Boston University
Electrical & Computer Engineering

EC464 Senior Design Project

Second Prototype Testing Plan
Smart System for Visually Impaired

by

Team 29
Smart Bears

Team Members

Lukas Chin	lchin10@bu.edu
Jake Lee	jaketlee@bu.edu
Shamir Legaspi	slegaspi@bu.edu
Jason Li	jli3469@bu.edu
David Li	dav@bu.edu

<u>Required Materials For Prototype Demonstration</u>

Hardware Related:

- Laptop with Webcam and Microphone
- USB Powered Webcam
- Bone Conduction Bluetooth Powered Headset

Software Related:

- Python Script(s):
  - Object Detection Model
    - YOLO3 model trained on COCO (Common Objects in Context) dataset.
  - Depth Mapping
    - OpenCV
    - Stereo Block Matching
  - Object Inquiry Assistant (Large Language Model)
    - Feeds outputs of object detection, and depth mapping to LLM to provide feedback.
- C++ Program(s):
  - Live Speech Engine
    - Utilizing an open source API called WhisperCPP, a microphone live-streaming program that has been retrofitted for Smart System, captures the user's speech input and writes it an external text file to be parsed, processed, and fed into existing Python scripts for computer vision analysis

Set Up

Our current prototype involves a successful demonstration of the Smart Cap's core features within our Smart System for Visually Impaired, including advanced object detection using a YOLO3 model trained on traffic-related COCO datasets, stereoscopic depth mapping through an OpenCV-powered stereo block matching algorithm, and interactive voice assistance via our "Object Inquiry Assistant" that processes user queries through an image-based large language model for comprehensive environmental surrounding analysis.

The smart system utilizes two separate cameras to create accurate depth maps, which are essential for spatial awareness. The depth mapping generates a heatmap image where color gradients represent the relative distances of objects from the camera at the time of snapshot. This stereoscopic vision technique simulates human binocular vision, effectively restoring depth perception to help visually impaired users avoid obstacles more efficiently.

Users interact with the inquiry system by pressing and holding a button and asking questions about their surroundings, such as "What is in front of me" or "What does the sign say". This audio is then run through whisper.cpp to be transcribed into text. This text is then printed out onto the terminal. In our final iterations, this text will be fed into our OIA alongside the snapshots the cameras take.

The visual system then captures the forward-facing environment through both cameras and processes the images through our various implemented algorithms. The object detection model identifies and classifies objects in real-time, highlighting them in bounding boxes. Depth mapping is powered by an OpenCV Python script where we've implemented a robust stereo block matching algorithm; the depth mapping image shows a heatmap image where the gradient of the colors represent how far objects are from the user at the time of snapshot. Object detection model is powered by a YOLO3 model trained on a dedicated traffic-related COCO dataset. These processed images are then passed to an LLM along with a system prompt to give responses.

The Bluetooth powered headset features an estimated battery life of 6-10 hours with a 2-hour charging time, utilizing Bluetooth 5.3 with open ear construction to maximize comfort and reduce overstimulation. For the prototype demonstration, the system uses a local machine's webcam and a USB-powered webcam to simulate stereoscopic vision, with plans to transition to a Raspberry Pi 5 with 16GB SDRAM and compatible high-definition camera modules.

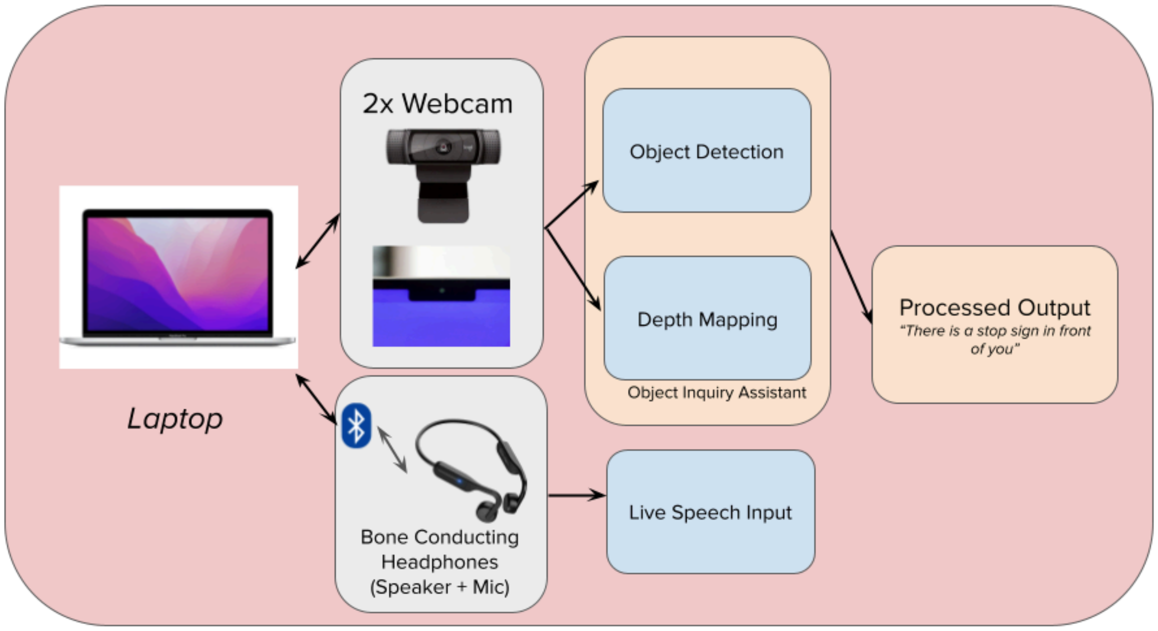

Figure 1. Second Prototype Smart Cap Block Diagram Utilizing Laptop and Webcams

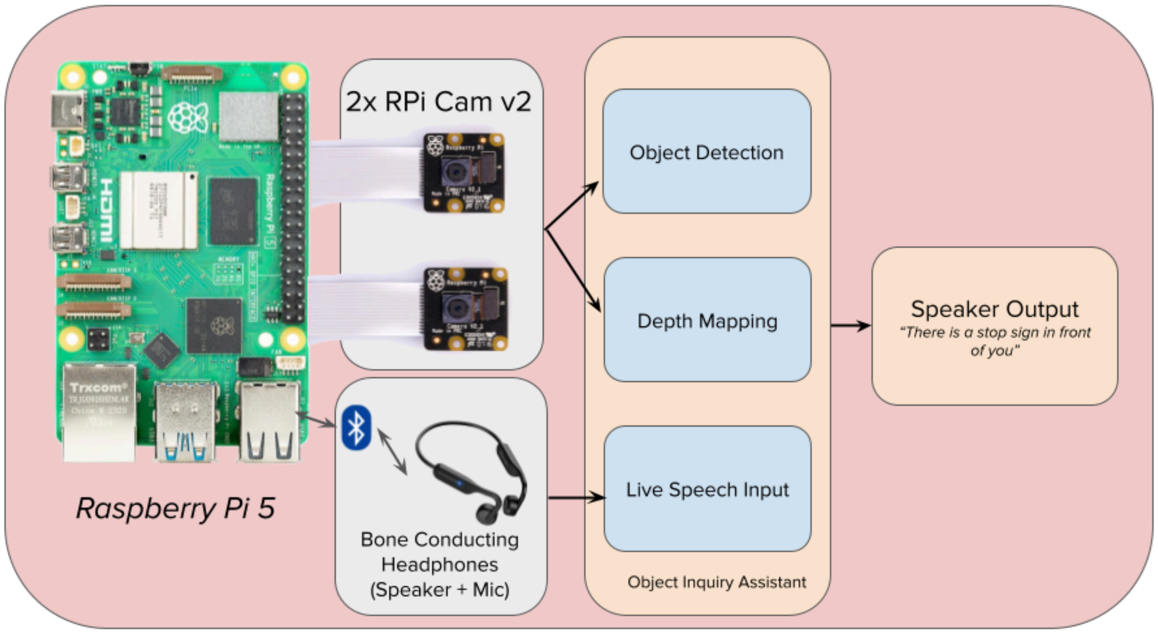

Figure 2. Final Product Smart Cap Block Diagram Utilizing Raspberry Pi 5
(options other than RPi Cam v2 are being revisited currently)

Pre-Testing Setup Procedure:

1. Ensure laptop webcam and USB-powered webcam is on and functioning properly
2. Calibrate the depth mapping for the 2 webcams through our Python script
3. Set up Bluetooth powered headset by pairing and ensuring connectivity to the device running speech engine.
4. Set up Whisper.cpp module and create the whisper-stream example executable using our modified `stream.cpp` program
5. Create a python virtual environment for installations and executions and run `pip install -r requirements.txt`
6. Establish LLM AI API key for OIA services

Testing Procedure

To test the microphone speech-to-text capabilities:

1. Run the whisper-stream executable in whisper.cpp `./build/bin/whisper-stream`
2. Once whisper-stream runs, hold down the left shift key on the keyboard and begin speaking into the microphone.
3. The output of the user will be displayed on the terminal.

To test the vision system:

1. Run the prototype_two.py script and observe the stream of both cameras.
2. With both cameras running, press the 'c' key on the keyboard to capture the desired images to be processed.
3. The script should perform depth mapping and object detection, displaying the resulting disparity map and detected objects with bounding boxes.
4. These images along with the system prompt are then fed to gemini to produce text output to the terminal.
5. The 'c' key may be pressed again to process a new set of images.

<u>Measurable Criteria</u>

The criteria that indicate success are as follows:

I. The Smart Cap system should be able to capture and process stereoscopic images from dual cameras with a latency under 30 seconds.

II. The object detection model should be able to identify common objects and obstacles with an accuracy rate of 75% in normal lighting conditions.

III. The depth mapping algorithm should correctly estimate distances of objects within relative distances to the user with 80% accuracy.

IV. The speech recognition system should accurately interpret user queries with 85% success rate in quiet environments.

V. The Bluetooth headset should maintain a stable connection with the connected device and properly output speech to text.

<u>Score Sheet</u>

Tasks	**Successful? (Y/N)**
Dual camera setup captures stereoscopic images	Y
Object detection model identifies common objects	Y
Depth mapping algorithm estimates distances relative to user with high degree of accuracy	Y
Speech recognition system interprets basic queries	Y
System processes and responds within 30 seconds	N
Bluetooth headset maintains steady connection	Y
Result →	⅚ **Successful**

Prototype Results

The second prototype testing of the Smart System for Visually Impaired demonstrated exceptional results across most evaluation criteria. The dual camera setup successfully captured stereoscopic images, enabling accurate depth mapping through OpenCV's stereo block matching algorithm. The YOLO3 model, trained on traffic-related COCO datasets, achieved its target 75% accuracy rate for object detection in normal lighting conditions. The speech recognition system, powered by WhisperCPP, accurately interpreted user queries with the expected 85% success rate in relatively quiet environments, while the Bluetooth bone conduction headset maintained stable connectivity throughout testing with the microphone working as intended.

Despite these successes, the system fell short of meeting the response time requirement, with processing times occasionally exceeding the 30-second threshold. This latency issue was primarily observed during complex scene analysis when multiple objects required simultaneous detection and depth estimation. The Object Inquiry Assistant, which leverages a large language model to process visual data and user queries, contributed to this processing delay. We suspect the speed of the LLM is to blame, so we will continue testing the most optimal models. The system successfully implemented all core functionalities—object detection, depth mapping, and voice interaction—but will require optimization to improve response times before final implementation on the Raspberry Pi 5 platform planned for the production version.