

# LG U+ Why Not SW Camp

## 최종 프로젝트 기획서

객체 탐지 기반 자율주행 차량 로봇의 포트홀 감지 및 자동 신고 서비스

# Dynamic Object Detection(D오디)

팀원 : 김승엽, 한은선, 김동혁, 김준희, 박장원, 김나현

---

## 목 차

### 1. 프로젝트 개요

- 1.1 프로젝트 배경
- 1.2 프로젝트 목적 및 필요성

### 2. 프로젝트 방향

- 2.1 자율주행 로봇 객체 인식 기반 행동 제어 서비스

### 3. 아키텍처

- 3.1 시스템 아키텍처
- 3.2 클라우드 아키텍처

### 4. 모델 선정 배경

- 4.1 YOLOv5 YOLOv11 비교

### 5. 모델 학습

- 5.1 YOLO 모델 학습 구현 과정

### 6. 라이브러리

- 6.1 SW 라이브러리
- 6.2 아두이노 라이브러리

### 7. 부록

- 7.1 아두이노 자율주행 제품
- 7.2 팀원 역할

## 1. 프로젝트 개요

### 1.1 프로젝트 배경

- 포트홀 사고 보상 통계 현황

# 고속도로 '포트홀', 최근 5년간 2만2752건…배상액 44억 3800만원

등록 2024.07.30 08:56:10 | 수정 2024.07.30 11:08:52



도로공사 통계 2019년 3717건→2023년 5801건

강우량 많은 7~8월 집중…중앙선·서해안선 발생

황희 민주당 의원 "사고발생 위험…예방책 필요"



[서울=뉴시스] 전남지역에서 도로파임(포트홀)을 정비하는 모습. 2024.07.30. (사진=전남도 제공) photo@newsis.com \*

재판매 및 DB 금지

최근 5년간 고속도로에서 발생한 도로파임(포트홀)이 총 2만2753건을 넘으며 이로 인한 배상액도 매년 증가하여 총 44억3800만원에 달하는 것으로 나타남. 포트홀에 바퀴가 빠질 경우 충격으로 인해 차량이 파손될 수 있으며, 이를 회피하려다 교통사고가 발생할 위험도 있음. 따라서 도로공사는 정기 및 수시 점검을 통해 포트홀 발생시 신속히 복구하고 포트홀 발생을 선제적으로 예방할 수 있는 대책을 마련해야 함.

특히 포트홀은 초기 발생 당시 즉각적으로 처리하지 않으면 포트홀의 깊이와 크기가 기하급수적으로 빠르게 깊어지고 커지므로 초반 신속한 대처가 필수적임.

## 인공지능으로 도로 위 '포트홀' 자동 탐지

김연균 기자 | 승인 2021.08.24 11:27 | 댓글 0



한국건설기술연구원  
5개 AI 추론 모델 완성  
예측-실제 파손 영역 비교



도로 위 포트홀 발생 정보를 한눈에 파악할 수 있는 AI 기반 감지 기술이 개발됐다. [사진=서울시]

일반적으로 포트홀 탐지 방식은 진동 기반 탐지, 레이저 계측 기반 탐지 및 영상 인식 기반 탐지로 구분됨. 진동 기반 탐지 방식은 저렴한 비용으로 도로 노면 상태를 실시간 평가하는 것에 적합하나 차량 바퀴의 주행 경로 외의 노면 손상을 탐지할 수 없으며, 노면 손상의 크기를 파악할 수 없다는 단점이 있음. 레이저 계측 기반 탐지는 광범위한 영역에서 발생한 다량의 포트홀을 신속하게 탐지 해내는 것에 다소 부적합하며 비용적 부담이 큼. 이에 반해 영상 인식 기반의 탐지 방식은 합리적인 비용으로 넓은 영역의 노면 상태를 분석할 수 있으며 최근 AI 기술의 발달로 더욱 주목받는 방식임. 특히 영상 인식 기반의 탐지방식은 지자체에 기술을 보급하기 용이하며, 탐지 결과가 사진으로 저장 되기 때문에 관련 전문가가 아닌 일반인도 직관적으로 파손 정보를 파악할 수 있다는 장점이 있음.

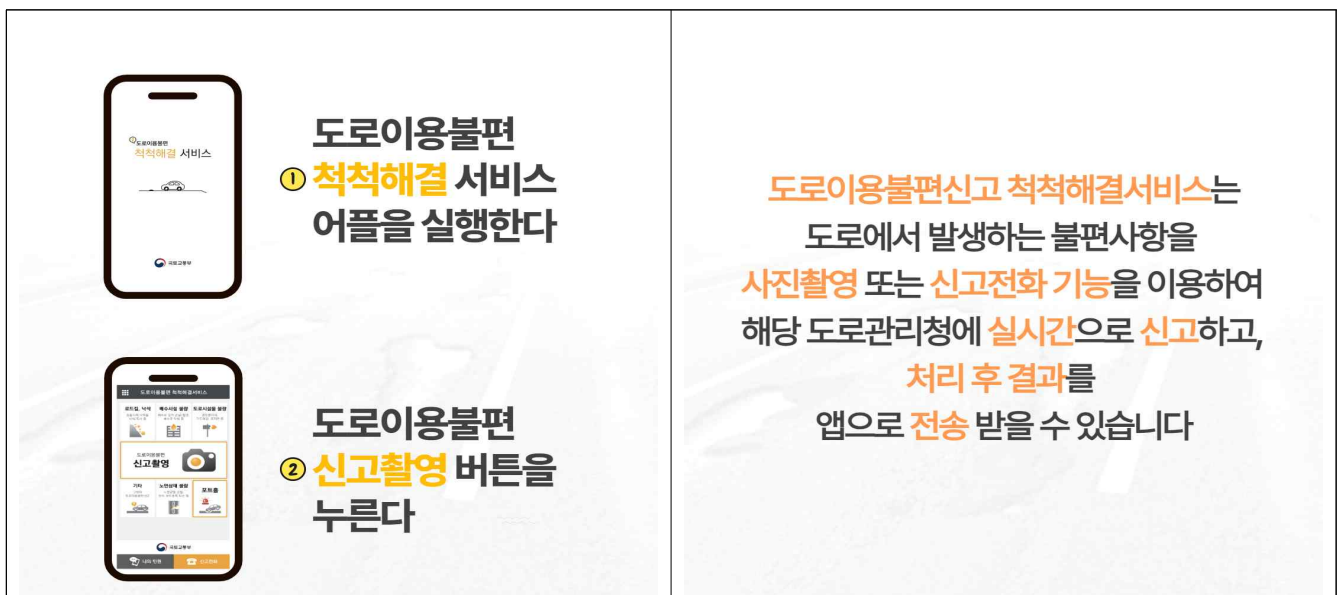
- 기존 서비스의 한계점

## 서울시 '택시 포트홀 신고시스템' 최우수 협업사례 선정

한철 기자 | ② 입력 2014.12.15 13:38 | 댓글 0



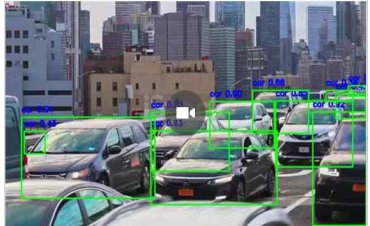


택시 포트홀 신고시스템은 서울시-한국스마트카드-서울개인택시운송사업조합이 체결한 업무 협약으로, 개인택시 운전자가 운전 중에 도로파손 발견시 택시 내 설치된 카드결제기의 버튼을 눌러 포트홀의 위치를 신고할 수 있음. 도로파손 위치는 카드결제기에 내장된 위치정보 추적기능(GPS)으로 한국스마트카드사의 택시정보시스템에 자동 전송되어 신고·접수·보수일시 등을 웹 지도에서 확인할 수 있음. 그러나 실제 개인택시 기사를 인터뷰 해본 결과, 직접 신고와 신고 후 확인 절차가 이루어지는 번거로움이 존재하는 것으로 드러남.





현재 국토교통부 시행 서비스인 "도로이용불편신고 적적해결서비스"는 사진 촬영 또는 전화로 사용자가 운전 중 직접 어플리케이션을 조작하여 수동으로 신고를 접수해야한다는 불편함이 존재함. 이러한 필요에 따라 도로공사가 신속하게 신고를 접수하고 보수하기 위해 손쉽게 포트홀을 신고하고 대응할 수 있는 시스템을 구축하고자함.

## 1.2 프로젝트 목적 및 필요성

목적	산출 목표	성과 목표	예시
2차	<ul style="list-style-type: none"> <li>이미지 전처리 (객체 분류, 바운딩박스 추출)</li> <li>모델 (모델 종류, 학습 정확도, 성능)</li> </ul>	<ul style="list-style-type: none"> <li>비실시간 동적 객체 인식</li> <li>객체 인식 정확도 90% 이상, 손실 1% 이하</li> </ul>	
3차	<ul style="list-style-type: none"> <li>자율주행 핵심 기술 구현을 위한 실시간 동적 객체 인식 (학습모델 최적화, 영상 객체 인식)</li> </ul>	<ul style="list-style-type: none"> <li>자율주행 핵심 기술 구현을 위한 실시간 동적 객체 인식</li> <li>실시간 동적 객체 인식 정확도 92% 이상, 손실 2% 이하</li> </ul>	
3차 최종	<ul style="list-style-type: none"> <li>자율주행 로봇을 이용한 객체 탐지 및 안전사고 감지</li> </ul>	<ul style="list-style-type: none"> <li>자율주행 로봇이 실시간 영상을 분석하여 포트홀을 인식하고, 위험 지역 정보를 신고하는 서비스</li> </ul>	

=> 실시간 동적 객체 인식은 자율주행에서 보행자, 차량 등 주변 환경의 변화를 즉각적으로 파악하고 반응하는 핵심 기술로, 안전하고 효율적인 주행을 보장하는 필수 요소

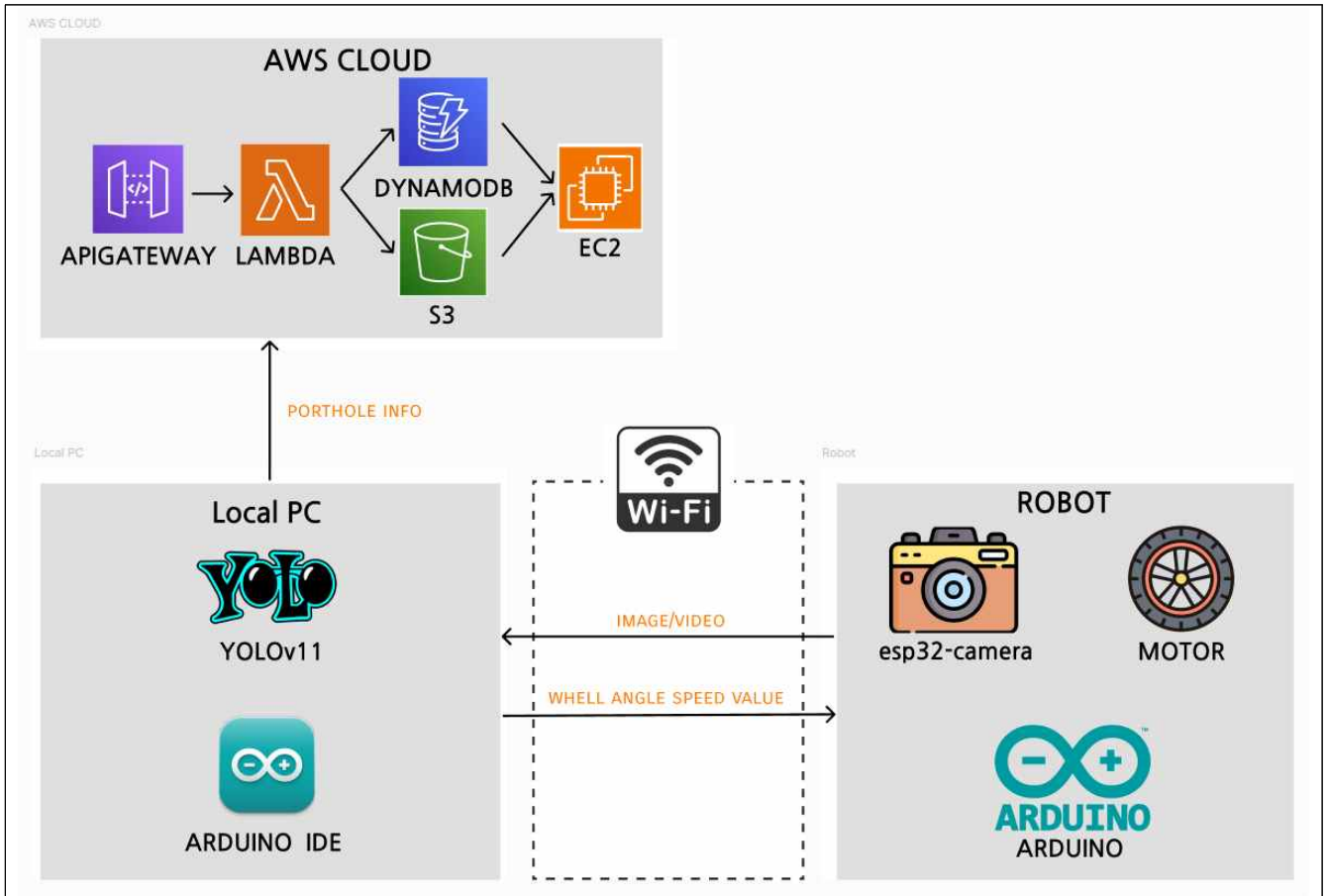
## 2. 프로젝트 방향

### 2-1 자율주행 로봇 객체 인식 기반 행동 제어 서비스

단계	설명	세부 단계	행동 및 출력 결과
1	로봇 카메라 영상 캡처 및 전송	<ul style="list-style-type: none"> <li>ESP-32CAM을 통한 Wi-Fi 전송</li> <li>프레임별 캡처 및 전송</li> </ul>	<ul style="list-style-type: none"> <li>실시간으로 PC에 영상 프레임 전송</li> </ul>
2	카메라를 통한 로컬 스트리밍	<ul style="list-style-type: none"> <li>로컬 IP 스트림 제공 (http://192.168.1.100:81/stream)</li> </ul>	<ul style="list-style-type: none"> <li>실시간 스트리밍 가능</li> </ul>
3	YOLO 기반 실시간 객체 인식	<ul style="list-style-type: none"> <li>Python/OpenCV로 스트림 연결</li> <li>영상 캡처 및 데이터 전처리</li> <li>YOLO 모델을 통한 학습</li> </ul>	<ul style="list-style-type: none"> <li>감지된 객체에 바운딩 박스를 그려 화면에 출력</li> </ul>
4	객체 분류 및 행동 태그 매칭	<ul style="list-style-type: none"> <li>분류된 객체별 사전 정의된 행동 태그와 매칭</li> </ul>	<ul style="list-style-type: none"> <li>분류된 객체에 따라 행동 준비</li> </ul>
5	상황에 따른 행동 선택	<ul style="list-style-type: none"> <li>정지: 즉각적인 위험</li> <li>느리게 이동: 즉각적인 위험이 아닐경우</li> </ul>	<ul style="list-style-type: none"> <li>정지: 모터 정지 및 브레이크</li> <li>느리게 이동: 속도 감소하여 천천히 이동</li> </ul>

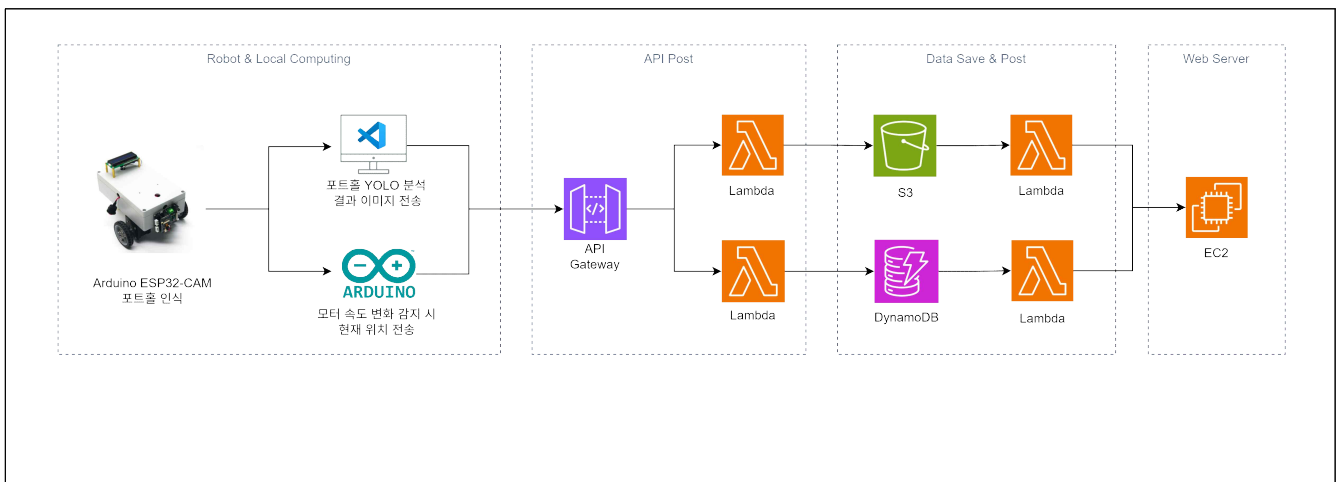
### 3. 아키텍처

#### 3.1 시스템 아키텍처



AWS Cloud	Local PC	Robot	통신
<ul style="list-style-type: none"> <li>API Gateway &amp; Lambda: 클라우드에서 요청을 처리하고 데이터를 관리</li> <li>DynamoDB &amp; S3: 데이터 저장 및 관리</li> <li>EC2 : 서버 컴퓨팅</li> </ul>	<ul style="list-style-type: none"> <li>YOLOv11: 객체 인식 모델 실행</li> <li>Arduino IDE: 로봇 제어 코드 작성 및 업로드</li> </ul>	<ul style="list-style-type: none"> <li>ESP32-Camera: 실시간 영상 수집 및 전송</li> <li>Arduino: 모터 및 로봇 동작 제어</li> </ul>	<ul style="list-style-type: none"> <li>Wi-Fi: 로컬 PC와 로봇간 영상 및 데이터 송수신</li> </ul>

#### 3.2 클라우드 아키텍처

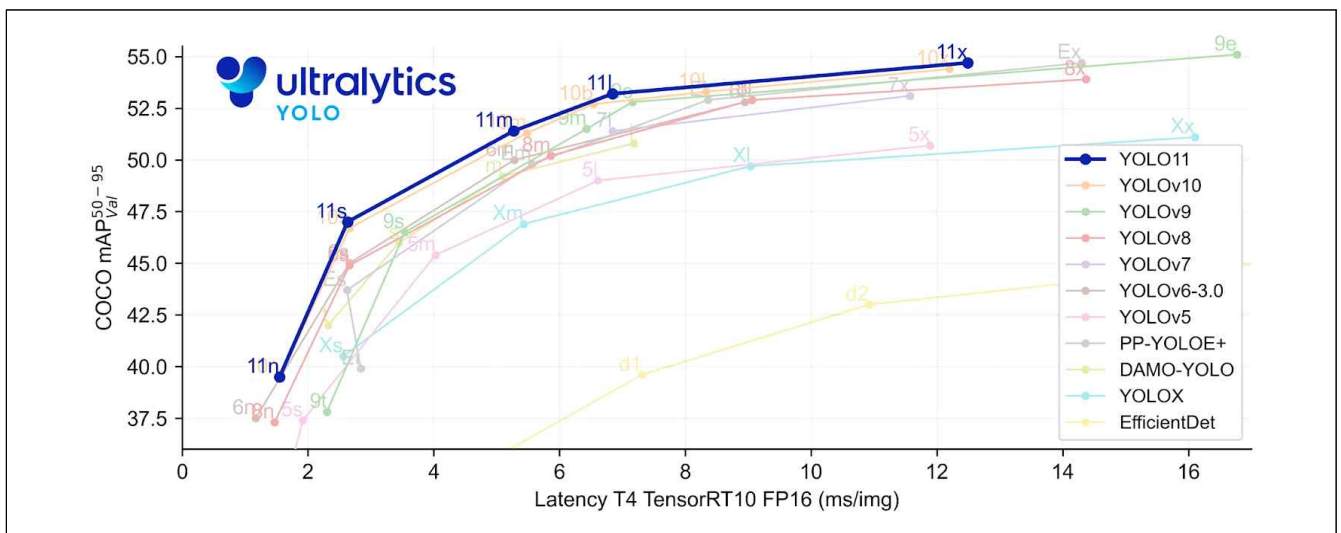


인프라 설정	컴퓨팅 리소스	스토리지	네트워크 설계
<ul style="list-style-type: none"> <li>API Post: ESP32-CAM에서 수집한 데이터 AWS 클라우드로 전송</li> <li>Data Save &amp; Post: S3 - 이미지 데이터 저장, DynamoDB - 위치 정보 데이터 저장</li> <li>Web Server: 포트홀 신고 웹페이지 구축</li> </ul>	<ul style="list-style-type: none"> <li>AWS Lambda: 서버리스 컴퓨팅을 활용하여 데이터 처리, 저장, 트리거 작업 수행</li> <li>EC2: 포트홀 신고 Web Server 구축</li> </ul>	<ul style="list-style-type: none"> <li>Amazon S3: 촬영된 분석 결과 이미지 데이터 저장</li> <li>DynamoDB: 객체 감지 데이터(위치)에 대한 정보 저장을 위한 빠르고 확장 가능한 데이터베이스</li> </ul>	<ul style="list-style-type: none"> <li>로컬 네트워크: ESP32-CAM이 Wi-Fi를 통해 클라우드와 연결</li> <li>클라우드 네트워크: Lambda 함수가 데이터를 처리하고 필요한 스토리지로 전송, EC2는 웹 인터페이스를 제공</li> </ul>

## 4. 모델 선정 배경

### 4.1 YOLOv5 YOLOv11 비교

- 초기에는 YOLOv5의 안정성과 검증된 성능을 고려하여 모델을 구성했으나, 프로젝트 진행 중 더 높은 정확도 (mAP)와 빠른 응답 시간이 요구됨에 따라 YOLOv11으로 변경
- YOLOv11은 최신 기술을 반영해 더 높은 정확도와 낮은 지연 시간을 제공



성능 지표	YOLOv5n	YOLOv11n	개선 사항
mAP (val 50-95)	34.3	39.5	+5.2 (15.2% 향상)
추론 속도 (CPU ONNX, ms)	73.6	56.1	17.5ms 빨라짐 (23.8% 향상)
파라미터 수 (M)	2.6	2.6	동일
FLOPs (B)	7.7	6.5	1.2B 감소 (15.6% 효율 향상)



## 5. 모델 학습

### 5.1 YOLO 모델 학습 구현 과정

-데이터 셋 : 00장

현재 성능 모델 구축						
	train			val		
성능지표	box	cls	df	box	cls	df
loss	0.941	0.807	1.160	1.579	1.279	1.643
성능지표	metrics/mAP_0.5			metrics/mAP_0.5:0.95		
mAP(정확도)	0.681			0.339		

-데이터 셋: 00장

목표 성능 모델 구축						
	train			val		
성능지표	box	cls	df	box	cls	df
loss	0.3이하	0.3이하	0.3이하	0.3이하	0.3이하	0.3이하
성능지표	metrics/mAP_0.5			metrics/mAP_0.5:0.95		
mAP(정확도)	0.9			0.8		

데이터 전처리	데이터 모델 학습	모델 최적화	테스트 및 검증
<ul style="list-style-type: none"> <li>이미지 리사이징</li> <li>라벨링</li> </ul>	<ul style="list-style-type: none"> <li>학습 진행</li> <li>YOLO 모델 사용</li> <li>학습 중 모니터링</li> </ul>	<ul style="list-style-type: none"> <li>후처리 최적화</li> <li>f1 스코어, 정확도, 손실 함수 결과</li> </ul>	<ul style="list-style-type: none"> <li>테스트 데이터 셋 평가</li> <li>실시간 성능 테스트</li> </ul>

## 6. SW 및 라이브러리

### 6.1 Local 라이브러리


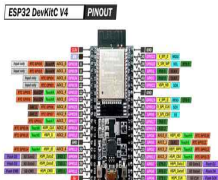


이름	버전	이유
Python	3.7 ~	YOLOv5는 최신 Python 버전에서 잘 동작하지만, PyTorch 및 기타 라이브러리의 호환성을 위해 3.7 이상이 좋음
PyTorch	2.0 ~	최신 PyTorch 버전은 GPU 가속, 최적화된 연산 등을 지원하므로, CUDA 버전에 따라 버전을 설정하였음
OpenCV	4.1 ~	실시간 이미지 처리, 비디오 스트림 처리에 사용할 때 해당 버전 권장
ultralytics	8.6 ~	실시간 객체 인식 성능을 극대화하기 위해 YOLOv5를 TensorRT로 변환하는 NVIDIA GPU를 사용하는 경우 유용하며 해당 cuda 버전에 따라 설정

### 6.2 Arduino 라이브러리

이름	이유
WiFi.h	ESP32의 Wi-Fi 연결을 설정하기 위해 사용
ESPAsyncWebServer.h	실시간 영상 스트리밍 서버를 구현하기 위해 사용
Adafruit_Sensor.h	센서 및 이미지 처리에 사용
ArduinoJson.h	로컬 PC와 데이터 통신 시 JSON 형식으로 데이터를 송수신할 때 사용
<HttpClient.h>	Arduino ide에서 클라우드로 데이터 전송을 위해 사용
<Arduino.h>	기본 GPIO(모터 연결)를 제어하기 위해 사용
<esp_camera.h>	ESP32-cam 카메라를 제어하기 위해 사용

## 7. 부록

### 7.1 아두이노 자율주행 제품

이름	기능	이미지
Arduino ESP32-CAM 자동차 키트	YOLO 분석을 통해 동작되어야 하는 차체	
ESP32	WIFI 모듈이 내장되어 로컬 PC와 통신하며, 모터 제어 담당	
ESP32-CAM	ESP32와 연결하여 카메라 기능을 담당	
18650 배터리 2개, 충전기, 홀더 세트	모터 제어에 있어서 ESP32를 동작시키기 위한 전원 공급처	

### 7.2 팀원 역할

이름	역할
김승엽(팀장)	PM
한은선	Data Analyst
김동혁	code worker
김준희	code worker
박장원	Cloud Engineer
김나현	Cloud Engineer