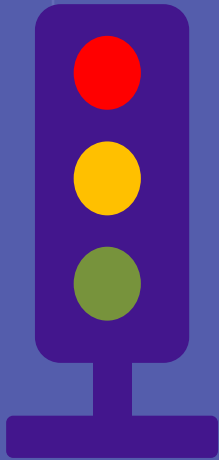


STLS

: Smart Traffic Light System

1st Implementation

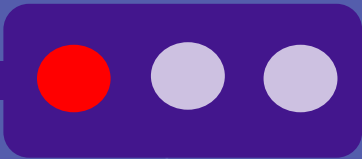


201411285
유종혁

201411286
유환규

201611308
최준오

201714166
신예슬



목차

1. 프로젝트 소개
2. 프로젝트 설명
3. 데모
4. 진행상황
5. 제약사항
6. 차후 계획

1. 프로젝트 소개

바쁜데 신호등은 언제 켜지는 거야



기존 도로 신호등 시스템 :

기존 도로 신호 시스템은 항상 정해진 스케줄대로 운영된다. (초록불 1분 30초, 좌회전 신호 2분 등)

따라서 실시간으로 변하는 도로 상황에 대응하지 못하기 때문에 비효율적이다.

대기하는 차량이 많은 곳에 신호를 우선적으로 줄 순 없을까?



딥러닝 기술을 이용해서 만들어 볼까?



실시간 신호등 시스템 :

딥러닝 기술을 이용해서 실시간으로 차량의 존재 여부 뿐만 아니라 대수를 파악하고 그에 따라 신호체계를 동적으로 변경하여 효율적인 교통체계를 구축한다.

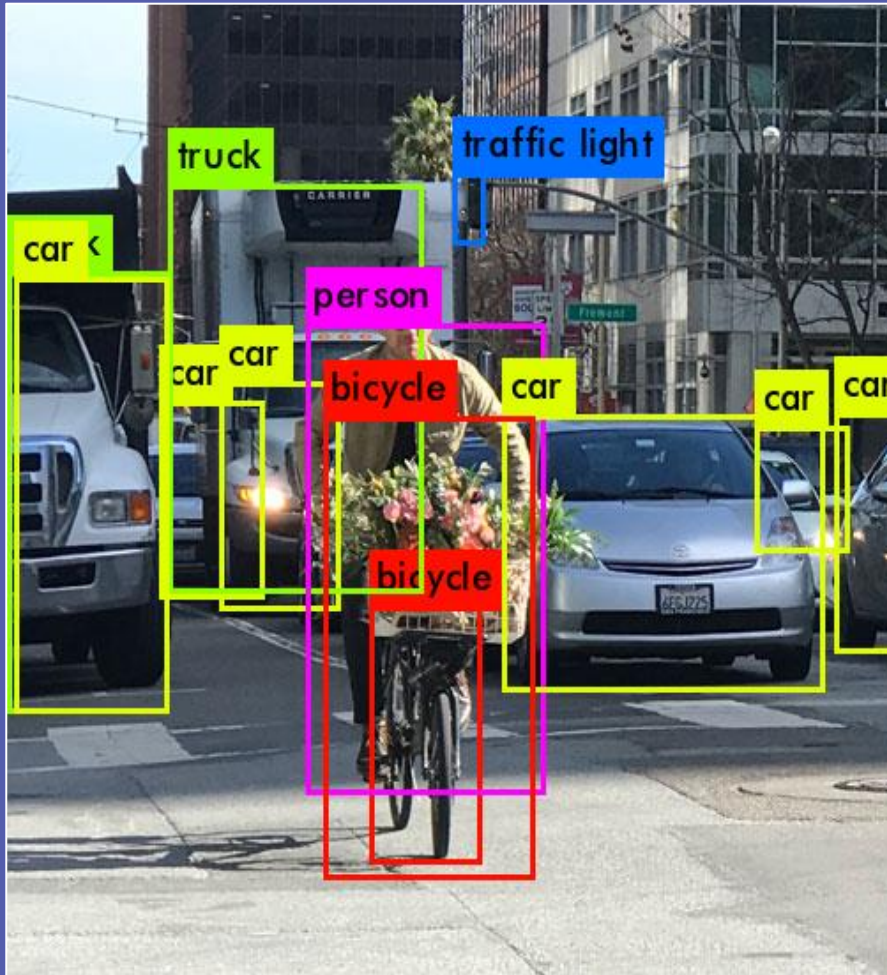


1. 프로젝트 소개

작품명 : Smart Traffic Light System

- 딥러닝 기술(Faster R-CNN)을 응용해 효율적인 신호체계를 구축한다.
- 실시간으로 도로 상황을 촬영하여 자동차의 대수를 파악하고 교통량이 많은 도로에 초록 신호를 우선적으로 부여한다.
- 도로에 구급차가 진입 시, 해당 도로에 높은 우선순위를 부여한다.
- 횡단보도의 보행자 수를 파악하고 교통량을 고려해서 보행자 신호를 조작한다.

1. 프로젝트 소개



Deep Learning Object Detection

Input : 현재 교통 상황 스냅샷

Output : 스냅샷에 존재하는 객체들을 분류하고 각 객체의 수를 카운팅한 결과

=> 객체의 수를 이용하여 도로 신호별 우선순위를 부여하여 신호 스케줄링

1. 프로젝트 소개

프로젝트 구성

1. 딥러닝 서버

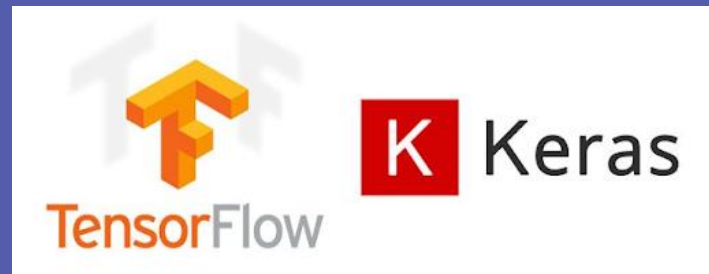
- Google Colab GPU 런타임 이용
- Tensorflow + Keras로 구현

2. 신호 관리 시스템

- 파이썬 GUI : PyQt5
- 신호 스케줄링 알고리즘 구현
- 딥러닝 서버와 통신

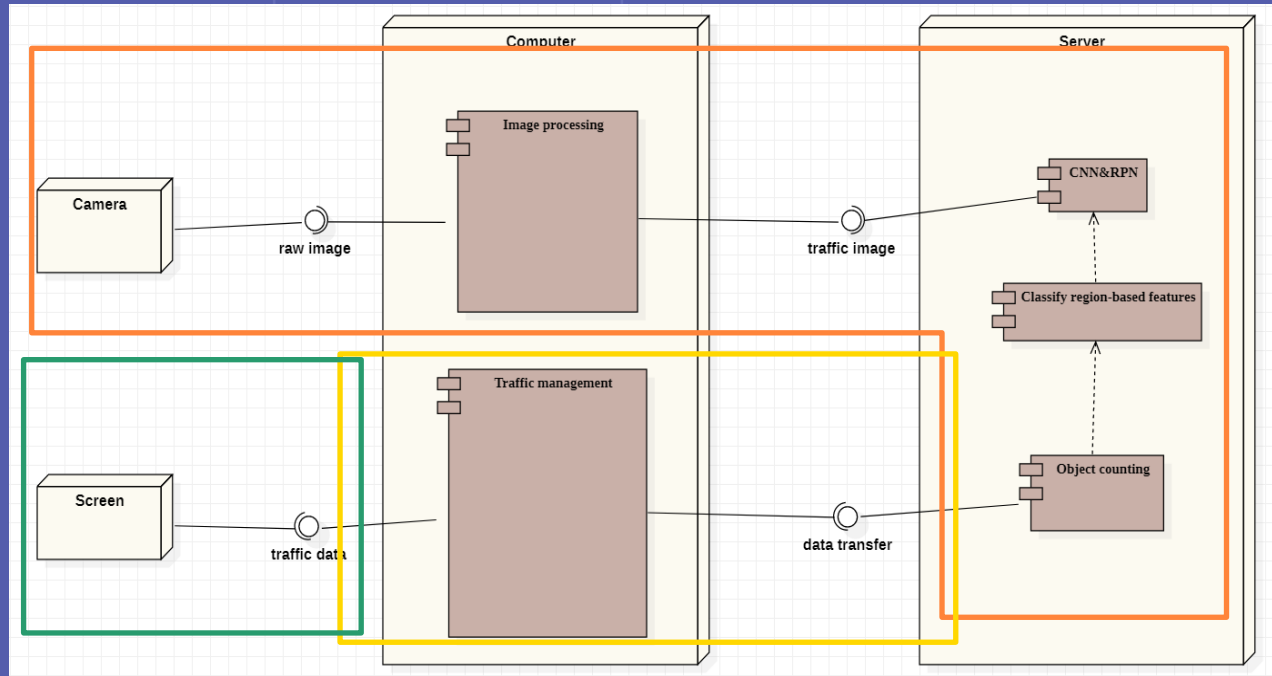
3. 시뮬레이션 세트장

- 사거리 도로 형태
- 1.5m x 1.5m 크기
- 차량 모형과 사람 모형 활용
- 4대의 웹캠으로 도로상황 캡처



2. 프로젝트 설명

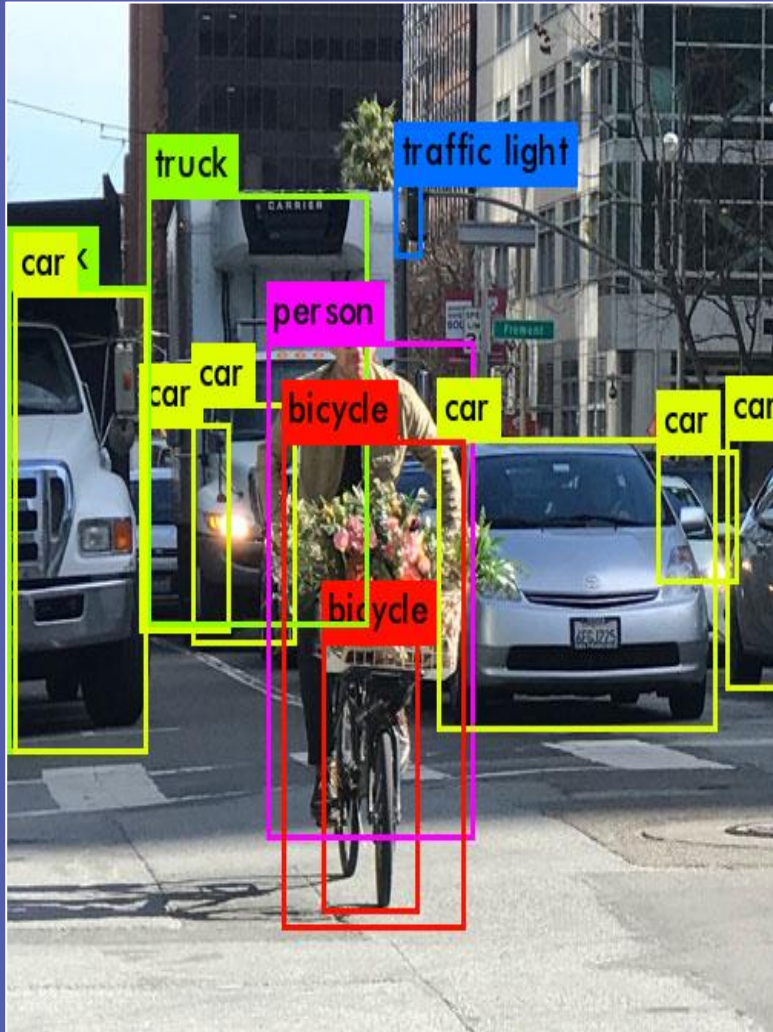
(1) 배치 다이어그램



- 웹캠으로부터 이미지를 받아서 자동차 객체, 사람 객체를 추출하고 추출한 객체 각각의 수를 센다.
- 자동차 수와 사람 수 데이터를 이용해 다음 신호에 부여할 시간을 계산한다.
- 계산된 시간을 바탕으로 화면에 현재 신호등의 상태를 보여준다.

2. 프로젝트 설명

(2) 딥러닝 모델



- 딥러닝 모델을 사용하여 효율적인 신호등 시스템을 만든다.

- 사용할 딥러닝 모델 : Faster R-CNN

기존 Fast R-CNN의 region proposal 방식인 selective search에서 region proposal network(RPN) 방식으로 변경하여 GPU를 통한 RoI 계산이 가능해지고 더 높은 정확도를 낼 수 있게 된 기술.

- Faster R-CNN을 이용해 이미지를 학습시키고 테스트를 통하여 객체 검출의 정확도와 객체 검출에 소요된 시간을 확인한다.

- 테스트시 원하는 결과가 나오도록 필요한 부분을 수정하여 사용한다.

- 신호관리 시스템에서 네트워크로 이미지를 받도록 수정.

- 객체 정의, 바운딩 박스 생성시 각 객체별로 개수를 카운팅하도록 수정.

- 객체별 카운팅 된 데이터를 네트워크로 신호관리 시스템에 보내주도록 수정.



2. 프로젝트 설명

(3) 신호 스케줄링

설명 : 도로에 설치된 카메라로 찍힌 이미지에서 파악된 차량 수와 사람 수를 토대로 현재 신호가 끝날 때 다음 신호의 시간을 결정한다.

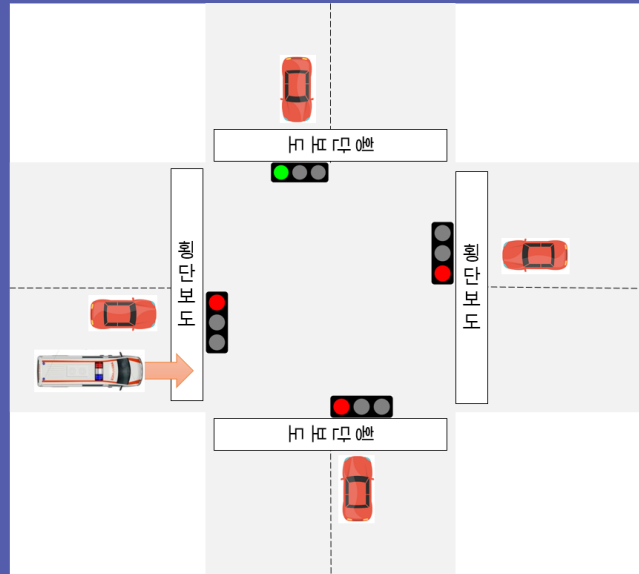
- 기본 메커니즘

- (1) 위에서 아래로 보는 것으로 기준을 삼을 때 왼쪽도로, 오른쪽도로, 아래쪽도로, 위쪽도로 순으로 신호를 준다.
- (2) 도로 신호등의 기본값은 40초, 보행자 신호의 기본값은 20초이다.
- (3) 신호 스케줄링은 한 신호에서 다음 신호로 바뀌기 2초 전, 황색 신호로 들어갔을 때 각도로의 차량대수, 보행자수를 계산하여 도로의 가중치로 환산하여 도로신호등의 기본값인 40초를 기준으로 다음 신호가 다른 신호에 비해 상대적으로 가중치가 높다면 신호의 시간을 늘려주고, 가중치가 상대적으로 적다면 신호의 시간을 줄여준다. 여기서 신호의 최대 지속시간은 60초, 최소 지속시간은 20초로 한다.
- (4) 가중치는 보행자 수와 차량 수를 이용하여 계산하지만, 보행자 수만 많은 경우에는 도로 신호 시간에 반영하지 않는다.

2. 프로젝트 설명

- 예외 사항

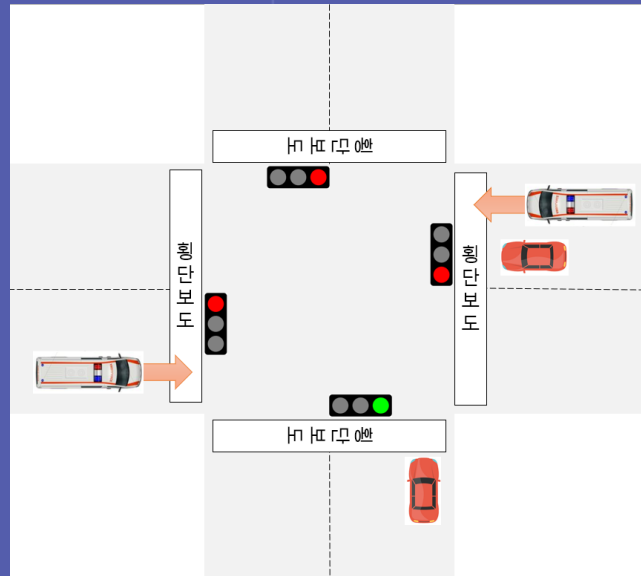
- (1) 구급차 출현 : 도로에 구급차가 나타났을 때 신호 스케줄링은 현재 신호를 잠시 멈추고 구급차가 나타난 도로 쪽에 신호를 주어야 한다.
- (1) - 1. 현재 켜진 도로에 구급차가 있는 경우, 신호의 남은 시간이 10초 이하인 경우에는 다시 10초를 부여해주고, 10초보다 많이 남아있는 경우 그대로 유지한다.



- (1) - 2. 현재 켜진 도로가 아닌 경우, 현재 켜진 도로의 시간이 10초 이하 인 경우에는 그대로 시간을 유지하고 다음 신호로 구급차가 있는 도로의 신호를 10초 부여한다.
- (1) - 3. 현재 켜진 도로가 아닌 경우, 현재 켜진 도로의 시간이 10초 이상인 경우에는 도로의 시간을 10초로 줄이고, 다음 신호로 구급차가 있는 도로의 신호를 10초 부여한다. 그 후 원래 켜졌던 도로에 남은 시간만큼의 시간을 다시 부여한다. 이때 남은 시간이 5초 이하라면 다음 신호로 넘어간다.

2. 프로젝트 설명

- 예외 사항



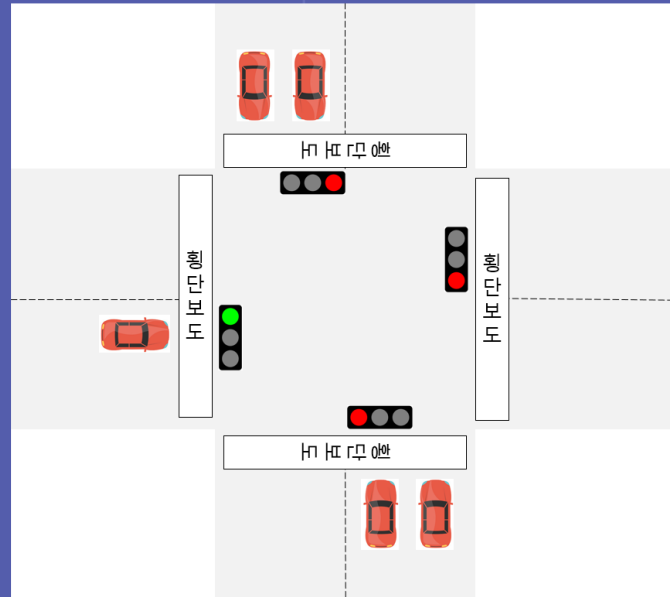
(1) - 4. 현재 켜진 도로가 아닌 경우, 현재 켜진 도로의 시간이 10초 이상인 경우에는 도로의 시간을 10초로 줄이고, 다음 신호로 구급차가 있는 도로의 신호를 10초 부여한다. 그 후 원래 켜졌던 도로에 남은 시간만큼의 시간을 다시 부여한다. 이때 남은 시간이 5초 이하라면 다음 신호로 넘어간다.

(1) - 5. 구급차가 통과해야 하는 도로의 보행자 신호가 켜져 있을 경우에는 10초 후에 도로 신호는 적색으로 변경하고 보행자 신호가 끝날 때까지 기다린 후에 구급차 쪽 도로신호를 10초 부여한다.

2. 프로젝트 설명

- 예외 사항

- (2) 도로에 차와 보행자 모두 없을 경우 : 도로에 차와 보행자가 없을 경우에는 해당 도로에 신호를 부여하지 않고 다음 도로 신호를 부여한다.



- (2) - 1. 다음 신호가 켜질 도로의 차량과 보행자가 없을 경우에는 해당 도로의 신호를 부여하지 않고, 그 다음 도로의 신호로 넘어간다.

- (2) - 2. 모든 도로에 차량과 보행자가 없을 경우에는 신호를 전부 적색신호로 유지하다가 새로운 차량이나 보행자가 들어오면 그 도로부터 다시 한 사이클의 신호 스케줄링을 진행한다.

3. 데모

main

인공지능 신호등시스템 관리

시스템 시간

거리	차량 수	구급차 수	사람
거리1	2	0	0
거리2	1	0	0
거리3	2	0	0
거리4	2	0	0

차량신호	신호	흐른시간	부여시간
차량신호1	적색	0	0
차량신호2	적색	0	0
차량신호3	적색	0	0
차량신호4	녹색	19.0	33
보행자신호1	녹색	19.0	20
보행자신호2	적색	0	0
보행자신호3	적색	0	0
보행자신호4	적색	0	0

차량신호

차량신호등	상태	부여시간	흐른시간
차량신호등1	적색	0	0
차량신호등2	녹색	40	3.9
차량신호등3	적색	0	0
차량신호등4	적색	0	0

보행자신호

보행자신호	상태	부여시간	흐른시간
보행자신호1	적색	0	0
보행자신호2	적색	0	0
보행자신호3	녹색	20	3.9
보행자신호4	적색	0	0

신호 스케줄링 시스템의 UI 모습

- 딥러닝 서버로부터 차량, 구급차, 사람 수를 카운팅한 데이터를 받아서 스케줄링 알고리즘에 맞게 신호 시간을 조절

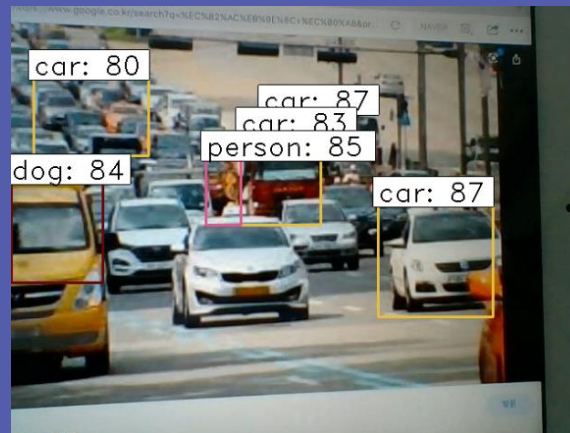
3. 데모

```
... Elapsed time = 0.45918846130371094
[('train', 72.85200953483582), ('train', 72.77485132217407), ('car', 93.28669309616089), ('car', 92.84161925315857), ('car', 83.17198157310486), ('car',
ack 받음
1 번째 파일
size: 87926
파일 [1.jpg] 전송종료. 전송량 [87926]
2 번째 파일
size: 86543
파일 [2.jpg] 전송종료. 전송량 [86543]
3 번째 파일
size: 85785
파일 [3.jpg] 전송종료. 전송량 [85785]
4 번째 파일
size: 85263
파일 [4.jpg] 전송종료. 전송량 [85263]
1.jpg
Elapsed time = 0.542186975479126
[('train', 85.11706590652466), ('car', 96.97282314300537), ('car', 89.90755081176758), ('car', 88.25700879096985), ('car', 79.23240661621094), ('car', 7
2.jpg
Elapsed time = 0.47322988510131836
[('car', 94.97711062431335), ('car', 80.36147356033325), ('car', 76.61072611808777), ('car', 76.29255652427673), ('train', 73.76813888549805)]
3.jpg
Elapsed time = 0.47130537033081055
[('train', 80.07380366325378), ('car', 97.47738242149353), ('car', 96.73161506652832), ('car', 87.94236779212952), ('car', 77.59767770767212), ('car', 7
4.jpg
Elapsed time = 0.4705688953399658
[('train', 95.43592929840088), ('car', 98.33458662033081), ('car', 93.27161312103271), ('car', 83.56776237487793)]
```

Google Colab 상에서 딥러닝 모델 테스트 진행중인 모습

신호 스케줄링 컴퓨터로부터 TCP/IP 소켓통신으로 사진을 받아오고 테스트 후 객체 카운팅 데이터 전달

3. 데모



태블릿에 인식할 사진 보이기 -> 노트북 웹캠으로 캡처 -> 딥러닝 모델 결과 도출

화질열화 + 웹캠 성능의 문제로 인식을 저조. 실제 파일로 하면 더 높은 정확도를 보임
= 실제 세트장을 찍을 시 정확도가 더 높을 것으로 예상

4.기능 및 비기능 진행상황

TEST no.	TEST DESCRIPTION	1 st implementation prototyping기능별 진행사항 확인 및 2 nd SRS 반영 예정
1.1기능	0.5 초에 1장씩 원하는 도로 상황의 사진을 촬영하고 이미지를 모델에 맞게 변환한다	현재 하나의 웹캠에서 도로 4방향의 이미지 정보를 얻어오는 프로토타입 기능 구현완료. 실제로는 4개의 웹캠에서 각 도로의 상황을 얻어올 예정.
1.2기능	기능 객체의 bounding box를 확인하고 객체가 있을 만한 부분을 검출한다	Faster-RCNN 260번 학습 후 진행 test 결과 객체 검출을 하였으나 정확도 미흡. 학습시간 부족으로 Pre Train Model에서 추가 학습 방식으로 객체 검출할지 논의 중으로 2 nd SRS 반영 예정.
1.3 기능	전송 받은 이미지에서 자동차, 응급차, 사람의 객체가 식별됨을 확인한다	기존 데이터 셋으로 자동차와 사람 객체의 경우 어느 정도 식별 가능하나 구급차 식별 부족으로 구급차Dataset 추가 학습 필요.
1.4기능	구급차가 아닌 일반 자동차의 경우 객체를 탐지한 후 대수를 정확히 셀 수 있는지 확인한다	일반 자동차 객체 탐지 후 객체 수를 셀 수 있으나 이미지 객체 탐지 정확도 부족으로 미흡
1.5기능	신호시스템의 결과가 UI에 잘 반영되는지 확인한다	받아온 도로 정보를 UI에 반영하는 기능 구현 성공. 모델로부터 받아온 도로 정보를 이용하여 도로신호 유지시간을 동적으로 변경하여 반영하고 MainUI에 각 도로에 검출된 자동차, 사람, 구급차를 보여주는 기능 구현 성공

4.기능 및 비기능 진행상황

TEST no.	TEST DESCRIPTION	1 st implementation prototyping기능별 진행사항 확인 및 2 nd SRS 반영 예정
1.6 기능	기본 신호 스케줄링	사거리에 구급차가 없고 차량이 존재할 때 신호 유지 시간을 동적으로 변경하면서 신호등 순서대로 켜지는 기본 스케줄링 구현 완료
1.7 기능	구급차가 지나가는 특수한 경우 신호시스템에 예외처리가 되는지 확인. 구급차가 있는 신호를 최대한 빠르게 녹색신호를 부여하는 기능	<p>1)현재 녹색신호인 도로에 구급차가 있을 경우 현재 녹색 신호 유지시간이 10초 미만 남은 경우 남은 시간을 10초로 연장하는 기능 구현완료</p> <p>2) 현재 적색 신호를 가진 도로에서 구급차가 출현한 경우 현재 녹색 신호인 도로 유지시간이 10초 이상 남은 경우 10초로 줄인 후 구급차가 있는 쪽으로 신호를 변경하는 기능 구현완료</p>
1.8 기능	사람 정보에 따라 실시간으로 신호 변경이 되는지 확인	사람 객체 수를 파악하여 도로 실시간 신호시간 변경 기능 구현완료
1.9 기능	차량 및 보행자 전부 없을 경우 신호를 모두 적색 신호로 변경 기능 확인	<p>차량이 전부 없을 경우 신호를 모두 적색으로 변경 새로 들어온 차량, 구급차, 사람이 있는 신호부터 실행하는 기능 구현완료.</p> <p>1) 구급차가 있는 쪽을 먼저 실행</p> <p>2) 차량 및 사람이 제일 많은 곳을 먼저 실행</p>

4.기능 및 비기능 진행상황

TEST no.	TEST DESCRIPTION	1 st implementation prototyping기능별 진행사항 확인 및 2 nd SRS 반영 예정
2.1비기능	모니터를 통해 화면을 분할하여 도로 신호와 보행자 신호를 한 번에 표시할 수 있어야 한다.	진행 상황 도로 신호를 보여주는 UI 와 보행자 신호를 보여주는 프로토타입 UI 구현 . 도로 신호와 보행자 신호 표시 확인 완료.
2.2비기능	딥러닝을 이용한 Object detect의 결과를 0.5초 이내로 얻을 수 있어야한다	딥러닝 사진분석 한 장당 걸리는 시간이 0.8초 4장의 사진 분석이 필요하여 3.2초 걸린다. GPU의 성능상 3초 이하로 줄일 수 없기 때문에 딥러닝 사진분석 주기를 3초로 최종 SRS 반영 예정
2.3비기능	분석된 데이터를 이용한 신호 스케줄링이 0.5초 이내로 끝나야 한다.	분석데이터를 이용하여 신호 스케줄링을 0.5초 안에 하도록 구현실패. 데이터 받아오는 시간이 3~4초 걸리므로 받아들 데이터가 없으면 이전 교통정보로 상황 판단하여 분석하는 방식으로 성능 향상 예정

4.기능 및 비기능 진행상황

TEST no.	TEST DESCRIPTION	1 st implementation prototyping기능별 진행사항 확인 및 2 nd SRS 반영 예정
2.4비기능	구급차 통과시 1초 안에 신호변경 요청을 반영한다.	딥러닝 사진 분석 시간 3초 이상 소요되기 때문에 분석데이터가 준비가 안된 경우 구급차가 없는 것으로 판단하고 진행하는 방식으로 해결 예정
2.5비기능	Object detect의 정확도를 70% 이상으로 한다.	학습 횟수 부족으로 Object 정확도 70% 달성하지 못한다. 학습 및 Pretrained Model 추가학습 고려2 nd SRS에 반영 예정.
2.6비기능	기존 신호시스템보다 실시간 신호시스템 반영 시 성능 향상 10% 이상을 목표로 한다	실시간 신호시스템과 기존 신호시스템과의 차이 비교분석 구현 후 테스트 예정



5. 제약사항

(1) 코로나 상황

- 사회적 거리두기 2.5단계 격상에 따라 모임 장소가 없음.
- 세트장을 제작해야 하는데 제작할 장소와 보관 장소가 없음.
- 시뮬레이션을 위한 웹캠 구매 지연

(2) 학습시간 부족

- 고성능 GPU의 부재로 모델의 정확도를 높이기 위한 학습 시간이 부족
- Google Colab의 GPU 런타임 대여시간이 12시간으로 제약
- 미리 학습된 모델을 사용 시 구급차 데이터를 추가학습 시키기 어려움

