

2024/2025

# Conception détaillée

Canne Intelligente pour Aveugle

**Encadrant :** Gilles VENTURINI

**Client :** Gilles VENTURINI

**Etudiant :** Irfan AKTURK

Document de conception détaillée			
Projet :	Projet : Canne Intelligente pour aveugle		
Emetteur :	Irfan AKTURK	Coordonnées : <a href="mailto:irfan.akturk@etu.univtours.fr">irfan.akturk@etu.univtours.fr</a>	
Date d'émission :			
Validation			
Nom	Date	Valide (Oui/Non)	Commentaires
M. Venturini	18/10/2024		
M.Chauvin	10/10/2024		
Historique des modifications			
Version	Date	Description	
1.0	04/11/2024	Création du document	

## Table des matières

Introduction .....	4
1. Spécifications des modules matériels.....	5
2. Description des modules logiciels .....	6
2.1 Module de capture d'images.....	6
2.2 Module de reconnaissance d'objets.....	7
2.3 Module de traitement des données de distance .....	7
2.4 Module de gestion des modes de fonctionnement.....	7
2.5 Module de gestion des retours (sonores et vibratoires) .....	8
2.6 Module de correction d'erreurs au démarrage .....	8
3. Diagrammes détaillés .....	9
3.1 Diagrammes de séquences logicielles .....	9
3.2 Diagramme d'interactions des modules logiciels .....	10
4. Test des composants .....	12
4.1 Tests de fonctionnement.....	12

# Introduction

Ce document de conception détaillée a pour but de décrire précisément chaque composant matériel et logiciel du projet de canne intelligente pour les personnes aveugles ou malvoyantes. En s'appuyant sur l'architecture générale, on va ici se concentrer sur les aspects techniques spécifiques, comme les réglages de chaque composant, les branchements et les interactions entre les différents modules du système.

L'idée est de fournir un guide pratique pour le montage et le développement, en s'assurant que chaque élément du système est bien documenté. Chaque section présente un composant ou un module en détail, pour qu'on ait une vue complète de l'ensemble de la canne et de son fonctionnement.

En suivant cette conception détaillée, on espère garantir que chaque partie du système fonctionnera correctement et répondra aux attentes en matière de performance et d'efficacité.

# 1. Spécifications des modules matériels

Dans cette partie nous allons voir les spécifications liées aux matériels, notamment sur le câblage de celui-ci sur la Jetson Nano et la communication utiliser.

## **Caméra IMX219-200 :**

Connecter au port CSI de la Jetson Nano pour assurer une transmission rapide.

## **Capteur à ultrasons A02YYUW :**

Le capteur fonctionne en liaison UART, le connecter au pins 8 / 10 de la Jetson Nano. Alimenter le capteur via un pin 5V.

## **Boutons poussoirs :**

Relier le bouton poussoir au pin 11, pas de résistance de pull-up/down nécessaire le bouton en intègre déjà un. Le bouton est à alimenter en 3,3V via un pin.

Paramétrer le pin 11 de la Jetson pour la configurer en tant que GPIO en input, afin d'acquérir l'état du bouton lorsqu'il est appuyé ou relâché.

## **Module vibreur :**

Le moteur du vibreur se contrôle en digital et non via un PWM, à connecter au pin 13 de la Jetson Nano. L'alimentation de ce vibreur se fait en 3,3V.

Paramétrer le pin 13 de la Jetson pour la configurer en tant que GPIO en output afin d'envoyer l'état voulu du vibreur pour contrôler son temps de vibration.

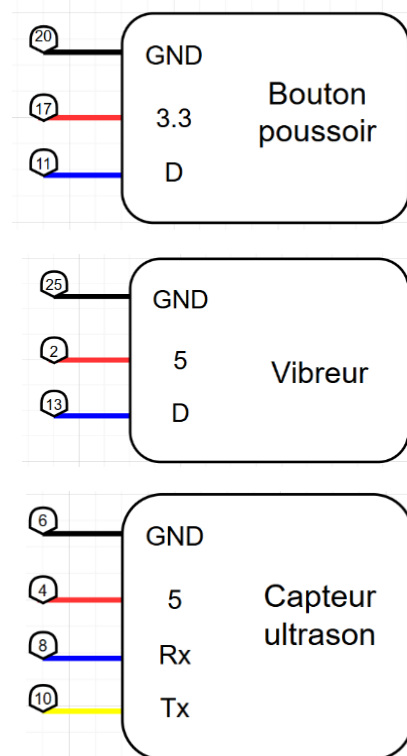
## **Batterie externe et interrupteur :**

Alimentation directe sur le port micro-USB de la Jetson Nano.

L'alimentation sera dimensionnée pour que le système puisse maintenir une durée de 1 heure pour ce prototype. Un câble munit d'un interrupteur sera également présent si l'utilisateur veut éteindre la canne.

Câblage des composants sur les pins de la jetson nano :

Sysfs	Name	Pin	Pin	Name	Sysfs
	3.3V DC	1	2	5V DC	
	I2C_2_SDA	3	4	5V DC	
	I2C_2_SCL	5	6	GND	
GPIO216	AUDIO_MCLK	7	8	UART_2_TX	
	GND	9	10	UART_2_RX	
GPIO50	UART_2_RTS	11	12	I2S_4_CLK	GPIO79
GPIO14	SPI_2_SCK	13	14	GND	
GPIO194	LCD_TE	15	16	SPI_2_CS1	GPIO232
	3.3VDC	17	18	SPI_2_CS0	GPIO15
GPIO16	SPI_1_MOSI	19	20	GND	
GPIO17	SPI_1_MISO	21	22	SPI_2_MISO	GPIO13
GPIO18	SPI_1_SCK	23	24	SPI_2_CS0	GPIO19
	GND	25	26	SPI_2_CS1	GPIO20
	IC2_1_SDA	27	28	I2C_1_SCL	
GPIO149	CAM_AF_EN	29	30	GND	
GPIO200	GPIO_PZO	31	32	LCD_BL_PWM	GPIO168
GPIO38	GPIO_PE6	33	34	GND	
GPIO76	I2S_4_LRCK	35	36	UART_2_CTSI	GPIO51
GPIO12	SPI_2_MOSI	37	38	I2S_4_SDIN	GPIO77
	GND	39	40	I2S_4_SDOUT	GPIO78



Sur le schéma ci-dessus, nous avons les différents composants branchés sur les pins de la Jetson nano. Ici ne sont pas représentés la caméra et les haut-parleurs car ils possèdent déjà leurs propres connexions qui ne sont pas présentes sur les pins.

## 2. Description des modules logiciels

La canne intelligente repose sur plusieurs modules logiciels, chacun ayant un rôle bien défini pour assurer les fonctionnalités de détection, d'analyse, et de retour d'information pour l'utilisateur. Voici les détails pour chaque module.

### 2.1 Module de capture d'images

- Configurer la caméra IMX219 via OpenCV pour capturer des images en continu.
- Utiliser OpenCV pour lire les images dans un flux vidéo continu. Ce flux sera ensuite transmis au module de reconnaissance d'objets pour le traitement.

- Mettre en place des exceptions pour vérifier que la caméra est bien connectée au démarrage. En cas de problème, un message d'erreur doit s'afficher et le système doit tenter de redémarrer la caméra.

## 2.2 Module de reconnaissance d'objets

- Charger un modèle CNN optimisé (comme ssd-mobilenet) avec TensorRT pour accélérer le traitement sur la Jetson Nano.
- À chaque image capturée, appliquer le modèle de détection pour identifier les objets. Le modèle doit être configuré pour identifier les obstacles dans l'environnement immédiat de l'utilisateur.
- Le niveau de détail de reconnaissance change en fonction du mode (marche/exploration). En mode marche, limiter la détection aux obstacles proches, tandis qu'en mode exploration, activer une détection plus étendue.
- Transmettre les objets détectés au module de gestion des retours pour que l'utilisateur soit informé en temps réel.

## 2.3 Module de traitement des données de distance

- En mode marche, les mesures sont prises en compte dans la détection d'obstacles immédiats, tandis qu'en mode exploration, elles indiqueront la distance de l'objet détecté.
- Convertir les relevés distance en mètres et centimètres pour donner un retour précis sur les objets détectés.
- On utilisera la bibliothèque « serial » et « time » pour programmer ce capteur à ultrason.

## 2.4 Module de gestion des modes de fonctionnement

- Capturer l'entrée du bouton associé aux modes. Une pression simple bascule entre les modes marche et exploration.
- En fonction du mode, activer ou désactiver certains modules, comme le capteur à ultrasons ou des fonctions du modèle de reconnaissance.
- Ajouter une sortie sonore indiquant le changement de mode pour informer l'utilisateur.

- On utilisera la bibliothèque « Jetson.GPIO » et « time » pour programmer ce bouton poussoir.

## 2.5 Module de gestion des retours (sonores et vibratoires)

- En fonction des données de détection, choisir un message vocal spécifique pour les obstacles et objets détectés.
- Selon la distance, ajuster l'intensité des vibrations. Par exemple, pour les obstacles entre 5 et 1 mètres, moduler l'intensité pour un retour progressif et informatif.
- En cas de détection, synchroniser les alertes sonores et vibratoires pour offrir une perception cohérente à l'utilisateur.
- On utilisera la bibliothèque « Jetson.GPIO » et « time » pour programmer le vibreur.

## 2.6 Module de correction d'erreurs au démarrage

- Au démarrage, vérifier que la caméra, le capteur à ultrasons, et la Jetson Nano sont bien initialisés.
- En cas de problème, redémarrer automatiquement le composant en échec et consigner l'erreur pour une vérification ultérieure.
- Envoyer une alerte sonore ou un message si un composant ne fonctionne pas correctement même après correction.



### 3. Diagrammes détaillés

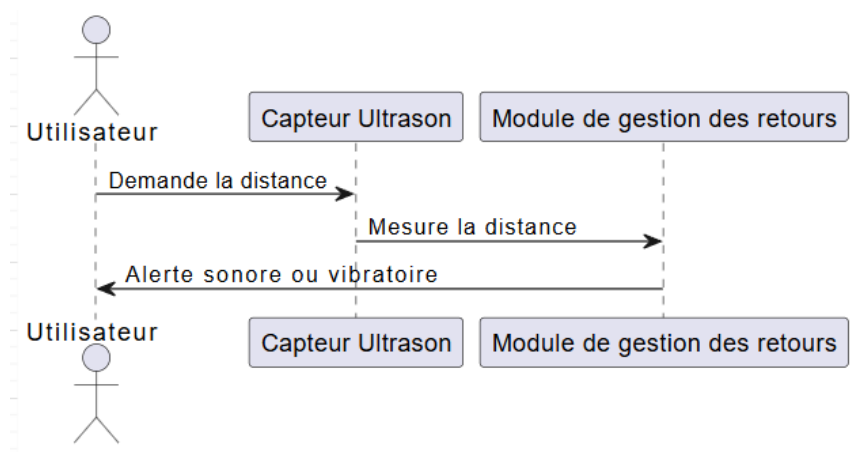
Cette section présente les schémas et diagrammes détaillés des connexions matérielles et des interactions logicielles, afin d'illustrer le fonctionnement du système dans son ensemble. Les diagrammes aident à visualiser le flux de données et les connexions entre les composants pour faciliter le développement et le dépannage.

#### 3.1 Diagrammes de séquences logicielles

Les diagrammes de séquences représentent le déroulement des actions entre les modules logiciels, depuis l'acquisition des données jusqu'aux retours utilisateur.

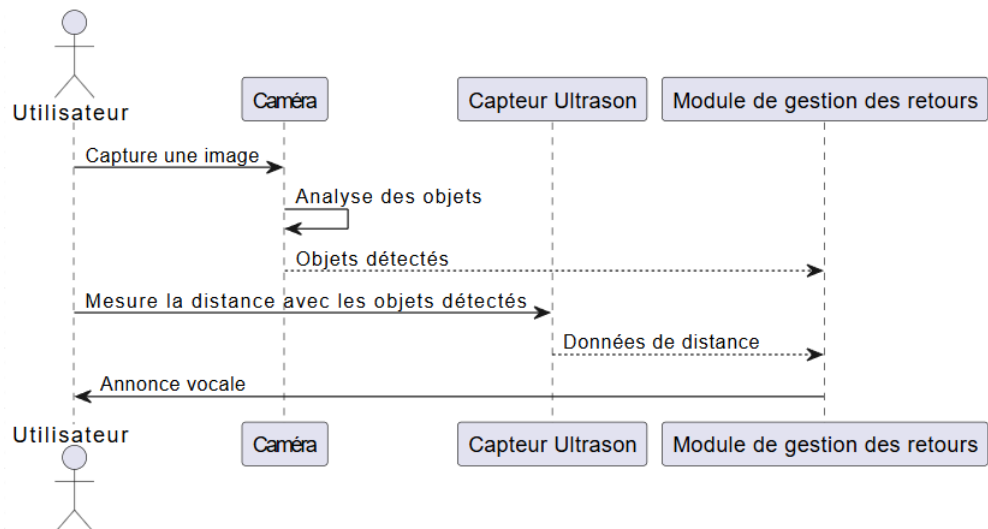
##### Séquence de détection d'obstacles en mode marche :

- L'utilisateur marche avec la canne et le capteur à ultrasons mesure constamment la distance.
- Le capteur à ultrasons mesure la distance et, si elle est inférieure à 5 mètres, transmet une alerte.
- En cas de détection, un retour sonore et/ou vibratoire est généré pour alerter l'utilisateur.



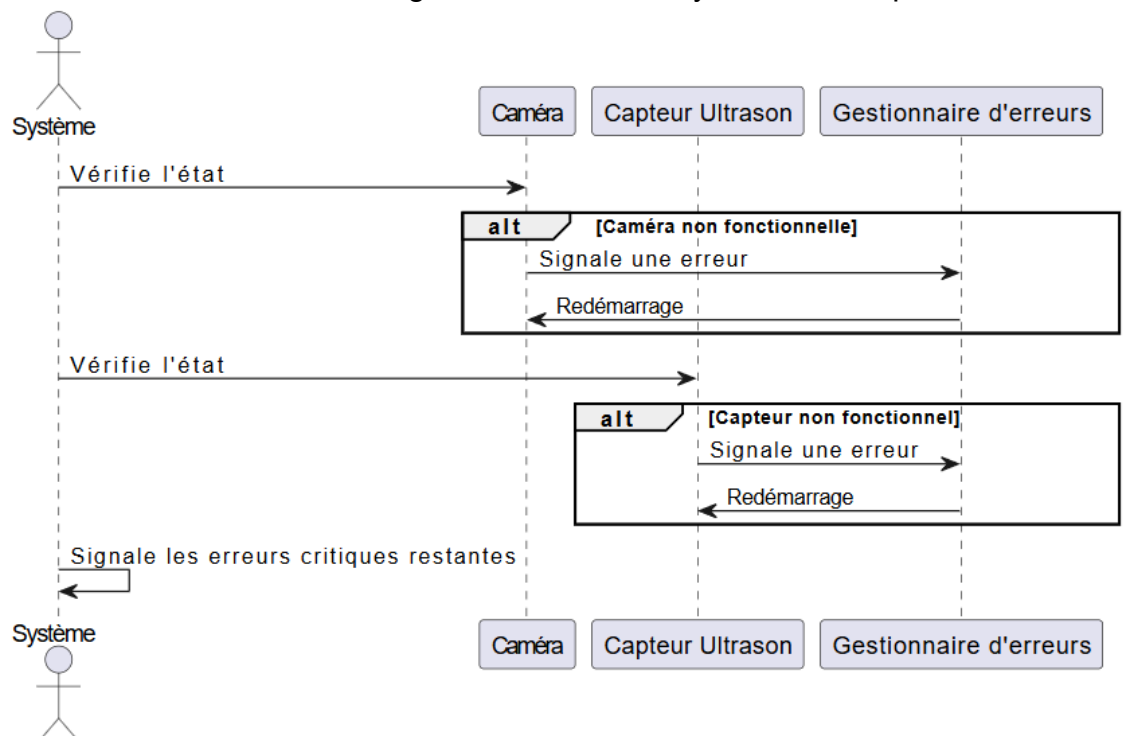
##### Séquence de détection en mode exploration :

- La caméra capture l'image et le module de reconnaissance identifie des objets spécifiques.
- Le capteur à ultrason effectue des relevés pour mesurer la distance avec l'objet détecté.
- En cas d'identification, un message vocal annonce l'objet détecté avec la distance de celui-ci.



### Séquence de correction d'erreurs au démarrage :

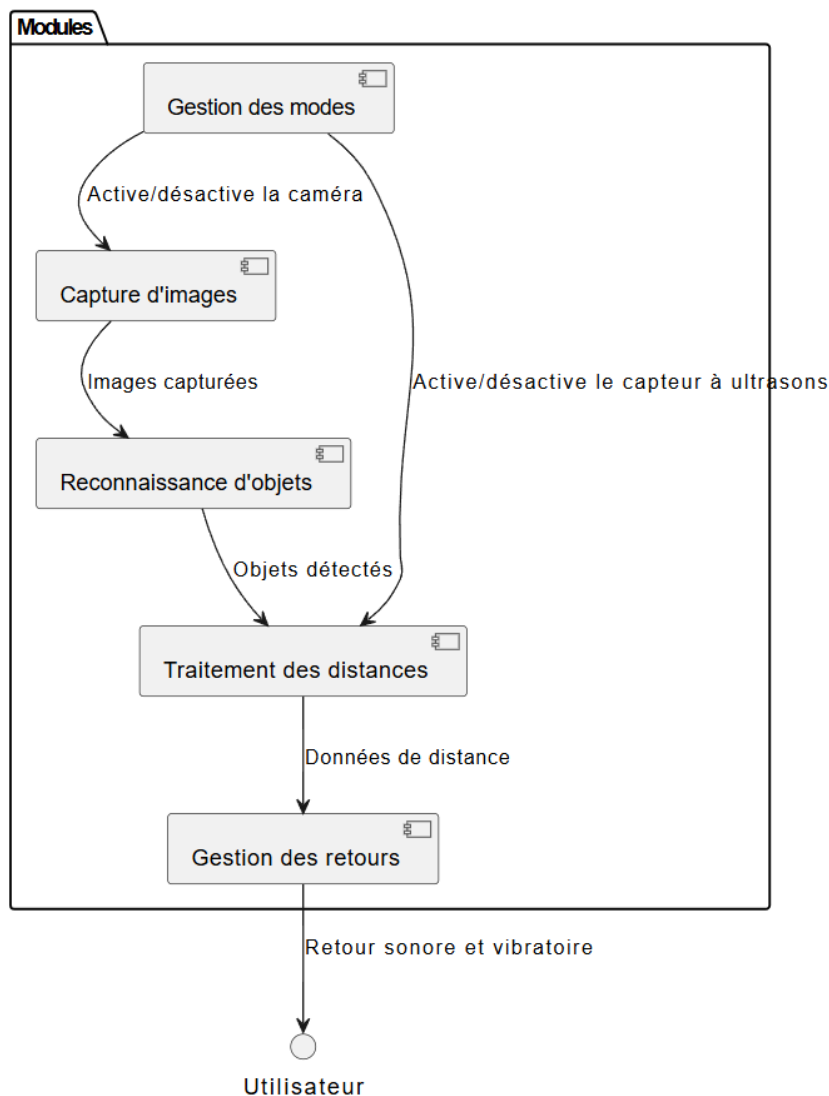
- Au démarrage, le système vérifie la connexion de chaque composant (caméra, capteur, etc.).
- Si un composant est en échec, le module tente un redémarrage de ce composant.
- En cas de nouvel échec, un message d'erreur est envoyé via le haut-parleur.



## 3.2 Diagramme d'interactions des modules logiciels

Ce diagramme montre les interactions entre les différents modules logiciels, illustrant les flux de données internes.

- Capture d'images : Les données capturées sont envoyées au module de reconnaissance d'objets.
- Reconnaissance d'objets et traitement de distance : Les résultats sont transmis au module de gestion des retours.
- Gestion des modes : Les boutons influencent les modules de reconnaissance et de traitement de distance, activant ou désactivant certains modules selon le mode choisi.
- Retour utilisateur : Les données traitées déclenchent les retours sonores et vibratoires, qui sont modulés en fonction des informations de détection.



## 4. Test des composants

Cette section présente les tests pour chaque composant matériel et logiciel du système, afin de garantir que l'ensemble des fonctionnalités fonctionne correctement et répond aux exigences du projet. Les tests sont organisés en trois étapes : vérification du matériel, validation des fonctionnalités logicielles, et gestion des erreurs.

### 4.1 Tests de fonctionnement

#### Capteur à ultrasons A02YYUW

- Test : Positionner des obstacles à des distances spécifiques et comparer les valeurs mesurées aux distances réelles.
- Critère de réussite : Les mesures sont précises dans un rayon de 5 mètres en mode marche, et les seuils d'alerte sont respectés.

#### Boutons poussoirs

- Test : Appuyer sur le bouton et observer le changement de mode.
- Critère de réussite : Les boutons changent instantanément de mode.

#### Module vibreur

- Test : Tester le vibreur avec différentes distances d'obstacle pour vérifier la modulation d'intensité.
- Critère de réussite : Le vibreur s'active et varie en intensité de manière progressive à mesure que la distance diminue.

#### Module de reconnaissance d'objets

- Test : Vérifier que le modèle de reconnaissance détecte les objets de manière fiable.
- Critère de réussite : Les objets courants dans l'environnement sont détectés.

#### Module de traitement des données de distance

- Test : Valider que les données de distance sont prises en compte correctement pour déclencher les alertes.
- Critère de réussite : Les alertes sonores et vibratoires sont activées selon les seuils configurés.

#### Module de gestion des modes de fonctionnement

- Test : Vérifier que le passage d'un mode à l'autre est fluide et entraîne les ajustements attendus.
- Critère de réussite : Les modes sont activés et les modules répondent en conséquence.

### **Module de gestion des retours**

- Test : S'assurer que les retours sonores et vibratoires sont bien synchronisés et reflètent correctement les obstacles détectés.
- Critère de réussite : Les retours sont cohérents et adaptés en intensité et en volume.

### **Module de correction d'erreurs au démarrage**

- Test : Simuler des erreurs de démarrage et vérifier que le module réagit correctement.
- Critère de réussite : Les erreurs sont détectées, et les tentatives de correction sont effectuées automatiquement.

### **Correction des erreurs et validation**

- Vérification des messages d'erreur : S'assurer que chaque composant envoie un message d'erreur en cas de problème.
- Test d'automatisation de correction : Confirmer que les erreurs sont détectées et que le système tente une relance.
- Validation finale : Après toutes les corrections, le système doit fonctionner sans erreur pendant une période d'au moins 1 heure sous des conditions normales d'utilisation.