# Guardian Eyes
# Final Presentation

김준호, 노길호, 이경준, 설진환, 이재필

# Contents

## Recap & Changes

Recap Idea

Rescope from midterm

## Demonstration

Usage Scenarios

Demo Video

## Technical Details

System architecture Overview

Technical Challenge

Approach & Solution

## Project Management

Timeline

Role & Contribution

## Conclusion

Conclusion

Lesson learnt

# Recap & Changes

Part.1

# Recap

**Motivation**

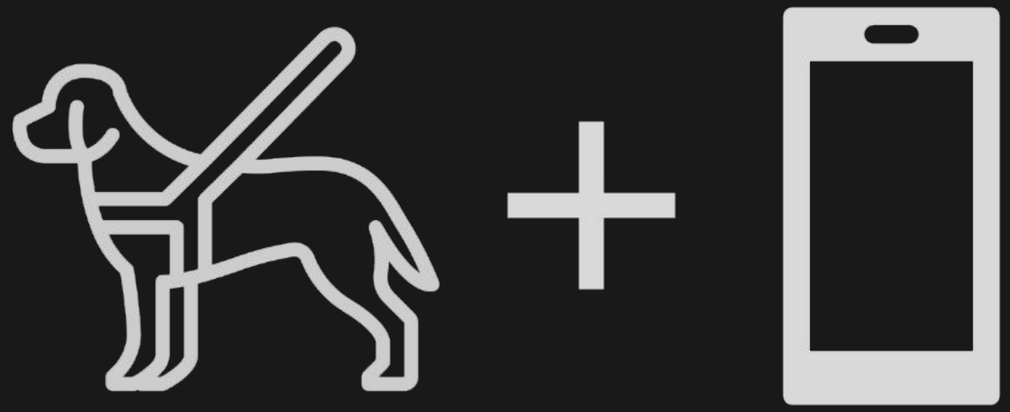
**: 모바일 환경을 이용한 시각장애인들을 위한 안내 도우미**

**Proposed Idea**

**: Tensorflow + ARcore + Sound feedback**

**Novelty**

**: Vision-based approach**



# Rescope from midterm

**Before midterm**

계단 등 바닥 장애물들을 **custom deeplearning model**을 통해 감지

**After midterm**

**Custom deeplearning model**을 개발하지 않고, **Google AR Core**의 기능을 이용해 지면 감지를 구현

# Demo

Part.2

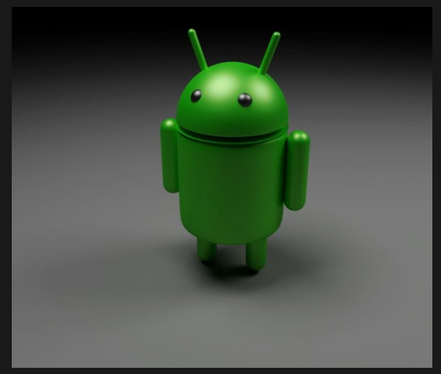
# Demo

# Technical details

Part.3

# Technical details

## DEVELOPMENT ENVIRONMENT



ARCore

OpenCV

OpenGL

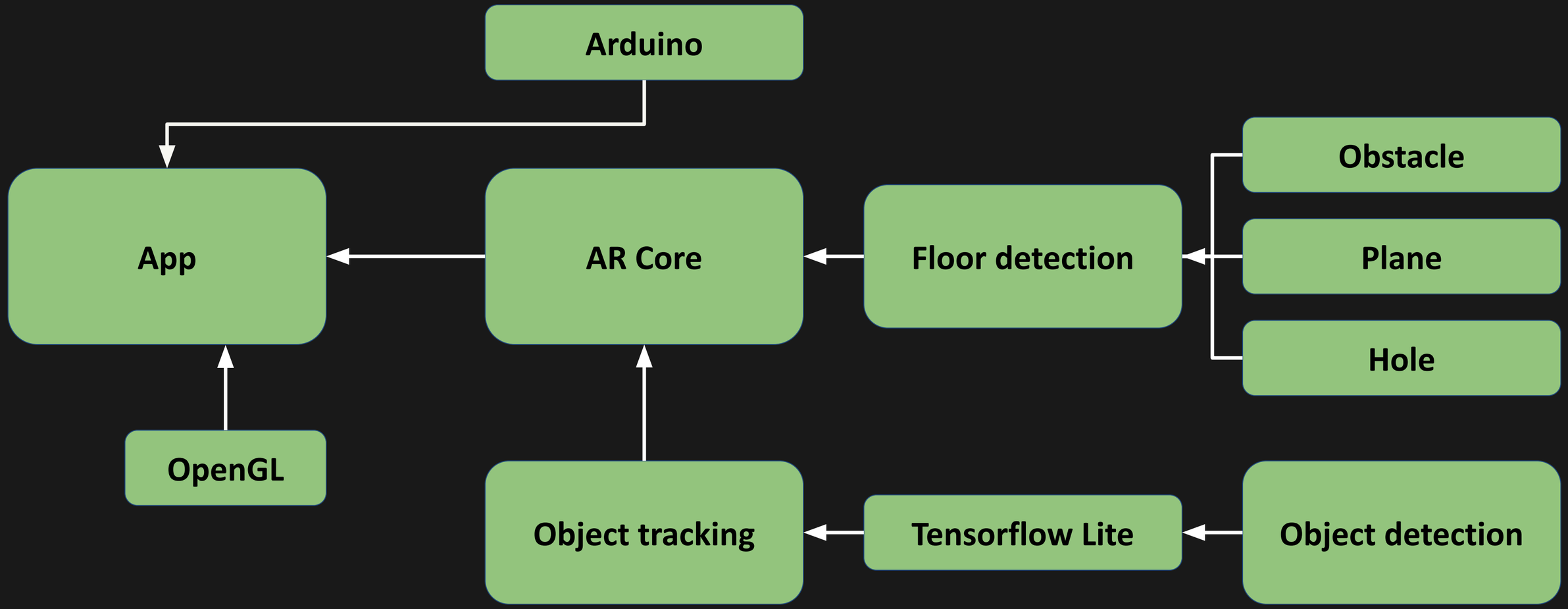
GitHub

TensorFlow Lite

ARDUINO

# Technical details

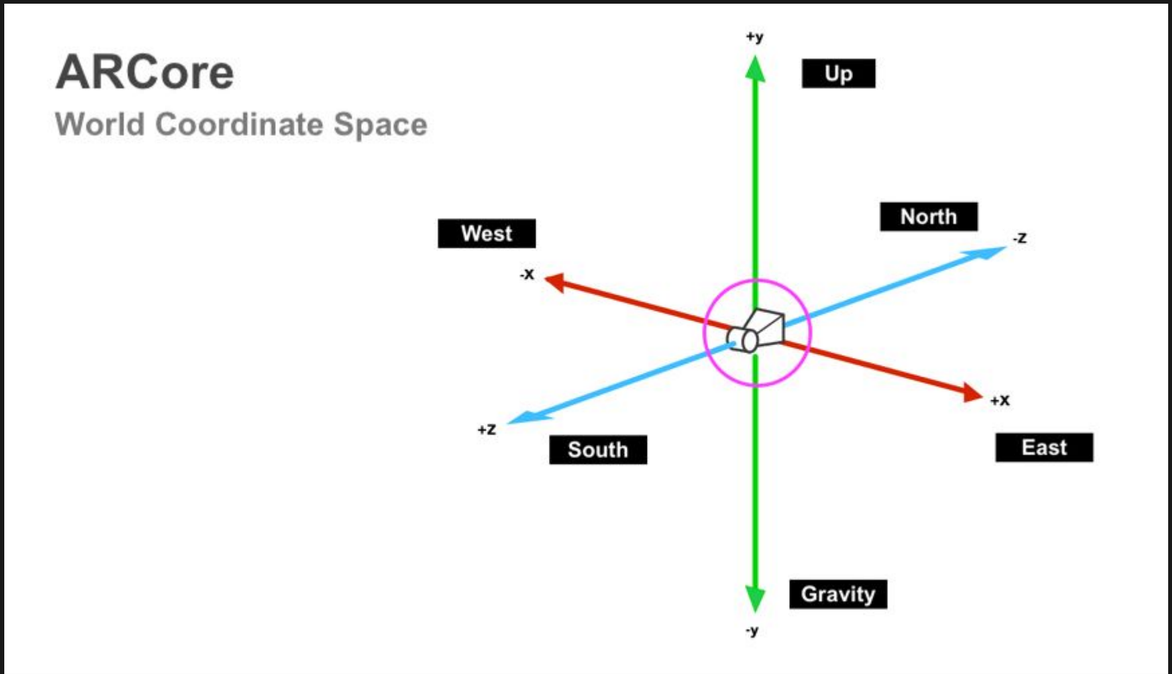
## SYSTEM ARCHITECTURE



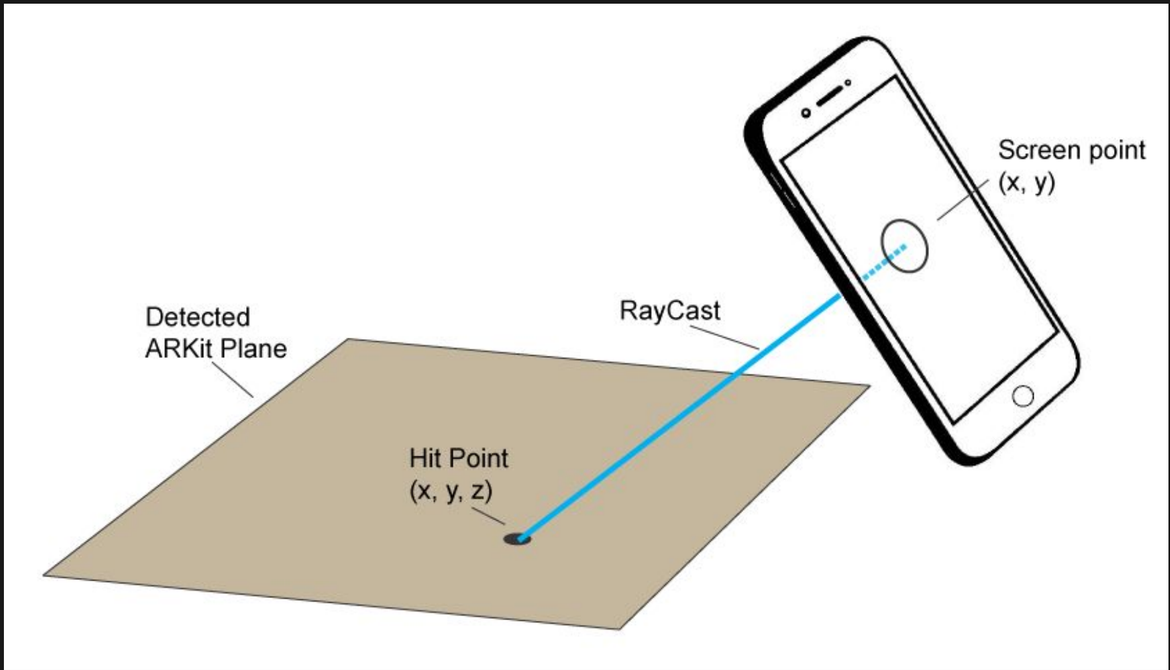
# Technical details

## (1) AR CORE : Floor detection

### Approach




Real-world 3d coordinate system



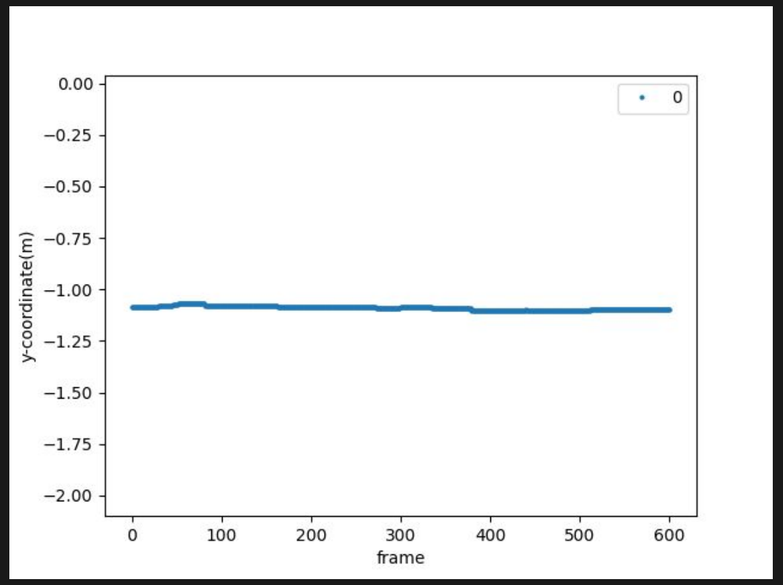

Get location of plane

# Technical details

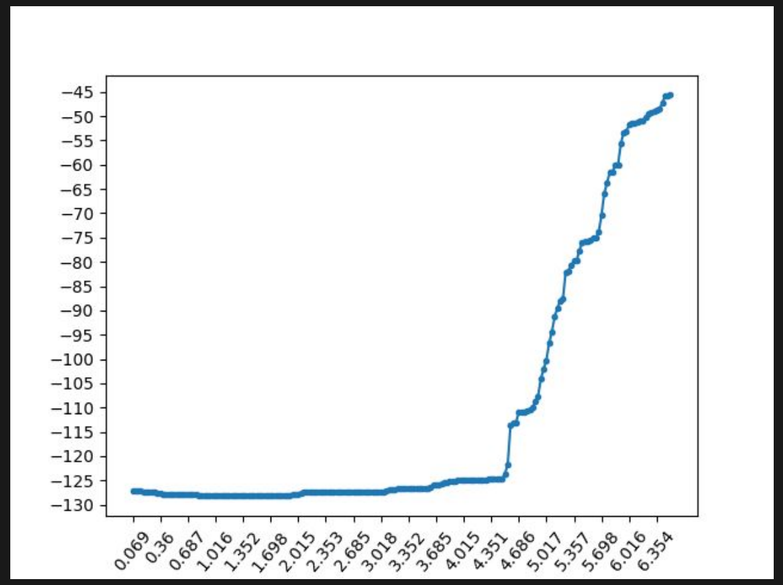
## (1) AR CORE : Floor detection

### Data

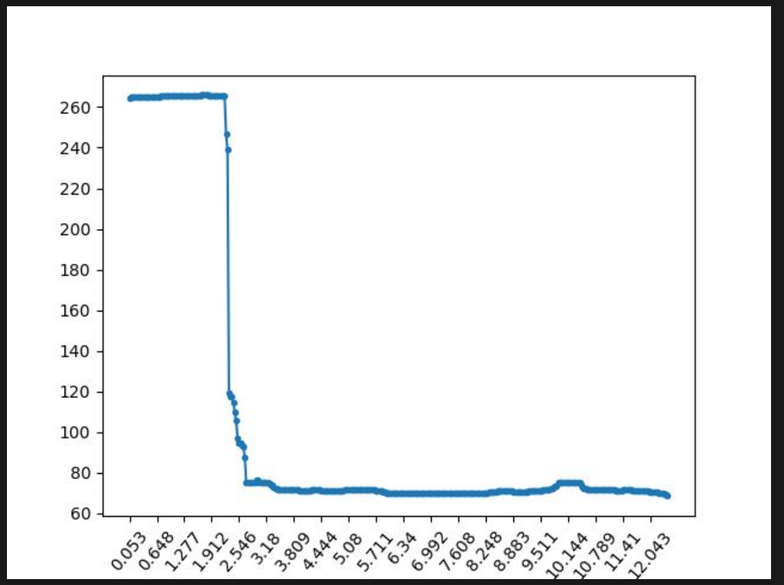
### Plane




### Obstacle




### Hole



# Technical details

## (1) AR CORE : Floor detection

**Implementation**

Slope Threshold

```
If (data.slope > threshold1)
    state = ‘Obstacle’
else if (data[start] - data[end] > threshold2)
    state = ‘Hole’
else
    state = ‘Plane’
```

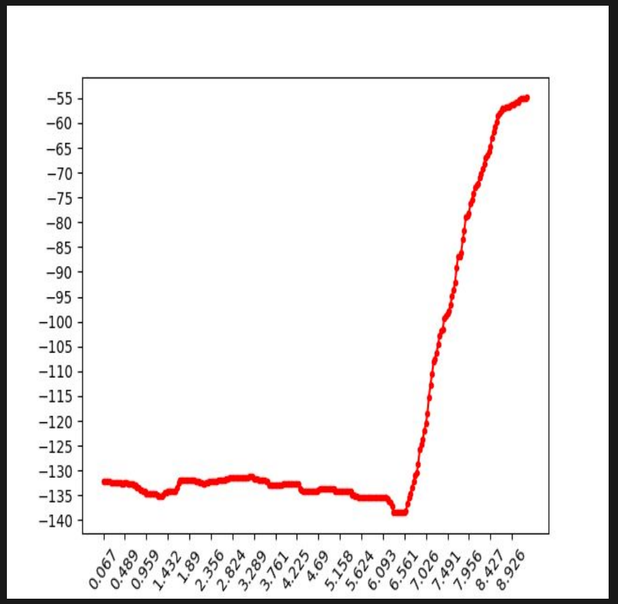
Height difference

# Technical details

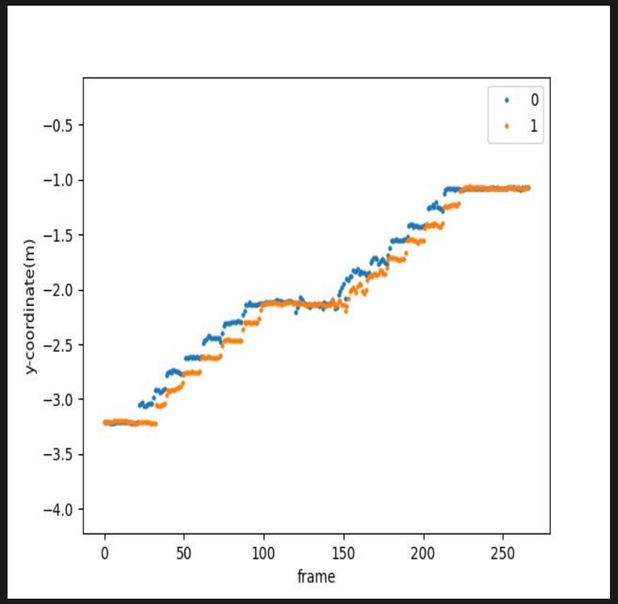
## (1) AR CORE : Floor detection

### Furthermore

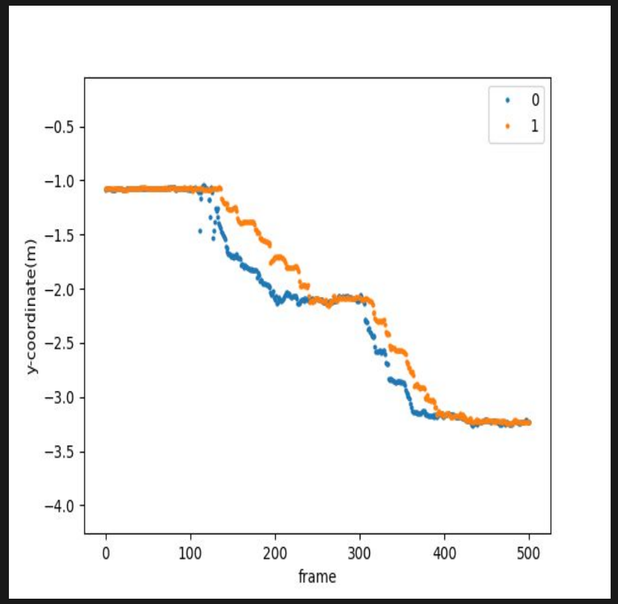
### Wall



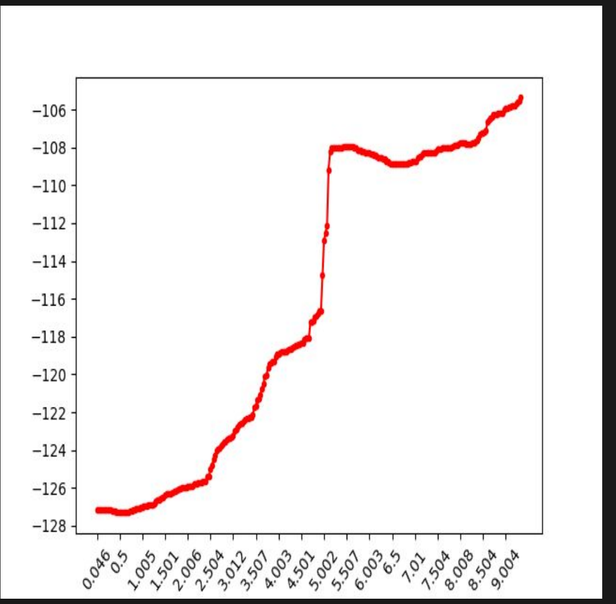

### Upstair




### Downstair




### Raised spot

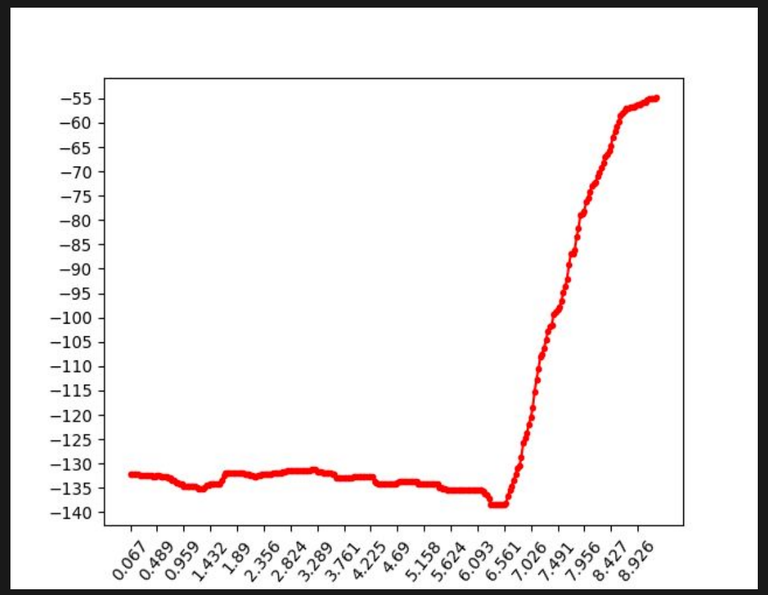


# Technical details

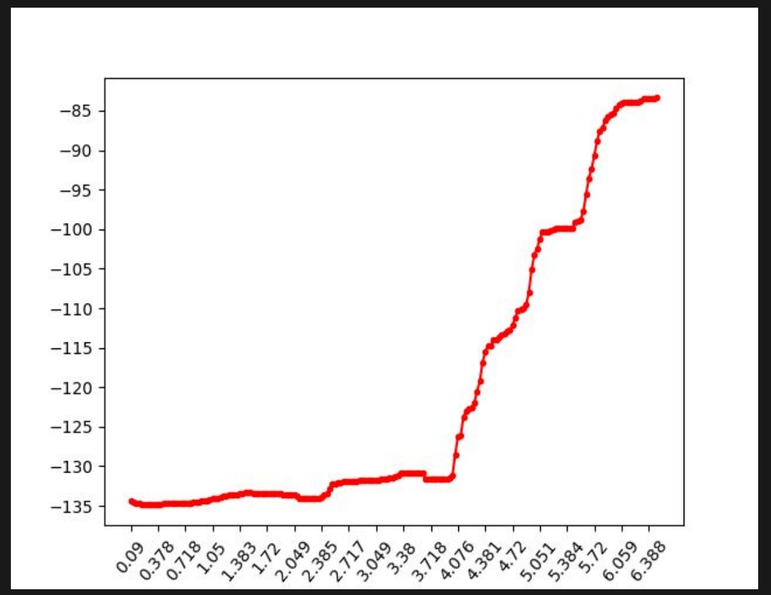
## (1) AR CORE : Floor detection

### Challenges 1.

**Wall**




**Upstair**

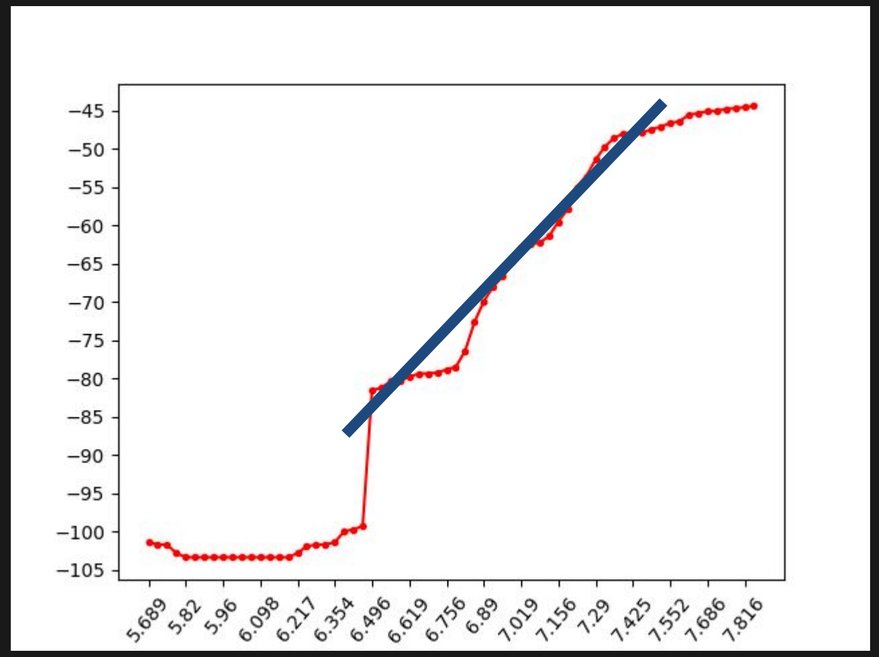


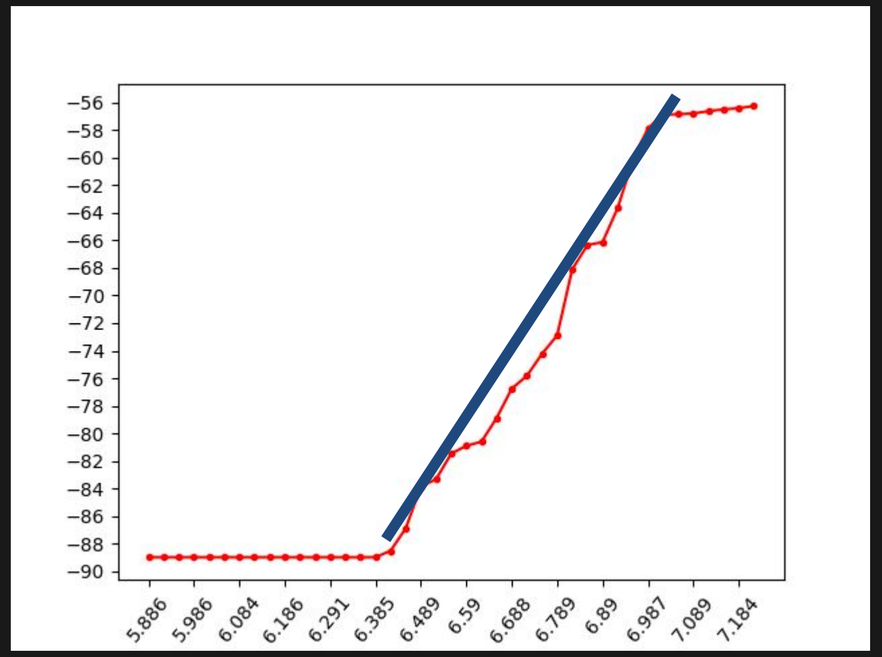

→ **There's not much difference in a short term.**

# Technical details

## (1) AR CORE : Floor detection

### Challenges 2.



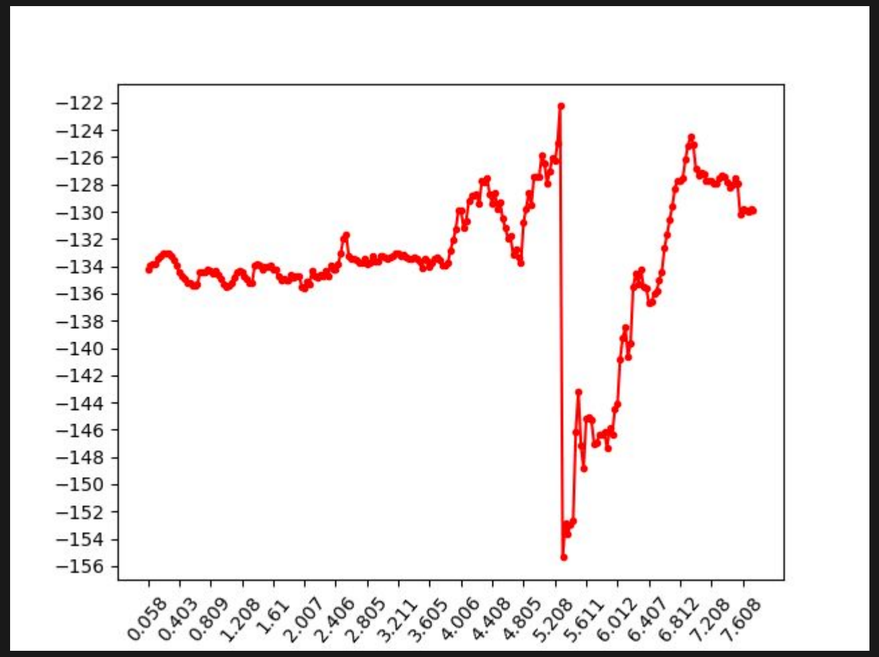



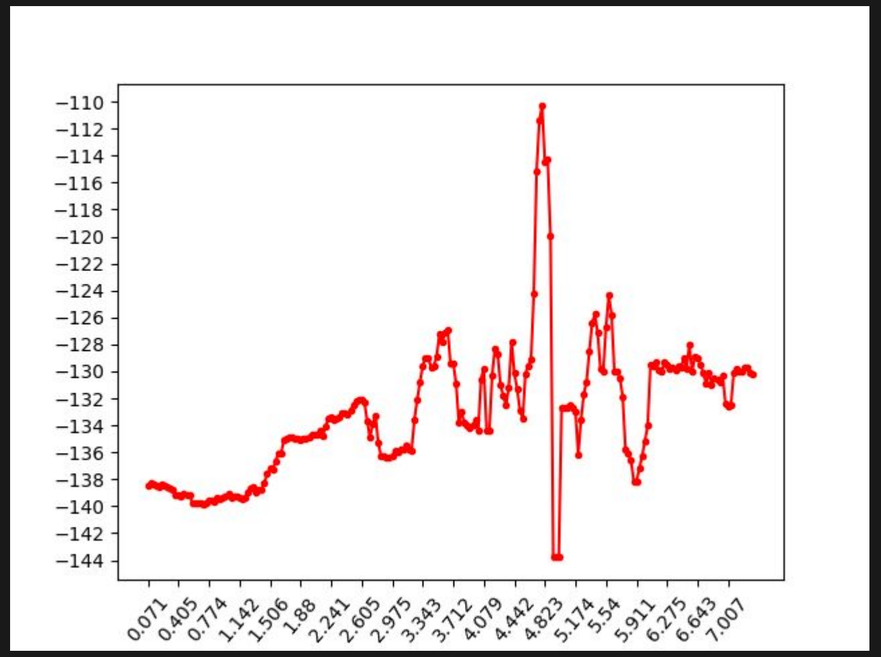

→ **Difference in data according to the walking speed.**

# Technical details

## (1) AR CORE : Floor detection

### Challenges 3.




→ **Too much noise in data**

# Technical details

## (1) AR CORE : Floor detection

**Possible Solution**

- Use a good noise cancellation algorithm
- State classification with deep learning
- Improve performance of AR Core

# Technical details

## (2) OBJECT TRACKING : object detection

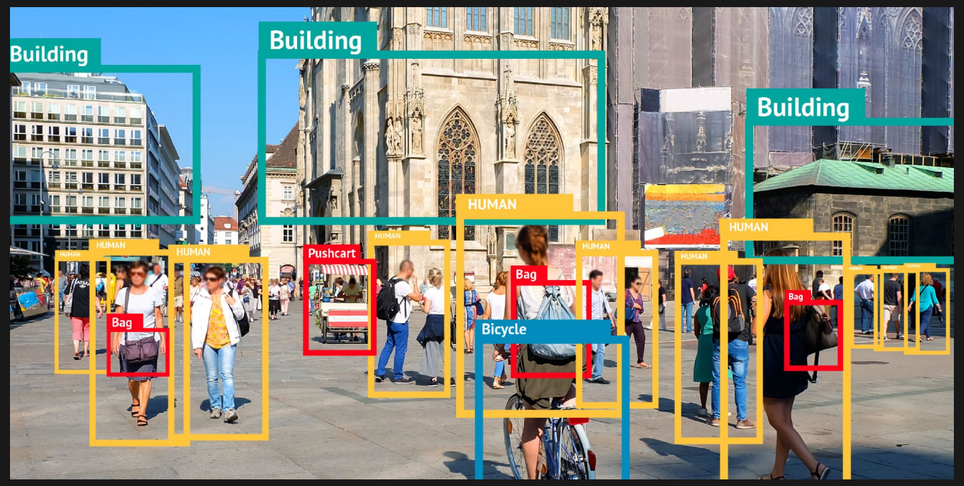
### Approach


**Warning in advance**
**if there’s an object at risk of hitting**

# Technical details

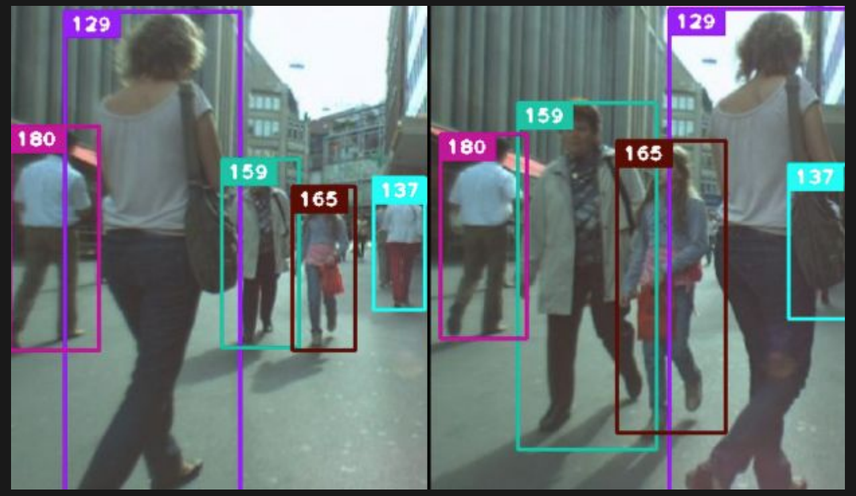
## (2) OBJECT TRACKING : object detection

### Structure



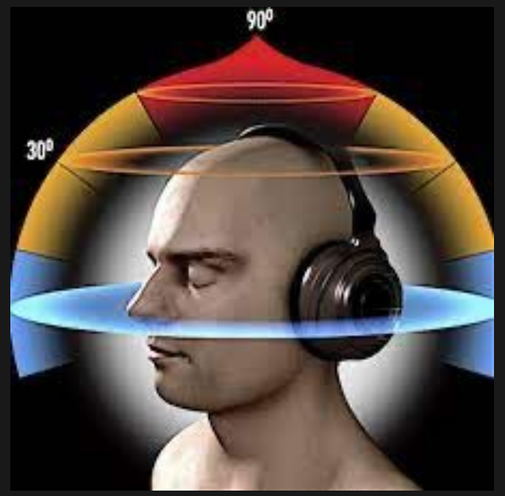

Object detection

+




Object tracking with SORT Algorithm

+

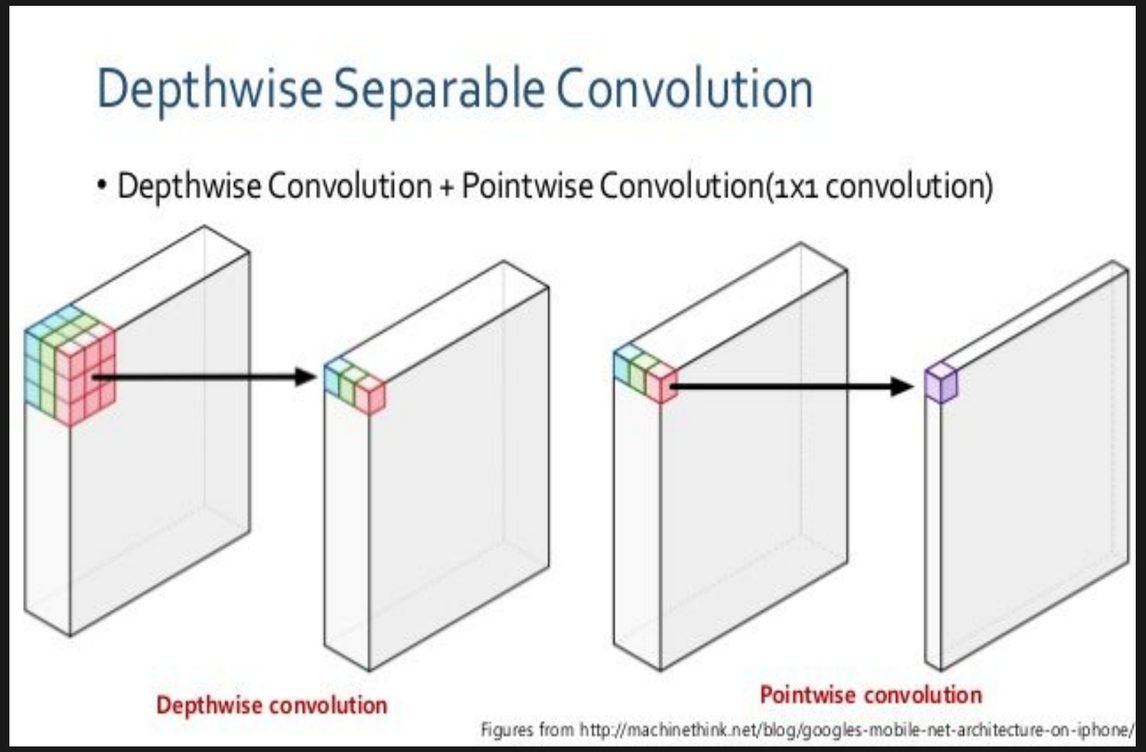



3D sound feedback

# Technical details

## (2) OBJECT TRACKING : object detection

### Object detection




**Mobilenet v1**

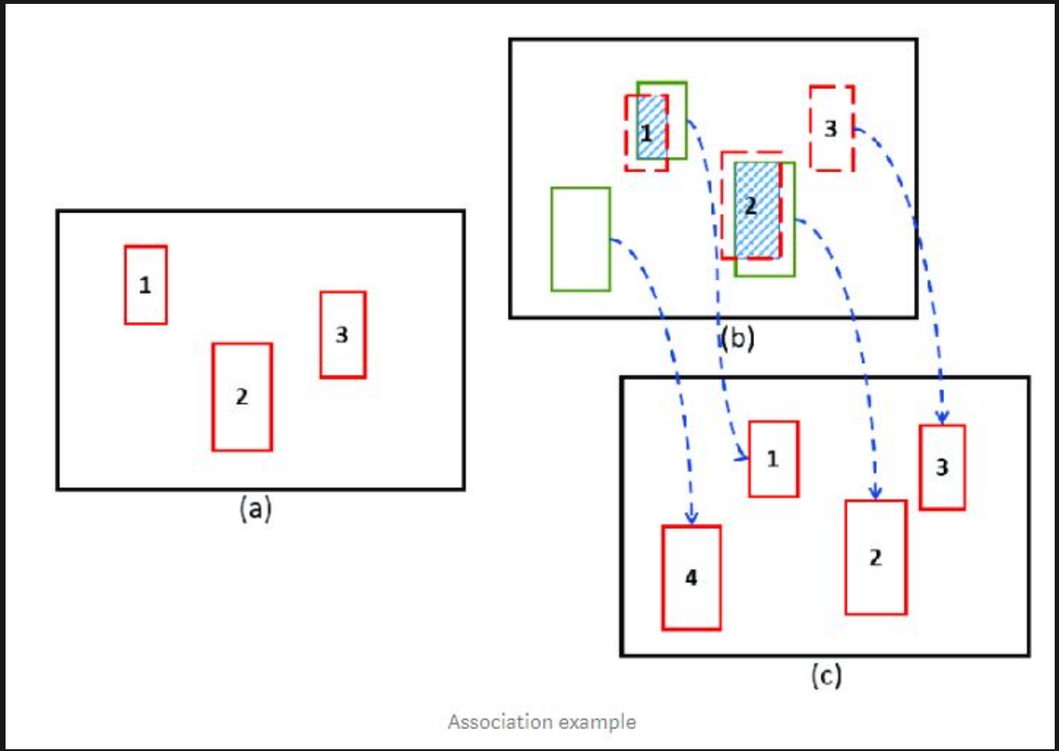
coco
Common Objects in Context

**Coco dataset**

# Technical details

## (2) OBJECT TRACKING : object detection

### Object tracking : SORT(Simple Online Real-time Tracking)



Association example

**Kalman Filter**

**+**

**Hungarian Algorithm**

# Technical details

## (2) OBJECT TRACKING : object detection

3D sound feedback

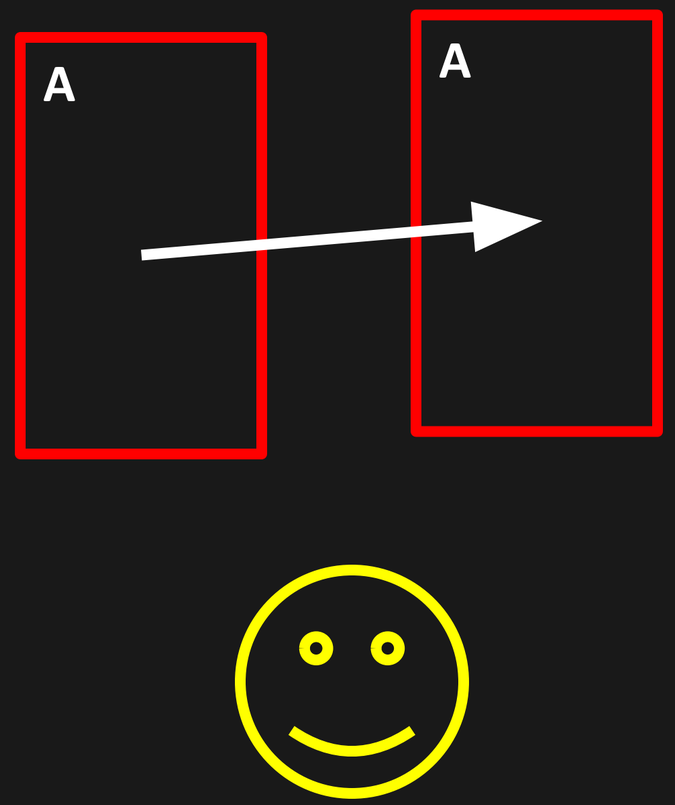
Google VR

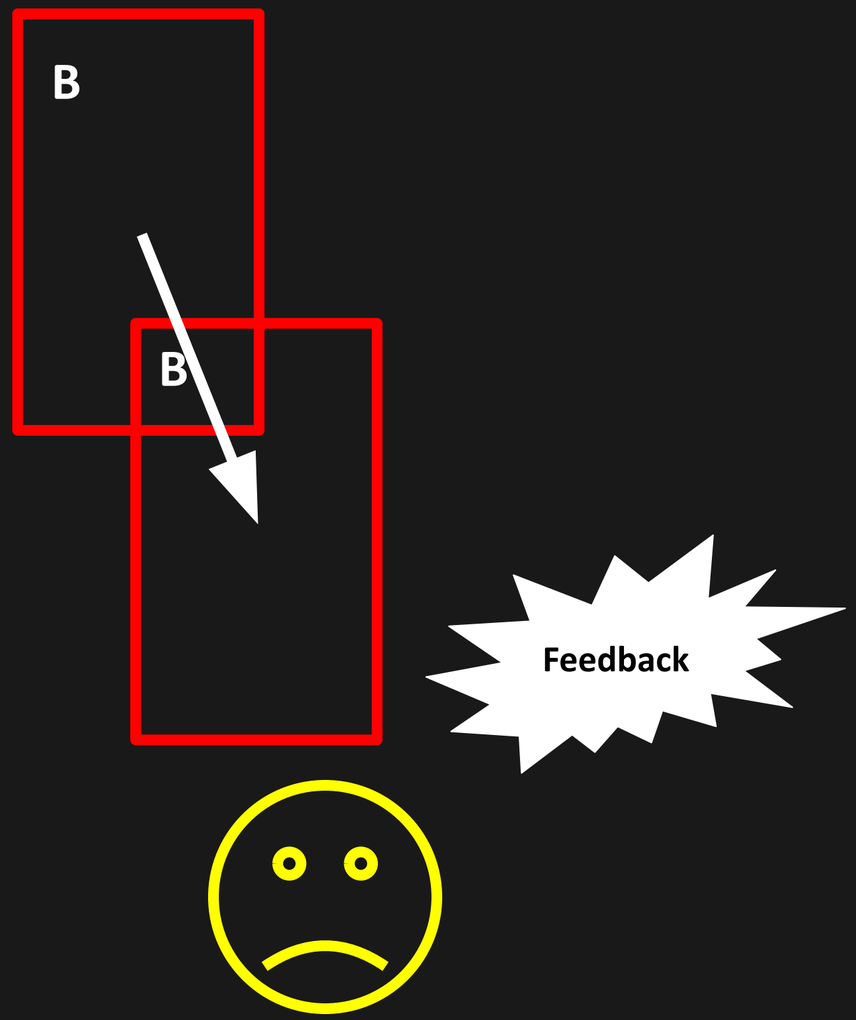
GVR Audio Engine

# Technical details

## (2) OBJECT TRACKING : object detection

### Detailed Implementation



# Technical details

## (2) OBJECT TRACKING : object detection

### Limitation

Too much noise in data → Velocity vector is not accurate

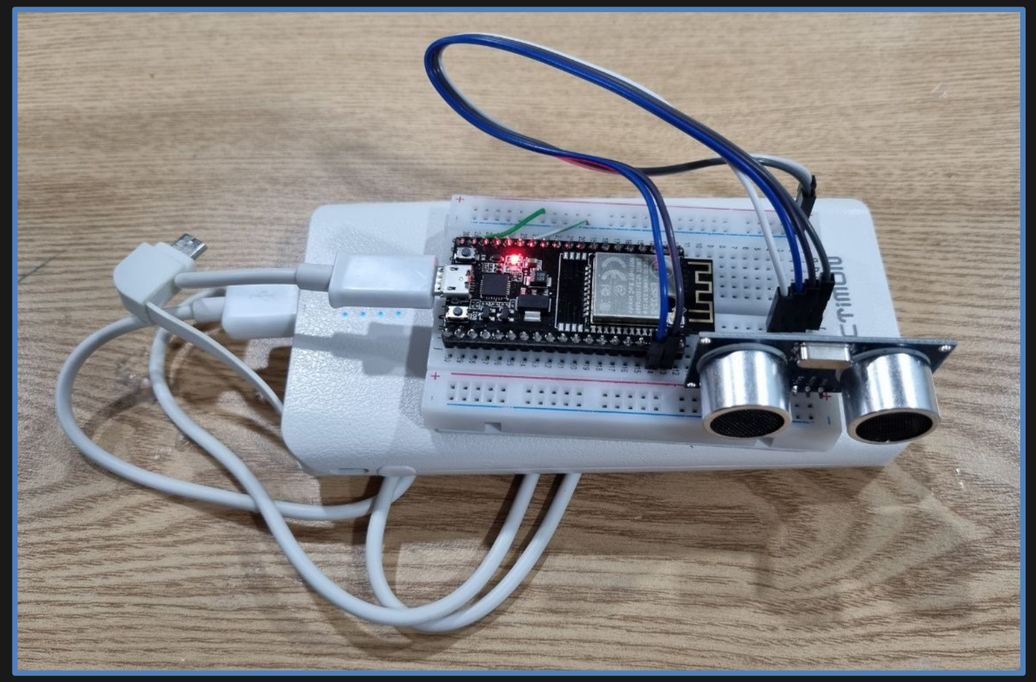
### Solution

Take a long frame to calculate the velocity vector

Use a function in GVR audio engine

# Technical details

## (3) ARDUINO : 그 외 위험요소 감지



**Arduino gives vibrate feedback**
**when object approaches from blind spot!**

# Technical details

## Evaluation and Success Criteria

### 사용 기기별 FPS

| 측정 기기 | 갤럭시 S21+ | 갤럭시 S20 | 갤럭시 노트10 | 갤럭시 S10 | 갤럭시 S9 |
|---|---|---|---|---|---|
| FPS | 28 | 25 | 20 | 17 | 5 |

# Technical details

## Evaluation and Success Criteria

Accuracy = (TP+TN) / (TP+TN+FP+FN)
Precision = TP / (TP+FP)
Recall (검출률) = TP / (TP+FN)

**# of FN = 0, 위험이 있는데 감지하지 못하는 상황은 없었다!**

Recall = 100%
Accuracy = Precision

# Technical details

## 올라가는 계단

| 평가항목 | 감지 성공 | 위험이 없는데 감지 | **Accuracy(%)** |
|---|---|---|---|
| Stationary object | - | 4 | 0 |
| Moving object | 1 | 0 | 100 |
| Obstacle | 2 | 0 | 100 |
| Hole | - | 0 | - |
| 총계 | 3 | 4 | 43 |

# Technical details

## 내려가는 계단

| 평가항목 | 감지 성공 | 위험이 없는데 감지 | Accuracy(%) |
|---|---|---|---|
| Stationary object | - | 4 | 0 |
| Moving object | 5 | 0 | 100 |
| Obstacle | - | 2 | 0 |
| Hole | 2 | 0 | 100 |
| 총계 | 7 | 6 | 54 |

# Technical details

## 강의실

| 평가항목 | 감지 성공 | 위험이 없는데 감지 | **Accuracy(%)** |
|---|---|---|---|
| Stationary object | 3 | 0 | 100 |
| Moving object | - | 0 | 0 |
| Obstacle | - | 0 | 0 |
| Hole | - | 0 | 0 |
| 총계 | 3 | 0 | 100 |

# Technical details

## 301동 1층, 실제 상황

| 평가항목 | 감지 성공 | 위험이 없는데 감지 | Accuracy(%) |
|---|---|---|---|
| Stationary object | 3 | 3 | 50 |
| Moving object | 5 | 0 | 100 |
| Obstacle | 1 | 4 | 20 |
| Hole | - | 2 | 0 |
| 총계 | 9 | 9 | 50 |

# Conclusion

1. 어려운 주제였음에도 적절한 scope 조절과 많은 heuristic을 통해 모든 위험을 감지하는데 성공했다.

2. 정확도는 50%를 달성하였다.

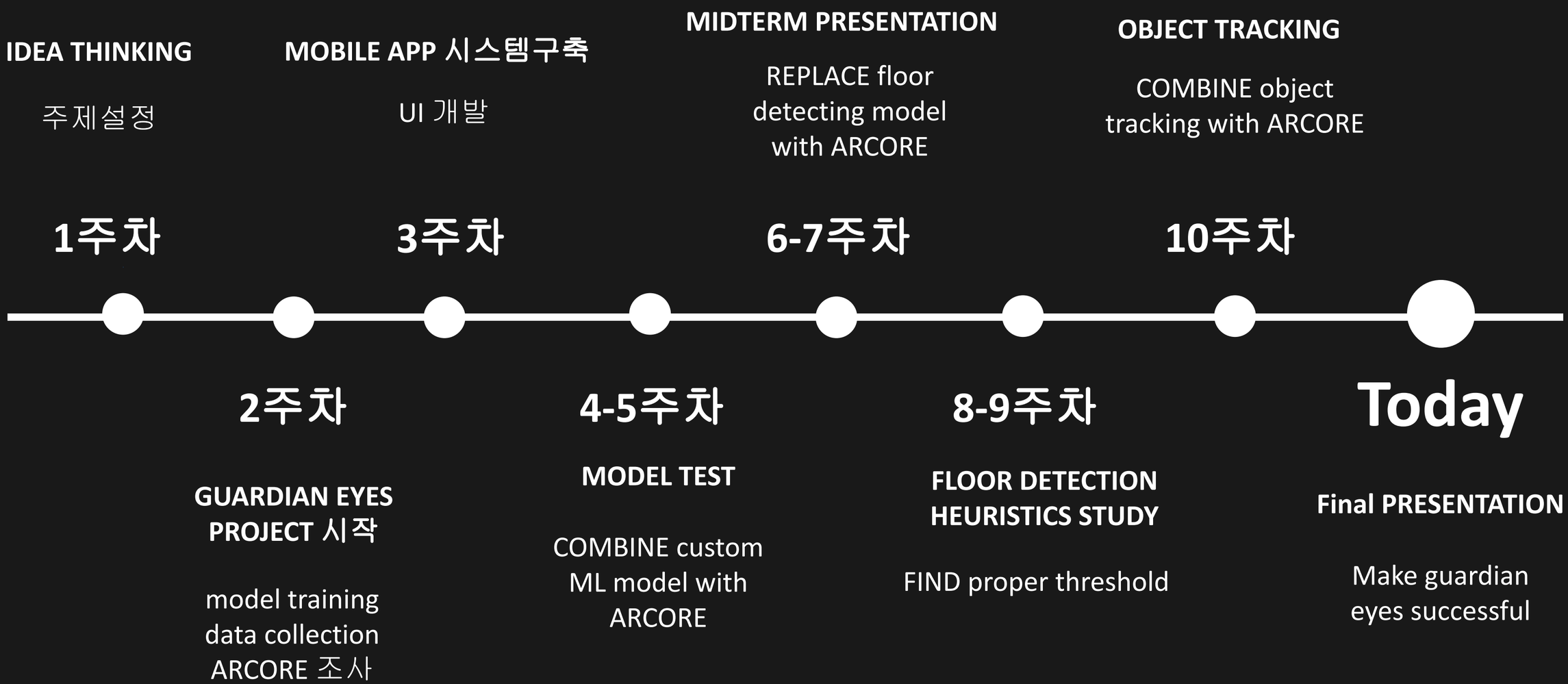
3. 하드웨어의 한계 내에서 가능한 최선의 성능을 끌어내는데 성공하였다.

# Project management

Part.4

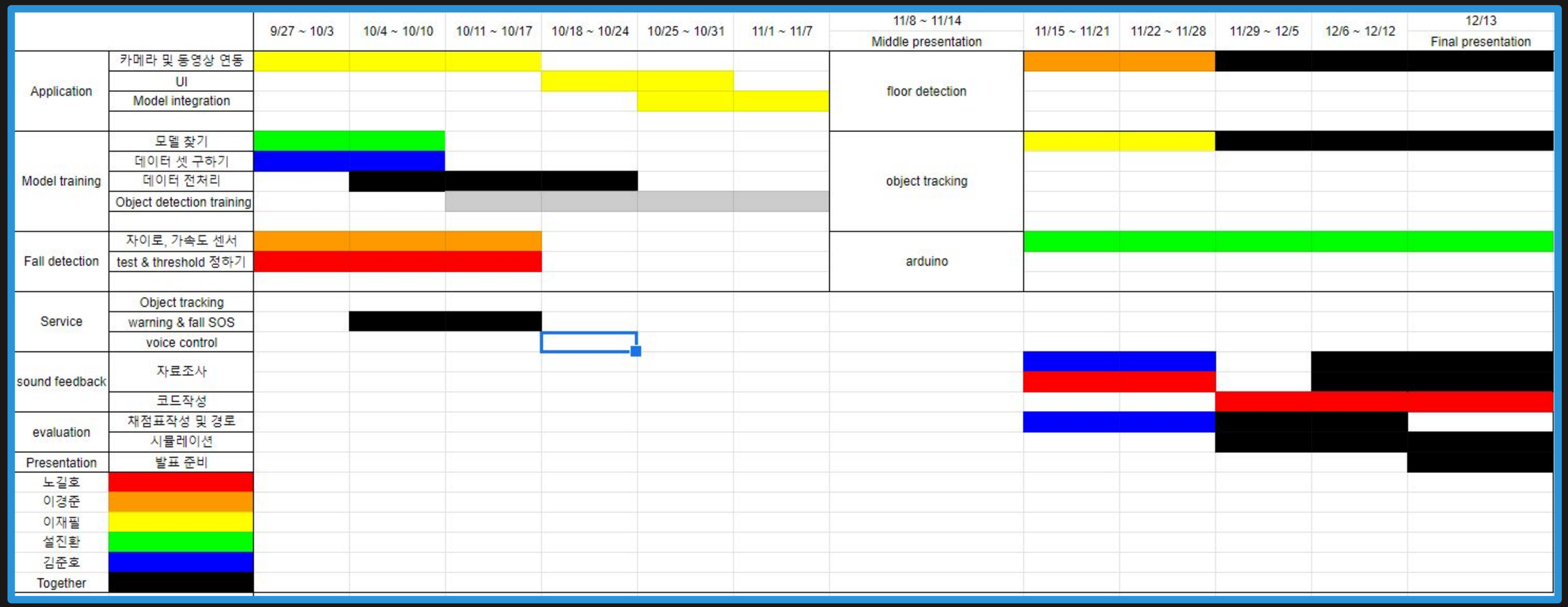
# Project management

## (1) TIMELINE



# Project management

## (2) ROLES AND CONTRIBUTIONS



# Lesson learnt

LIMITATION IN MOBILE APP

THE IMPORTANCE OF DEEP LEARNING TO OBJECT DETECTION & CLASSIFICATION

VARIOUS RESEARCH TOPICS REMAIN IN MOBILE RESEARCH

# Thank you for listening