MASTER INGÉNIERIE ET INNOVATION EN IMAGES ET RÉSEAUX

M2 – PARCOURS IMAGES

2022 - 2023

CONTRIBUTION AU DÉVELOPPEMENT D'UN OUTIL DE SUBSTITUTION SENSORIELLE POUR LES PERSONNES AVEUGLES ET MALVOYANTES

SEKKAI SAMY

CENTRE DE DÉVELOPPEMENT DES TECHNOLOGIES AVANCÉES
(ALGERIE)

CDTA

RESPONSABLE DE STAGE : DR DJEKOUNE A. OUALID

TUTEUR : PROFESSEUR AZEDDINE BEGHDADI

Remerciements

Je remercie Allah, qui m'a donné le courage et la force tout au long de mon cursus, et qui m'a aidé à mener à terme ce travail.

J'adresse ma reconnaissance et mes remerciements à mon tuteur Pr Beghdadi, qui m'a donné la chance d'intégrer ce Master, pour la qualité de formation, rigueur scientifique, ainsi que son support et encouragement. Je remercie aussi tous les enseignants de l'institut Galilée qui ont contribué à ma formation durant ce Master.

Je remercie Dr Djekoune du CDTA pour son encadrement, sa patience, et support durant mon stage de Master.

Merci à mes parents, frères et sœurs pour leur soutien tout au long de mes années d'études.

MEM : Micro Electro-Mecanique
MCU : Microcontroller Unit
BLE : Bluetooth Low Energy
SLAM : Simultaneous Localization and Mapping
MBL : Model Based Localisation
VB-GPS : Vision-Based Global Position System
SSD : Single Shot Detection
NAVI : Navigation Assistance for the Visually Impaired
OCR : Optical Character Recognition

Introduction générale

L'accès à l'information visuelle est essentiel dans notre société moderne où de nombreuses activités quotidiennes dépendent de la perception visuelle. Cependant, pour les personnes souffrant de déficiences visuelles, l'accès à cette information peut être entravé, limitant ainsi leur participation active dans divers domaines de la vie. Les systèmes de substitution pour les malvoyants représentent une avancée technologique majeure visant à pallier cette lacune en fournissant des moyens alternatifs d'interagir avec le monde visuel qui les entoure.
Au cours des dernières décennies, la technologie a joué un rôle crucial dans l'amélioration de la qualité de vie des personnes malvoyantes. Les systèmes de substitution visuelle ont émergé comme une réponse novatrice à ce défi, offrant aux individus aveugles ou malvoyants des moyens pour percevoir et comprendre l'information visuelle. Ces systèmes varient en fonction de leurs fonctionnalités, complexités, et portabilité, allant des dispositifs portables portés sur la tête aux applications mobiles intelligentes.

C'est dans ce contexte que se positionne notre travail. Ce dernier consiste en une contribution à la conception et la réalisation d'un système de substitution basé sur la vision artificielle destiné aux malvoyants pour mieux percevoir leur environnement en utilisant d'autres sens que la vue. Il concerne la transformation des données visuelles à partir d'un système binoculaire en signaux auditif et/ou vibratoire.

Ce travail a été réalisé au sein de l'équipe Interaction homme-machine et. Réalité Virtuelle et Augmentée, (IRVA) de la division robotique, du Centre de développement des technologies avancées, (CDTA), Le Centre de Développement des Technologies Avancées (CDTA) est un établissement public à caractère scientifique et technologique, « EPST ».
Ce mémoire est divisé en trois chapitres ; le premier est consacré à un état de l'art sur les systèmes de substitution pour les malvoyants. ; le chapitre 2 est consacré aux systèmes de substitution basés sur la vision artificielle, leurs principes et les algorithmes de traitement d'image utilisés dans leurs conceptions. Enfin, le dernier chapitre est consacré à la présentation du système de substitution que nous avons proposé et la discussion des résultats obtenus.

Chapitre 1

Systèmes de substitution : Etat de l'art

Dans ce chapitre, nous allons présenter les différents systèmes de substitution, leurs principes, les composants qui les constituent et leurs moyens d'interaction système-malvoyant et système-environnement. Nous finirons ce chapitre par un tableau résumant les systèmes de substitution existant dans la littérature, leurs Réponse en sortie ainsi que leur assistance fournie.

1.Systèmes de substitution

1.1. Définition

La substitution visuelle est une approche qui vise à compenser la perte ou la diminution de la perception en utilisant d'autres sens ou techniques. Elle consiste à remplacer ou à suppléer la fonction visuelle par d'autres moyens perceptuels, tels que l'ouïe, le toucher ou la proprioception, ainsi que par l'utilisation de dispositifs technologiques spécifiques. L'objectif de la substitution visuelle est de permettre aux personnes ayant une déficience visuelle de percevoir et d'interagir avec leur environnement de manière plus autonome et fonctionnelle. Cela peut inclure l'utilisation d'applications, de dispositifs d'assistance ou de techniques de formation spécifiques pour améliorer les capacités de navigation, de reconnaissance d'objets ou de lecture [1].

1.2. Architecture de base d'un système de substitution :

Tout système de substitution est constitué de trois composantes principales présentées en figure I.1 :

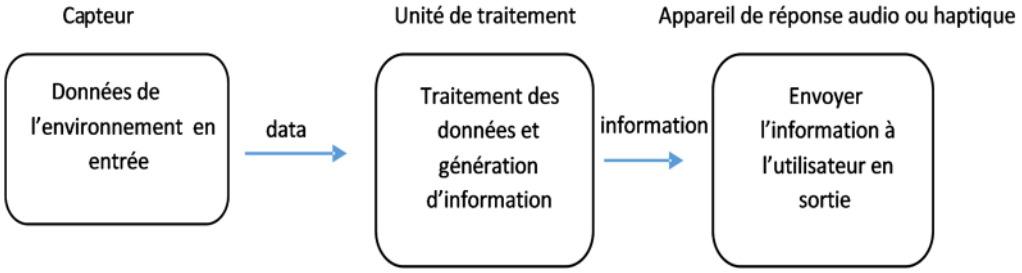

Figure I.1 : composantes principales d'un système de substitution

Le premier module, qui est un capteur de proximité ou de vision, collecte les données de l'environnement sous forme de signaux. Ces derniers sont ensuite transmis au deuxième module qui prend en charge le traitement et la transformation des donnés en information interprétable à transmettre vers le troisième module. Le troisième module est un appareil audio ou haptique qui transmet l'information à l'utilisateur.

1.3. Types de systèmes de substitution :

1.3.1. Systèmes basés sur les capteurs de proximité :

Ces systèmes utilisent des capteurs ultrasoniques ou infra rouge pour mesurer la distance entre l'utilisateur et les objets environnants. Ce qui permettra à ce dernier de détecter et éviter les obstacles proches. [1] Dans ces systèmes, lorsque le capteur détecte un objet à proximité, le système envoie des informations tactiles ou auditives à l'utilisateur pour l'avertir (vibrations, des sons spécifiques ou des signaux sonores de plus en plus forts à mesure que l'objet se rapproche). Ces systèmes sont particulièrement utiles dans les situations de déplacement et de navigation, en aidant les utilisateurs à se déplacer de manière autonome et en toute sécurité en évitant les collisions avec des obstacles tels que des murs, des meubles ou d'autres personnes. Les systèmes basés sur les capteurs ultrasoniques sont utilisés de manière répondue pour les systèmes à canne et les dispositifs portables. Les systèmes

basés sur l'infrarouge sont utilisés avec les dispositifs portables. La composition des deux types de capteurs peut être réalisée pour plus de précision.

Un bon nombre de solutions utilisant les capteurs de proximité ont été développés sous forme de canne intelligente. Une canne blanche intelligente a été réalisée dans [2] sous forme d'un système radar à courte portée pouvant être monté sur une canne blanche traditionnelle. Cet objectif a été atteint en créant un système économique basé sur des composants disponibles dans le commerce et une section micro-ondes utilisant la technologie des micro-rubans, ce qui facilite son intégration avec des antennes faites maison ainsi que la réalisation d'un prototype en utilisant des installations internes. Il permet ainsi d'informer l'utilisateur de la présence d'un obstacle dans une plage plus large et plus sûre. La figure I.2 présente le prototype laboratoire du radar

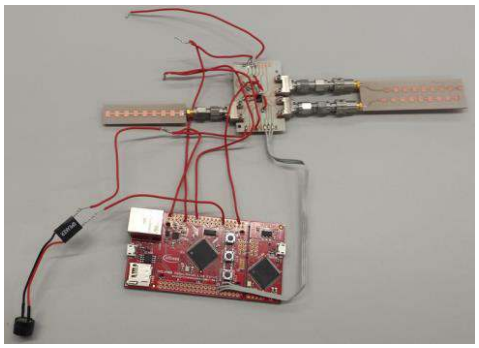

Figure I.2 : prototype du laboratoire de radar [2].

Dans le même objective de détection d'obstacles pour les personnes aveugles, le travail de recherche dans [5] a abouti à la conception d'un prototype de bâton destiné aux personnes aveugles (figure I.3), utilisant la technologie des capteurs pour faciliter l'alerte et les déplacements des personnes aveugles, il est capable de détecter les objets à une distance minimale de 7 centimètres avec une sortie sonore et vibratoire. Le bâton résultant se compose d'une structure en PVC de 0,5 pouce comprenant deux parties : la tige du bâton et l'unité de capteur ultrasonique HS-SRF04 pour la détection des objets.

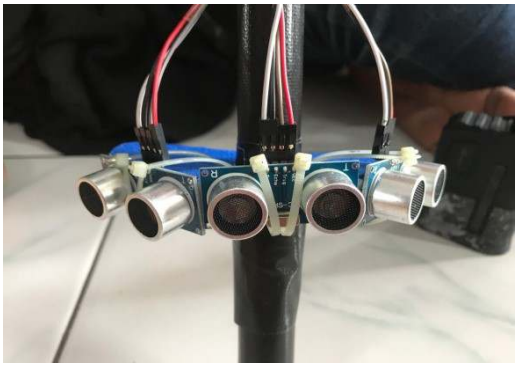

Figure I.3 : Prototype du bâton ultrasonique pour les aveugles [5].

Dans [10] une autre solution sous la forme d'une canne intelligente est proposée ; elle est équipée d'un capteur infrarouge pour détecter les escaliers, ainsi que d'une paire de capteurs ultrasoniques afin de repérer tout autre obstacle se trouvant devant l'utilisateur dans un rayon de quatre mètres (figure I.4). De plus, un autre capteur est placé à la base de la canne pour éviter les flaques d'eau. Un message vocal d'avertissement est déclenché dès qu'un obstacle est détecté.

(A) structure physique du bâton, (B) prototype du bâton

Figure I.4 : Structure physique et prototype de la canne intelligente [10].

Un travail similaire a été réalisé dans [16] en ajoutant un capteur d'humidité. Par conséquent, la canne blanche intelligente proposée comporte des capteurs ultrasoniques à des positions spécifiques sur la canne, fournissant ainsi des informations sur l'environnement à l'utilisateur grâce à l'activation d'un signal sonore de type buzzer. Un microcontrôleur traite les signaux et alerte la personne malvoyante en présence de tout obstacle, d'eau ou de zones sombres, à l'aide de sons de bip. Le système a été conçu, programmé en langage C (figure I.5).

Figure I.5 : Canne intelligente ARDUINO pour les personnes malvoyantes [16]

Un système combinant des lunettes intelligentes et une canne intelligente a été proposé dans [14]. L'objectif est la détection d'obstacles et d'alerte en cas de chute. Les lunettes intelligentes intègrent divers composants, tels qu'un module de capteur émetteur-récepteur infrarouge, un module de capteur MEM (micro-électromécanique), une unité de microcontrôleur (MCU), un module de communication sans fil Bluetooth à faible consommation d'énergie (BLE) et un module de charge de batterie (figure I.6). D'autre part, le prototype de canne intelligente (figure I.7). Ce système est équipé d'un moteur à vibration, d'un module GPS, d'un MPU, d'un module de communication sans fil BLE, d'un module de communication sans fil LPWAN basé sur LoRa, et d'un module de capteur MEM (micro-électromécanique). L'utilisateur du système doit porter les lunettes intelligentes et tenir la canne. Ainsi, les obstacles devant eux peuvent être détectés par les lunettes intelligentes et rappelés par la canne intelligente. De plus, en cas de chute ou de collision, les informations pertinentes (GPS, chute, etc.) seront enregistrées et téléchargées sur la plateforme d'information en ligne. Les informations pertinentes seront également immédiatement communiquées aux personnes concernées (la famille ou soignants) via l'application sur les appareils mobiles proposée.

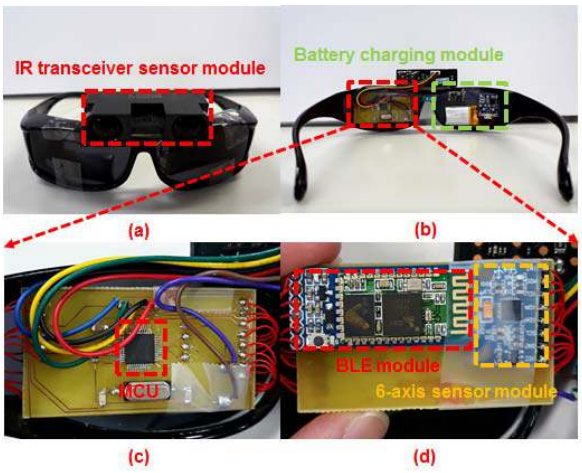

Figure I.6 : Prototype des lunettes intelligentes portables proposées [14].

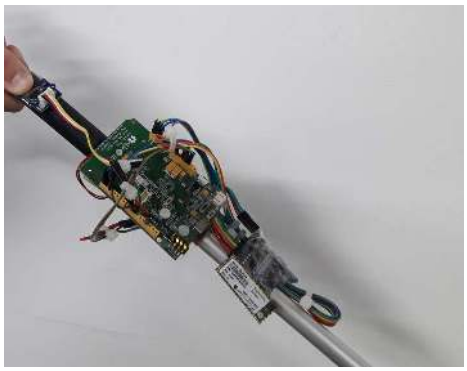

Figure I.7 : Prototype de la canne de marche intelligente proposée. [14].

1.3.2. Systèmes de substitution basés sur la vision :

Les systèmes de substitution visuelle basés sur la vision sont conçus pour aider les personnes déficientes visuelles en capturant, analysant et présentant les informations visuelles de leur environnement. Ils utilisent des technologies d'acquisition d'image et des algorithmes de vision par ordinateur, tels que SIFT, SURF et MSER, pour détecter et décrire les caractéristiques visuelles. Ces systèmes ont diverses applications, comme la lecture de textes, la reconnaissance d'objets, la navigation et la reconnaissance faciale.

Dans [4], une chaussure intelligente appelée NavGuide a été développée pour détecter précisément les obstacles dans différentes directions et les sols mouillés (figure I.8). Elle comprend des capteurs ultrasoniques, des moteurs à vibrations et une batterie, avec trois modules : collecte de données,

construction de carte logique et retour d'information tactile ou audio pour l'utilisateur.

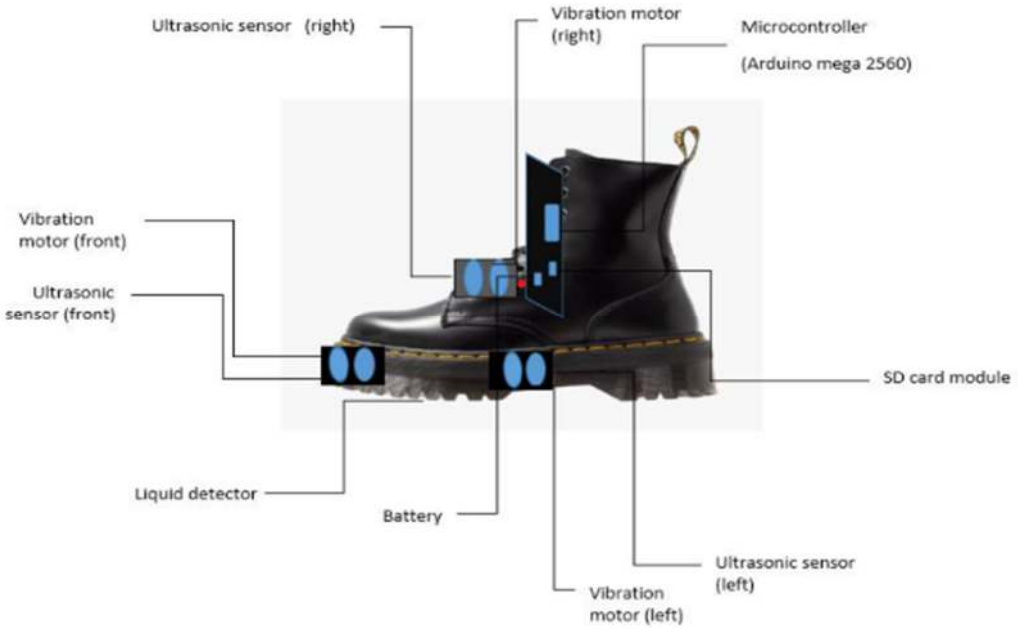

Figure I.8 : Chaussure NavGuide montrant l'emplacement de ses composants [4].

Un système basé sur la vision par ordinateur 3D et Deep Learning (figure I.9) a été développé dans [30] ; il est conçu pour les personnes atteintes de troubles de la vision. L'appareil est portable, l'utilisateur reçoit des messages audios et des retours tactiles qui lui fournissent des informations essentielles sur son environnement et l'aidant à se déplacer et à éviter les obstacles. Un capteur RGBD personnalisé basé sur la vision stéréo passive est utilisé pour obtenir les nuages de points 3D. Les obstacles détectés sont sémantiquement catégorisés grâce au système Deep Learning développé. L'utilisateur est alerté grâce à un gant avec des micromoteurs pour un retour tactile et un casque à conduction osseuse ce qui permet à l'utilisateur de rester conscient de son environnement.

Figure I.9 : À gauche, les composants matériels de l'aide à la mobilité proposée. À droite, le système global porté par un utilisateur [30]

Les auteurs dans [29] ont introduit en 2019, une solution innovante pour une assistante contextuelle hybride (intérieure/extérieure) des personnes atteintes de perte de vision périphérique dans leur navigation en détectant et classifiant les dangers (figure I.10). Ceci est réalisé grâce à des lunettes intelligentes équipées d'une caméra grand-angle. Ce système complète la vision existante des utilisateurs en leur fournissant des notifications adaptées, significatives et intelligentes afin d'attirer leur attention sur d'éventuels obstacles ou dangers situés dans leur champ de vision périphérique. Un détecteur d'objets basé sur le Deep Learning est utilisé pour reconnaître en temps réel les objets statiques et en mouvement. Après avoir détecté les objets, un tracker multi-objets basé sur le filtre de Kalman est utilisé pour suivre ces objets dans le temps et déterminer leur modèle de mouvement. & Pour chaque objet suivi, son modèle de mouvement représente sa façon de se déplacer autour de l'utilisateur. Les caractéristiques de mouvement sont extraites lorsque l'objet se trouve encore dans le champ de vision de l'utilisateur. Ces caractéristiques sont ensuite utilisées pour quantifier le danger

en utilisant un classifieur basé sur un réseau neuronal et comprenant cinq classes de dangers prédéfinies. Les notifications (alertes) sont présentées sous forme visuelles dans le champ de vision sain de l'utilisateur.

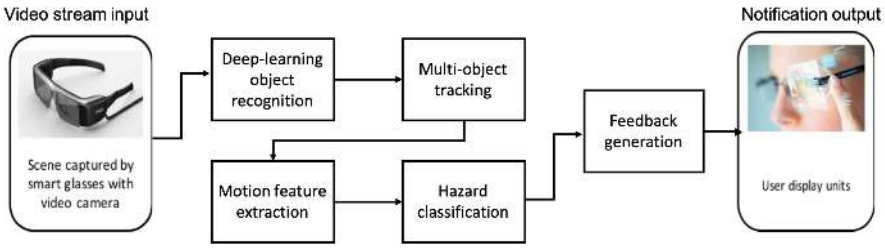

Figure I.10 : Présentation générale du système, y compris les principaux éléments du projet [29]

Un assistant visuel pour une personne malvoyante a été proposé dans [28] et a pour objectif d'améliorer la perception d'environnement des malvoyants. Le système se compose d'un dispositif portable équipé d'une caméra stéréo RGBD et d'un écouteur, d'un puissant processeur pour les inférences Deep Learning (Inception V3) et d'un smartphone pour l'interaction tactile (figure I.11). L'approche d'apprentissage adoptée dans ce système est basée sur les données RGBD et une carte sémantique. Cette dernière aide également les personnes malvoyantes à comprendre les objets en 3D et la disposition de leur environnement grâce à des interactions bien conçues sur l'écran tactile.

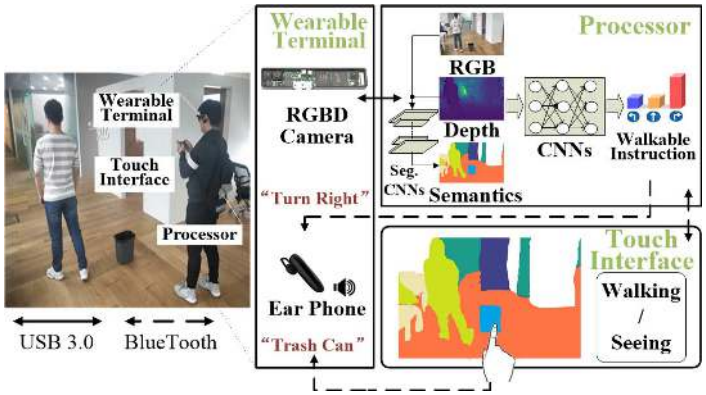

Figure I.11 : vue globale de l'assistant visuel pour une personne malvoyante [28]

Utilisant une caméra multimodale, les auteurs dans [23] ont proposé une approche qui utilise un pipeline de localisation visuelle exploitant des images de différentes modalités pour générer des descripteurs locaux et globaux robustes, puis obtenir des résultats précis grâce à la vérification géométrique et la correspondance de séquences pour la navigation assistée. Le pipeline proposé comprend un réseau de descripteurs profonds, une vérification géométrique 2D-3D et une correspondance de séquences en ligne. Les images de différentes modalités (RGB, infrarouge et profondeur) sont entrées dans le réseau « Dual Desc » pour générer des descripteurs globaux et des caractéristiques locales robustes. Les caractéristiques locales 2D, ainsi que le nuage de points 3D épars, sont utilisées dans la vérification géométrique pour sélectionner les résultats optimaux parmi les candidats récupérés. Enfin, la correspondance de séquences renforce les résultats de localisation en synthétisant les résultats vérifiés des images consécutives. Des écouteurs à conduction osseuse sont utilisés pour alerter l'utilisateur, le dispositif réalisé est présenté en figure I.12.

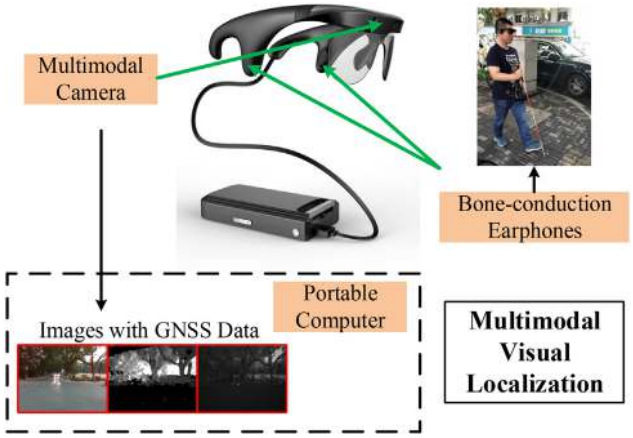

Figure I.12 : Dispositif d'assistance Intoer utilisé pour enregistrer des données de plusieurs modes simultanément [23]

V-Eye qui est un système de navigation basé sur la vision a été proposé dans [24] ; il a pour objectif de permettre une navigation fluide des malvoyants/aveugles dans les espaces intérieurs ou extérieurs, en leur fournissant une orientation précise en temps réel et détecter les obstacles en mouvement. Ceci est réalisé en utilisant une méthode de localisation globale unique appelée VB-GPS (Vision-based Global Position System) et des techniques de segmentation d'image pour améliorer la compréhension de la scène à l'aide d'une seule caméra. Le VB-GPS combine deux approches de localisation basées sur la vision, à savoir la SLAM visuelle (Simultaneous Localization and Mapping) et la localisation basée sur un modèle (MBL). Les tests expérimentaux réalisés sur V-Eye ont montrés qu'il est capable de détecter des obstacles imprévus et de faciliter la navigation à la fois à l'intérieur et à l'extérieur des environnements. Quant à la localisation le système peut fournir de manière fiable des informations précises sur la position et l'orientation, avec une erreur médiane d'environ 0,27 mètre et 0,95 degré respectivement, l'architecture du V-Eye est montrée en figure I.13.

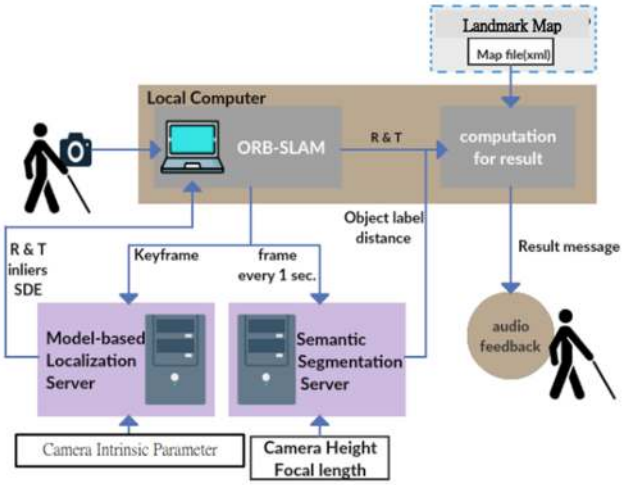

Figure I.13 : Architecture de V-Eye [24]

Les systèmes basés sur la vision ont vu l'introduction de la caméra PI et la carte Raspberry PI dans leurs conceptions ces dernières années. Les auteurs dans [11] proposent un système basé sur le Deep Learning et la segmentation d'image capable de détecter des objets en temps réel, fournir des réponses rapides et précises à l'utilisateur de l'appareil concernant les objets détectés, leur position par rapport à la personne et la précision de la détection (figure I.14). Pour ce faire, une combinaison du Single Shot Detection (SSD)de l'architecture MobileNet est utilisée, permettant une détection rapide et en temps réel de plusieurs objets. Selon le développeur du système, SSD est beaucoup plus rapide et

précis par rapport à d'autres méthodes. MobileNet est utilisé comme réseau de base. Le réseau de base peut être considéré comme les couches standard de l'architecture prédéfinie utilisées aux premières étapes du modèle de réseau pour une classification d'image améliorée. MobileNet est adapté aux appareils tels que Raspberry Pi, smartphones, etc. Le système convertit le résultat qui est en format texte à une sortie audio avec un convertisseur texte-to-speech. Dans cette solution les développeurs se sont intéressés à la construction d'un appareil compact, portable, avec un temps de réponse minimal.

SSD : *Le système SSD (Single-Shot Detection) est une méthode de détection d'objets en vision par ordinateur qui permet de localiser et de classifier des objets dans une image en un seul passage. Contrairement à certaines approches traditionnelles qui nécessitent plusieurs étapes pour détecter les objets, le système SSD effectue cette tâche en une seule étape, ce qui le rend plus rapide et plus efficace. Il utilise un réseau de neurones convolutifs (CNN) pour extraire des caractéristiques de l'image, puis une série de prédicteurs pour estimer la position et la classe des objets détectés [11].*

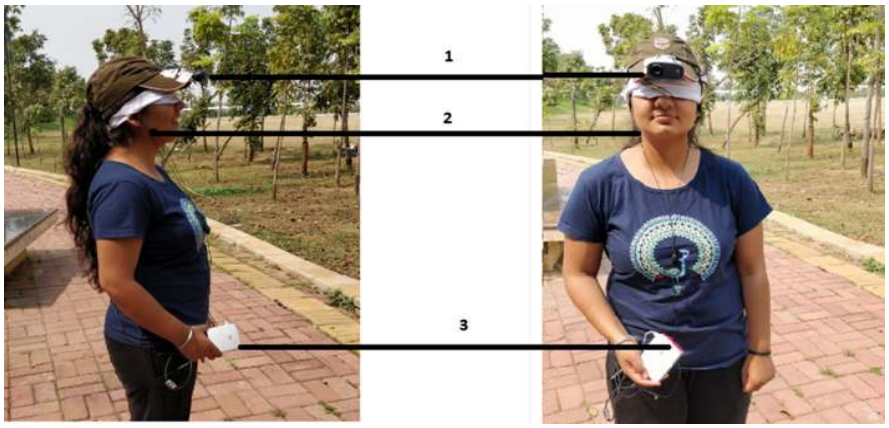

Figure I.14 : appareil et composant :(1) module caméra fixé sur une casquette, (2) écouteur, (3) Raspberry pi [11]

Un travail similaire à celui proposé dans [11] a été réalisé dans [13] avec plus de fonctionnalités de reconnaissance et sans l'utilisation d'une carte Raspberry (figure I.15). Ce système se base sur l'apprentissage par transfert (Transfer Learning) pour appliquer le mécanisme de SSD à la détection et à la classification des objets. De plus, le modèle Inception v3 est utilisé pour reconnaître les visages humains et les billets de banque s'ils sont détectés. Le résultat est présenté à l'utilisateur sous format audio, les résultats générés par le cadre peuvent être présentés aux personnes malvoyantes sous forme audio.

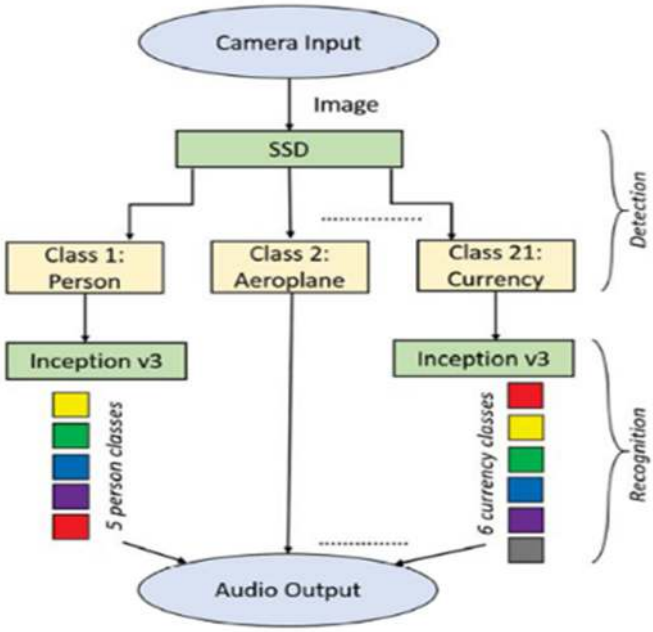

Figure I.15 : Représentation schématique du diagramme de flux de contrôle du framework d'apprentissage profond [13]

Les auteurs dans [20] proposent un système pour la détection et l'identification d'objets communs dans une maison (figure I.16). En utilisant une caméra PI l'objet est détecté et capturé ; la classification est réalisée en utilisant l'architecture SSDlite et MobileNet V2. Le système est capable de calculer la distance entre l'utilisateur et les objets détectés. Le système génère une sortie audio fournissant des informations en anglais ou en hindi. L'évaluation du système montre une précision de 0,85 et un rappel de 0,8, avec un délai de sortie audio de 2 secondes.

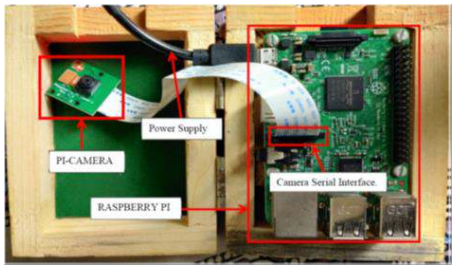

Figure I.16 : Mise en place matérielle du système.[20]

1.4. Étude comparative entre les systèmes de substitution

Le (tableau A.1 en annexe) résume les systèmes de substitution existant dans la littérature en termes de capteurs utilisés, Réponse en sortie et le dispositif développé, ainsi que l'assistance fournie.

1.5 Conclusion :

Dans ce chapitre, nous avons présenté les différents systèmes de substitution de la littérature. Nous avons expliqué leurs principes, présenté les composants qui les constituent et les interactions avec le malvoyant. Nous avons aussi présenté une liste non exhaustive de systèmes de substitution en montrant leurs réponses en sortie ainsi que leur assistance fournie. Nous avons montré que ces systèmes se réunissent dans leurs constituants principaux à savoir : Les capteurs transmettant l'information environnante aux malvoyants, d'une unité de calcul pour le traitement de cette information et d'un dispositif fournissant des vibrations, des sons ou autres types d'information

assistant ainsi le malvoyant dans ses déplacements.
Dans le chapitre suivant, nous nous focaliserons principalement sur les systèmes de substitution basés sur les techniques de la vision artificielle.

Chapitre 2

Systèmes de substitution basés sur la vision artificielle

Les systèmes de substitution basés sur la vision visent à fournir une perception visuelle artificielle aux personnes aveugles ou ayant une vision limitée. Ils utilisent des caméras et des logiciels pour capturer et traiter des images de l'environnement, convertissant ensuite les informations visuelles en formats adaptés à l'utilisateur, tels que des signaux auditifs ou des représentations tactiles. Ces systèmes permettent aux personnes déficientes visuelles de reconnaître des objets, de naviguer, de lire du texte, d'identifier des visages, et plus encore, améliorant ainsi leur autonomie. Les avancées technologiques continuent d'améliorer ces systèmes, offrant de nouvelles opportunités aux personnes atteintes de déficiences visuelles. Dans ce qui suit, nous présenterons les différents types de systèmes de substitution basés sur la vision de la littérature. Une description détaillée des algorithme utilisés par ces systèmes seront présentée en annexe

1.Systèmes de substitution basés vision

1.1. Données RGB-D

Le système de Navigation Assistée pour les Personnes Malvoyantes (NAVI) (figureII.1) décrit dans [34] utilise une caméra RGB-D Asus Xtion Pro attachée au cou de l'utilisateur pour enregistrer des données visuelles et de portée. Ces données sont traitées par un algorithme en C++ avec ROS, OpenCV, et la bibliothèque PCL sur un ordinateur portable HP EliteBook 8440p placé dans un sac à dos. Les informations sont transmises à l'utilisateur via un casque d'écoute qui génère des signaux sonores pour signaler la présence d'obstacles.
Le système a été testé avec succès dans divers environnements intérieurs complexes, mais il présente des limitations en extérieur en raison de la variation des relevés de la caméra RGB-D en présence de lumière du soleil.

Figure II.1 : La configuration expérimentale consiste à suspendre la caméra de portée autour du cou de l'utilisateur. Cet appareil est léger et confortable à porter.[34]

Dans [35] un prototype portable a été développé pour aider les personnes malvoyantes en utilisant un capteur RGB-D et un Intel RealSense R200 qui étend la détection de la zone traversable par le mal voyant (figure II.2). Un filtrage guidé par une image RVB est utilisé pour améliorer la perception réelle de l'image de profondeur. De plus, la segmentation par consensus d'échantillons aléatoires et l'estimation du vecteur normal de surface sont utilisées pour couvrir la zone traversable. Le système comprend une camera RealSense R200, une caméra infrarouge, une caméra RVB, un processeur, un capteur d'attitude et un casque à conduction osseuse. Le prototype portable proposé fonctionne à la fois en intérieur et en extérieur. Cependant, son fonctionnement n'est pas optimisé car le prototype portable proposé est lent et ne couvre qu'une courte distance.

Figure II.2 : Le système d'assistance est composé d'une monture qui contient le RealSense R200 et le capteur d'attitude, d'un processeur et d'un casque à conduction osseuse [35]

1.2. Systèmes multi-agents

Dans [36], un système de Navigation Assistée pour les Personnes Malvoyantes (NAVIG) a été développé pour permettre une navigation sécurisée dans des environnements inconnus (figure II.3). Le prototype fonctionne sur un ordinateur portable équipé d'un processeur Intel i7 QuadCore. L'architecture est basée sur un cadre multi-agent utilisant le middleware IVY, permettant aux agents de se connecter dynamiquement à différents flux de données.

L'agent "vision artificielle" utilise l'algorithme SNV de Spikenet Technology, Inc., avec des flux vidéo provenant d'une caméra stéréoscopique BumbleBee. L'agent "fusion" intègre les données de positionnement de l'agent "vision", incluant un GPS ANGEO et deux trackers de mouvement inertiel. L'interaction avec l'utilisateur se fait à travers des casques acoustiquement transparents et un microphone. Une interface utilisateur utilise des informations sonores et des commandes vocales pour fournir des instructions de navigation, générant une carte sonore avec des bips stéréo correspondant à la distance aux obstacles.

Le système NAVIG peut également assister la navigation en intérieur en utilisant une carte du bâtiment intégrant des modèles de cibles localisées fixes. Il peut s'appuyer sur la vision, la position estimée, et la localisation en intérieur pour calculer un itinéraire approprié.

En plus de la navigation, NAVIG propose une fonctionnalité de localisation et d'identification d'objets similaires, tels que des billets de banque. Les tests ont montré une bonne utilisabilité, mais le dispositif est coûteux et lourd.

Figure II.3 : prototype NAVIG [36]

1.3. Apprentissage profond

Dans [37], un outil portable utilisant la segmentation sémantique pixel par pixel a été développé pour aider les personnes malvoyantes (figure II.4). Ce système identifie les zones praticables, les escaliers, les obstacles d'eau, les trottoirs, et évite les piétons rapides, les obstacles proches et les véhicules. Il repose sur l'apprentissage en profondeur avec une base de données ADE20K pour l'entraînement.

Le dispositif comprend une caméra infrarouge, une caméra RGB, un processeur d'image RealSense, et des casques à conduction osseuse pour transmettre les résultats aux utilisateurs sous forme de sons stéréo. Cependant, des limitations subsistent, notamment en termes de robustesse pour les scénarios inter-saisons, de perception RGB-D en environnements à haute dynamique, de plage de détection, de couverture des éléments de scène aux intersections, et de prise en compte des bordures dangereuses et des flaques d'eau pour améliorer la perception sémantique liée à la praticabilité.

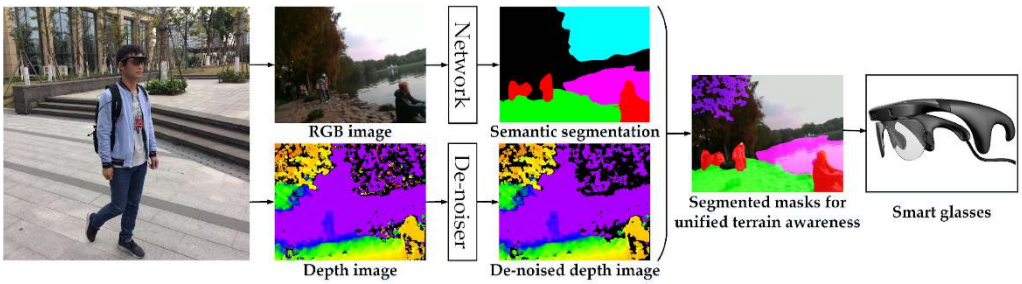

Figure II.4: Résumé du système d'assistance à la navigation [37]

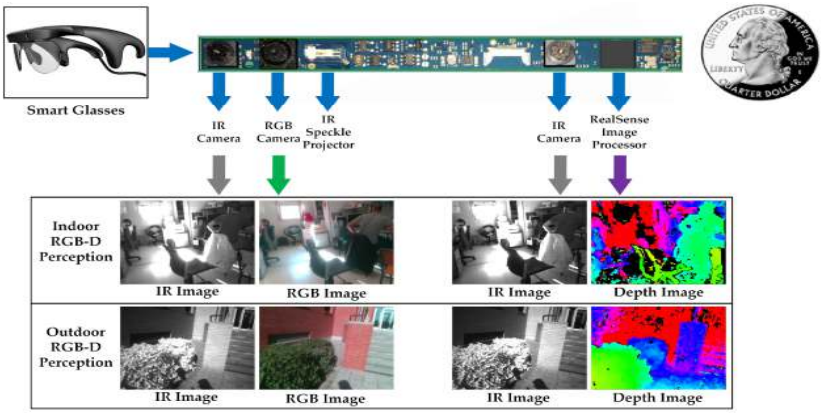

Figure II.5 : Perception RGB-D des lunettes intelligentes dans des environnements intérieurs et extérieurs [37]

1.4. Systèmes d'assistance par module égomotion

Dans [40], un prototype a été développé pour permettre aux personnes malvoyantes de se déplacer de manière autonome et de reconnaître des objets dans des environnements publics intérieurs (figure II.6) . Le système comprend un module de reconnaissance vocale pour acquérir les instructions de l'utilisateur, un capteur laser pour détecter les obstacles, un module d'égomotion pour estimer la position de l'utilisateur, et un module de planification de trajectoire pour calculer un trajet sûr. Une caméra portable capture la scène, et un module de synthèse vocale fournit des retours audios. Le prototype permet à l'utilisateur de se déplacer et d'interagir vocalement pour obtenir des informations sur les objets remarquables tout en évitant les obstacles.

Cependant, l'environnement intérieur doit être adapté aux besoins des personnes malvoyantes, et l'interaction se fait principalement par la voix. Le prototype est capable d'éviter les obstacles statiques et dynamiques

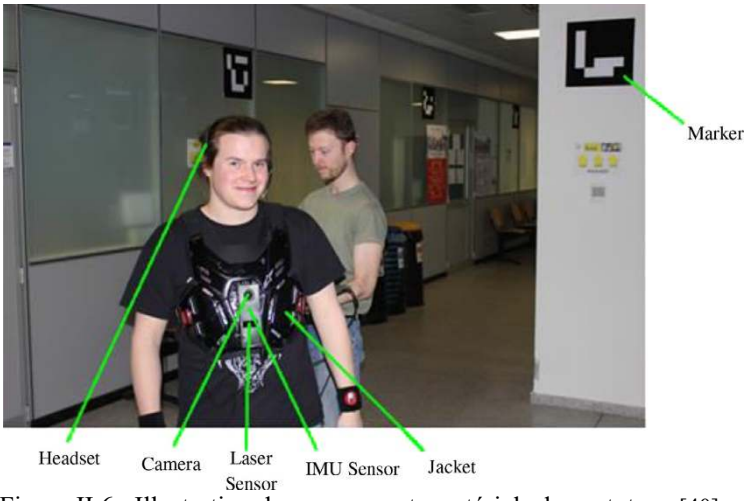

Figure II.6 : Illustration des composants matériels du prototype [40]

1.5. Système de guidage à caméra multiple

Dans [41], un système de guidage pour les personnes aveugles et malvoyantes est présenté, visant à faciliter leur déplacement en intérieur et en extérieur en fournissant des informations en temps réel sur l'environnement et la circulation (figure II.7). Le système repose sur une caméra stéréo composée de trois types de caméras, générant des signaux audios pour la navigation. Les modules sont montés sur un casque portable, tandis que l'unité de traitement est un ordinateur portable avec un processeur Intel Core i5 3210M et 8 Go de RAM, placé dans un sac à dos.

En intégrant un module de communication mobile, le système peut se connecter aux services de transport en commun pour aider à identifier les bus, facilitant ainsi l'embarquement pour les personnes malvoyantes. Les obstacles sont détectés en analysant les vecteurs de mouvement, permettant de classifier les objets en fonction de leur trajectoire.

Cependant, l'unité de traitement du prototype se limite aux entrées vidéo, et certains utilisateurs

malvoyants ont trouvé le système bruyant et obsolète

Figure II.7 : Un participant teste le prototype lors de la démonstration publique [41]

1.6. Système de navigation utilisant une caméra IP

L'article [42] décrit un système de navigation intérieure pour les personnes malvoyantes et aveugles, utilisant des caméras IP au plafond de chaque pièce et un smartphone comme interface homme-machine (figure II.8). Les caméras transmettent des images à un ordinateur qui analyse l'environnement, détecte les objets et guide l'utilisateur par le biais d'alertes et d'instructions vocales via une application mobile.

Ce système offre une architecture simple qui favorise l'indépendance de l'utilisateur en intérieur, en fournissant une assistance à la navigation, la détection d'obstacles et la reconnaissance d'objets. Les premiers retours d'expérience sont positifs, les utilisateurs se sentent en sécurité et peuvent se déplacer librement.

Cependant, le système a des limitations, notamment le fait qu'il ne s'active pas automatiquement lors du mouvement de l'utilisateur, une amélioration est nécessaire pour l'algorithme de calcul du trajet, la nécessité d'améliorer le temps de réponse, et le fait qu'il ne fonctionne qu'en intérieur.

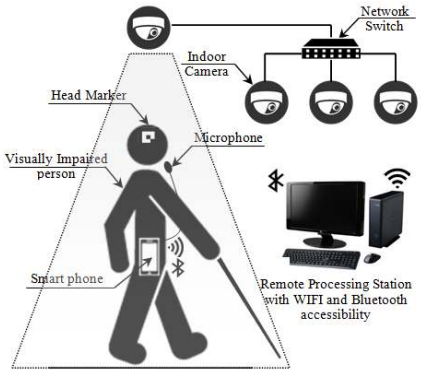

Figure II.8 : Architecture de l'approche du système proposé [42]

1.7. Cartographie SLAM :

Un système de navigation pour les personnes malvoyantes a été proposé par l'auteur dans [43]. Le système comprend un casque équipé de caméras stéréo orientées vers l'avant, un smartphone basé sur

Android, une application web et une plateforme de calcul en nuage (figure II.9). Le cœur du système réside dans la plateforme de calcul en nuage, qui intègre des algorithmes d'apprentissage profond pour la détection et la reconnaissance d'objets, la reconnaissance optique de caractères (OCR), le traitement vocal, la localisation et la cartographie simultanées basées sur la vision (SLAM) et la planification de trajets. Les utilisateurs malvoyants interagissent avec le système par le biais de commandes vocales à l'aide d'un microphone. La plateforme en nuage communique avec le smartphone via le Wi-Fi ou la technologie de communication mobile 4G. Deux séries de tests ont été réalisées pour évaluer les performances du système : la perception et la navigation. Les résultats montrent que le système proposé détecte et reconnaît la monnaie chinoise (RMB). Avec une précision de 99,9%

Cependant, le suivi de la localisation n'est pas assez précis et le guidage n'est pas assez efficace car la fin du réseau d'apprentissage profond doit être ajustée. Aussi le nombre d'objets identifiés est limité.

Figure II.9 : Système de navigation utilisant le cloud computing et la technologie de vision par ordinateur [43]

La description des algorithmes utilisés par les systèmes de substitutions présenté dans ce chapitre est détaillée en annexe B.

2. Conclusion

Les systèmes de substitution basés sur la vision artificielle utilisent en général des caméras qui permettent de capturer l'information sur l'espace environnant. A la différence des autres types de systèmes de substitution, ces systèmes utilisent des algorithmes de vision par ordinateur pour traiter l'information et la fournir aux malvoyants. Dans ce chapitre, nous avons présenté les différents types de systèmes de substitutions basé sur la vision existant dans la littérature avec une vue critique de ces derniers, nous avons aussi donné une description détaillée de ces systèmes en annexe.

Dans le chapitre suivant, nous allons présenter le système de substitution que nous avons conçu et réalisé.

Chapitre 3

Système de substitution proposé : Conception et réalisation.

Ce chapitre est consacré à la partie réalisation de notre module de vision stéréo du système de substitution que nous proposons pour les malvoyants. Il est divisé en trois parties : la partie hardware, la partie software et la partie discussions des résultats obtenus.

I. Partie hardware

Notre module de vision est constitué de deux caméras et d'une unité de calcul (UC) qui fournit des avertissements sonores au malvoyant (figure III.1).
Les caméras servent d'acquisition de l'information visuelle de l'environnement proche au malvoyant tandis que l'unité de calcul (UC) sert à faire des traitements pour calculer la distance des différents obstacles se trouvant sur son chemin

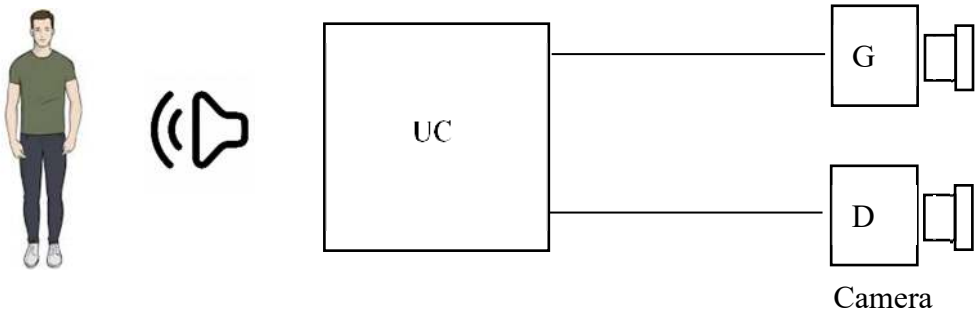

Figure III.1 : Schéma bloc du dispositif

I.1. Stéréovision

I.1.1. Définition

Dans le domaine médical, la vision stéréoscopique, ou stéréopsie binoculaire, est la capacité de déterminer la distance des objets en fonction de la différence de leurs positions telles qu'elles sont perçues par chaque œil. Ceci est réalisé en comparant les positions relatives de l'objet dans les deux yeux pour en déduire la perception de la profondeur.
Dans le traitement d'image, la vision stéréoscopique est une technique de vision artificielle capable de fournir des mesures tridimensionnelles sur l'ensemble du champ de vision en utilisant deux caméras ou plus. Le fondement de la vision stéréoscopique est similaire à la perception 3D dans la vision humaine et repose sur la triangulation des rayons provenant de multiples points de vue [56].

I.1.2. Implémentation de la Stéréovision

Pour implémenter la stéréovision, deux caméras doivent être utilisées ; elles doivent être assez espacées l'une de l'autre. Dans le cadre de notre travail, nous avons utilisé deux caméras *Logitech* C910 espacée de 10 cm (ligne de base) (figure III.2) et dont les caractéristiques sont présentées dans le tableau C.1 en annexe. Elles sont montées sur un support rigide comme le montre la figure III.4. Les images de notre module de vision stéréoscopique sont générées à une cadence de 30 frames/s et ayant la taille de 320 X 240 pixels par image

Figure III.2 : Camera Logitech C910

L'unité de calcul, que nous avons utilisée, est une carte *Raspberry Pi 4* model B (figure III.3) permettant d'effectuer des prétraitements/traitements sur les images et générer des avertissements sonores aux malvoyants. Les caractéristiques de la carte sont présentées dans le tableau C.2 en annexe.

a b

Figure III.3: Carte Raspberry Pi 4 model B : (a) Le boîtier, (B) La carte mère.

Figure III.4 : Caméras stéréoscopiques.

Figure III.5 : Exemple d'images stéréoscopiques acquises par notre module de vision stéréoscopique.

II. Partie software

Cette partie concerne les softwares utilisés pour notre module de vision stéréoscopique qui sont le système d'exploitation de la carte *Raspberry Pi*, le langage de programmation *Python*, ainsi que les différentes librairies et algorithmes de prétraitement et de traitement utilisés (*Opencv*, *Numpy*, *PyTorch*, *jupyterlab*). Ces softwares nous ont permis d'obtenir une carte de disparité et de distance par rapport aux objets constituant l'environnement proche vue par le module de vision.

II.1. Système d'exploitation Raspberry Pi

Le *Raspberry Pi OS* (anciennement Raspbian) est un système d'exploitation de type Unix construit sur la base de la distribution Debian Linux et conçu pour la famille d'ordinateurs monocarte compacts *Raspberry Pi*. Il a été initialement développé indépendamment en 2012 et est le principal système d'exploitation pour ces cartes depuis 2013 [57].

II.2. Langage de programmation python

Python est un langage de programmation interprété, orienté objet et de haut niveau, doté de sémantiques dynamiques. Ses structures de données intégrées de haut niveau, associées à un typage dynamique et une liaison dynamique, contribuent à son attrait pour le développement rapide d'applications. De plus, *Python* sert de langage de script polyvalent, permettant la connexion transparente des composants existants [58].
Dans notre cas deux scripts *Python* ont été utilisés :
Le premier permet la capture des images de qualité qui seront ultérieurement employées dans la calibration des deux caméras, incluant les corrections de la distorsion et la calibration stéréoscopique.

Quant au deuxième, il concerne la vision stéréoscopique. Les lignes épipolaires sont alignées pour faciliter la mise en correspondance des pixels des deux images pour obtenir ainsi une bonne carte de disparité essentielle pour l'estimation de distance entre les objets et le système stéréoscopique. Cette distance est obtenue en exploitant une équation basée sur une ligne droite obtenue expérimentalement en prenant des valeurs de distance dans un intervalle et en faisant la correspondance entre ces valeurs de distance et de disparité. La carte de disparité est filtrée avec un filtre WLS pour une meilleure visualisation de la carte de disparité

II.3. Bibliothèques utilisées

II.3.1. Open CV

OpenCV (Open Source Computer Vision Library) est une bibliothèque logicielle open source dédiée à la vision par ordinateur et à l'apprentissage automatique. Il a été développé pour fournir une infrastructure commune aux applications de vision par ordinateur et pour accélérer l'utilisation de la perception machine dans les produits commerciaux. Dans notre cas nous avons utilisés cette bibliothèque pour sauvegarder les images sur le disque, capturer, rectifier, calibrer, afficher les images du système stéréoscopique, et générer la carte de disparité [59]

II.3.2. Numpy

NumPy est le paquetage fondamental pour le calcul scientifique en *Python*. Il agit en tant que bibliothèque Python essentielle offrant un objet tableau multidimensionnel, ainsi que plusieurs objets connexes tels que des tableaux masqués et des matrices. *NumPy* propose également une gamme de fonctions conçues pour des opérations rapides sur les tableaux, englobant des opérations mathématiques, logiques, de manipulation de forme, de tri, de sélection, d'entrée/sortie, de transformations discrètes de Fourier, d'algèbre linéaire de base, d'opérations statistiques de base, de simulation aléatoire, et de nombreuses autres fonctionnalités essentielles pour les tâches de calcul scientifique. Dans notre cas, cette bibliothèque a été utilisée pour effectuer des transformations de base et concaténer les images capturées par le système stéréoscopique pour les afficher par pairs.

II.4 Interface de développement JupyterLab :

JupiterLab représente l'environnement interactif basé sur le web le plus récent conçu pour le développement de notebooks, de code et de données. Son interface adaptable permet aux utilisateurs de structurer et d'organiser des flux de travail adaptés à la science des données, au calcul scientifique, au journalisme computationnel et à l'apprentissage automatique. Dans notre cas, cette interface a été utilisée pour écrire et exécuter le code python de notre projet.

II.5. Algorithmes de prétraitement et de traitement

II.5.1. Importation des packages

Le programme a besoin des packages suivants :

En *Python*, la version 4.8 d'*OpenCV*, incluant opencv_contrib avec des fonctions stéréo, est référencée sous le nom de "cv2". Ce package englobe :

- Une bibliothèque de traitement d'image,
- Les fonctions de vision stéréo,

La version 1.24.3 de *Numpy* est utilisée pour les opérations matricielles essentielles dans le traitement des images (voir code en annexe figure C.1).

II.5.2. Lecture du flux vidéo

Pour interagir avec les caméras, un processus d'activation est nécessaire. Cela implique d'utiliser la fonction cv2.VideoCapture(), qui déclenche l'activation des deux caméras. Ceci est accompli en désignant les chemins relatifs des caméras dans le système. Par conséquent, deux objets de programme distincts sont instanciés, où chacun est capable d'exploiter les fonctionnalités encapsulées dans les méthodes de la classe cv2.VideoCapture() (voir code en annexe figure C.2)

Pour acquérir une image des caméras, le processus implique l'utilisation de la méthode cv2.VideoCapture().read(). Cette opération produit une image représentant la vue actuelle capturée par la caméra au moment de l'invocation de cette fonction. Pour générer un flux vidéo, il est essentiel d'appeler cette méthode de manière répétée au sein d'une boucle en cours.

Pour améliorer l'efficacité, une approche judicieuse consiste à convertir les images BGR acquises en images en niveaux de gris. Cette tâche est réalisée à l'aide de la fonction cv2.cvtColor()(voir code en annexe figure C.3).

Dans le but d'afficher le flux vidéo sur un écran d'ordinateur, la fonction cv2.imshow() entre en jeu. Cette fonction facilite la création d'une fenêtre dédiée à la présentation du contenu vidéo(voir code en annexe figure C.4) .

Pour sortir de la boucle perpétuelle, une instruction *break* est utilisée. Cette instruction *break* est déclenchée lorsque l'utilisateur appuie sur la barre d'espace. La détection de cet événement de pression de touche est réalisée en utilisant la fonction cv2.waitKey().

Une fois la tâche terminée, il est essentiel de désactiver les deux caméras en utilisant la méthode cv2.VideoCapture(). release(). De plus, toutes les fenêtres actives qui ont été ouvertes doivent être fermées, ce qui est accompli en appliquant la fonction cv2.destroyAllWindows()(voir code en annexe figure C.5)

II.5.3. Géométrie épipolaire

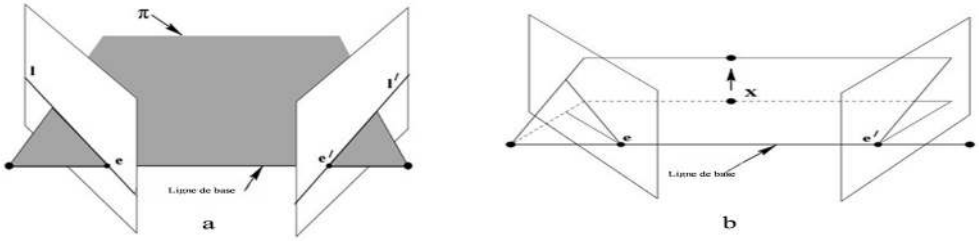

Figure III.6 : Géométrie épipolaire. (a), la ligne de base de la caméra intersecte chaque plan d'image en des points appelés épipôles, notés e et e0. Tout plan Π qui contient la ligne de base est appelé plan épipolaire. Ces plans épipolaires croisent les plans d'image, donnant ainsi naissance aux lignes épipolaires correspondantes l et l'. (b) Lorsque la position du point tridimensionnel X varie, les plans épipolaires subissent une rotation autour de la ligne de base. Cette collection de plans est appelée faisceau épipolaire. Remarquablement, toutes les lignes épipolaires provenant de différents points se croisent à l'épipôle. Cette configuration est fondamentale pour comprendre la géométrie de la vision stéréo et la relation entre les points dans différentes vues [56].

Les concepts de la géométrie épipolaire sont représentés dans la figure III.6, où:
Épipôle : L'épipôle est le point où la ligne reliant les centres des caméras (la ligne de base) intersecte le plan d'image. Alternativement, l'épipôle est l'image dans une vue du centre de la caméra dans l'autre vue. Il sert également de point de fuite pour la direction de la ligne de base (translation).
Plan épipolaire : Un plan épipolaire est un plan qui inclut la ligne de base. Il existe une famille à un paramètre (appelé faisceau) de plans épipolaires.
Ligne épipolaire : Une ligne épipolaire est l'intersection d'un plan épipolaire avec le plan d'image. Toutes les lignes épipolaires convergent à l'épipôle. Un plan épipolaire intersecte les plans d'image de gauche et de droite pour former des lignes épipolaires. Il définit la correspondance entre ces lignes. [55]

II.5.4. Rectification stéréoscopique

L'objectif de la rectification est de projeter les deux images sur un plan commun, en alignant précisément les rangées de pixels. Cet alignement transforme les lignes épipolaires en lignes horizontales, simplifiant la tâche de trouver des points correspondants dans les deux images.

Résultat de cet alignement, huit paramètres sont dérivés dont quatre pour chaque caméra :

1) Un vecteur de distorsion,
2) Une matrice de rotation Rrect, qui est appliquée à l'image,
3) Une matrice de caméra rectifiée Mrect,
4) Une matrice de caméra non rectifiée M,

Pour calculer ces paramètres, deux algorithmes sont couramment utilisés : l'algorithme de *Hartley* et l'algorithme de *Bouguet*. Ces algorithmes jouent un rôle crucial dans le processus de rectification, garantissant que les images sont ajustées pour une correspondance efficace dans la vision stéréo. Grâce à la rectification, le système de vision stéréo est bien préparé pour identifier des points correspondants et percevoir avec précision les informations de profondeur.
Un avantage de l'algorithme de Hartley est sa capacité à effectuer l'étalonnage uniquement sur la base de l'observation des points dans la scène. Cependant, une limitation notable est son absence d'information d'échelle dans les images. Il fournit des informations relatives de distance, ce qui signifie qu'il peut déterminer les relations entre les distances mais pas les distances absolues elles-mêmes. Par conséquent, cette méthode ne permet pas de mesurer précisément la distance exacte d'un objet par rapport aux caméras.
L'algorithme de Bouguet tire parti de la matrice de rotation calculée et du vecteur de translation pour faire pivoter les deux plans projetés de 180 degrés (un demi-tour), les alignant dans le même plan. Cet alignement conduit à des rayons principaux parallèles et à des plans coplanaires.

II.5.5. Calibrage

Au moment de l'exécution du calibrage du module stéréoscopique, les deux caméras sont activées, ce qui entraîne l'ouverture de deux fenêtres permettant à l'utilisateur d'observer la position de l'échiquier (mire de calibrage) dans la paire d'images. L'organigramme expliquant le fonctionnement du programme prendre_images_pour_calibration.ipynb est montré en figure III.7, les images capturées à travers le script "prendre_images_pour_calibration.ipynb" ont pour but de rectifier la distorsion et calibrer les deux caméras.

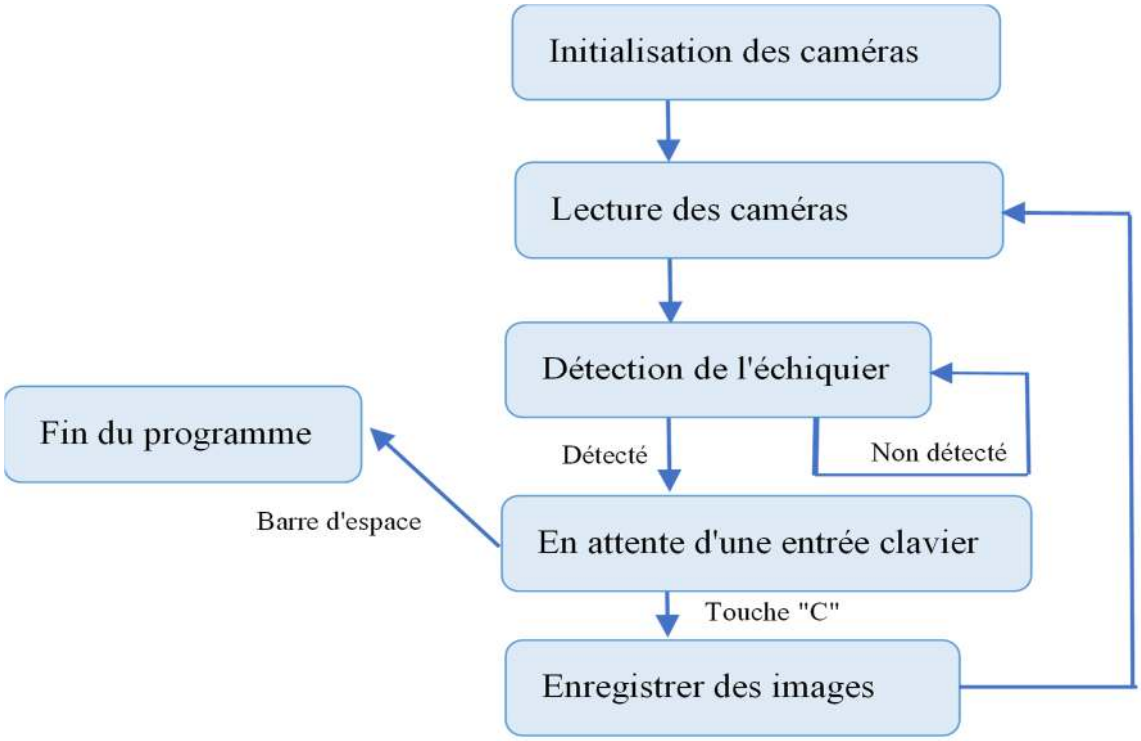

Figure III.7 : Organigramme du programme prendre_images_pour_calibration.ipynb

La fonction cv2.findChessboardCorners() détectera un nombre prédéfini de coins sur un échiquier, et les vecteurs suivants seront générés :

- imgpointsD : Ces vecteurs contiennent les coordonnées des coins dans l'image de droite (dans l'espace de l'image).
- imgpointsG : Ces vecteurs contiennent les coordonnées des coins dans l'image de gauche (dans l'espace de l'image).
- objpoints : Ces vecteurs renferment les coordonnées des coins dans l'espace objet.

Afin d'améliorer la précision des coordonnées des coins repérés, la fonction cv2.cornerSubPix() est appliquée.

Une fois que le programme a reconnu la position des coins de l'échiquier sur les deux images, deux nouvelles fenêtres s'ouvrent où l'utilisateur peut évaluer les images capturées. Si les images ne sont pas floues et semblent bonnes, il peut appuyer sur la touche « s » (Enregistrer) pour enregistrer les images. Dans le cas contraire, l'utilisateur peut appuyer sur la touche « c » (Annuler) pour que les images ne soient pas enregistrées.

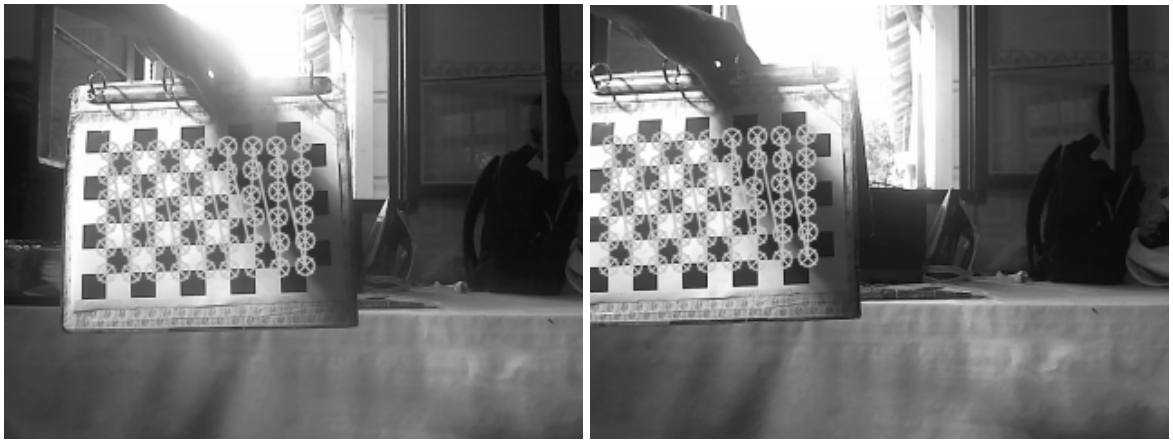

Figure III.8 : Reconnaissance des coins de l'échiquier

Ce processus de calibrage repose sur les données collectées par le script "prendre_images_pour_calibration.ipynb", où les coordonnées des coins de l'échiquier sont stockées dans Imgpoints et Objpoints. La fonction cv2.calibrateCamera() est utilisée pour obtenir des matrices de caméra mises à jour (représentant la transformation d'un point tridimensionnel du monde en une image bidimensionnelle), des coefficients de distorsion, ainsi que des vecteurs de rotation et de translation pour chaque caméra individuelle. Ces paramètres sont ensuite utilisés pour corriger la distorsion dans la sortie de chaque caméra. Pour obtenir des matrices de caméra optimales pour chaque dispositif, la fonction cv2.getOptimalNewCameraMatrix() entre en jeu, améliorant la précision (voir code en annexe figure C.6)

Dans le processus de calibration stéréoscopique, nous ferons appel à la fonction cv2.StereoCalibrate(). Cette fonction calcule la relation de transformation entre les deux caméras, l'une servant de référence pour l'autre.

Par la suite, la fonction cv2.stereoRectify() est utilisée pour aligner les lignes épipolaires des deux caméras sur un plan commun. Cette transformation simplifie la tâche effectuée par la fonction chargée de générer la carte de disparités, car le bloc de correspondance n'est requis que dans une seule dimension. De plus, cette fonction fournit la matrice essentielle et la matrice fondamentale, qui sont essentielles pour les étapes ultérieures

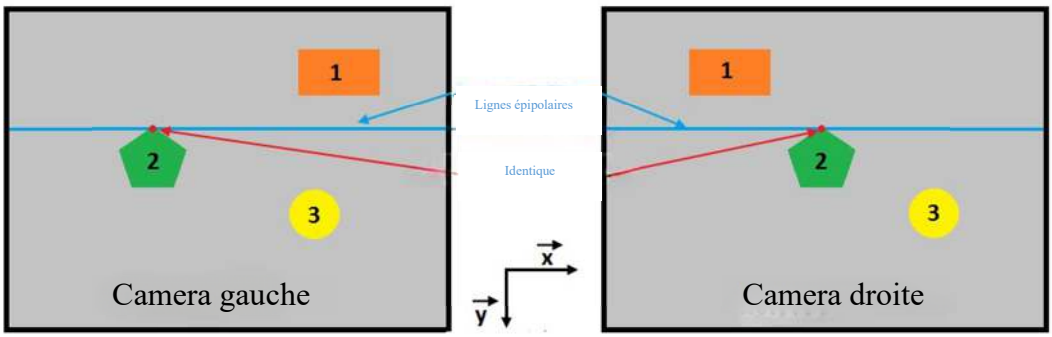

Figure III.9 : Calibrage de la distorsion

La méthode cv2.initUndistortRectifyMap() permet d'obtenir une image dépourvue de distorsion, laquelle est ensuite employée pour calculer la carte des disparités.

L'image sans distorsion et avec distorsion est montrée dans la figure III.10 et III.11.

Figure III.10 : Image avec distorsion (non calibré)

Figure III.11 : Image sans distorsion (calibrée)

II.6. Déroulement du programme de génération de carte de disparité et de calcul de distances

Au démarrage du programme « prog_Stereo_Vision_princip.ipynb », les caméras subissent d'abord une calibration individuelle pour corriger toute distorsion. Ensuite, un processus de calibration stéréoscopique est réalisé pour éliminer les effets de rotation et aligner les lignes épipolaires. Dans une boucle continue, les images sont traitées pour générer une carte de disparité.

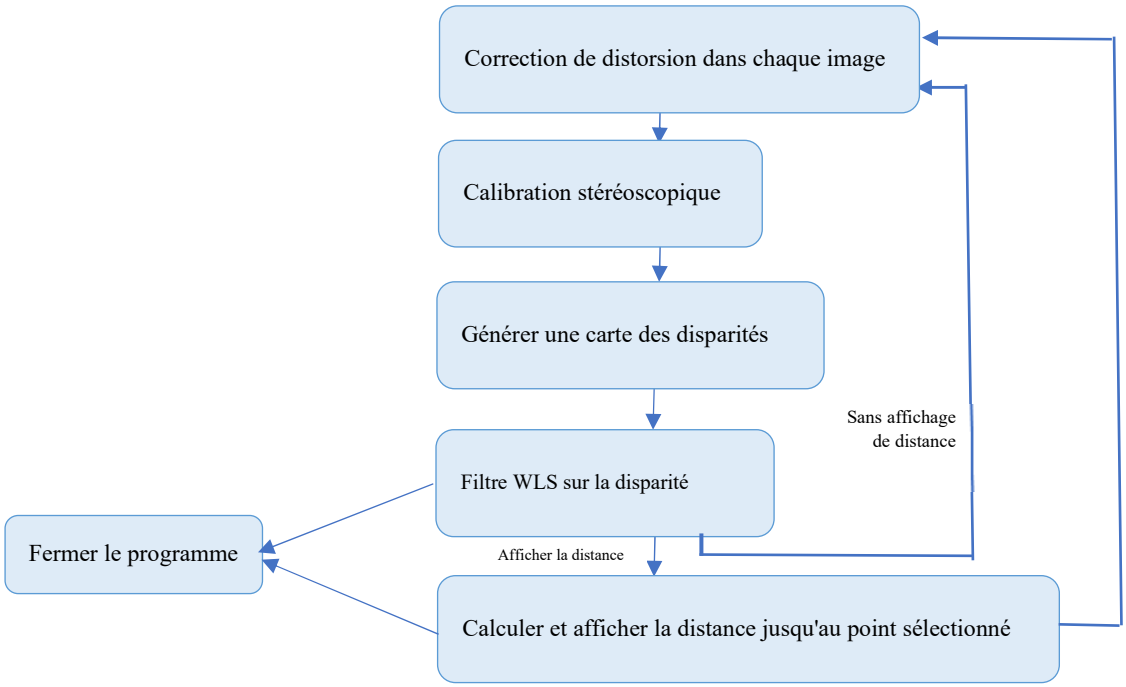

Figure III.12 : Fonctionnement du programme « prog_Stereo_Vision_princip.ipynb»

III. Algorithmes de traitement

III.1. Calcul de la carte de disparité

Afin de générer la carte des disparités, on instancie un objet StereoSGBM en utilisant la fonction cv2.StereoSGBM_create(). Cette classe emploie un algorithme d'appariement semi-global (Hirschmüller, 2008) pour établir des correspondances stéréo entre les images capturées par les caméras droite et gauche.
Les dimensions des blocs sont spécifiées dans les paramètres d'entrée, où ces blocs prennent la place des pixels lorsque leur taille dépasse 1. L'objet SGBM ainsi créé compare les blocs d'une image de référence avec ceux d'une image correspondante. Par exemple, dans un étalonnage stéréo correct, un bloc de la quatrième rangée de l'image de référence est apparié avec tous les blocs de l'image correspondante se trouvant également sur la quatrième rangée. Cette approche optimise le calcul de la carte des disparités (voir code en annexe figure C.7).

En utilisant les paramètres que nous avons définis lors de l'initialisation, le résultat obtenu pour la carte de disparité est montré en figure III.13.

Figure III.13 : Résultat pour la carte des disparités

Dans cette carte de disparité, il subsiste un niveau notable de bruit. Pour réduire cet effet, un filtre morphologique est appliqué. Plus précisément, un filtre de fermeture ("Closing Filter") est employé à l'aide de la fonction OpenCV cv2.morphologyEx(cv2.MORPH_CLOSE) pour éliminer les petits points noirs indésirables, la figure III.14 montre la carte de disparité filtrée avec le filtre de fermeture.

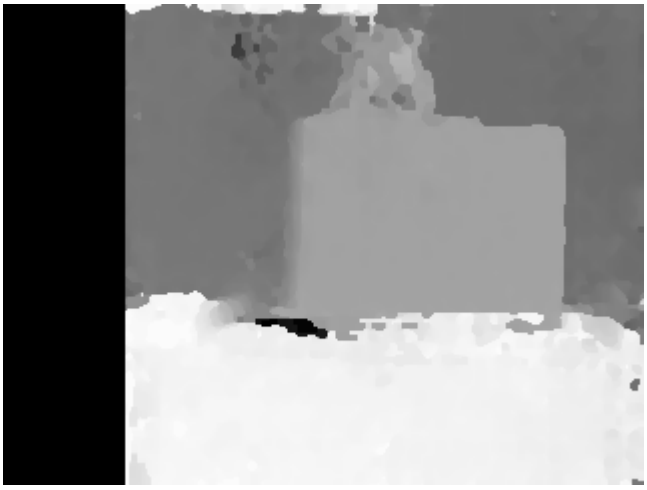

Figure III.14 : Résultat pour la carte de disparité après un filtre de fermeture

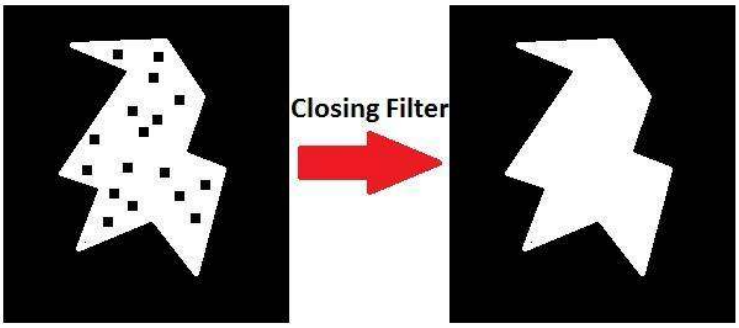

Figure III.15 : Exemple de filtre de fermeture

Le résultat est satisfaisant, cependant, la visibilité des contours des objets demeure compliquée en raison du bruit. Par conséquent, un filtre WLS (Weighted Least Squares) est mis en œuvre. En amont, les paramètres du filtre doivent être définis durant l'initialisation (voir code en annexe figure C.8).
Le paramètre Lambda est généralement fixé à 8000, et à mesure que sa valeur augmente, les contours de la carte des disparités se rapprochent davantage des contours de l'image de référence. Nous l'avons

ajusté à 80000, car cette valeur a produit de meilleurs résultats dans notre cas. Le paramètre Sigma quantifie la précision du filtre sur les contours des objets.
Ensuite, une autre instance de la classe StereoSGBM est instanciée avec la fonction cv2.ximgproc.createRightMatcher(), et elle est basée sur la première instance. Ces deux instances sont ensuite employées dans le filtre WLS pour créer une carte de disparité améliorée.
L'étape suivante consiste à créer une instance du filtre à l'aide de la fonction cv2.ximgproc.createDisparityWLSFilter() (voir code en annexe figure C.9)
Ensuite, pour appliquer l'instance du filtre WLS, on utilise la méthode cv2.ximgproc.createRightMatcher().filter(). Par la suite, les valeurs de notre filtre sont soumises à une normalisation en utilisant la fonction cv2.normalize()(voir code en annexe figure C.10).
Afin d'améliorer la visualisation, une COLORMAP Ocean a été appliquée à l'aide de la fonction cv2.applyColorMap(). Dans cette coloration, plus la teinte est sombre, plus l'objet concerné est éloigné de la caméra stéréo.

Figure III.16 : ColorMap Ocean

Cela aboutit à la création d'une image qui réussit à mettre en évidence les contours, bien qu'au détriment de la précision, et les valeurs précises de disparité ne sont plus conservées (étant donné que la carte de disparité est stockée en tant que float32, tandis que le résultat du filtre WLS est en uint8).
Pour garantir l'utilisation des valeurs appropriées de disparité en vue de mesurer ultérieurement les distances par rapport aux objets, les coordonnées x et y de l'image filtrée par le filtre WLS sont extraites à l'aide d'un double-clic. Ces coordonnées (x, y) sont ensuite employées pour extraire la valeur de disparité de la carte des disparités, ce qui permet la mesure des distances. La valeur renvoyée correspond à la moyenne des valeurs de disparité au sein d'une matrice de pixels de 9x9.
Une fois que la carte de disparités a été créée, il est nécessaire de calculer les distances correspondantes. Cette étape implique l'établissement d'une relation entre les valeurs de disparité et les distances réelles. Pour accomplir cela, des mesures expérimentales des valeurs de disparité ont été effectuées en divers points, dans le but de déterminer une régression appropriée.

Figure III.17 : Application du filtre WLS à une carte de disparités.

Cela conduit à la création d'une image qui parvient à mettre en évidence les contours, mais qui sacrifie une partie de sa précision et n'incorpore plus les valeurs précises de disparité (étant donné que la carte

des disparités est encodée en float32 tandis que le résultat du filtre WLS est encodé en uint8).
Afin d'assurer l'utilisation des valeurs appropriées de disparité pour mesurer ultérieurement les distances par rapport aux objets, les coordonnées x et y de l'image filtrée par le filtre WLS sont capturées via un double clic. Par la suite, ces coordonnées (x, y) sont employées pour extraire la valeur de disparité depuis la carte des disparités, permettant ainsi la mesure de la distance. La valeur retournée correspond à la moyenne des disparités au sein d'une matrice de pixels de 9x9.

IV. Mesure de distances :

Une fois la carte de disparités créée, il devient nécessaire de calculer les distances correspondantes. Cette tâche implique d'établir une corrélation entre la valeur de disparité et la distance réelle. Pour ce faire, nous avons effectué des mesures expérimentales des valeurs de disparité en différents points, afin de déterminer une régression appropriée. La figure III.18 représente la disparité en fonction de la distance et l'équation de la droite de régression est montrée en figure III.19

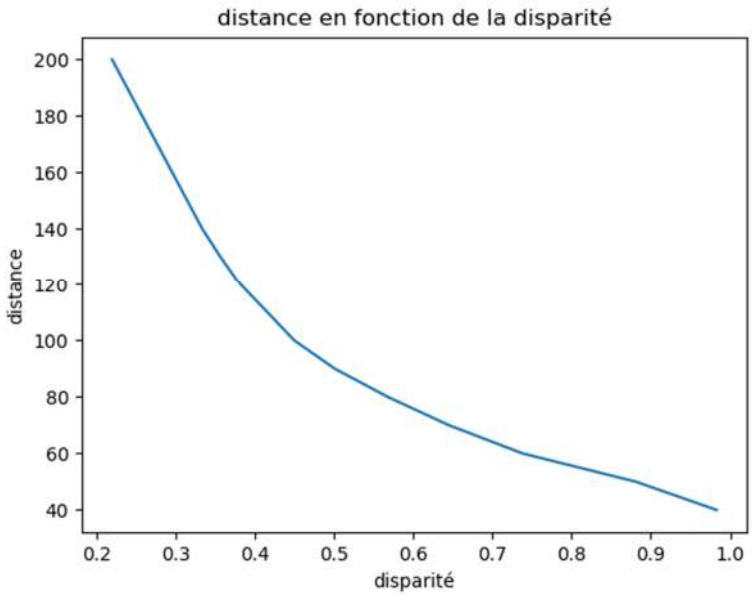

Figure III.18 : Distance en fonction de la disparité

```
# Equation pour la mesure de distance
Distance= -637.58*average**(3) + 1475.09*average**(2) -1202.68*average + 401.76
```

Figure III.19 : formule de régression

Pour obtenir cette équation linéaire de régression, la fonction np.polyfit a été employée (voir code en annexe figure C.11). Il est important de noter que la mesure de distance est fiable dans la plage allant de 40 cm à 200 cm, afin d'obtenir des résultats satisfaisants. La précision de la mesure dépend également de la qualité de l'étalonnage. Dans notre travail avec le système stéréo, nous sommes en mesure de mesurer la distance jusqu'à un objet avec une précision d'environ +/- 5 cm.

V. Discussions des résultats

V.1. Triangulation :

Pour reconstruire la scène en 3D, les informations obtenues par le système stéréoscopique doivent être utilisées.

Premièrement, les points de correspondance entre les deux images doivent être trouvés. Pour accomplir cela, l'algorithme SIFT a été utilisé. Ces points peuvent être notés "x" pour la caméra gauche et "x'" pour la caméra droite, ou alternativement, ils peuvent être exprimés en utilisant le format de pixel (ul, vl), (ur, vr).

Le point d'intersection où les rayons de la caméra passent par leurs points de pixel respectifs est l'endroit où se situe le point de la scène, correspondant aux deux points de l'image. Ainsi pour obtenir une projection en 3d les équations suivantes doivent être utilisé :

$$X = B\frac{(u_l - c_x)}{(u_l - u_r)} (1)$$

$$Y = B\frac{f_x(v_l - c_x)}{f_y(u_l - u_r)} (2)$$

$$Z = B\frac{f_x}{(u_l - u_r)} (3)$$

Ou X, Y, Z sont les points obtenus en 3d et B est la distance entre le centre des deux caméras, fx et fy sont les distances focales obtenu horizontalement et verticalement, cx et cy sont les points principaux selon l'axe des x et y respectivement [63].

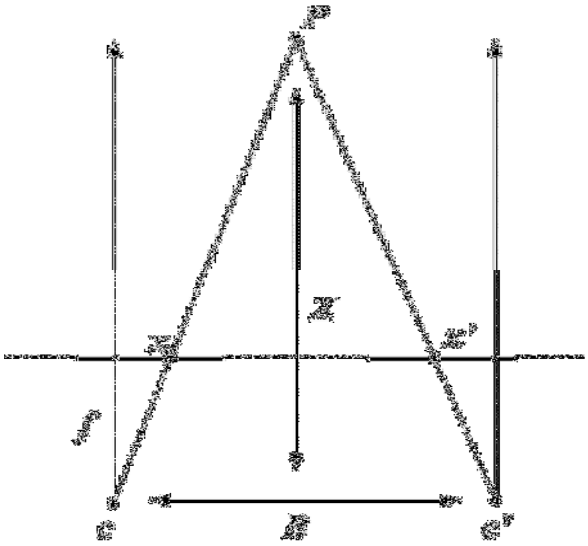

Figure III.20 : Système stéréo simple [63]

L'équation pour la profondeur Z et la Figure III.19 illustrent que la profondeur Z est inversement liée à la disparité. Cela implique que la disparité sera plus grande lorsque le point de la scène est en proximité rapprochée des deux caméras et diminue à mesure qu'il s'éloigne des caméras. À mesure que le point de la scène approche d'une profondeur infinie, la disparité converge vers zéro, indiquant qu'il n'y a aucune différence discernable dans la position du point dans les images.

De même, la disparité est directement proportionnelle à la ligne de base B. Lorsque la ligne de base augmente, la différence, et par conséquent, la disparité entre les deux images augmente également

[63].

V.2 Représentation de scènes en 3D :

Dans le but d'obtenir une représentation en 3D d'une scène à partir d'une paire d'image, nous avons tout d'abord utilisé l'algorithme SIFT pour la mise en correspondance des points entre une paire d'images stéréoscopique (figure III.21).

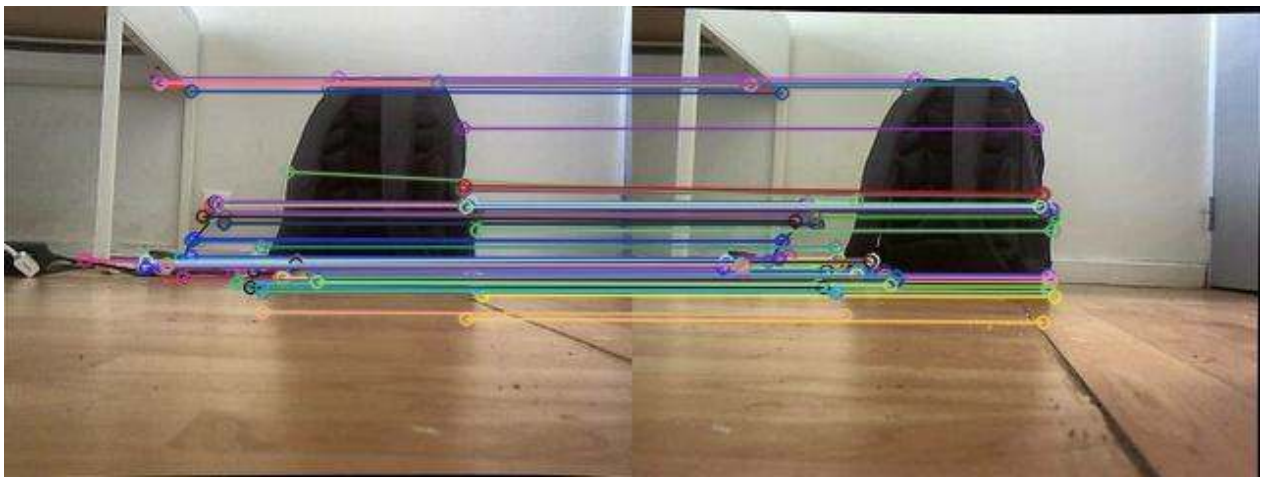

Figure III.21 : Pair d'image apparié (avec calibrage stéréo)

Les points appariés dans l'image gauche et de l'image droite sont montrés dans les figures III.22 et III.23.

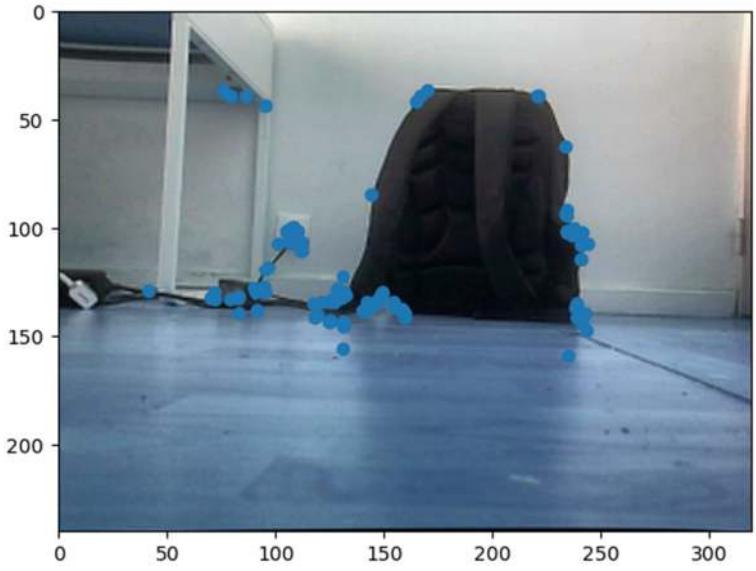

Figure III.22 : Point apparié dans l'image gauche

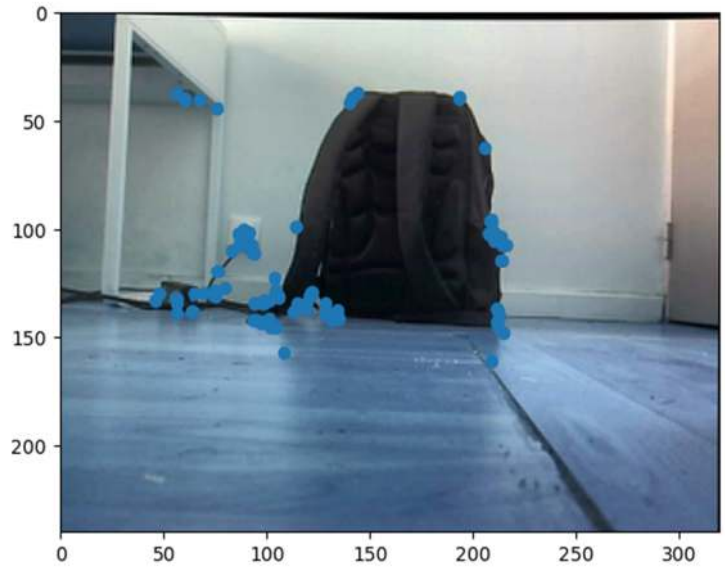

Figure III.23 : Points appariés dans l'image droite

A partir des coordonnées des points appariés et en utilisant les équations 1,2 et 3, nous avons pu obtenir leurs coordonnées en 3D. Les points 3D obtenus à partir de la paire d'image appariée ci-dessus est montrée dans la figure III.24.

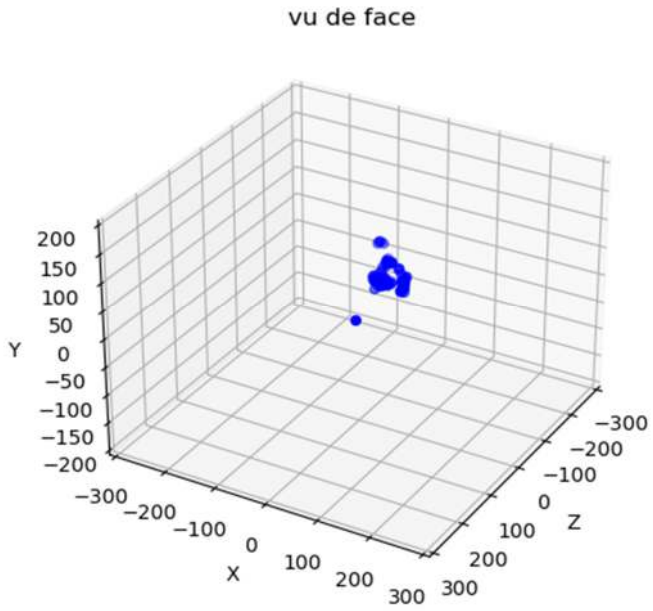

Figure III.24 : Points 3D obtenus à partir des points 2D appariés

En comparant la figure III.22 et III.24 nous pouvons déduire visuellement que la représentation en 3D des points est une représentation exacte de la scène. Ces derniers peuvent être utilisés pour donner une information 3D au malvoyant puisque ce dernier se situe au milieu du repère 3D, c'est-à-dire, centre du repère du système stéréoscopique.
Dans le but d'obtenir une meilleure compréhension de position des points 3D dans la scène réelle, nous avons effectué des projections sur les trois plans : Vue de face, vue de profil et Vu d'en haut. Ces vues sont représentées dans les figures III.25, III.26 et III.27 respectivement.

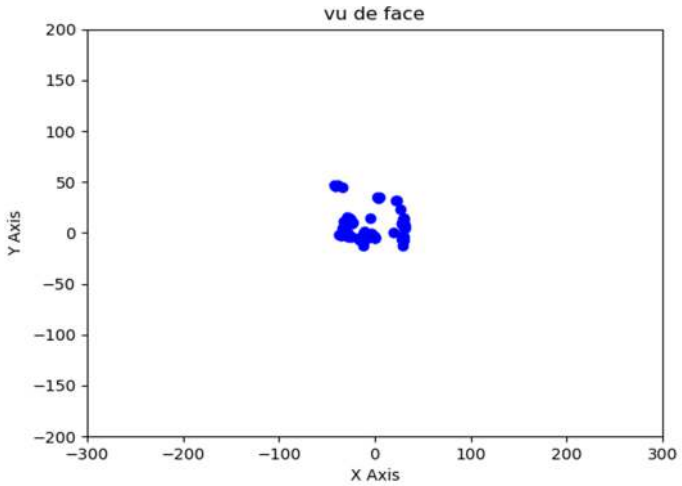

Figure III.25 : Vue de face

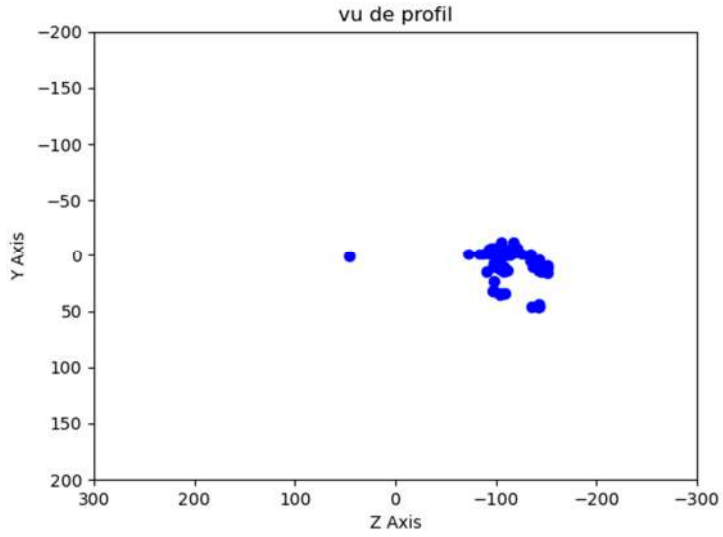

Figure III.26: Vu de profil

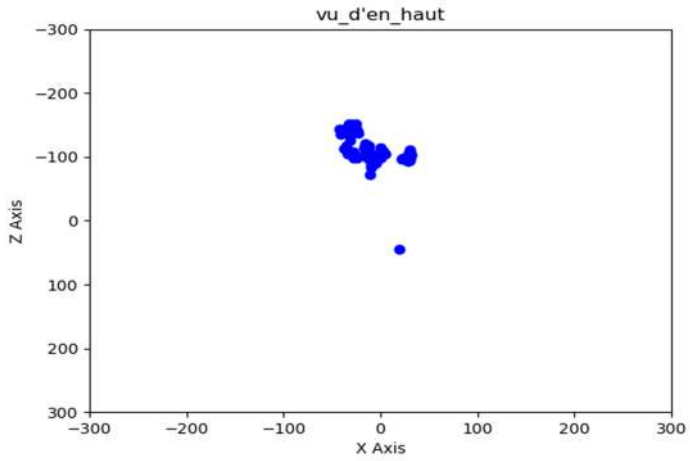

Figure III.27 : Vu d'en haut

Dans le but de tester notre système nous avons pris des pairs d'images représentant des scènes d'intérieur et d'extérieur :

V.3. Représentation en 3D de scène d'extérieur et d'intérieur

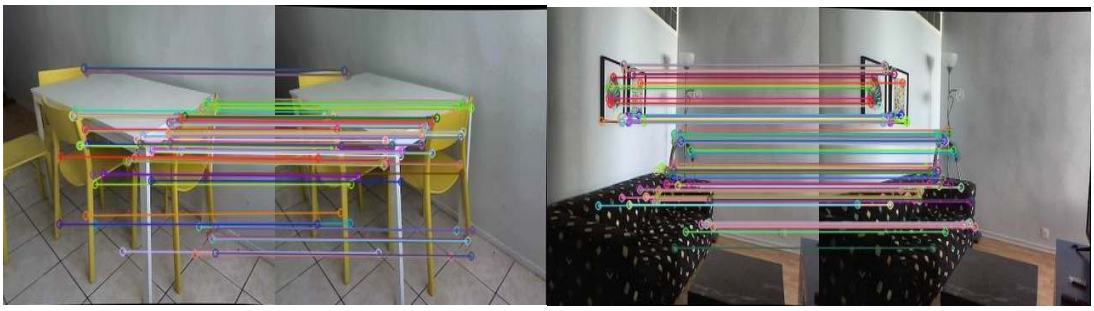

Figure III.28 : pair d'image apparié (intérieur)

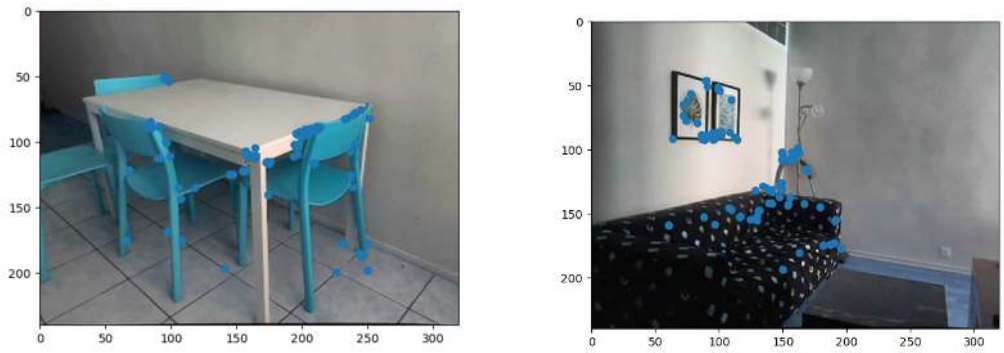

Figure III.29 : Point apparié dans l'image gauche(intérieur)

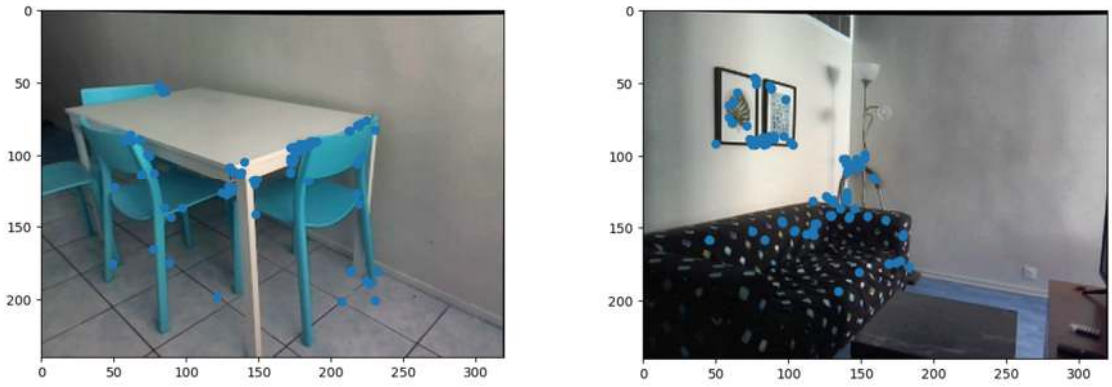

Figure III.30 : Point apparié dans l'image gauche(intérieur)

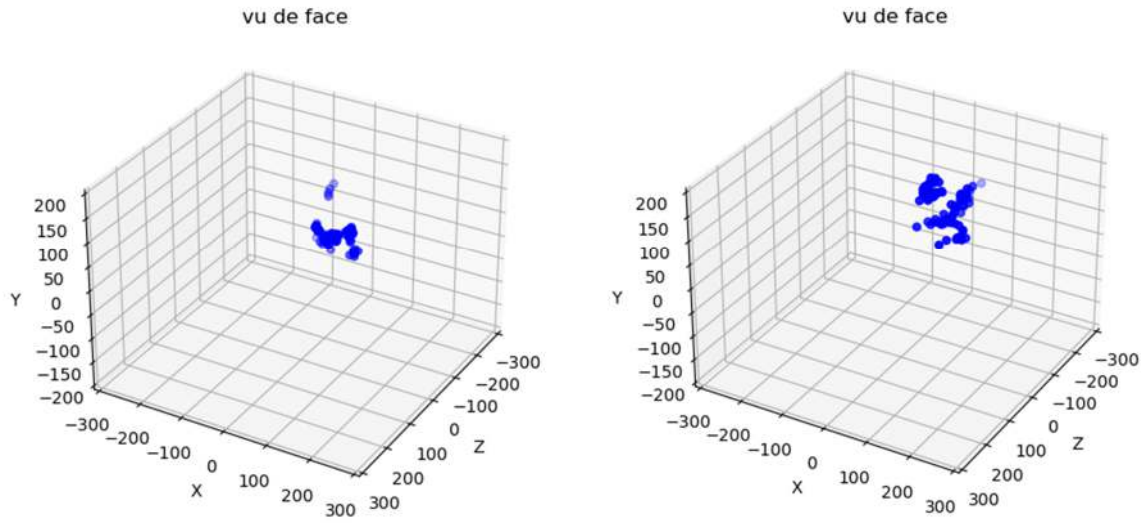

Figure III.31 : Affichage 3D obtenu à partir des points appariés(intérieur)

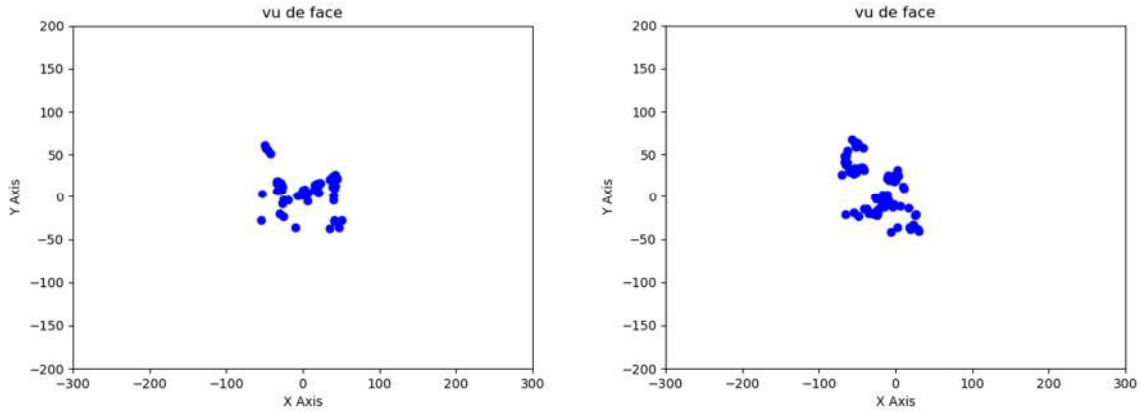

Figure III.32 : Vue de face(intérieur)

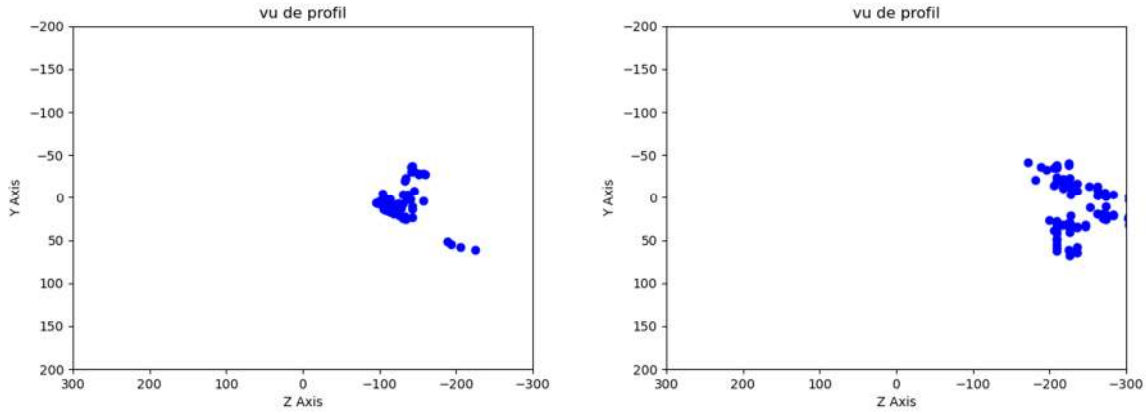

Figure III.33: Vu de profil (intérieur)

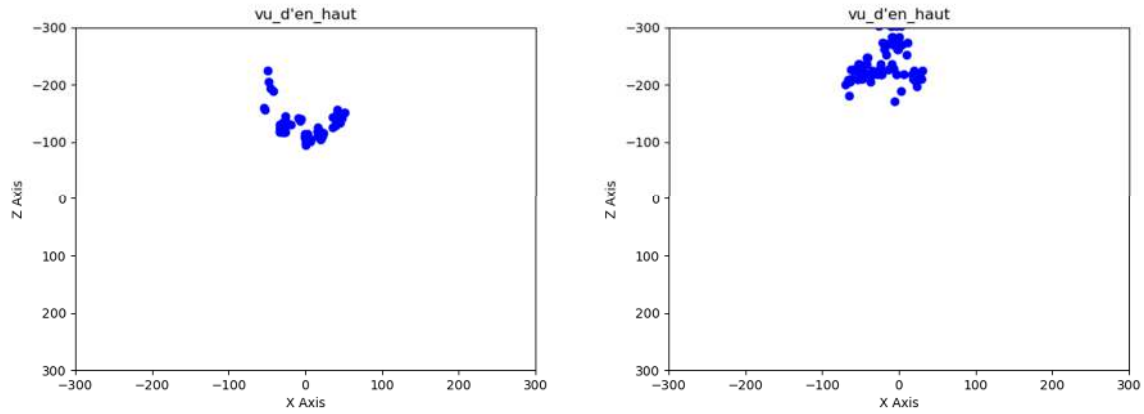

Figure III.34: Vu d'en haut(intérieur)

Figure III.35 : Pair d'image apparié (extérieur)

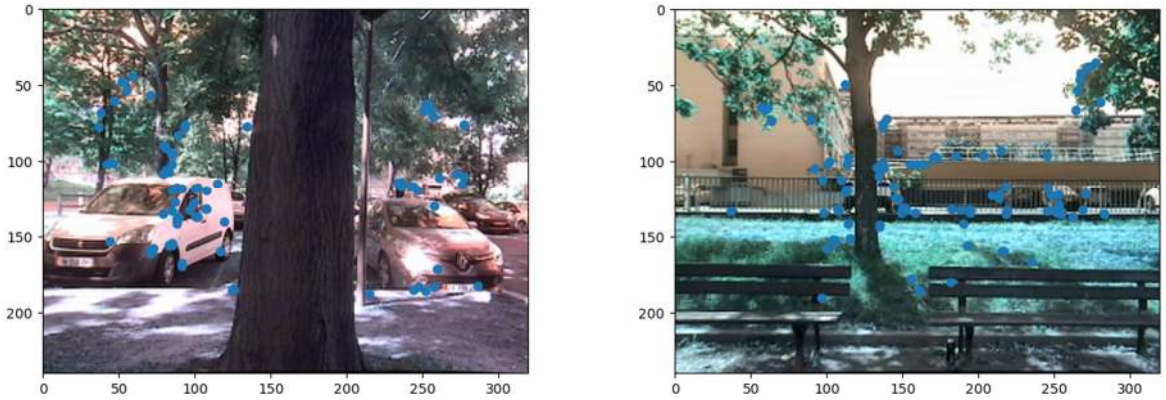

Figure III.36: Point apparié dans l'image gauche(extérieur)

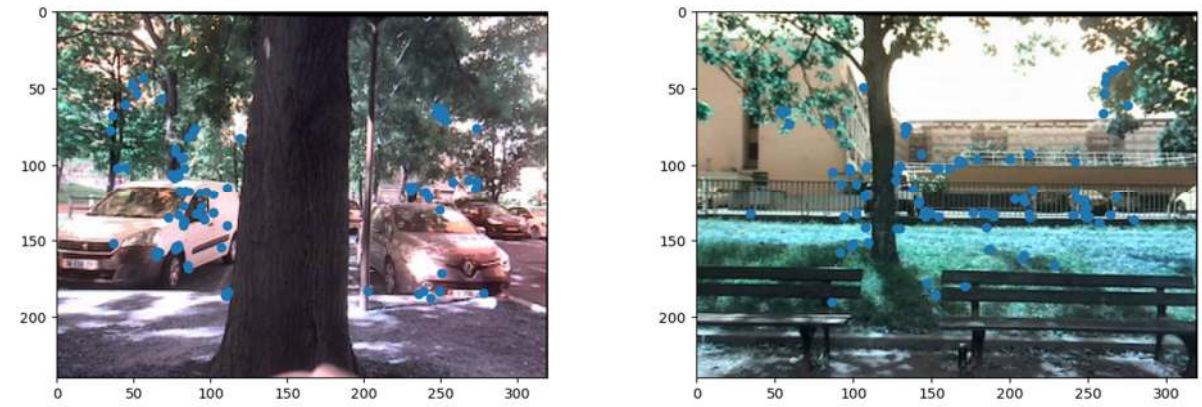

Figure III.37 : : Point apparié dans l'image droite(extérieur)

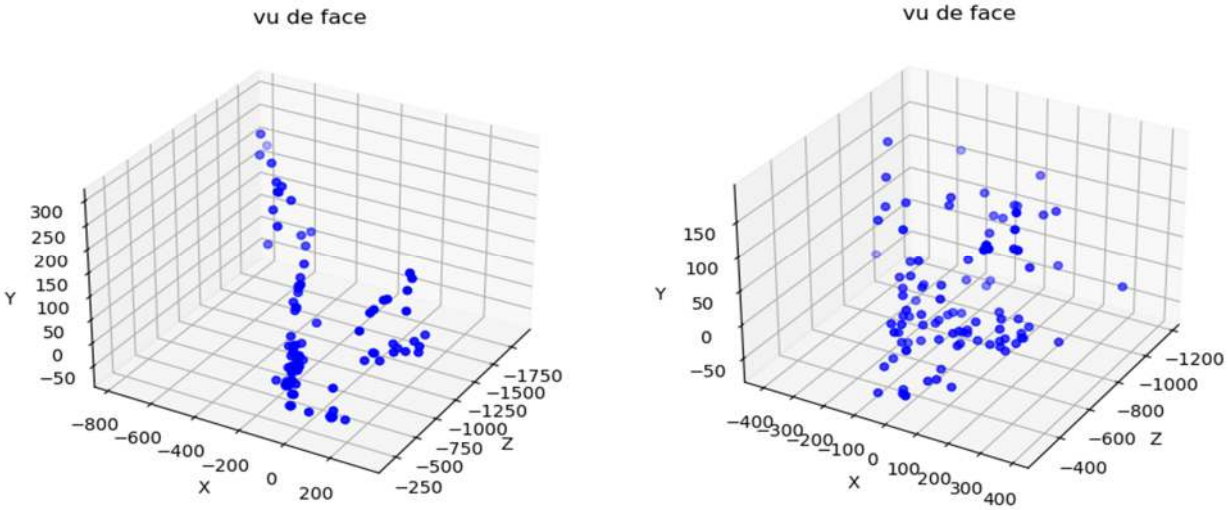

Figure III.38 : Affichage 3D obtenu à partir des points appariés (extérieur)

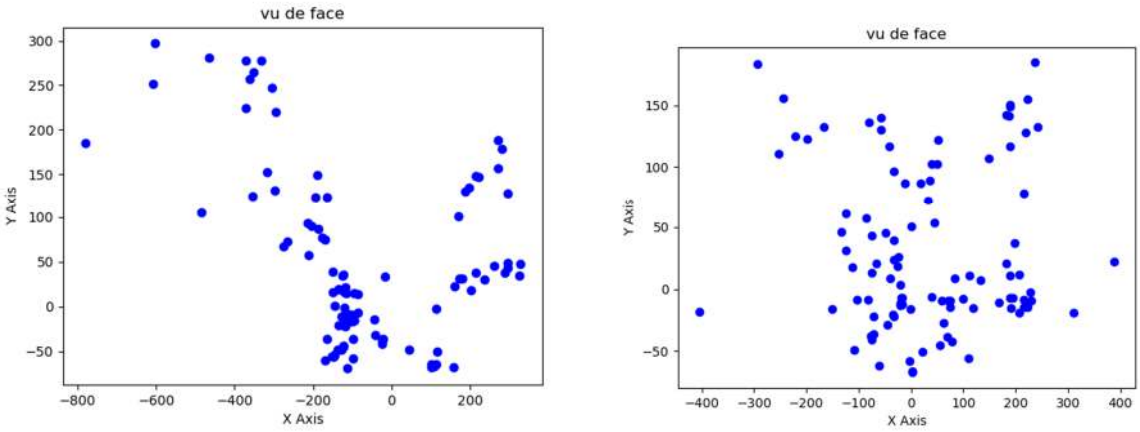

Figure III.39 : Vue de face (extérieure)

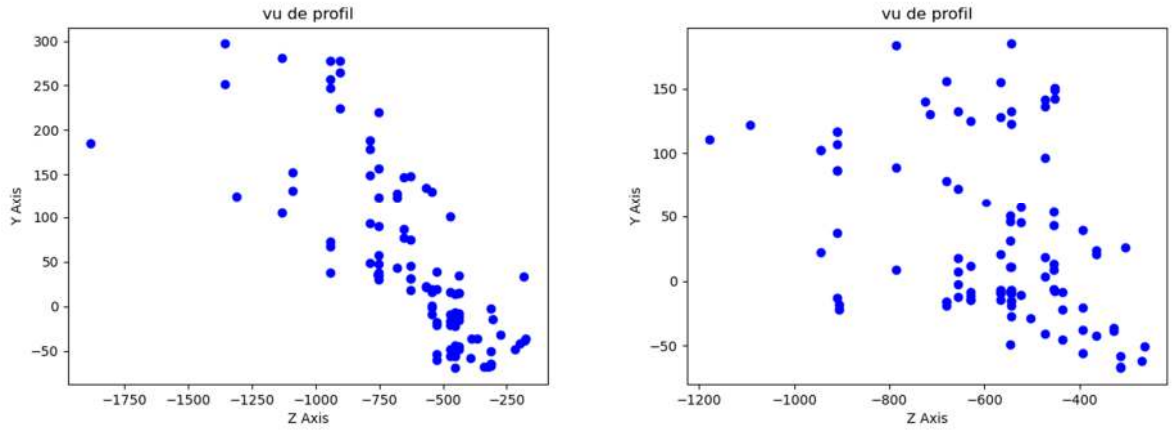

Figure III.40 : Vue de profil (extérieure)

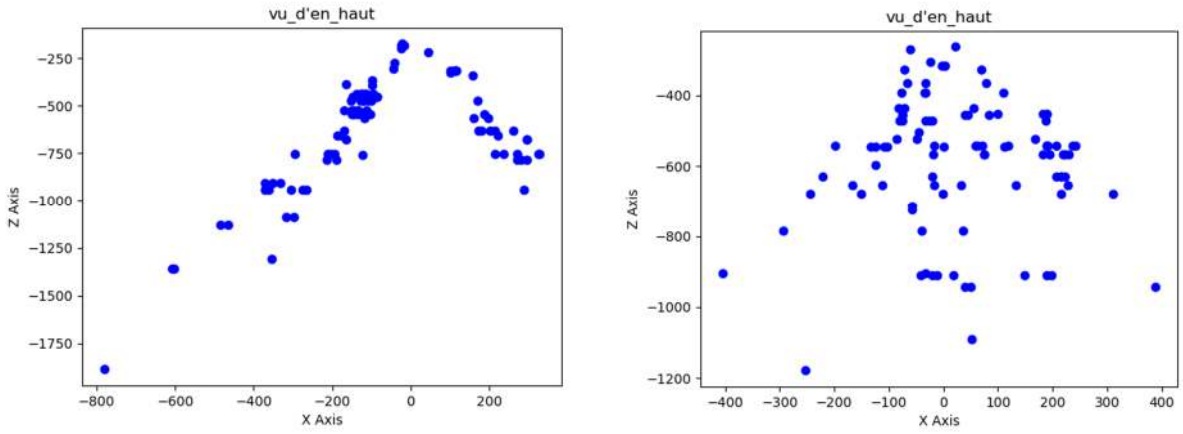

Figure III.41 : Vue d'en haut (extérieur)

En comparant la reconstruction 3D ainsi que les distances estimées par la triangulation (axe z) des vues d'extérieur et d'intérieur, il est possible de conclure visuellement et expérimentalement que notre système nous donne de bons résultats et une bonne estimation de distance.

V.4. Comparaison de distance :

Dans le but de comparer les distances mesurées en temps réels et ceux obtenues par la triangulation, nous avons pris des scènes d'intérieur et d'extérieur et nous avons pris comme référence pour la mesure de distance en temps réel les points appariés de l'image gauche. Les séries d'images avec les points appariés en intérieur et en extérieur sont montrés en figure III.42 et figure III.43. Les tableaux III.1 et III.2 présentent les différentes valeurs de distance obtenues pour chaque scène d'intérieure et d'extérieur respectivement.

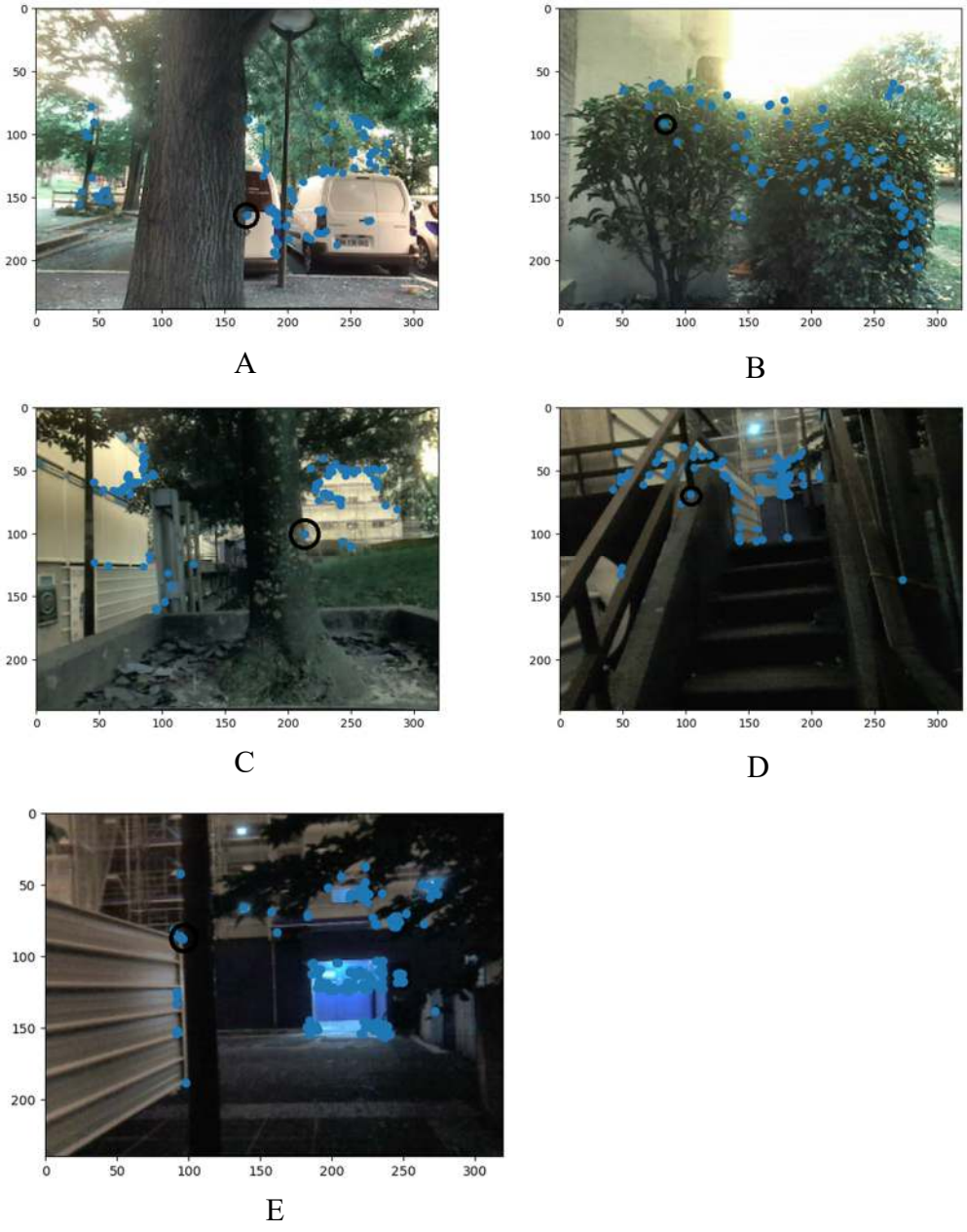

Figure III.42 : Image avec points apparié(extérieure)

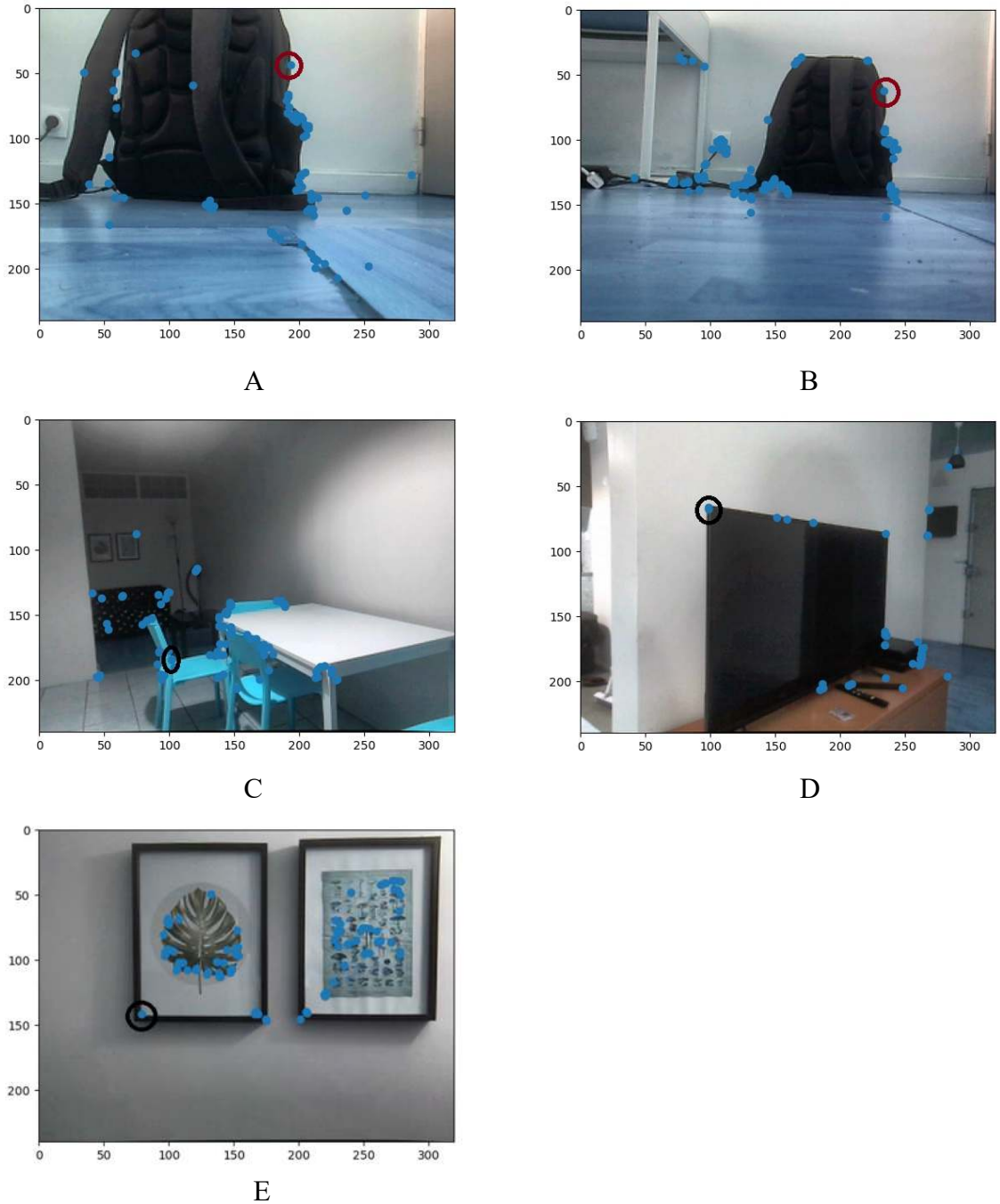

Figure III.43 : Image avec points apparié(intérieure)

	A	B	C	D	E
Distance mesuré	194	158	113	192	121
Distance par triangulation	188	161	109	189	126

Tableau III.1 : Comparaison entre la distance mesurée et celle obtenue par la triangulation (scène d'extérieur)

	A	B	C	D	E
Distance mesuré	66	100	197	85	87
Distance par triangulation	72	98	192	83	83

Tableau III.2 : Comparaison entre la distance mesurée et celle obtenue par la triangulation (scène d'intérieur).

En comparant les distances réelles et celles obtenues par triangulation, il est possible d'affirmer que les distances obtenues par la triangulation et les distances réelles se rapprochent. Donc, l'algorithme de triangulation donne des valeurs qui sont proches de la réalité.

VI. Tests et discussions :

Pour tester notre système, nous avons utilisé les algorithmes SIFT et la triangulation pour estimer en temps réel la distance entre la caméra à un point particulier (détecté), ainsi que la bibliothèque playsound pour générer un son si les valeurs de distance obtenues par triangulation sont inférieures à 70 mètres.
Nous avons testé notre système stéréoscopique dans un environnement intérieur et extérieur. Durant ce test, nous avons vu que le système réagit en temps réel en générant un son à l'approche de murs et d'Objets texturés.

VII. Conclusion :

Dans ce chapitre, nous avons présenté la conception et la réalisation de notre système de substitution basé sur les principes de la vision artificielle. Le système de substitution proposé se compose de deux caméras RGB pour capturer l'information visuelle de l'environnement proche au malvoyant. Cette dernière est transmise vers la carte Raspberry Pi pour être traitée puis présentée au malvoyant sous la forme d'un avertissement sonore.

Nous avons effectué une opération de rectification stéréoscopique, ensuite, les deux caméras ont été calibrées et les lignes épipolaires des deux caméras ont été alignées sur un plan commun. La carte de disparité a été obtenue à partir d'une image stéréoscopique sans distorsions.

Cependant, la carte de disparité contient du bruit. Pour remédier à cela, nous avons utilisé un filtre de fermeture pour éliminer les petits points noirs indésirables, nous avons ensuite employé le filtre WLS pour créer une carte de disparités meilleure et lisse.

Dans le but de déterminer la relation entre la disparité et la distance réelle, nous avons effectué une mesure expérimentale de distance. Nous avons mesuré la distance entre la caméra et un objet réel puis nous avons fait la correspondance de cette distance avec la disparité calculée, ceci, nous a conduit à obtenir une droite de régression.

Par la suite, nous avons reconstruit les points 3D des scènes d'intérieur et d'extérieur en utilisant l'algorithme SIFT pour la détection des points d'intérêt dans la paire stéréoscopique et l'algorithme de triangulation. Ensuite, nous avons fait une comparaison entre les distances obtenues des deux scènes d'intérieur et d'extérieur. Cette comparaison nous a permis de déduire que les distances mesurées et celles obtenues par la triangulation sont presque identiques.

Conclusion générale

Conclusion générale

Les troubles visuels peuvent être définis comme la cécité ou la perte de vision. Cette déficience entraîne de nombreuses difficultés dans les activités quotidiennes des personnes malvoyantes telles que la lecture, la marche, les interactions sociales et la conduite. Un grand nombre d'individus souffrant de déficiences visuelles existent à travers le monde, et de nombreuses études ont été menées pour développer une assistance technologique pour eux.

Dans ce travail, nous avons tenté de réaliser un système de substitution basé sur la vision artificielle en utilisant deux caméras RGB et une carte Raspberry Pi. Nous avons aussi utilisé des bibliothèques dédiées et le langage de programmation Python ainsi que le système d'exploitation de Raspberry Pi. Nous avons effectué une opération de rectification et de calibrage du système stéréoscopique pour l'alignement des lignes épipolaires pour générer la carte de disparité. Cette dernière, est ensuite filtrée en utilisant le filtre WLS. Nous avons appliqué l'algorithme SIFT aux images stéréoscopiques rectifiées pour détecter les points d'intérêt et faire la reconstruction 3D des points extraits et appariés et obtenir ainsi les valeurs de distance souhaitées par la triangulation. L'expérimentation a montré que les valeurs réelles et celles mesurées sont presque identiques.

Limitation :

- Notre système de vision ne détecte pas les obstacles bas et de couleur uniforme ainsi que les murs (régions non texturées),
- Selon l'incidence des rayons lumineux, certains points ne sont pas détectés par l'algorithme SIFT.,

Comme perspective et suite à ce travail :

- Possibilité d'entraîner et optimiser un modèle de détection d'Object yolov 3 pour détecter les objets et leur position relative par rapport aux deux caméras RGB.
- Utilisation d'une carte Raspberry Pi plus récente et plus performante.
- Essayer d'utiliser un langage de programmation bas niveau (C++) pour optimiser la performance totale de notre système.
- Possibilité d'émettre des sons ou autres types d'indications (vibrations par exemple) au malvoyant pour l'orienter dans son environnement.

Ce projet m'a permis d'apprendre les différentes techniques et approches pour le développement des systèmes de substitution et approfondir mes connaissances dans le domaine de la vision par ordinateur.

Le projet m'a aussi permis d'intégrer une équipe de recherche et m'initier à la recherche, il m'a aussi permis d'améliorer mon habilité au travail d'équipe et à la collaboration. Cette expérience humaine sera un atout qui me permettra de m'intégrer plus facilement dans un environnement professionnel

RÉFÉRENCES

[1]. Motshoane, K., Tu, C., & Owolawi, P. A. (2017, December). An overview of vision substitution system for the visually impaired. In 2017 3rd IEEE International Conference on Computer and Communications (ICCC) (pp. 2802-2806). IEEE.

[2]. Cardillo, E., Di Mattia, V., Manfredi, G., Russo, P., De Leo, A., Caddemi, A., & Cerri, G. (2018). An electromagnetic sensor prototype to assist visually impaired and blind people in autonomous walking. IEEE Sensors Journal, 18(6), 2568-2576.

[3]. Ganesan, J., Azar, A. T., Alsenan, S., Kamal, N. A., Qureshi, B., & Hassanien, A. E. (2022). Deep Learning Reader for Visually Impaired. Electronics, 11(20), 3335.

[4]. Patil, K., Jawadwala, Q., & Shu, F. C. (2018). Design and construction of electronic aid for visually impaired people. *IEEE Transactions on Human-Machine Systems*, *48*(2), 172-182.

[5]. Budilaksono, S., Bertino, B., Suwartane, I. G. A., Rosadi, A., Suwarno, M. A., Purtiningrum, S. W., ... & Riyadi, A. A. (2020, February). Designing an ultrasonic sensor stick prototype for blind people. In Journal of Physics: Conference Series (Vol. 1471, No. 1, p. 012020). IOP Publishing.

[6]. Bai, J., Lian, S., Liu, Z., Wang, K., & Liu, D. (2017). Smart guiding glasses for visually impaired people in indoor environment. *IEEE Transactions on Consumer Electronics*, *63*(3), 258-266.

[7]. Hoffmann, R., Spagnol, S., Kristjánsson, Á., & Unnthorsson, R. (2018). Evaluation of an audio-haptic sensory substitution device for enhancing spatial awareness for the visually impaired. *Optometry and Vision Science*, *95*(9), 757.

[8]. Bai, J., Lian, S., Liu, Z., Wang, K., & Liu, D. (2018). Virtual-blind-road following-based wearable navigation device for blind people. *IEEE Transactions on Consumer Electronics*, *64*(1), 136-143.

[9]. Bai, J., Liu, Z., Lin, Y., Li, Y., Lian, S., & Liu, D. (2019). Wearable travel aid for environment perception and navigation of visually impaired people. *Electronics*, *8*(6), 697.

[10]. Pathak, A., Adil, M., Rafa, T. S., Ferdoush, J., & Mahmud, A. (2020). An IoT based voice controlled blind stick to guide blind people. *International Journal of Engineering Inventions*, *9*(1), 9-14.

[11]. Arora, A., Grover, A., Chugh, R., & Reka, S. S. (2019). Real time multi object detection for blind using single shot multibox detector. *Wireless Personal Communications*, *107*, 651-661.

[12]. Asati, C., Meena, N., & Orlando, M. F. (2019, October). Development of an intelligent cane for visually impaired human subjects. In *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)* (pp. 1-5). IEEE.

[13]. Bhole, S., & Dhok, A. (2020, March). Deep learning based object detection and recognition framework for the visually-impaired. In *2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC)* (pp. 725-728). IEEE.

[14]. Chen, L. B., Su, J. P., Chen, M. C., Chang, W. J., Yang, C. H., & Sie, C. Y. (2019, January). An implementation of an intelligent assistance system for visually impaired/blind people. In *2019 IEEE International Conference on Consumer Electronics (ICCE)* (pp. 1-2). IEEE.

[15]. Chiranjevulu, D., Sanjula, D., Kumar, K. P., Murali, U. B., & Santosh, S. (2020). Intelligent walking stick for blind people using Arduino. *Int. J. Eng. Res. Appl*, *10*, 42-45.

[16]. Gbenga, D. E., Shani, A. I., & Adekunle, A. L. (2017). Smart walking stick for visually impaired people using ultrasonic sensors and Arduino. *International journal of engineering and technology*, *9*(5), 3435-3447.

[17]. Mahalakshmi, S., Veena, N., Kumari, A., & Karnataka, M. (2020). Visual Assistance for Blind using Image Processing. *International Research Journal of Engineering and Technology (IRJET)*, *7*(07).

[18]. Johari, R., Gaurav, N. K., Chaudhary, S., & Pramanik, A. (2020, October). START: Smart Stick based on TLC Algorithm in IoT Network for Visually Challenged Persons. In *2020 Fourth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)* (pp. 605-610). IEEE.

[19]. Moharkar, L., Varun, S., Patil, A., & Pal, A. (2020). A scene perception system for visually impaired based on object detection and classification using CNN. In *ITM Web of Conferences* (Vol. 32, p. 03039). EDP Sciences.

[20]. Mule, N., Patil, D. D., & Chavhan, Y. D. (2020). In-house object detection system for visually impaired. *International Journal of Future Generation Communication and Networking*, *13*(4), 4919-4926.

[21]. Swetha, P., & Swami, T. (2021). AI Based Assistance for Visually Impaired People Using TTS (Text To Speech). *International Journal of Innovative Research in Science and Technology*, *1*(1), 8-14.

[22]. Sumathy, B., Pavithran, K. M., Nizam, N., & Surya, V. A. (2021). Smart Guidance System for Blind with Wireless Voice Playback. In *IOP Conference Series: Materials Science and Engineering* (Vol. 1012, No. 1, p. 012045). IOP Publishing.

[23]. Cheng, R., Hu, W., Chen, H., Fang, Y., Wang, K., Xu, Z., & Bai, J. (2021). Hierarchical visual localization for visually impaired people using multimodal images. *Expert Systems with Applications*, *165*, 113743.

[24]. Duh, P. J., Sung, Y. C., Chiang, L. Y. F., Chang, Y. J., & Chen, K. W. (2020). V-eye: A vision-based navigation system for the visually impaired. *IEEE Transactions on Multimedia*, *23*, 1567-1580.

[25]. Long, N., Yan, H., Wang, L., Li, H., & Yang, Q. (2022). Unifying Obstacle Detection, Recognition, and Fusion Based on the Polarization Color Stereo Camera and LiDAR for the ADAS. *Sensors*, *22*(7), 2453.

[26]. Joshi, R. C., Yadav, S., Dutta, M. K., & Travieso-Gonzalez, C. M. (2020). Efficient multi-object detection and smart navigation using artificial intelligence for visually impaired people. *Entropy*, *22*(9), 941.

[27]. Islam, M. M., Sadi, M. S., & Bräunl, T. (2020). Automated walking guide to enhance the mobility of visually impaired people. *IEEE Transactions on Medical Robotics and Bionics*, *2*(3), 485-496.

[28]. Lin, Y., Wang, K., Yi, W., & Lian, S. (2019). Deep learning based wearable assistive system for visually impaired people. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops* (pp. 0-0).

[29]. Younis, O., Al-Nuaimy, W., Rowe, F., & Alomari, M. H. (2019). A smart context-aware hazard attention system to help people with peripheral vision loss. *Sensors*, *19*(7), 1630.

[30]. Poggi, M., & Mattoccia, S. (2016, June). A wearable mobility aid for the visually impaired based on embedded 3D vision and deep learning. In *2016 IEEE symposium on computers and communication (ISCC)* (pp. 208-213). IEEE.

[31]. Long, N., Wang, K., Cheng, R., Hu, W., & Yang, K. (2019). Unifying obstacle detection, recognition, and fusion based on millimeter wave radar and RGB-depth sensors for the visually impaired. *Review of Scientific Instruments*, *90*(4), 044102.

[32] Mancini, A., Frontoni, E., & Zingaretti, P. (2018). Mechatronic system to help visually impaired users during walking and running. *IEEE transactions on intelligent transportation systems*, *19*(2), 649-660.

[33]Bułat, J., & Głowacz, A. (2016, September). Vision-based navigation assistance for visually impaired individuals using general purpose mobile devices. In *2016 International Conference on Signals and Electronic Systems (ICSES)* (pp. 189-194). IEEE.

[34] Aladren, A., López-Nicolás, G., Puig, L., & Guerrero, J. J. (2014). Navigation assistance for the visually impaired using RGB-D sensor with range expansion. *IEEE Systems Journal*, *10*(3), 922-932.

[35] Yang, K., Wang, K., Hu, W., & Bai, J. (2016). Expanding the detection of traversable area with RealSense for the visually impaired. *Sensors*, *16*(11), 1954.

[36] Kammoun, S., Parseihian, G., Gutierrez, O., Brilhault, A., Serpa, A., Raynal, M., ... & Jouffrais, C. (2012). Navigation and space perception assistance for the visually impaired: The NAVIG project. *Irbm*, *33*(2), 182-189.

[37] Yang, K., Wang, K., Bergasa, L. M., Romera, E., Hu, W., Sun, D., ... & López, E. (2018). Unifying terrain awareness for the visually impaired through real-time semantic segmentation. *Sensors*, *18*(5), 1506.

[38] Kang, M. C., Chae, S. H., Sun, J. Y., Lee, S. H., & Ko, S. J. (2017). An enhanced obstacle avoidance method for the visually impaired using deformable grid. *IEEE Transactions on Consumer Electronics*, *63*(2), 169-177.

[39] Kang, M. C., Chae, S. H., Sun, J. Y., Yoo, J. W., & Ko, S. J. (2015). A novel obstacle detection method based on deformable grid for the visually impaired. *IEEE Transactions on Consumer Electronics*, *61*(3), 376-383.

[40] Mekhalfi, M. L., Melgani, F., Zeggada, A., De Natale, F. G., Salem, M. A. M., & Khamis, A. (2016). Recovering the sight to blind people in indoor environments with smart technologies. *Expert systems with applications*, *46*, 129-138.

[41] Söveny, B., Kovács, G., & Kardkovács, Z. T. (2014, November). Blind guide-A virtual eye for guiding indoor and outdoor movement. In *2014 5th IEEE Conference on Cognitive Infocommunications (CogInfoCom)* (pp. 343-347). IEEE.

[42] Chaccour, K., & Badr, G. (2016, September). Computer vision guidance system for indoor navigation of visually impaired people. In *2016 IEEE 8th international conference on intelligent systems (IS)* (pp. 449-454). IEEE.

[43] Bai, J., Liu, D., Su, G., & Fu, Z. (2017, April). A cloud and vision-based navigation system used for blind people. In *Proceedings of the 2017 international conference on artificial intelligence, automation and control technologies* (pp. 1-6).

[44] https://ml-explained.com/blog/mean-shift-explained

[45] https://medium.com/@ankushsharma2805/yolo-v1-v2-v3-architecture-1ccac0f6206e

[46] https://pjreddie.com/darknet/yolov2/

[47] https://viso.ai/deep-learning/yolov3-overview/#:~:text=Computer%20Vision%20Teams-,What%20is%20YOLOv3%3F,network%20to%20detect%20an%20object.

[48] https://maelfabien.github.io/deeplearning/inception/#

[49] https://huggingface.co/docs/timm/models/inception-resnet-v2

[50] https://towardsdatascience.com/review-mobilenetv1-depthwise-separable-convolution-light-weight-model-a382df364b69

[51] https://medium.datadriveninvestor.com/review-on-mobile-net-v2-ec5cb7946784

[52] https://www.educative.io/answers/what-is-sift

[53] https://medium.com/data-breach/introduction-to-surf-speeded-up-robust-features-c7396d6e7c4e

[54] https://gisresources.com/what-is-slam-algorithm-and-why-slam-matters/

[55] https://es.mathworks.com/help/lidar/ug/get-started-pointpillars.html

[56] Read, Jenny CA. "Stereo vision and strabismus." *Eye* 29, no. 2 (2015): 214-224.

[57] https://en.wikipedia.org/wiki/Raspberry_Pi_OS

[58] https://www.python.org/doc/essays/blurb/

[59] https://opencv.org/about/

[60] https://numpy.org/doc/stable/user/whatisnumpy.html

[61] https://www.digitalocean.com/community/tutorials/how-to-install-and-use-pytorch.

[62] https://jupyter.org/

[63] Sombekke, Niels, Arnoud Visser. "Triangulation for Depth Estimation." (2022).

[64] Horloger Frédéric, Stéphane Vujasinovic. (30.09.2017) “Stereo vision”, University of Applied Sciences Hochschule Karlsruhe, Germany, p30

Annexe A

	Proximité	Vision	Combinaison Vision et Proximité	Autre combinaisons
Capteurs	Électromagnétique [2,] IR [14,10,22] Ultrasonique [4,5,10,22]	Camera [12,13,24,29] Camera Pi [3,11 ,19,20] Stereo-based RGBD Camera [28,23,30] IR [23,30]	Polariation color D Camera [25] Lidar [25] Camera [26] D-camera [6,7,8] Ultrasonique [6,8, 17,26,27] Capteur infrarouge [7] Fisheye camera [8] RGB-D camera [9,31] IMU [9] Camera Pi [17,27] IR camera [31] Radar MMW [31]	Ultrasonique [15,16,18,21] GPS [21,18] LDR [15] Humidité [16] Couleur [18] Camera Pi [21]
Assistance fournie	Détection d'obstacle [2,10,14,15,16,22] Détection d'objets [5] Détection d'escalier [10] Détection de lumière [15,16] Détection d'humidité [16]	OCR [3 ,19] Identification d'objet [11,13,20,30] Détection d'objet [13,20,29,30] Identification de visage [13] Identification de monnaie [13] Localisation [23] Détection d'obstacle [24,28,29,30] Navigation [24] Perception de l'environnement [28] Suivi d'objet [28]	Navigation [6,7,8 ,9] Détection d'obstacle [12,21,26 ;27,31] Détection nid de poule [27] Détection d'objet [17,26,31,32] Identification d'objet [17,32] Détection de texte [21] OCR [21]	Navigation [4] Détection d'obstacles [18], Détection des couleurs de feu de signalisation [18]
Dispositif	Canne [2,10,14,15,16,22]	Lunettes [20,29] Headgear[23] Ordinateur portable [23,30] Canne [24] Camera portable [24,28] Écouteur [28]	Lunettes [6,8,9,27] Headgear[7] Ceinture tactile [7] Canne [12] Appareil portable [21,26,31] Casque à conduction osseuse [31]	Soulier [4] Canne [18]

Annexe A

		Gants|30] Casque à conduction osseuse [23,30]		
Réponse en sortie	Audio [2,5,10,15,22] Vibration [5,14,15,16]	Audio [3,11,19,20,24,28,23] Visuel [29,30] Tactile [30]	Audio [6,7,8,9,12,17,21,26,27,31,25] Visuel [6,8]	Audio [18] Tactile [4]

Tableau A.1: Résumé des différents systèmes de substitution

Annexe B

i. Plateformes et Algorithmes de traitement d'image :

i.1. Algorithme MeanShift :

L'algorithme MeanShift est un type de méthode de regroupement non supervisée conçue pour identifier des regroupements dans une distribution continue de point de données. Cette approche basée sur les centroïdes met à jour les centroïdes potentiels en calculant la moyenne des points situés dans une région spécifique, également appelée bande passante. Par la suite, ces centroïdes potentiels font l'objet d'une étape de post-traitement visant à éliminer les similaires, ce qui donne l'ensemble définitif de centroïdes. De manière générale, l'algorithme MeanShift fonctionne comme suit :
Créer une fenêtre coulissante ou un cluster pour chaque point de données.

1. Déplacer chaque fenêtre coulissante vers les régions de densité plus élevée en déplaçant leur centroïde (le centre de la fenêtre coulissante) vers la moyenne des points de données à l'intérieur de cette fenêtre. Répéter cette étape jusqu'à ce qu'aucun déplacement supplémentaire n'entraîne une densité plus élevée (c'est-à-dire plus de points à l'intérieur de la fenêtre coulissante).
2. Sélectionner les fenêtres coulissantes en supprimant celles qui se chevauchent. Lorsque plusieurs fenêtres coulissantes se chevauchent, conserver celle qui contient le plus de points et supprimer les autres.
3. Assigner les points de données à la fenêtre coulissante dans laquelle ils se trouvent.

MeanShift, déplace les fenêtres vers une région de densité plus élevée en déplaçant leur centroïde (centre de la fenêtre coulissante) vers la moyenne des points de données à l'intérieur de la fenêtre coulissante. Il est possible de le voir également en considérant les points de données comme une fonction de densité de probabilité [44].

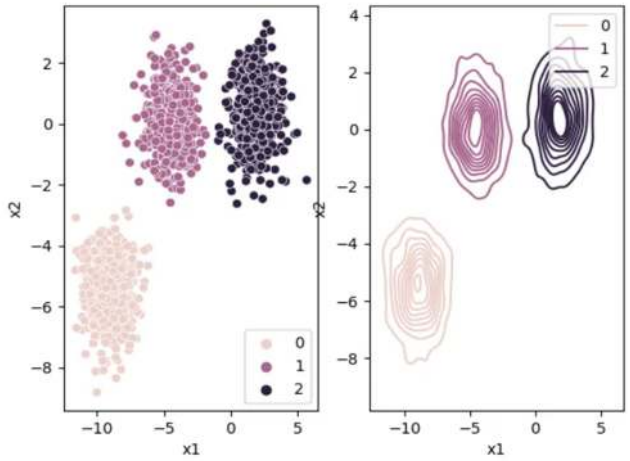

Figure B.1 : Image qui montre différente valeur de largeur de bande (cas extrême) [44].

Les régions avec une plus grande concentration de points de données sont appelées régions de densité plus élevée, tandis que les régions avec un nombre plus faible de points sont considérées comme des régions de densité plus faible. L'objectif de MeanShift est de localiser ces régions de haute densité en déplaçant de manière itérative la fenêtre coulissante vers le pic de chaque région. Ce processus est également connu sous le nom de recherche de pente.
L'apparence des clusters peut varier considérablement en fonction de la largeur de bande choisie. Pour illustrer un scénario extrême, il faut envisager de sélectionner une largeur de bande extrêmement

étroite, où chaque point forme son propre cluster distinct. En revanche, l'utilisation d'une largeur de bande considérablement élevée résulterait en un seul cluster englobant tous les points de données [44].

La figure B.2 est une image montrant différentes valeurs de largeur de bande (moins extrêmes que celles mentionnées ci-dessus):

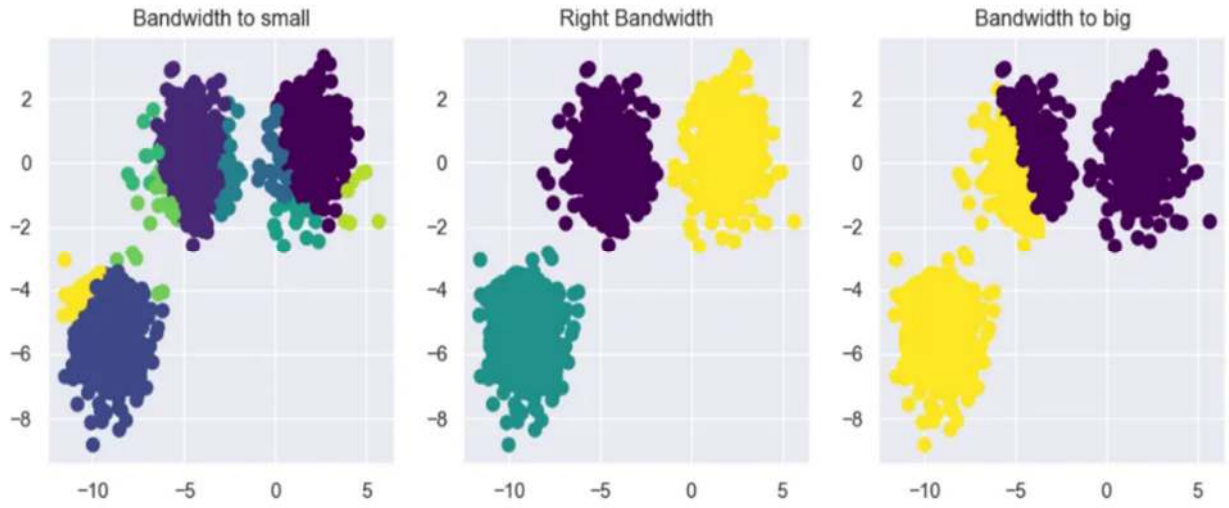

Figure B.2 : Image qui montre différente valeur de largeur de bande [44].

La sélection manuelle de la largeur de bande appropriée peut être réalisable pour de petits ensembles de données bidimensionnels, mais devient de plus en plus difficile à mesure que l'ensemble de données devient de plus en plus grand.

MeanShift est une méthode de regroupement simple qui fonctionne très bien sur des données de forme sphérique. De plus, il détermine automatiquement le nombre de clusters. De plus, la sortie de MeanShift n'est pas influencée par le processus d'initialisation car chaque point est initialement considéré comme un cluster individuel.

La scalabilité de l'algorithme est limitée en raison de la nécessité d'effectuer plusieurs recherches de voisins les plus proches pendant son exécution. Sa complexité est de O(n^2). De plus, la sélection manuelle de la largeur de bande peut être difficile, et choisir une valeur incorrecte peut entraîner de mauvais résultats [44].

i.2. YOLOv1

L'algorithme YOLO, nommé "you only look once", tire son nom de son utilisation des convolutions 1×1 pour la prédiction. Cette conception garantit que la taille de la carte de prédiction correspond précisément à la taille de la carte de caractéristiques précédente.

YOLOv1(You Only Look Once, Version 1), est une approche de détection d'objets incroyablement rapide et qui réalise un traitement d'images en temps réel à une vitesse remarquable de 45 images par seconde. YOLO se distingue par sa simplicité, en utilisant un unique réseau de convolution qui prédit efficacement plusieurs boîtes englobantes et les probabilités de classe correspondantes [45].

Annexe B

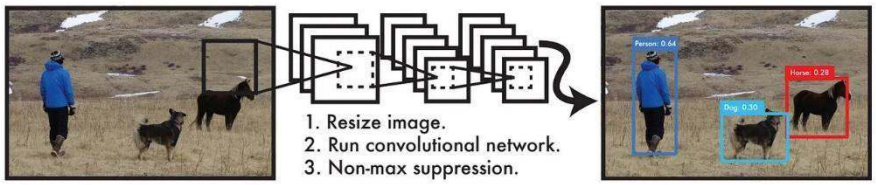

Figure B.3 : Système de détection YOLO [45].

Contrairement à d'autres méthodes de détection qui utilisent une approche de fenêtre glissante pour extraire les caractéristiques, YOLO tire parti de l'ensemble de l'image à la fois lors de l'entraînement et des tests. Cela permet à YOLO de capturer des informations contextuelles sur les classes et leurs apparences. Comparé à Fast R-CNN, YOLO commet significativement moins d'erreurs de fond, ce qui le rend plus précis [45].
YOLO acquiert des représentations d'objets généralisées, ce qui démontre sa polyvalence. En s'entraînant sur des images naturelles et en les testant sur des œuvres d'art, YOLO dépasse largement les techniques de détection de premier plan telles que DPM et R-CNN [45]

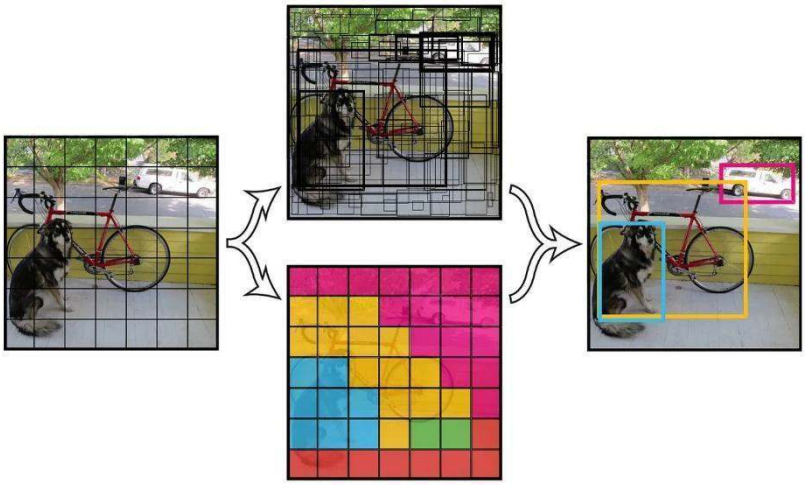

Figure B.4 : L'image est divisée en boîtes de taille uniforme, et les boîtes englobantes sont dessinées avec des épaisseurs de ligne variables, reflétant les scores de confiance [45].

Les performances du modèle YOLO ont été évaluées en utilisant le jeu de données de détection Pascal VOC. Les couches de convolution initiales du réseau extraient des caractéristiques de l'image, tandis que les couches entièrement connectées prédisent les probabilités de sortie et les coordonnées. Le modèle comprend 24 couches de convolution, suivies de 2 couches entièrement connectées [45].

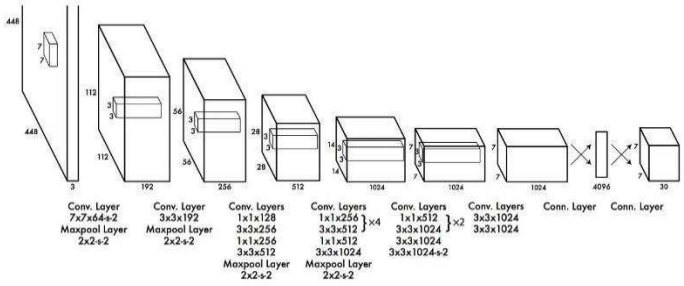

Figure B.5 : Architecture de YOLOV1 [45].

YOLO impose des contraintes spatiales strictes sur les prédictions des boîtes englobantes en

permettant à chaque cellule de grille de prédire uniquement deux boîtes et de les associer à une seule classe. Cette limitation spatiale restreint la capacité du modèle à prédire un grand nombre d'objets proches. Par conséquent, le modèle rencontre des difficultés lorsqu'il s'agit de traiter de petits objets qui apparaissent en grappes, tels que des groupes d'oiseaux [45].

i.3. YOLOv2

YOLOv2(You Only Look Once, Version 2), également connu sous le nom de YOLO9000, est un modèle de détection d'objets en temps réel qui fonctionne en une seule étape. Il améliore YOLOv1 grâce à diverses améliorations, telles que l'intégration de Darknet-19 en tant que base, l'utilisation de la normalisation par lots, l'emploi d'un classifieur à haute résolution et la mise en œuvre de boîtes d'ancrage pour les prédictions de boîtes englobantes, parmi d'autres améliorations.

Les classificateurs ou localisateurs sont réutilisés dans les systèmes de détection précédents pour effectuer la détection. Le modèle est appliqué à une image à plusieurs emplacements et échelles. Les détections sont considérées dans les régions de l'image qui produisent des scores élevés.

Une approche totalement différente est utilisée. Un seul réseau neuronal est appliqué à l'image complète. L'image est divisée en régions par ce réseau, et des boîtes englobantes et des probabilités sont prédites pour chaque région. Les probabilités prédites sont utilisées pour pondérer les boîtes englobantes [46].

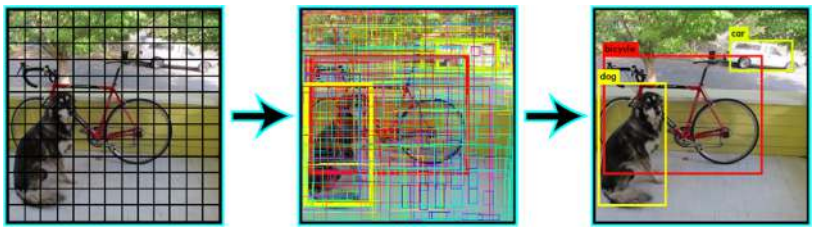

Figure B.6: Approche de l'implémentation de l'algorithmes YOLOv2 [46].

YOLOv2 intègre plusieurs techniques pour améliorer l'entraînement et augmenter les performances. Il utilise un modèle entièrement convolutif, mais il s'entraîne sur des images complètes plutôt que de se concentrer uniquement sur les exemples difficiles. De manière similaire à Faster R-CNN, il adapte les priorités sur les boîtes englobantes au lieu de prédire directement leur largeur et leur hauteur. Néanmoins, il prédit directement les coordonnées x et y [46].

i.4. Yolo V3

YOLOv3 (You Only Look Once, Version 3) est un algorithme avancé de détection d'objets en temps réel conçu pour identifier des objets spécifiques dans des vidéos, des flux en direct ou des images. Il exploite un réseau de neurones convolutifs profonds pour apprendre des caractéristiques pertinentes à la détection d'objets [47].

Annexe B

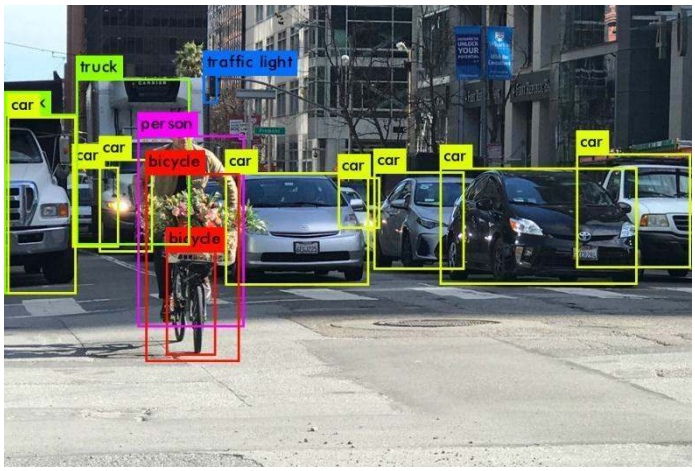

Figure B.7 : Exemple d'application de vision par ordinateur de YOLOV3 [47].

i.5. Inception v1

Inception, introduit en 2014, est une architecture de réseau neuronal convolutif profond qui s'est imposée comme le vainqueur du défi de reconnaissance visuelle à grande échelle ImageNet (ILSVRC14). La majorité de son développement a été menée par des chercheurs de Google. Le nom "Inception" a été inspiré par le film du même nom.

Dans les réseaux de neurones convolutifs (CNN), un aspect important consiste à sélectionner la couche appropriée à appliquer parmi plusieurs options couramment utilisées (comme les filtres 1x1, les filtres 3x3, les filtres 5x5 ou le max-pooling). L'objectif est d'identifier l'agencement local optimal et de le répliquer spatialement. La clé réside dans la recherche de la configuration locale la plus adaptée et son application cohérente sur l'ensemble du réseau [48].

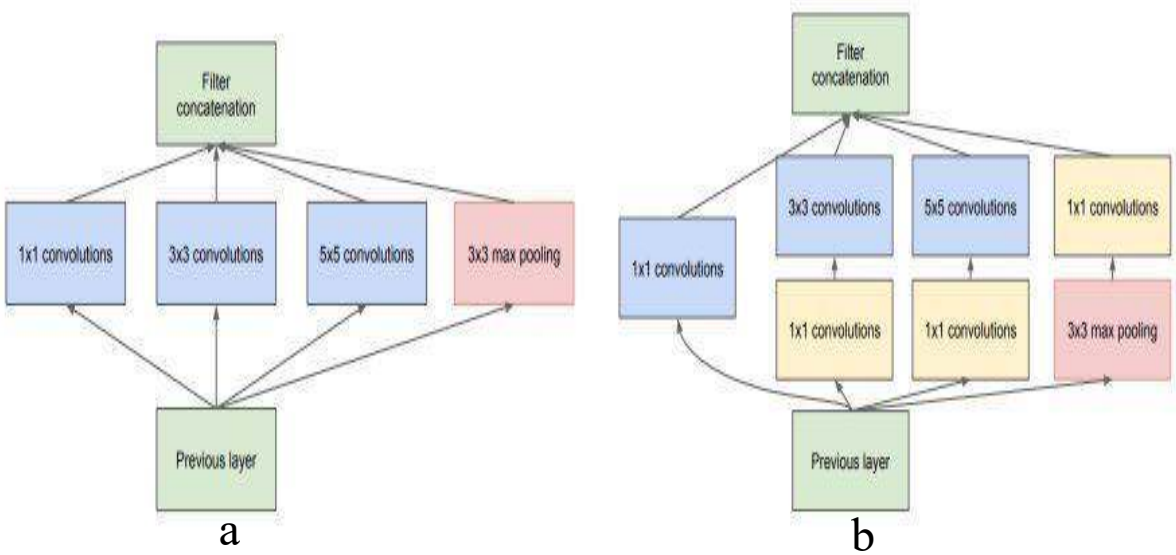

Figure B.8 : (a) module inception : version naïve, (b) : module inception avec réduction de dimension [48]

Comme les "modules Inception" sont empilés de manière en cascade, on s'attend à ce que les statistiques de corrélation de leurs sorties varient. Avec les couches supérieures capturant des caractéristiques plus abstraites, la concentration spatiale de ces caractéristiques à tendance à diminuer. Cela suggère que le ratio de convolutions 3×3 et 5×5 devrait augmenter à mesure que l'on avance vers les couches supérieures.

Cependant, la mise en œuvre de cette solution implique un coût computationnel élevé. Pour résoudre ce problème, des techniques de réduction de dimension utilisant des convolutions 1x1 sont employées, comme le montre la figure B.8 (b)[48].

i.6. Inception v3

Inception v3 vise principalement à réduire l'utilisation des ressources computationnelles grâce à des adaptations par rapport aux conceptions précédentes d'Inception.

Au sein du modèle Inception v3, différentes stratégies ont été proposées pour améliorer l'optimisation du réseau, le rendant plus adaptable en relâchant les contraintes. Ces techniques englobent les convolutions factorisées, la régularisation, la réduction de dimension et les calculs parallélisés. [49].

i.7. MobileNet

MobileNet est une catégorie de réseau neuronal convolutif (CNN) publiée en tant que projet open-source par Google. Il sert de base exceptionnelle pour l'entraînement de classificateurs qui sont remarquablement compacts et offrent une vitesse exceptionnelle. La vitesse et la consommation d'énergie du réseau sont directement proportionnelles au nombre de multiplications-accumulations (MACs), qui mesure le nombre total d'opérations de multiplication et d'addition fusionnées

Le concept des convolutions séparables découle de l'idée que les dimensions spatiales et la profondeur d'un filtre peuvent être traitées séparément, d'où le terme "séparable".

La convolution séparable en profondeur (depthwise separable convolution) implique deux étapes : la convolution en profondeur (depthwise convolution) et la convolution ponctuelle (pointwise convolution).

- La convolution en profondeur consiste à effectuer des convolutions spatiales individuelles de taille DK×DK sur chaque canal d'entrée séparément. Par exemple, s'il y a cinq canaux d'entrée, il y aura cinq convolutions spatiales de taille DK×DK.
- La convolution ponctuelle, quant à elle, est une convolution 1×1 qui vise à changer la dimensionnalité des données.

La convolution en profondeur fonctionne sur chaque canal d'entrée de manière indépendante, ce qui entraîne le même nombre de canaux de sortie que de canaux d'entrée. Son coût de calcul peut être calculé comme suit : $Df^2 * M * Dk^2$, où Df représente la dimension spatiale, M désigne le nombre de canaux d'entrée, et Dk représente la taille du noyau.

La convolution ponctuelle (pointwise convolution) est une convolution avec une taille de filtre de 1x1 qui combine simplement les caractéristiques créées par la convolution en profondeur. Son coût de calcul est : $M * N * Df^2$, où M représente le nombre de canaux d'entrée, N désigne le nombre de canaux de sortie et Df représente la dimension spatiale [50].

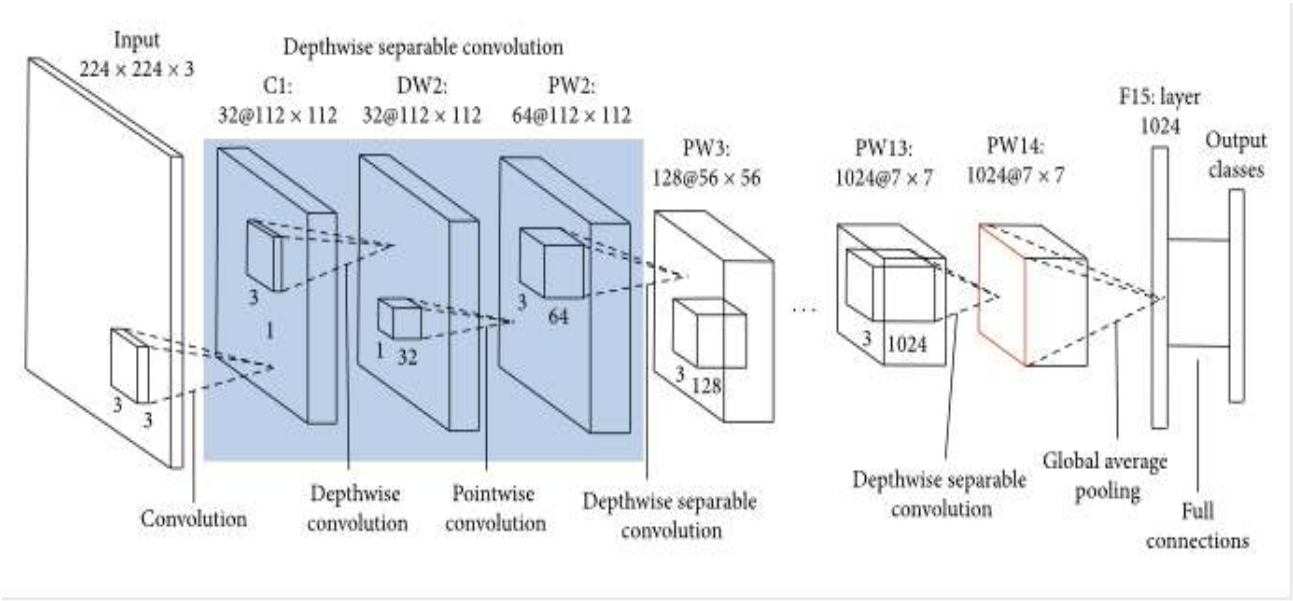

Figure B.9: Architecture du MobileNet [50]

i.8. MobileNetV2

MobileNetV2 introduit un module amélioré appelé la structure résiduelle inversée. Dans cette version, les non-linéarités des couches étroites sont éliminées. En utilisant MobileNetV2 pour l'extraction des caractéristiques, d'impressionnants résultats sont également obtenus dans les tâches de détection d'objets et de segmentation sémantique.

MobileNetV2 introduit deux types de blocs : le bloc résiduel avec un pas de 1 et le bloc de réduction avec un pas de 2. Les deux types de blocs sont composés de trois couches. Dans ce cas, la couche initiale est une convolution 1×1 avec une activation ReLU6. La deuxième couche applique une convolution en profondeur, et la troisième couche est une convolution 1×1 sans aucune non-linéarité. Il est souligné que l'omission de ReLU dans cette dernière couche restreint les réseaux profonds à fonctionner uniquement comme des classificateurs linéaires sur la partie du domaine de sortie où la valeur est non nulle. De plus, il existe un facteur d'expansion constant noté "t", qui reste fixé à 6 pour toutes les expériences principales. Si l'entrée contient 64 canaux, la sortie interne obtenue comprend 64×t=64×6=384 canaux [51].

Annexe B

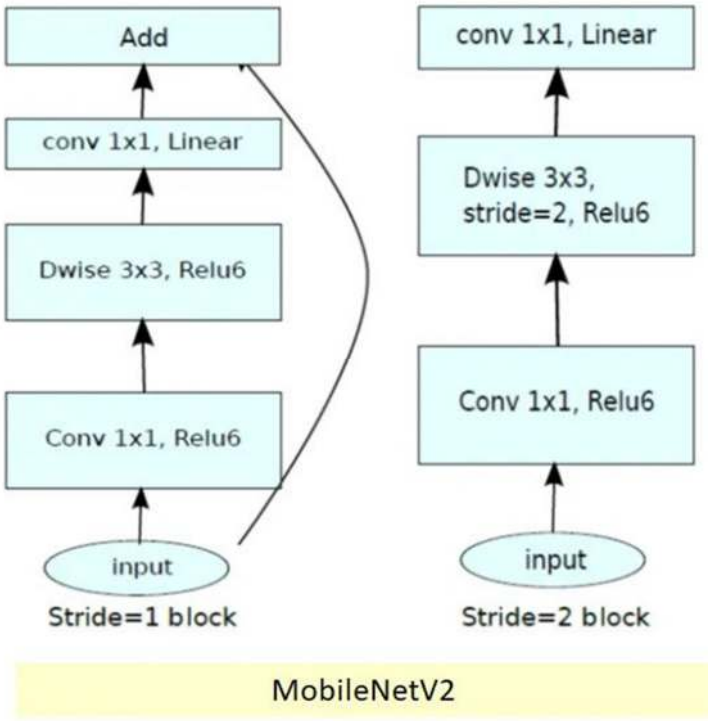

Figure B.10: architecture du mobilenetv2 [51].

i.9. Scale Invariant Feature Transform (SIFT)

L'algorithme de transformation de caractéristiques visuelles invariantes à l'échelle (SIFT) est une méthode de vision par ordinateur utilisée pour identifier et représenter les caractéristiques d'une image. Il identifie des points clés uniques ou des caractéristiques au sein d'une image qui restent cohérents même lorsque l'image subit des altérations d'échelle, de rotation ou de transformations affines. SIFT fonctionne en reconnaissant les points clés grâce à leurs extrêmes d'intensité locaux et en générant des descripteurs qui encapsulent les détails de l'image environnante associés à ces points clés. Ces descripteurs peuvent ensuite être utilisés pour des activités telles que la comparaison d'images, la reconnaissance d'objets et la recherche d'images [52].

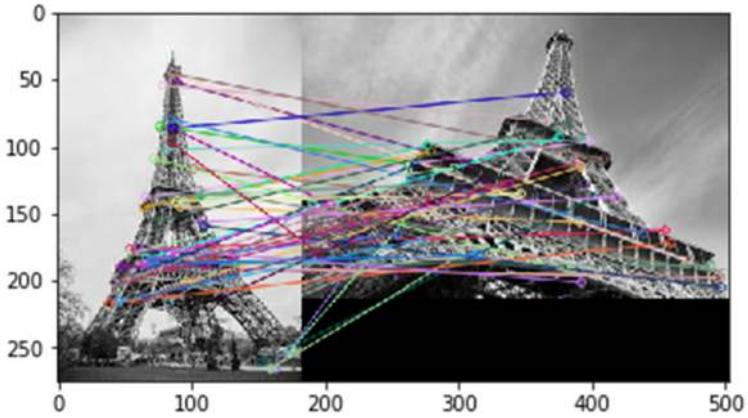

Figure B.11 : exemple de l'algorithmes SIFT [52]

Avantages	Explication
Localité,	Les caractéristiques extraites peuvent être comparées à de vastes ensembles de données d'objets.
Distinctivité	Étant donné que les caractéristiques extraites à l'aide de SIFT sont locales, elles ne sont pas affectées par le bruit et l'encombrement présents dans l'image.
Quantité,	SIFT a la capacité de générer une multitude de caractéristiques même à partir de petits objets.
Efficacité	L'algorithme démontre des performances équivalentes aux capacités en temps réel

Tableau B.1 : Avantage et inconvénient de SIFT [52]

i.10. SURF (Speeded Up Robust Features)

SURF pour caractéristiques robustes accélérées est un algorithme de détection de caractéristique et un descripteur utilisé dans la vision par ordinateur. Il est capable d'accomplir des fonctions telles que la reconnaissance d'objets, l'alignement d'images, la classification et la reconstruction tridimensionnelle [53].

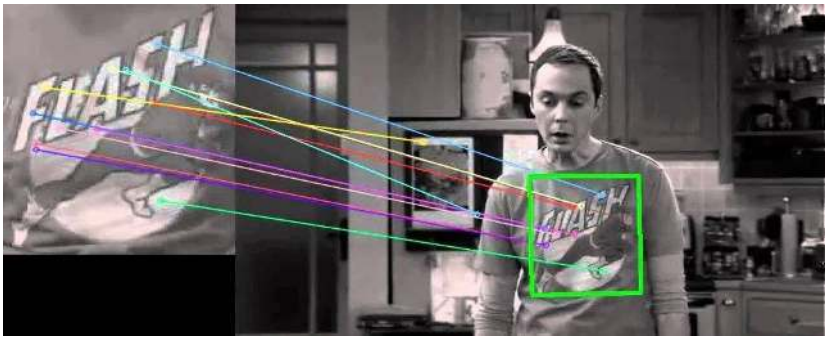

Figure B.12 : Exemple d'utilisation de l'algorithmes SURF [53].

i.11. SLAM (Simultaneous Localization and Mapping)

Le SLAM (Simultaneous Localization and Mapping) est un algorithme essentiel pour les appareils et les robots, leur permettant de créer une carte de leur environnement tout en déterminant leur propre position en temps réel. Cela joue un rôle crucial dans diverses applications, des véhicules autonomes aux environnements intérieurs, extérieurs, aériens et sous-marins. Il utilise divers capteurs, tels que des caméras, des RADAR, des SONAR et des LiDAR, ainsi qu'une unité de mesure inertielle (IMU), pour collecter des données visibles et invisibles et estimer la position du véhicule. Le SLAM implique des calculs mathématiques et la fusion de données de capteurs, se décomposant en deux étapes : la collecte de données en amont (SLAM visuel et SLAM LiDAR) et le traitement des données en aval (méthodes parcimonieuses et méthodes denses). Pour améliorer la vitesse de traitement, le SLAM

LiDAR peut être associé à d'autres mesures, telles que l'odométrie des roues, le GNSS et les données de l'IMU [54].

i.12. Pointpillar

PointPillars est une méthode de détection d'objets 3D utilisant des couches convolutifs 2D. Le réseau PointPillars dispose d'un encodeur apprenable qui utilise des PointNets pour apprendre une représentation des nuages de points organisés en piliers (colonnes verticales). Le réseau exécute ensuite un réseau de neurones convolutifs (CNN) 2D pour produire des prédictions du réseau, décode ces prédictions et génère des boîtes englobantes 3D pour différentes classes d'objets tels que voitures, camions et piétons.

Le réseau PointPillars comprend ces étapes principales :

1. Utiliser un encodeur de caractéristiques pour convertir un nuage de points en une pseudo-image sparse.
2. Traiter la pseudo-image pour obtenir une représentation de haut niveau à l'aide d'une structure de convolution 2D.
3. Détecter et régresser les boîtes englobantes 3D à l'aide de têtes de détection.

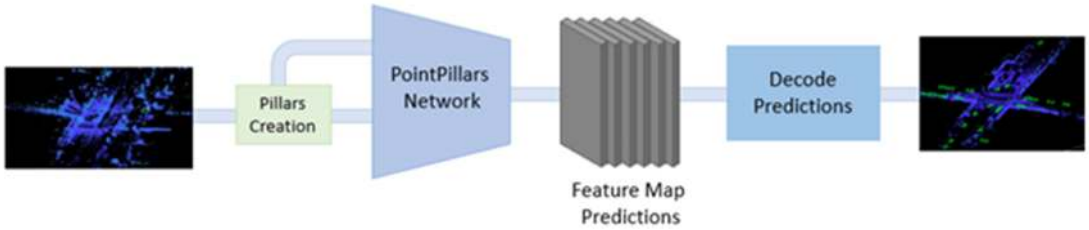

Figure B.13 : Les trois étapes du réseau point pillar [55]

Un réseau PointPillars nécessite deux entrées : les indices de piliers sous forme d'une matrice de dimension P-par-2 et les caractéristiques des piliers sous forme d'une matrice de dimension P-par-N-par-K. P représente le nombre de piliers dans le réseau, N représente le nombre de points par pilier et K représente la dimension des caractéristiques.

- Le réseau commence par un encodeur de caractéristiques, qui est une version simplifiée de PointNet. Il comprend une série de couches de convolution, de normalisation en lots et de fonctions d'activation ReLU, suivies d'une couche de maxpooling. Une couche de dispersion à la fin mappe les caractéristiques extraites dans un espace 2D en utilisant les indices de piliers.
- Ensuite, le réseau dispose d'une structure de convolution 2D comprenant des blocs encodeurs-décodeurs. Chaque bloc encodeur est constitué de couches de convolution, de normalisation en lots et de fonctions d'activation ReLU pour extraire les caractéristiques à différentes résolutions spatiales. Chaque bloc décodeur est constitué de couches de convolution transpose, de normalisation en lots et de fonctions d'activation ReLU.
- Le réseau concatène ensuite les caractéristiques de sortie à la fin de chaque bloc décodeur et fait passer ces caractéristiques à travers six têtes de détection comprenant des couches de convolution et de sigmoïde pour prédire l'occupation, la position, la taille, l'angle, l'orientation et la classe des objets [55].

Annexe B

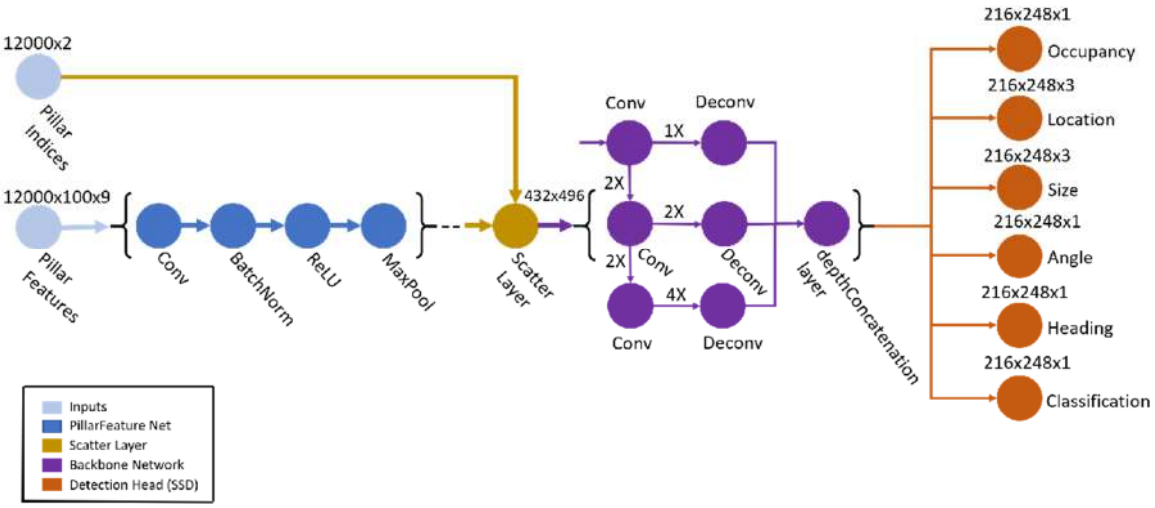

Figure B.14 : Architecture du PointPillar [55].

Capteur	2.1 Mpx
Résolution audio et vidéo	1920 x 1080
Flux vidéo max. (en images/s)	30

Tableau C.1 : Caractéristiques de la caméra Logitech C910

Processeur	Broadcom BCM2711, quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5GHz
Mémoire	4GB LPDDR4
Connectivité	2.4 GHz et 5.0 GHz IEEE 802.11b/g/n/ac wireless LAN, Bluetooth 5.0, BLE Gigabit Ethernet 2 × USB 3.0 ports 2 × USB 2.0 ports.
GPIO	Standard 40-pin GPIO header (Entièrement rétro compatible avec les cartes précédentes)
Son Video	2 × micro HDMI ports (jusqu'à 4Kp60 supporté) 2-lane MIPI DSI display port 2-lane MIPI CSI camera port 4-pole stereo audio et composite video port
Multimédia	H.265 (4Kp60 décodé); H.264 (1080p60 décodé, 1080p30 codé); OpenGL ES, 3.0 graphics
Carte Sd supporté	Micro SD card slot pour charger le système d'exploitation et le stockage des données
Puissance d'entrée	5V DC via connecteur USB-C (minimum 3A) 5V DC via l'en-tête GPIO (minimum 3A) Power over Ethernet (PoE) - activé (Nécessite un chapeau PoE séparé)
Environnement	Température de fonctionnement 0–50ºC

Tableau C.2 : Caractéristiques de la carte Rasperry Pie 4 model

```
# importation des packages
import numpy as np
import cv2
```

Figure C1 : Importation des packages.

```
# Appel des deux cameras
CamD= cv2.VideoCapture('/dev/video0')
CamG= cv2.VideoCapture('/dev/video2')
CamD.set(cv2.CAP_PROP_FRAME_WIDTH,320)
CamD.set(cv2.CAP_PROP_FRAME_HEIGHT,240)
CamG.set(cv2.CAP_PROP_FRAME_WIDTH,320)
CamG.set(cv2.CAP_PROP_FRAME_HEIGHT,240)
```

Figure C.2 : Activer les caméras avec OpenCV

```
retG, frameD= CamD.read()
retL, frameG= CamG.read()
grayD= cv2.cvtColor(frameD,cv2.COLOR_BGR2GRAY)
grayG= cv2.cvtColor(frameG,cv2.COLOR_BGR2GRAY)
```

Figure C.3: Traitement d'image

```
                cv2.imshow('VideoD',grayD)
                cv2.imshow('VideoG',grayG)
```

Figure C.4 : montrer des images

```
            # Terminer le programme.
            if cv2.waitKey(1) & 0xFF == ord(' '):
                break
        # Libérer les caméras
        CamD.release()
        CamG.release()
        cv2.destroyAllWindows()
```

Figure C.5 : Sortie typique d'un programme

```
# Déterminer les nouvelles valeurs pour différents paramètres
#   coté droit
retR, mtxR, distR, rvecsR, tvecsR = cv2.calibrateCamera(objpoints,
                                                        imgpointsR,
                                                        ChessImaR.shape[::-1],None,None)
hR,wR= ChessImaR.shape[:2]
OmtxR, roiR= cv2.getOptimalNewCameraMatrix(mtxR,distR,
                                                    (wR,hR),1,(wR,hR))

#   coté gauche
retL, mtxL, distL, rvecsL, tvecsL = cv2.calibrateCamera(objpoints,
                                                        imgpointsL,
                                                        ChessImaL.shape[::-1],None,None)
hL,wL= ChessImaL.shape[:2]
OmtxL, roiL= cv2.getOptimalNewCameraMatrix(mtxL,distL,(wL,hL),1,(wL,hL))
```

Figure C.6 : Calibrage de la distorsion en Python

```
# Créez un object StereoSGBM et préparez tous les paramètres
window_size = 4
# min_disp = 20
min_disp = 1
num_disp = 60
stereo = cv2.StereoSGBM_create(
        minDisparity=min_disp,
        numDisparities=num_disp,
        blockSize=window_size,
        P1=232,
        P2=2103,
        disp12MaxDiff=20,
        uniquenessRatio=0,
        speckleWindowSize=10,
        speckleRange=1,
        preFilterCap=20,
        mode=cv2.STEREO_SGBM_MODE_HH)
```

Figure C.7 : Paramètres pour l'instance de StereoSGBM

```
# Parametre du filtre WLS
lmbda = 80000
sigma = 2.3
visual_multiplier = 1.0
```

Figure C.8 : Paramètres du filtre WLS

```
# Utilisé pour l'image filtrée
stereoR=cv2.ximgproc.createRightMatcher(stereo) # Create another stereo for right this time
```

```
wls_filter = cv2.ximgproc.createDisparityWLSFilter(matcher_left=stereo)
wls_filter.setLambda(lmbda)
wls_filter.setSigmaColor(sigma)
```

Figure C.9: Création d'un filtre WLS en Python

```
# Utilisation du filtre WLS
filteredImg= wls_filter.filter(displ,grayL,None,dispR)
filteredImg = cv2.normalize(src=filteredImg, dst=filteredImg, beta=0, alpha=255, norm_type=cv2.NORM_MINMAX);
filteredImg = np.uint8(filteredImg)
```

Figure C.10: implémentation du filtre WLS en Python

```
import matplotlib.pyplot as plt
import numpy as np
disparity=[0.983,0.881,0.738,0.645,0.568,0.501,0.45,0.416,0.382,0.357,0.334,0.315,0.296,0.277,0.258,0.239,0.22]
distance=[40,50,60,70,80,90,100,110,120,130,140,150,160,170,180,190,200]
plt.title('distance en fonction de la disparité')
plt.xlabel('disparité')
plt.ylabel('distance')
plt.plot(disparity,distance)
plt.show()
```

Figure C.11 : Code permettant d'obtenir la droite de régression

Résumé :

Les personnes atteintes de déficiences visuelles rencontrent d'importantes difficultés lorsqu'elles se déplacent d'un endroit à un autre. Elles font face à des obstacles lorsqu'elles sortent à l'extérieur et cherchent à se protéger des objets en mouvement et immobiles, ce qui conduit souvent à une diminution de leur confiance en elles. L'augmentation substantielle de la population atteinte de déficience visuelle a suscité d'importantes recherches dans la création de dispositifs d'assistance.

Ce mémoire est divisé en trois partie, une partie qui est consacré à l'état de l'art des systèmes de substitutions, le second chapitre est consacré aux systèmes de substitution basé vision avec une vue critique de ces derniers ainsi que l'état de l'art des algorithmes de vision par ordinateur utilisés par ces systèmes , le chapitre 3 détaille le système de substituions qu'on a développé et il présente les différents composant matérielle et logiciel ainsi que son l'implémentation et les résultats obtenus.

L'objectif de ce travail est de concevoir un système de substitution basé vision pour les malvoyants avec deux caméras et une carte Raspberry pi. Plusieurs bibliotéque ont été utilisé en utilisant le langage de programmation python. Nous avons aussi utilisé le système d'exploitation Raspberry pie ainsi que l'environnement de développement interactif basé sur le web jupyterLab.

Une rectification et un calibrage stéréo a été effectué en utilisant une mire de calibrage pour obtenir une paire d'image stéréo auxquelles les lignes épipolaire sont alignées puis cette paire d'image est utilisée pour obtenir une carte de disparité puis un filtre de fermeture à étè utilisé pour enlever les point noir de la carte de disparité puis un filtre WLS est utilisé pour obtenir une carte de disparité ou les contours sont plus visible, puis une étude expérimentale nous a permis d'aboutir à une droite de régression distance en fonction de la disparité. En utilisant l'algorithme SIFT sur un pair d'image calibré, nous avons pu obtenir une représentation 3D et 2D précise d'une scène en intérieure et extérieur à partir des points appariés. Nous avons tenté de comparer les distances réel et obtenue par triangulation, cette comparaison nous a permis de déduire que ces deux distances se rapprochent.