

캡스톤 디자인(1) 최종 보고서

MIND EYE (7조)

캡스톤디자인(1) 02분반

이준우 교수님

학과 : 소프트웨어학과

7조

20201044 진용욱

20200278 유윤재

20213640 정지현

제출일 : 2024년 12월 24일 화요일

구성

GITHUB 및 매뉴얼	3
1. GITHUB 주소	3
2. 매뉴얼	3
프로젝트 소개	4
1. 도보 현황 분석	4
2. 타겟	5
3. 프로젝트 개요	5
사용 시나리오	6
구현 방법	9
1. 음성 인식과 안내 (TTS - STT)	9
2. 장애물 인식	9
3. 점자 블록 인식	11
4. 길 안내 (네비게이션)	11
5. Architecture	15
6. DataSet	17
7. 개발 목표	17
8. 정확도 향상 평가	18
9. 인식 속도 평가	19
서비스 필요성	20
1. 기존 서비스와의 비교	20
릴루미노 글래스	21
ORCAM	22
2. 심층 인터뷰	22
확장 가능성	23
1. 실제 사용 시 불편한 점	23
2. 도로 Segmentation	24

GITHUB 및 매뉴얼

1. GITHUB 주소

<https://github.com/CAU-7>

2. 매뉴얼

- application
 - APP_UI repository에서 파일 다운로드
 - Mac cmd

```
npm start  
npm run ios
```

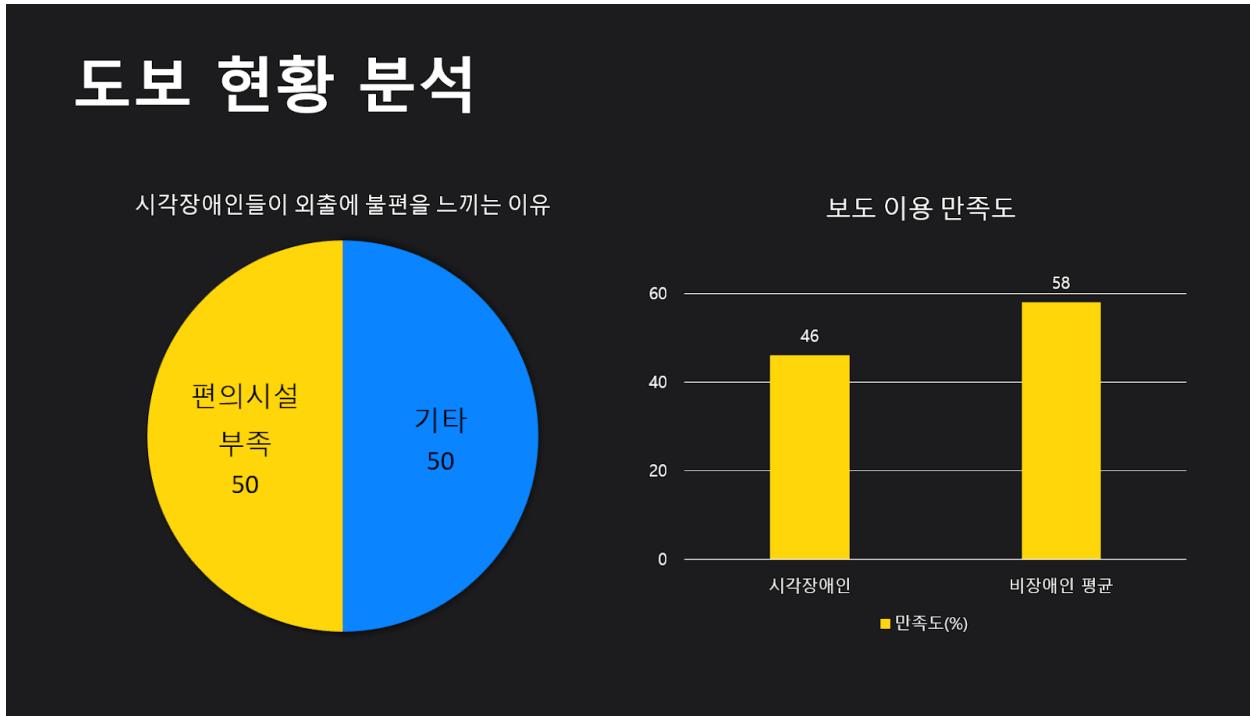
- object detection visual test
 - Object-Detection-Test repository에서 사용 가능
 - mac cmd

```
git clone https://github.com/CAU-7/Object-Detection-Test.git  
(object-detection-test)  
cd object-detection-test  
python3.9 -m venv venv  
source venv/bin/activate  
pip install -r requirements.txt  
python3.9 main.py
```

다른 영상을 사용하려면 main.py 43 line 경로 변경

프로젝트 소개

1. 도보 현황 분석



한국 소비자원의 2020년 시각 장애인 보행 안전실태조사에 따르면, 대한민국에는 약 25만명의 시각 장애인이 있음.

(외출 빈도) 장애인의 외출 빈도는 거의 매일 외출하는 경우가 70.1%로 가장많고, 주 1~3회 19.5%, 월 1~3회 5.9%의 순임.

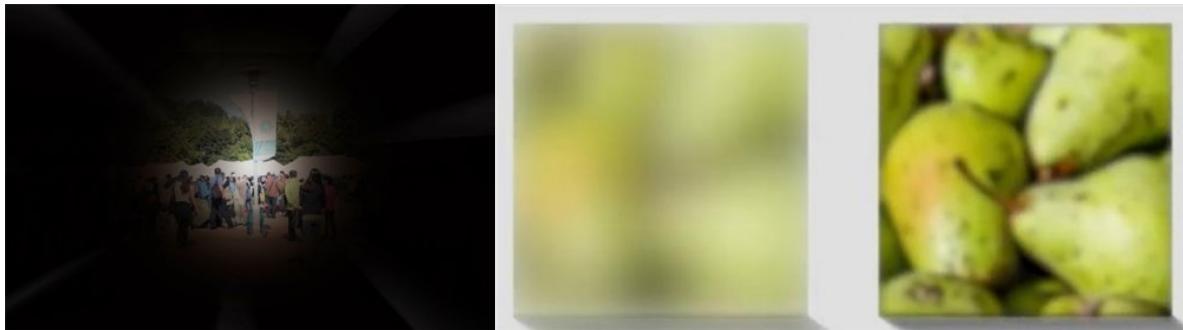
(외출 불편 사유) 시각장애인의 외출 시 불편을 느끼는 이유에 대해 조사한 결과, 조사대상 약 50%에 해당하는 시각장애인들이 편의시설 부족으로 인해 불편함을 겪고 있는 것으로 나타남.

[2017년 장애인 실태조사(2017, 보건복지부), 2016년 교통약자 이동편의 실태조사(2016, 한국교통연구원)]

2.

타겟

시각 장애인의 86%는 전맹이 아니라 사진과 같이 잔존시력이 있다는 걸 알게 되었음. 개발한 어플리케이션이 시각 장애인의 길찾기와 장애물 회피를 보조해주는 어플리케이션인 만큼, 전맹이신 분들 보다는 시야각이 매우 좁거나, 시력이 잘 보이지 않는 시각 장애인 분들께 더욱 유용할 것이라고 판단함.



3.

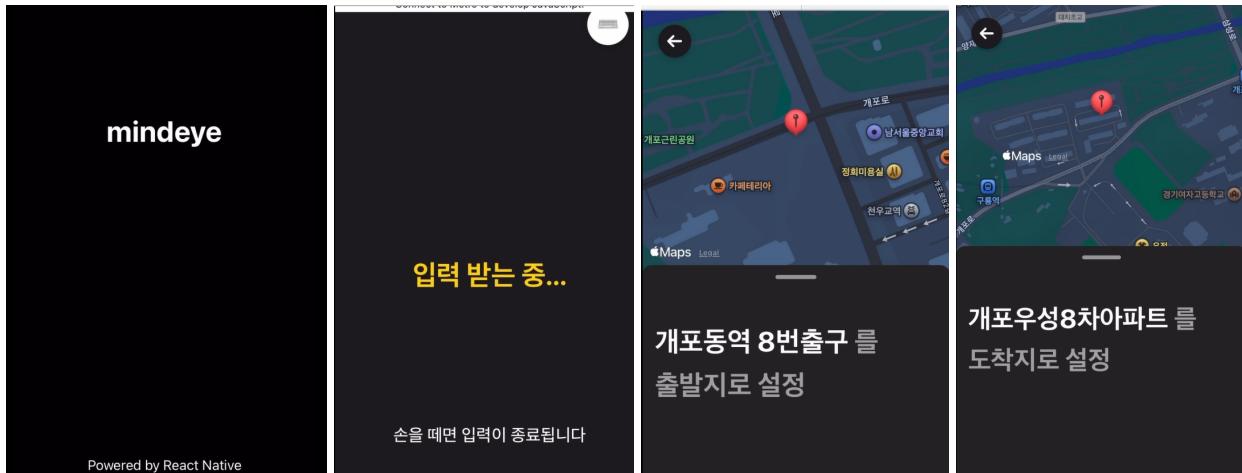
프로젝트 개요

시각장애인을 원하는 출발지에서 원하는 목적지까지 안전하게 이동할 수 있도록 하는 것을 목표로 함. 음성과 키보드를 이용해 출발지와 목적지를 선택하고, GPS와 카메라를 이용해서 장애물을 피해 목적지까지 이동할 수 있도록 함. 카메라를 사용해 장애물을 인식하기 때문에 카메라가 사용자의 정면을 바라보고 있어야 하므로 사용자가 핸드폰을 가슴 부근에 있는 포켓에 넣거나, 핸드폰 목걸이를 이용해 정면으로 고정한다고 가정.

사용 시나리오

출발지 : 개포동역 8번 출구

도착지 : 개포우성 8차 아파트



1. 핸드폰 자체 음성 기능이나 직접 클릭을 통해 앱 실행

2. 출발지를 입력 받기 위한 화면

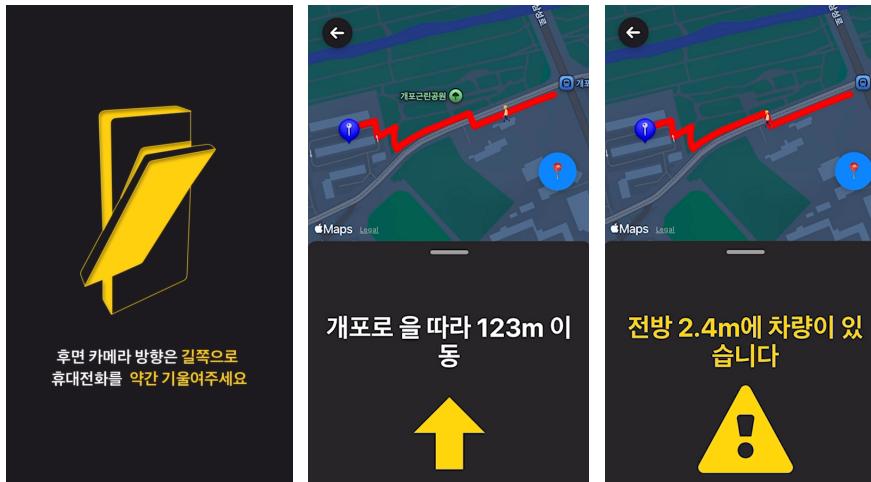
음성으로 “출발지를 입력해주세요. 화면을 길게 누르면 음성으로 입력할 수 있습니다.”라는 안내 메시지가 나오게 됨. 손으로 화면을 누르게 되면, 음성으로 출발지를 입력 받을 수 있음.

3. 출발지 입력 완료

“개포동역 8번 출구”라고 음성으로 출발지를 입력하게 되면, 다음과 같은 화면이 보이게 됨.

4. 도착지 입력 완료

음성으로 “도착지를 입력해주세요. 화면을 길게 누르면 음성으로 입력할 수 있습니다.”라는 안내 메시지가 나오게 되고, 동일한 방식으로 도착지를 입력 할 수 있음.



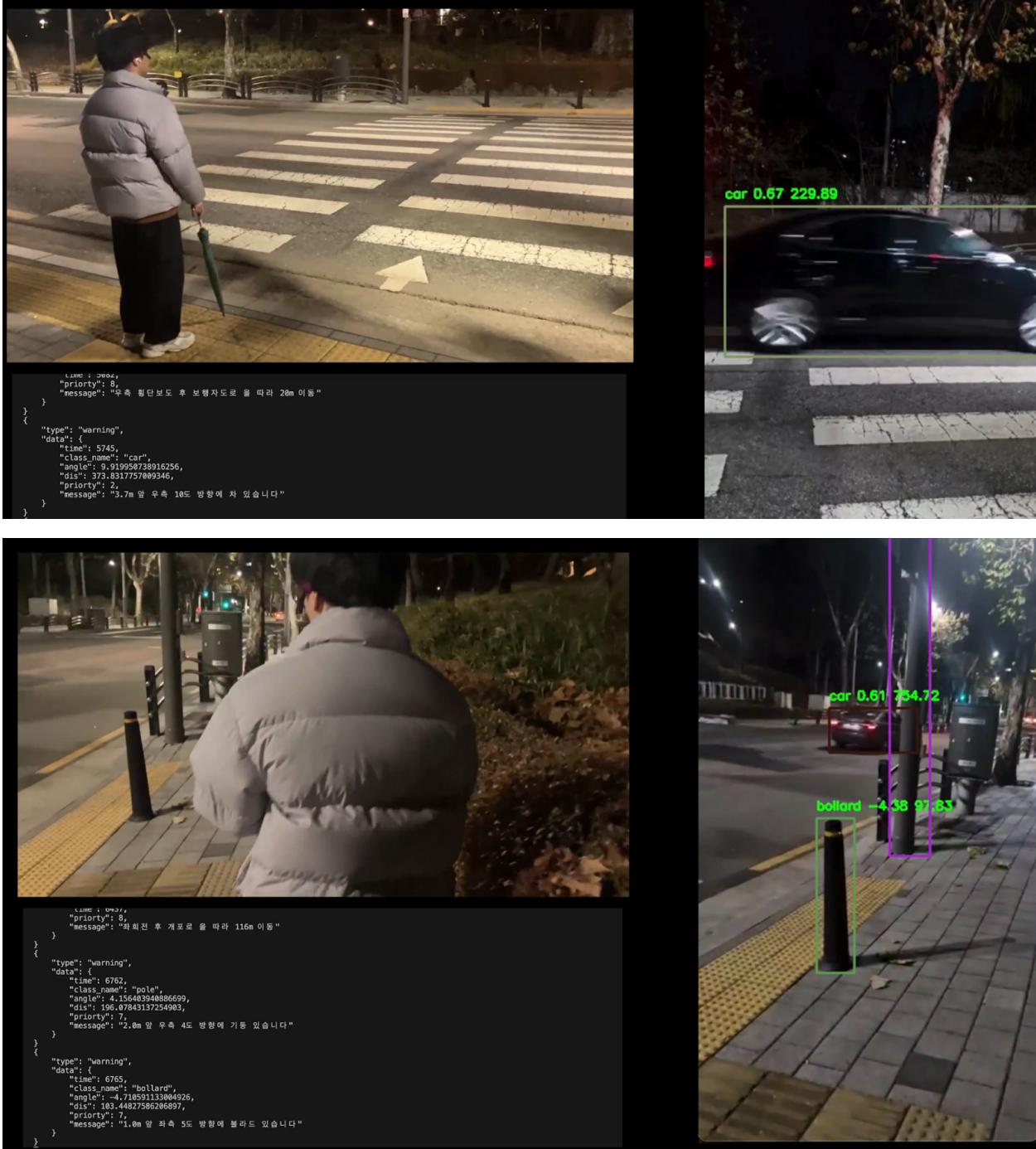
5. 카메라 이용을 위한 안내

핸드폰 카메라를 이용하여 장애물을 인식하기 위해서는 카메라의 방향이 전방을 향하고 있어야 하기 때문에 음성으로 “후면 카메라 방향은 길쪽으로 휴대전화를 약간 기울여주세요.”라는 안내가 나오게 됨.

6. 안내 시작

핸드폰 gps 기능을 활용해서 사용자에게 음성으로 올바른 길 안내를 해줌.

차량이나 장애물 발견 시 다음과 같은 화면이 표시됨과 동시에, 장애물에 대한 경고 음성을 사용자에게 들려주게 됨.



장애물 인식 부분은 위와 같음.

구현 방법

1. 음성 인식과 안내 (TTS - STT)

React-native 의 module 을 활용해서 text to speech 와 speech recognition 기능을 구현하였다.

TTS 의 경우, react-native-tts module 을 import 하여 앱 조작 방식과 경로 설정 관련 안내를 하도록 구현하였다.

STT 의 경우, react-native-voice module 을 import, 사용자가 음성으로 입력한 출/도착지를 텍스트로 변환하여 결과를 출력하는데 사용하였다.

2. 장애물 인식

- Object Distance, Angle



$$\text{angle} = \left(\frac{\text{object_center_x} - \frac{\text{image_width}}{2}}{\text{image_width}} \right) \times \text{horizontal_fov}$$

$$D = \frac{F \times T}{W}$$

도보를 이용할 때 장애물이 될 수 있는 것들을 인식하여 음성으로 안내한다.

카메라의 초점 거리와 클래스의 예상 크기를 이용하여 object와의 거리를 구한다.

카메라의 FOV(화각)과 object의 화면상의 위치를 이용해 각도를 구한다,

- Object tracking

```
If (Pixel_Diff * Distance <= threshold * (instance_speed + 1)
```

```
-> same object
```

동일한 object를 여러번 안내하지 않기 위해 위와 같은 공식을 활용하여 이전 프레임과 현재 프레임의 인식된 object의 거리가 예상 속도보다 작다면 가장 가까운 동일한 class의 object를 이전 프레임과 동일한 object로 인식하여 한번만 안내한다. 화면 상에는 동일한 객체로 인식되면 bounding box의 색이 동일하게 나타남

- 장애물 안내

구한 거리와 각도를 바탕으로 부딪힐 위험이 있는 object들의 정보를 담아 front로 전송한다.

안내시에는 object를 탈 것 (차량, 오토바이, 자전거 등), 움직이는 물체 (사람, 강아지, 리어카 등), 고정체 (나무, 기둥, 블라드 등)으로 나눠 우선순위와 인식 거리, 각도를 다르게 하여 안내한다.

탈 것 -> 움직이는 물체 -> 고정체 순으로 안내하는 인식 거리와 각도가 줄어들고 우선순위가 감소한다.

구한 거리와 각도를 바탕으로 부딪힐 위험이 있는 object들의 정보를 담아 front로 전송한다.

안내시에는 object를 탈 것 (차량, 오토바이, 자전거 등), 움직이는 물체 (사람, 강아지, 리어카 등), 고정체 (나무, 기둥, 블라드 등)으로 나눠 우선순위와 인식 거리, 각도를 다르게 하여 안내한다.

탈 것 -> 움직이는 물체 -> 고정체 순으로 안내하는 인식 거리와 각도가 줄어들고 우선순위가 감소한다.

3.

점자 블록 인식



Segmentation을 통해 점자 블록의 영역을 인식하고, 앞의 방향에 점자 블록이 있다면 해당 정보를 front로 전송한다.

4.

길 안내 (네비게이션)

Tmap에서 지원하는 API를 사용하여 도보 안내 기능을 구현하였다.

출/도착지는 사용자의 입력(음성 혹은 키보드)에 가장 적합한 결과를 tmap api의 POI 검색 기능을 적용해 설정하였다.

경로 설정은 tmap api 의 도보 경로 안내 기능을 활용, 응답 데이터로 주어진 turnpoint 와 linestring geometry 정보를 활용하여 각 turn point 에 가까워지면 거리와 방향을 안내하도록 구현하였다.



```
{  
  "type": "Feature",  
  "geometry": {  
    "type": "Point",  
    "coordinates": [  
      127.06582082272675,  
      37.488929158125096  
    ]  
  },  
  "properties": {  
    "totalDistance": 390,  
    "totalTime": 323,  
    "index": 0,  
    "pointIndex": 0,  
    "name": "",  
    "description": "개포로 을 따라 123m 이동",  
    "direction": "",  
    "nearPoiName": "",  
    "nearPoiX": "0.0",  
    "nearPoiY": "0.0",  
    "intersectionName": "",  
    "facilityType": "11",  
    "facilityName": "",  
    "turnType": 200,  
    "pointType": "SP"  
  }  
},
```

TMap API에서 보행자 경로 요청 시 응답 데이터 (일부)



길안내 시에 실행되는 화면은 다음과 같음. 화면 상단에는 현재 지도의 모습을, 하단에는 안내 정보를 크게 띄워주었음. 사용자 위치를 gps로 추적하면서, 올바른 방향으로 사용자를 안내함.

안내

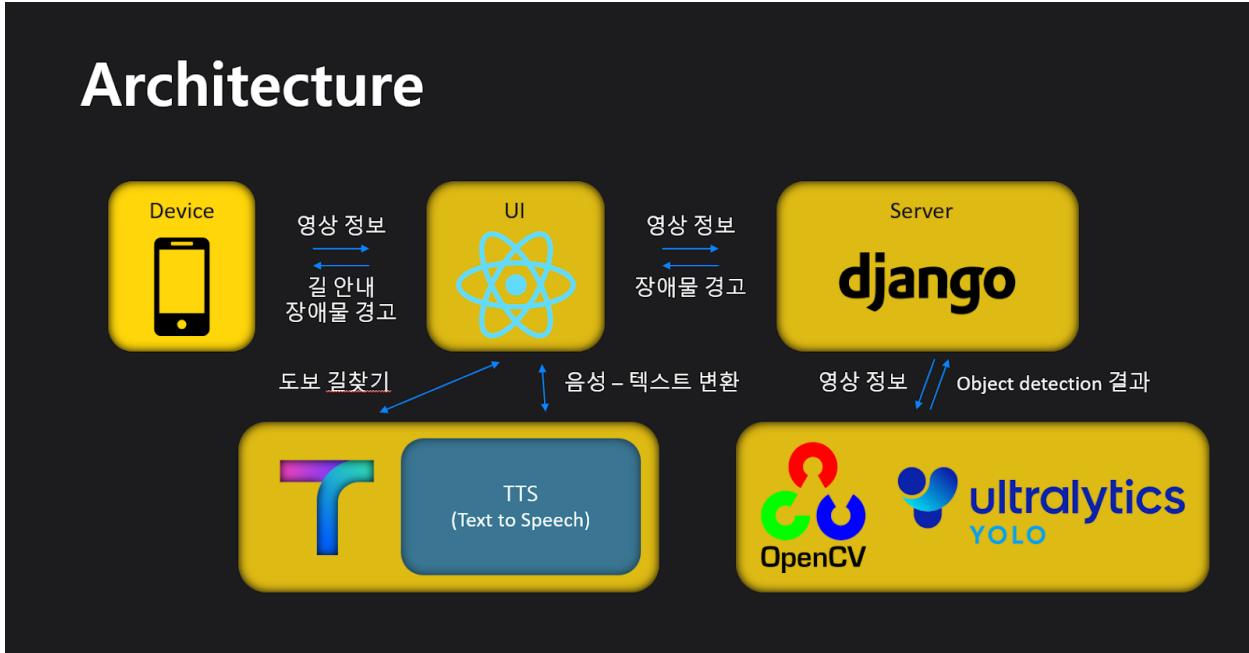
길 안내, 장애물 경고, 점자 블록 안내에서 얻은 정보들을 우선순위 Queue 형태로 저장하여 경고 -> 점자 블록 -> 길 안내의 순서대로 안내한다.



5.

Architecture

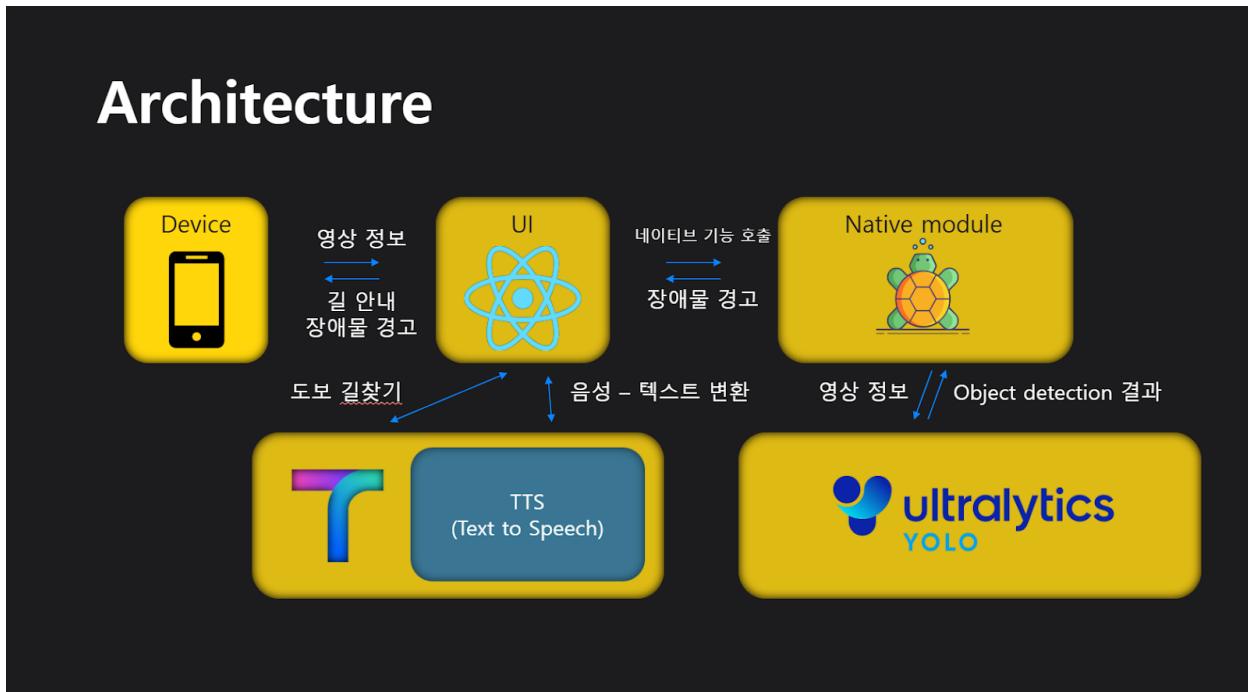
초기



초기 아키텍처 설계 시에는 react native로 앱을 만들고, 해당 앱에서 카메라 영상을 실시간 캡처를 통해서 django서버로 보내고, 서버에서 해당 image를 분석하여, 그 결과를 다시 앱으로 보내는 방식으로 설계함. 도보 길찾기의 경우, 도보 api를 제공해주는 T map의 api를 이용해 구현하였음.

- 시행착오 : 실시간 영상 공유, 혹은 실시간 캡처를 통한 이미지 전송 시, 최소 0.5 ~ 1 second의 전송 delay 발생. 실시간성이 매우 중요한 프로젝트인 만큼, 치명적인 약점이라고 판단을 하여 아키텍처를 수정하게 되었음.

최종



초기 아키텍처와 다르게 모든 것을 앱 내부에서 처리할 수 있도록 설계함. react native에서 react-native-vision-camera와 같은 여러 native module들을 통해, 카메라 영상을 실시간 캡처하여 yolo모델에 전달함. yolo 모델의 경우, yolo 5 nano 모델을 기반으로 재학습시켜서 앱 내부에서도 가볍게 돌아 갈 수 있도록 하였음. 이미지 분석 결과를 반환하게 되면, text to speech 기능을 통해 음성으로 사용자에게 전달할 수 있도록 하였음. 길찾기 기능의 경우, 이전 아키텍처와 동일함.

6.

DataSet



A screenshot of a web page from AI Hub. At the top, there's a navigation bar with categories like "#도로 서비스 개발", "#보도블록", "#점자블록", "#이동체 장애물", "#고정체 장애물", "#인도 노면", and "#자전거 도로 노면". Below the navigation is a large thumbnail image showing two stylized human figures in blue and purple walking. The title "인도보행 영상" (Pedestrian Walking Video) is displayed in large, bold Korean text. Underneath the title are several small buttons: "분야" (Category), "교통물류" (Traffic Logistics), "유형" (Type), and "이미지" (Image). Below these buttons is some descriptive text: "구축년도: 2019", "갱신년월: 2021-03", "조회수: 22,631", "다운로드: 6,050", and "용량: 607.40 GB". At the bottom of the screenshot are three main buttons: a red "다운로드" (Download) button, a white "샘플 데이터" (Sample Data) button with a download icon, and a white "관심데이터 등록" (Register Interest Data) button with a "103" notification badge.

이미지 인식 모델의 학습을 위해서 AI hub의 인도보행 영상의 dataset를 이용하였음. 보행시 충돌 위험이 존재하는 29종의 이동체와 고정체 데이터에 대한 600GB 용량의 데이터를 제공하여, 인식의 정확도를 높이는 데 기여하였음.

7.

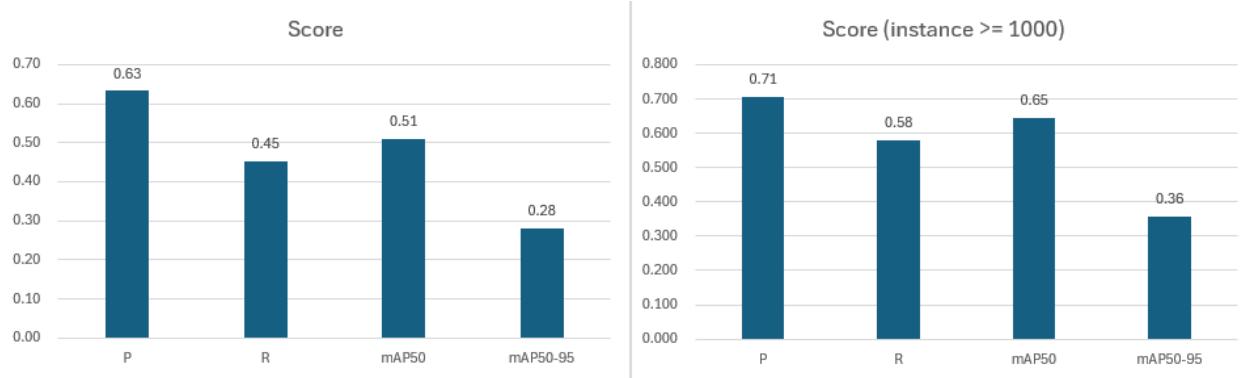
개발 목표

프로젝트 핵심 목표 두 가지는 영상의 장애물을 분석하는 속도와 정확도이다. 이미지 분석 모델을 도로 상황에 맞게 특화시켜서, 높은 퍼포먼스를 보이게 만들고자 하였음. 현재 사용하고 있는 yolov5 nano 신경망을 사용하여 점자블록과 도로 지형지물 등을 학습시켜 정확도를 향상시키는 것을 목표로 하였음.

8.

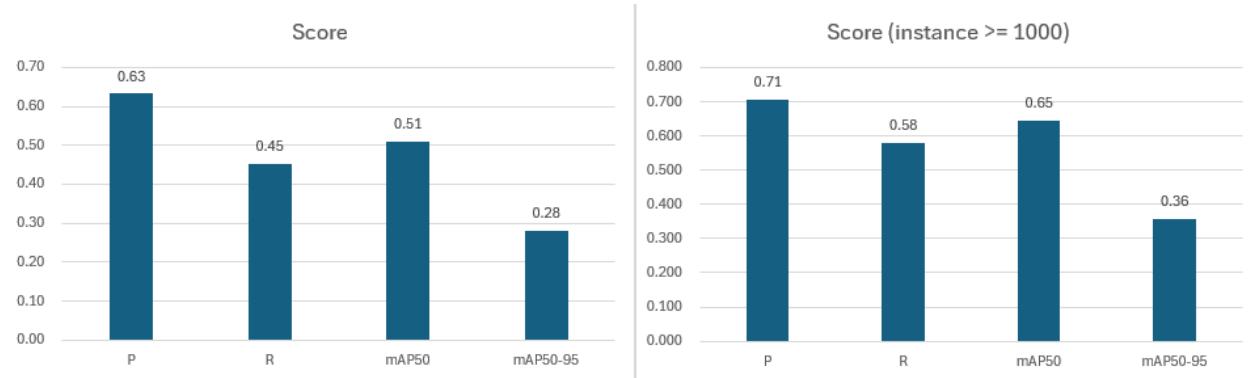
정확도 향상 평가

기존 yolov5n 모델의 mAP50과 비교하면 45.7% -> 51%로 정확도가 향상되었다.

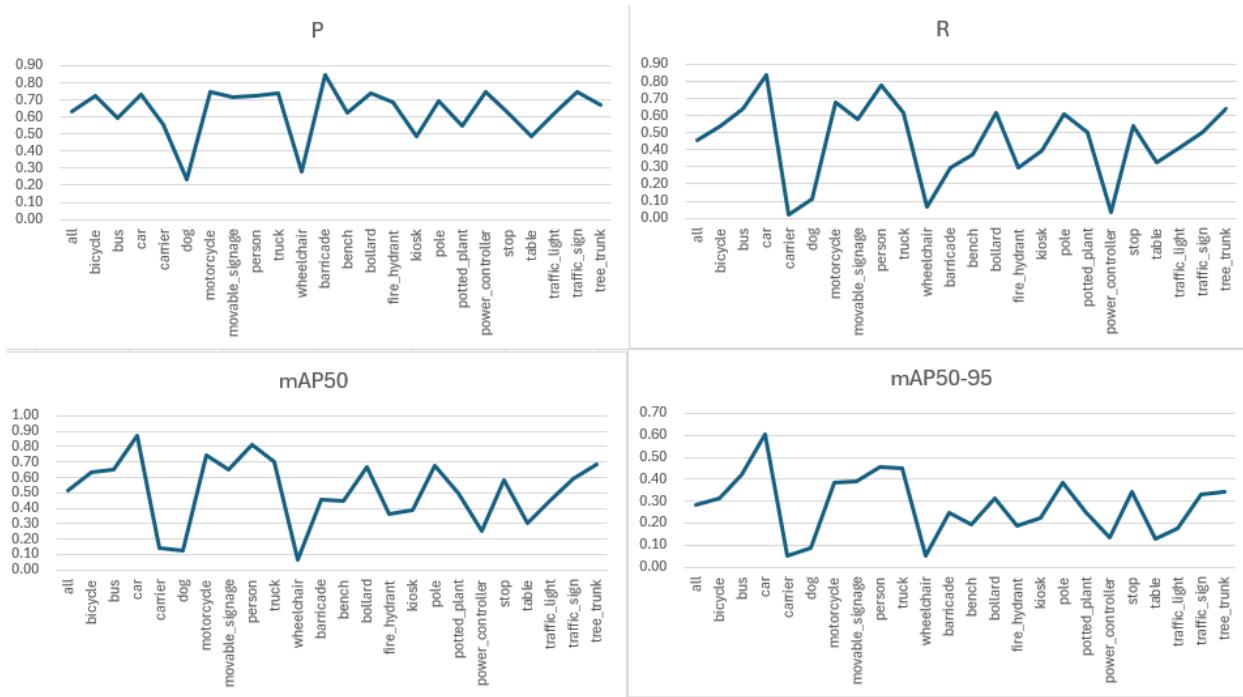


Model	size (pixels)	mAP ^{val} 50-95	mAP ^{val} 50	Speed CPU b1 (ms)	Speed V100 b1 (ms)	Speed V100 b32 (ms)	params (M)	FLOPs @640 (B)
YOLOv5n	640	28.0	45.7	45	6.3	0.6	1.9	4.5

instance가 1000개 이상인 class에 대해서는 66%로 더 높은 정확도를 보였다. (차, 사람, 자전거, 오토바이 등 도로에 자주 있는 객체 class)



각 Class별 Score



9.

인식 속도 평가

핸드폰에서 영상의 각 frame별 처리 시작 ~ 프론트로 메세지 전송까지의 시간 log

평균적으로 frame당 약 0.052정도의 시간이 소요된다.

1s / 0.052s ≈ 19.2, 초당 약 19번의 frame을 인식하고 처리 가능

동일한 환경에서 기존 yolov5 nano의 0.055s 정도의 시간이 소요된다, 0.055s → 0.052s 약간의 속도 개선이 이루어졌다.

목표로 했던 30fps에 못 미치긴 하였으나 0.052초의 delay로 새로운 장애물을 인식하여 안내해주는건 충분히 유의미하다고 판단 된다. (실제 18fps로 개발함.)

```

frame detect elapsed time: 0.050403 seconds
frame detect elapsed time: 0.052455 seconds
frame detect elapsed time: 0.050644 seconds
{'type': 'warning', 'data': {'time': 294, 'class_name': 'person', 'angle': 10.584975369458128, 'dis': 196.07843137254903, 'priority': 5, 'message': '2.0m 앞 우측 11도 방향에 person 있습니다'}}
frame detect elapsed time: 0.050044 seconds
frame detect elapsed time: 0.051355 seconds
frame detect elapsed time: 0.051356 seconds
frame detect elapsed time: 0.053254 seconds
frame detect elapsed time: 0.052231 seconds
frame detect elapsed time: 0.050607 seconds

```

10.

서비스 필요성

1. 기존 서비스와의 비교

서비스	길 찾기	음성 안내	실시간 인식	특징
G-EYE	○	○	X	자주 가는 경로 저장 가능
Walk With	○	○	X	산책로 추천 출발-도착 고정
Look Out	X	○	○	단순 음성 안내
7조의 앱	○	○	○	장애물 특화 인식 방향, 위치 안내

기존에도 시각장애인을 타겟으로 한 도보 안내 서비스가 존재했으나, 도보 길찾기가 아닌 산책 경로 안내를 주요 서비스로 하거나, 이번 프로젝트에서 주요한 구현 목표로 삼은 실시간 도로 상황 인식 기능은 없는 상황이 있다. 따라서 기존의 서비스를 보완한 새로운 어플리케이션을 개발하는 것에 가치가 있다고 판단하였다.

릴루미노 글래스

삼성전자 릴루미노 개발 타임라인

자료 : 삼성전자

Relumino

일자	내용
2016년 5월	삼성전자 C-Lab 과제 선정
9월	릴루미노 버전 1 개발
10월	릴루미노 버전 2 개발
2017년 2월	MWC 2017 전시
8월	릴루미노 버전 3 개발
	릴루미노 글래스 신규 과제 착수
2018년 2월	CES 2018 시제품 전시
2020년 7월	이재용 당시 삼성전자 부회장 참관
2021년 5월	식약처로부터 '안과학 진료용 SW' 품목허가(제허 21-426호)
2022년 11월	국립전파연구원 적합성평가(전파인증) 획득
2023년 3월	릴루미노 글래스 시범 보급



중간 발표 때 특별히 피드백 요청이 있었던 릴루미노 글래스에 대해 살펴본 결과, 릴루미노 글래스는 우리 서비스의 방향성과는 사용 목적 자체가 다른 제품임을 확인할 수 있었다.

먼저, 해당 제품은 물체를 인식하여 정보를 전달하는데, 실시간으로 물체를 인식한다는 점은 유사하다고 판단할 수 있으나 도보 이용 시 장애물을 판단하여 경고/안내를 하는 기능은 없는 것으로 확인하였다.

또한, 제품 사용 주의사항을 참고하면, 해당 제품을 착용한 채 걷거나 이동할 수 없는 것으로 나와있으며, 짧은 주기로 착용과 휴식을 반복해야한다는 점에서 도보 안내에는 적합하지 않은 제품인 것을 확인할 수 있었다.

ORCAM



올캠(ORCAM) 마이아이 제품을 상세히 분석해보았다. 자석을 통해서 안경에 부착하여 사용하는 방식으로 텍스트 읽기나 얼굴 인식, 바코드 인식, 지폐 인식 등 기능이 있어 주로 글자를 읽을 때 유용하게 사용할 수 있음.

무게는 대략 22.5g이고 한쪽 안경에 카메라가 달려있음. 충전 40분으로 2시간 이용가능함.
상대적으로 가격이 비싸다는 평가.

2. 심층 인터뷰

도보 이용 시 불편한 점에 대해 우리 제품의 타겟에 해당하는 시각장애인에게 물어본 결과,

1. 목적지까지 이동 경로 파악이 어려우며
2. 사람이나 장애물을 피하기 힘들다

는 의견을 얻을 수 있었다.

확장 가능성

1. 실제 사용 시 불편한 점

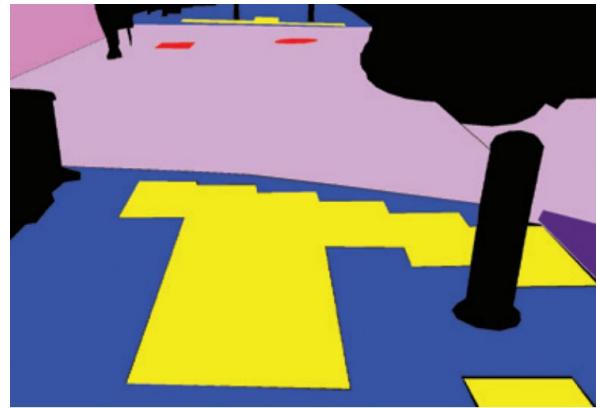
실제 시각장애인 체험을 통해 데모를 진행해본 결과, 장애물 경고가 도움이 많이 되었으나 보도블럭이 없는 인도에서 도로를 따라 걸어가기가 힘든 점을 인식했다. 또한 차도 뿐만 아니라 하수도, 맨홀, 가로수 영역, 흙길 같은 경우에 보행에 큰 어려움을 느꼈다.



2.

도로 Segmentation

사용했던 데이터셋과 더불어 도로 Segmentation을 통해 길을 따라 잘 걷고 있는지, 보행자 앞의 길이 어느 종류인지를 안내해준다면 더 많은 도움이 될 것이라 생각한다.



alley (사람과 차가 함께 다닐 수 있는 길)	crosswalk (횡단보도) damaged (파손) normal (일반) speed_dump (과속방지턱)
bike_lane (자전거도로)	(속성값 없음)
braille_guide_blocks (점자블록)	damaged (파손) normal (일반)
caution_zone (주의구역)	grating (그레이팅) manhole (맨홀) repair_zone (보수구역) stairs (계단) tree_zone (가로수영역)
roadway (차만 다닐 수 있는 길)	crosswalk (횡단보도) normal (일반)
sidewalk (인도)	asphalt (아스팔트) blocks (보도블럭) cement (시멘트) damaged (파손) other (기타) soil_stone (흙/돌) urethane (우레탄)