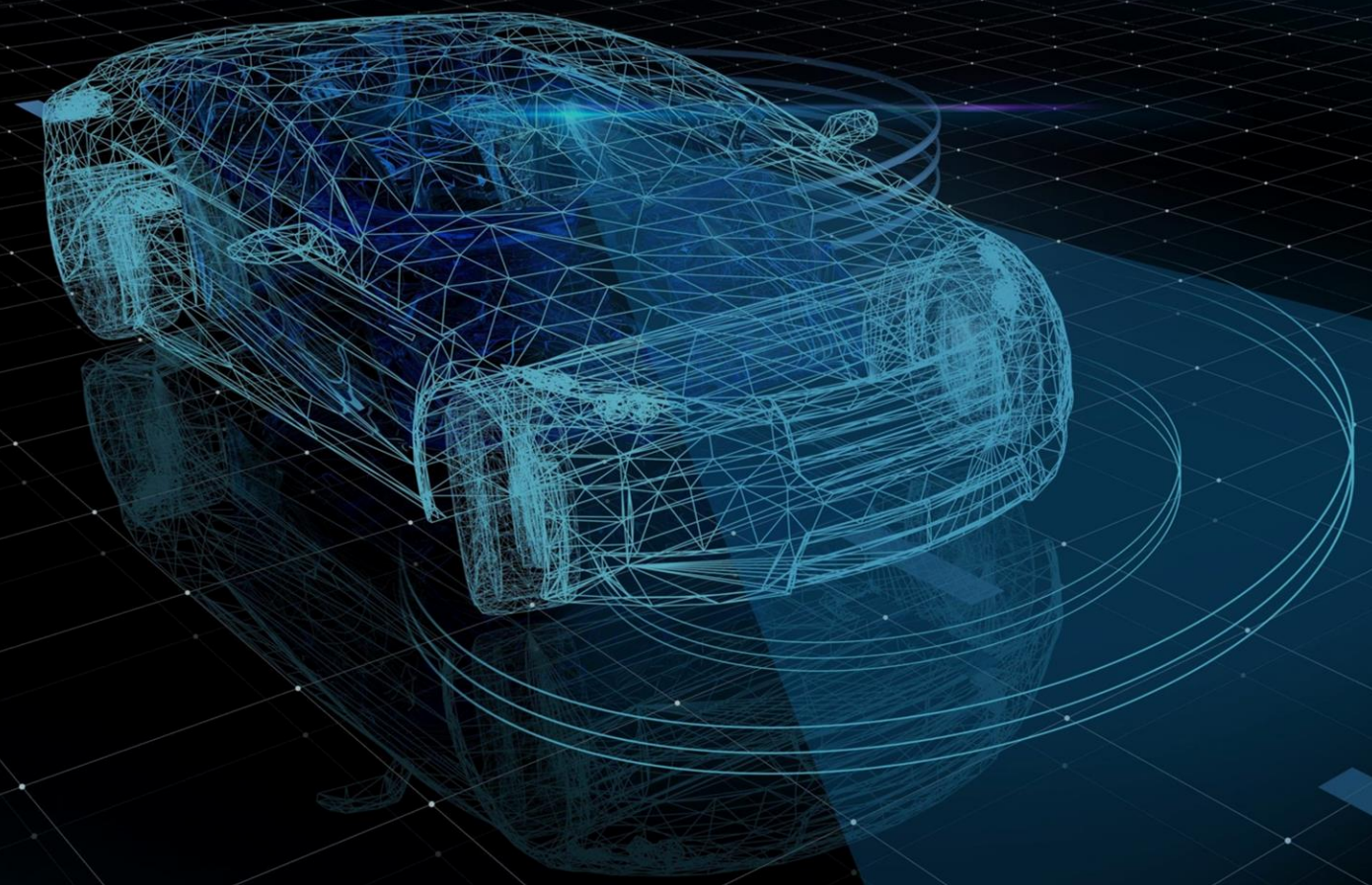


2022 국제 대학생
창작자동차경진대회

알파 프로젝트 중간 발표

인하대학교 TEAM THE GIGACHA

INHA UNIVERSITY

목차

01 **개발 일정 및 TEST 환경**
1년 프로세스 , 개발 및 테스트 환경

02 **기술 소개**
Vision , LiDAR , Localization, Planning, Control

03 **대회 미션**
트랙 미션 , 예/본선 미션

04 **결과 및 고찰**
대회 결과, 고찰 , 후배들에게 해주고 싶은 말

05 **Q&A**

1. 개발 일정 및 TEST 환경

- 전체 개발 일정표
- 주행 TEST 환경

전체 일정

		4월	5월	6월	7월	8월	9월	D-Day
Integration	▪ROS msg, node 설계 ▪GUI 구현 ▪각 파트 ROS 통합 및 미션 구현							
Vision	▪Camera & LiDAR Sensor Fusion ▪YOLOv4 - Object Detection ▪Open Dataset & Custom Dataset 수집							
LiDAR	▪Object Distance Estimation ▪Object Detection ▪Camera & LiDAR Sensor Fusion							
Local	▪ GPS, IMU Sensor Fusion by EKF ▪ Dead Reckoning ▪ Precise Map Making by QGIS							
Planning	▪path planning 알고리즘 적용 ▪본선 및 예선 미션 구현 ▪스레드 처리 ▪GUI 구현							
Control	▪lateral controller – Pure pursuit / Stanley ▪longitudinal controller – 곡률기반 속도제어							

주행 검증

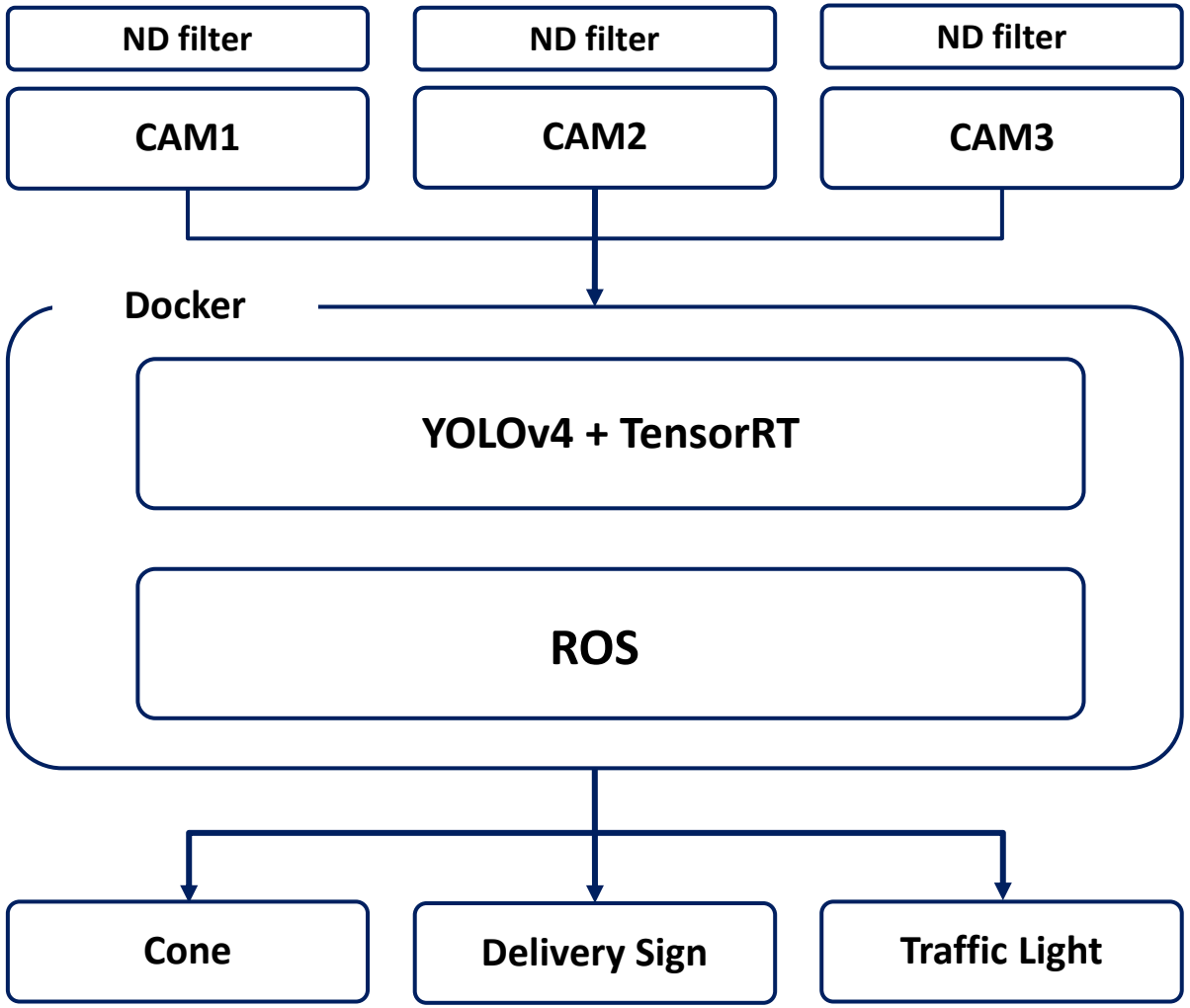
TEST 환경

서울대학교 시흥캠퍼스

2. 기술 소개

- VISION
- LiDAR
- Localization
- Planning
- Control

주요 기술 Flow Chart



Vision – H/W

Camera

USB 웹캠 Logitech Brio 4k 3대를 사용하여 미션에 맞게 Cone, Traffic light, Sign 을 각각 볼 수 있도록 차량에 마운트 함

ND filter

컬러밸런스를 유지시킨 채 전체적인 노출의 양을 감소시키는 기능을 하며,

역광 문제를 하드웨어적으로 해결하기 위해 webcam 전용 ND 필터가 없는 것을 고려하여 3D 프린팅으로 고정 케이스를 제작 후 부착

도커 (Docker)

리눅스 응용 프로그램들을 소프트웨어 컨테이너 안에 배치시키는 일을 자동화 하는 오픈 소스 프로젝트 애플리케이션

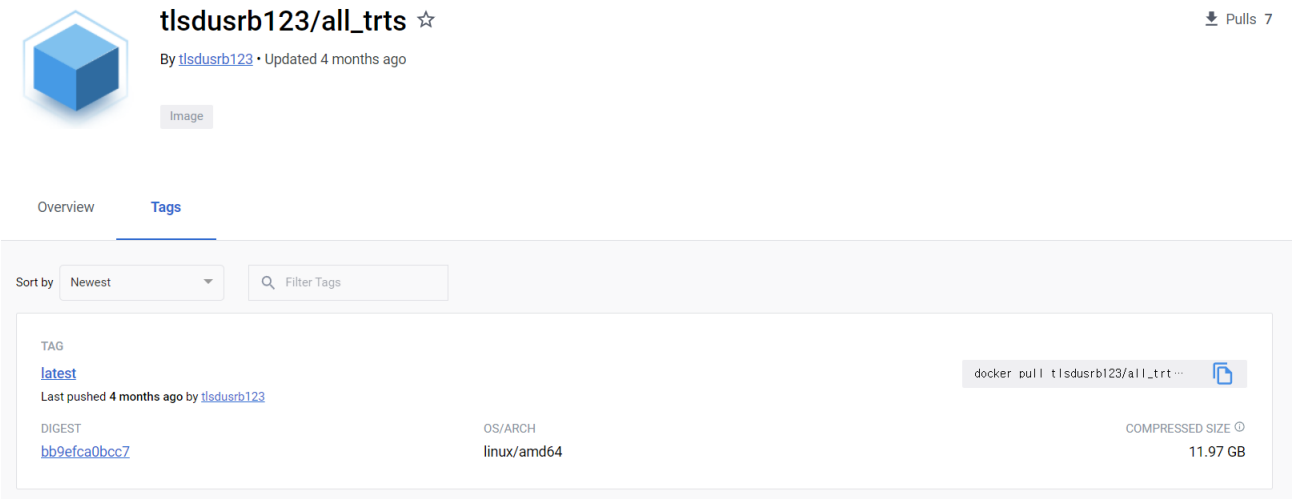
운영을 표준화하고 리소스 사용률을 높여 비용을 절감 하게 됨

Image Composition

Base Image : nvcr.io/nvidia/tensorrt:20.11-py3

Dependencies :

- Ubuntu 18.04
- NVIDIA CUDA 11.1.0
- NVIDIA cuDNN 8.0.4
- TensorRT 7.2.1
- OpenCV 3.4.2
- Python 3.6.9
- ROS-melodic



Vision – S/W

YOLOv4

객체 탐지를 위한 딥러닝 모델로 각 미션에 맞게 데이터 취득 및 전 처리 과정을 거쳐 Training을 진행 함

TensorRT

TensorRT는 학습된 딥러닝 모델을 최적화하여 NVIDIA GPU 상에서의 추론 속도를 수배~수십 배 까지 향상시켜 딥러닝 서비스를 개선하는데 도움을 줄 수 있는 모델

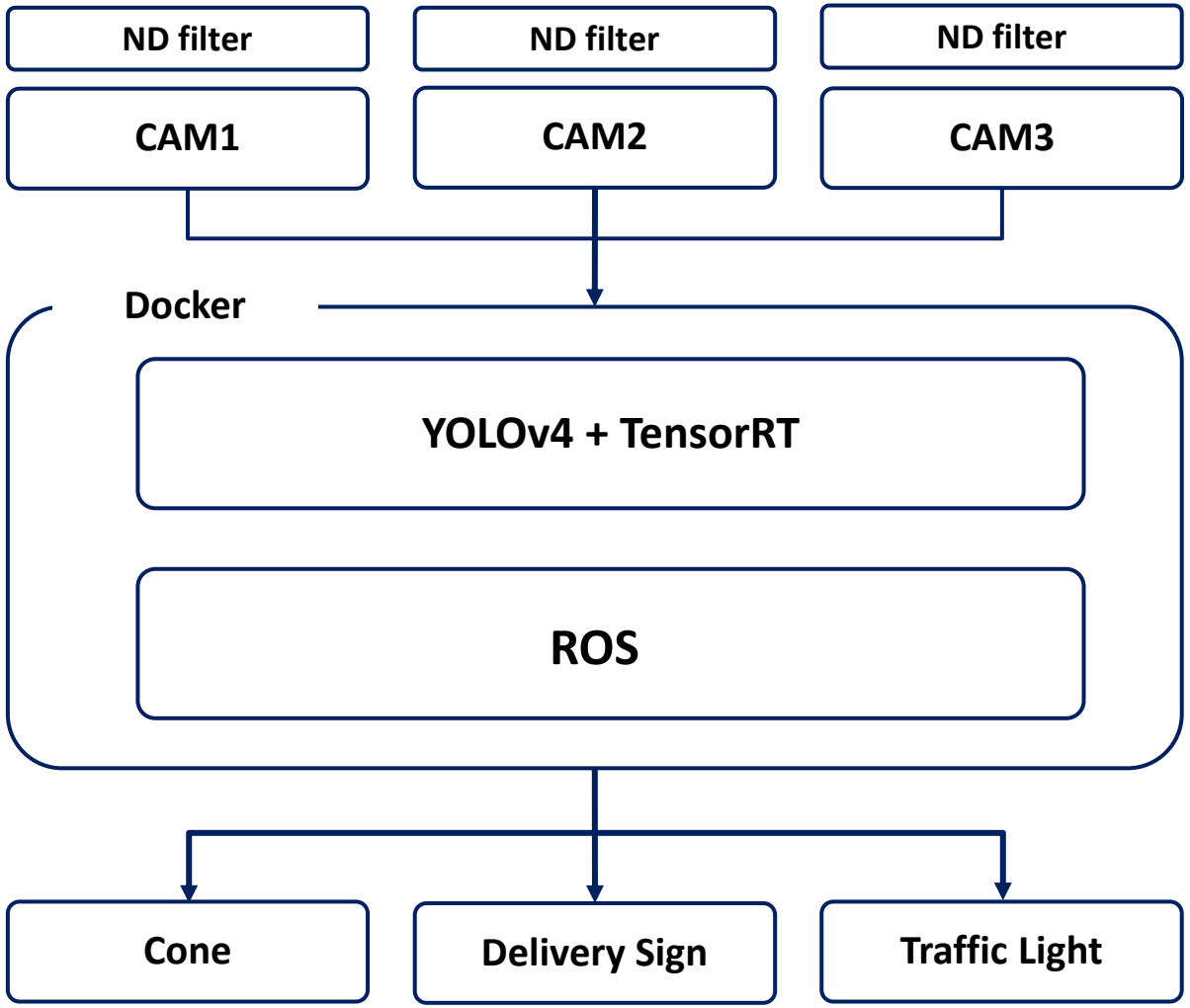
학습된 모델을 현재 사용하고 있는 GPU에 맞춰 가속화를 진행하여 inference 속도를 높여 최대 120FPS 까지 개선이 됨

ROS

ROS 통신을 사용하여 Docker container 내에서 출력되는 미션 별 객체 탐지 값을 판단팀의 local로 보내 줌

vision_msgs를 이용하여 YOLOv4 의 예측값 인 bounding box , conf, class정보를 박스 크기 내림차순으로 정렬하여 담아 보냄

Vision


ND filter
ND filter
ND filter
CAM1
CAM2
CAM3
Docker
YOLOv4 + TensorRT
ROS
Cone
Delivery Sign
Traffic Light

배달 표지판 데이터 구성

	Class	Training Set	Validation Set
출 처	-	시흥 취득	작년 팀
환 경	-	실내/실외	실외
BBox 수	A1	3298	764
	A2	3358	785
	A3	3377	762
	B1	3854	691
	B2	4202	770
	B3	3716	592
	SUM	21805	4364

콘 데이터 구성

	Class	Training Set	Validation Set
출 처	-	시흥 취득 +(웹, roboflow)	웹, roboflow
환 경	-	실내/실외	실외
BBox 수	Yellowcone	7132	1598
	Bluecone	3418+(3652)	1386
	SUM	14202	2984

신호등 데이터 구성

	Class	Training Set	Validation Set	Test Set
출 처	-	ETRI 제공*	K-city 취득	K-city 취득
환 경	-	대전, 세종, 서울 도로 주행	대회 환경	대회 환경
BBox 수	Red	7707	2784	2784
	Yellow	4249	985	985
	Green	5513	2062	2062
	Red+Left	3210	1091	1091
	Green+Left	2622	670	670
	SUM	23301	7592	7592

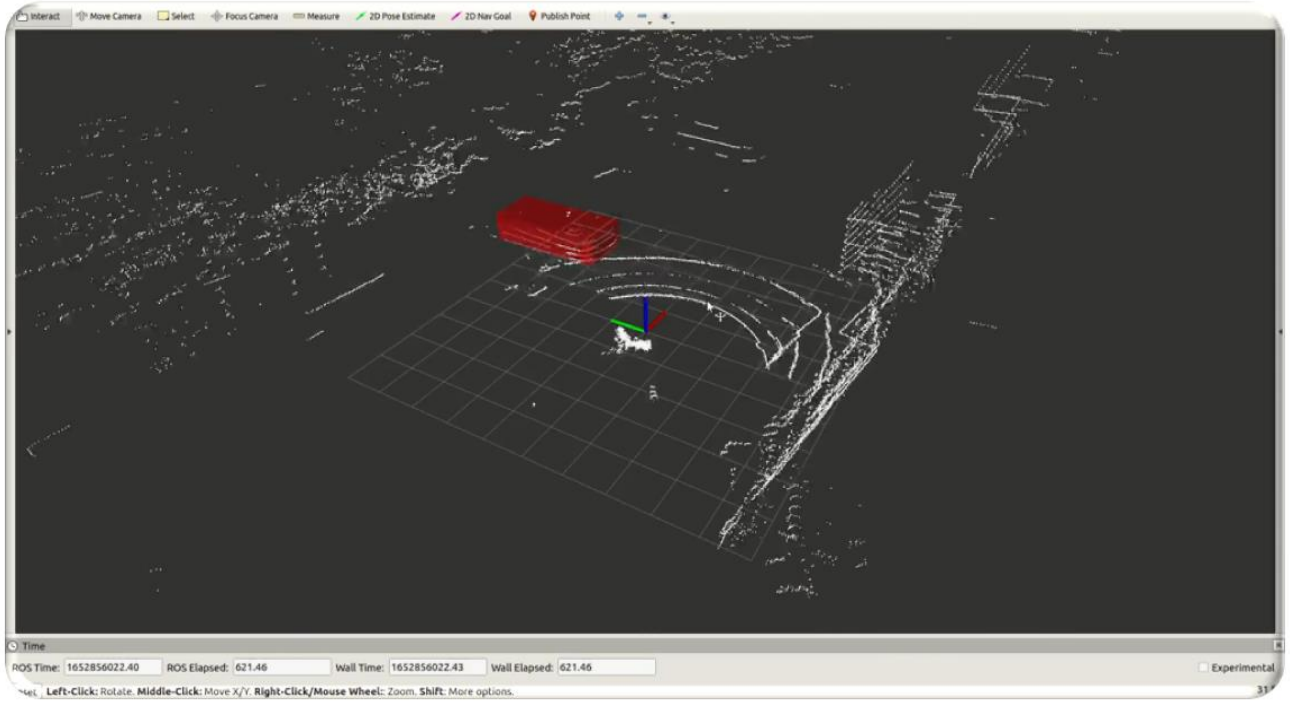
학습 결과 분석 (Test set)

Class	TP	FP	AP@0.50 [%]	IoU@0.50 [%]
Red	2757	27	99.69	-
Yellow	975	10	99.98	
Green	2009	53	99.12	
Red+Left	1084	7	99.81	
Green+Left	655	15	99.35	
SUM	7480	112	**99.59**	89.02

*제공자: 한국전자통신연구원 자율주행지능연구실, 본 연구는 과학기술정보통신부 및 정보통신기술진흥센터의 정보통신 · 방송 연구 개발사업의 일환으로 수행하였음. [2017-0-00068, 운전자 주행경험 모사기반 일반 도로환경의 자율주행4단계(SAE)를 지원하는 주행판단엔진 개발 / 2018-0-00327, 고정밀 맵 음영 환경의 완전자율주행 네비게이션 인공지능 기술개발]

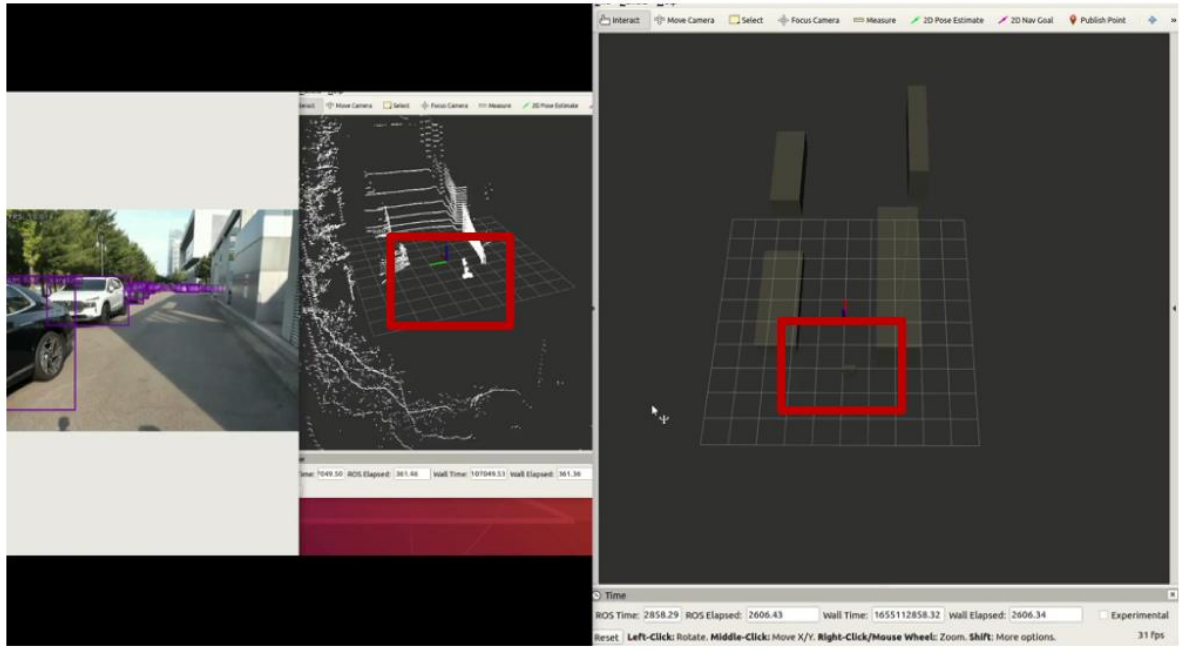
LiDAR

Preprocessing using PCL → Segmentation using RANSAC → DBSCAN Clustering → 3D Bounding Box

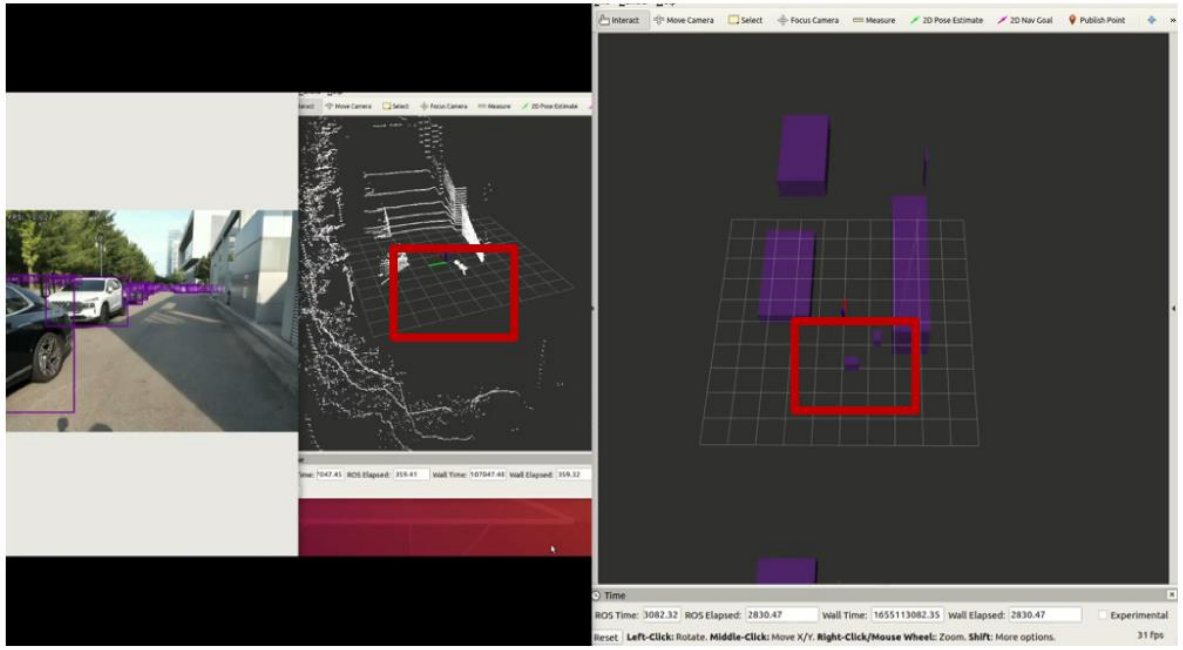


LiDAR 3D Object Detection

3D Obstacle Detection Results

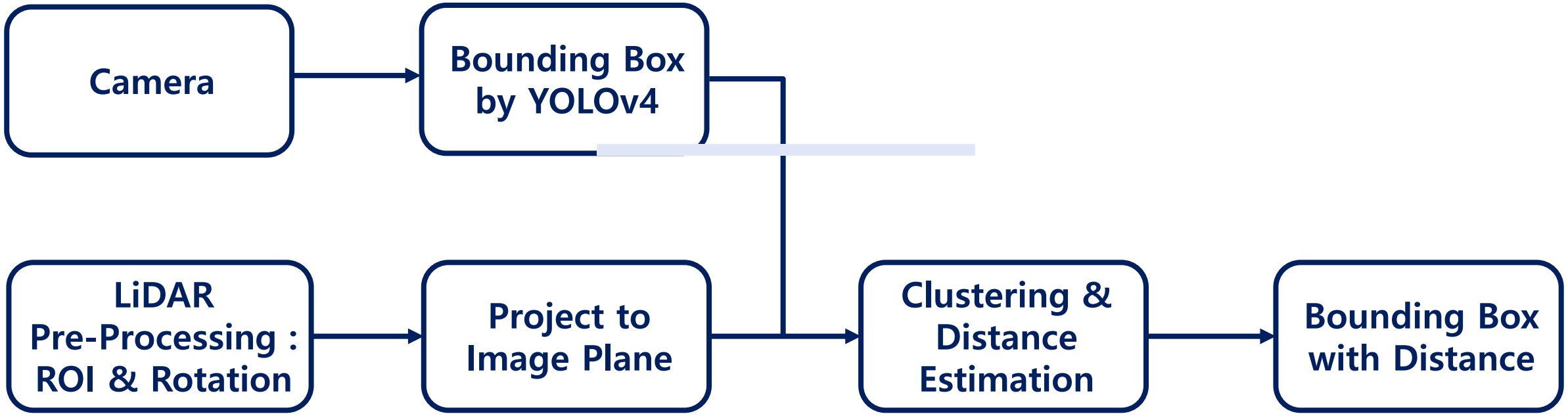


Detection 잘 되지 않은 경우



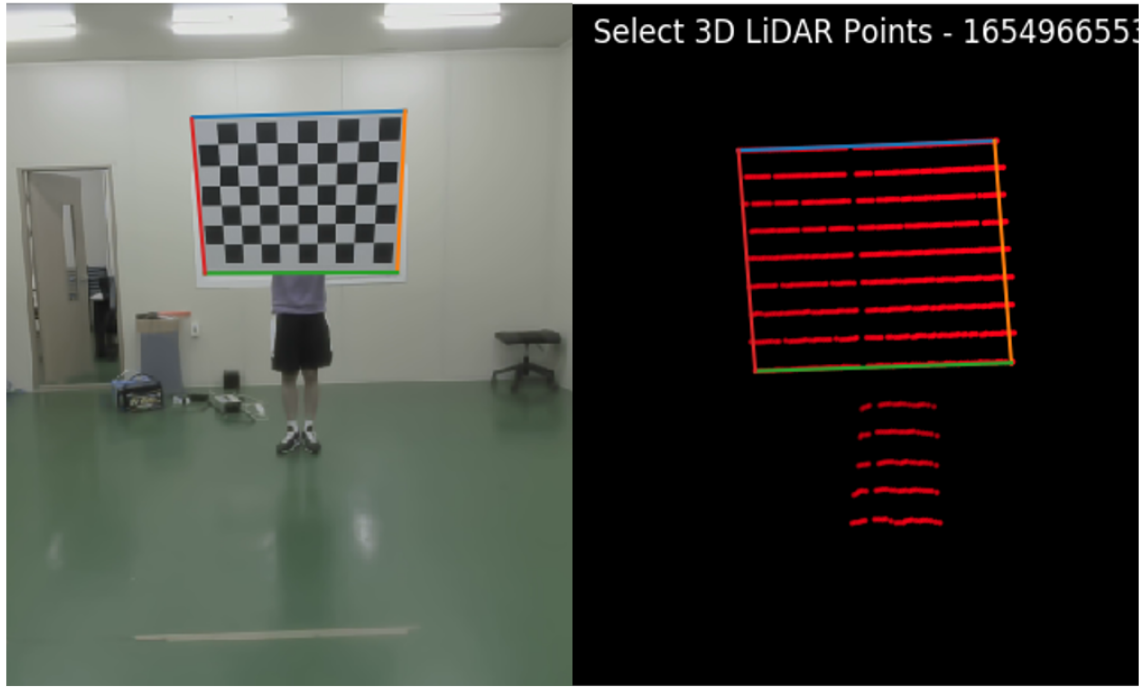
Detection 잘 된 경우

Camera & LiDAR Sensor Fusion



Camera & LiDAR Sensor Fusion

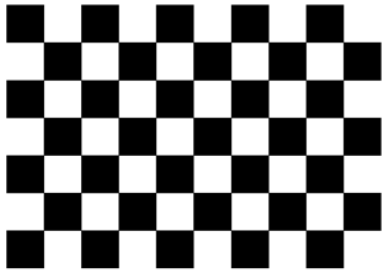
Sensor Fusion



Camera-LiDAR Calibration

Calibration Method

- Method : Checkerboard
- Checkerboard Size : A0
- Square Length : 115mm
- Pattern Size : 8x6
- Used Data Images : 50 Images



Calibration Result

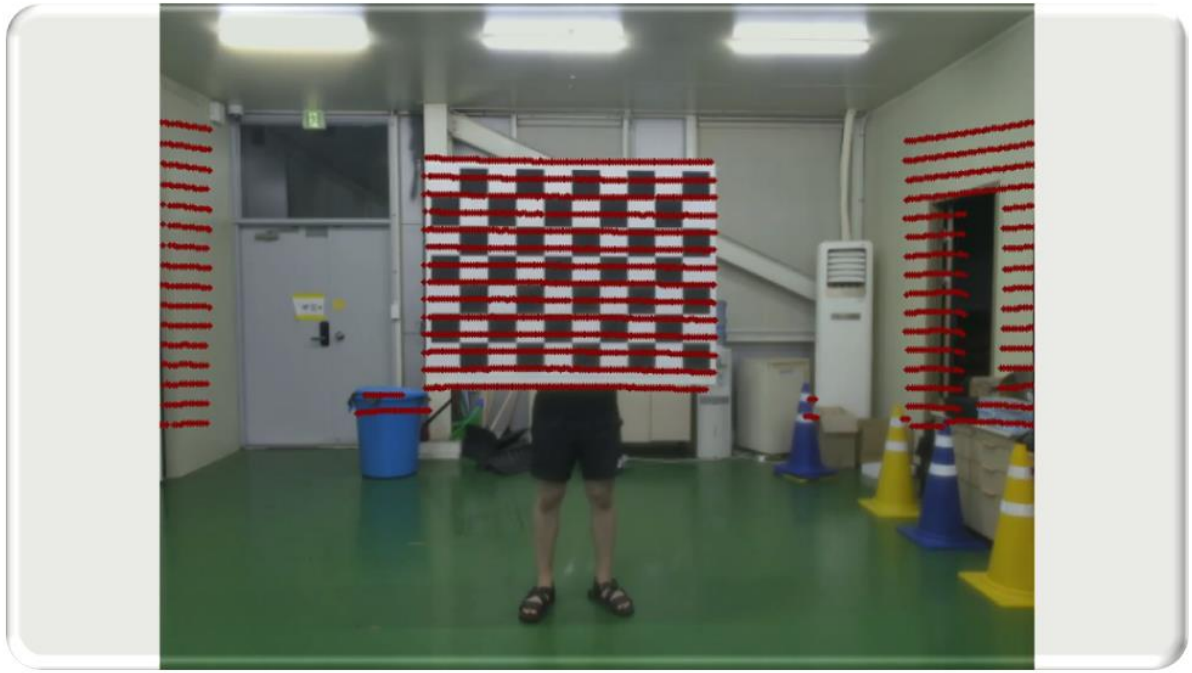
- Rotation Matrix

  : [[0.11322666 -0.99348988 -0.01255328],
  [-0.28690784 -0.02059697 -0.95773674],
  [0.9512432 0.11204297 -0.28737216]]
- Translation Offsets

  : [[-0.28996874 0.46188266 0.11583001]]

Camera & LiDAR Sensor Fusion

Sensor Fusion

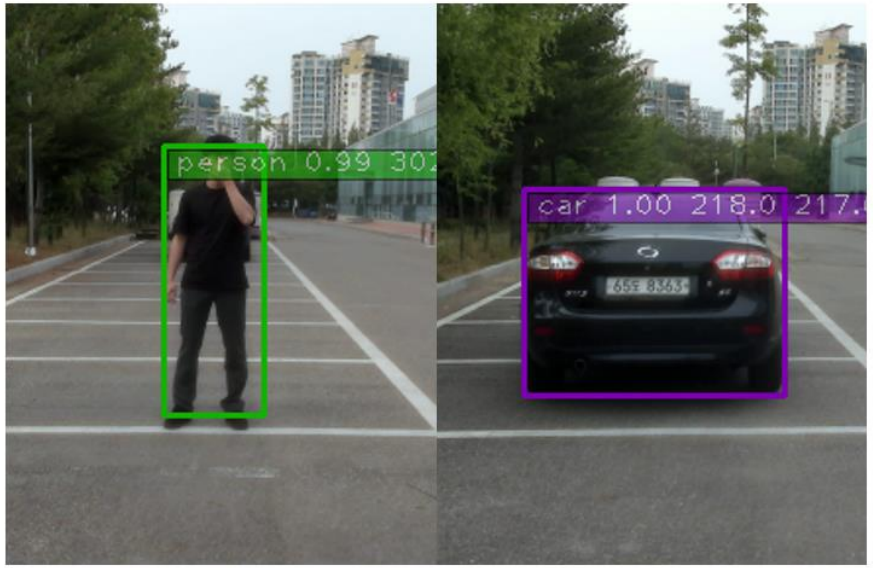


LiDAR Points Image Projection 결과

Obstacle Detection

- YOLOv4에서 탐지된 객체의 Bounding box 정보와 라이다 PointCloud 이용
- Bounding Box를 포인트 클라우드 filtering에 사용하여 정제된 PointCloud를 생성
- Filtering 된 PointCloud를 기반으로 군집화하여 객체를 검출하고 객체까지의 거리를 계산

Clustering Parameter Adjust using Bbox



사람, 차량 Bounding Box Size 측정

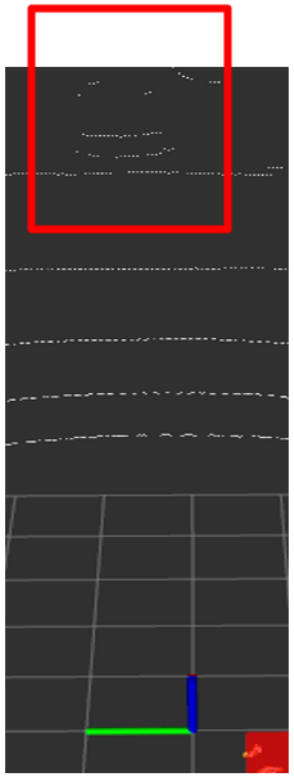
- 거리 별 Bounding Box Size(pixel, width x height)

Object	5m	10m	15m	20m
Person	51 x 137	37 x 72	21 x 52	14 x 41
Car	133 x 105	78 x 60	55 x 43	41 x 33

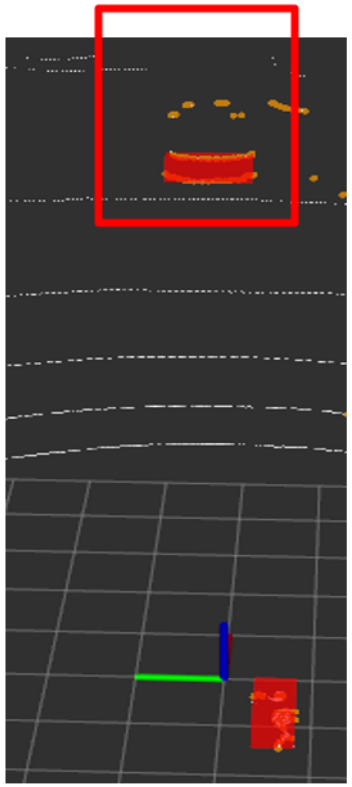
- Bounding Box Size 별 DBSCAN Radius 설정(m)

Pixel x Pixel	7000 ~	1100 ~ 2700	600 ~ 1100	0 ~ 600
Person	0.1	0.2	0.3	0.4
Pixel x Pixel	14000 ~	2400 ~ 4700	1400 ~ 2400	0 ~ 1400
Car	0.1	0.2	0.3	0.4

Clustering Parameter Adjust Result

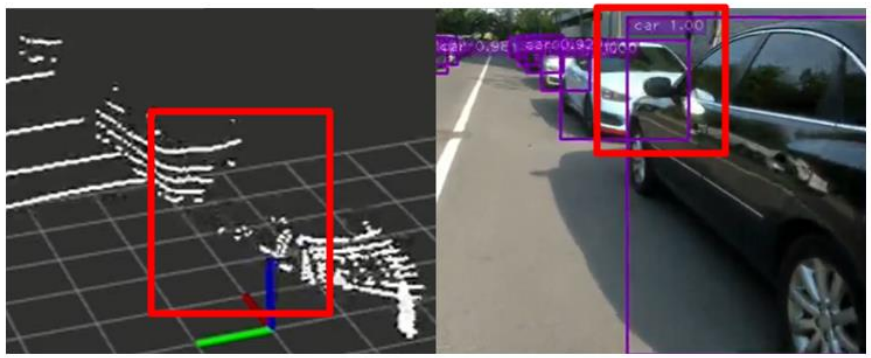


적용 전 20m 3D Detection



적용 후 20m 3D Detection

Bounding Box Overlap Problem



Cam, LiDAR Detection Result

Data Overlap

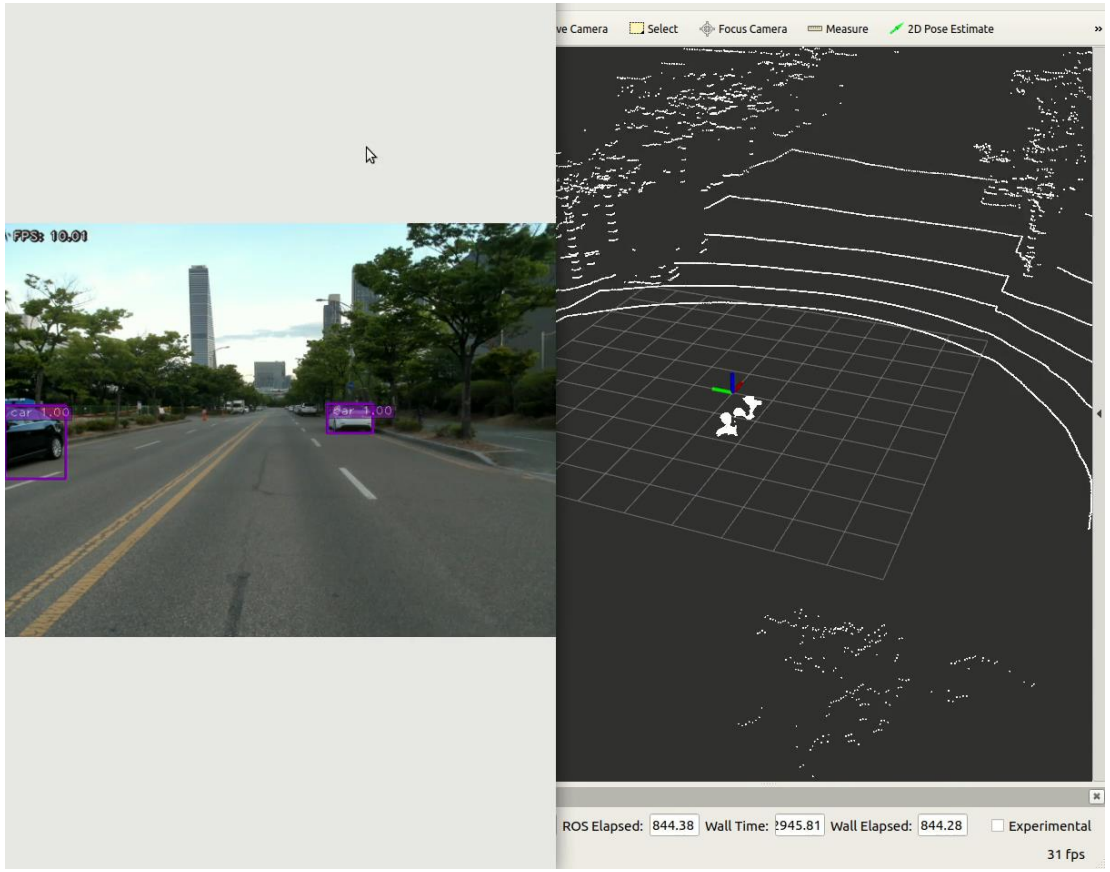
- 도로 주행 시 YOLO Detection 과정에서 Bounding Box 가 겹치는 현상 발생
- 단순히 Bounding Box 내 Point Cloud Data Clustering 시 앞 차량의 Data 가 Clustering이 될 수 있음

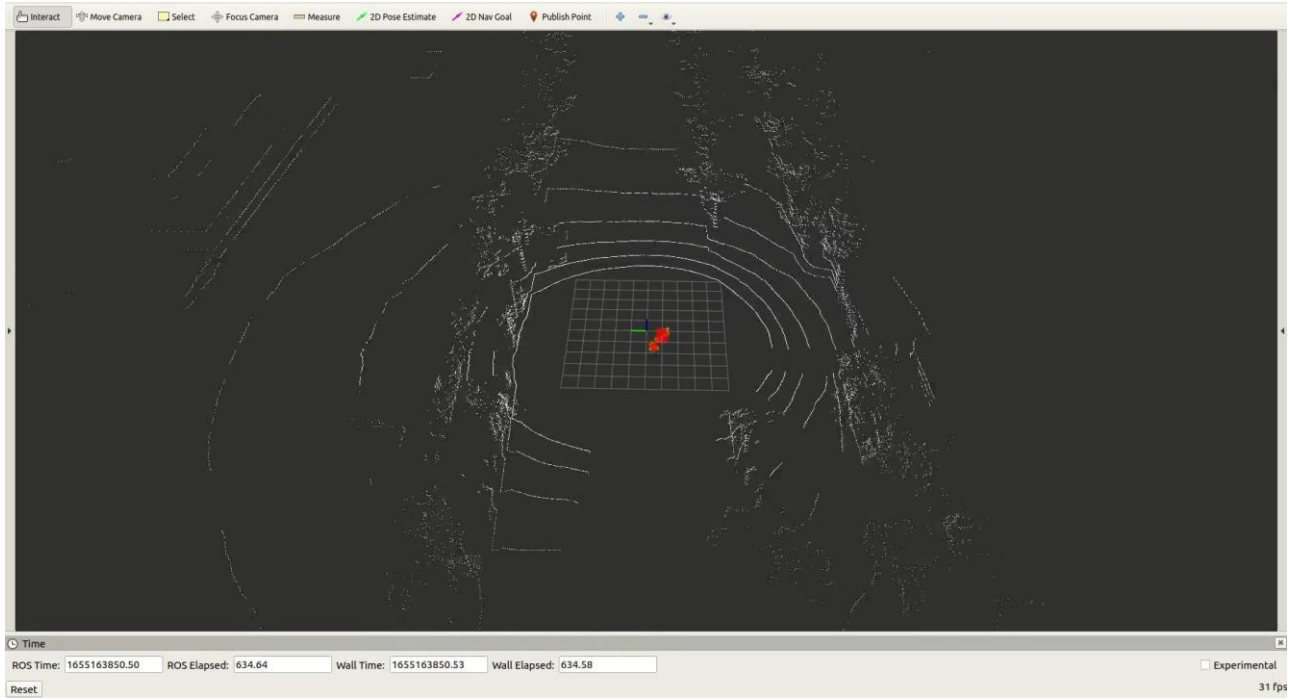
Sorting Result

Bounding Box Sorting

- Bounding Box Size 크기 순서로 정렬
- 크기가 큰 Bounding Box 내부 Point Cloud Data 부터 Clustering 후 Data 제거 진행
- Bounding Box가 겹칠 경우 후방 차량의 Data 만으로 Clustering 가능

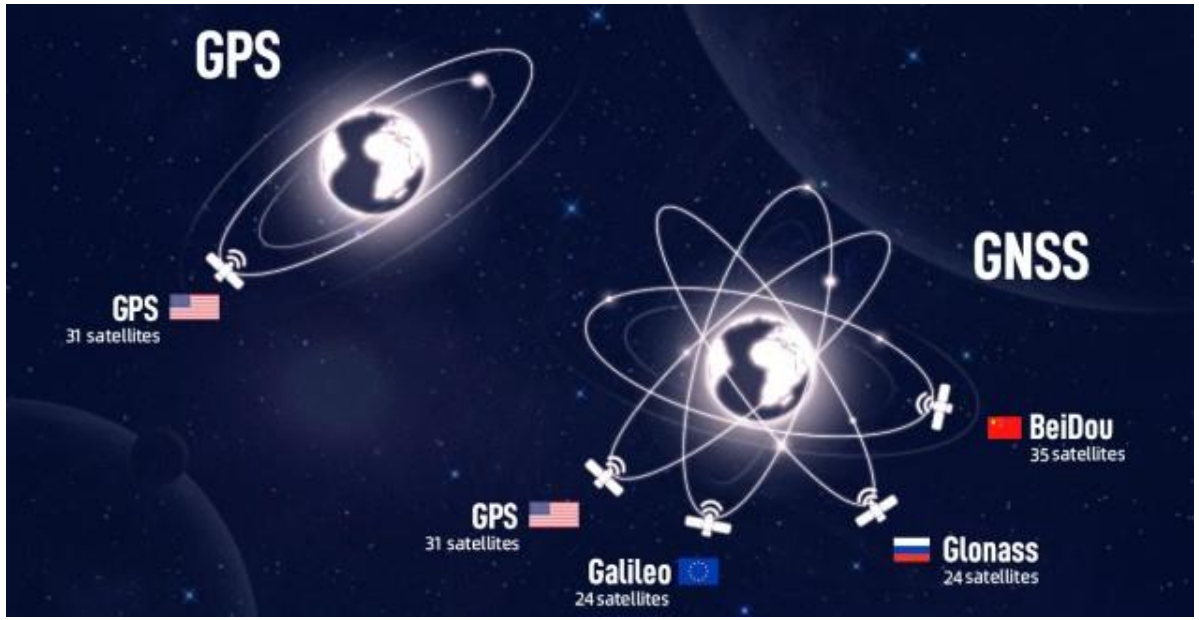
3D Object Detection Result



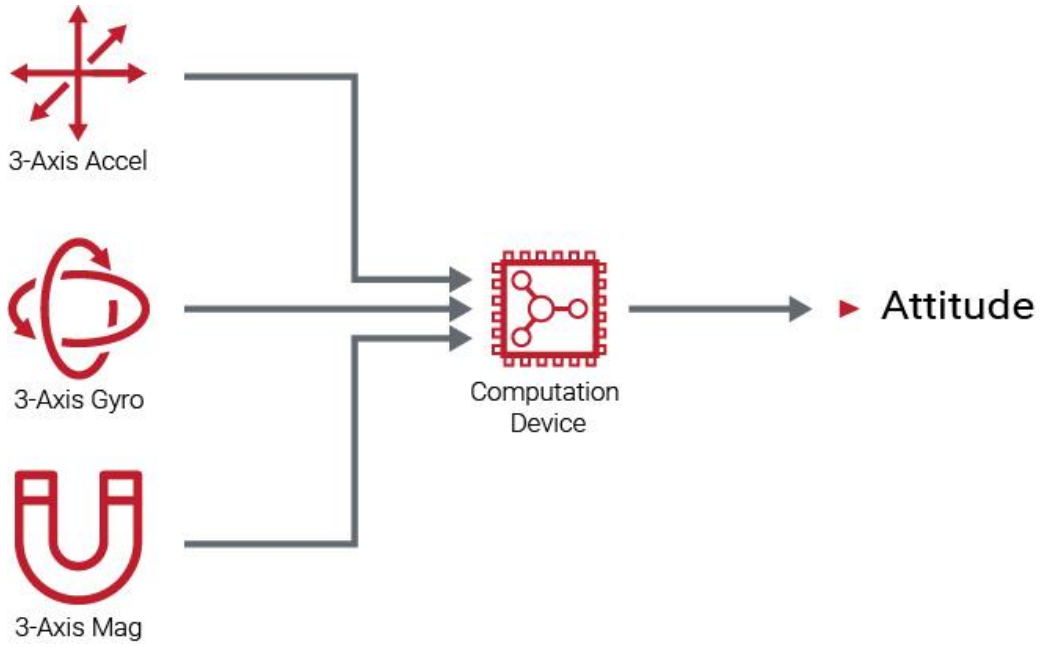


- **기술 구현의 이유 :** 측위 기반 자율주행을 위해서는 지도상의 Ego의 위치 파악이 선행되어야 하며 매우 중요함.
- **기술 구현의 목표 :** 최대 1cm의 오차만을 허용하는 차량의 위치와 Heading 값을 판단 / 제어부로 전달하는 것.

GNSS

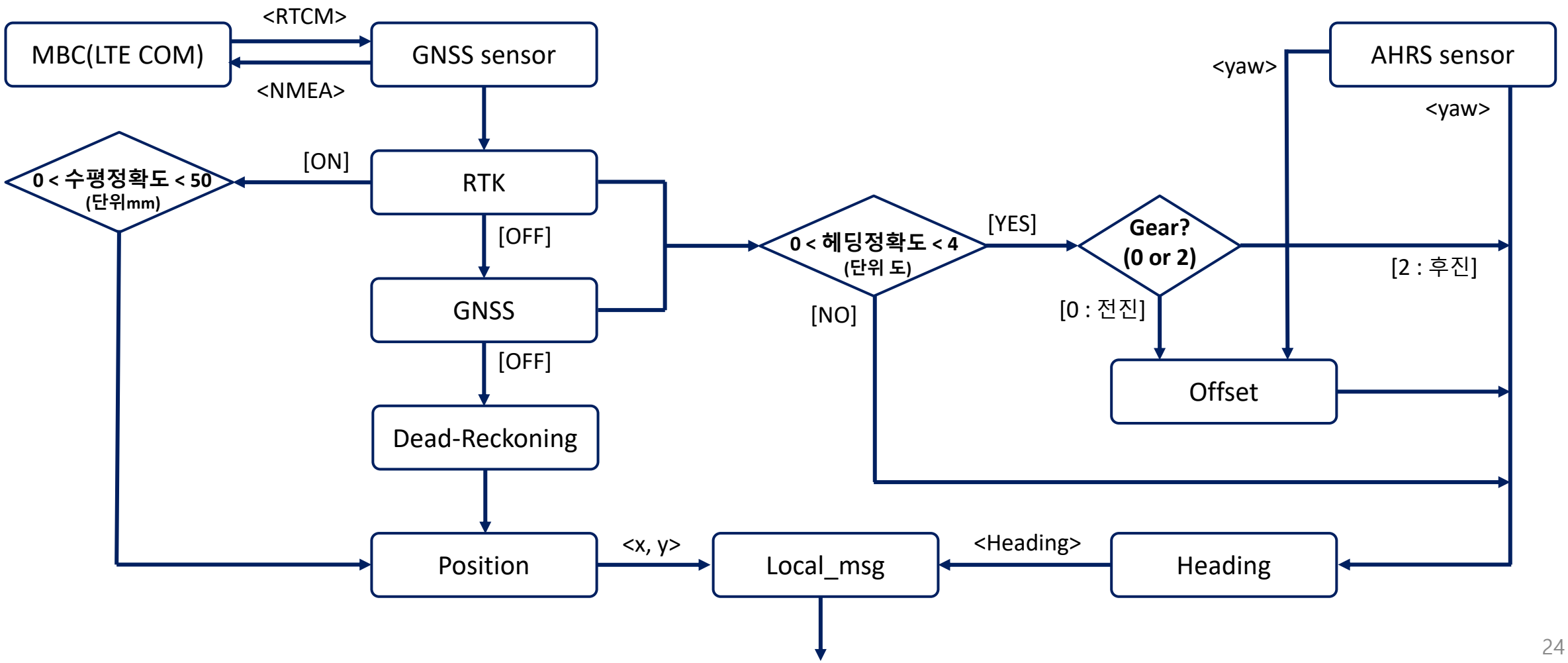


관성센서



Position 계산 순서도

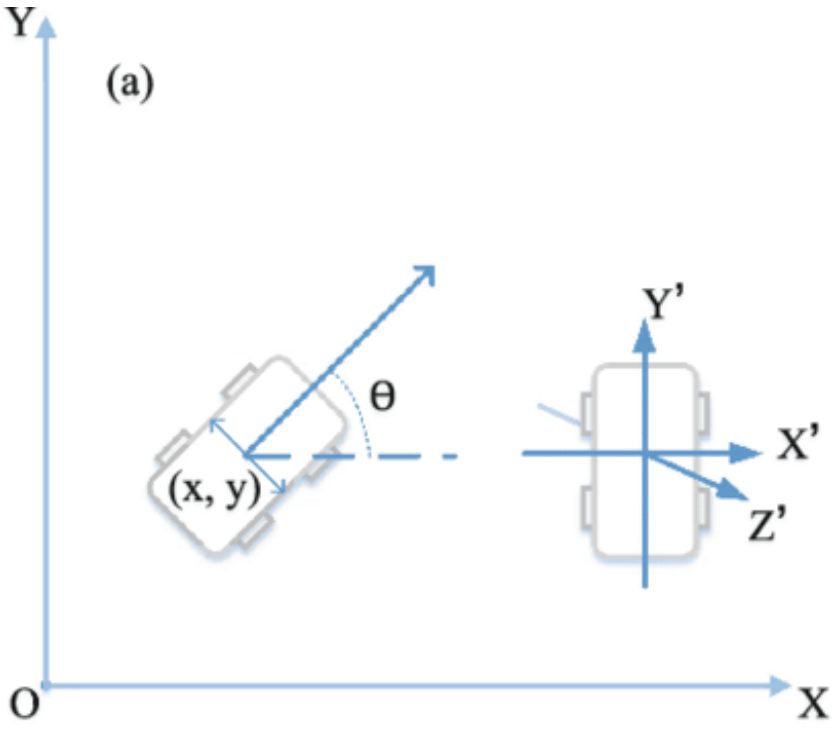
Heading 계산 순서도

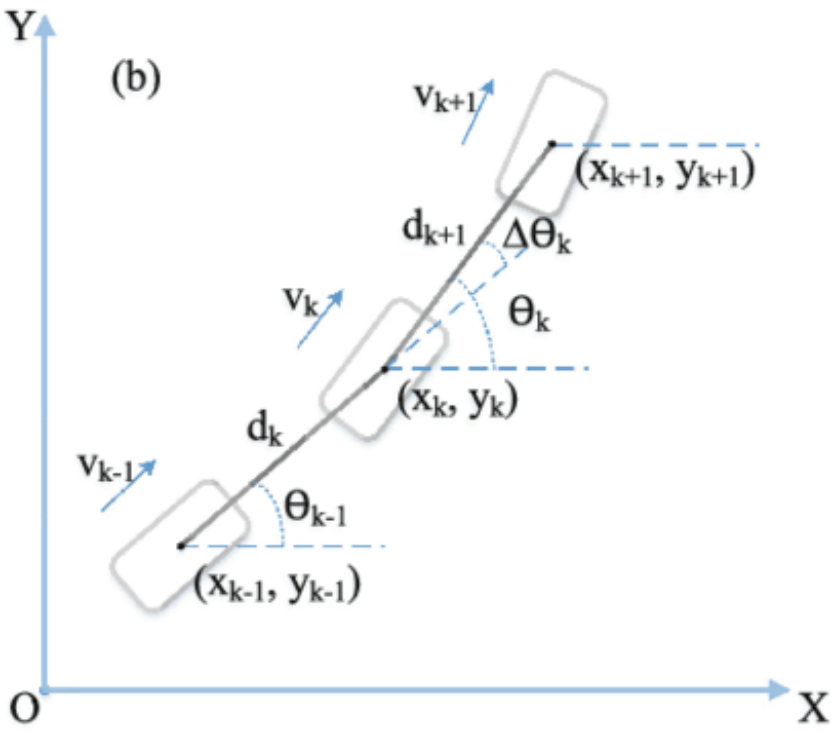


- **기술 구현의 이유 :** 자율주행 중, GNSS센서의 신호가 미약한 사각지대가 생기기 마련(ex 터널, 도심지)
- **기술 구현의 목표 :** 관성센서와 엔코더를 이용해서 GNSS센서를 이용한 것 만큼 정밀한 측위가 가능한 알고리즘 개발

AHRS

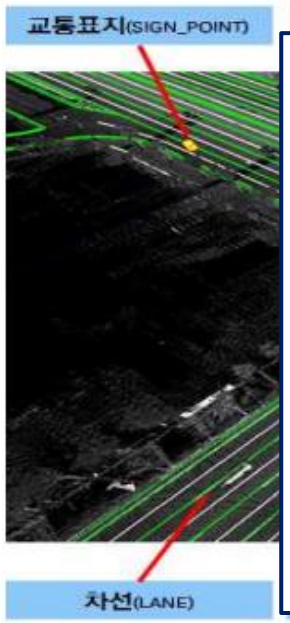
ENCODER



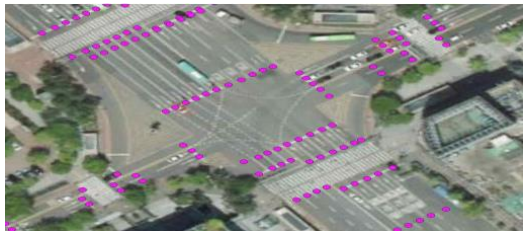


- **기술 구현의 이유 :** 기존의 GPS를 활용해 경로를 만드는 것보다 더 정확한 레퍼런스를 얻을 수 있어 경로 좌표 값에 대한 신뢰도 향상을 기대.
- **기술 구현의 목표 :** GIS의 shp파일을 json파일 형식으로 변환하여 주행에 필요한 경로 좌표 값을 얻는 것.

정밀지도란?



1. 주행경로노드(A1_Node)

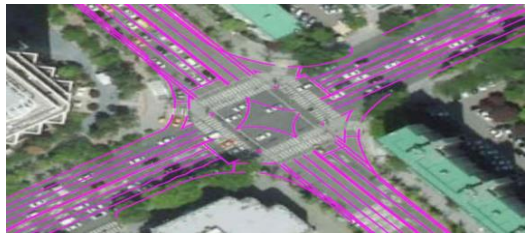


교통의 흐름 변환점

2. 주행경로링크(A2_Link)

주행 경로

7. 노면선표시(B2_Surfaceline Mark)

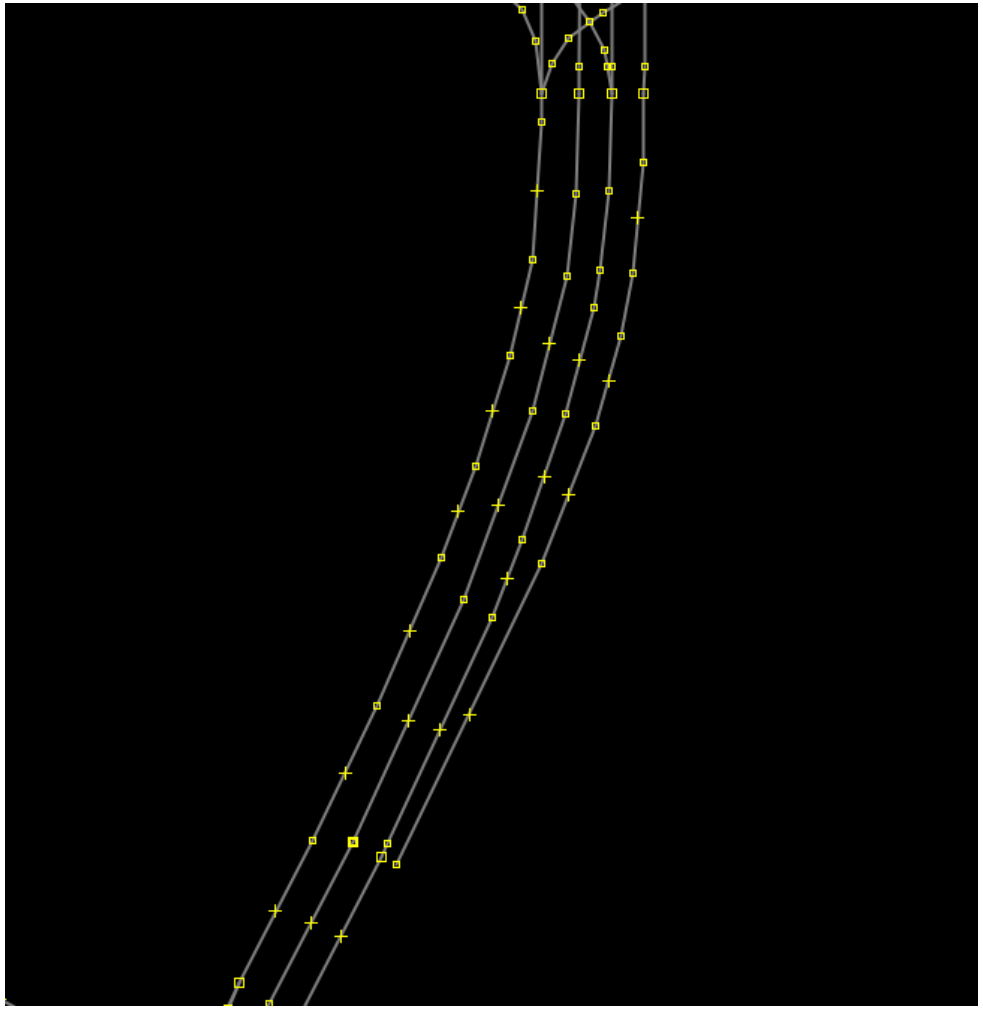


차선 및 차선변경

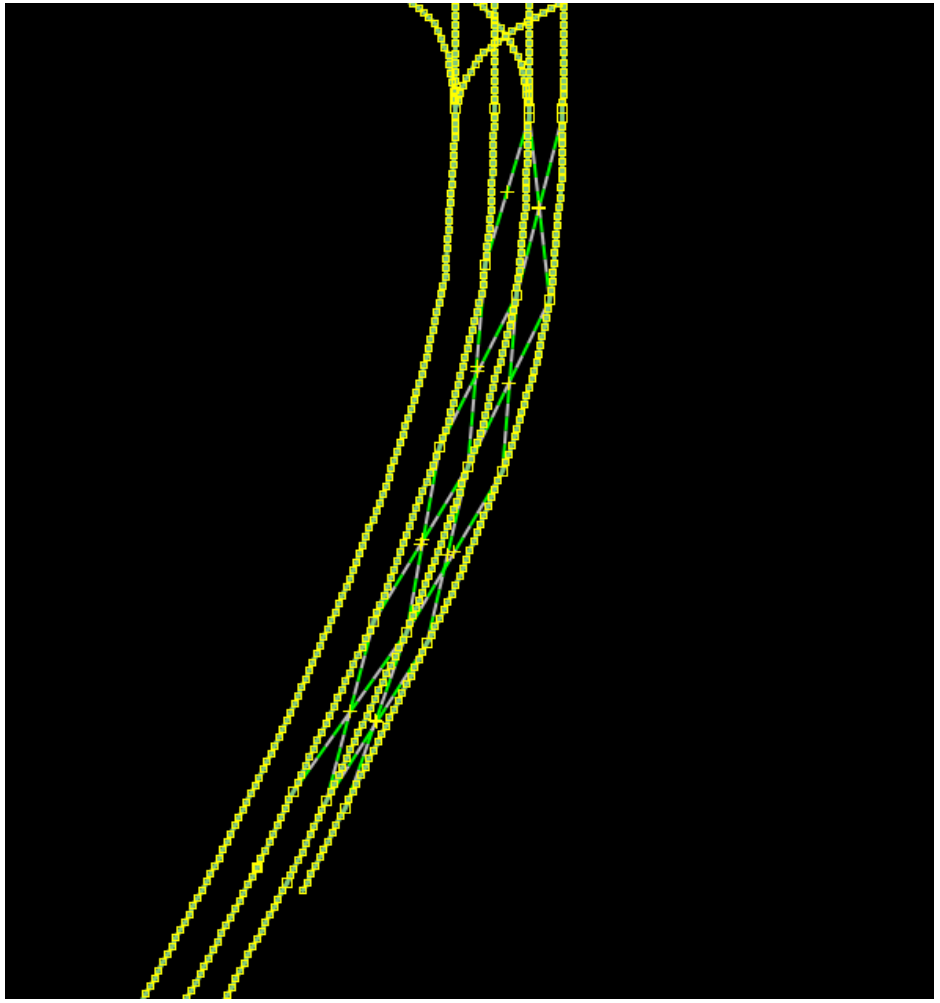
3_Surfacemark)
_Traffic Light)
트(C2_Kilopost)
안전시설
rotection Safety)
턱(C4_Speedbump)
물(C5_HeightBarrier)
Postpoint)

출처 : 국토 정보 플랫폼 (map.ngii.go.kr)

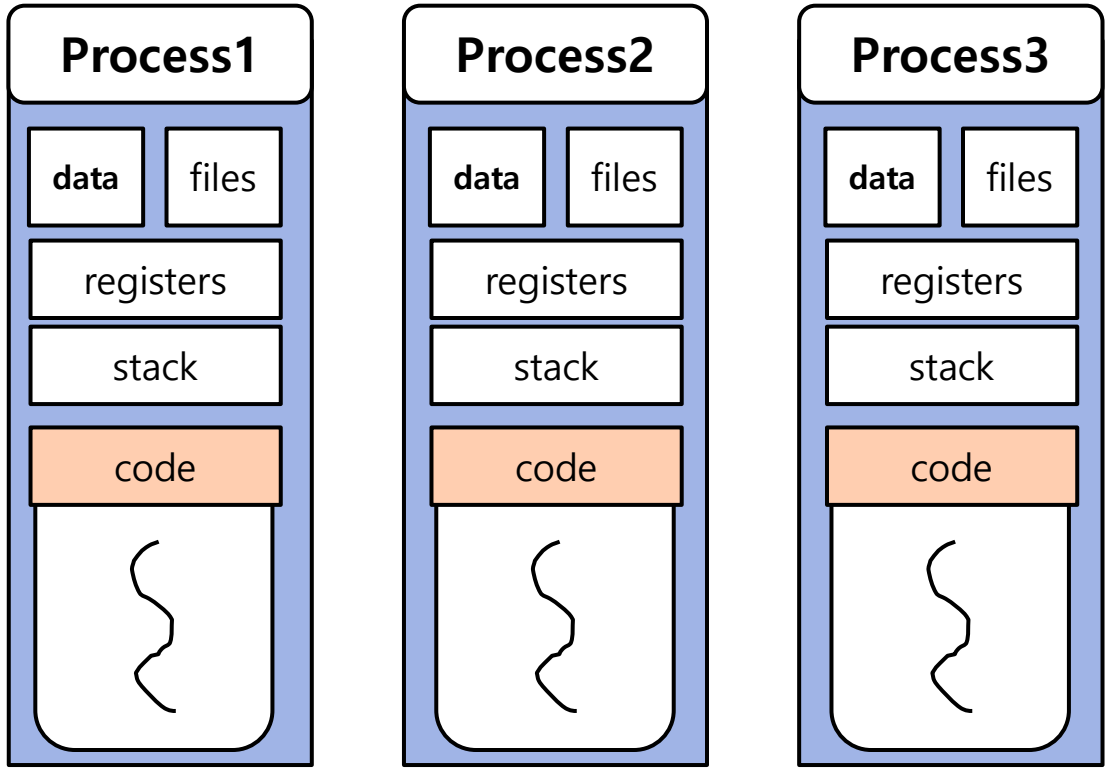
Localization



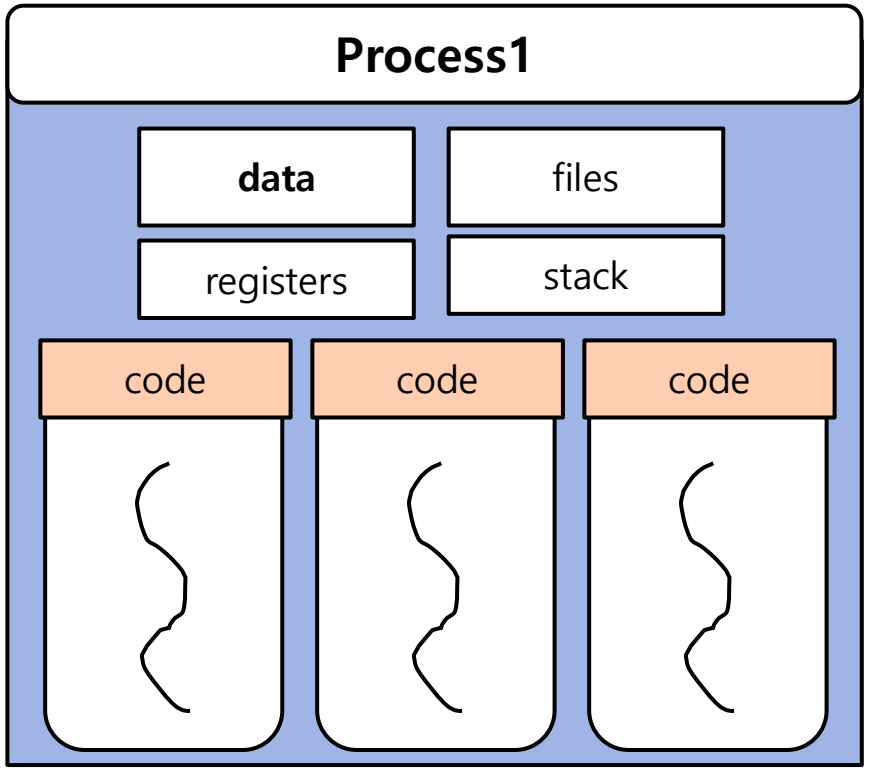
정제 전



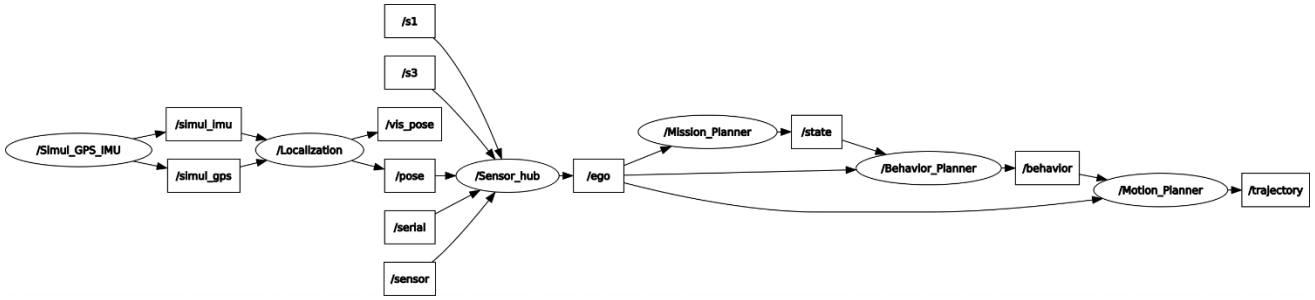
정제 후



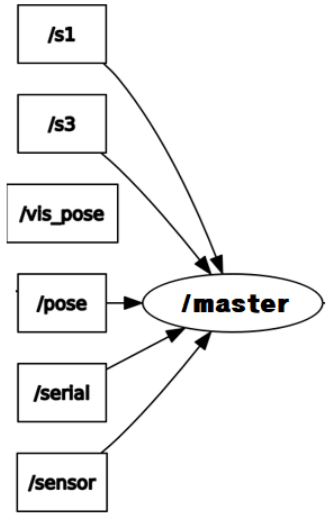
< Multi Process >



< Multi Thread >



< Multi Process>



< Multi Thread >

❖ Multi Thread 기반의 시스템 구성

- 장점
  - 처리 시간 감소
  - CPU, 메모리 자원 최소화
- 단점
  - topic msg 확인 불가로 인한 디버깅의 어려움
  - 두 개 이상의 스레드의 공유변수 접근 문제
  - GIL 기반의 파이썬에서 효율 감소
  - 자식 스레드 오류 시 부모 스레드에도 영향

Mission Planner

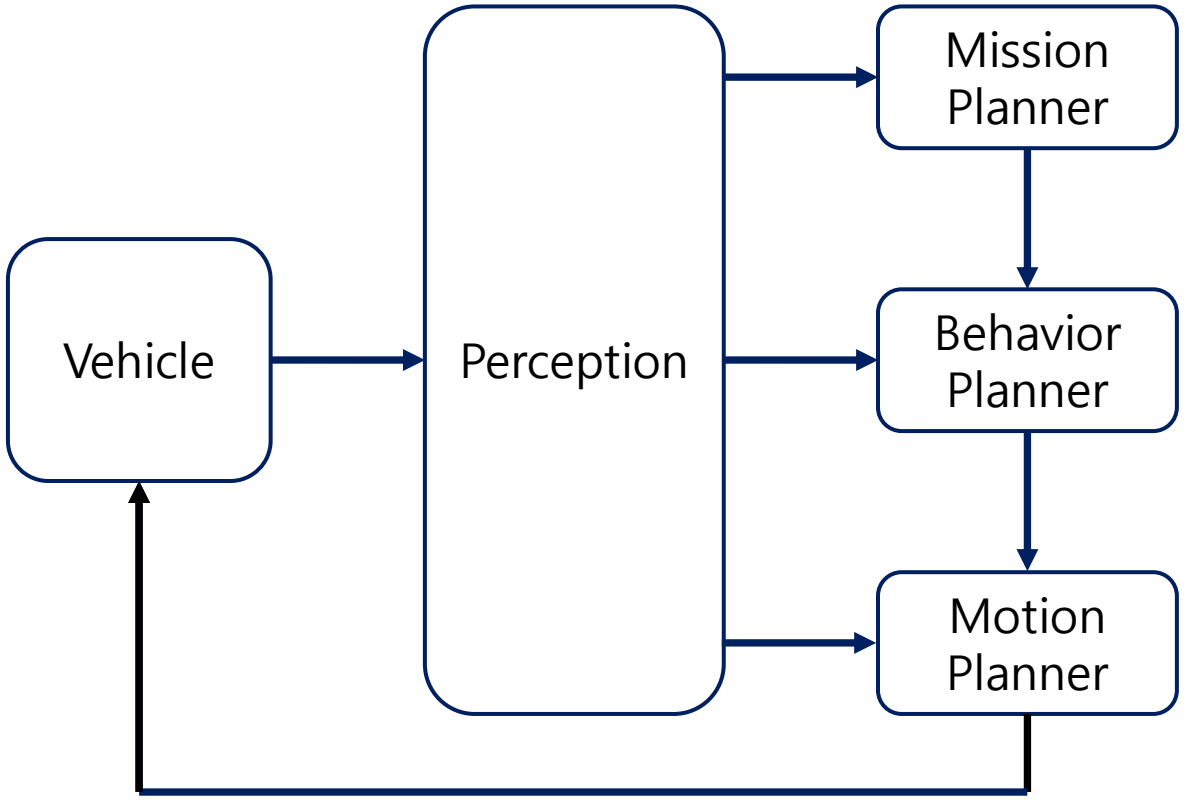
- 인지 정보를 바탕으로 차량의 state 생성
- 수행 예정 mission과 state를 Behavior Planner에 전달

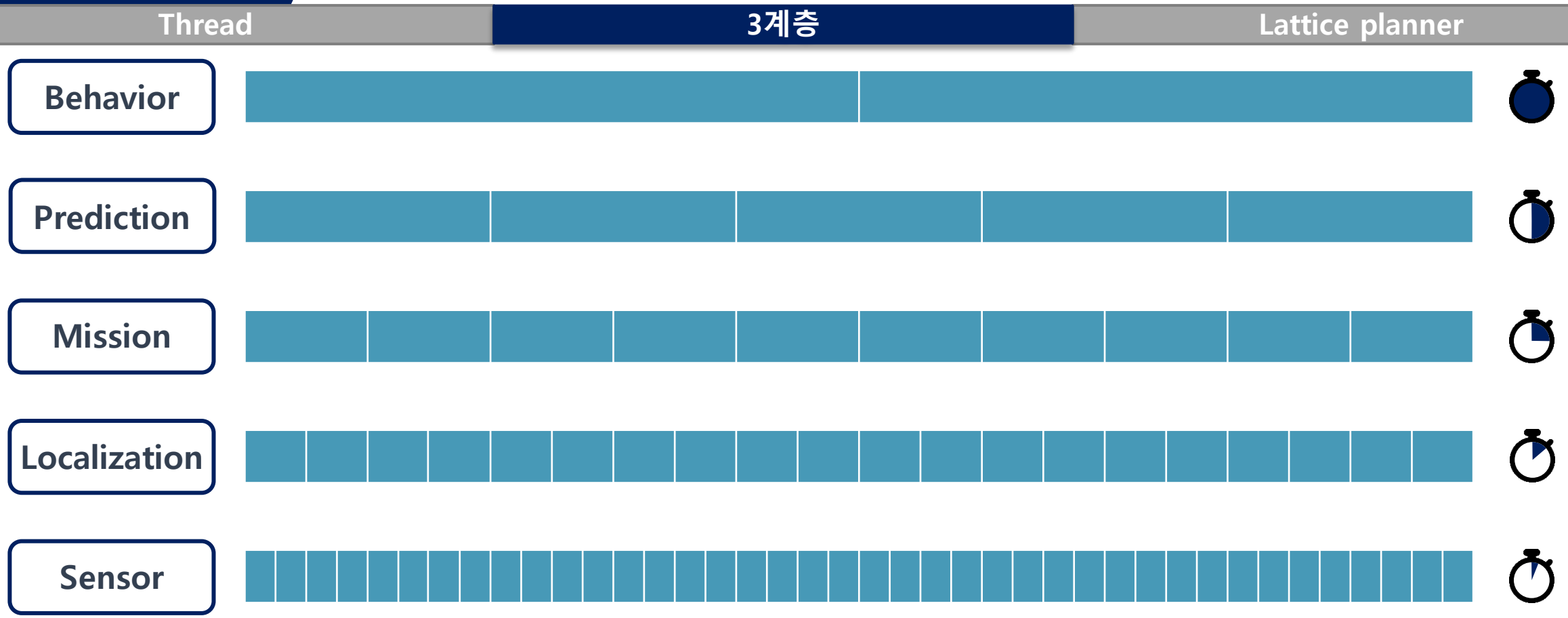
Behavior Planner

- 수행할 mission과 인지 정보를 통해 실제 차량 Model 설정
- 설정한 Behavior Motion Planner로 전달

Motion Planner

- Behavior를 토대로 실제 Path Planning 진행
- Local Path 생성 후 Controller로 전달
- Lattice Planner 기반 Local Path Planning 진행

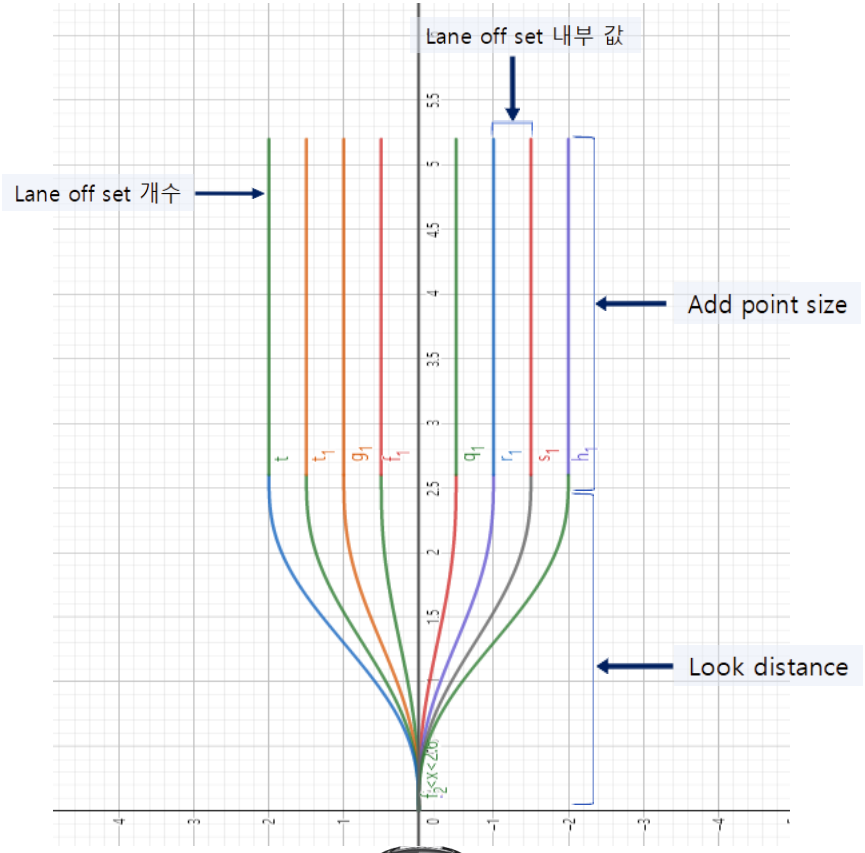




Scheduling Compute Time

- Sensor 및 Localization 모듈이 가장 높은 빈도로 업데이트 되고 사용하기 쉬우며, Mission, Prediction, Behavior 순으로 빈도가 높음
- Behavior Module 에서 내린 결정이 더 오랜 시간을 소비하고 자주 변경되지 않기 때문에 가장 낮은 빈도로 업데이트

Thread | 3계층 | Lattice planner



Lattice Planner

- Global Path 기반의 Local Path Planner
- 현재 차량의 위치로 부터 대칭의 경로 생성
- 일정 거리 까지 3차 또는 5차 함수의 2차원 경로 생성
- 일정 거리 이후 Global Path 와 같은 경로로 이동
- 원하는 개수 만큼의 경로 생성 가능
- Cost Function 을 통해 이동할 경로 설정

논문을 기반으로 input, output 설정

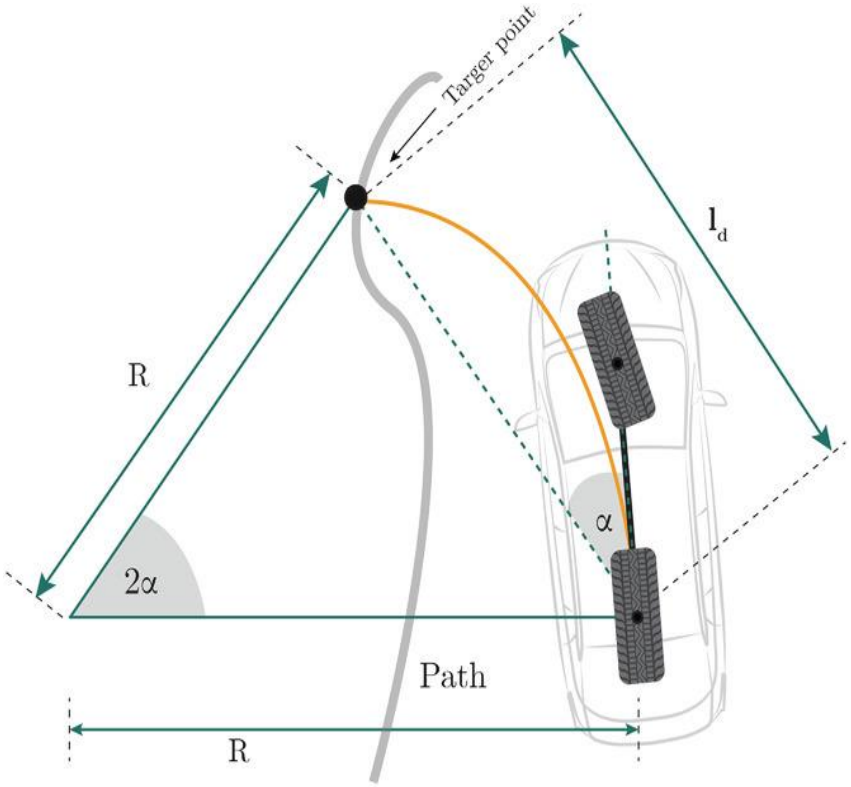
Input parameters

- Look distance
- Lane off set 내부 값
- Add point size
- Lane off set 개수

Output

- collision status
- Appropriate point까지 시간 or distance

Pure Pursuit



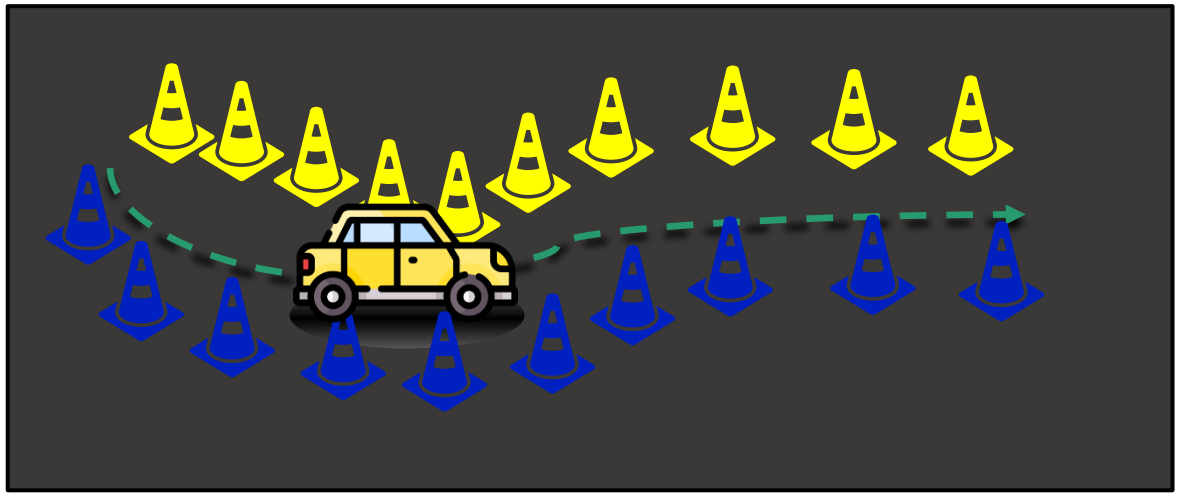
$$steering\ angle\ \delta = \tan^{-1}\left(\frac{2l\sin\alpha}{l_d}\right)$$

- l : *wheel base distance*
- l_d : *look ahead distance*
- α : *angle between heading and look ahead line*
- *Radius of the arc* $R = \frac{l_d}{2\sin\alpha}$

3. 대회 미션

- Track Mission
- Autonomous Parking System
- Intersection Driving
- Autonomous Emergency Brake
- Pick Up & Delivery

Track Mission ?



노란 콘과 파란 콘으로 구성 된 협로를 주행하는 미션

전체 과정

Sensor Fusion

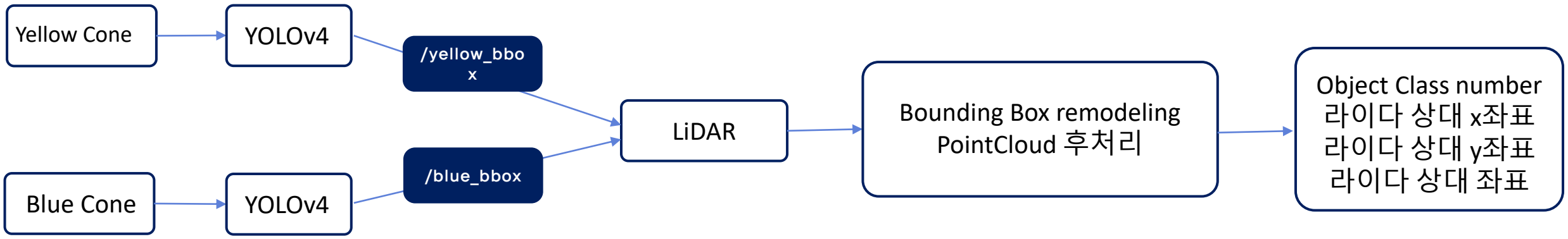
Camera + LiDAR

→

Planning & Control

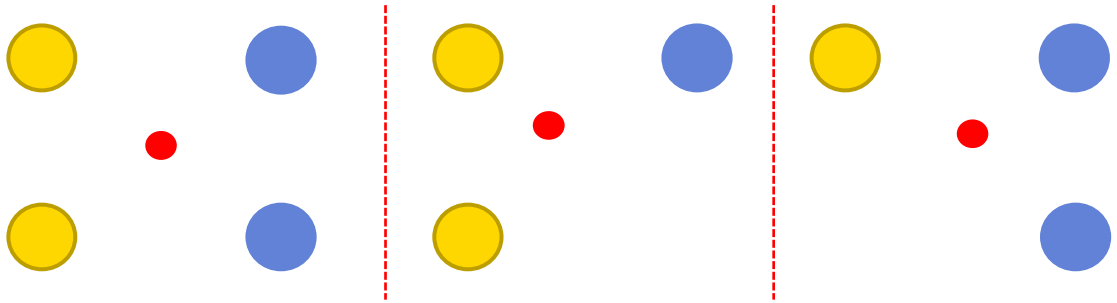
Target Point + Control

Perception

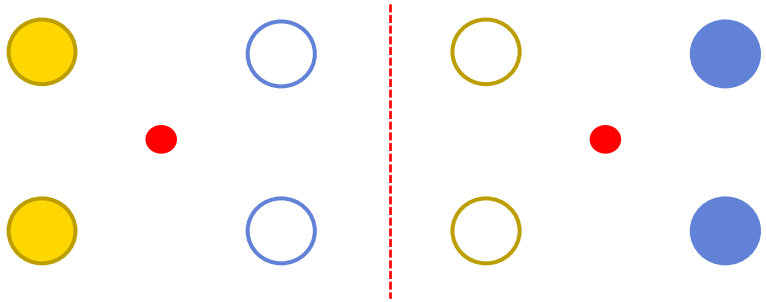

Yellow Cone
YOLOv4
/yellow_bbox
Blue Cone
YOLOv4
/blue_bbox
LiDAR
Bounding Box remodeling
PointCloud 후처리
Object Class number
라이다 상대 x좌표
라이다 상대 y좌표
라이다 상대 좌표

Planning

무게중심 알고리즘



월 팔로잉 알고리즘

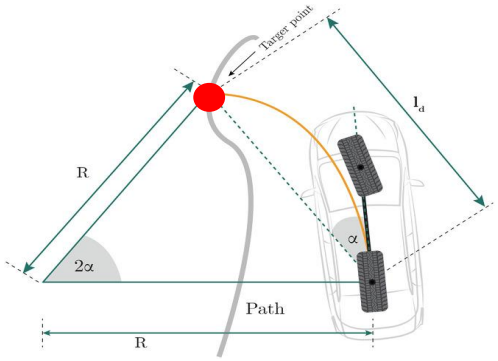


Control

종방향 제어

steer\speed	speed>10	10=>speed>7	7=>speed
steer>15	Speed:7 Brake:20	Speed:7 Brake:10	Speed:7
steer<=15	Speed:10 Brake:20	Speed:10 Brake:10	Speed:13

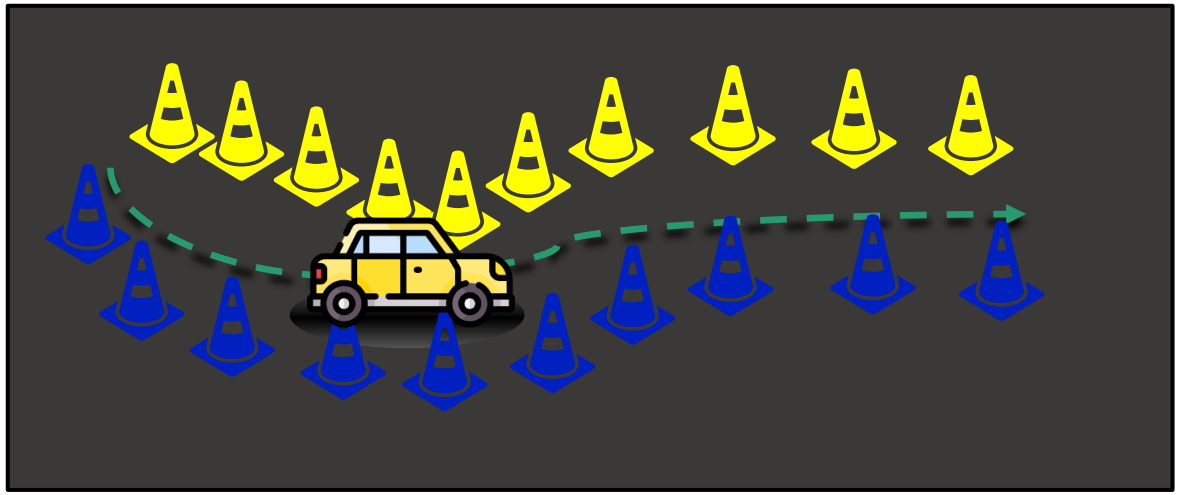
횡방향 제어

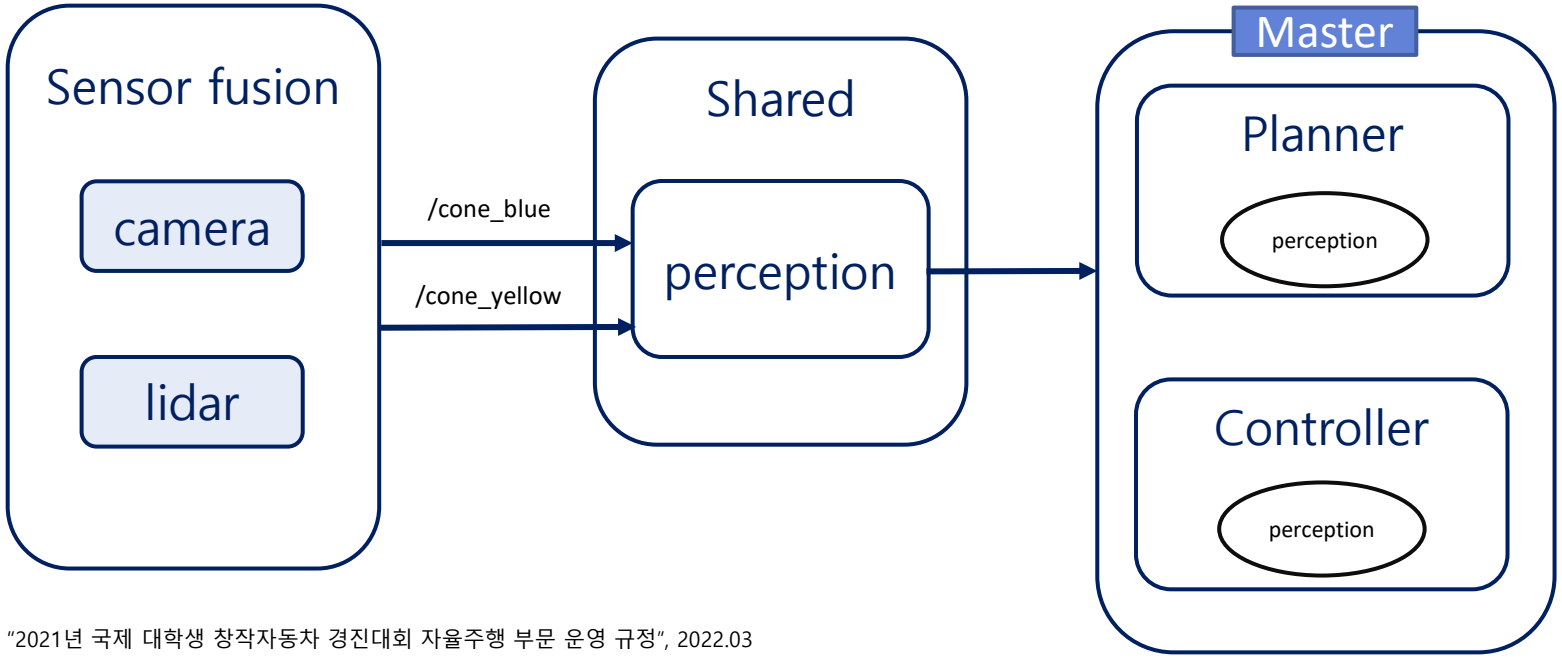


$$steering\ angle\ \delta = \tan^{-1}\left(\frac{2l\sin\alpha}{l_d}\right)$$

- l : *wheel base distance*
- l_d : *look ahead distance*
- α : *angle between heading and look ahead line*
- *Radius of the arc* $R = \frac{l_d}{2\sin\alpha}$

Track Mission



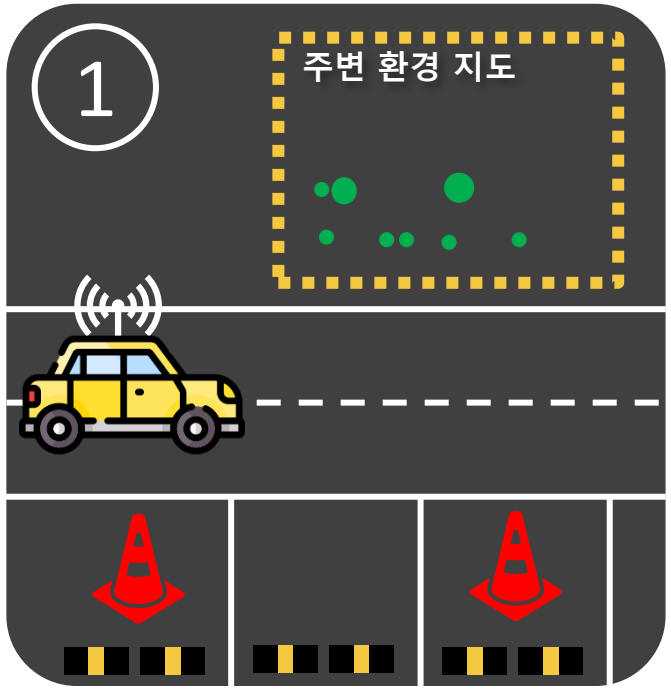


Pickup Car : flaticon.com.
국제 대학생 창작자동차 경진대회 조직 위원회, "2021년 국제 대학생 창작자동차 경진대회 자율주행 부문 운영 규정", 2022.03

Track Mission

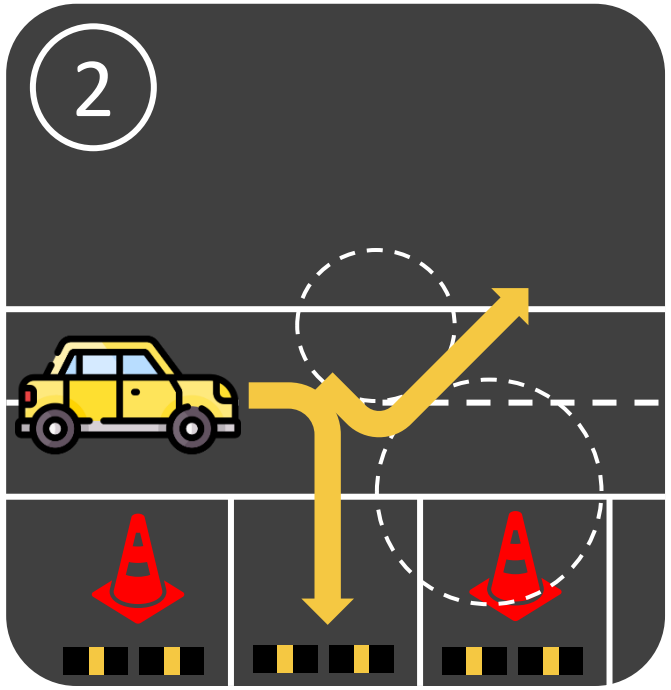
Autonomous Parking System

차량 주변 환경 인식



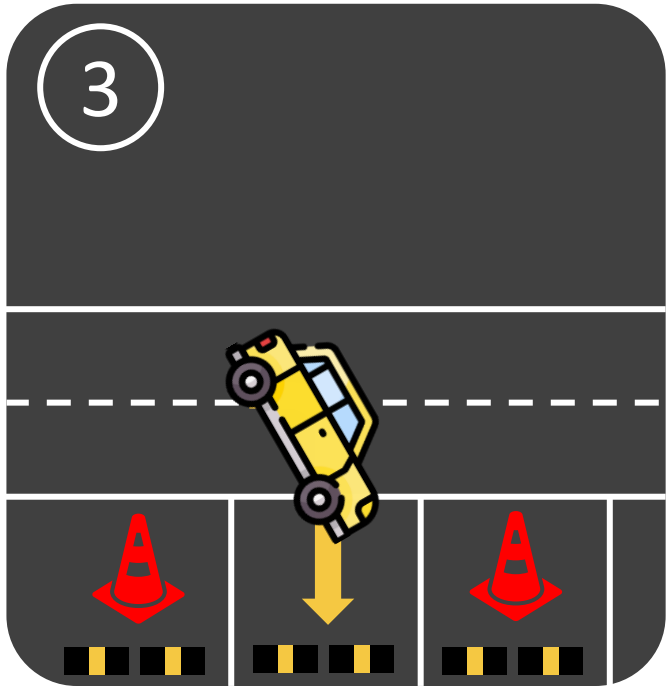
라이다 센서를 이용하여
주차 가능 공간 탐색

주차 경로 생성

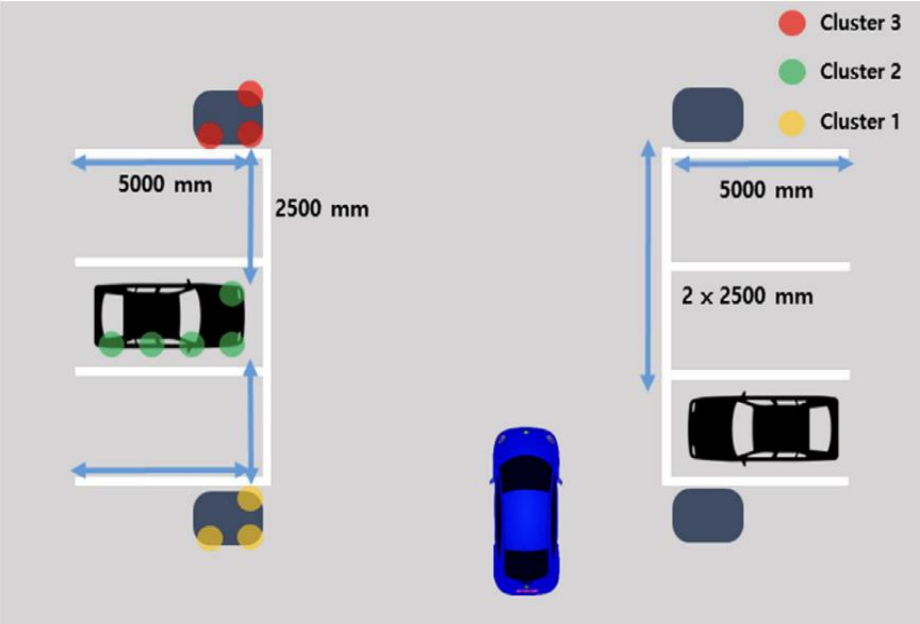


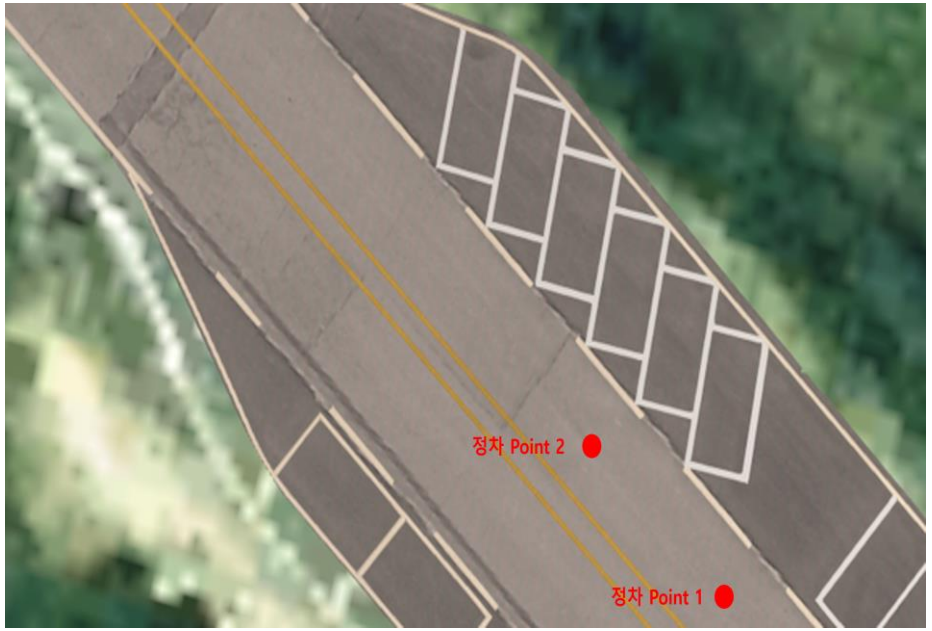
주차 가능 공간 주변 가변
원을 바탕으로 경로 생성

경로 추종



기하학적 방법을 이용하여
생성된 경로 추종





LiDAR 를 이용한 주차 공간 탐색

주차 공간에 대한 정보는 Infra에게서 전달 받는다고 가정

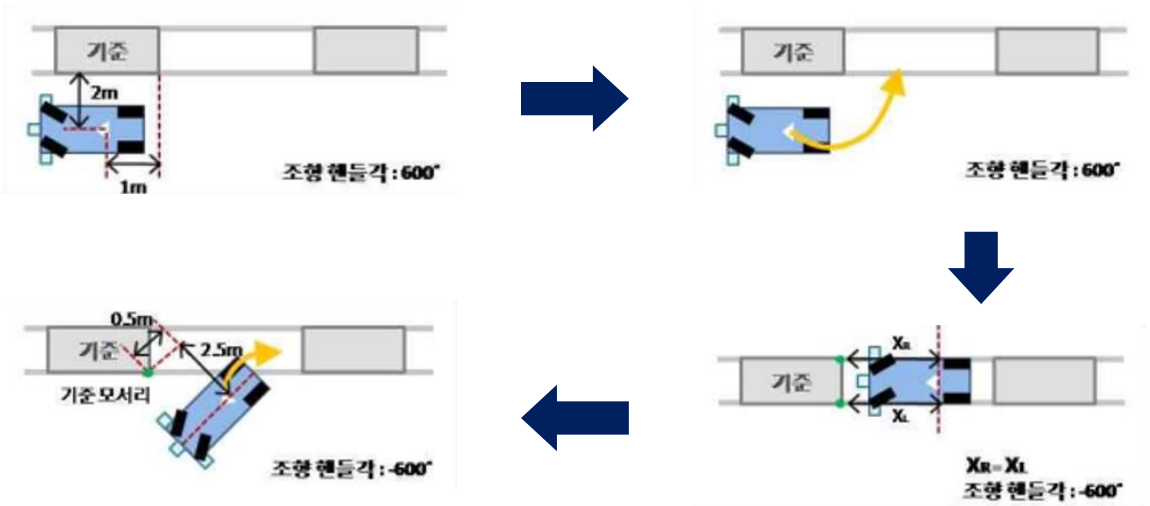
영역 내 감지되는 LiDAR의 Point의 개수가 10개 이상일 경우 장애물이 존재하는 것으로 판단

확실한 판단을 위해 2개의 정차 위치를 지정 후 2번에 걸쳐 주차 공간을 판단

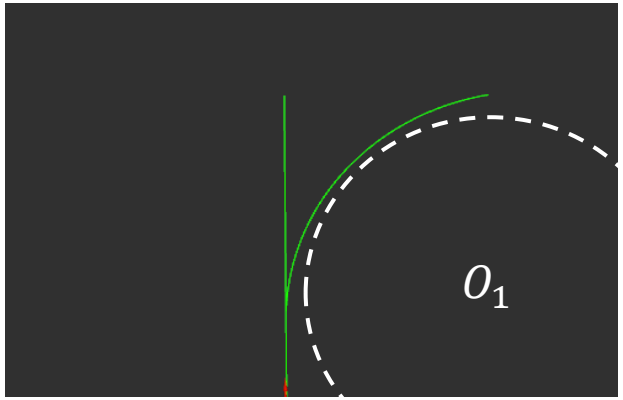
해당 정보를 차량에게 전달하여 주차 공간까지의 경로를 생성

Autonomous Parking System

수평 주차 경로 추종



사선 주차 경로 생성



[ROS Rviz]

Path generation using two virtual circles

- LiDAR를 이용하여 파악한 주차 가능 공간에 후면 주차 혹은 정면 주차하기 위한 경로를 생성
- 사선 주차의 경우 주차 가능 공간 입구의 중심점을 기준으로 전역경로의 주변으로 가상의 원을 생성
- 수평 주차의 경우 차량의 위치에 따라 조향 핸들각을 조정하는 방식으로 주차

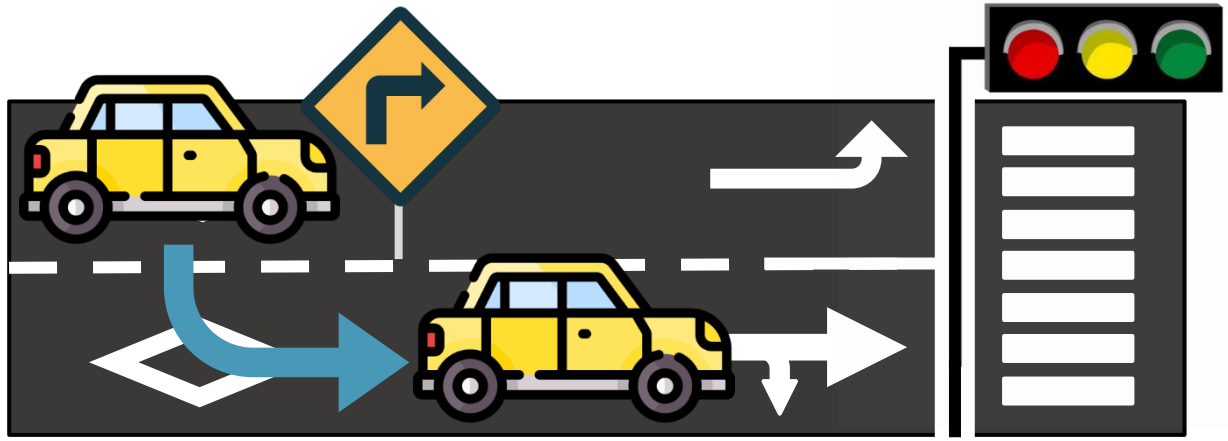
한국 자동차 공학회, "자동 주차 시스템의 경로 생성 및 추종 알고리즘 개발", 2020.05

Autonomous Parking System

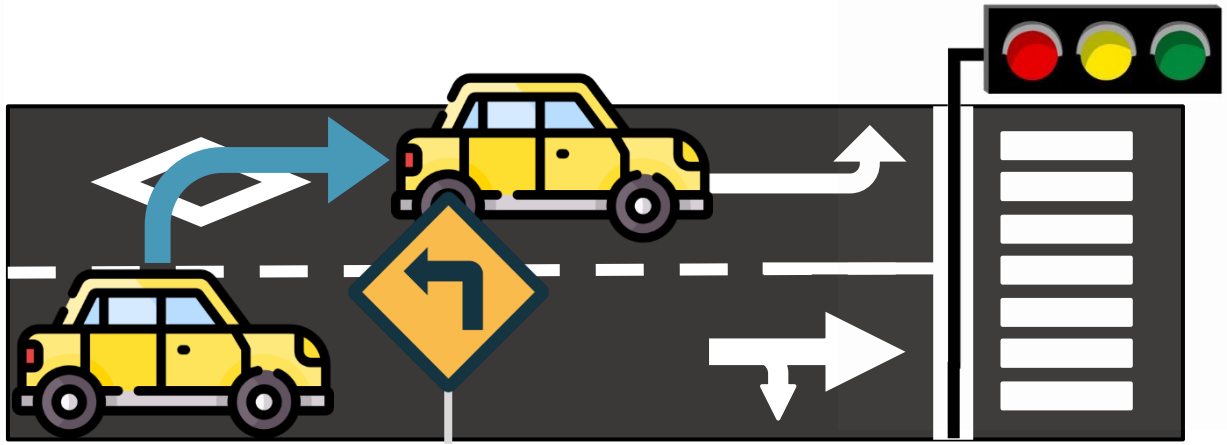
사선 주차 | 평행 주차

사선 주차 | 평행 주차

Intersection Driving



Turn Right Driving Case



Turn Left Driving Case

Intersection

- 직진 및 좌·우회전 수행
- 신호등 정보 인지 후 경로에 맞는 차선 선택 및 차선과 장애물을 고려한 주행
- 신호등 정보가 목표 상황과 다를 경우 정지선 인식 및 정지선 앞 정지
- 신호등 정보가 목표 상황과 같을 경우 정해진 path에 따라 주행

Intersection Driving

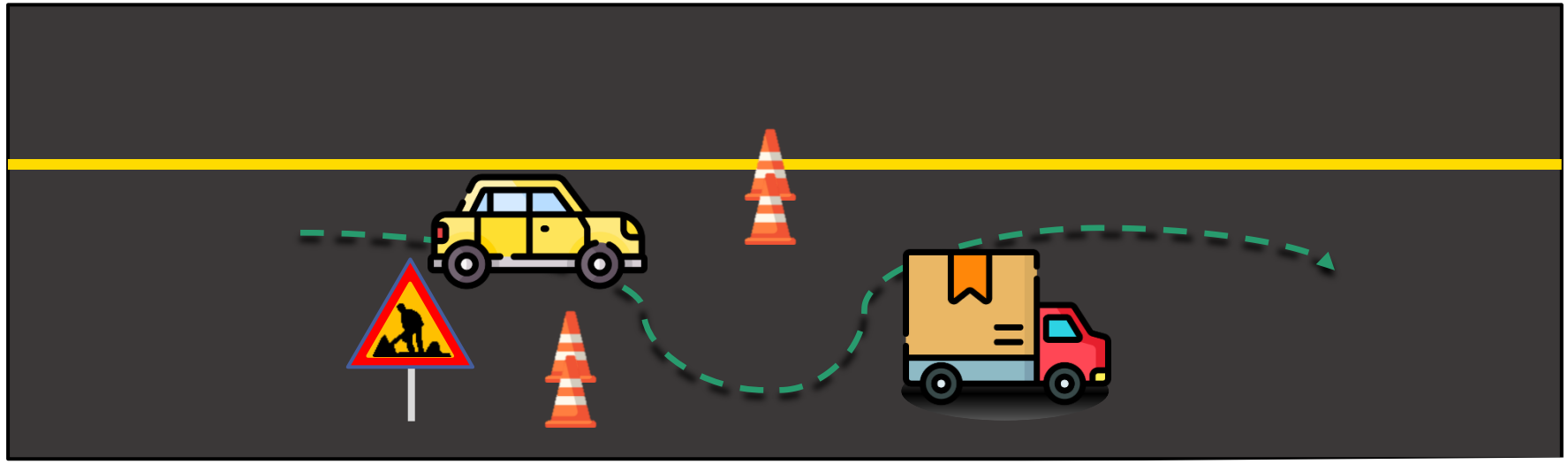
전체 과정

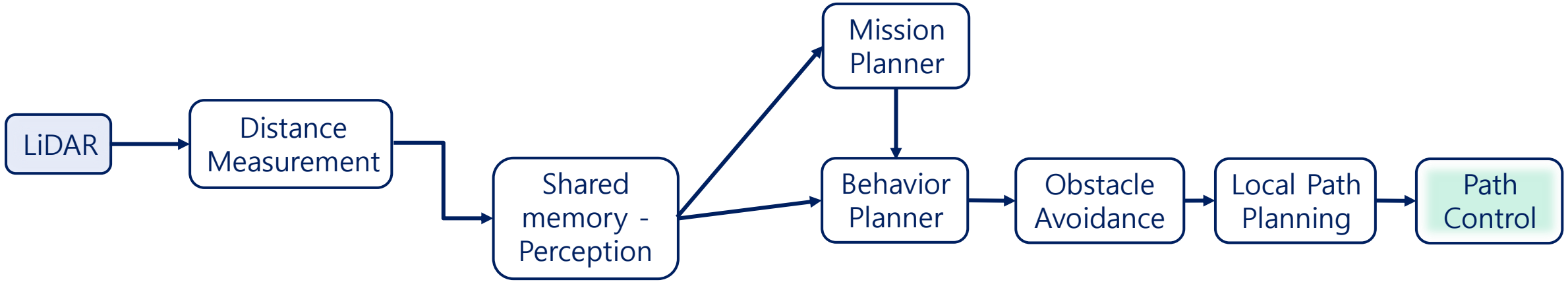
Class Name	Class Number
Red	0
Yellow	1
Green	2
Red_Left	3
Green_Left	4

Class	Stop	Turn Left	Turn Right	U-Turn	Go Straight
0	0	X	X	X	X
1	0	X	X	X	X
2	X	X	0	X	0
3	X	0	X	0	X
4	X	0	0	0	0

Intersection Driving

Obstacle Avoidance



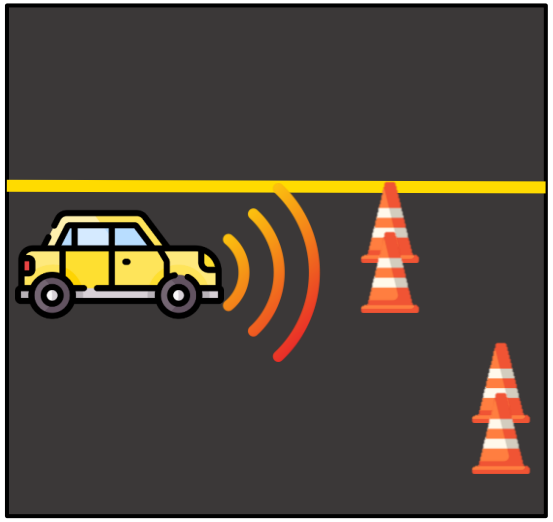


Pickup Car, Shipping Truck, walking, Traffic Cone: flaticon.com.
국제 대학생 창작자동차 경진대회 조직 위원회, "2021년 국제 대학생 창작자동차 경진대회 자율주행 부문 운영 규정", 2022.03

Obstacle Avoidance

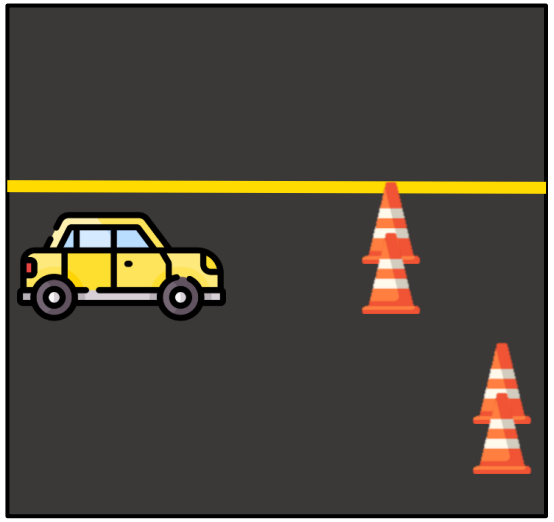
Step 1

LiDAR로부터
객체와의 거리를
전달받음



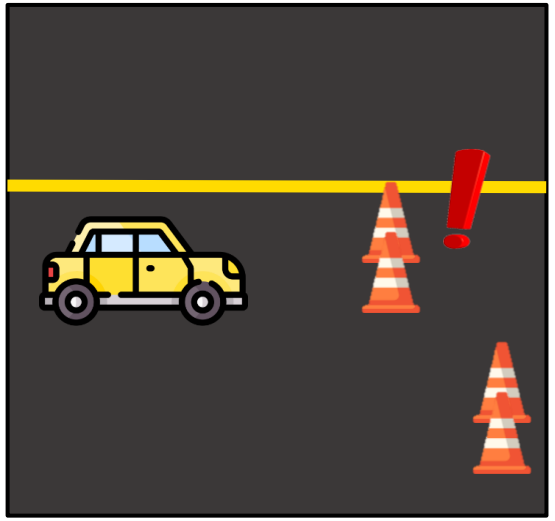
Step 2

LiDAR 기준 상대좌표를
ENU좌표계로 변환



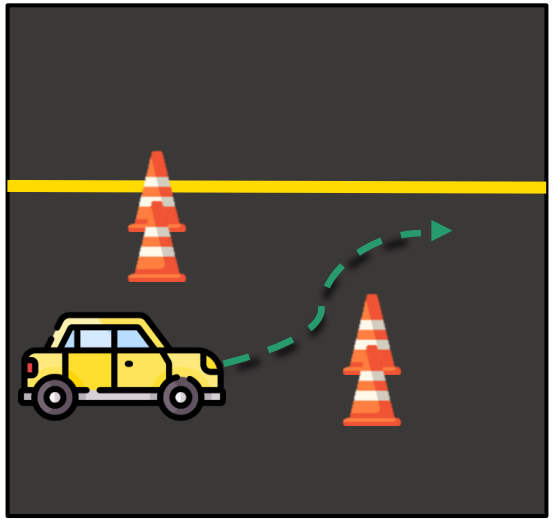
Step 3

장애물이 주행중인
Lattice 경로 위에 있는지
판단



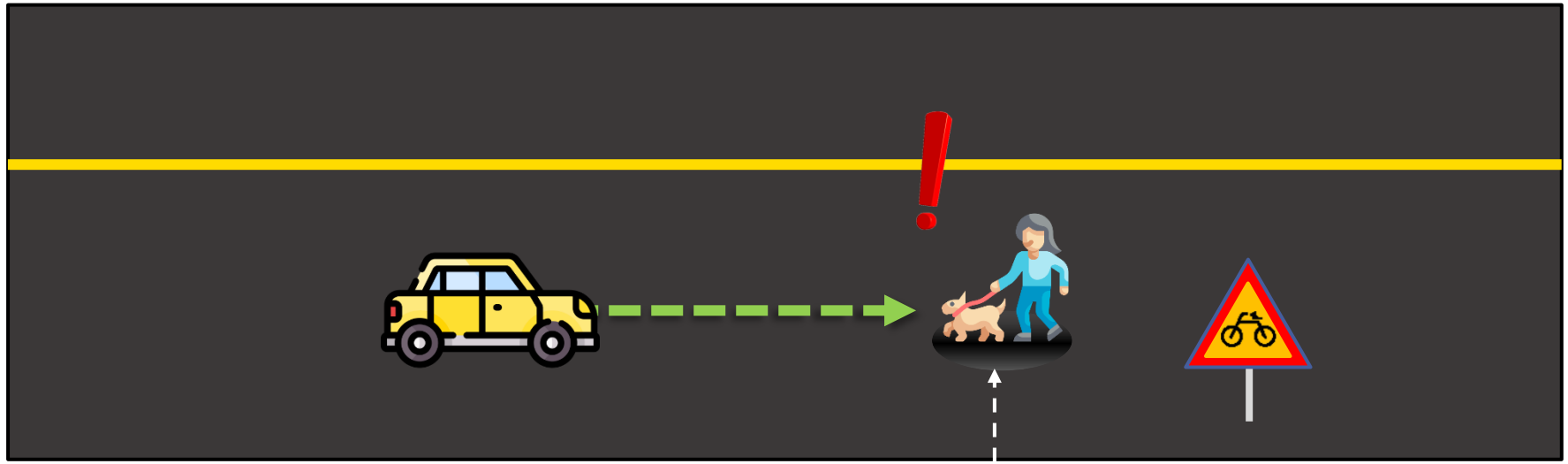
Step 4

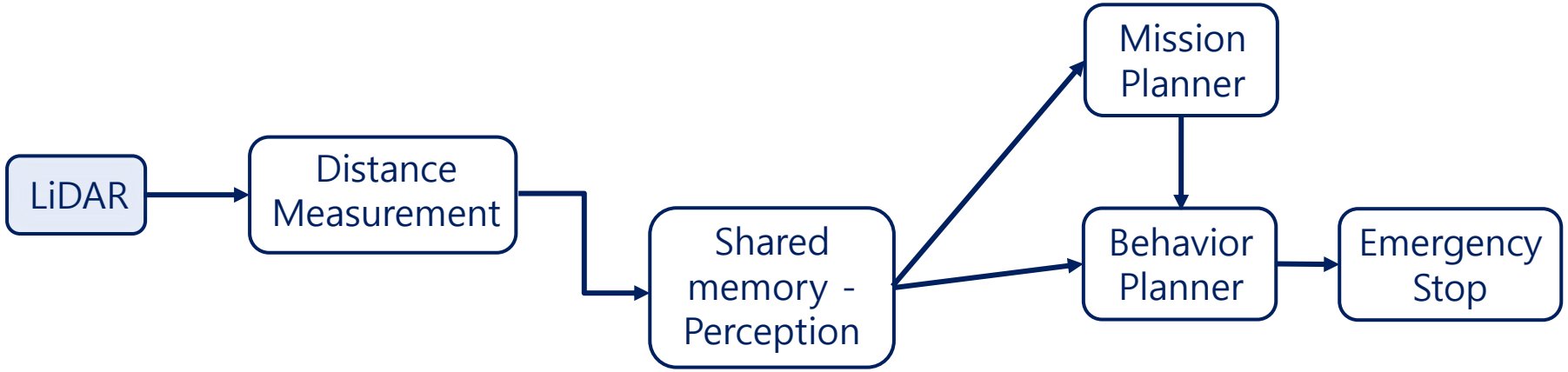
장애물 회피
OR
현재 경로로 주행



Obstacle Avoidance

Autonomous Emergency Brake





Pickup Car, Shipping Truck, walking, Traffic Cone: flaticon.com.
국제 대학생 창작자동차 경진대회 조직 위원회, "2021년 국제 대학생 창작자동차 경진대회 자율주행 부문 운영 규정", 2022.03

Autonomous Emergency Brake

Step 1

LiDAR로부터
객체와의 거리를
전달받음

Step 2

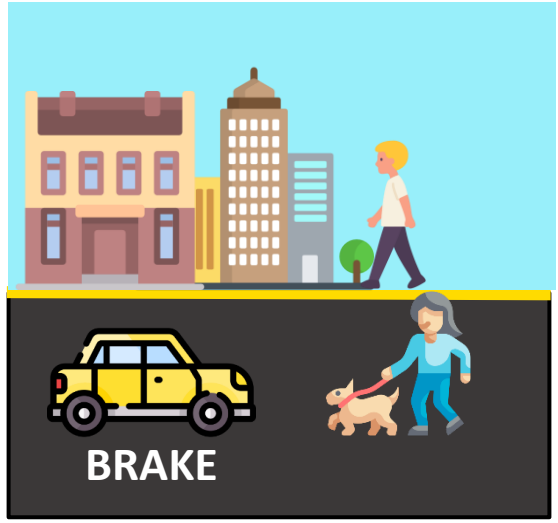
LiDAR 기준 상대좌표를
ENU좌표계로 변환

Step 3

장애물이
도로위에 있는지 판단

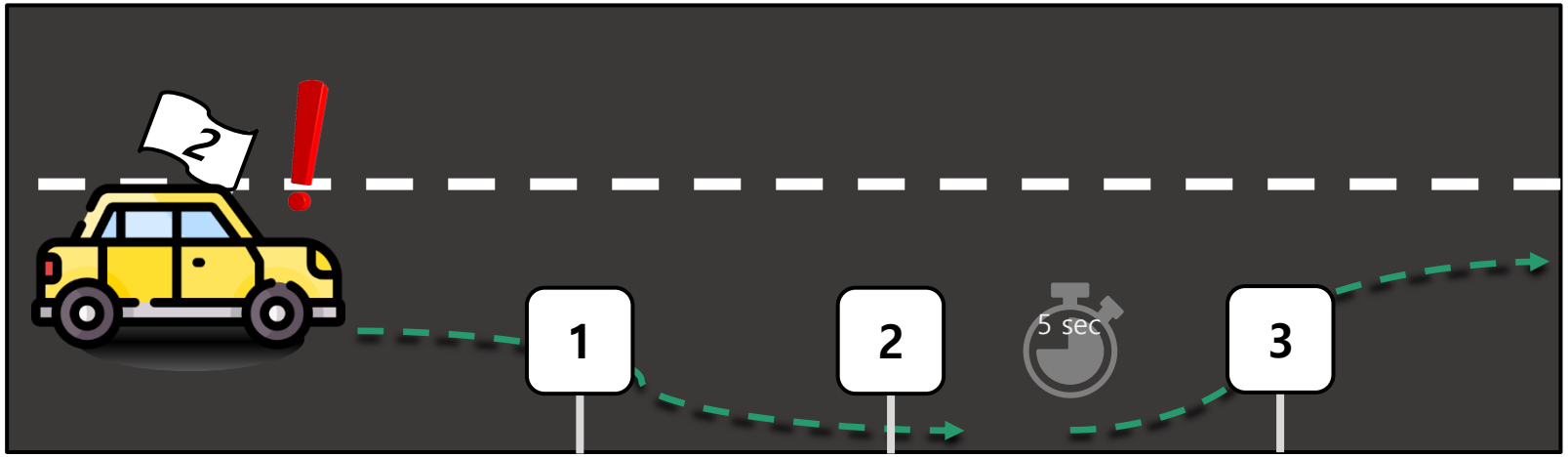
Step 4

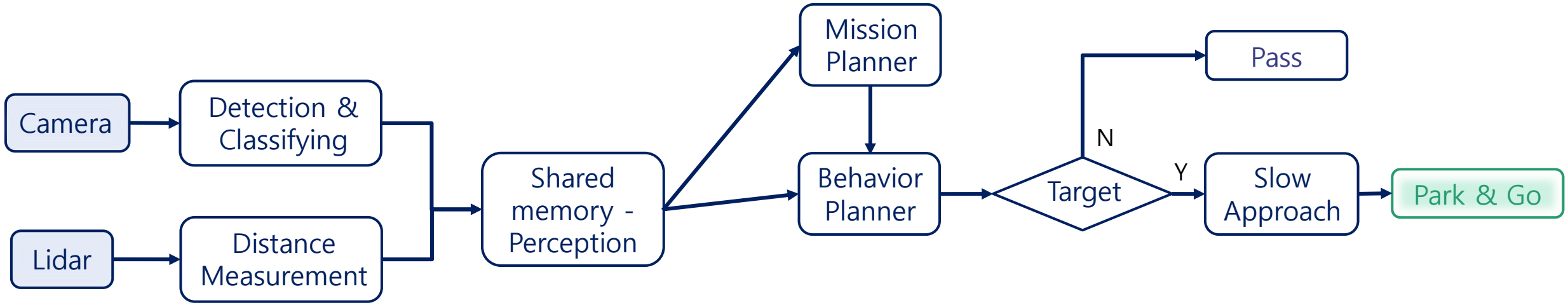
긴급정지



Autonomous Emergency Brake

Pick Up & Delivery

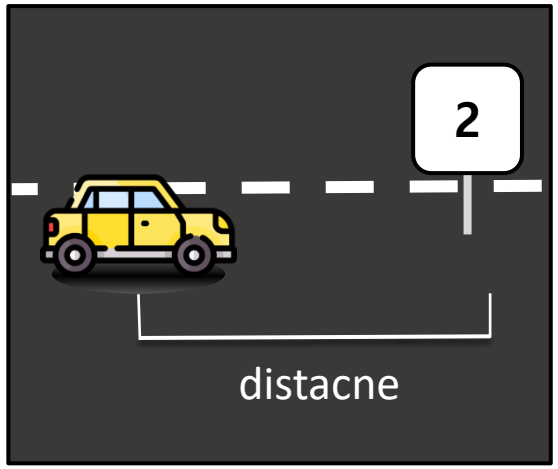




Pickup Car: flaticon.com.
국제 대학생 창작자동차 경진대회 조직 위원회, "2021년 국제 대학생 창작자동차 경진대회 자율주행 부문 운영 규정", 2022.03

Step 1

LiDAR, 카메라
캘리브레이션 정보를
통해 Pick-up 표지판까지의
distance 및 위치를 알아냄

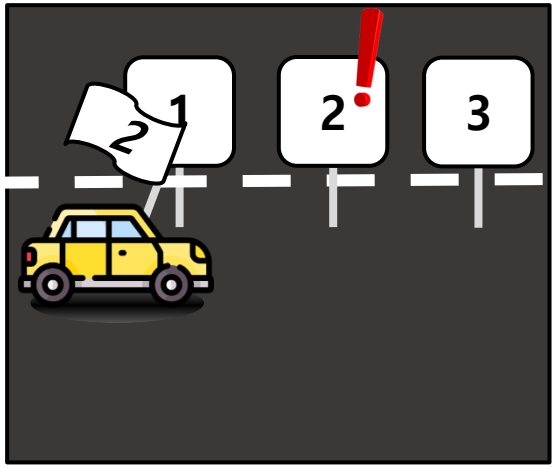


Step 2

배달지 확정
and
표지판 옆에서 정차

Step 3

LiDAR, 카메라
캘리브레이션 정보를
통해 배달 표지판을 인지한 후
배달 위치를 알아냄



Step 4

배달 장소에 정차

Pick Up & Delivery

4. 결과 및 고찰

- 대회 결과
- 고찰

대회 예선 트랙 주행

대회 예선 미션 주행

LOCAL

1. (정확한 Yaw Rate가 나오는)IMU센서를 사서 Kalman을 적용한 INS를 구축하기 (Offset 알리즘 이용 X)
2. 단일 GNSS수신기로 정확한 헤딩 잡는 법이 필요.
3. 정밀지도에서 위, 경도 외에 여러 정보들도 사용해보기
4. 정밀지도 외에도 직접 맵 만드는 것을 체계화 하기.

LiDAR

1. 딥러닝 기술 이용
2. 주요 인지 센서로 활용하지 말고 비전 많이 활용
3. 고도화되는 미션 난이도에 맞춘 객체 추적 알고리즘
4. 센서 퓨전 이용하려면 비전 요구 명세 및 라이다 요구 명세 명확하게

VISION

1. 인지 결과 중 신뢰성 있는 거 필터링해서 보내기
2. 라이다-카메라 센서 퓨전 권장
3. 고조도/저조도 환경 대응
4. 고해상도/최적화
5. 네트워크 선정 (구/신? Object detection? Segmentation?)

PLANNING & CONTROL

1. 곡률정보를 이용한 속도제어
2. 다양한 횡제어기 경험 후 각각의 정확도 분석하여 결정
3. 속도 등 안정성이 최우선 된 주행 고려
4. 멀티 프로세스 & 멀티 스레드 성능 비교
5. 센서와 차량의 좌표 통일

하드웨어

1. 운영규정 확인 후 적절한 재료로 하드웨어 구상
2. 소프트웨어 알고리즘 개발이 선행된 뒤에 하드웨어 제작
3. 차량동역학에 기반한 하드웨어 설계

통합

1. 다양한 상황에서의 실차 검증
2. 전부 명시적으로 체크리스트 만들어서 체크
3. 최적의 파라미터 값들 전부 픽스하기
4. 통합환경 미리 테스트 해보기
5. 코드 공유 담당 인원이 신경써서 관리 (git)

5. Q & A