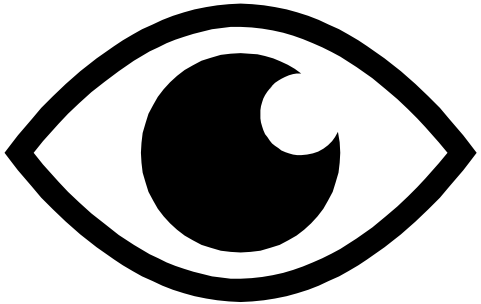
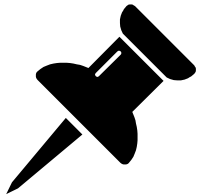


SightMate

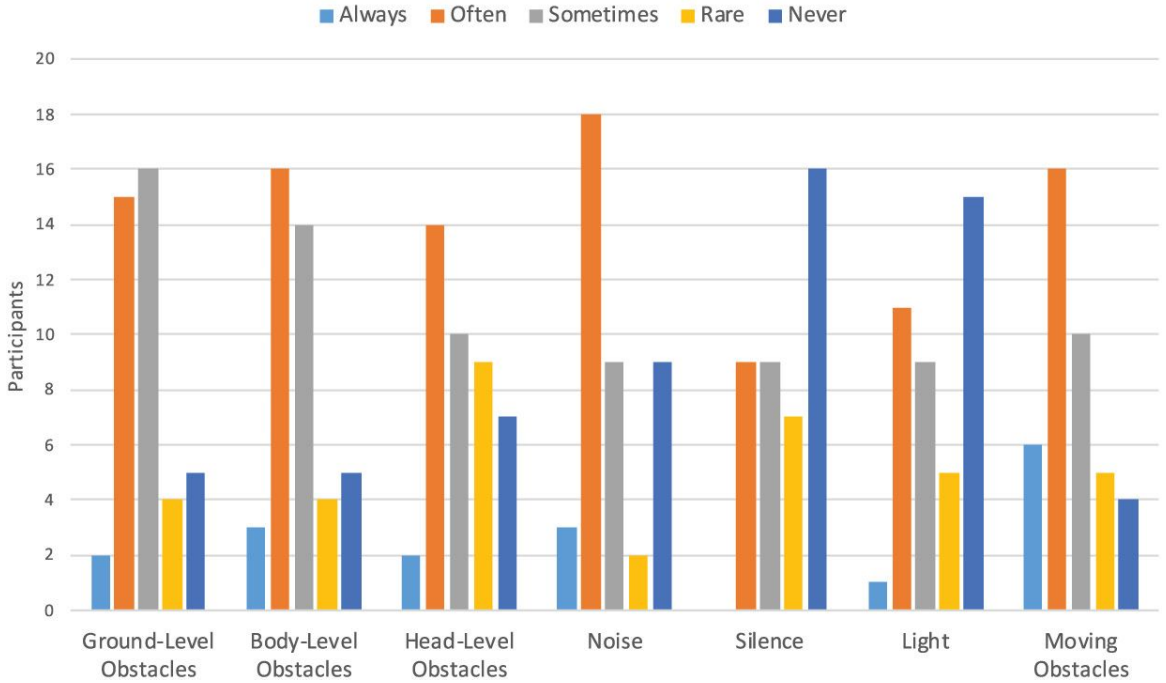
by Meera Pandya, Josh Joung, Shakthi Angou

Introduction



- **Background:** Challenging for VI people to navigate indoor spaces due to changing layout, unclear directions, and many obstacles
- **Issue:** Guide dogs are inaccessible, and reliance on sighted guides makes independent navigation difficult
- **Goal:** Build an automated wearable navigation system that alerts the user of objects near them

Common Hazards



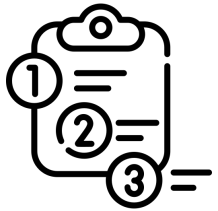
https://journals.sagepub.com/doi/10.1177/0264619619833723

Use Case

- Accessible and affordable alternative for guide-dogs or sighted guides in indoor navigation
- Will be used along with a cane, which is the most commonly used assistive device for the visually-impaired
- Project scope restricted to well-lit indoor spaces with minimal to medium-level object crowding

https://www.raspberrypi.com/news/wearable-pi-zero-camera/

Requirements



Battery Life

minimum of **4 hours** because a guide dog usually takes a break every 4 hours.

Accuracy

at least **70%** because it is the minimum qualification to become a guide dog.

Measuring Distance

minimum of **2 meters** because a user would need enough distance to avoid the obstacle

Weight

no more than **200 grams**, battery pack may be offloaded to waist if needed

Recognition Delay

less than **2.5 seconds** to permit 2-meter detection (blind pedestrians walk at .8 m/s)

Noise Detection

user should be able to **hear surrounding noises** regardless of the audio device

Challenges & Solutions

1. Detecting an object with an accuracy of at least 70%

Test several options of existing object recognition models and build off of the one with best results

2. Handling consecutive changes in directions

Create a threshold for which the user needs to wait for a few seconds to detect the object before moving

3. Minimal delay between device modules

Reduce data latency within each module with a simple architecture to optimize performance time

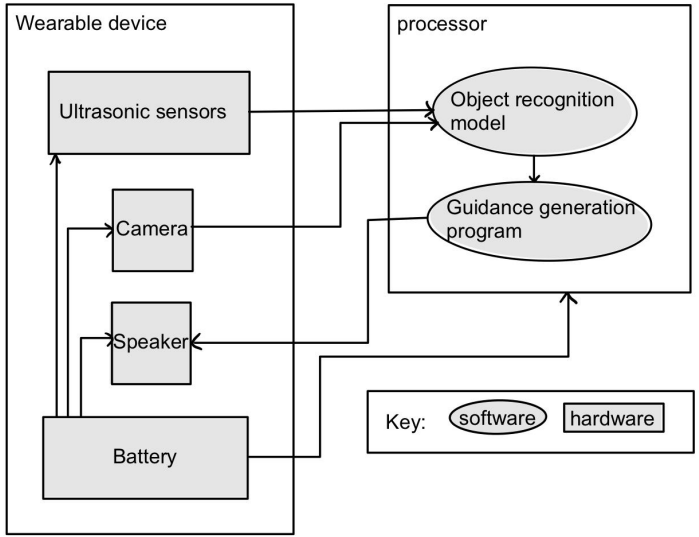
4. Connecting a sensor to an object recognition program

Use an object recognition program that incorporates data collected from depth detectors or implement the OR model to identify objects by distance

5. Molding into a stable and comfortable device

Conduct weight testing to verify that the device is wearable without significantly restraining movement

Architecture



Camera → Detection → Transmission → Audio

Tools

Software

Tensorflow

PyTorch

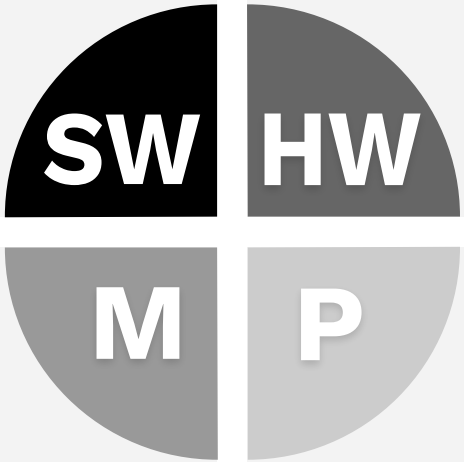
OpenCV

Hardware

Raspberry-Pi

Camera

Ultrasonic Sensor



Scikit-learn

YOLOv7

Module

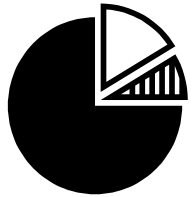
VSCode

Github

Raspberry-Pi OS

Platform

Testing & Verification & Metrics



Testing	Verification	Metrics
Object recognition model	Identify the **closest** object	> 70% on identifying an object
Distance detection module	Measure the **distance** of the closest material	>70% on noticing the existence of the closest object within 2m
Text-to-speech module	Able to hear both the audio output and **background noise**	> 90% on a noise testing to identify both speech and background noise
Vibration module	**Vibrate** based on the distance of an object	>90% on vibrating once the device detects the object within 2m
Device controls (buttons)	Turns on and off based on the **user input**	>90% on testing whether switches correspond to the user input
Module integration	The **recognition delay** is minimal The weight is distributed evenly	< 2.5s to recognize an object >90% on user survey on the distribution of weight
Functionality	The device **detects and alerts** the closest object within 2m of range from the user	>70% on the accuracy

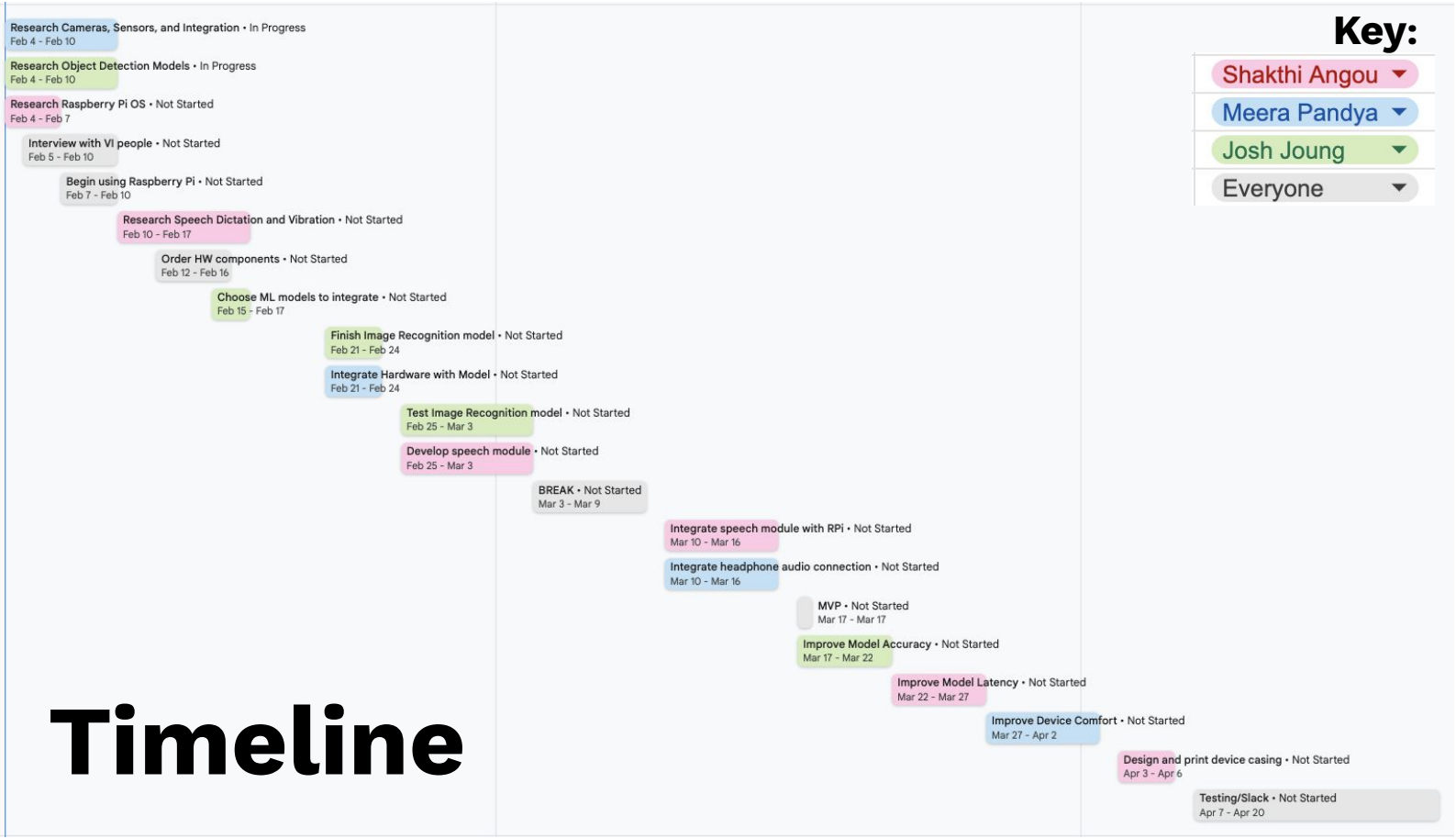
Risk Mitigation

- **RPi Heatsink Issue →** Pivot to using network connection (such as ESP_NOW) with an external processor for ML models
- **Device Overweight →** Offload battery and/or RPi to a pack worn on the user's waist
- **Poor Sensor Integration →** Modify recognition model to estimate distances using camera data

Division of Labor

Tasks	Detail	PIC
Hardware Implementation	Raspberry Pi Setup	Meera Pandya
	Camera, Audio device, Ultrasonic sensor, RPi Integration	
Software Program Implementation	Object Recognition ML Model	Josh Joung
	Raspberry Pi OS	
Software Module Implementation & Device Integration	Speech and Vibration Module	Shakthi Angou
	Device Design	

Timeline


Research Cameras, Sensors, and Integration • In Progress
Feb 4 - Feb 10
Research Object Detection Models • In Progress
Feb 4 - Feb 10
Research Raspberry Pi OS • Not Started
Feb 4 - Feb 7
Interview with VI people • Not Started
Feb 5 - Feb 10
Begin using Raspberry Pi • Not Started
Feb 7 - Feb 10
Research Speech Dictation and Vibration • Not Started
Feb 10 - Feb 17
Order HW components • Not Started
Feb 12 - Feb 16
Choose ML models to integrate • Not Started
Feb 15 - Feb 17
Finish Image Recognition model • Not Started
Feb 21 - Feb 24
Integrate Hardware with Model • Not Started
Feb 21 - Feb 24
Test Image Recognition model • Not Started
Feb 25 - Mar 3
Develop speech module • Not Started
Feb 25 - Mar 3
BREAK • Not Started
Mar 3 - Mar 9
Integrate speech module with RPi • Not Started
Mar 10 - Mar 16
Integrate headphone audio connection • Not Started
Mar 10 - Mar 16
MVP • Not Started
Mar 17 - Mar 17
Improve Model Accuracy • Not Started
Mar 17 - Mar 22
Improve Model Latency • Not Started
Mar 22 - Mar 27
Improve Device Comfort • Not Started
Mar 27 - Apr 2
Design and print device casing • Not Started
Apr 3 - Apr 6
Testing/Slack • Not Started
Apr 7 - Apr 20
Key:
Shakthi Angou
Meera Pandya
Josh Joung
Everyone