الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي

UNIVERSITÉ BADJI MOKHTAR - ANNABA
BADJI MOKHTAR – ANNABA UNIVERSITY

جامعة باجي مختار
عنابـــــــــة

Faculté : TECHNOLOGIE
Département : ELECTRONIQUE
Domaine : SCIENCES ET TECHNOLOGIES
Filière : ELECTRONIQUE
Spécialité : INSTRUMENTATION

Mémoire

Présenté en vue de l'obtention du Diplôme de Master
Thème :

Evitement des obstacles pour la navigation d'un handicapé visuel

Présenté par : *DJEDDI Ikram*
Encadrant : *BOUSBIA SALAH Mounir* Pr *Université Badji Mokhtar-Annaba*

Jury de Soutenance :

BENSAOULA Salah	MCA	Université Badji Mokhtar-Annaba	Président
BOUSBIA SALAH Mounir	Pr	Université Badji Mokhtar-Annaba	Encadrant
HAMDI Rachid	Pr	Université Badji Mokhtar-Annaba	Examinateur

Année Universitaire : 2022/2023

Remerciement

Je remercie Dieu qui m'a donné la volonté et le courage pour la réalisation de ce travail.

De m'avoir guidé dans le chemin de la science.

J'exprime ma sincère gratitude envers Monsieur « **Bousbia Salah Mounir** » pour son encadrement de qualité exceptionnelle. Sa rigueur et son esprit critique ont constamment guidé mes pas dans la bonne direction. Son attitude exemplaire tout au long de ce travail m'a inspiré une véritable passion pour ce domaine. Je le remercie chaleureusement d'avoir éveillé en moi cette passion et d'avoir contribué à la réalisation de ce travail.

Je tiens à remercier chaleureusement « **Loubna Bougheloum** », la doctorante exceptionnelle qui m'a apportée une aide inestimable. Ses conseils éclairés et son soutien indéfectible ont été essentiels pour surmonter les défis auxquels j'ai été confronté. Je lui suis infiniment reconnaissante pour sa patience et son engagement envers mon succès.

Je remercie les membres du jury qui ont fait l'honneur d'évaluer ce travail

Je voudrais également exprimer ma profonde gratitude envers mes parents, mes piliers et mes guides dans la vie. Leur amour inconditionnel, leur soutien sans faille et leurs sacrifices ont été les fondements de ma réussite. Je suis comblée par leur présence réconfortante et leur encouragement constant.

Je n'oublie pas mon cher frère « **Billel** », qui a toujours été là pour moi. Son soutien indéfectible et ses conseils avisés ont été d'une importance capitale dans mon parcours. Je lui suis infiniment reconnaissant(e) pour sa présence précieuse dans ma vie.

Je remercie aussi ma belle-sœur **Chaima**.

Je souhaite également exprimer ma reconnaissance envers ma famille élargie, qui m'a soutenu et encouragé tout au long de mon chemin. Leur amour et leur soutien inconditionnels ont été une source de réconfort et de motivation.

Enfin, mes amis fidèles **Abd El Barie, Rami, Ismail, Yahia, Zoubir, Dorsaf, Chahinez, Wejdan.**. Leur amitié précieuse, je tiens à vous remercier du fond du cœur pour votre amitié indéfectible. Votre présence joyeuse, votre soutien inconditionnel et votre confiance en moi m'ont permis de grandir et de surmonter les obstacles. Je suis honorée de vous avoir dans ma vie.

Que ces mots sincères témoignent de ma gratitude infinie envers toutes les personnes qui ont contribué à ma réussite. Leur impact restera à jamais gravé dans ma mémoire, et je suis béni(e) de les avoir à mes côtés.

Avec une gratitude éternelle.

Djeddi Ikram

Dédicace :

Je dédie ce travail a :

À mes chers parents,

Vous êtes les piliers de ma vie, mes guides et mes plus grands supporters. Votre amour, votre soutien indéfectible et vos sacrifices ont été essentiels à ma réussite. Ce mémoire est dédié à vous, pour avoir toujours cru en moi et pour m'avoir encouragé à poursuivre mes rêves. Je vous suis éternellement reconnaissant.

À mon cher grand-père,

Vous avez été une source d'inspiration constante pour moi. Vos récits, ta sagesse et votre amour inconditionnel m'ont motivé à me surpasser. Ce mémoire est dédié à vous, en reconnaissance de tous les précieux enseignements que vous m'avez transmis. Vous resterez à jamais dans mon cœur.

À mon frère bien-aimé,

Vous avez été mon compagnon de route, mon allié et mon meilleur ami. Votre présence encourageante et votre soutien infaillible ont été une force motrice tout au long de ce parcours. Ce mémoire est dédié à vous, en témoignage de notre lien indéfectible et de notre complicité inébranlable.

À la chère doctorante Loubna qui m'a réellement aidée,

Vous êtes véritablement un trésor précieux dans mon parcours. Vous avez apporté une aide réelle et un soutien inestimable. Vous avez été un pilier solide, une source d'inspiration et de guide. Ce mémoire est dédié à vous, en reconnaissance de vos précieux conseils et de votre contribution majeure à la réussite de cette recherche. Je vous serai éternellement reconnaissante.

À tous les enseignants du département d'électronique,

Vous avez partagé avec passion vos connaissances et votre expertise. Vos enseignements ont été une source d'inspiration et ont façonné ma compréhension du domaine. Ce mémoire est dédié à vous, en reconnaissance de votre dévouement à former la prochaine génération d'ingénieurs. Vos efforts ont fait une différence significative dans ma vie.

Djeddi Ikram

الملخص:

لتسهيل التنقل وضمان سلامة الأشخاص المعاقين بصريًا، تم تطوير العديد من الحلول على مر السنين. فعلى سبيل المثال، تُستخدم العصي البيضاء التقليدية كأداة مساعدة لاكتشاف العوائق القريبة من الأرض. ومع ذلك، فقد أتاحت التطورات التكنولوجية الحديثة فرصًا لظهور حلول أكثر تطورًا. يركز هذا العمل على تطوير نظام لتجنب العوائق للأشخاص المعاقين بصريًا بناءً على المنطق الغامض، باستخدام منصة تطوير غوغل كولاب. يهدف هذا النهج التكنولوجي إلى تعزيز قدرة الأشخاص المعاقين بصريًا على التنقل والاستقلالية والسلامة في رحلاتهم اليومية. يستخدم النظام مستشعرات لاكتشاف العوائق المحيطة وجمع البيانات مثل المسافة والزاوية المتعلقة بهذه العوائق. من خلال استخدام المنطق الغامض، ستتم معالجة المعلومات الحسية بطريقة مرنة ومتكيفة، مما يولد قرارات مناسبة لتجنب العوائق. ستوفر هذه المساعدة إمكانيات جديدة لتحسين جودة حياة الأشخاص المعاقين بصريًا من خلال تعزيز قدرتهم على التنقل والسلامة في بيئة معقدة ومتغيرة باستمرار.

الكلمات الرئيسية: المعاقين بصريًا، العوائق، المنطق الغامض، التجنب، غوغل كولاب.

Résumé :

Pour faciliter la mobilité et assurer la sécurité des non-voyants, de nombreuses solutions ont été développées au fil des années. Les cannes blanches traditionnelles, par exemple, sont largement utilisées comme outil d'assistance pour détecter les obstacles physiques proches du sol. Cependant, les avancées technologiques récentes ont ouvert la voie à de nouvelles solutions plus avancées. Ce travail se concentre sur le développement d'un système d'évitement d'obstacles pour les non-voyants basé sur la logique floue en utilisant Google Colab comme plateforme de développement. Cette approche technologique permettra d'améliorer la mobilité, l'autonomie et la sécurité des non-voyants dans leurs déplacements quotidiens. Ce système utilisera des capteurs pour détecter les obstacles environnants et collecter des données telles que la distance et l'angle par rapport à ces obstacles. En utilisant la logique floue, les informations sensorielles seront traitées de manière flexible et adaptative, permettant ainsi de générer des décisions appropriées pour éviter les obstacles. Cet aide offrira de nouvelles possibilités pour améliorer la qualité de vie des non-voyants en favorisant leur mobilité et leur sécurité dans un environnement complexe et changeant.

Mots clés : Non-voyants, obstacles, logique floue, Google Colab, évitement.

Abstract:

To facilitate mobility and ensure the safety of visually impaired individuals, numerous solutions have been developed over the years. Traditional white canes, for example, are widely used as an assistive tool to detect physical obstacles close to the ground. However, recent advancements in technology have created opportunities for more sophisticated solutions to emerge. This work focuses on the development of an obstacle avoidance system for visually impaired individuals based on fuzzy logic, using Google Colab as the development platform. This technological approach aims to enhance the mobility, autonomy, and safety of visually impaired individuals in their daily journey. The system will utilize sensors to detect surrounding obstacles and collect data such as distance and angle in relation to these obstacles. By employing fuzzy logic, the sensory information will be processed in a flexible and adaptive manner, thereby generating appropriate decisions to avoid obstacles. This aid will provide new possibilities for improving the quality of life of visually impaired individuals by promoting their mobility and safety in a complex and ever-changing environment.

Keywords: Visually impaired, obstacles, fuzzy logic, Google Colab, avoidance.

Table des matières

Liste des figures

Liste des tableaux

Introduction générale :

Introduction :

Actuellement les avancées technologiques ont considérablement amélioré la qualité de vie des personnes en situation d'handicap. Parmi ces personnes, les non-voyants sont confrontés à des défis spécifiques lorsqu'ils se déplacent au quotidien, en raison de l'absence de perception visuelle de leur environnement. Cette limitation sensorielle rend leurs déplacements plus complexes et peut entraîner des situations dangereuses. Voici quelques-unes des difficultés les plus courantes auxquelles les non-voyants sont confrontés :

- **Détection des obstacles physiques :** Sans la capacité de voir, les non-voyants doivent compter sur d'autres sens tels que le toucher et l'ouïe pour détecter les obstacles physiques tels que les meubles, les murs, les marches d'escalier, les poteaux, etc. Cela nécessite une grande concentration et une perception sensorielle développée.

- **Navigation dans des espaces publics :** Les non-voyants peuvent éprouver des difficultés à se déplacer dans des espaces publics tels que les rues animées, les centres commerciaux ou les gares, où il peut y avoir une densité élevée de personnes et d'obstacles imprévus. La gestion de l'orientation et de la navigation devient un défi majeur.

- **Identification des indications et des panneaux :** Les non-voyants peuvent avoir des difficultés à lire et à interpréter les indications écrites ou les panneaux de signalisation, ce qui peut entraîner une confusion quant à la direction à suivre ou aux obstacles à éviter.

- **Identification des passages piétons et des intersections :** Les passages piétons et les intersections peuvent être des zones critiques pour les non-voyants, car ils doivent être en mesure de détecter les feux de signalisation, les passages pour piétons et les flux de trafic. L'absence de ces repères visuels peut rendre ces situations particulièrement dangereuses.

- **Planification des itinéraires :** La planification d'itinéraires efficaces et sécurisés peut être complexe pour les non-voyants, car ils doivent prendre en compte les informations sur les transports en commun, les aménagements urbains accessibles et les conditions de circulation pour éviter les obstacles et arriver à destination de manière sûre et efficace.

Ces difficultés soulignent l'importance de développer des solutions technologiques innovantes pour aider les non-voyants dans leurs déplacements. Les avancées dans les domaines de la robotique, de l'intelligence artificielle et de la logique floue offrent des opportunités pour créer des systèmes d'assistance et des dispositifs adaptés qui peuvent améliorer la mobilité et l'autonomie des non-voyants, en leur permettant de naviguer de manière plus sûre et confiante dans leur environnement.

Ce mémoire s'inscrit dans ce cadre et y contribue au niveau de l'amélioration de la mobilité et de la sécurité des non-voyants en proposant un système d'évitement d'obstacles innovant basé sur la logique floue comme méthode de prise de décision. La recherche bibliographique montre que les solutions qui existent pour aider les non-voyants à éviter les obstacles, sont basés sur l'utilisation de cannes blanches, l'entraînement des chiens guides et les systèmes d'assistance à la mobilité basés sur des capteurs et des signaux sonores. Cependant, ces solutions présentent certaines limitations, notamment en termes de précision, de réactivité et de capacité à traiter des situations complexes.

La logique floue offre une approche prometteuse pour aborder ces limitations et développer des systèmes d'évitement d'obstacles plus efficaces pour les non-voyants. En permettant la modélisation de la perception incertaine et des décisions adaptatives, la logique floue peut aider à traiter les situations complexes auxquelles les non-voyants sont confrontés dans leur environnement.

Ainsi, l'objectif principal de ce mémoire est de concevoir et de simuler un système d'évitement d'obstacles basé sur la logique floue, en utilisant la plateforme Google Colab comme environnement de développement. Google Colab offre une interface conviviale et des ressources de calcul puissantes, ce qui en fait un outil idéal pour développer et tester des modèles de logique floue.

Dans ce mémoire, on va explorer en détail les différentes étapes nécessaires à la création de ce système d'évitement d'obstacles. L'organisation du mémoire a été organisé comme suit :

Chapitre 1 : traite les revues de la littérature pour examiner les travaux de recherche existants dans le domaine de la mobilité des non-voyants et de l'utilisation de la logique floue.

Chapitre 2 : présente le cadre théorique de la logique floue et explique comment elle peut être appliquée à la prise de décision.

Chapitre 3 : aborde une introduction complète à Python, fournissant les connaissances de base nécessaires pour comprendre et écrire du code Python. Cette compréhension de Python sera essentielle pour la mise en œuvre du système d'évitement d'obstacles basé sur la logique floue dans le chapitre suivant du mémoire.

Chapitre 4 : porte sur la modélisation du système d'évitement d'obstacles, en détaillant les variables floues, les ensembles flous et les règles d'interférence. Ainsi qu'une définition des capteurs et des paramètres d'entrée, tels que la distance et l'angle, qui seront utilisés pour détecter et évaluer les obstacles environnants.

Une fois le modèle de logique floue développé, on procédera à sa simulation sur la plateforme Google Colab. On effectuera des tests en variant les paramètres d'entrée en analysant les résultats obtenus.

Enfin, on conclut ce mémoire en discutant les résultats obtenus et des perspectives d'amélioration futures. On examinera les avantages et les limites du système d'évitement d'obstacles basé sur la logique floue et on proposera des recommandations pour son déploiement et son utilisation pratique.

Chapitre 1

Etat de l'art sur l'évitement des obstacles pour les aveugles

Introduction :

La navigation autonome est un défi quotidien pour les personnes atteintes de déficience visuelle. La simple tâche de se déplacer d'un endroit à un autre peut rapidement devenir complexe et potentiellement dangereuse lorsqu'il s'agit d'éviter les obstacles qui se dressent sur leur chemin. Cependant, grâce aux avancées technologiques et à l'émergence de nouvelles solutions, il est désormais possible d'améliorer considérablement l'expérience de navigation des personnes handicapées visuelles. Dans ce chapitre, on explique les différentes technologies innovantes et les approches pratiques qui permettent de surmonter cet obstacle majeur et de faciliter la navigation indépendante des personnes atteintes de déficience visuelle.

1.1. Les différentes technologies d'aide à la navigation pour les personnes handicapées visuelles :

Une aide à la navigation pour les malvoyants est un dispositif technologique qui permet à une personne malvoyante de se déplacer de façon autonome et en toute sécurité et de s'orienter dans son environnement. Ces systèmes utilisent des technologies telles que la géolocalisation, la reconnaissance vocale, la synthèse vocale et des capteurs pour aider les personnes aveugles ou malvoyantes à comprendre leur environnement, à se déplacer et à interagir avec leur entourage.

La géolocalisation : C'est une technologie qui permet aux personnes malvoyantes de se repérer et de trouver leur chemin sans se perdre ou rencontrer des obstacles sur leur chemin. Grâce à un système de navigation, la géolocalisation fournit des instructions précises et des itinéraires adaptés aux besoins de l'utilisateur, en prenant en compte les obstacles qu'il peut rencontrer.

La synthèse vocale : C'est une technologie qui permet au système de navigation de fournir des instructions vocales à l'utilisateur sur les obstacles à éviter. Par exemple, le système peut donner des avertissements sonores tels que "Attention, trottoir à venir" ou "Attention, obstacles sur votre chemin".

Les capteurs : Ils sont également une technologie qui peut aider les personnes malvoyantes à éviter les obstacles en les détectant sur leur chemin et en fournissant des avertissements sonores ou tactiles. Par exemple, un système de navigation équipé de capteurs peut vibrer ou émettre un son pour avertir l'utilisateur de la présence d'un obstacle à proximité, comme un poteau ou une voiture garée.

La navigation assistée par GPS : Les systèmes de navigation par GPS peuvent être utilisés par les personnes malvoyantes pour trouver leur chemin et éviter les obstacles en leur fournissant des instructions de navigation adaptées à leurs besoins spécifiques. Ces instructions peuvent inclure des conseils pour contourner les obstacles tels que les escaliers, les trottoirs ou les constructions.

Les aides à la mobilité : Les cannes blanches, les chiens-guides et autres aides à la mobilité sont des dispositifs qui aident les personnes malvoyantes à se déplacer de manière autonome en détectant les obstacles sur leur chemin. Ces aides peuvent aider l'utilisateur à éviter les obstacles et à prévenir les accidents.

Les lunettes à réalité augmentée : Les lunettes à réalité augmentée sont équipées de caméras, de capteurs et de logiciels qui peuvent aider les personnes malvoyantes à détecter les obstacles et à recevoir des instructions en temps réel pour les éviter. Ces lunettes peuvent fournir des informations visuelles ou sonores pour guider l'utilisateur de manière autonome.

Les applications mobiles : Les applications mobiles peuvent aider les personnes malvoyantes à détecter les obstacles et à recevoir des instructions pour les éviter en fournissant des informations sur l'environnement qui les entoure. Par exemple, certaines applications peuvent signaler la présence de trottoirs ou d'autres obstacles sur le chemin de l'utilisateur et lui fournir des itinéraires alternatifs pour les éviter.

1.2. Différentes méthodes d'évitement d'obstacles pour la navigation des individus atteints de troubles visuels :

Dans un système de navigation d'assistance, un module de navigation est nécessaire pour aider à la mise à jour de l'orientation et à la planification du trajet. Cependant, un module d'évitement des obstacles est bénéfique dans un tel système pour augmenter la sécurité. Les systèmes de navigation d'assistance en intérieur aident les utilisateurs dans leurs compétences en matière de mobilité tout en maintenant leur sécurité, comme les aides techniques pour aveugles et malvoyants. Pour garantir la sécurité, les utilisateurs doivent être informés des dangers potentiels et des obstacles. Pour surmonter ce problème, différentes méthodes d'évitement des obstacles ont été proposées ci-dessous. Elles sont basées sur la fonction de champ de potentiel, le sonar, le télémètre laser et la vision par ordinateur et sont brièvement discutées.

1.2.1. La fonction du champ de potentiel de direction et la logique floue :

La combinaison de champs de potentiel directionnels et de la logique floue est une approche avancée utilisée pour la navigation et la planification de trajectoires dans des environnements complexes. Le champ de potentiel directionnel guide le mouvement en attribuant des valeurs de potentiel et des directions à chaque point de l'espace. Cela permet à un objet ou à un agent de se déplacer vers des zones de faible potentiel tout en évitant les obstacles ou les zones de potentiel élevé. D'autre part, la logique floue est une technique qui permet de traiter des concepts vagues ou incertains en utilisant des ensembles flous et des règles floues. Elle est basée sur le principe que les valeurs peuvent être partiellement vraies ou fausses, plutôt que strictement vraies ou fausses.

Lorsqu'on combine le champ de potentiel directionnel avec la logique floue, cela permet de prendre en compte des informations supplémentaires et de rendre la planification de trajectoire plus flexible et adaptative.

La logique floue peut être utilisée pour ajuster les paramètres du champ de potentiel directionnel en fonction de conditions dynamiques ou de préférences spécifiques. Par exemple, des règles floues peuvent être définies pour moduler la vitesse, l'attraction ou la répulsion d'un objet ou d'un agent en fonction de la situation. Cela permet d'adapter le comportement en fonction des circonstances, comme éviter certains types d'obstacles, privilégier certains objectifs ou ajuster la vitesse en fonction des conditions environnementales.

La combinaison de champs de potentiel directionnels et de la logique floue offre donc une approche plus souple et adaptative pour la navigation et la planification de trajectoires. Elle permet de prendre en compte des informations plus complexes et d'ajuster le comportement en fonction de critères flous ou incertains. Cette approche est particulièrement utile dans des environnements dynamiques et complexes, où des décisions rapides et flexibles sont nécessaires pour une navigation efficace [1].

En 2018, BADOUCHE F [2] a développé un système de navigation et d'évitement d'obstacles en utilisant une combinaison de champs de potentiel directionnels et de logique floue. Cette méthode se base sur l'utilisation des directions relatives de la cible et des obstacles, la distance de la cible et la largeur angulaire des obstacles afin de déterminer le potentiel approprié pour orienter le robot mobile. Le potentiel calculé permet de contrôler l'accélération angulaire du robot mobile, le guidant ainsi vers la cible tout en l'éloignant des obstacles. Cette approche est particulièrement adaptée à la navigation locale des robots mobiles non holonomes, car elle permet de commander directement la direction, conduisant à des trajectoires fluides et à courbure continue. Elle a été conçue pour fonctionner avec une simple caméra de vision, sans informations de profondeur, mais peut également être utilisée avec d'autres types de capteurs. L'objectif est de mettre en place et de tester cette méthode sur un robot mobile non holonome. La logique floue est utilisée pour la méthode de contrôle des mouvements du même robot mobile. Pour générer la vitesse de rotation permettant au robot mobile de se déplacer vers sa destination tout en évitant les obstacles dans l'environnement, le contrôleur flou utilise quatre variables d'interaction entre le robot mobile et l'environnement inconnu. La figure 1 représente le schéma d'un Contrôleur flou évitement d'obstacles.

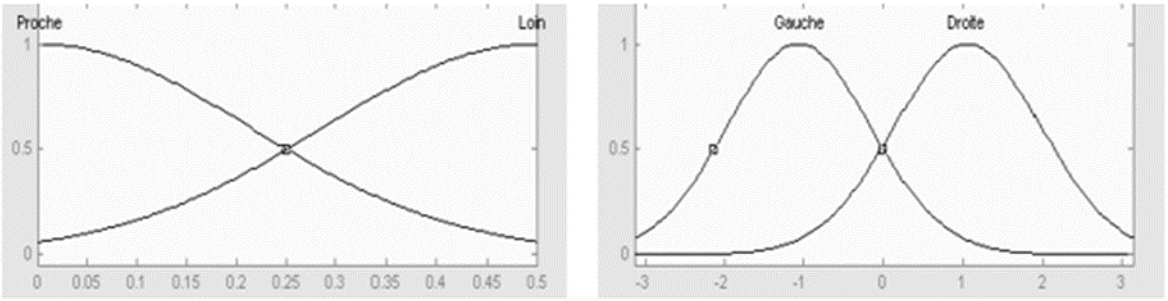

Fig.1 (a) À gauche fonction d'appartenance de d_O et **(b)** à droite celle de Θ_0.

Entrée d_O est la distance de l'obstacle par rapport au robot mobile

Entrée Θ_0 est l'angle de l'obstacle par rapport au robot mobile.

Sortie ω_o est la vitesse angulaire du robot mobile.

1.2.2. Le Sonar :

Le sonar est une méthode de détection de distance basée sur la réflexion du son. Dans cette méthode, un émetteur et un récepteur acoustique sont nécessaires. Tout d'abord, l'émetteur émet un signal acoustique court. Le chronomètre démarre et s'arrête lorsque le récepteur détecte la réflexion du signal acoustique. Il y a une limite de temps, si le chronomètre dépasse cette limite, il s'éteint. La distance entre le capteur et l'objet est égale au produit de la vitesse du son par le temps. Le temps est le temps total divisé par deux car le signal se déplace vers l'objet et revient. La vitesse du son dépend de la température et de l'humidité. Avec une bonne approximation, l'air peut être considéré comme un gaz idéal.

Les systèmes de navigation basés sur des capteurs ultrasoniques peuvent être considérés comme un choix courant dans la conception après les solutions basées sur la vision (caméra). Les systèmes proposés utilisant cette technologie fonctionnent en conjonction avec des cartes électroniques telles que Raspberry Pi ou Arduino. Une canne pour aveugles ultrasonique avec une sensibilité réglable à l'aide d'un capteur de proximité ultrasonique et d'un module GPS a été développé en [3]. Cette canne est basée sur la carte ARDUINO UNO. Elle est composée de trois sous-systèmes :

- Système de détection d'obstacle : Le circuit de détection d'obstacles est composé d'un capteur ultrasonique interfacé avec la carte Arduino Uno. Le capteur détecte la présence d'un obstacle dans chaque direction, puis la portée de l'obstacle est calculée. Si la distance est inférieure à 70 cm, le moteur de vibration vibrera avec la plus grande intensité et le buzzer sera également activé. Si la distance est comprise entre 70 cm et 150 cm, le moteur de vibration vibrera avec une intensité moyenne, et si la distance est supérieure à 250 cm, l'intensité du moteur de vibration sera moindre.
- Système de détection des trous : il est composé d'un capteur ultrasonique et d'un buzzer interfacé avec l'Arduino Uno. Le fonctionnement de ce circuit repose sur l'hypothèse que la hauteur du capteur ultrasonique monté sur la canne restera constante en cas de chemin plat. Mais si la hauteur augmente de manière significative par rapport au sol, dépassant un certain seuil, alors le buzzer commencera à retentir. Cela aidera les personnes malvoyantes à détecter la présence d'un trou ou d'un escalier devant eux.
- Appel d'urgence : Le module GSM_GPS est utilisé dans les situations d'urgence. Ce module reçoit les informations du satellite GPS au format NMEA (National Marine & Electronics Association) et transfère les informations de latitude et de longitude sous forme de message SMS vers un numéro de téléphone mobile prédéfini en cas d'urgence.

La figure 2 représente le synoptique de cette canne.

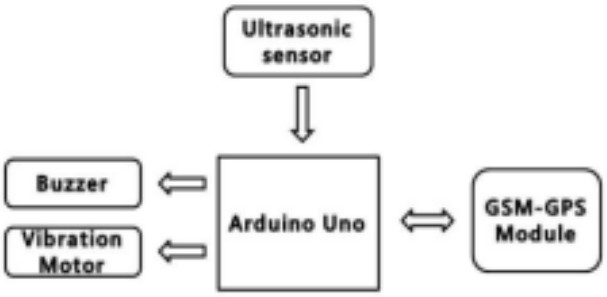

Fig 2. Synoptique de la canne intelligente

1.2.3. Le télémètre laser :

La télémétrie laser est une technique utilisée pour mesurer et obtenir des informations précises sur des objets ou des distances à l'aide de la lumière laser. Elle repose sur le principe de l'émission d'un faisceau laser vers un objet cible, puis la réception du faisceau réfléchi ou rétrodiffusé pour déterminer la distance, la vitesse ou d'autres paramètres.

Le processus de télémétrie laser implique généralement les étapes suivantes :

1. Émission du faisceau laser : Un laser émet un faisceau lumineux cohérent et directionnel vers l'objet cible que l'on souhaite mesurer.
2. Interaction avec l'objet cible : Le faisceau laser entre en contact avec l'objet cible et est réfléchi ou rétrodiffusé en fonction des propriétés de surface et de la composition de l'objet. Lors de cette interaction, le faisceau peut être diffusé, réfléchi ou absorbé.
3. Réception du faisceau réfléchi : Un récepteur ou un détecteur sensible reçoit le faisceau réfléchi ou rétrodiffusé provenant de l'objet cible. Ce faisceau contient des informations sur la distance, la vitesse ou d'autres caractéristiques de l'objet.
4. Analyse du signal reçu : Le signal reçu est analysé pour extraire les informations pertinentes. Cela peut être fait en mesurant le temps de vol du faisceau laser, c'est-à-dire le temps nécessaire pour que le faisceau aller-retour atteigne l'objet cible et revienne au détecteur. En utilisant la vitesse de la lumière comme référence, la distance peut être calculée avec précision.
5. Traitement des données : Les données de télémétrie laser peuvent être traitées et interprétées pour obtenir des informations telles que la distance, la vitesse, la position ou d'autres caractéristiques de l'objet cible. Ces données peuvent être utilisées dans diverses applications, notamment la cartographie, la surveillance, la robotique, la mesure de distance précise, etc.

La télémétrie laser offre de nombreux avantages, tels que des mesures rapides, précises et sans contact. Elle est utilisée dans divers domaines, notamment la topographie, l'industrie, la navigation, l'astronomie, la recherche scientifique et bien d'autres. Parmi les capteurs basés sur la télémetrie, le LIDAR.

Le LIDAR (Light Detection and Ranging) un capteur basé sur la télémétrie laser utilisé pour mesurer les distances et obtenir des informations précises sur l'environnement. Il trouve de

nombreuses applications dans divers domaines et offre des avantages tels que des mesures précises, des cartographies détaillées et une perception en temps réel de l'environnement.

En [4] l'auteur propose un système de navigation pour les personnes aveugles basé sur un système de navigation à détection de lumière et de mesure de distance (LiDAR). Les composants de cet appareil comprennent un système de charge, un capteur LiDAR, un microcontrôleur, une canne d'appel, deux buzzer, un moteur de vibration avec un émetteur RF, et un récepteur pour détecter les objets et fournir une assistance en temps réel aux personnes aveugles. Le fonctionnement de ce système consiste à détecter les obstacles le long du trajet d'une personne aveugle et à notifier l'utilisateur à l'aide du buzzer et du moteur de vibration. De plus, en cas d'urgence, le buzzer se déclenche et informe la personne malvoyante utilisant l'appareil. D'autre part, le microcontrôleur et les autres modules de l'appareil entretiennent une relation continue qui est bénéfique pour l'utilisateur. Un autre avantage appréciable de cet appareil est sa capacité à recharger la batterie. Il s'agit d'une solution peu coûteuse, rapide, simple à utiliser et novatrice pour les personnes aveugles. Le circuit du système est illustré dans la figure 3.

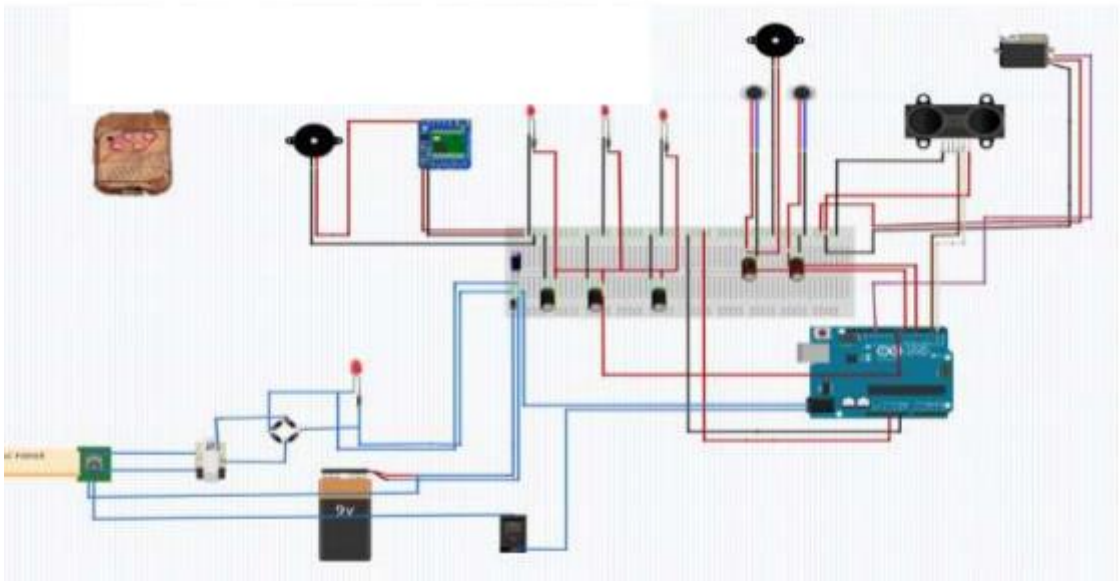

Fig 3. Circuit du système.

1.2.4. La vision par ordinateur :

Les méthodes basées sur la vision par ordinateur sont couramment utilisées dans les systèmes d'évitement d'obstacles. Pour utiliser ces techniques de vision par ordinateur, différents types de caméras ont été utilisés. En fonction du type de caméra, différentes approches ont été mises en œuvre dans le but d'éviter les obstacles. Le type de caméra principal utilisé dans ce cas est la caméra monoculaire. Comme son nom l'indique, cette méthode n'utilise qu'une seule caméra. De manière intuitive, la distance entre la caméra et l'obstacle est proportionnelle à la distance focale de la caméra.

Contrairement à la vision monoculaire, la stéréo vision utilise deux caméras pour détecter les obstacles et estimer la distance des objets par rapport aux caméras. Cette méthode est largement utilisée dans l'évitement d'obstacles et la détection d'objets. Les caméras stéréo suivent les mêmes principes qu'une paire d'yeux. Dans cette méthode, les deux caméras sont positionnées en direction du même point. Elles sont placées côte à côte à une courte distance l'une de l'autre. Elles sont positionnées de manière à ce que leurs axes focaux soient parallèles. Dans cette configuration, chaque caméra voit une vue légèrement différente. La différence est due à la courte distance entre les emplacements des caméras. Cette distance permet le calcul de la position des objets grâce à la triangulation. La position précise de la caméra est essentielle pour une estimation précise de l'emplacement des obstacles. La stéréo vision détermine la distance jusqu'à un objet en comparant les deux images. Les informations de distance fournies par cette procédure permettent au système de construire une image en 3D de l'environnement. La carte en 3D dense, qui est créée à partir de la carte de disparité, est utile pour détecter les obstacles et estimer leurs distances par rapport à l'utilisateur. La figure 4 représente le principe de fonctionnement de la stéréo vision [5].

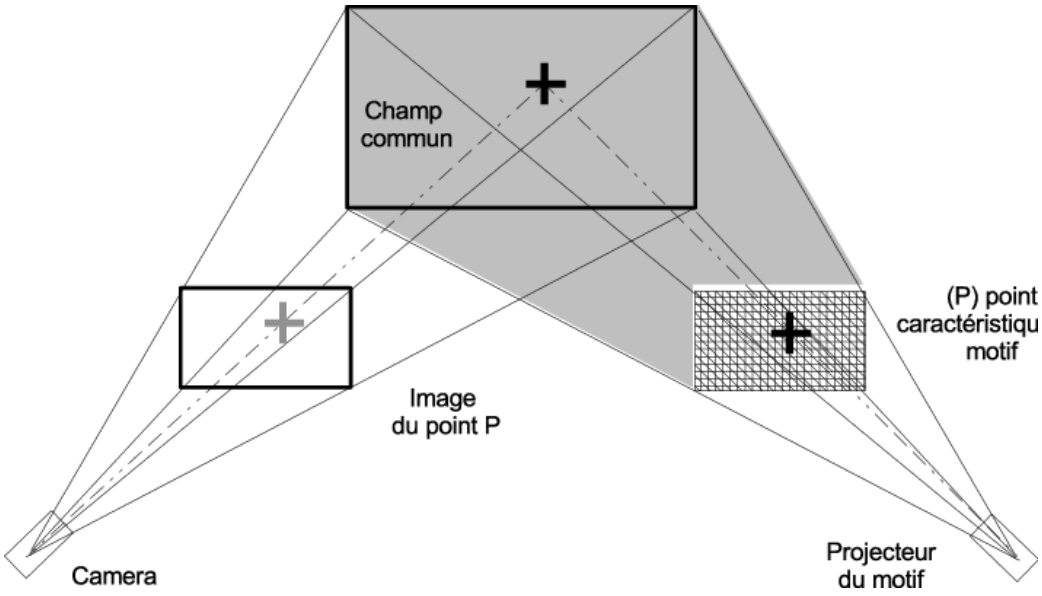

Fig 4. Principe de fonctionnement de la stéréo vision

En [6] un système qui intègre à la fois la vision stéréo et les capteurs ultrasoniques pour obtenir une conscience précise des obstacles présents dans l'environnement en temps réel a été développé. Le concept central de ce système est de catégoriser les points de sommet pour la détection d'objets dans l'espace tridimensionnel en fonction de la hauteur, de la largeur et de la pente des sommets adjacents présents dans les images capturées par les caméras CCD. Le système d'assistance prévu vise à être plus rentable pour les personnes malvoyantes grâce à l'utilisation de caméras CCD et de capteurs à faible coût. Dans sa phase de reconnaissance, le système d'assistance utilise l'algorithme de détection d'obstacles proposé en utilisant la vision stéréo pour reconnaître et suivre les obstacles situés à une plus grande distance. En revanche, il utilise des capteurs ultrasoniques pour détecter les obstacles présents à des distances plus courtes. Le système surmonte également le problème de la détection des trous et des escaliers descendant, rencontré par les dispositifs utilisant une vision monoculaire. Des fonctionnalités supplémentaires telles que le buzzer, le GPS, le module vocal et les messages d'alerte sont également incorporées. La figure 5 représente le synoptique de ce système.

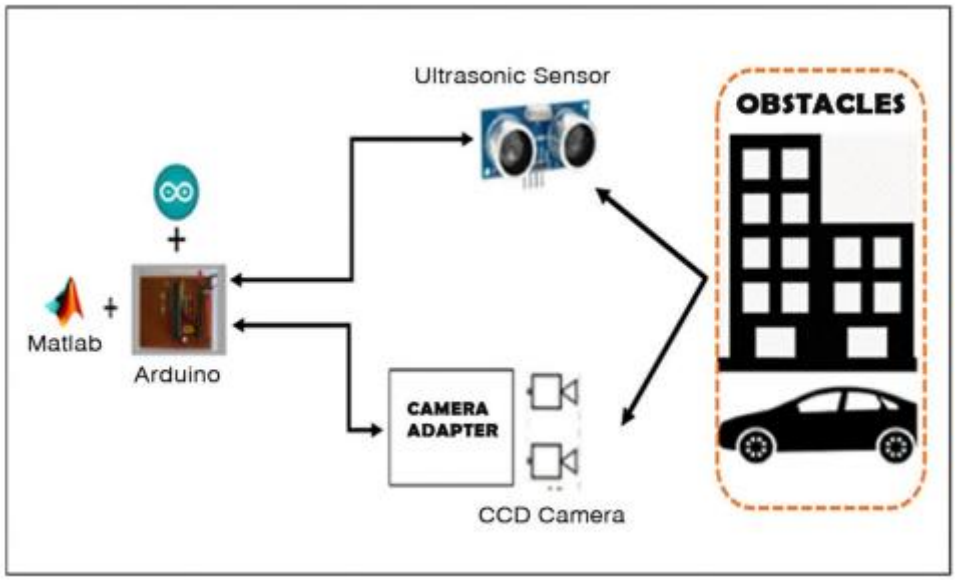

Fig 5. Synoptique du système à stéréovision

La méthodologie du système est représentée dans la figure 6. Le système se compose d'une combinaison de la stéréo vision et de capteurs. Le calcul des obstacles situés à longue distance se fait grâce à la stéréo vision, tandis que les obstacles situés à courte distance sont détectés à l'aide du capteur ultrasonique, permettant d'obtenir une conscience précise des obstacles dans l'environnement en temps réel. L'architecture interne comprend le module de calcul de distance des obstacles par stéréo vision utilisant des techniques de traitement d'image telles que l'échantillonnage d'image, la conversion du rouge au niveau de gris, l'estimation de disparité, la génération de carte de profondeur, la génération de carte d'obstacles et la génération de carte polaire. Le module de l'architecture interne est connecté au module Arduino (ATmega8), qui est également connecté à d'autres composants tels que les capteurs ultrasoniques (Ultrasonic-HCHR04), le module Wi-Fi (ESP8266), le module GSM (SIM9004), le capteur LDR (LDR07), le module vocal, le buzzer et le module GPS

(GPSNEO1), qui constituent ensemble l'architecture externe du système. Lorsque le système est allumé, les obstacles situés à plus grande distance sont détectés à l'aide de la stéréo vision (qui peut être installée sur un casque, reliée au module Arduino présent sur la canne grâce à un câble) et le capteur ultrasonique détecte les obstacles situés à plus courte distance, permettant d'obtenir un calcul précis de tous les obstacles présents dans l'environnement. En cas de présence d'un obstacle, le buzzer se met à sonner, alertant l'utilisateur afin qu'il prenne les précautions nécessaires. L'utilisateur peut également envoyer un message d'alerte/notification à l'aide du module Wi-Fi et du module GSM. Le module Wi-Fi envoie la localisation de l'utilisateur au cloud connecté. Le module GSM permet d'envoyer une notification au numéro de téléphone enregistré (qui peut être la personne de contact en cas d'urgence) en appuyant sur un bouton. La notification fournit également la localisation actuelle de la personne à l'aide du module GPS. Le buzzer et l'alerte de notification fonctionnent simultanément, ce qui en fait un système de réponse immédiate. Le module GSM permet également de recevoir des messages provenant du numéro enregistré. Le message sera communiqué à l'utilisateur à l'aide d'un module vocal. Parfois, l'utilisateur peut se déplacer dans un endroit sombre où il serait difficile pour les personnes environnantes d'identifier la présence d'une personne handicapée. Ainsi, le capteur LDR est activé et produit une source de lumière pour une meilleure indication aux personnes aux alentours.

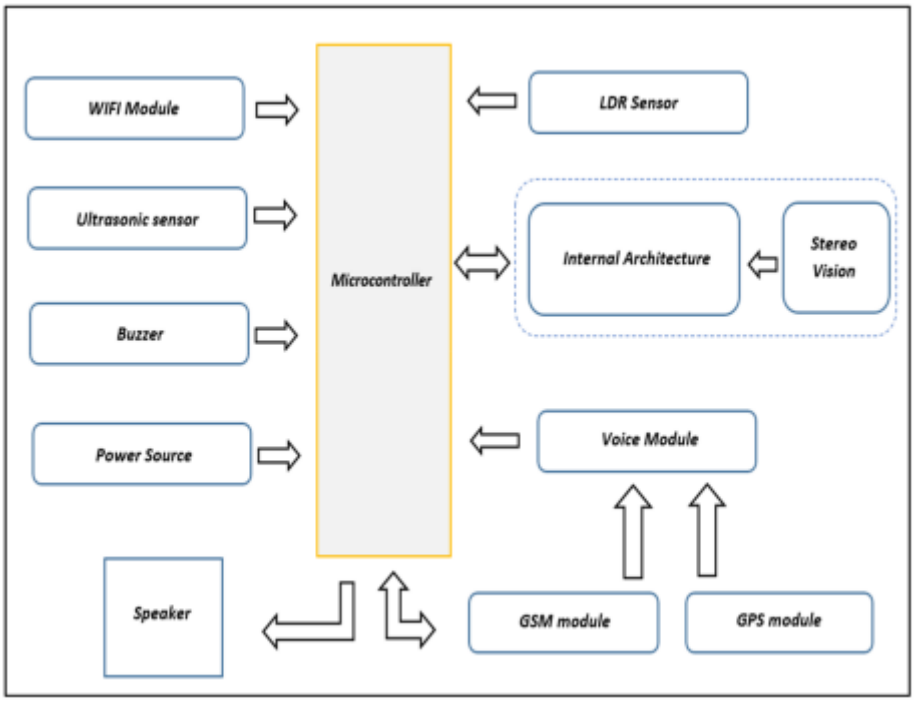

Fig 6. La méthodologie du système

Conclusion :

Dans ce chapitre, nous avons exploré différentes méthodes d'évitement d'obstacles utilisées pour les aveugles. Chaque technique présente des avantages et des limitations spécifiques, et leur choix dépendra des exigences du système et des contraintes de l'environnement.

- La combinaison de la fonction de potentiel directionnel et de la logique floue offre une approche flexible et adaptative pour la navigation autonome. En utilisant la fonction de potentiel directionnel, le robot peut déterminer une direction préférée en fonction des forces attractives et répulsives exercées par les obstacles. La logique floue permet de prendre en compte des facteurs contextuels et d'ajuster les poids des potentiels en fonction de la situation, ce qui rend le système plus résilient et capable de prendre des décisions intelligentes.
- Le sonar, ou capteur à ultrasons, est une méthode couramment utilisée pour détecter les obstacles à proximité. Il émet des ondes sonores et mesure le temps de retour de l'écho pour estimer la distance aux objets. Le sonar est robuste, abordable et convient aux environnements intérieurs ou extérieurs.
- La télémétrie laser, ou LIDAR, utilise des faisceaux laser pour mesurer la distance et la position des objets dans l'environnement. Cette méthode offre une grande précision et permet de construire une carte 3D de l'environnement. Les capteurs LIDAR sont couramment utilisés dans les véhicules autonomes et les robots de navigation.
- La vision par ordinateur est une approche basée sur l'analyse des images pour détecter et éviter les obstacles. Elle peut utiliser des caméras mono ou stéréo pour estimer la distance et la position des objets. La vision par ordinateur offre une grande flexibilité et la possibilité de traiter des informations visuelles complexes, mais peut être plus sensible aux variations d'éclairage et aux conditions environnementales.

En conclusion, chaque méthode d'évitement d'obstacles présente ses propres avantages et limitations, et leur choix dépendra des exigences spécifiques du système et des contraintes de l'environnement. La combinaison de différentes techniques peut également être envisagée pour améliorer la robustesse et la précision de détection et d'évitement des obstacles.

Chapitre 2

La logique floue

Introduction :

Dans de nombreux domaines de la science et de l'ingénierie, nous sommes souvent confrontés à des situations où l'incertitude est omniprésente. La logique traditionnelle binaire, qui se base sur des valeurs vraies ou fausses, peut être limitée pour traiter ces situations complexes. C'est là que la logique floue entre en jeu. La logique floue, également connue sous le nom de logique floue ou logique incertaine, est une extension de la logique classique qui permet de modéliser et de raisonner avec des concepts vagues ou imprécis. Dans ce chapitre, on va présenter les principes fondamentaux de la logique floue et son application dans divers domaines.

2.1. La logique floue :

La logique floue est une technique pour le traitement de connaissances imprécises basées sur des termes linguistiques ; elle donne les moyens de convertir une commande linguistique basée sur le raisonnement humain, en une commande automatique, permettant ainsi la commande des systèmes complexes dont les informations sont exprimées d'une façon vague et mal définie.

Le concept de la logique floue fut réellement introduit en 1965 par Lotfi Zadeh, un professeur d'électronique à l'université de Berkeley (USA). Sa théorie « fuzzy set theory » n'eut pas un succès immédiat, elle fut développée surtout en Europe et au japon. Elle est très attractive, parce qu'elle est basée sur le raisonnement intuitif et prend en compte la subjectivité et l'imprécision. Mais elle n'est pas imprécise. C'est une théorie mathématique rigoureuse, adaptée au traitement de tout ce qui est subjectif et/ou incertain [7].

Le système à base de la logique floue est composé de quatre blocs principaux (Figure 7) :

- Fuzzificateur.
- Base de connaissances floues.
- Interférence floue.
- Défuzzicateur.

Chacun de ces blocs fera l'objet d'un développement détaillé.

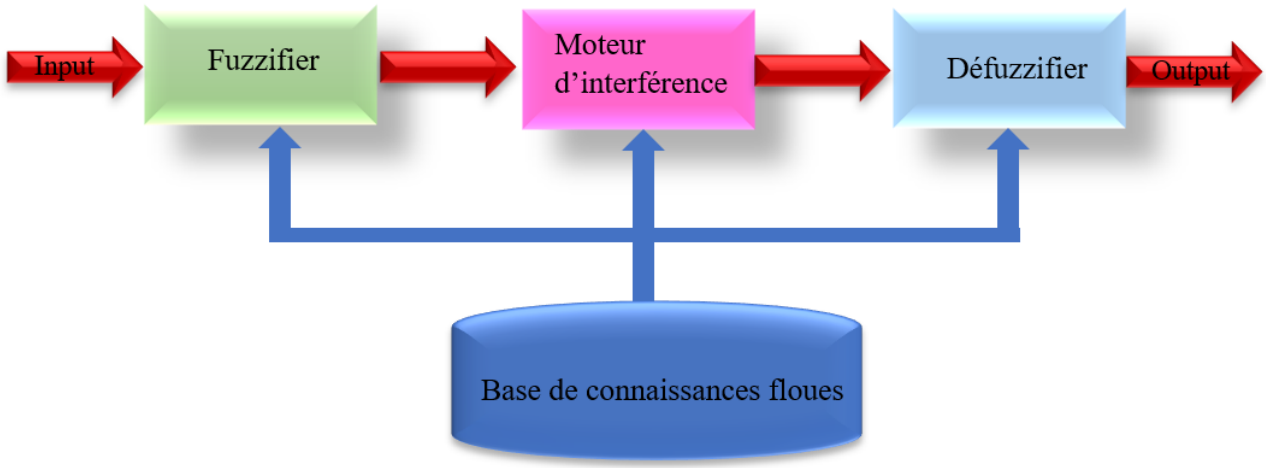

Fig 7. Structure générale d'un système basé sur la logique floue

2.1.1. Fuzzification :

La fuzzification est un process qui effectue la conversion de l'entrée numérique observée [8].

$U_0 = (U_{01}, U_{02}, ..., U_{0n}) \in U$ En un ensemble flou $F_X = F_{X1} \times F_{X2} \times ..... \times F_{Xn}$ définie dans U (2.1)

Pour le système flou, la fuzzification des variables est une étape importante du processus de mise en œuvre. Ses étapes consistent à :

- Etablir les variables linguistiques.
- Établir les quantificateurs flous (nombre de valeurs linguistiques)
- Attribuer une signification numérique à chaque quantificateur flou : fonction d'appartenance.

Le choix de la forme des fonctions d'appartenance (triangulaires, trapézoïdales, exponentielles, gaussiennes, …) est arbitraire. Les formes triangulaires facilitent la programmation ce qui explique qu'elles soient le plus fréquemment utilisées. Quant au nombre de fonctions d'appartenance, il est généralement impair car elles se répartissent autour de zéro.

Dans les applications de la logique floue, les données observées sont souvent numériques, mais dans le système flou la manipulation des données est basée sur la théorie des ensembles flous. Il y a au minimum deux choix pour cette conversion :

- **Fuzzification singleton :**

Où l'opérateur de fuzzification converti l'entrée numérique $U_0 \in U$ en un singleton flou F_X dans U tel que :

$$\begin{aligned} \mu_{Fx}(U) &= 1 \quad si \quad U = U_0 \\ \mu_{Fx}(U) &= 0 \quad si \quad U \neq U_0 \end{aligned} \qquad (2.2)$$

Cette stratégie est largement utilisée dans les applications de contrôle flou, car elle est facile à implémenter.

- **Fuzzification non – singleton :**

Est une fuzzification pour laquelle $\mu_{Fx}(U)$ est égal à l'unité si $U = U_0$ et décroît quand on s'éloigne de U_0.

2.1.2. Interférence floue :

On appelle règles d'interférence, l'ensemble des différentes règles reliant les variables floues d'entrée d'un système aux variables floues de sortie de ce système [9]. Ces règles se présentent sous la forme :

- **Si** condition 1 **et/ou** condition 2 **alors** action sur les sorties.

En d'autres termes, cette opération quantifie la force de la liaison entre la prémisse et la conclusion de la règle.

Les opérateurs les plus courants en commande sont de type conjonctif :

- L'implication de Mamdani (1974) :

$$\mu_R(x,y) = \min(\mu_A(x), \mu_B(x)) \tag{2.3}$$

- L'implication de Larsen (1980) :

$$\mu_R(x,y) = \mu_A(x) \times \mu_B(x) \tag{2.4}$$

- Dans ce travail, on va s'intéressé à l'implication de Mamdani.

- **La méthode de Mamdani :**

Cette méthode réalise :

- L'opérateur "ET" par la fonction "Min"
- La conclusion "ALORS" de chaque règle par la fonction "Min"
- 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.

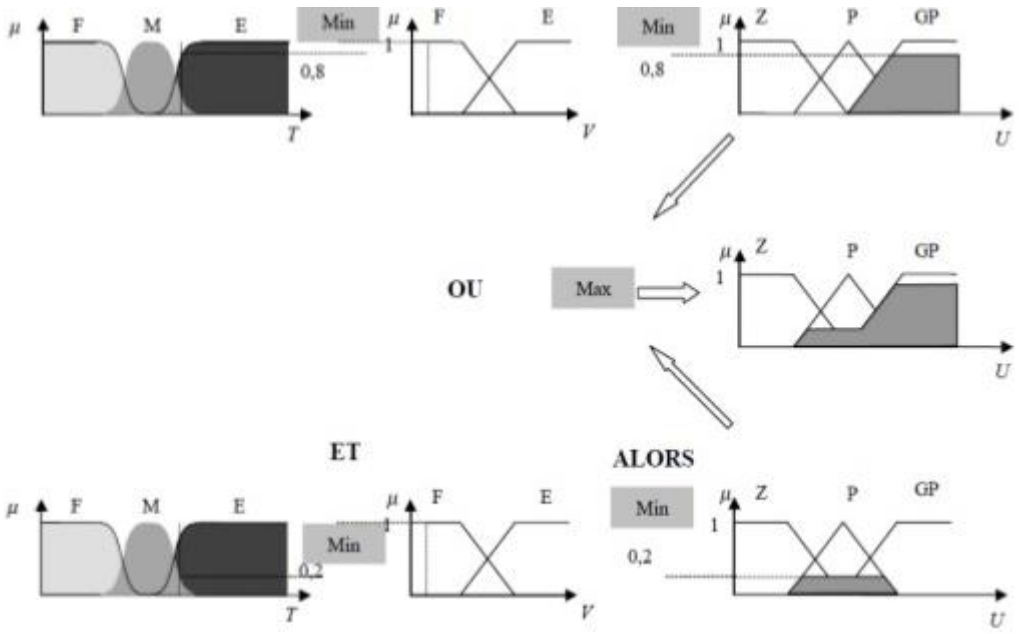

Fig 8. Exemple d'interférence Max-Min

2.1.3. La défuzzification :

Les valeurs obtenues lors de la combinaison des règles appliquées aux intervalles flous de la variable de sortie, définies une fonction d'appartenance [10]. Il s'agit de convertir cette

information en une grandeur physique. Plusieurs façons de faire, peuvent être envisagées mais, en pratique, on utilise surtout les deux méthodes suivantes :

- Défuzzification par calcul du centre de gravité.
- Défuzzification par calcul de moyenne de maximum.

En général, c'est la défuzzification par calcul du centre de gravité qui est la plus utilisée.

- **Défuzzification par calcul du centre de gravité :**

Il s'agit de calculer le centre de gravité de la fonction d'appartenance de la variable de sortie. Le calcul du centre de gravité permet bien d'obtenir une seule valeur pour la grandeur de sortie. Son calcul est cependant relativement complexe puisqu'il nécessite le calcul d'une intégrale, ou dans le cas simple de fonctions d'appartenance en raies, d'une somme pondérée. Le calcul de la solution précise U_0 , d'une solution floue à l'aide de la méthode du centre de gravité est défini par :

$$U_0 = \frac{\sum_{i=1}^{n} \mu_i(x).x_i}{\sum_{i=1}^{n} \mu_i(x)} \qquad (2.5)$$

Avec :

$\mu_i(x)$: fonction d'appartenance

x_i : les abscisses

- **Méthode de moyenne de maximum :**

Cette méthode génère une commande précise en calculant la moyenne des valeurs pour lesquelles l'appartenance est maximale.
Si la fonction est discrétisée, la valeur défuzzifiée est donnée par :

$$U_0 = \frac{\sum_{i=1}^{l} r_i}{l} \qquad (2.6)$$

Avec :
l : le nombre de valeurs quantifiées
r : valeurs pour lesquelles l'appartenance est maximale.

2.1.4. Base de connaissance (Base de règles) :

C'est la partie du contrôleur flou dans lequel se trouvent sous forme de règles heuristiques, les connaissances de l'opérateur [11]. Elle est composée d'une base de données et d'une base de règles.

- La base de données regroupe :
- Les ensembles flous associés aux variables d'entrée et de sortie du contrôleur flou.
- Les facteurs d'échelle (gains) en entrée (normalisation) et en sortie (dénormalisation).
- La base de règle contient des règles de la forme :

« **Si** X_1 est A_1 et X_2 est A_2 …et X_n est A_n **Alors** Y est B »
Suivant la nature de *B* on parlera de :

- Règles à conclusion symbolique (contrôleur de type Mamdani) : *B* est une valeur linguistique.
- Règles à conclusion algébrique (contrôleur de Sugeno) : *B* est une valeur numérique (singleton) ou une équation mathématique bien précise (non floue).

2.2. Principe de fonctionnement de la logique floue :

Le principe de fonctionnement de la logique floue repose sur la représentation et la manipulation de l'incertitude à l'aide de concepts flous plutôt que binaires [12]. Contrairement à la logique classique, qui utilise des valeurs vraies ou fausses, la logique floue permet de modéliser des degrés de vérité partiels.

Voici les étapes du fonctionnement de la logique floue :

- Définition des ensembles flous : La logique floue utilise des ensembles flous pour représenter des concepts vagues ou imprécis. Chaque ensemble flou est défini par une fonction d'appartenance qui associe à chaque élément un degré d'appartenance compris entre 0 et 1. Par exemple, pour représenter la notion de "hauteur élevée", un ensemble flou pourrait être défini avec une fonction d'appartenance qui attribue des degrés d'appartenance élevés aux éléments considérés comme ayant une hauteur élevée et des degrés d'appartenance faibles aux éléments considérés comme ayant une hauteur basse.

- Utilisation de variables linguistiques : La logique floue utilise des variables linguistiques pour décrire des concepts vagues. Au lieu d'utiliser des valeurs numériques précises, nous utilisons des termes linguistiques tels que "grand", "petit", "chaud" ou "froid". Ces termes sont associés à des ensembles flous qui représentent les degrés d'appartenance correspondants. Par exemple, une variable linguistique "température" peut être associée à un ensemble flou "chaud" dont la fonction d'appartenance attribue des degrés d'appartenance élevés aux températures élevées et des degrés d'appartenance faibles aux températures basses.

- Raisonnement flou : La logique floue permet de raisonner avec des ensembles flous en utilisant des opérations spécifiques. L'opération de complément flou permet de représenter la négation d'un ensemble flou, tandis que l'opération d'intersection floue permet de combiner deux ensembles flous pour représenter leur coïncidence. Par exemple, si nous avons un ensemble flou représentant la "hauteur élevée" et un autre ensemble flou représentant la "largeur élevée", nous pouvons utiliser l'opération d'intersection floue pour obtenir un nouvel ensemble flou représentant les objets qui sont à la fois "hauts" et "larges".

- Inférence floue : L'inférence floue est un processus permettant de déduire de nouvelles informations à partir de règles floues. Celles-ci sont des énoncés de la forme "Si [condition], alors [conclusion]", où la condition et la conclusion sont des propositions floues. Les règles floues peuvent être utilisées pour prendre des décisions ou tirer des conclusions à partir de valeurs d'entrée incertaines. Par exemple, si nous avons une

règle floue telle que "Si la température est élevée, alors la climatisation doit être activée", nous pouvons utiliser cette règle pour décider si la climatisation doit être activée en fonction du degré d'appartenance de la température à l'ensemble flou "élevée".

En combinant ces étapes, la logique floue permet de modéliser et de raisonner avec des concepts vagues ou incertains de manière plus naturelle. Elle offre une approche flexible et puissante pour gérer l'incertitude et prendre des décisions dans des environnements complexes.

2.3. Application de la logique floue :

La logique floue trouve des applications dans de nombreux domaines en raison de sa capacité à gérer l'incertitude et les concepts vagues de manière efficace. Voici quelques exemples supplémentaires d'applications de la logique floue :

2.3.1. Systèmes de recommandation :

Les systèmes de recommandation utilisent souvent la logique floue pour modéliser les préférences des utilisateurs de manière plus précise [13]. En utilisant des ensembles flous pour représenter les préférences et les caractéristiques des produits, les systèmes de recommandation basés sur la logique floue peuvent fournir des recommandations plus personnalisées et pertinentes. Voici quelques exemples d'application de cette technique dans ces systèmes :

- Recommandations de produits : Dans les systèmes de recommandation basés sur la logique floue, les préférences des utilisateurs et les caractéristiques des produits sont représentées à l'aide d'ensembles flous. Ceux-ci permettent de capturer les nuances et les degrés de préférence des utilisateurs. Par exemple, au lieu de simplement indiquer si un utilisateur aime ou n'aime pas un produit, la logique floue permet de représenter des évaluations telles que "fortement préféré", "modérément préféré" ou "légèrement préféré". Cela permet de fournir des recommandations plus personnalisées et adaptées aux préférences individuelles des utilisateurs.
- Recommandations de films et de musique : Dans le domaine des recommandations de films et de musique, la logique floue peut être utilisée pour modéliser les goûts des utilisateurs et les caractéristiques des œuvres artistiques. Les variables linguistiques telles que "joyeux", "mélancolique", "énergique" ou "calme" peuvent être utilisées pour décrire les genres ou les ambiances des films et de la musique. En utilisant des ensembles flous pour représenter ces variables, les systèmes de recommandation basés sur la logique floue peuvent proposer des suggestions qui correspondent mieux à l'humeur ou à l'ambiance recherchée par les utilisateurs.
- Recommandations de livres et de produits similaires : La logique floue peut également être appliquée dans les recommandations de livres et de produits similaires. En utilisant des ensembles flous pour représenter des critères tels que le genre, le style d'écriture, le niveau de difficulté ou le niveau d'intérêt, les systèmes de recommandation peuvent suggérer des livres ou des produits qui correspondent aux préférences des utilisateurs. Par exemple, un utilisateur qui préfère les romans "romantiques" et "faciles à lire" peut recevoir des recommandations qui répondent à ces critères en particulier.
- Recommandations de restaurants : Dans les systèmes de recommandation de restaurants, la logique floue peut être utilisée pour prendre en compte des facteurs tels

que la cuisine, l'ambiance, les prix et les évaluations des utilisateurs. Les ensembles flous peuvent être utilisés pour représenter des préférences linguistiques telles que "très épicé", "ambiance décontractée" ou "abordable". Cela permet de fournir des recommandations de restaurants qui correspondent aux préférences culinaires et aux attentes des utilisateurs.

2.3.2. Robotique :

La logique floue est largement utilisée dans le domaine de la robotique pour permettre aux robots de prendre des décisions en fonction de données sensorielles incertaines [14]. Les robots peuvent utiliser des ensembles flous pour modéliser des informations telles que la distance, l'orientation et l'obstacle évitant ainsi de manière plus flexible et réactive.

La logique floue trouve de nombreuses applications dans le domaine de la robotique, permettant aux robots de prendre des décisions basées sur des données sensorielles incertaines. Voici quelques exemples d'application de la logique floue dans la robotique :

- Navigation robotique : Dans la navigation robotique, la logique floue peut être utilisée pour permettre au robot de prendre des décisions de déplacement en fonction de données sensorielles telles que la distance aux obstacles, l'orientation et les conditions environnementales. En utilisant des ensembles flous pour représenter ces informations, le robot peut déterminer la direction et la vitesse de déplacement appropriées, en tenant compte des variations et des incertitudes présentes dans l'environnement.
- Contrôle de mouvement : La logique floue est utilisée dans le contrôle de mouvement des robots pour réguler la vitesse, la force et la précision des mouvements. En utilisant des ensembles flous pour représenter les objectifs de mouvement et les mesures de performance, le système de contrôle peut ajuster en temps réel les paramètres du mouvement pour s'adapter à des situations variables et incertaines. Cela permet aux robots d'effectuer des tâches avec plus de fluidité et de s'adapter à des conditions de fonctionnement changeantes.
- Planification de trajectoire : La logique floue peut être utilisée pour la planification de trajectoire des robots, en prenant en compte des facteurs tels que la sécurité, l'efficacité et les contraintes d'environnement. En utilisant des ensembles flous pour représenter les objectifs de la trajectoire et les conditions environnementales, le système de planification peut générer des trajectoires qui tiennent compte de la variabilité et des incertitudes présentes dans l'environnement. Cela permet aux robots de naviguer de manière plus souple et d'éviter les obstacles de manière plus efficace.
- Manipulation d'objets : Dans la manipulation d'objets, la logique floue peut être utilisée pour réguler la force, la précision et la stabilité du mouvement du robot lorsqu'il interagit avec des objets. En utilisant des ensembles flous pour représenter les caractéristiques de l'objet, les capacités du robot et les mesures de performance, le système de contrôle peut ajuster la force et la vitesse de la manipulation pour s'adapter à des variations et des incertitudes dans les propriétés des objets et les conditions de la tâche.
- Prise de décision autonome : La logique floue est également appliquée dans la prise de décision autonome des robots. En utilisant des ensembles flous pour représenter les objectifs, les contraintes et les risques, les robots peuvent prendre des décisions en tenant compte de multiples facteurs et en évaluant les alternatives de manière plus flexible. Cela permet aux robots de s'adapter à des situations imprévues et de prendre des décisions appropriées dans des environnements dynamiques et incertains.

2.3.3. Traitement du langage naturel :

La logique floue est également appliquée dans le domaine du traitement du langage naturel pour gérer l'ambiguïté et l'imprécision des expressions linguistiques [15]. En utilisant des ensembles flous pour représenter la signification des mots et des phrases, la logique floue permet d'améliorer la compréhension des instructions et des requêtes formulées en langage naturel. Voici quelques exemples d'application de la logique floue dans le traitement du langage naturel :

- Systèmes de question-réponse : Les systèmes de question-réponse utilisent souvent la logique floue pour comprendre les questions formulées en langage naturel et fournir des réponses appropriées. En utilisant des ensembles flous pour représenter les concepts et les relations sémantiques, les systèmes de question-réponse basés sur la logique floue peuvent traiter des questions ambiguës ou imprécises en générant des réponses qui correspondent à un certain degré de pertinence plutôt qu'à une réponse binaire.
- Analyse des sentiments : Dans l'analyse des sentiments, la logique floue peut être utilisée pour évaluer les opinions et les émotions exprimées dans des textes. En utilisant des ensembles flous pour représenter des concepts tels que "positif", "négatif" et "neutre", les systèmes d'analyse des sentiments basés sur la logique floue peuvent attribuer des degrés de polarité aux expressions linguistiques, permettant ainsi une évaluation plus fine des sentiments exprimés.
- Reconnaissance d'intention : La logique floue est utilisée dans la reconnaissance d'intention pour comprendre les intentions des utilisateurs à partir de requêtes en langage naturel. En utilisant des ensembles flous pour représenter des intentions telles que "demander des informations", "faire une réservation" ou "passer une commande", les systèmes de reconnaissance d'intention basés sur la logique floue peuvent attribuer des degrés de pertinence à différentes intentions, facilitant ainsi la compréhension des requêtes ambiguës ou partielles.
- Traduction automatique : Dans la traduction automatique, la logique floue peut être utilisée pour gérer les ambiguïtés et les variations dans les traductions. En utilisant des ensembles flous pour représenter des concepts linguistiques et des règles de correspondance, les systèmes de traduction automatique basés sur la logique floue peuvent générer des traductions qui prennent en compte les variations sémantiques et syntaxiques entre les langues, offrant ainsi des résultats plus adaptés et naturels.
- Correction automatique de texte : La logique floue est appliquée dans la correction automatique de texte pour suggérer des corrections aux erreurs orthographiques ou grammaticales. En utilisant des ensembles flous pour représenter les probabilités de correction et les alternatives possibles, les systèmes de correction automatique basés sur la logique floue peuvent générer des suggestions de correction qui tiennent compte du contexte et des erreurs potentielles, permettant ainsi des corrections plus précises.

2.3.4. Prévisions météorologiques :

La logique floue est utilisée dans les modèles de prévision météorologique pour traiter l'incertitude associée aux conditions atmosphériques [16]. En utilisant des ensembles flous pour représenter les variables météorologiques telles que la température, l'humidité et la pression atmosphérique, la logique floue permet de générer des prévisions plus précises et adaptées aux variations naturelles.

La logique floue trouve des applications dans les prévisions météorologiques en raison de sa capacité à traiter l'incertitude et les variations des conditions atmosphériques. Voici quelques exemples d'application de la logique floue dans les prévisions météorologiques :

- Prévisions de précipitations : La logique floue est utilisée pour prédire les niveaux de précipitations en prenant en compte des facteurs tels que l'humidité, la pression atmosphérique, la température et les modèles de circulation atmosphérique. En utilisant des ensembles flous pour représenter ces variables, les systèmes de prévision météorologique basés sur la logique floue peuvent générer des estimations des niveaux de précipitations, en tenant compte des incertitudes inhérentes aux mesures et aux modèles.
- Prévisions de température : La logique floue peut être utilisée pour prédire les variations de température en utilisant des ensembles flous pour représenter les données historiques, les tendances saisonnières et les conditions atmosphériques actuelles. En considérant des variables floues telles que "chaud", "froid" et "tempéré", les systèmes de prévision météorologique basés sur la logique floue peuvent générer des estimations de température qui reflètent les variations et les incertitudes inhérentes aux prévisions.

- Prévisions de vent : La logique floue peut être appliquée dans les prévisions de vent en utilisant des ensembles flous pour représenter des variables telles que la vitesse du vent, la direction du vent et les conditions topographiques. En considérant des variables linguistiques telles que "fort", "faible", "nord", "sud", "côtier" ou "montagneux", les systèmes de prévision météorologique basés sur la logique floue peuvent générer des prévisions de vent plus précises et adaptées aux variations locales.

- Prévisions d'ensoleillement : La logique floue peut être utilisée pour prédire l'ensoleillement en tenant compte de facteurs tels que les nuages, les conditions météorologiques et la position du soleil. En utilisant des ensembles flous pour représenter des variables linguistiques telles que "ensoleillé", "partiellement nuageux" ou "couvert", les systèmes de prévision météorologique basés sur la logique floue peuvent générer des estimations de l'ensoleillement qui prennent en compte les variations et les incertitudes inhérentes aux conditions atmosphériques.

- Prévisions de visibilité : La logique floue peut être appliquée pour prédire la visibilité en tenant compte de facteurs tels que le brouillard, la pollution atmosphérique et les conditions météorologiques. En utilisant des ensembles flous pour représenter des variables linguistiques telles que "claire", "bonne", "modérée" ou "mauvaise", les systèmes de prévision météorologique basés sur la logique floue peuvent générer des estimations de visibilité qui reflètent les variations et les incertitudes inhérentes aux conditions atmosphériques.

2.3.5. Contrôle de la qualité :

La logique floue est appliquée dans le contrôle de la qualité pour évaluer les caractéristiques et les performances des produits. En utilisant des ensembles flous pour représenter les critères de qualité et les mesures, la logique floue permet de prendre des décisions sur l'acceptabilité des produits en tenant compte de la variabilité et des tolérances.

La logique floue est utilisée dans le contrôle de qualité pour prendre en compte les variations et les incertitudes présentes dans les processus de production et les mesures de qualité. Voici quelques exemples d'application de la logique floue dans le contrôle de qualité :

- Contrôle dimensionnel : Dans le contrôle dimensionnel, la logique floue peut être utilisée pour évaluer les mesures de pièces ou de produits par rapport aux spécifications. En utilisant des ensembles flous pour représenter des concepts tels que "bon", "acceptable" ou "hors tolérance", les systèmes de contrôle de qualité basés sur la logique floue peuvent évaluer les dimensions d'une pièce en tenant compte des variations et des incertitudes de mesure, permettant ainsi une évaluation plus précise de la conformité.
- Contrôle de la qualité des produits : La logique floue est utilisée pour évaluer la qualité globale des produits en prenant en compte plusieurs caractéristiques de qualité. En utilisant des ensembles flous pour représenter des concepts de qualité tels que "excellent", "bon", "moyen" ou "médiocre", les systèmes de contrôle de qualité basés sur la logique floue peuvent agréger les mesures de différentes caractéristiques de qualité pour donner une évaluation globale de la qualité d'un produit, en tenant compte des variations et des incertitudes dans les mesures individuelles.
- Contrôle de processus : La logique floue peut être utilisée dans le contrôle de processus pour détecter les variations et les anomalies dans les paramètres de production. En utilisant des ensembles flous pour représenter des règles de contrôle et des seuils de performance, les systèmes de contrôle de qualité basés sur la logique floue peuvent évaluer la stabilité d'un processus et détecter les écarts par rapport aux normes attendues, permettant ainsi une détection précoce des problèmes de qualité et des ajustements en temps réel.
- Contrôle de la conformité réglementaire : La logique floue est utilisée dans le contrôle de conformité réglementaire pour évaluer si un produit ou un processus respecte les normes et les réglementations en vigueur. En utilisant des ensembles flous pour représenter les exigences réglementaires et les mesures de conformité, les systèmes de contrôle de qualité basés sur la logique floue peuvent évaluer le degré de conformité d'un produit ou d'un processus, en tenant compte des variations et des incertitudes dans les mesures de conformité.
- Contrôle de la satisfaction client : La logique floue est utilisée dans le contrôle de la satisfaction client pour évaluer la satisfaction des clients en fonction de différentes mesures de performance. En utilisant des ensembles flous pour représenter des concepts tels que "très satisfait", "satisfait", "neutre" ou "insatisfait", les systèmes de contrôle de qualité basés sur la logique floue peuvent agréger les évaluations des clients pour donner une évaluation globale de la satisfaction, en tenant compte des variations et des incertitudes dans les évaluations individuelles.

2.3.6. Finance et investissement :

La logique floue est utilisée dans les modèles financiers pour évaluer et gérer le risque associé aux investissements [17]. En utilisant des ensembles flous pour représenter des facteurs tels que la volatilité, la liquidité et les performances passées, la logique floue permet d'effectuer des analyses plus nuancées et de prendre des décisions d'investissement plus éclairées.

Voici quelques exemples d'application de la logique floue dans le domaine du financement et de l'investissement :

- Évaluation du risque : La logique floue peut être utilisée pour évaluer le risque financier associé à un investissement. En utilisant des ensembles flous pour représenter des variables telles que la volatilité des prix, la stabilité financière et la liquidité, les systèmes d'évaluation du risque basés sur la logique floue peuvent générer des évaluations du risque qui prennent en compte les incertitudes et les variations inhérentes aux marchés financiers.
- Allocation d'actifs : La logique floue est utilisée dans l'allocation d'actifs pour décider de la répartition optimale des investissements entre différentes classes d'actifs. En utilisant des ensembles flous pour représenter des variables telles que le rendement attendu, le niveau de risque et les préférences des investisseurs, les systèmes d'allocation d'actifs basés sur la logique floue peuvent générer des recommandations d'allocation qui tiennent compte des objectifs individuels et des conditions du marché.
- Prise de décision d'investissement : La logique floue peut être utilisée dans la prise de décision d'investissement pour évaluer la rentabilité potentielle d'une opportunité d'investissement. En utilisant des ensembles flous pour représenter des variables telles que la rentabilité attendue, la liquidité et les contraintes de temps, les systèmes de prise de décision d'investissement basés sur la logique floue peuvent générer des évaluations de rentabilité qui prennent en compte les incertitudes et les variations inhérentes aux marchés financiers.
- Modélisation des préférences des investisseurs : La logique floue peut être utilisée pour modéliser les préférences des investisseurs et prendre des décisions d'investissement personnalisées. En utilisant des ensembles flous pour représenter des variables telles que l'aversion au risque, l'horizon d'investissement et les objectifs financiers, les systèmes de modélisation des préférences des investisseurs basés sur la logique floue peuvent générer des recommandations d'investissement adaptées à chaque investisseur en tenant compte des variations et des incertitudes individuelles.
- Évaluation des performances des fonds d'investissement : La logique floue peut être utilisée pour évaluer les performances des fonds d'investissement en tenant compte de différents critères de performance tels que le rendement, la volatilité et la cohérence des résultats. En utilisant des ensembles flous pour représenter des variables de performance, les systèmes d'évaluation des performances basés sur la logique floue peuvent générer des évaluations plus nuancées et précises des fonds d'investissement, en tenant compte des variations et des incertitudes dans les mesures de performance.

Conclusion :

En conclusion, la logique floue est un outil puissant pour traiter l'incertitude, la variabilité et l'imprécision présentes dans de nombreux domaines d'application. Sa capacité à représenter et à raisonner sur des concepts linguistiques vagues permet d'obtenir des résultats plus flexibles et adaptatifs par rapport aux méthodes traditionnelles basées sur la logique binaire.

Ce chapitre a exploré différentes applications de la logique floue dans divers domaines tels que les systèmes de recommandation, la robotique, le traitement de langage naturel, les prévisions météorologiques, le contrôle de qualité et le financement/ investissement. Dans chaque domaine, la logique floue offre des avantages en termes de prise de décision, de modélisation de l'incertitude et de prise en compte des variations.

L'application de la logique floue permet de prendre des décisions plus nuancées, de gérer l'incertitude et de s'adapter aux changements des conditions environnementales. Elle facilite

également l'interaction entre les systèmes automatisés et les utilisateurs humains en utilisant des concepts et des règles linguistiques familiers.

Cependant, la logique floue n'est pas une solution universelle et peut avoir ses propres limites. Elle nécessite une modélisation appropriée des ensembles flous, des règles d'inférence pertinentes et des méthodes d'apprentissage adéquates pour obtenir des résultats précis et fiables.

Dans l'ensemble, la logique floue offre un cadre flexible et puissant pour modéliser et raisonner sur des concepts vagues, contribuant ainsi à améliorer la prise de décision et les performances des systèmes dans de nombreux domaines d'application. Son utilisation continue de la logique floue et des techniques associées ouvre de nouvelles perspectives pour traiter l'incertitude et les situations complexes, favorisant ainsi le développement de systèmes plus intelligents et adaptatifs

Chapitre 3 :

Introduction à Python

Introduction :

L'art de la programmation consiste à écrire un code informatique qui peut être exécuté par une machine. Il s'agit de créer des instructions dans un langage de programmation qui décrivent les activités que l'ordinateur doit effectuer à son tour. Les programmes informatiques peuvent être utilisés à diverses fins, notamment l'analyse de données, la création de sites web, la résolution de problèmes mathématiques complexes, etc. Il existe une grande variété de langages de programmation, chacun présentant des avantages et des inconvénients uniques. Python, Java, C++, JavaScript, PHP et Ruby sont quelques-uns des langages de programmation les plus utilisés. Dans ce chapitre on va définir l'environnement de programmation Python et comment l'installer. On va également explorer les principes fondamentaux de Python, y compris les structures de contrôle, les fonctions, les classes et les modules. Aussi comment utiliser les bibliothèques et les frameworks populaires pour créer des applications plus avancées.

3.1. Définition :

Python est un langage de programmation de haut niveau interprété pour la programmation à usage général. Créé par Guido van Rossum, et publié pour la première fois en 1991. Python repose sur une philosophie de conception qui met l'accent sur la lisibilité du code, notamment en utilisant des espaces significatifs. Il fournit des constructions permettant une programmation claire à petite et grande échelle. Python propose un système de typage dynamique et une gestion automatique de la mémoire. Il prend en charge plusieurs paradigmes de programmation, notamment orienté objet, impératif, fonctionnel et procédural, et dispose d'une bibliothèque standard étendue et complète. Python est un langage de programmation open-source, développé pour une utilisation avec une large gamme de systèmes d'exploitation. Il est qualifié de langage de programmation le plus puissant en raison de sa nature dynamique et diversifiée. Il est facile à utiliser avec une syntaxe super simple très encourageante pour les apprenants débutants, et très motivante pour les utilisateurs chevronnés [18].

3.2. Installation de Python :

Pour commencer à programmer en Python, on doit installer l'interpréteur Python sur l'ordinateur. Voici les étapes à suivre pour installer Python :

1. Visite du site officiel de Python : https://www.python.org. On s'assure de télécharger la dernière version stable de Python, qui est recommandée pour la plupart des utilisateurs.

2. On choisit la version de Python adaptée au système d'exploitation. Python est compatible avec Windows, macOS et Linux. Pour chaque système d'exploitation, on trouve des liens de téléchargement spécifiques.

3. On télécharge l'installateur Python correspondant au système d'exploitation utilisé. L'installateur est généralement un fichier exécutable (.exe pour Windows, .pkg pour macOS, .tar.gz pour Linux).

4. Ensuite on éxecute l'installateur Python en double-cliquant sur le fichier téléchargé. Après il ne reste plus qu'à suivre les instructions de l'installateur pour procéder à l'installation.

Une fois l'installation terminée, on vérifie si Python est correctement installé en ouvrant l'invite de commandes (ou le terminal) et en exécutant la commande suivante :

```
python --version
```

Cela affichera la version de Python installée sur le système.

En plus de l'interpréteur Python, on peut également choisir d'utiliser un environnement de développement intégré (IDE) pour écrire et exécuter votre code Python. Certains IDE populaires pour Python incluent PyCharm, Visual Studio Code, Sublime Text et Atom. Ces IDE offrent des fonctionnalités supplémentaires telles que la coloration syntaxique, le débogage et la complétion automatique du code, ce qui facilite le développement. Pour ce projet, on a utilisé la platforme Google Colab.

3.2.1. Google Colab :

Google Colab, également connu sous le nom de Colaboratory, est une plateforme de développement et d'exécution de code Python basée sur le cloud. Il s'agit d'un service gratuit proposé par Google qui permet aux utilisateurs d'écrire, d'exécuter et de partager du code Python de manière collaborative, sans avoir à installer Python localement sur leur machine.

Google Colab est basé sur Jupyter Notebook, ce qui signifie qu'il offre une interface interactive où on peut écrire et exécuter du code Python par blocs appelés "cellules". On peut également ajouter des textes explicatifs, des images et des visualisations pour créer des rapports ou des tutoriels interactifs.

Google Colab offre plusieurs avantages pour les utilisateurs :

- **Gratuité :** Google Colab est entièrement gratuit à utiliser. On peut accéder à toutes les fonctionnalités sans frais, ce qui en fait une option attrayante, en particulier pour les étudiants, les chercheurs et les développeurs qui ont un budget limité.

- **Environnement basé sur le cloud :** On a pas besoin d'installer Python ou d'autres bibliothèques sur l'ordinateur. Colab fonctionne entièrement dans le cloud, ce qui signifie qu'on peut y accéder à partir de n'importe quel navigateur et de n'importe quel appareil (ordinateur portable, tablette, smartphone).

- **Puissance de calcul :** Google Colab utilise des machines virtuelles puissantes pour exécuter le code. On peut bénéficier de GPU (Graphics Processing Units) et de TPU (Tensor Processing Units) pour accélérer les calculs intensifs, tels que l'apprentissage automatique et l'analyse de données. Cela permet d'exécuter des tâches complexes plus rapidement que sur une machine locale ordinaire.

- **Pré-installation des bibliothèques courantes :** Colab est préconfiguré avec de nombreuses bibliothèques Python populaires, telles que NumPy, pandas, Matplotlib, TensorFlow, et bien d'autres. On peut commencer à utiliser ces bibliothèques immédiatement sans avoir à les installer.

- **Collaboration en temps réel :** Colab permet la collaboration en temps réel avec d'autres utilisateurs. On peut partager des notebooks avec des collègues ou des amis, et travailler ensemble sur le même projet. Les modifications apportées aux notebooks sont instantanément visibles par tous les participants, ce qui facilite la collaboration à distance.

- **Stockage et sauvegarde automatiques :** Les notebooks Colab sont automatiquement enregistrés dans le compte Google Drive. Cela garantit que les travaux sont sauvegardés et accessibles à partir de n'importe quel appareil. De plus, on peut facilement revenir à des versions précédentes du notebook si nécessaire.

- **Intégration avec d'autres services Google :** Google Colab s'intègre étroitement avec d'autres services Google tels que Google Drive, GitHub et BigQuery. On peut importer et exporter facilement des données, des notebooks et des bibliothèques à partir de ces services, ce qui facilite le partage et la collaboration.

Google Colab offre une expérience de développement Python pratique, flexible et gratuite. Avec ses fonctionnalités avancées, sa puissance de calcul et sa collaboration en temps réel, il est devenu un choix populaire pour les projets d'apprentissage automatique, l'analyse de données et la programmation en Python en général.

3.3. Les bases de la syntaxe :

La syntaxe de Python est connue pour sa clarté et sa lisibilité, ce qui facilite l'apprentissage et la compréhension du langage [19].

3.3.1. Commentaires :

Les commentaires sont des annotations utilisées pour expliquer le code et sont ignorés lors de l'exécution. En Python, les commentaires commencent par le symbole dièse (#). Exemple :

```
# Ceci est un commentaire
```

3.3.2. Variables :

Les variables sont des conteneurs utilisés pour stocker des données. En Python, On peut déclarer une variable en lui attribuant une valeur. Le type de données d'une variable est déterminé automatiquement en fonction de la valeur qui lui est assignée. Exemple :

```
nom = "Alice"
age = 25
pi = 3.14159
est_vrai = True
```

3.3.3. Types de données :

Python prend en charge plusieurs types de données courants, tels que les entiers (int), les nombres à virgule flottante (float), les chaînes de caractères (str) et les booléens (bool). Exemple :

```
nombre_entier = 42
nombre_float = 3.14
texte = "Bonjour, monde!"
est_vrai = True
```

3.3.4. Opérations arithmétiques :

Python offre un ensemble d'opérations arithmétiques courantes, telles que l'addition (+), la soustraction (-), la multiplication (*), la division (/), le modulo (%) et l'exponentiation (**). Exemple :

```
a = 10
b = 3
addition = a + b
soustraction = a - b
multiplication = a * b
division = a / b
modulo = a % b
exponentiation = a ** b
```

3.3.5. Structure de contrôle :

Python propose des structures de contrôle telles que les instructions conditionnelles (if-else) et les boucles (for et while) pour contrôler le flux d'exécution du programme.

Exemple d'instruction conditionnelle :

```
x = 5
if x > 5:
    print("x est supérieur à 5")
elif x == 5:
    print("x est égal à 5")
else:
    print("x est inférieur à 5")
```

Exemple de boucle for :

```
for i in range(1, 6):
    print(i)
```

Exemple de boucle while :

```
x = 1
while x <= 5:
    print(x)
    x += 1
```

3.3.6. Les fonctions :

Les fonctions permettent de regrouper des morceaux de code réutilisables. En Python, on peut définir des fonctions propres à l'aide du mot-clé "def".

```
def saluer(nom):
    print("Bonjour,", nom)

saluer("Alice")
```

3.4. Les bibliothèques de Python :

En Python, les bibliothèques (ou modules) sont des ensembles de fonctions et de classes pré écrites qui étendent les fonctionnalités de base du langage. Elles permettent aux développeurs d'accéder à des fonctionnalités spécifiques sans avoir à les implémenter eux-mêmes. Les bibliothèques sont un élément essentiel de l'écosystème Python, offrant une vaste gamme de fonctionnalités prêtes à l'emploi pour divers domaines tels que la science des données, le développement web, l'apprentissage automatique, les graphiques, etc. [20]

Voici quelques bibliothèques populaires de Python :

3.4.1. NumPy :

NumPy (Numerical Python) est une bibliothèque fondamentale en Python qui fournit des structures de données et des fonctionnalités pour effectuer des calculs numériques efficaces. Elle est largement utilisée dans le domaine de la science des données, du calcul scientifique et de l'apprentissage automatique. Voici quelques-unes des fonctionnalités clés de NumPy :

Tableaux NumPy : NumPy introduit un nouveau type de données appelé ndarray (n-dimensional array), qui est un tableau multidimensionnel homogène. Les tableaux NumPy sont plus efficaces que les listes Python standard pour stocker et manipuler de grandes quantités de données numériques. Les tableaux peuvent être unidimensionnels (vecteurs), bidimensionnels (matrices) ou de dimensions supérieures.

Opérations mathématiques : NumPy offre un large éventail d'opérations mathématiques sur les tableaux, telles que l'addition, la soustraction, la multiplication, la division, les opérations matricielles, les fonctions trigonométriques, les logarithmes, etc. Ces opérations sont optimisées pour des performances élevées et permettent d'effectuer des calculs efficaces sur de grands ensembles de données.

Indexation et tranchage : NumPy permet d'accéder facilement aux éléments d'un tableau en utilisant l'indexation et le tranchage. Vous pouvez extraire des éléments spécifiques, des sous-tableaux ou des colonnes entières en spécifiant des indices ou des plages d'indices.

Fonctions statistiques : NumPy fournit de nombreuses fonctions statistiques pour effectuer des calculs sur les tableaux, tels que la moyenne, la variance, l'écart-type, le minimum, le maximum, la médiane, etc. Ces fonctions simplifient l'analyse statistique des données et permettent d'obtenir rapidement des informations utiles.

Broadcasting : Broadcasting est une fonctionnalité puissante de NumPy qui permet d'effectuer des opérations sur des tableaux de formes différentes sans avoir à les étendre explicitement. Cela facilite l'écriture de code vectorisé, évitant ainsi les boucles et les opérations élément par élément.

NumPy offre de nombreuses autres fonctionnalités, y compris des outils pour la manipulation d'images, le filtrage de données, les opérations logiques, la génération de nombres aléatoires,

etc. Son utilisation combinée avec d'autres bibliothèques, telles que Pandas et Matplotlib, permet d'effectuer des analyses de données complètes et des visualisations graphiques.

3.4.2. Pandas :

Pandas est une bibliothèque Python populaire utilisée pour la manipulation et l'analyse de données. Elle offre des structures de données flexibles et performantes, ainsi que des fonctionnalités avancées pour le nettoyage, la transformation, l'exploration et l'agrégation de données. Voici quelques-unes des fonctionnalités clés de Pandas :

DataFrames : Le DataFrame est la structure de données centrale de Pandas. Il s'agit d'un tableau bidimensionnel contenant des colonnes étiquetées, similaires à une feuille de calcul ou à une table SQL. Les DataFrames permettent de manipuler facilement des données tabulaires, d'effectuer des opérations sur les colonnes, de filtrer les lignes, de joindre des tables, etc

Importation et exportation de données : Pandas offre des fonctions pour importer et exporter des données depuis et vers différents formats tels que CSV, Excel, SQL, JSON, etc. Cela permet de charger facilement des données externes dans un DataFrame et d'enregistrer les résultats de l'analyse.

Manipulation des données : Pandas offre une large gamme de fonctionnalités pour manipuler les données dans un DataFrame. Vous pouvez effectuer des opérations de filtrage, de tri, de groupement, de fusion, de pivotement, de renommage de colonnes, de calcul de statistiques descriptives, etc. Ces opérations permettent de nettoyer les données, de créer de nouvelles variables et de préparer les données pour l'analyse.

Gestion des valeurs manquantes : Les données réelles contiennent souvent des valeurs manquantes ou des données erronées. Pandas offre des outils pour gérer ces valeurs manquantes, telles que le repérage, le filtrage et le remplacement des valeurs manquantes. Cela permet d'effectuer des analyses sur des ensembles de données incomplets.

Visualisation des données : Pandas s'intègre étroitement avec la bibliothèque de visualisation Matplotlib, permettant de créer facilement des graphiques et des visualisations à partir des données du DataFrame. Cela facilite l'exploration et la communication des résultats de l'analyse de données.

Pandas offre de nombreuses autres fonctionnalités, telles que la fusion de DataFrames, les opérations temporelles, la gestion des données catégorielles, l'échantillonnage, la création de fonctions personnalisées, etc. Son utilisation conjointe avec NumPy et d'autres bibliothèques d'analyse de données permet de réaliser des tâches complexes de manière efficace et productive.

3.4.3. Matplotlib :

Matplotlib est une bibliothèque de visualisation de données en 2D largement utilisée en Python. Elle permet de créer une grande variété de graphiques, de diagrammes et de visualisations pour explorer et présenter les données de manière claire et informative. Voici quelques-unes des fonctionnalités clés de Matplotlib.

Graphiques de lignes : Matplotlib permet de tracer des graphiques de lignes pour représenter des séries de données continues. Ces graphiques sont utiles pour visualiser des tendances, des variations et des relations entre les variables.

Diagrammes à barres : Matplotlib permet de créer des diagrammes à barres pour représenter des données catégorielles ou des comparaisons entre différentes catégories. Ces diagrammes sont souvent utilisés pour visualiser des fréquences, des effectifs ou des mesures comparatives.

Diagrammes circulaires : Matplotlib permet de créer des diagrammes circulaires, également appelés diagrammes en secteurs ou diagrammes en camembert, pour représenter la répartition des catégories dans un ensemble de données. Ces diagrammes permettent de visualiser les proportions et les parts relatives.

Diagrammes à barres empilées : Matplotlib permet de créer des diagrammes à barres empilées pour représenter des comparaisons de plusieurs variables ou catégories avec des barres superposées. Ces diagrammes permettent de visualiser les contributions relatives de chaque variable.

Diagrammes de dispersion : Matplotlib permet de tracer des diagrammes de dispersion pour visualiser les relations entre deux variables continues. Ces diagrammes sont utiles pour détecter des modèles, des tendances ou des corrélations entre les variables.

Matplotlib offre de nombreuses autres fonctionnalités pour personnaliser les graphiques, tels que la modification des couleurs, des styles, des légendes, des axes, des grilles, etc. On peut également créer des sous-graphiques, des graphiques en 3D, des cartes, des diagrammes en boîte, des histogrammes et bien plus encore. Matplotlib est une bibliothèque très flexible qui permet de créer des visualisations personnalisées et professionnelles pour répondre aux besoins spécifiques.

3.4.4. TensorFlow :

TensorFlow est une bibliothèque open-source d'apprentissage automatique (machine learning) et de calcul numérique développée par Google. Elle est largement utilisée pour créer, former et déployer des modèles d'apprentissage automatique, en particulier dans le domaine du deep learning. Voici quelques-unes des fonctionnalités clés de TensorFlow :

Tenseurs et opérations : TensorFlow utilise des tenseurs comme principales structures de données. Un tenseur est une généralisation des vecteurs et des matrices à des dimensions

supérieures. TensorFlow fournit des opérations mathématiques puissantes pour manipuler ces tenseurs, telles que l'addition, la multiplication, le calcul de gradients, etc.

Construction de graphiques computationnels : TensorFlow utilise un modèle de programmation par graphe computationnel. Vous construisez un graphe computationnel qui représente les opérations que vous souhaitez effectuer, puis vous exécutez ce graphe dans une session TensorFlow. Cela permet d'optimiser l'exécution des opérations et de les exécuter sur différentes plates-formes, telles que CPU, GPU ou TPU.

Apprentissage automatique avec TensorFlow : TensorFlow fournit des outils puissants pour créer, former et évaluer des modèles d'apprentissage automatique. Vous pouvez utiliser les API de haut niveau, telles que Keras, qui simplifient la création de modèles d'apprentissage profond. TensorFlow prend également en charge l'optimisation, les fonctions de coût, les algorithmes de descente de gradient, et bien plus encore.

Déploiement de modèles : TensorFlow permet de sauvegarder et de charger des modèles entraînés pour une utilisation ultérieure. Vous pouvez exporter des modèles dans différents formats, tels que SavedModel ou TensorFlow Lite, afin de les déployer sur différentes plates-formes, y compris des appareils mobiles et des systèmes embarqués.

Accélération matérielle : TensorFlow tire parti de l'accélération matérielle, en utilisant des bibliothèques optimisées telles que CUDA pour l'exécution sur des GPU. Cela permet d'accélérer considérablement les calculs, en particulier pour les modèles de deep learning qui impliquent des calculs intensifs sur des matrices.

TensorFlow est une bibliothèque polyvalente et puissante pour l'apprentissage automatique et le calcul numérique. Pour créer des réseaux de neurones profonds, effectuer des calculs numériques complexes ou déployer des modèles dans des applications, TensorFlow offre une gamme complète d'outils et de fonctionnalités.

3.4.5. scikit-fuzzy :

Scikit-fuzzy est une bibliothèque open-source en Python dédiée au traitement des systèmes de logique floue (fuzzy logic). Elle fournit des outils pour la modélisation et la simulation de systèmes flous, ainsi que des algorithmes pour l'inférence floue, la défuzzification et d'autres opérations liées à la logique floue. Voici les principales fonctionnalités de scikit-fuzzy :

Variables linguistiques : Scikit-fuzzy permet de définir et de gérer des variables linguistiques, qui sont des variables qui utilisent des termes linguistiques plutôt que des valeurs numériques. Vous pouvez définir des ensembles flous et des fonctions d'appartenance pour représenter les différentes catégories ou concepts liés à votre système flou.

Opérations floues : Scikit-fuzzy propose des opérations floues telles que l'intersection, l'union, la complémentation, la fusion, etc. Ces opérations permettent de combiner les ensembles flous et de manipuler les fonctions d'appartenance pour effectuer des calculs sur les variables linguistiques.

Inférence floue : Scikit-fuzzy fournit des outils pour réaliser des inférences floues à l'aide de règles if-then. Vous pouvez définir des règles basées sur les ensembles flous et les opérations floues, et utiliser ces règles pour inférer de nouvelles connaissances à partir des variables linguistiques. L'inférence floue peut être utilisée pour prendre des décisions, contrôler des systèmes, modéliser des situations complexes, etc.

Défuzzification : Scikit-fuzzy offre des méthodes pour la défuzzification, qui consiste à convertir les résultats flous en valeurs numériques précises. Différentes méthodes de défuzzification sont disponibles, telles que le centre de gravité, le maximum, le sommet, etc. Ces méthodes permettent de transformer les résultats flous en valeurs compréhensibles et exploitables.

Scikit-fuzzy est utilisé dans divers domaines où la logique floue est appliquée, tels que le contrôle de processus, la robotique, les systèmes de recommandation, l'intelligence artificielle, etc. Il offre une interface simple et intuitive pour modéliser et simuler des systèmes flous, et permet d'explorer les concepts et les principes de la logique floue de manière efficace.

3.5. Les frameworks :

Il existe plusieurs frameworks populaires en Python pour créer des applications web et des applications de bureau [21]. Voici une brève explication de certains des frameworks les plus utilisés :

Django : Django est un framework web puissant et complet, largement utilisé pour le développement d'applications web robustes et évolutives. Il suit le principe du "batteries included", offrant un ensemble complet de fonctionnalités pour gérer l'authentification, les bases de données, la gestion des formulaires, la génération de pages dynamiques, la gestion des URL, etc. Django suit l'architecture Modèle-Vue-Contrôleur (MVC) et facilite le développement rapide d'applications web.

Flask : Flask est un framework web léger et flexible qui se concentre sur la simplicité et la modularité. Il offre une grande liberté aux développeurs pour choisir les outils et les bibliothèques dont ils ont besoin, ce qui en fait un choix populaire pour les petites applications et les API REST. Flask est facile à apprendre et à utiliser, tout en offrant des fonctionnalités telles que le routage, la gestion des cookies, les sessions, les templates Jinja2, etc.

Pyramid : Pyramid est un framework web minimaliste, mais extrêmement flexible et puissant. Il suit le principe "ne vous imposez rien", ce qui signifie qu'il ne présume pas de l'architecture ou des outils spécifiques que vous devez utiliser. Cela donne aux développeurs une grande liberté pour créer des applications web personnalisées selon leurs besoins spécifiques. Pyramid offre une architecture modulaire, une gestion des vues basée sur les classes, une intégration simple avec les bases de données, etc.

Tkinter : Tkinter est un framework pour le développement d'interfaces graphiques (GUI) en Python. Il est livré avec Python par défaut, ce qui signifie qu'il n'est pas nécessaire d'installer des bibliothèques supplémentaires. Tkinter permet de créer des fenêtres, des boutons, des formulaires, des boîtes de dialogue et d'autres éléments d'interface utilisateur pour les applications de bureau. Bien qu'il soit considéré comme relativement simple par rapport à d'autres frameworks GUI, Tkinter est largement utilisé pour créer des applications de bureau multiplateformes.

PyQT : PyQT est une liaison Python pour le framework de développement d'interfaces graphiques Qt. Qt est un framework populaire et puissant pour le développement d'applications de bureau. PyQT offre une grande variété de fonctionnalités, y compris la création d'interfaces utilisateur avancées, la gestion des événements, les fonctionnalités multimédias, l'accès aux bases de données, etc. PyQT est apprécié pour sa flexibilité et sa convivialité, et il est couramment utilisé pour développer des applications de bureau riches en fonctionnalités.

Ces frameworks offrent des fonctionnalités variées et s'adressent à différents besoins de développement d'applications.

Conclusion :

En conclusion, Python a permis d'explorer divers aspects de la programmation avec ce langage polyvalent. Dans ce chapitre on a commencé par une introduction à la programmation Python, mettant en évidence sa popularité et sa facilité d'apprentissage. Ensuite, on a abordé l'installation de Python, en détaillant les différentes méthodes disponibles pour configurer l'environnement de développement. On a ensuite exploré les bases de la syntaxe de Python, couvrant les types de données, les opérateurs, les structures de contrôle et les fonctions. Cette partie a fourni une base solide pour comprendre et écrire des programmes Python.

Ensuite on a expliqué l'utilisation des bibliothèques en Python, en mettant l'accent sur des bibliothèques populaires telles que NumPy, Pandas, Matplotlib , TensorFlow et Scikit fuzzy. Ces bibliothèques offrent des fonctionnalités avancées pour le traitement des données, la visualisation, l'apprentissage automatique et bien plus encore. Leur utilisation permet d'étendre les capacités de Python et de faciliter le développement d'applications complexes.

Enfin, on a également abordé les frameworks populaires en Python pour la création d'applications web et de bureau, tels que Django, Flask et Tkinter. Ces frameworks offrent des outils et des fonctionnalités permettant de développer des applications puissantes et personnalisées, répondant ainsi à divers besoins de développement.

En somme, Python est un langage polyvalent et populaire, largement utilisé dans le domaine de la programmation. Sa syntaxe simple et expressive, sa vaste collection de bibliothèques et de frameworks, ainsi que sa communauté active en font un choix privilégié pour une variété de projets.

Chapitre 4 :

Simulations et résultats

Introduction :

Différentes solutions ont été développées pour aider les non-voyants à éviter les obstacles lors de leurs déplacements. Parmi les solutions les plus courantes, on trouve l'utilisation de cannes blanches, l'entraînement des chiens guides, ainsi que des technologies plus avancées telles que les systèmes d'assistance à la mobilité basés sur des capteurs et des signaux sonores. Cependant, ces solutions présentent certaines limitations, notamment en termes de précision, de réactivité et de capacité à traiter des situations complexes. L'objectif de cette simulation est d'aller au-delà des limitations des solutions existantes en développant un système d'évitement d'obstacles pour les non-voyants basé sur la logique floue, avec un guidage audio, en utilisant la plateforme Google Colab. Ce système vise à améliorer la précision et la capacité à traiter des situations complexes lors de la navigation des non-voyants.

Dans ce chapitre, on présentera les capteurs qui peuvent être utilisés dans un tel système, tels que les capteurs d'angle, les capteurs sonores ou les capteurs de proximité. On va également expliquer en détail le principe de fonctionnement du modèle de logique floue pour l'évitement d'obstacles. Ensuite, on va procéder à la création d'un modèle de logique floue qui peut prendre en compte les informations sensorielles disponibles pour un non-voyant, telles que les signaux sonores captés par les capteurs. Ce modèle utilisera des variables d'entrée floues, telles que la distance et l'angle par rapport aux obstacles, et générera des commandes appropriées pour éviter les obstacles et se déplacer en toute sécurité. Une fois le modèle de logique floue mis en place, on testera le système d'évitement en variant les paramètres d'entrées des obstacles. Cela permettra d'évaluer les performances du système dans différents scénarios et de déterminer son efficacité dans la prise de décisions d'évitement d'obstacles.

4.1. Les capteurs utilisés dans un système d'évitements d'obstacle basé sur la logique floue :

4.1.1. Les capteurs d'angles :

Les capteurs d'angle sont des dispositifs utilisés pour mesurer l'angle de rotation ou la position angulaire d'un objet par rapport à une référence donnée. Ils sont largement utilisés dans de nombreuses applications, y compris les systèmes d'évitement d'obstacles pour les non-voyants basés sur la logique floue.

Voici quelques exemples de capteurs d'angle couramment utilisés :

a) Potentiomètre :

Il s'agit d'un capteur analogique qui mesure l'angle de rotation en fonction de la variation de la résistance électrique. L'angle de rotation peut être déterminé en mesurant la tension de sortie du potentiomètre.

Fig 9. Capteur Potentiomètre

b) Codeur rotatif :

Il s'agit d'un capteur numérique qui convertit l'angle de rotation en une série de signaux numériques. Les codeurs rotatifs peuvent être absolus, fournissant une valeur précise de l'angle, ou incrémentaux, fournissant des informations sur le mouvement relatif.

Fig 10. Codeur rotatif (absolu et incrémental)

c) Gyroscope :

Un gyroscope mesure l'orientation angulaire d'un objet en détectant les changements de force exercés sur l'objet lorsqu'il tourne. Les gyroscopes peuvent être utilisés pour mesurer des angles de rotation dans les trois dimensions de l'espace.

Fig 11. Capteur gyroscope

d) Accéléromètre :

Bien que principalement utilisé pour mesurer l'accélération linéaire, un accéléromètre peut également être utilisé pour estimer l'angle d'inclinaison d'un objet par rapport à la gravité. En utilisant les mesures de l'accélération dans différentes directions, il est possible de calculer l'angle d'inclinaison.

Fig 12. Accéléromètre

e) Capteur à effet Hall :

Ce type de capteur mesure la présence et l'intensité d'un champ magnétique et peut être utilisé pour déterminer l'angle de rotation en utilisant des aimants placés sur un objet en mouvement.

Fig 13. Capteur à effet de hall

Le choix du meilleur capteur d'angle pour un système d'évitement d'obstacles basé sur la dépend des exigences spécifiques de l'application et des contraintes techniques. Différents capteurs d'angle ont leurs avantages et leurs limites, et il est important de considérer plusieurs facteurs lors de la sélection du capteur approprié. Voici les éléments à prendre en compte :

- Précision : La précision de mesure de l'angle est un aspect crucial. Certains capteurs, tels que les codeurs rotatifs absolus, offrent une précision élevée et fournissent une mesure directe de l'angle sans nécessiter de recalibrage. Les gyroscopes peuvent également fournir une précision élevée, mais ils peuvent nécessiter une compensation des dérives.
- Plage de mesure : Il est essentiel de choisir un capteur dont la plage de mesure correspond aux besoins de l'application. Certains capteurs, comme les potentiomètres, peuvent avoir une plage limitée, tandis que d'autres, comme les gyroscopes, offrent une mesure continue sur 360 degrés.
- Réactivité : La vitesse de réponse du capteur est importante pour un système d'évitement d'obstacles en temps réel. Certains capteurs, comme les codeurs rotatifs incrémentaux,

peuvent fournir une réponse rapide, tandis que d'autres peuvent avoir une latence plus élevée.

- Environnement et conditions d'utilisation : Il est crucial de considérer les contraintes environnementales dans lesquelles le système sera utilisé. Certains capteurs peuvent être sensibles aux vibrations, à la poussière ou aux températures extrêmes. Il convient donc de choisir un capteur adapté à l'environnement spécifique.
- Coût : Le coût du capteur est également un facteur à prendre en compte, en fonction du budget alloué au projet.

4.1.2. Les capteurs de distance :

Les capteurs de distance sont des dispositifs utilisés pour mesurer la distance entre un objet et un capteur. Ils sont largement utilisés dans les systèmes d'évitement d'obstacles pour les non-voyants afin de détecter la présence d'obstacles et d'assister la navigation en fournissant des informations sur la distance par rapport à ces obstacles. Voici des exemples de capteurs de distance couramment utilisés :

a) Capteurs à ultrasons :

Ces capteurs émettent des ondes ultrasonores et mesurent le temps qu'il faut pour que les ondes se reflètent sur un objet et reviennent au capteur. En utilisant la vitesse du son, la distance peut être calculée avec précision.

Fig 14. Capteurs à ultrasons

b) Capteurs infrarouges :

Ces capteurs utilisent des émetteurs et des récepteurs infrarouges pour mesurer la distance en détectant le temps de retour du signal infrarouge réfléchi par un objet.

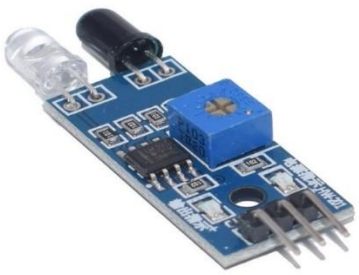

Fig 15. Capteur infrarouge

c) Capteurs de vision :

Certains capteurs de vision, tels que les caméras stéréoscopiques ou les capteurs de profondeur basés sur la technologie TOF (Time of Flight), peuvent être utilisés pour estimer la distance en mesurant les disparités entre les images capturées par différentes perspectives ou en calculant le temps de vol des impulsions lumineuses.

Fig 16. Capteur de vision

d) Capteurs laser :

Les capteurs laser, tels que les télémètres laser ou les lidars, émettent des faisceaux laser et mesurent le temps de retour du faisceau réfléchi pour déterminer la distance avec précision. Les lidars sont particulièrement utilisés pour la détection d'obstacles à longue distance et dans des environnements complexes.

Fig 17. Capteur laser (Lidar)

Le choix du capteur de distance dépend des besoins spécifiques du système, tels que la précision requise, la portée de mesure, la fréquence d'échantillonnage, la résolution, la taille et les contraintes de coût. Il est important de sélectionner le capteur approprié en tenant compte de ces facteurs et des caractéristiques de l'environnement dans lequel le système sera utilisé.

4.2. Principe de fonctionnement du système d'évitement d'obstacle :

La figure Suivante résume le fonctionnement du système d'évitement d'obstacle pour les non-voyants avec un audio feedback.

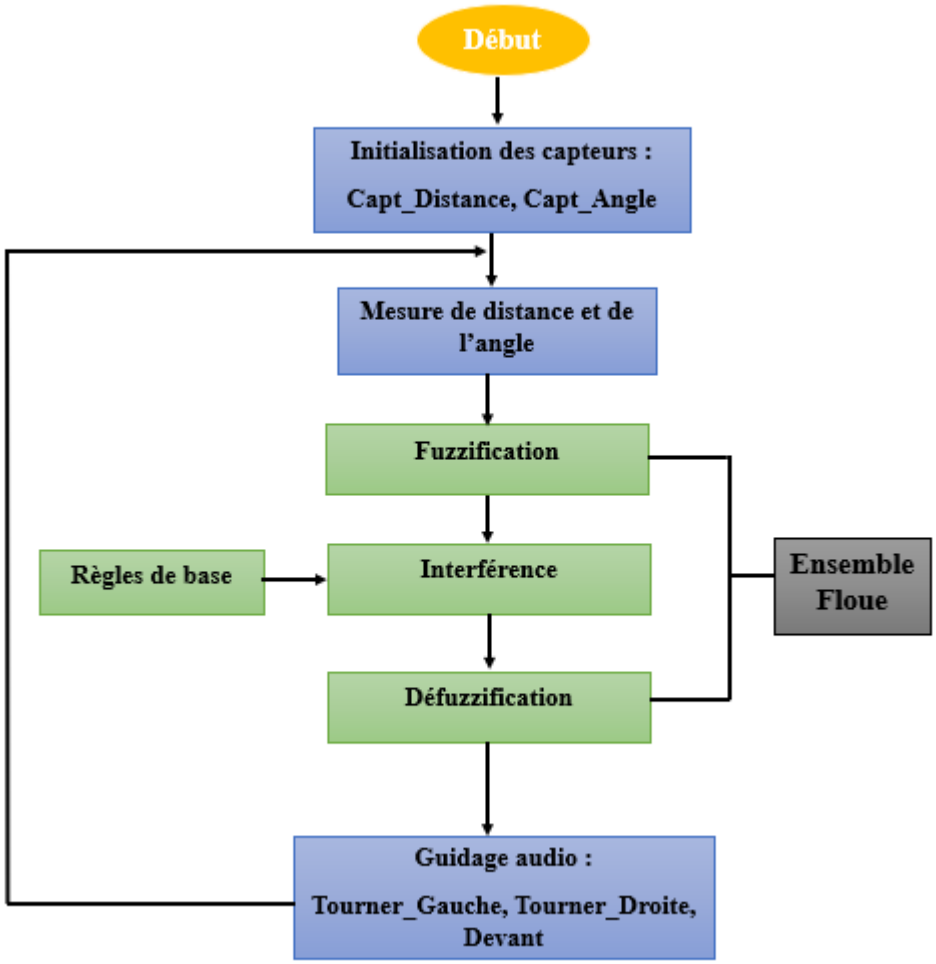

Fig 18. Organigramme du système d'évitement

Le principe de fonctionnement d'un système d'évitement d'obstacles pour les non-voyants en utilisant la logique floue avec audio feedback peut être décrit de la manière suivante :

- **Acquisition des données sensorielles** : Le système utilise des capteurs adaptés tels que des capteurs d'angles, des capteurs sonores, des capteurs de proximité ou des caméras pour recueillir des informations sur l'environnement environnant. Ces capteurs détectent la présence d'obstacles et fournissent des données sur leurs distances et leurs positions.

- **Prétraitement des données** : Les données collectées par les capteurs sont prétraitées pour éliminer les bruits indésirables et améliorer la qualité des informations. Cela peut inclure le filtrage du signal, la normalisation des valeurs et d'autres techniques de traitement du signal.

- **Modélisation floue des entrées :** Les données sensorielles prétraitées sont converties en variables floues à l'aide de la logique floue. Par exemple, la distance aux obstacles peut être représentée par des ensembles flous tels que "proche", "moyennement proche" et "loin". De même, la position (angle) des obstacles peut être représentée par des ensembles flous tels que "gauche", "devant" et "droite".

- **Définition des règles d'inférence :** Des règles d'inférence floue sont définies pour prendre des décisions en fonction des données sensorielles. Ces règles sont basées sur des relations entre les variables floues et déterminent les actions à prendre. Par exemple, une règle pourrait être "Si la distance est proche et l'angle est gauche, alors avertir l'utilisateur de la présence d'un obstacle à sa gauche".

- **Interférence floue :** À partir des variables floues et des règles d'interférence, le système utilise des opérations de logique floue pour interférer les actions à entreprendre. Cela peut inclure l'utilisation de techniques d'interférence floue telles que la méthode du centre de gravité ou le modèle de Mamdani.

- **Génération du feedback audio :** Les actions déduites par le système sont converties en instructions audio pour l'utilisateur non-voyant grâce à la bibliothèque gTTS (Google Text-To-Speech) sur Python (Google Colab). Par exemple, le système peut générer des messages tels que "Attention, un obstacle est détecté à gauche, tourner à droite" ou "Continuez tout droit". L'audio est transmis à l'utilisateur non-voyant pour l'informer de la présence et de la position des obstacles et l'orienter vers un chemin plus sécurisé. L'utilisateur peut ajuster sa trajectoire ou prendre les mesures nécessaires pour éviter les obstacles en fonction des instructions reçues.

- **Boucle de rétroaction :** Le système continue de collecter les données sensorielles, de mettre à jour les variables floues et d'ajuster les actions et le feedback audio en fonction des changements dans l'environnement. Cela permet au système de s'adapter en temps réel et de fournir un guidage précis et réactif à l'utilisateur non-voyant.

4.3. Création du modèle de simulation du système d'évitement d'obstacle :

Voici les étapes de création du modèle du système d'évitement d'obstacles pour les non-voyants basé sur la logique floue en utilisant l'environnement « Google Cola » :

4.3.1. Importation des bibliothèques :

- On commence par importer les bibliothèques nécessaires, telles que numpy, skfuzzy et matplotlib, en utilisant les commandes suivantes :

```
import numpy as np
import skfuzzy as fuzz
from skfuzzy import control as ctrl
import matplotlib.pyplot as plt
```

4.3.2. Définition des variables floues :

a) Distance :

La variable "distance" représente la mesure de la distance entre l'utilisateur non-voyant et un obstacle détecté. Elle est définie comme une variable d'entrée. Dans cette étape, on a défini les valeurs possibles de la distance et la granularité de cette variable. Par exemple, on a considéré les valeurs de distance allant de 0 à 6 mètres avec un pas de 1.

```
distance= ctrl.Antecedent(np.arange(0, 7, 1), 'distance')
```

b) Angle :

La variable "angle" représente l'orientation de l'obstacle par rapport à l'utilisateur non-voyant. Elle est également définie comme une variable d'entrée. On a défini les valeurs possibles de l'angle et la granularité de cette variable. Par exemple, on a considéré les valeurs d'angle allant de 0 degrés à 180 degrés avec un pas de 1 degré.

```
angle= ctrl.Antecedent(np.arange(0,190,1), 'angle')
```

c) Direction :

La variable "Direction" représente la commande de déplacement générée par le système d'évitement d'obstacles. Elle est définie comme une variable de sortie. On a défini les valeurs possibles de la commande qui dépendent des actions que l'utilisateur non-voyant peut effectuer pour éviter les obstacles telles que -1 pour tourner à droite, 0 pour aller tout droit et 1 pour tourner à gauche.

```
direction=ctrl.Consequent(np.arange(-1,2,1), 'direction')
```

Ces définitions des variables floues fournissent une base pour la modélisation du système d'évitement d'obstacles. Elles définissent les valeurs possibles et la résolution de chaque variable, ce qui permettra de créer des ensembles flous et des règles d'inférence appropriés pour le système.

4.3.3. Définition des variables flous :

a) Distance :

Pour définir les ensembles flous pour la variable "distance", on divise la plage de valeurs de distance en ensembles pertinents pour représenter différentes distances par rapport à l'obstacle tels que « Prés », « Loin », et « Moyen »

Chaque ensemble flou est défini en utilisant une fonction d'appartenance appropriée. Dans ce cas on a utilisé la fonction trimf (triangulaire) pour définir un ensemble flou « Prés » avec des valeurs allant de 0 à 2 mètres, un ensemble flou « Moyen » avec des valeurs allant de 2 à 4

mètres, et un ensemble flou « Loin » avec des valeurs allant de 4 à 6 mètres, comme le montre la figure 19.

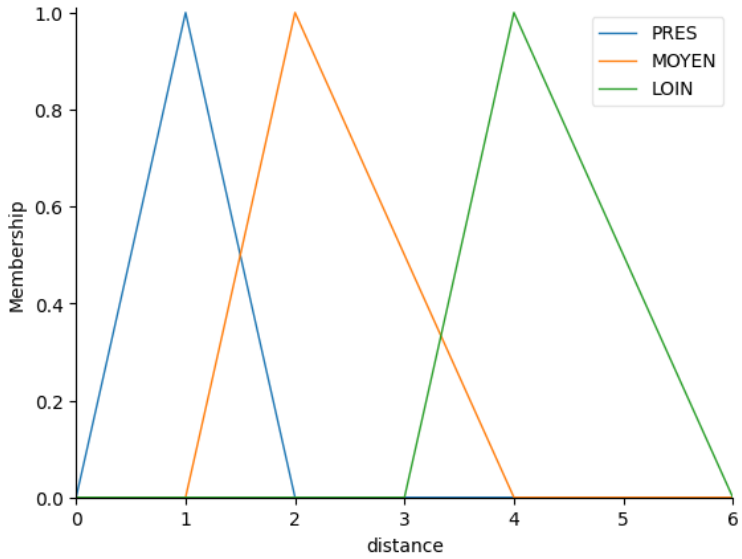

Fig 19. Ensemble flous de la distance

b) Angle :

Pour définir les ensembles flous pour la variable "Angle", on divise la plage de valeurs de « Angle » en ensembles pertinents pour représenter différentes orientations de l'obstacle tels que « Gauche », « Droite », et « Devant »

De même on a utilisé la fonction trimf (triangulaire) pour définir un ensemble flou « Droite » avec des valeurs allant de 0 à 60 degrés, un ensemble flou « Devant » avec des valeurs allant de 60 à 110 degrés, et un ensemble flou « Gauche » avec des valeurs allant de 110 à 180 degrés. La figure 20 illustre l'ensemble flou de l'angle.

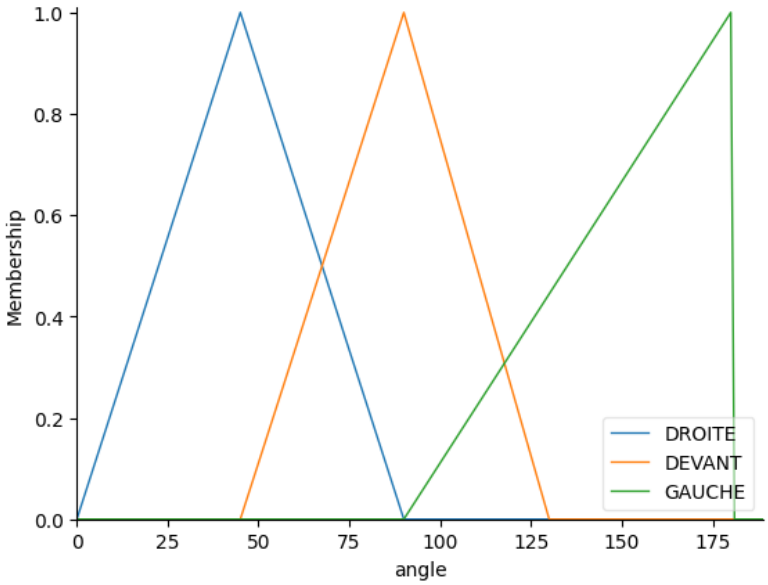

Fig 20. Ensemble flou de l'Angle.

c) Direction :

Pour définir les ensembles flous pour la variable "direction", on a représenté les actions que l'utilisateur non-voyant peut effectuer pour éviter les obstacles tels que "Tournez à gauche" avec les valeurs 1, "Tournez à droite" pour les valeurs de -1 et "Tout droit" pour les valeurs 0. La figure 21 représente les ensembles flous de la direction.

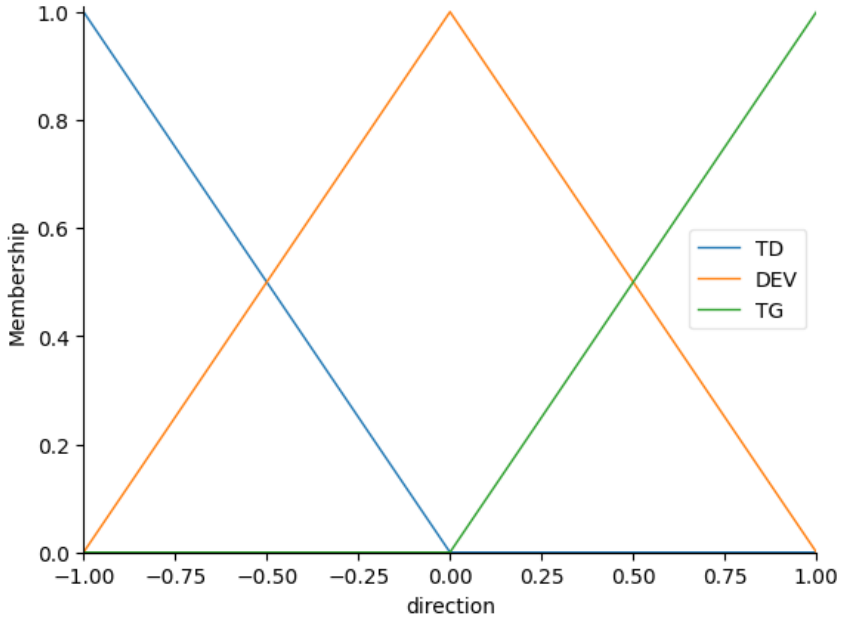

Fig 21. Ensemble flou de la direction

4.3.4. Définition des règles d'interférences :

La définition des règles d'inférence consiste à spécifier comment les ensembles flous d'entrée sont combinés pour obtenir les ensembles flous de sortie dans un système d'évitement d'obstacles basé sur la logique floue avec audio feedback en utilisant la distance et l'angle comme entrée. La définition des règles d'interférences se résume comme suit :

- Pour chaque règle, on identifie les conditions basées sur les ensembles flous d'entrée qui doivent être satisfaites pour activer la règle.
- On identifie également les conclusions basées sur les ensembles flous de sortie qui seront générées lorsque la règle est activée.
- Ensuite on utilise le langage de la logique floue pour exprimer les règles d'inférence. Chaque règle peut être exprimée sous la forme « SI (condition) ALORS (conclusion) ». Par exemple, une règle peut être formulée comme suit : "SI la distance est prés ET l'angle est à gauche ALORS la direction est « Tournez à droite »
- Pour chaque règle, on combine les ensembles flous d'entrée correspondants en utilisant l'opérateur "minimum". L'opérateur "minimum" est utilisé pour capturer l'activation partielle de chaque règle en fonction de la satisfaction de la condition. Par exemple, si on a la règle "SI la distance est prés ET l'angle est à gauche", on combine les

ensembles flous "distance_prés" et "angle_gauche" en utilisant l'opérateur "minimum" :

```
Combine_entré = np.fmin(distance_proche, angle_gauche)
```

- Egalement pour chaque règle on combine les ensembles flous de sortie correspondants en utilisant l'opérateur "maximum". L'opérateur "maximum" est utilisé pour regrouper les résultats de toutes les règles activées et déterminer les ensembles flous de sortie correspondants. Si plusieurs règles sont activées, on combine les ensembles flous de sortie "direction_gauche", "direction_droit" et "direction_tout_droit" en utilisant l'opérateur "maximum" :

```
combine_sortie = np.fmax(commande_gauche, np.fmax(commande_droit,
commande_tout_droit))
```

- Les règles principales d'interférence se résume dans le tableau suivant :

Angle / Distance	D	G	Dv
P	TG	TDr	TD
M	TDr	TD	TG
L	TDr	TDr	TDr

Tableau 1. Règles d'interférences

Avec :

P : Prés

M : Moyen

L : Loin

D : Droite

G : Gauche

Dv : Devant

TG : Tournez à Gauche

TD : Tournez à droite

TDr : Tout droit

4.3.5. Défuzzification :

Après avoir combiné les ensembles flous de sortie à l'aide de l'opérateur "maximum" dans la partie précédente, il est nécessaire de défuzzifier la sortie pour obtenir une valeur concrète qui représente la commande de déplacement finale. La défuzzification est le processus de conversion des ensembles flous en une valeur numérique unique :

- Chaque ensemble flou de sortie est représenté par une courbe de membership qui indique la pertinence de la commande correspondante.
- Le centre de gravité est une mesure pondérée qui représente la position centrale de la courbe de membership combinée.
- On Calcule le centre de gravité en effectuant la somme des produits des valeurs de la courbe de membership combinée avec leurs poids respectifs, puis on divise le résultat par la somme des valeurs de la courbe de membership combinée.
- La valeur obtenue à partir du centre de gravité représente la commande de déplacement finale. Cette valeur peut être utilisée pour générer des signaux audios ou des instructions pour guider le non-voyant dans l'évitement des obstacles à l'aide de la bibliothéque gTTS.

4.3.6. Visualisation des résultats :

Cette partie consiste à représenter de manière graphique les ensembles flous d'entrée, les ensembles flous de sortie et la valeur défuzzifiée afin de mieux comprendre le fonctionnement du système d'évitement d'obstacles basé sur la logique floue avec audio feedback utilisant la distance et l'angle comme entrée et la direction comme sortie.

- Pour activer les règles d'interférence et afin d'évaluer le contrôleur d'évitement, on va considérer les paramètres d'entrées suivant :

```
# Distance=1 mètre ; Angle=50 degrés
activ.input['distance'] = 1
activ.input['angle'] = 50
```

Test 1 :

La distance = 1 mètre

L'angle = 50 degrés

Résultat attendue :

Direction = TG [0,1]

Autrement dit, SI la distance est « Prés » ET l'angle est « Droite » ALORS la direction est « Tournez à gauche ». La valeur de déffuzzification obtenue de cette simulation est :

```
activ.output['direction']
0.6629629629629629
```

Le résultat obtenu est de valeur supérieure à 0.5, qui veut dire que la direction est dans la plage de l'ensemble flou de « Tournez à gauche », la figure suivante représente le résultat graphiquement :

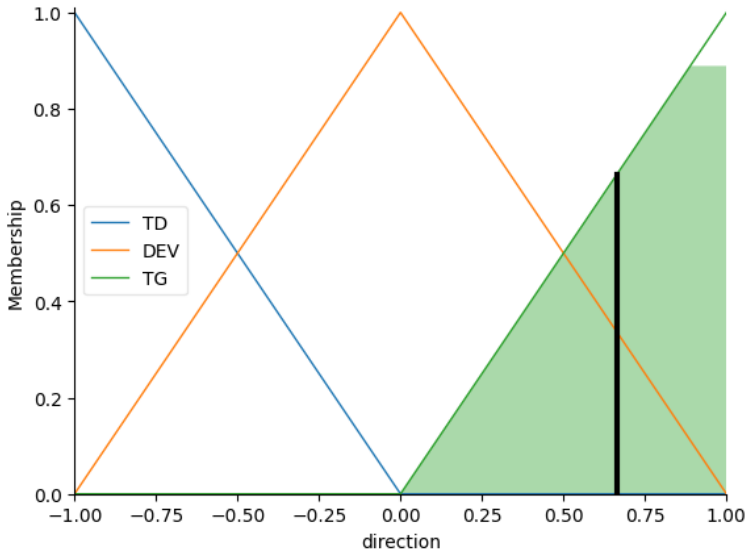

Fig 22. Résultat de simulation 1

Test 2 :

La distance = 3 mètres

L'angle = 120 degrés

```
# Distance=3 métre; Angle=120 degré
activ.input['distance'] = 3
activ.input['angle'] = 120
activ.compute()
```

Résultat attendue :

Direction = TD [-1, 0]

Autrement dit, SI la distance est « Moyen » ET l'angle est « Gauche » ALORS la direction est « Tournez à droite ». La valeur de défuzzification obtenue de cette simulation est :

```
activ.output['direction']
-0.5777777777777778
```

Le résultat obtenu est de valeur égale à -0.57, qui veut dire que la direction est dans la plage de l'ensemble flou de « Tournez à droite », la figure 23 illustre ce résultat :

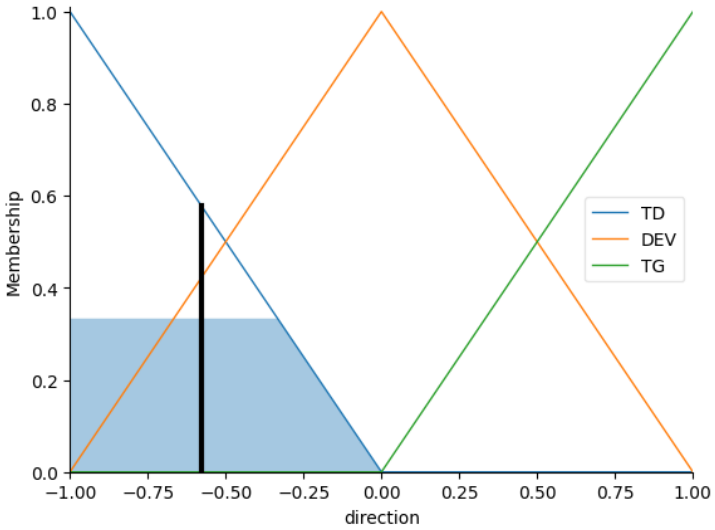

Fig 23. Résultat de simulation 2

Test 3 :

La distance = 5 mètres

L'angle = 90 degrés

```
# Distance=5 métre; Angle=90 degré
activ.input['distance'] = 5
activ.input['angle'] = 90
activ.compute()
```

Résultat attendue :

Direction = TDr [0]

Autrement dit, SI la distance est « Loin » ET l'angle est « Devant » ALORS la direction est « Tout droit ». La valeur de défuzzification obtenue de cette simulation est :

```
activ.output['direction']

-1.850371707708594e-17
```

Le résultat obtenu est de valeur égale à - 1.850371707708594e-17, cette valeur est une notation scientifique pour représenter un nombre très proche de zéro qui veut dire que la direction est dans la plage de l'ensemble flou de « Tout droit ». Ce résultat est représenté dans la figure 24.

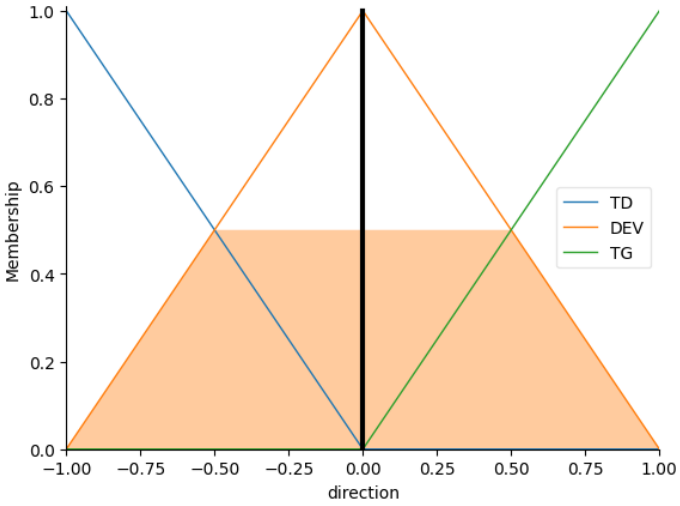

Fig 24. Résultat de simulation 3

4.3.7. Discussion des résultats :

L'efficacité de la logique floue peut être évaluée en fonction de différents critères, tels que la précision de l'évitement des obstacles, la réactivité du système aux changements de l'environnement et la capacité à traiter des situations complexes.

Les résultats des simulations 1,2 et 3 ont montrés que le système d'évitement d'obstacles basé sur la logique floue parvient à éviter les obstacles de manière précise, cela indique une efficacité satisfaisante. La réactivité du système aux changements de l'environnement est également un aspect important. Si le système est capable de réagir rapidement aux nouveaux obstacles ou aux modifications de la configuration de l'environnement, cela témoigne de son efficacité. Par exemple, lorsqu'on a changé les paramètres par rapport à la 1ere simulation (la 2éme et 3éme simulation) le système était capable d'ajuster rapidement la trajectoire de déplacement en fonction des nouvelles informations captées par les capteurs, cela démontre une réactivité appropriée. De plus, le modèle flou était capable de prendre en compte plusieurs paramètres d'entrée, tels que la distance et l'angle, et de générer des commandes de déplacement appropriées. Cela démontre sa capacité à gérer des situations réalistes et variées.

Conclusion :

Ce chapitre de simulation a permis de développer un modèle de système d'évitement d'obstacles pour les non-voyants en utilisant la logique floue. Cela a été accompli en définissant les variables floues, les ensembles flous, les règles d'interférence et en effectuant la combinaison des ensembles flous d'entrée pour obtenir des commandes de déplacement appropriées. Cette simulation offre une approche prometteuse pour aider les non-voyants à éviter les obstacles et à se déplacer en toute sécurité. En effet, la logique floue offre une approche efficace pour modéliser et gérer l'incertitude dans un système d'évitement d'obstacles pour les non-voyants. Elle permet une flexibilité, une adaptabilité et une prise en compte des multiples facteurs tout en offrant une explication claire des résultats générés. Cependant, une conception soignée et une calibration précise sont essentielles pour garantir une efficacité optimale du système.

Conclusion Générale

Conclusion et perspectives :

L'objectif de ce mémoire est d'offrir une solution prometteuse pour améliorer la mobilité et la sécurité des personnes non-voyantes en constituant un système d'évitement d'obstacles basé sur la logique floue sur Google Colab. Il représente une étape importante vers une société plus inclusive, où les personnes non-voyantes peuvent se déplacer de manière autonome et en toute confiance dans leur environnement.

En premier lieu, on a procédé à une recherche bibliographique approfondie afin d'explorer les technologies existantes dans le domaine de la mobilité des personnes non-voyantes. Ces approches mettent en avant les avantages et les perspectives d'amélioration de la mobilité et de la sécurité des non-voyants, en offrant des solutions adaptées à leurs besoins spécifiques. Cependant, des défis et des opportunités de recherche supplémentaires sont également identifiés, ce qui ouvre la voie à de nouvelles études et développements dans ce domaine.

Par la suite, on a procédé à la présentation détaillée de la théorie de la logique floue, en mettant l'accent sur sa pertinence et son application à la prise de décision. On a expliqué les principes fondamentaux de la logique floue, tels que la modélisation des incertitudes et la représentation des connaissances vagues. On a également discuté des méthodes d'interférence floue, telles que l'utilisation de règles d'interférence et de fonctions d'appartenance, pour guider la prise de décision dans un environnement complexe et incertain.

Dans le troisième chapitre, on a consacré une section à l'initiation à Python, mettant l'accent sur la compréhension et la manipulation efficace du langage pour la création du système. On a abordé plusieurs aspects clés de Python, notamment sa syntaxe, les différents types de données qu'il offre, les opérations que l'on peut effectuer, les fonctions qui permettent de regrouper des blocs de code réutilisables, ainsi que les bibliothèques et les Framework. Grâce à cette initiation, on est désormais en mesure d'utiliser les concepts essentiels de Python dans la mise en œuvre du système d'évitement d'obstacles.

En dernier lieux, on a créé le modèle du système d'évitement floue en utilisant la plateforme Google Colab, avec comme paramètres d'entrée la distance et l'angle des obstacles. Des tests ont été effectués en variant les paramètres d'entrée, et les résultats obtenus ont été analysés. L'efficacité du système d'évitement d'obstacles a été évaluée en fonction de critères tels que la précision de la détection des obstacles, la réactivité aux changements de l'environnement et la capacité à traiter des situations complexes. Les résultats obtenus ont démontré que le système d'évitement d'obstacles basé sur la logique floue, était capable de prendre des décisions appropriées pour éviter les obstacles et assurer la sécurité des non-voyants. Il a démontré sa capacité à s'adapter à des situations variées et à fournir des réponses adaptées en temps réel.

Enfin, le système d'évitement d'obstacles pour les non-voyants basé sur la logique floue, développé et simulé sur la plateforme Google Colab, constitue une solution prometteuse pour améliorer la mobilité et la sécurité des non-voyants dans leurs déplacements quotidiens. Son utilisation de variables floues, de règles d'interférence et de capteurs d'entrée permet une prise de décision flexible et adaptative, offrant ainsi une réponse précise et efficace face aux obstacles. Ce système ouvre de nouvelles perspectives pour la conception de solutions technologiques innovantes au service des personnes en situation de handicap visuel. Parmi ses perspectives à prendre en considération :

- **Intégration avec d'autres dispositifs d'assistance :** Le système d'évitement d'obstacles peut être intégré avec d'autres dispositifs d'assistance pour les non-voyants, tels que les systèmes de navigation GPS ou les aides à la communication. Cette intégration permettrait une expérience utilisateur plus complète et synergique.

- **Expérimentation sur le terrain :** Pour évaluer pleinement l'efficacité du système, il est essentiel de mener des expérimentations sur le terrain avec des utilisateurs réels. Cela permettra

de recueillir des retours d'expérience, d'identifier les éventuels défis et d'apporter des ajustements pour répondre aux besoins spécifiques des utilisateurs.

- **L'implémentation sur un Raspberry Pi :** Le système d'évitement d'obstacles basé sur la logique floue pourrait être exécuté localement sur le Raspberry Pi, permettant ainsi une prise de décision en temps réel pour éviter les obstacles. Les résultats de la détection et de la prise de décision pourraient être affichés sur un petit écran intégré ou communiqués à l'utilisateur par le biais d'une interface audio.

- **L'intégration de la fonctionnalité de marche arrière :** En ajoutant la fonctionnalité de marche arrière au système, celui-ci serait capable de détecter les obstacles derrière l'utilisateur et de générer des décisions appropriées pour éviter les collisions. Cela permettrait d'améliorer la flexibilité et la réactivité du système dans des situations où la seule option est de reculer pour éviter un obstacle.

Références

[1]. Obe, Olumide, Dumitrache, I. Fuzzy control of autonomous mobile robot. U.P.B. Sci. Bull., Series C, Vol. 72, Iss. 3, 2010

[2]. Bedaouche, F. Navigation visuelle et évitement d'obstacles utilisant la fonction du champ de potentiel de direction et la logique floue. Doctoral dissertation, 2018

[3]. Muriel Pinto, Rose Denzil Stanley, Sheetal Malagi, Veena Parvathi K., Ajithanjaya Kumar M. K. Smart Cane for the Visually Impaired. American Journal of Intelligent Systems 7(3): 73-76 2017. DOI: 10.5923/j.ajis.20170703.07

[4]. Felix, F., Swai ,R., Dida M., Sinde, R. Development of Navigation System for Blind People based on Light Detection and Ranging Technology (LiDAR). International Journal of Advances in Scientific Research and Engineering 2022. 8. 47 to 55. DOI: 10.31695/IJASRE.2022.8.8.6

[5]. Prince, S. J. D. (2012). Computer Vision: Models, Learning, and Inference. Cambridge University Press.

[6]. Bansal, V., Balasubramanian, K., & Natarajan, P. Obstacle avoidance using stereo vision and depth maps for visual aid devices. SN Applied Sciences 2020, 2, 1-17.

[7]. Chevrie, F., & Guély, F. (1998). La logique floue. Cahier technique, 191, 1-28.

[8]. Godjevac, J. (1999). Idées nettes sur la logique floue. PPUR presses polytechniques.

[9]. Chen, G., Pham, T. T., & Boustany, N. M. (2001). Introduction to fuzzy sets, fuzzy logic, and fuzzy control systems. Applied Mechanics Reviews, 54(6), B102-B103.

[10]. Zadeh, L. A. (2023). Fuzzy logic. In Granular, Fuzzy, and Soft Computing (pp. 19-49). New York, NY: Springer US.

[11]. Klir, G., & Yuan, B. (1995). Fuzzy sets and fuzzy logic (Vol. 4, pp. 1-12). New Jersey: Prentice hall.

[12]. Berenji, H. R. (1992). Fuzzy logic controllers. An introduction to fuzzy logic applications in intelligent systems, 69-96.

[13]. Kacprzyk, J., Szmidt, E., Zadrożny, S., Atanassov, K. T., & Krawczak, M. (Eds.). (2017). Advances in Fuzzy Logic and Technology 2017: Proceedings of: EUSFLAT-2017–The 10th Conference of the European Society for Fuzzy Logic and Technology, September 11-15, 2017, Warsaw, Poland IWIFSGN'2017–The Sixteenth International Workshop on Intuitionistic Fuzzy Sets and Generalized Nets, September 13-15, 2017, Warsaw, Poland, Volume 2 (Vol. 642). Springer.

[14]. Amador-Angulo, L., Mendoza, O., Castro, J. R., Rodríguez-Díaz, A., Melin, P., & Castillo, O. (2016). Fuzzy sets in dynamic adaptation of parameters of a bee colony optimization for controlling the trajectory of an autonomous mobile robot. Sensors, 16(9), 1458.

[15]. Zadeh, L. A. (2002). Toward a perception-based theory of probabilistic reasoning with imprecise probabilities. In Soft Methods in Probability, Statistics and Data Analysis (pp. 27-61). Physica-Verlag HD.

[16]. Du, K. L., Swamy, M. N. S., Du, K. L., & Swamy, M. N. S. (2014). Pattern Recognition for Biometrics and Bioinformatics. Neural Networks and Statistical Learning, 727-745.

[17]. Gil-Lafuente, A. M. (2005). Fuzzy logic in financial analysis (pp. 318-328). Berlin: Springer.

[18]. Derfoufi, Y. (2019). Programmation en langage Python.

[19]. Python, W. (2021). Python. Python Releases Wind, 24.

[20]. Lutz, M. (2001). Programming python. " O'Reilly Media, Inc.".

[21]. Lutz, M. (2010). Programming python: powerful object-oriented programming. " O'Reilly Media, Inc.".