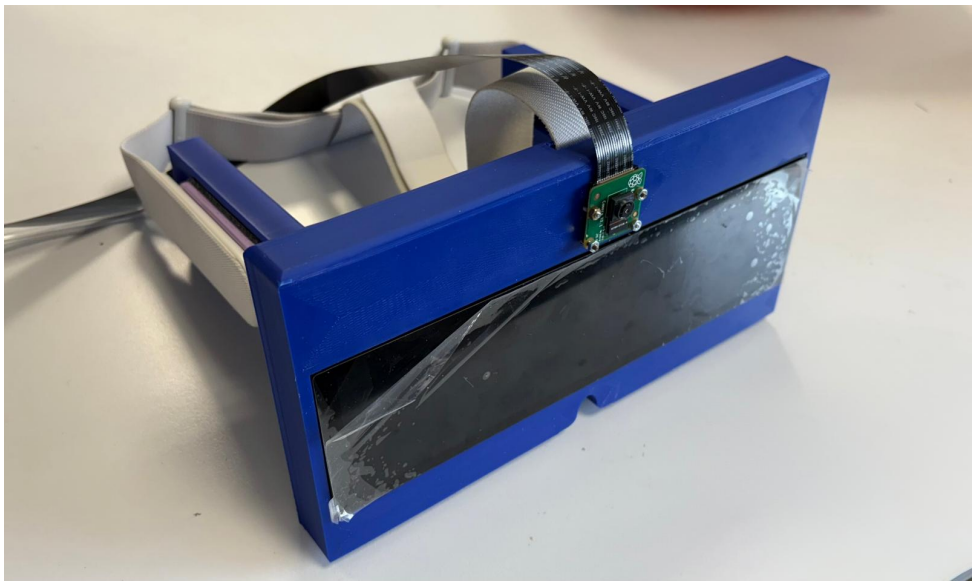
SORBONNE UNIVERSITÉ

Smart Glasses

Authors : Sarah Nacim Evinia Oualid Romane

Master 2 : MSR 2025-2026



SORBONNE UNIVERSITÉ

Table des matières

1 Introduction

1.1 Project Context

With the rise of wearable technology, smart glasses are undergoing significant development. Companies such as Meta, with their Ray-Ban Meta glasses, are now offering devices capable of capturing images and video, interacting via voice commands, and accessing AI-based assistants.

At the same time, within the field of assistive technology, several research projects and commercial products aim to support visually impaired or blind individuals specifically through obstacle detection, scene description, and the reading of visual information. Furthermore, recent breakthroughs in artificial intelligence, particularly in natural language processing (NLP) and computer vision, are opening up new possibilities to enhance visual assistance tools.

1.2 Objectives

In this context, our project follows a similar path, but with a specific focus on social interaction and assistance. The objective is to design smart glasses capable of detecting both the emotions of the person facing the user and any obstacles within the environment.

However, to ensure reliable operation and straightforward interaction, these two features do not run simultaneously. The system is organized into different operating modes, accessible through an interface of physical buttons integrated into the glasses. This allows the user to switch between "obstacle detection" and "emotion detection" modes, depending on their immediate needs.

This approach simplifies data processing, improves the clarity of haptic feedback, and provides the user with greater control over the device's behavior.

2 Materials Used

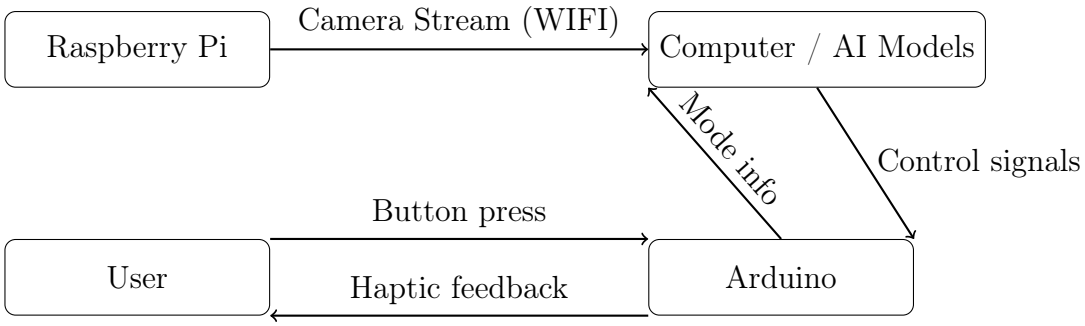
Component	**Number**	**Reference / Model**	**Description / Role**
Camera	1	Camera Raspberry module 2	Sony IMX219 8-megapixel sensor
Raspberry Pi	1	Raspberry Pi 3B+	Used to process image data and send it to the computer
Arduino	1	Arduino Uno	Used to control the haptic feedback system (ie pneumatic actuators...)
Differential Pressure sensor	2	DFRobot SEN0343	Used to detect pressure into balloons
Motor driver	3	L298N	Used to control 2 motors per driver
Valves	2	DFR0866	Used to switch between inflate and deflate

Component	Number	Reference / Model	Description / Role
Pumps	4	DFRobot 370 Mini Vacuum Pump	2 used to inflate and 2 to deflate balloons
Balloons	2	Standard party balloons	Used to do haptic feedback
Push buttons	2	Standard push buttons	Used to switch between modes
Computer	1	PC linux	Used to run the AI models for emotion and obstacle detection
Battery 12V	1	lead acid battery 12V	Used to power the pumps
Battery 5V and <2.5A	2	Standard Power bank 5V	Used to power the Raspberry Pi and Arduino

TABLE 1: Main hardware components used in the project

3 Implementation

3.1 Overall Architecture



3.2 Camera and Raspberry pi

Initially, we attempted to implement a stereo camera system using an Arducam module to obtain depth information directly from two synchronized camera feeds. However, the image quality was insufficient for reliable image processing, which led to poor performance in depth estimation. Moreover, the Raspberry Pi 4B frequently crashed when using the Arducam module, likely due to insufficient processing power and high resource demands of the stereo setup. This instability made the configuration unreliable for real-time operation.

As a result, we switched to using a single camera, combined with AI processing on the computer block to infer depth information from the captured images. This approach allowed us to maintain accurate obstacle and emotion detection while simplifying the hardware setup.

For transmitting the camera stream, we initially experimented with USB gadget mode on a Raspberry Pi 4B. However, this setup proved unstable and presented power supply challenges, as the USB port could not simultaneously provide enough power and transmit data reliably.

To overcome these issues, we opted for a Raspberry Pi 3B+ equipped with Wi-Fi for streaming the camera feed to the computer. Although this solution introduces a small amount of latency, it provides a stable and functional system for real-time image analysis.

All the configuration steps and installations required to use the Raspberry Pi 3B+ with Wi-Fi for the system are detailed in the file `procedure_camera.md`.

3.3 Pneumatic/haptic system

3.4 Communication Arduino - Computer

Communication between the Arduino and the computer is carried out via a serial port (USB). The Arduino sends mode commands to the PC using the format `MODE:X`, where X represents the mode selected by the user through the push buttons.

A launcher program (`launcher.py`) running on the computer continuously listens to the serial port. Upon receiving a mode command, it manages the start and stop of the corresponding Python processes :

— **Mode 1** : Launch of the emotion detection module (`emotion_detection2_pneu.py`)
— **Mode 2** : Launch of the obstacle detection module using YOLO (`objet.py`)
— **Mode 3** : Stop of all active scripts (neutral mode)

This orchestration system ensures that only one AI module runs at a time, optimizing the use of computational resources and simplifying the delivery of haptic feedback to the user. In the event of an interruption (mode change), the launcher cleanly stops the current process before starting a new one.

3.5 AI Models and Processing

The algorithms described below are not executed manually from a terminal, but are launched through the `launcher.py` script, which manages communication with the Arduino and handles mode switching. To check the required installations needed to run the scripts, refer to the `requirements.txt` file.

Emotion Detection

The emotion detection module uses the **FER** (Facial Expression Recognition) library, which is based on deep neural networks. FER detects seven facial emotions : happy, surprise, sad, angry, fear, disgust, and neutral.
To simplify the haptic feedback system and reduce computational load, the seven emotions are grouped into three main categories :

— **Happy** : happy, surprise
— **Sad** : sad, fear, disgust
— **Neutral** : angry, neutral

A confidence threshold of 0.35 is applied to ignore low-confidence detections. Each reduced emotion is associated with a confidence score computed from the FER probabilities.

To allow real-time processing, several optimizations are applied :

— **Frame skipping** : only one frame out of three is processed (FRAME_SKIP = 3)
— **Resizing** : images are scaled to 60% of their original size
— **Centered detection** : only the face closest to the center of the image is processed

Each reduced emotion triggers a specific haptic pattern using the pneumatic balloons :

— **Happy** : right balloon inflates and deflates, left balloon inactive
— **Sad** : left balloon inflates and deflates, right balloon inactive
— **Neutral** : both balloons remain inactive

The Arduino receives the commands through a serial connection and controls the pneumatic system.

The program receives JPEG-compressed video frames through a UDP socket from the Raspberry Pi. Each frame is decoded and processed by FER at regular intervals. The corresponding haptic commands are then sent to the Arduino. A live display shows the detected face and the recognized emotion.

The emotion is detected only once in order to avoid overstimulating the user.

Obstacle Detection

The obstacle detection module is based on the **YOLO** (You Only Look Once) object detection model combined with the **Depth Anything V2** network for depth estimation. YOLO is used to detect relevant objects in the scene, while Depth Anything provides a depth map that allows estimating the relative distance of each detected object.

Only a subset of COCO classes is considered, corresponding to potential obstacles for the user (persons, chairs, beds, couches, benches, handbags, suitcases).

The processing pipeline is structured as follows :

— **Frame acquisition** : The program receives JPEG-compressed video frames via a UDP socket from the Raspberry Pi.
— **Object detection** : YOLO detects objects and provides bounding boxes and segmentation masks.
— **Depth estimation** : Depth Anything computes a dense depth map from the RGB image.
— **Depth aggregation** : For each object, the average depth is computed using its segmentation mask.
— **Region of Interest (ROI)** : Only objects located inside a trapezoidal region in front of the user are considered.
— **Temporal filtering** : An object must be detected continuously for at least one second to be considered stable.
— **Side selection** : Only one object per side (left and right) is kept, corresponding to the closest obstacle.
— **Intensity computation** : The closest object is assigned an intensity of 100, while the others are scaled proportionally.
— **Smoothing** : Haptic intensity variations are limited to avoid sudden pressure changes.

The resulting haptic commands are sent to the Arduino, which controls the pneumatic system to provide feedback to the user. The left and right balloons correspond to obstacles detected on the respective sides of the user.