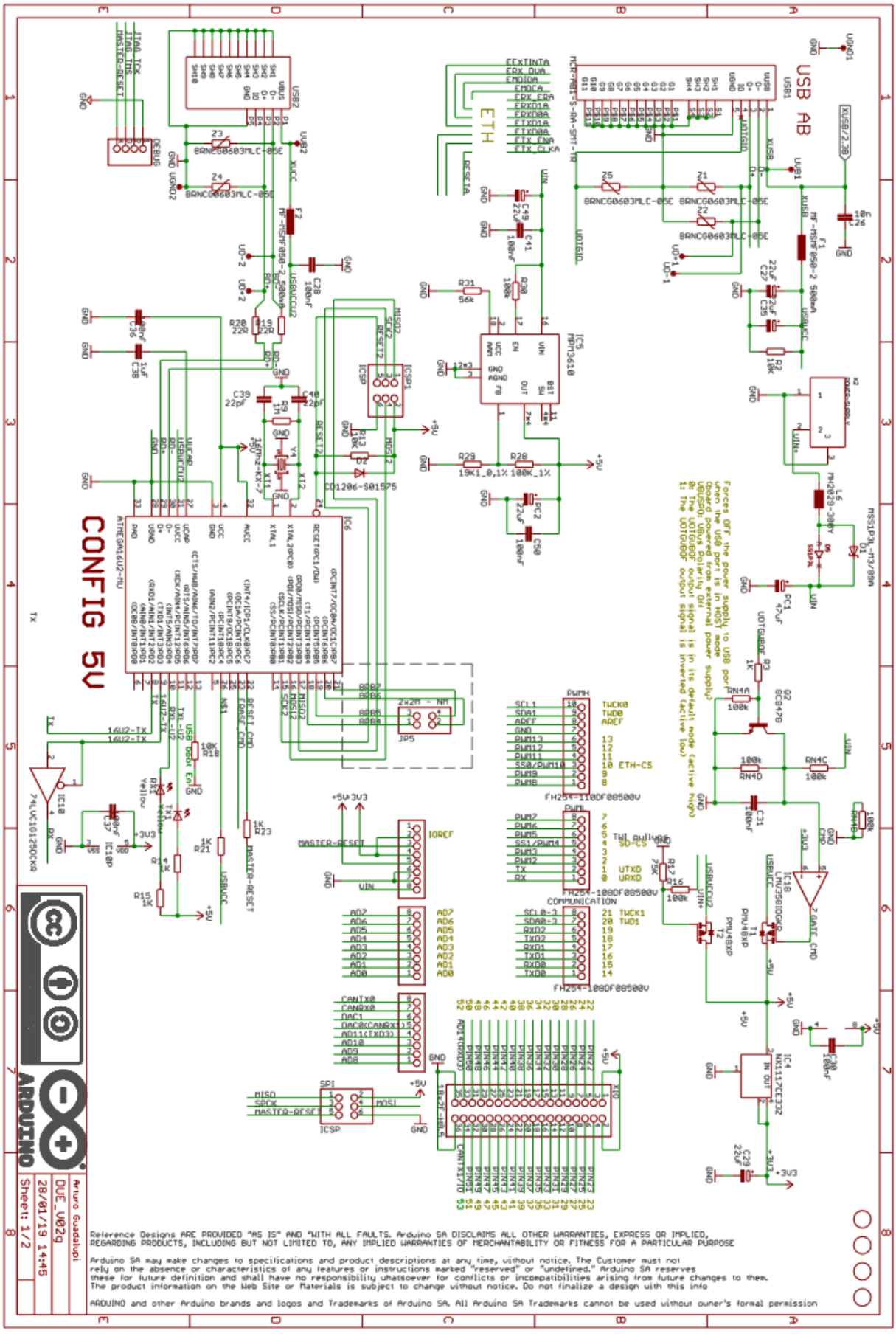
FIGURE 4.15 – Module L298N.

Le Tableau4.2 présente les fonctions des pins du circuit L298N.

Nom de la pin	Description
S_A	Moteur A
S_B	Moteur B
$+12V$	Entrée d'alimentation $12V$
GND	Terre
$+5V$	Sortie $5V$
E_A	2 Entrées logiques pour le moteur A
A_B	2 Entrées logiques pour le moteur B
EN_A	Contrôle du moteur A
EN_B	Contrôle du moteur B
EN_$5V$	Activation $5V$

TABLE 4.2 – Fonctions des pins du L298N.

Arduino DUE :


CONFIG 5V
USB AB
ETH
ARDUINO
DUE_V02g
28/01/13 14:45
Sheet: 1/2
Arturo Guadalupi
Reference Designs ARE PROVIDED "AS IS" AND "WITH ALL FAULTS. Arduino SA DISCLAIMS ALL OTHER WARRANTIES, EXPRESS OR IMPLIED, REGARDING PRODUCTS, INCLUDING BUT NOT LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE
Arduino SA may make changes to specifications and product descriptions at any time, without notice. The Customer must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Arduino SA reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The product information on the Web Site or Materials is subject to change without notice. Do not finalize a design with this info
ARDUINO and other Arduino brands and logos are Trademarks of Arduino SA. All Arduino SA Trademarks cannot be used without owner's formal permission

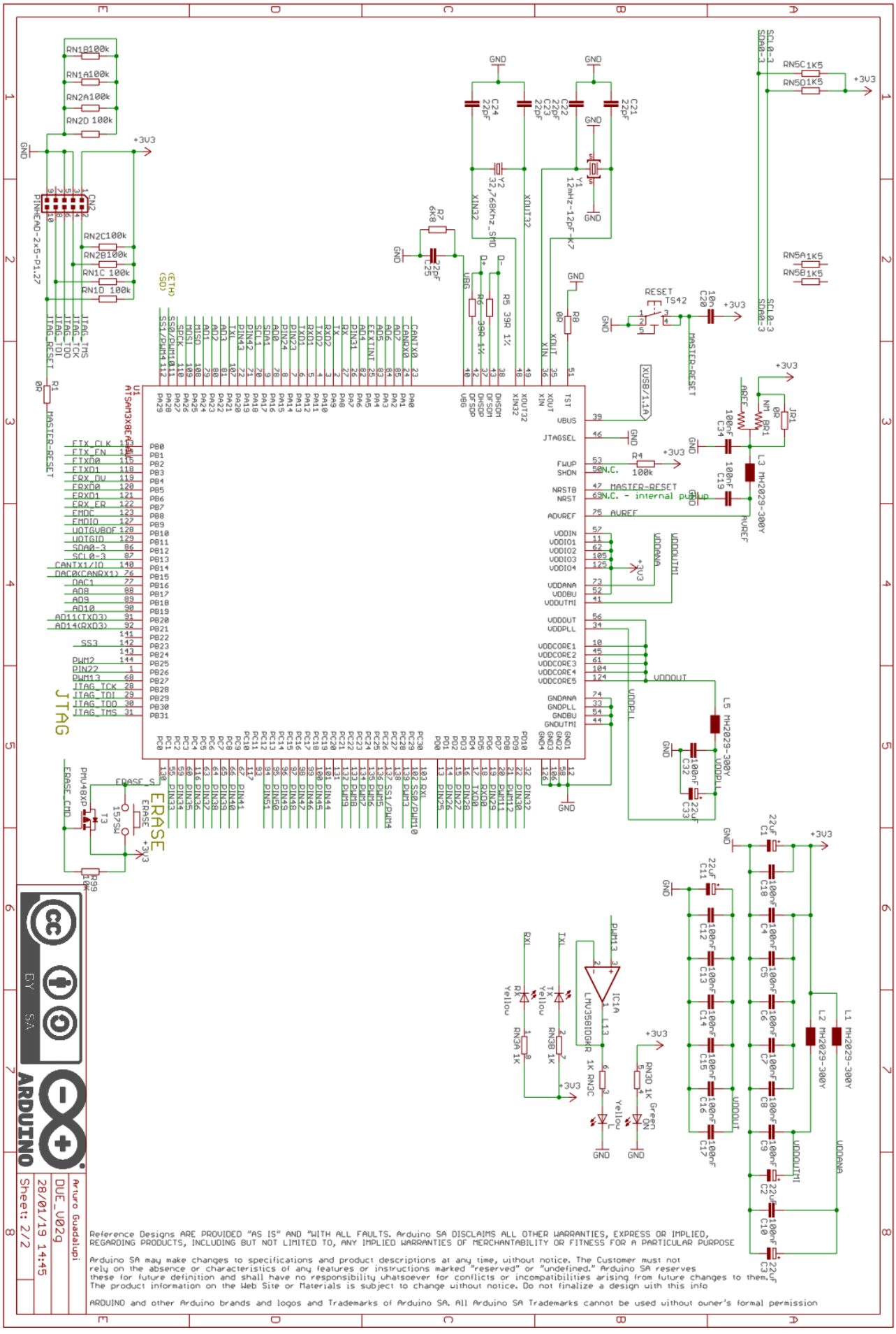


FIGURE 4.16 – Schémas éléctrique de la carte Arduino DUE.

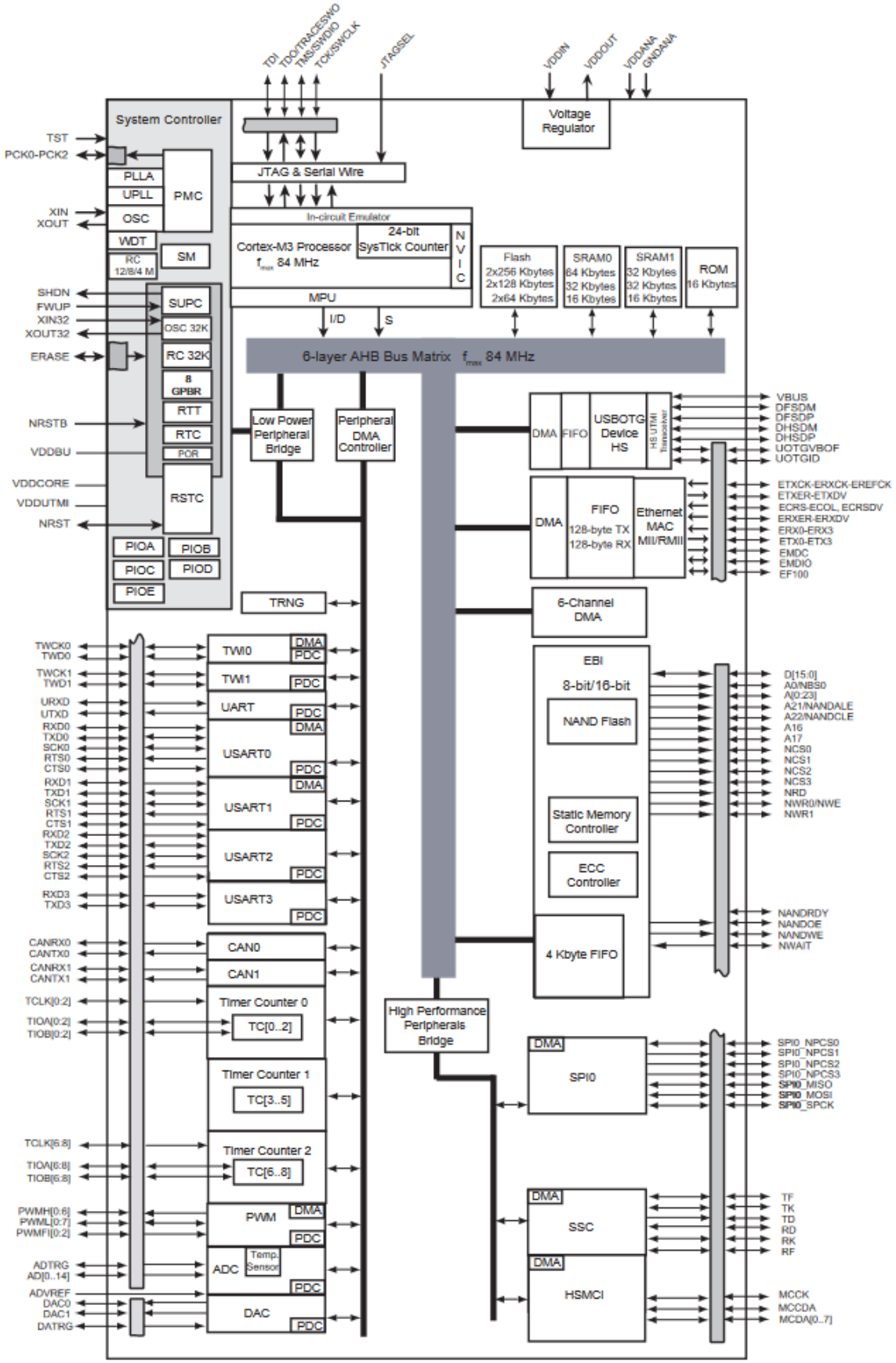


FIGURE 4.17 – Architecture du microcontrôleur.