

지능 로봇 – II

Autonomous Vehicles

이건명

충북대학교 대학원 산업인공지능학과

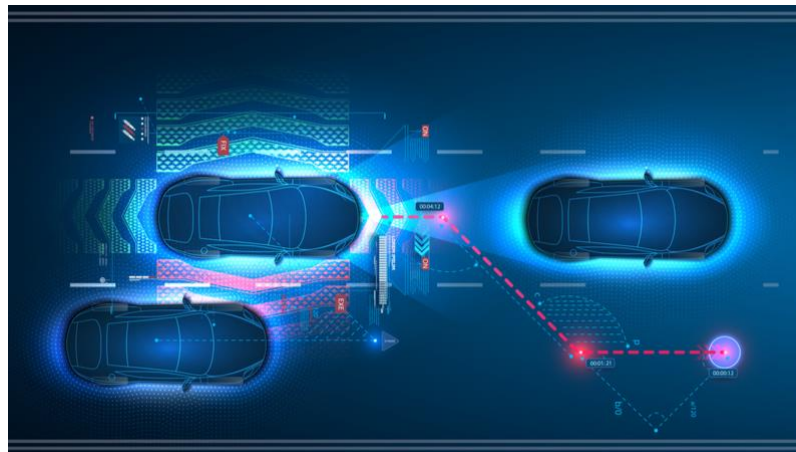
학습 내용

- 자율주행차에 대해서 알아본다.
- 자율주행 핵심요소인 인식 문제와 센서 융합, 위치 추정, 계획 수립, 제어 등에 대해서 알아본다.

1. 자율주행

❖ 자율 주행

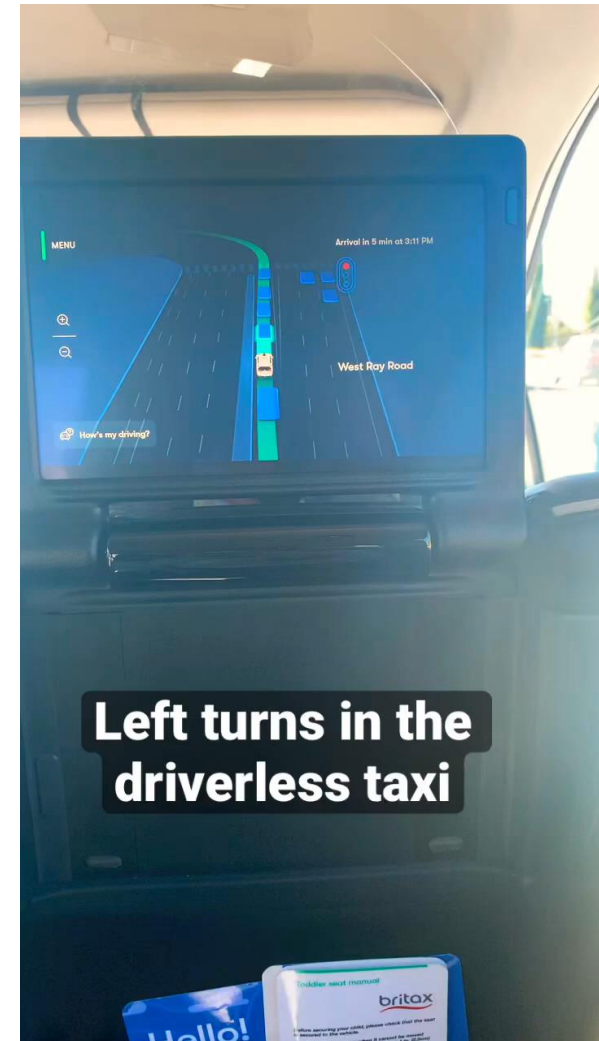
- self-driving car, autonomous vehicle, driverless car, robotic car (robo-car)
- **SAE**(Society of Automotive Engineers) 자율주행 수준
Level 0 - 비자동화;
Level 1 - 운전자 지원(hands on/shared control);
Level 2 - 부분 자동화(hands off);
Level 3 - 조건부 자동화(eyes off);
Level 4 - 고도 자동화(mind off);
Level 5 - 완전 자동화(steering wheel optional)



자율주행

❖ 자율주행 운행 사례

- 승용차 자율주행 모드, 자율주행 셔틀, 자율주행 택시
- 자율 군집 트럭
- 전국적 시범 서비스
 - 서울시(국회의사당 부근, 청계천, 청와대 등)
 - 인천공항, 세종, 제주
 - ...



자율주행의 핵심요소

❖ 인식(perception)

- 주변환경과 장애물 인식

❖ 위치추정(localization)

- 환경 내에서 차량의 위치 추정 (cm 단위의 정확도 요구)

❖ 계획수립(planning)

- 인식 내용 및 추정 위치를 토대로 원하는 위치로 이동을 위한 궤적 생성

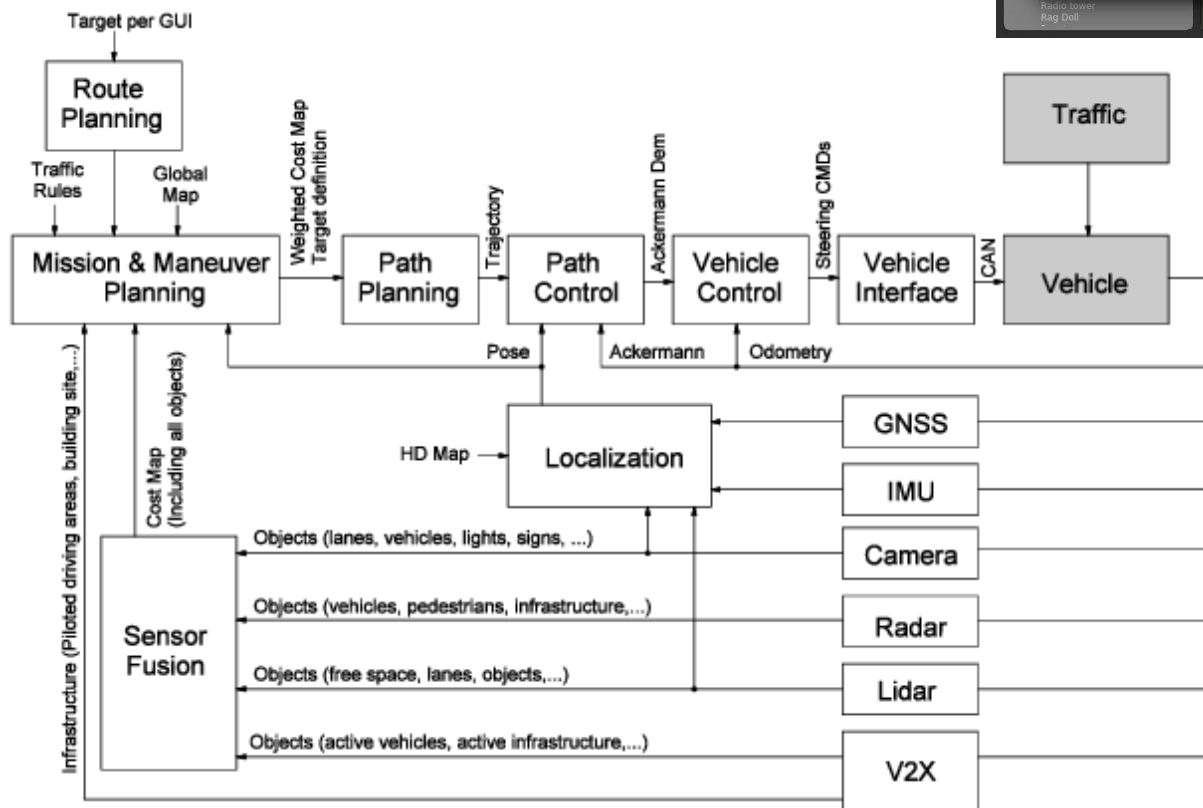
❖ 제어(control)

- 궤적을 따라 이동하기 위해 조향 및 가속정보 생성

자율주행 개발 플랫폼

❖ ROS 2 기반 자율주행 구조

- ROS 2 Iron Irwini : 실시간 지원
- Gazebo 시뮬레이터 : 시뮬레이션 환경 제공



자율주행 개발 플랫폼

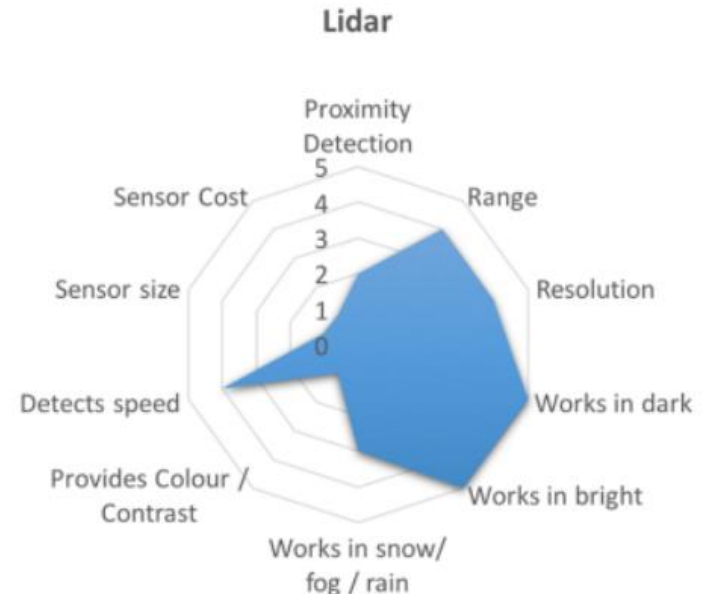
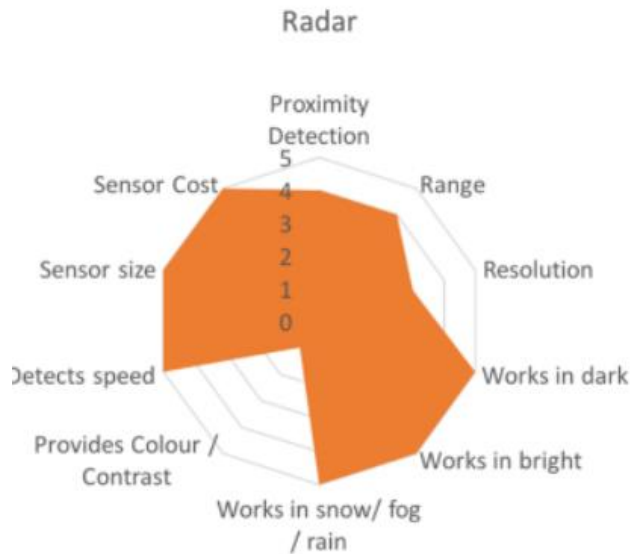
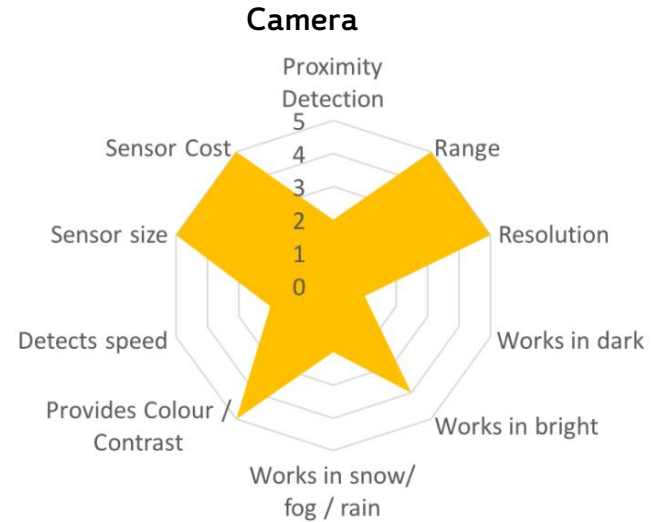
❖ ROS 2 기반 자율주행 시스템 개발 과정

1. **ROS 2 설치 및 환경 구성**
2. **센서 통합**: 다양한 센서(LIDAR, 카메라, GPS, IMU 등)와 인터페이스하기 위해 ROS 2 노드(개별동작 프로세스) 사용
3. **데이터 처리**: 센서 데이터 융합, 객체 탐지, 위치 확인, 매핑을 위한 알고리즘 구현 또는 ROS2 데이터 처리 패키지 사용
4. **제어 시스템**: 해석된 센서 데이터를 처리하여 조향, 가속, 제동 등의 주행 동작을 실행하는 차량 제어 시스템을 위한 노드 개발
5. **시뮬레이션**: Gazebo 등의 ROS 2 호환 시뮬레이터를 사용하여 가상 환경에서 자율주행 알고리즘 테스트
6. **통신**: ROS 2 메시징 시스템을 활용하여 노드 간 통신을 구현. 자율주행 파이프라인을 통해 데이터의 원활한 통신 구현 및 확인.
7. **안전성 및 중복성**: 시스템에 안전 조치와 장애 처리를 위한 중복 시스템이 포함되어 있는지 확인
8. **테스트 및 검증**: 제어된 환경에서 자율주행 시스템을 철저히 테스트하고, 엄격한 안전 프로토콜 하에 실제 환경의 시나리오에서 테스트
9. **반복 및 최적화**: 테스트 데이터와 실제 주행 피드백을 바탕으로 알고리즘과 시스템을 지속적으로 업데이트하고 최적화

2. 자율주행 인식 문제

❖ 인식 센서의 종류

- 카메라
- Lidar
- Radar



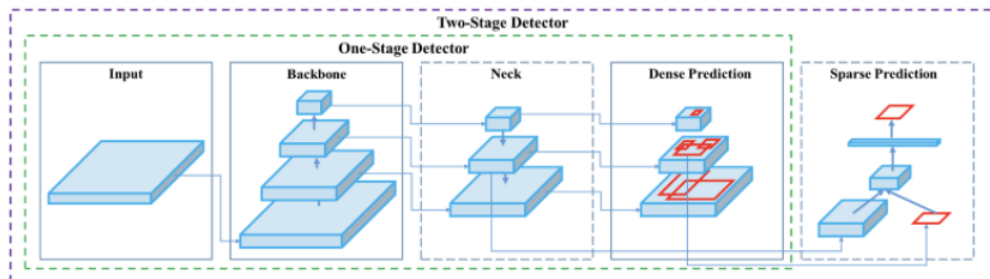
자율주행 인식 문제

❖ 2D 물체 감지

- 물체의 **경계상자**(bounding box) 정보 제공



- YOLO, SSD, RetinaNet



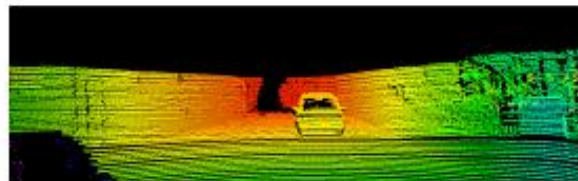
자율주행 인식 문제

❖ 3D 물체 감지

- RGB-D 센서 데이터
- Point Cloud 데이터
- RGB + Point Cloud 데이터

Frustum PointNet

Depth



RGB

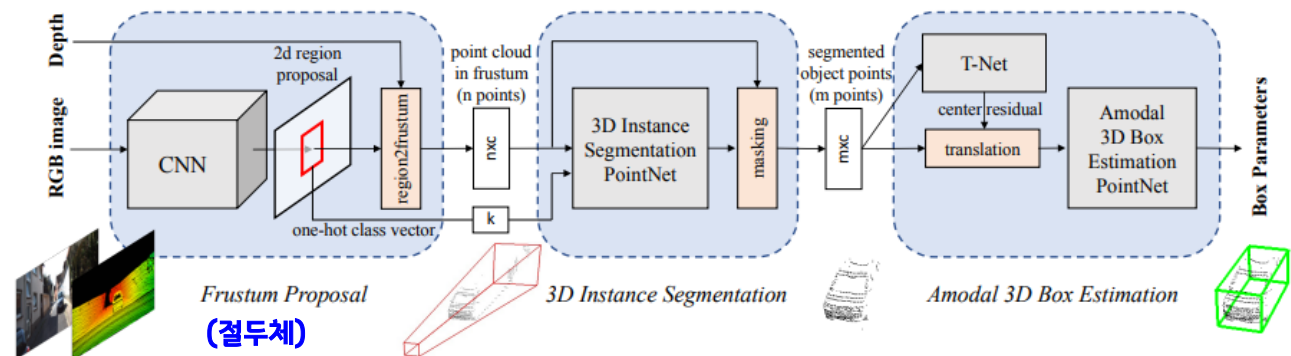


depth to point cloud

3D box (from PointNet)

3차원 상자

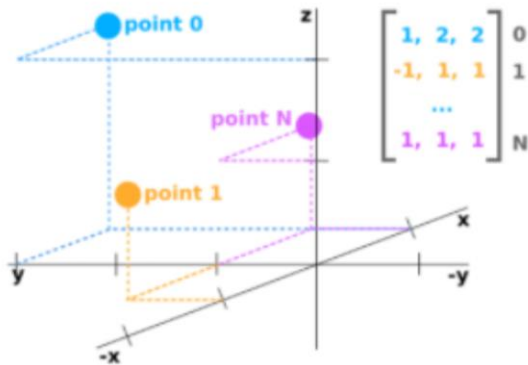
2D region (from CNN) to 3D frustum



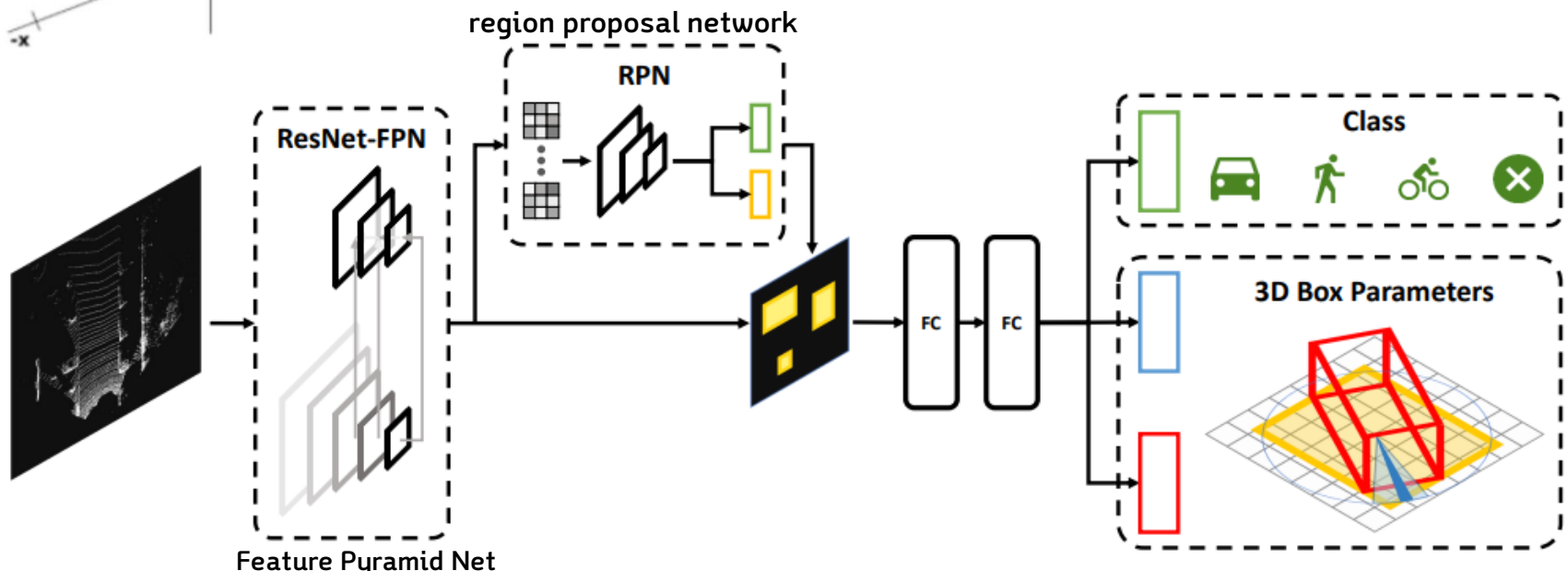
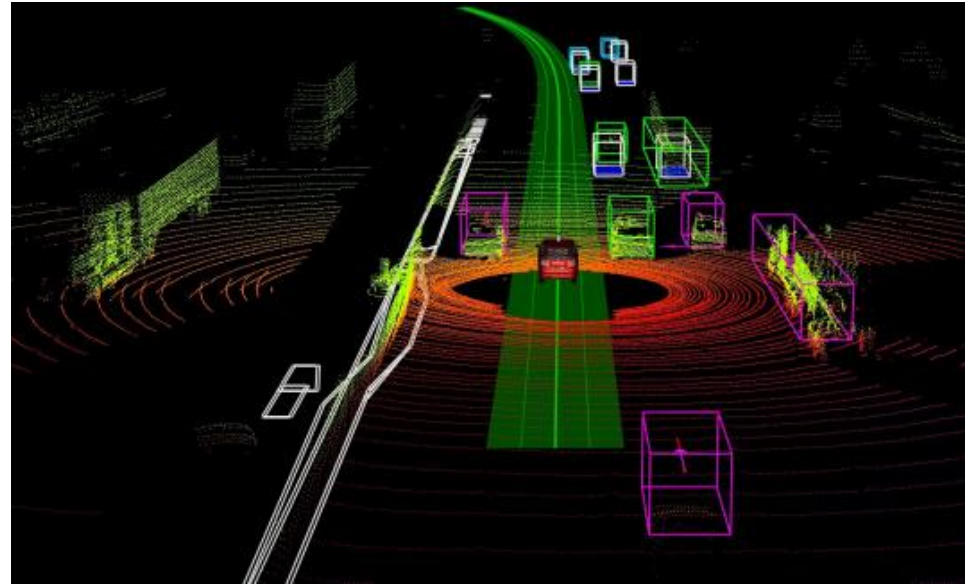
자율주행 인식 문제

❖ 3D 물체 감지 – cont.

- **Point Cloud 데이터**
(lidar 데이터)



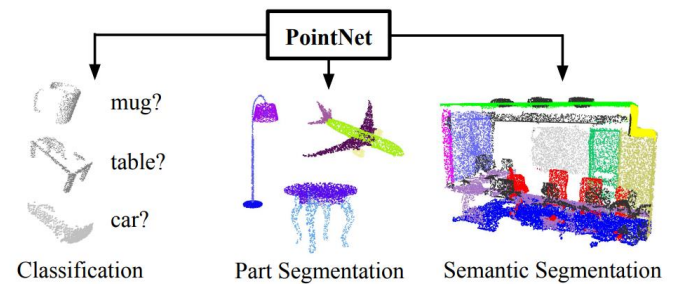
BirdNet+



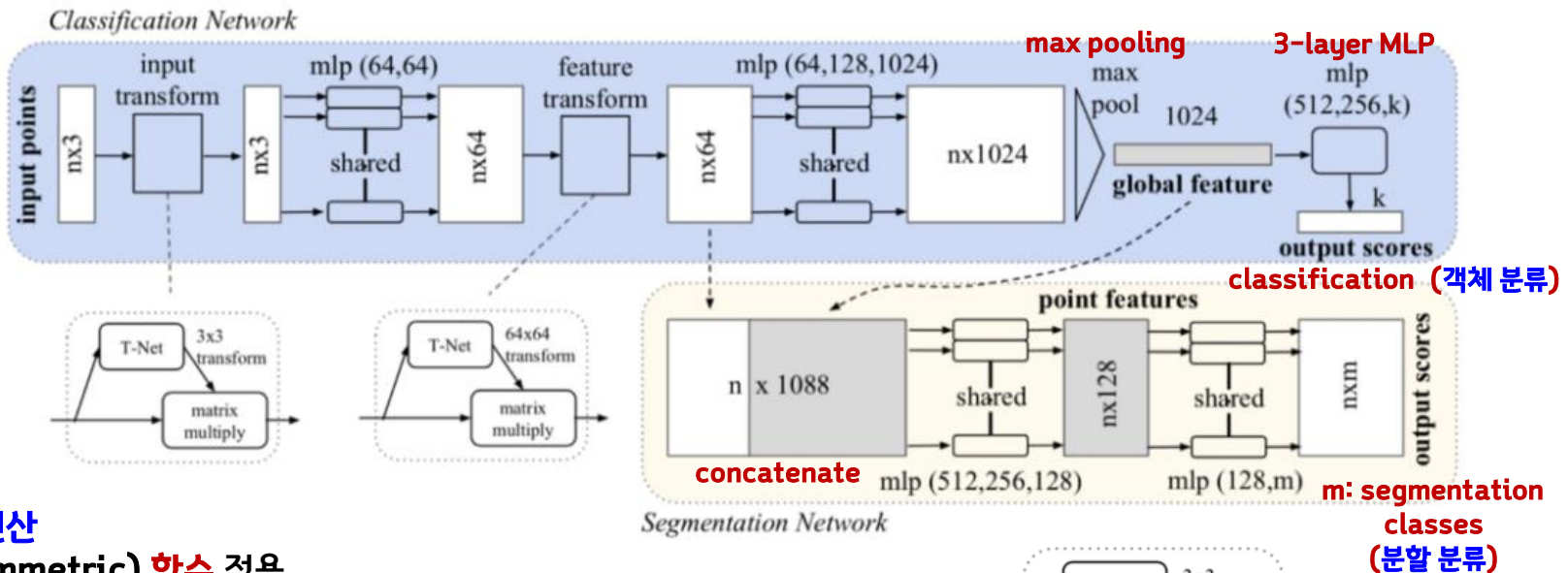
자율주행 인식 문제

PointNet (Qi et al., 2016)

- 순서불변(permutation/order invariance), 변환(transformation) 불변
- Point cloud 데이터



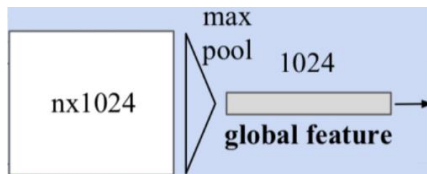
n (x, y, z)
입력



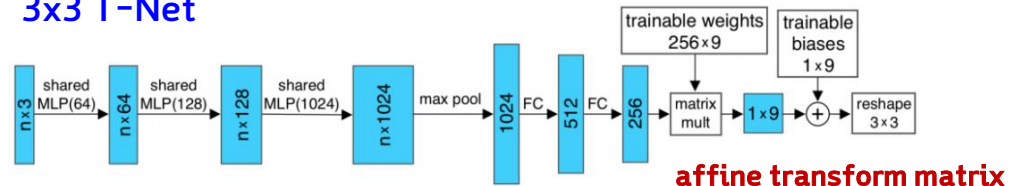
변환(transformation) 불변

순서 불변 연산

- 대칭(symmetric) 함수 적용
- $\text{sum}(a, b) = \text{sum}(b, a)$
- $\text{average}(a, b) = \text{average}(b, a)$
- $\text{max}(a, b) = \text{max}(b, a)$



3x3 T-Net

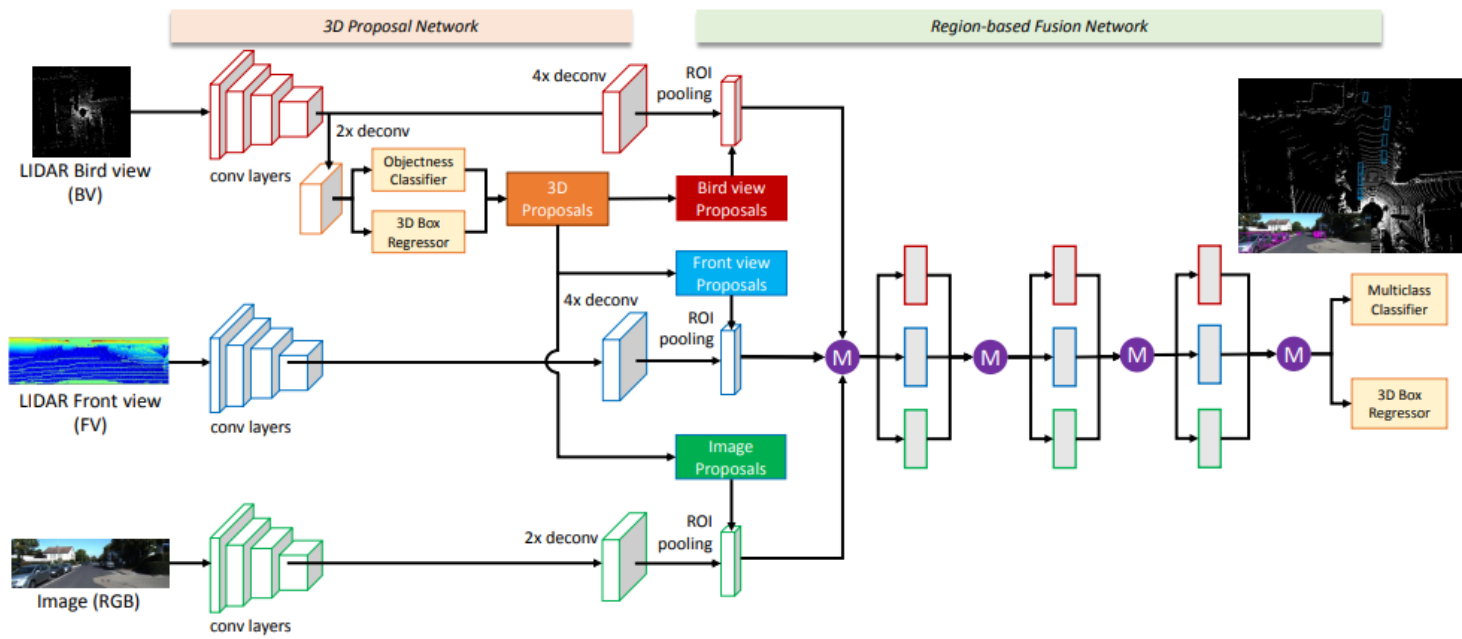
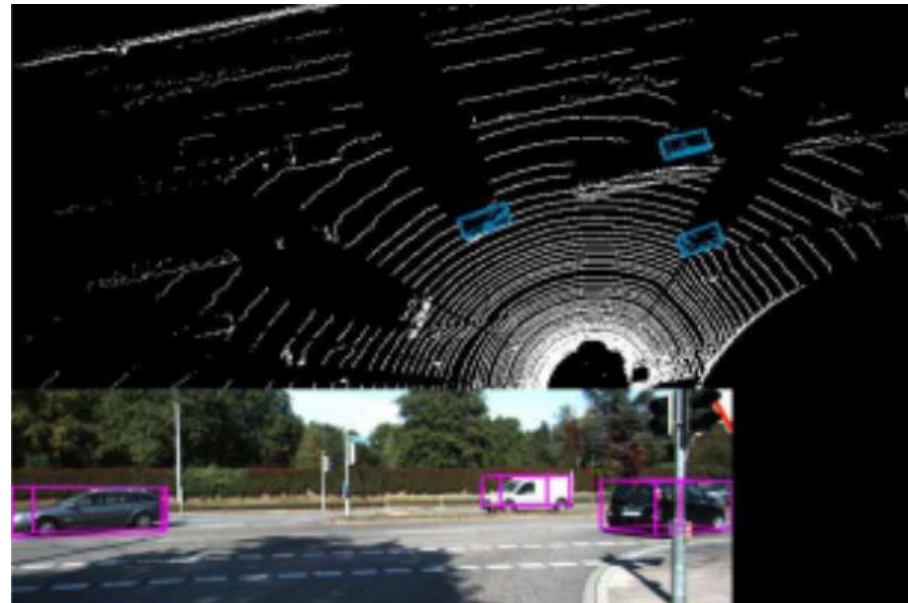


자율주행 인식 문제

❖ 3D 물체 감지 – cont.

- RGB + Point Cloud 데이터

Multi-View 3D networks (MV3D)



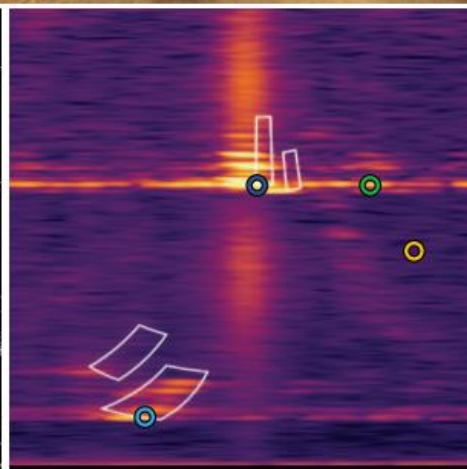
자율주행 인식 문제

❖ Radar 데이터

camera



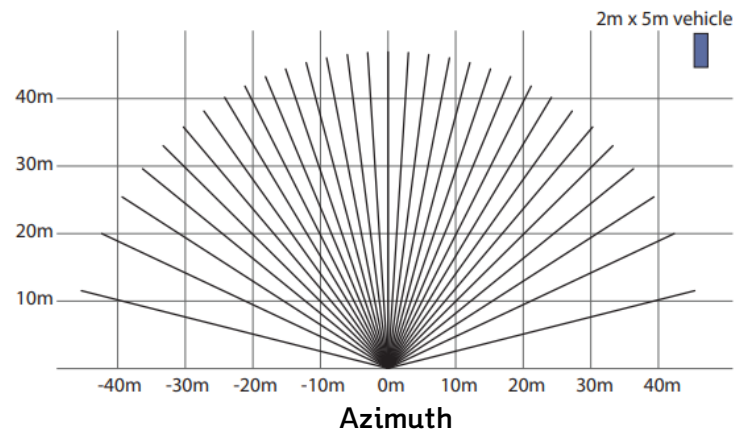
lidar



azimuth (angle, 각도)

radar

range
(거리)



Doppler Radar



Approaching



Reflection is Higher frequency



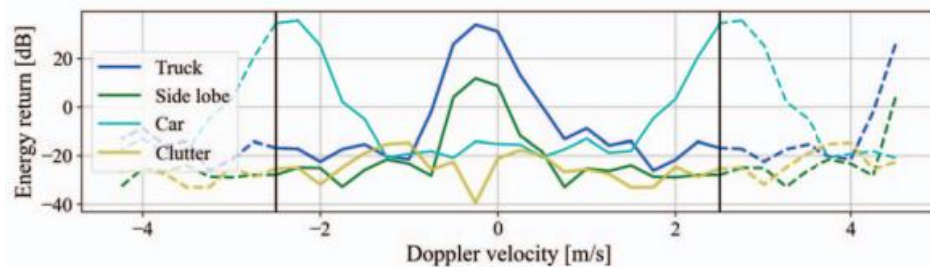
Radar Transmit Frequency



Receding



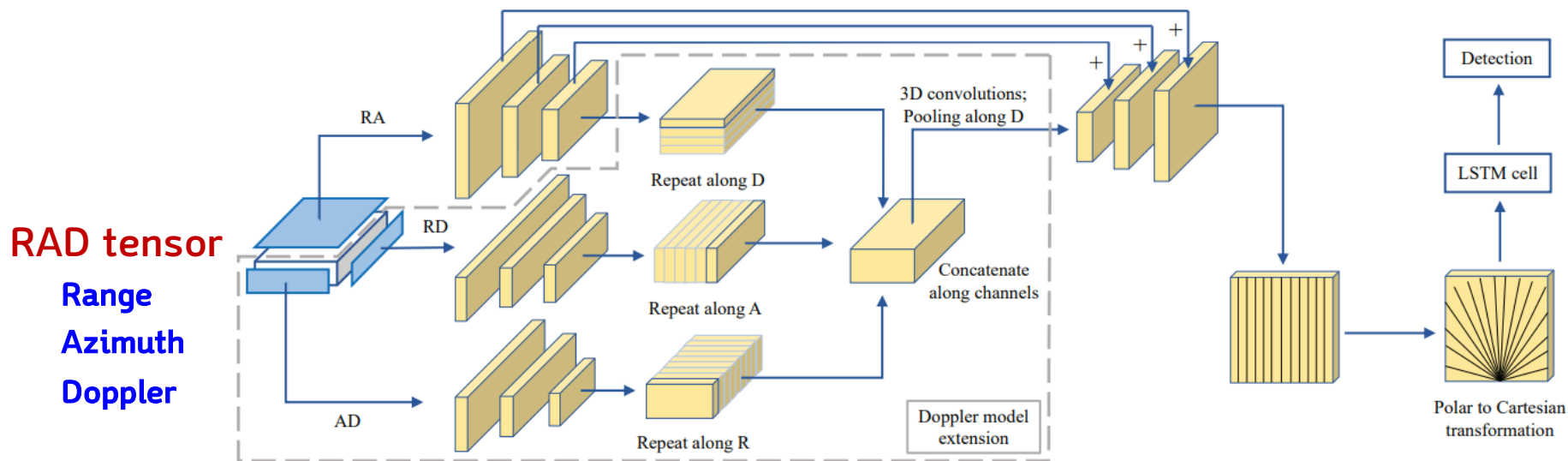
Reflection is Lower frequency



자율주행 인식 문제

❖ Radar 기반 차량 감지 (Major et al. 2019)

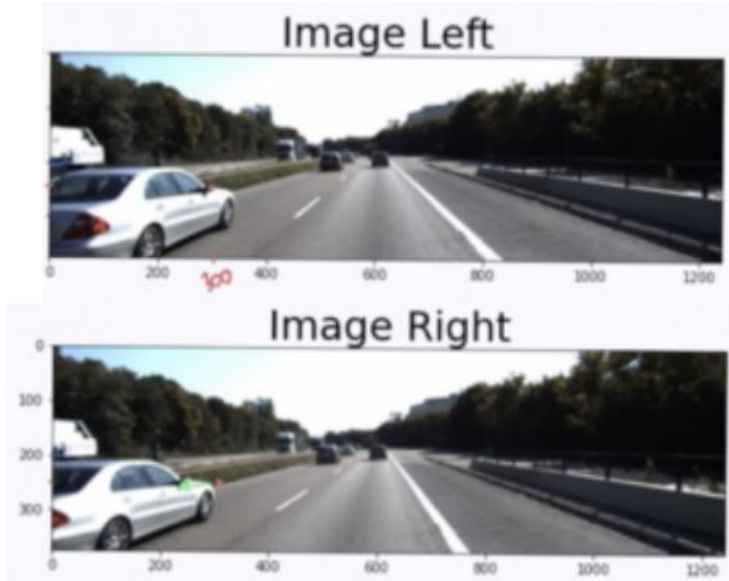
- 레이다 데이터로 부터 주변 차량 감지



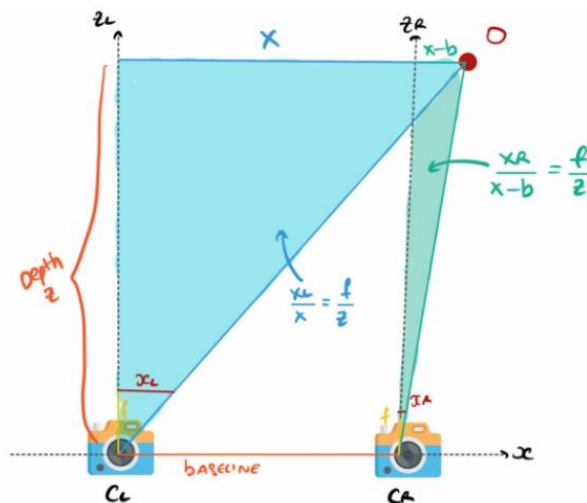
자율주행 인식 문제

❖ 스테레오 비전(stereo vision)

- 2개의 카메라 사용
- 물체까지의 거리 추정



<https://medium.com/think-autonomous/pseudo-lidar-stereo-vision-for-self-driving-cars-41falac42fc9>



$$Z = \frac{f \cdot b}{x_L - x_R} = \frac{f \cdot b}{d}$$

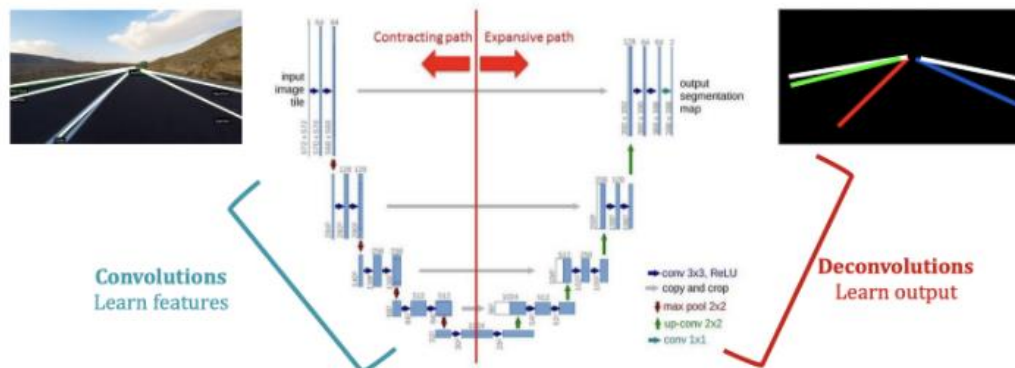
자율주행 인식 문제

❖ 차선 감지

- 차선을 표현하는 식의 계수 추정



- U-Net 기반 차선 감지

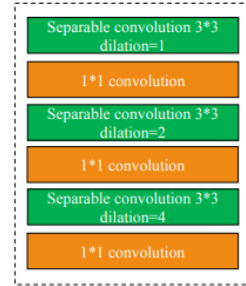
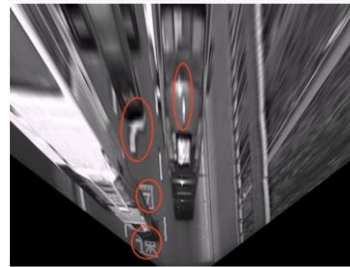


자율주행 인식 문제

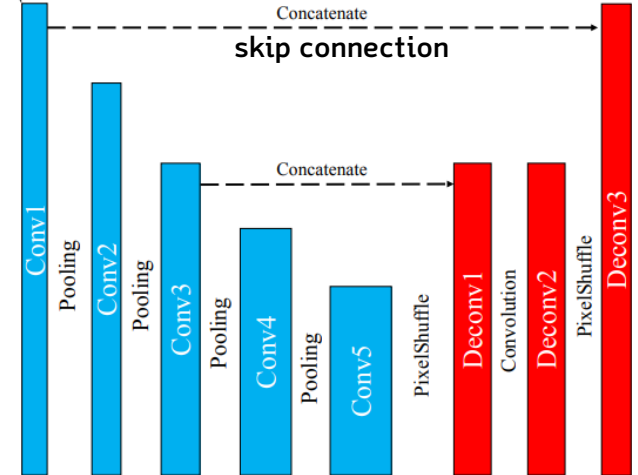
❖ 차선 감지 – cont.

▪ LaneNet

- 2개의 네트워크 구성



Lane edge proposal network - 차선 후보 픽셀 추천



Lane line localization network

- 차선 검출

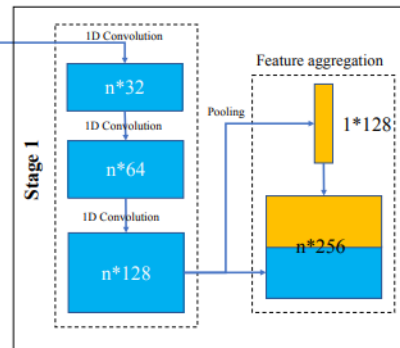
binary lane
edge map



좌표
데이터



Input $n \times 2$

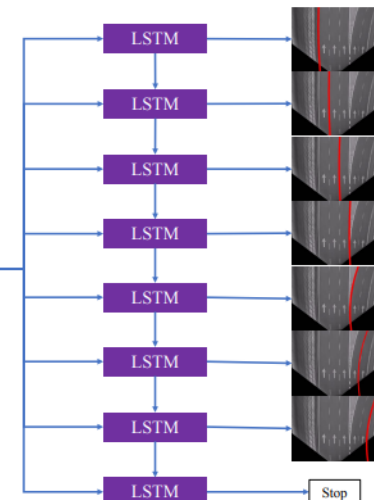


PointNet과 유사한 순서불변 연산

Stage 2

...

Stage k



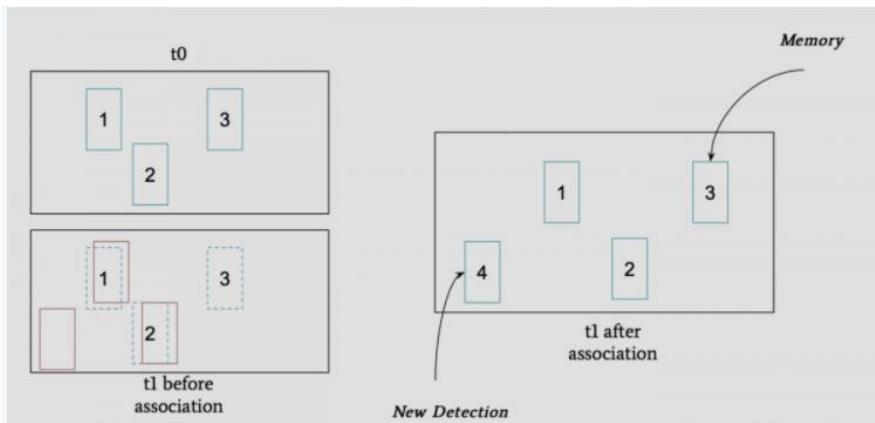
자율주행 인식 문제

❖ 객체 추적(object tracking)

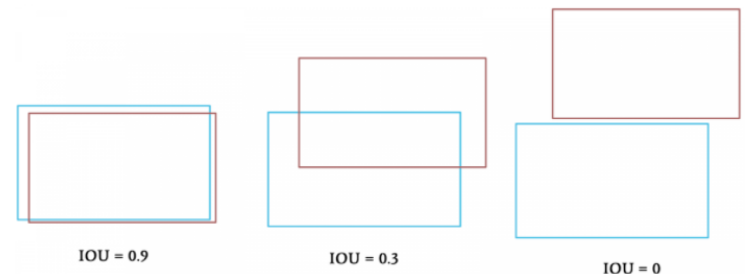
- 동일 대상을 시간에 따라 연속해서 추적
 - 행동 예측



- 물체 감지 → 인접시간의 동일물체 여부 판단, 가림(occlusion) 대응
 - Hungarian 알고리즘**
 - Kalman filter**



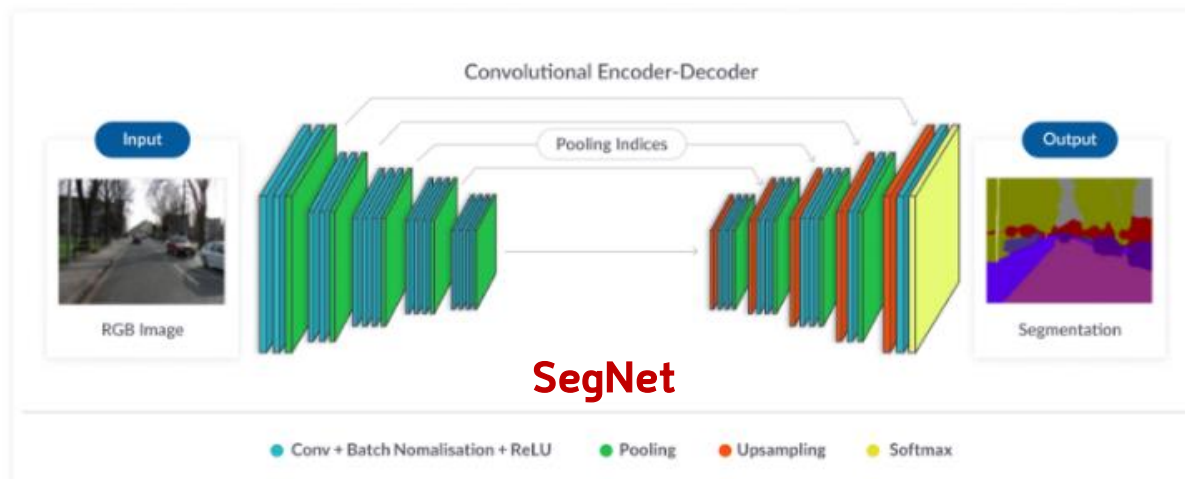
Hungarian Algorithm (Kuhn-Munkres)



자율주행 인식 문제

❖ 빈공간(freespace) 감지

- 차량이 없는 비어있는 공간 감지



자율주행 인식 문제

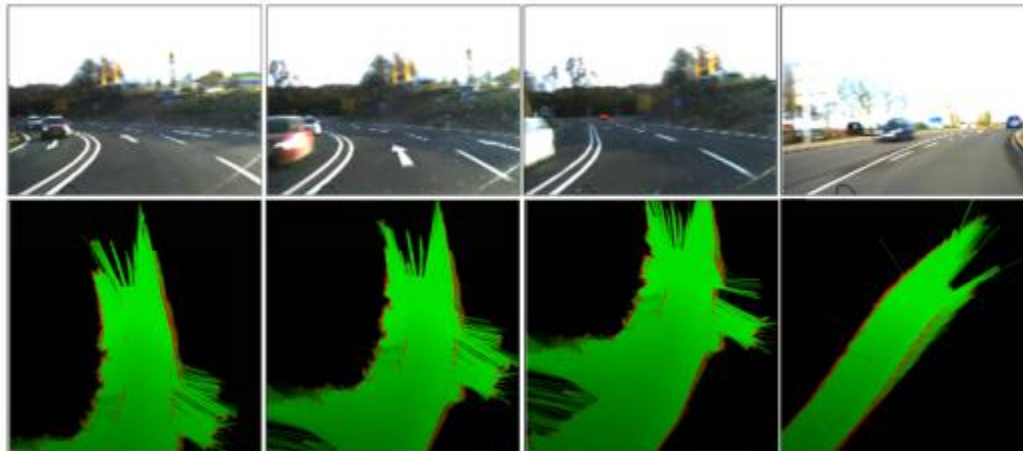
❖ 장면 인식(scene perception)

▪ 의미적 영역분할(semantic segmentation)



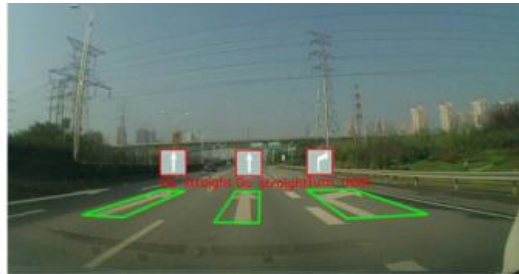
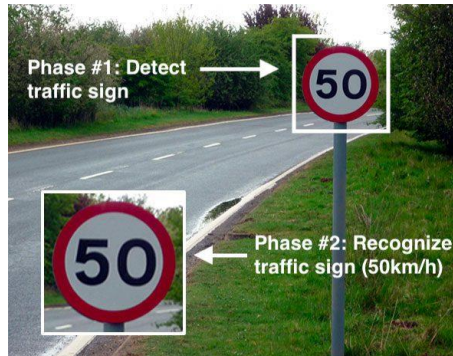
▪ 점유도 (occupancy map, Occupancy Grids)

- 주행공간을 셀로 분할하고 점유 확률 계산



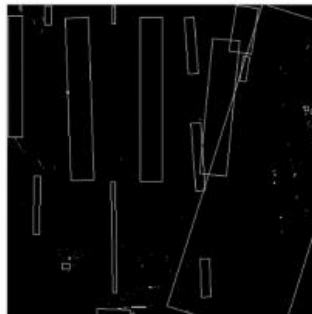
자율주행 인식 문제

❖ 장애물, 도로 표지, 신호등 감지



(a) The original image

(b) detection results of road markings



(c) Candidate region segmentation

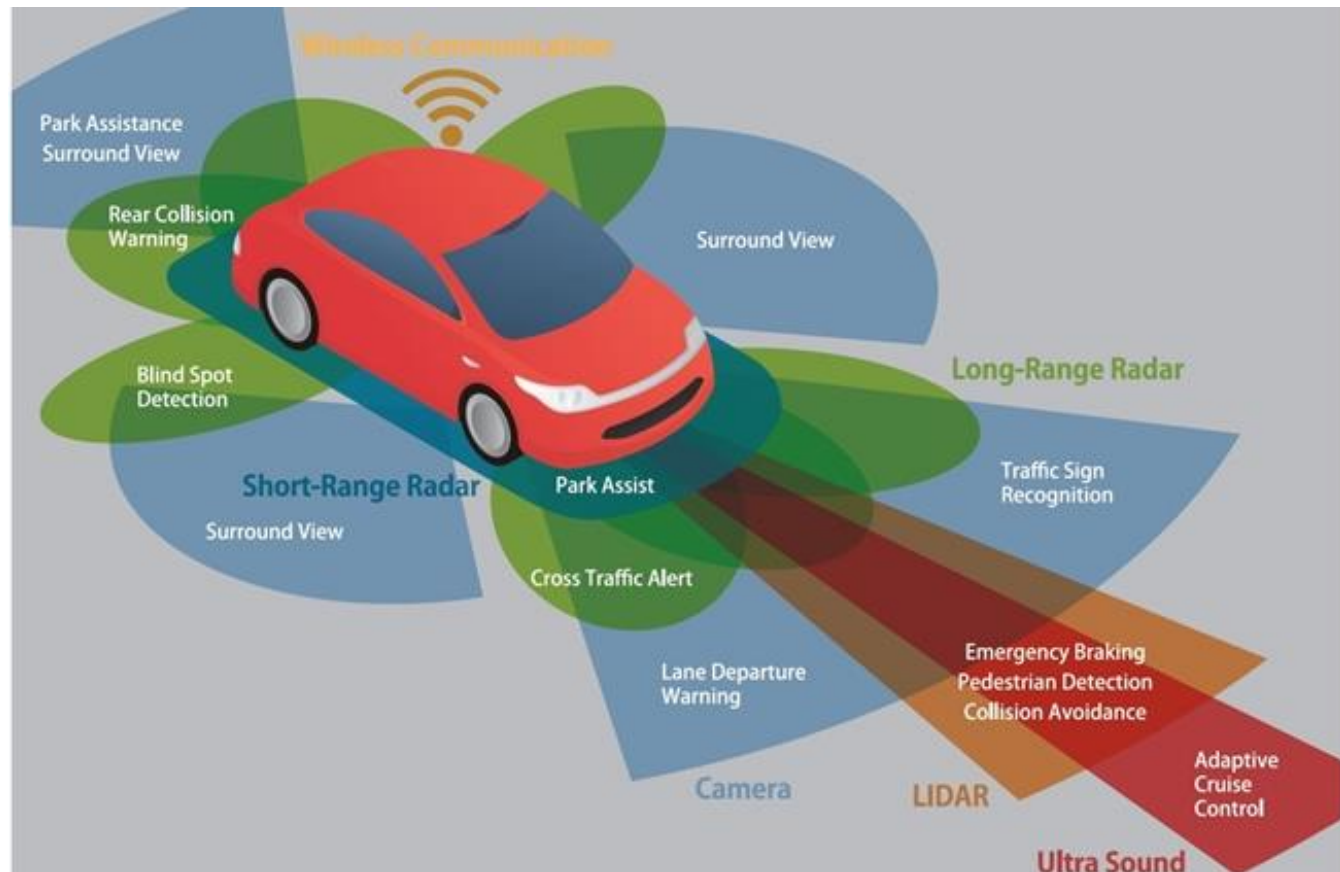
(d) Minimum surrounding contour

(e) Candidate region screening results

센서 융합


❖ 센서 융합(sensor fusion)

- 여러 센서의 데이터를 결합하여 **정확도 높은 추정**을 하는 것
- 완벽한 센서의 부재




센서 융합

❖ 센서별 장단점




RADAR

- Long-range sensing
- Object movement
- All-weather performance



LIDAR

- Precise 3D object detection
- Range accuracy
- Free-space detection



CAMERA

- Object classification
- Object angular position
- Scene context

	RADAR	LIDAR	CAMERA	FUSION
Object detection	+	+	○	+
Pedestrian detection	—	○	+	+
Weather conditions	+	○	—	+
Lighting conditions	+	+	—	+
Dirt	+	○	—	+
Velocity	+	○	○	+
Distance - accuracy	+	+	○	+
Distance - range	+	○	○	+
Data density	—	○	+	+
Classification	—	○	+	+
Packaging	+	—	○	+

+

 = Strength

○

 = Capability

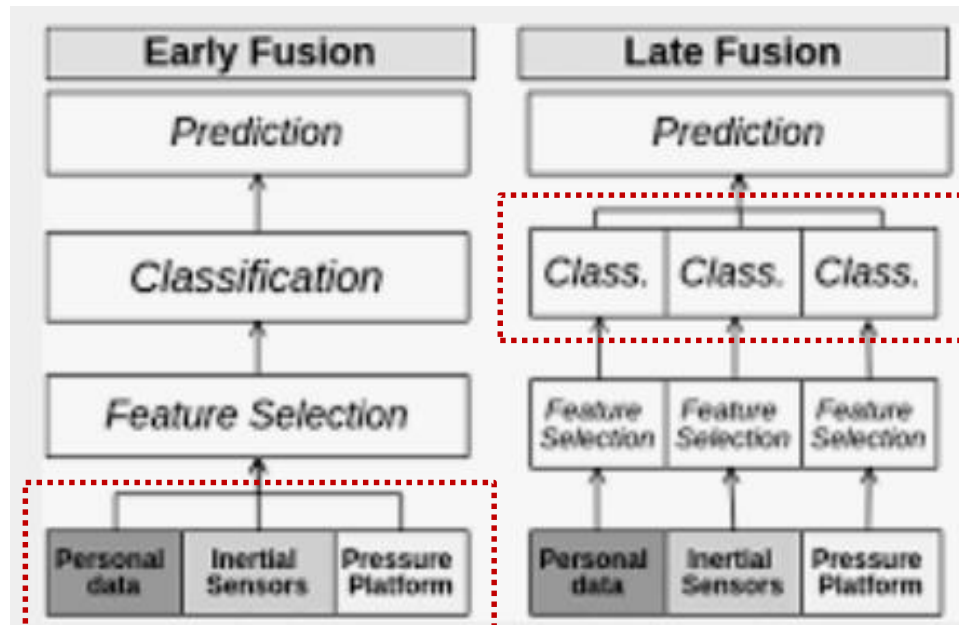
—

 = Weakness

센서 융합

❖ 융합 시점

- 초기 융합(early fusion)
 - 원 데이터의 결합
 - 예. Lidar point cloud와 이미지의 픽셀의 결합
- 지연 융합(late fusion)
 - 2D 또는 3D 경계상자(bounding box) 검출기 출력의 결합

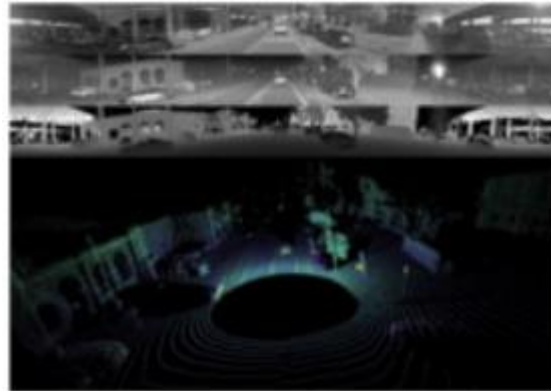


센서 융합

❖ 융합 형태

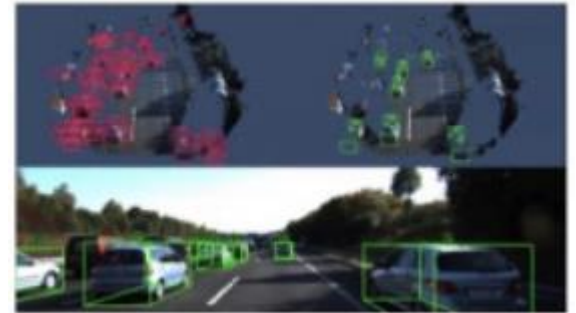
▪ Lidar / Camera

- 초기 융합
- 지연 융합



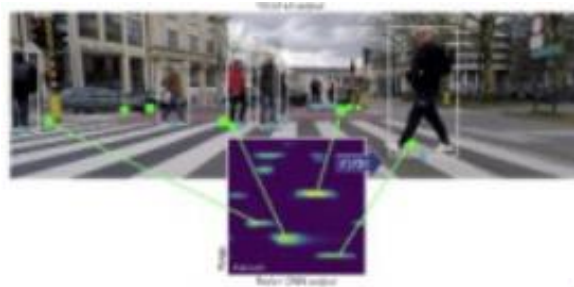
▪ Lidar / Stereo camera

- 지연 융합



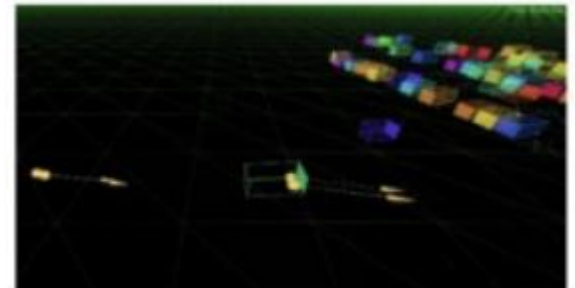
▪ Radar / camera

- 지연 융합



▪ Radar / Lidar

- 지연 융합



3. 위치추정

❖ 위치추정(localization)

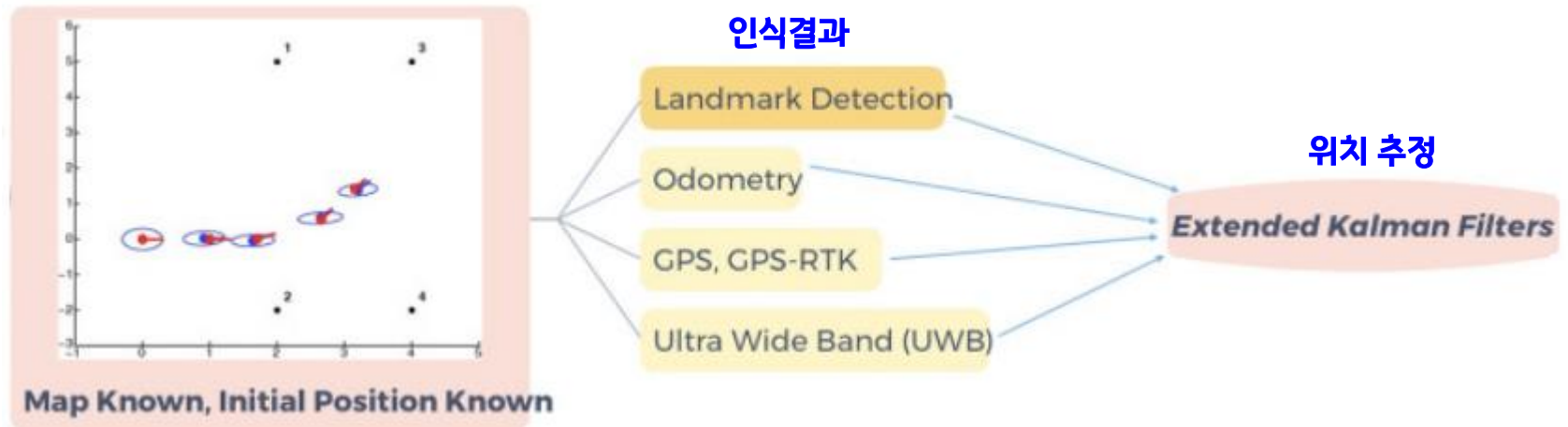
- 환경 내에서 차량 자신의 위치 추정

❖ 위치추정 설정

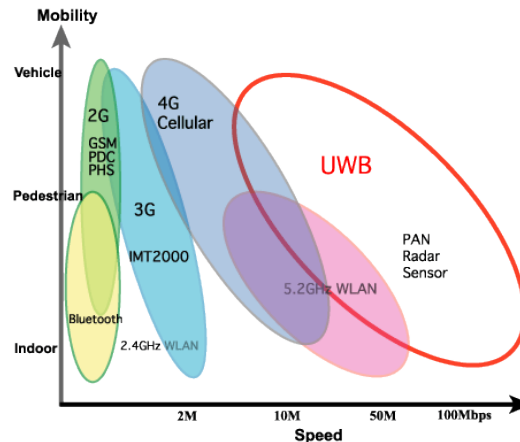
- 지도와 시작위치 제공 상황
- 지도 제공 + 시작위치 미제공 상황
- 지도와 시작위치 미제공 상황
 - Simultaneous Localization And Mapping (SLAM)

위치추정

❖ 지도와 시작위치 제공 위치 추정



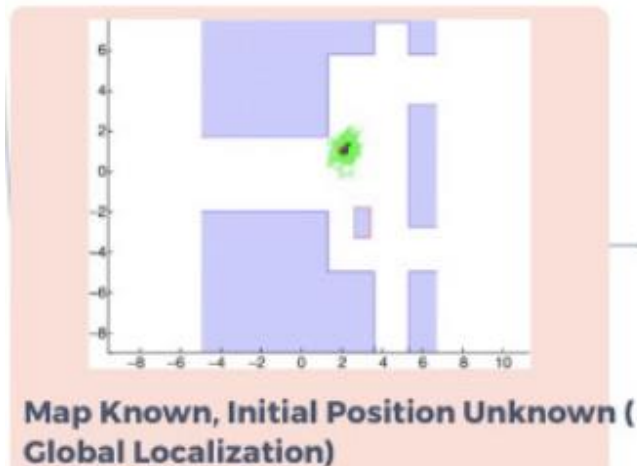
Odometry: 시간에 따른 위치 변화를 추정하기 위해 모션 센서의 데이터(바퀴 회전수 등)를 사용하는 것



UWB(Ultra Wide Band) : 저전력, 고정밀도

위치추정

❖ 지도 제공 + 시작위치 미제공 위치추정

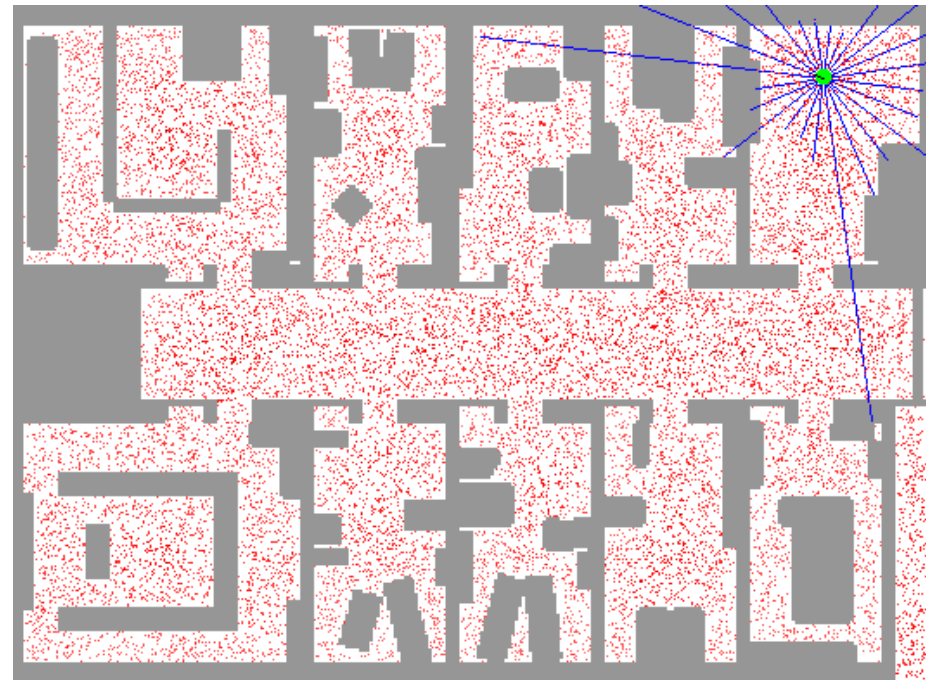


Landmark Detection

Data Association

많은 샘플의 위치 추정 및 수렴 위치 활용

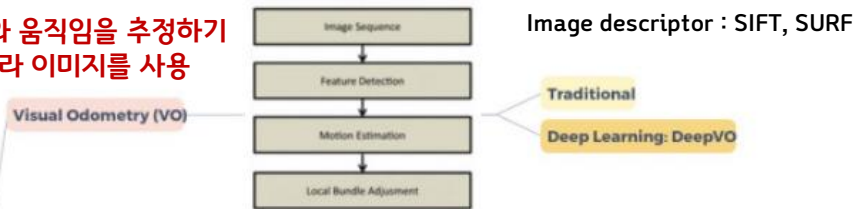
Particle Filters (Monte Carlo Localization)



위치추정

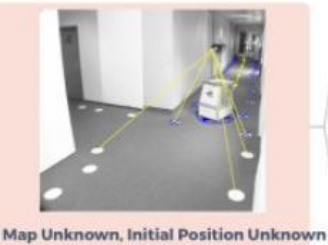
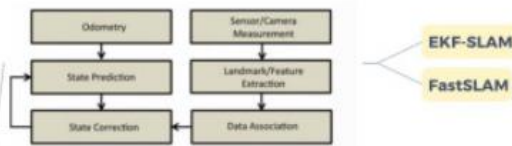
❖ 지도와 시작위치 미제공 위치 추정 (SLAM)

자신의 위치와 움직임을 추정하기
위해 카메라 이미지를 사용



이미지 쌍 매칭, 카메라 모션 추정, 경로 추정

Filter-Based SLAM

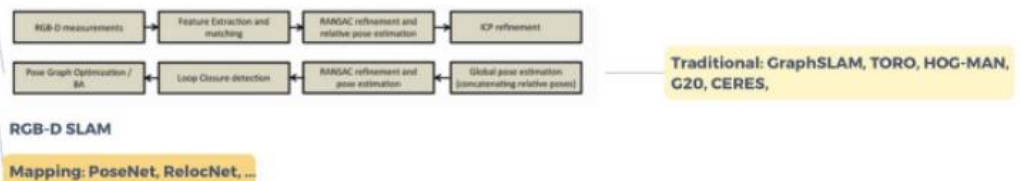
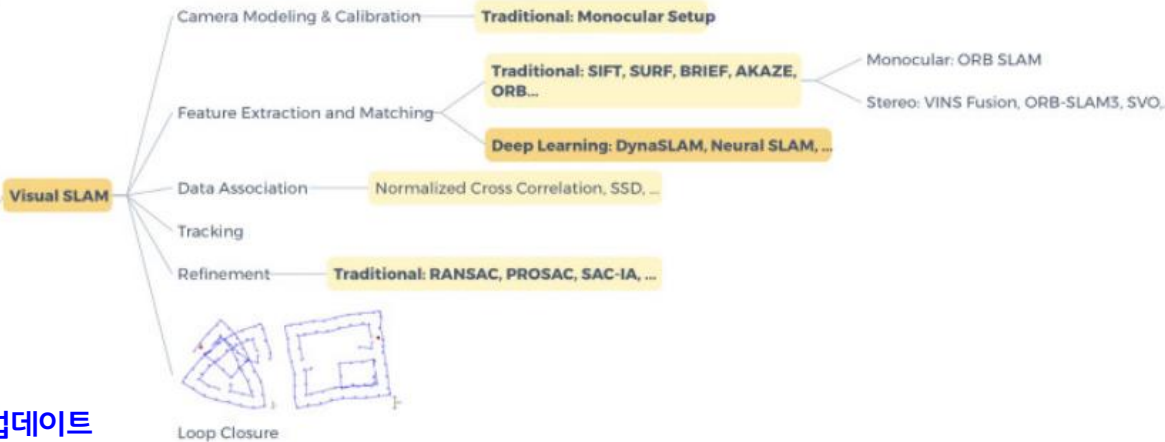


Map Unknown, Initial Position Unknown

Simultaneous Localization And Mapping (SLAM)

미지의 환경에서 자신의 위치를 추정하고
동시에 그 환경의 지도를 작성

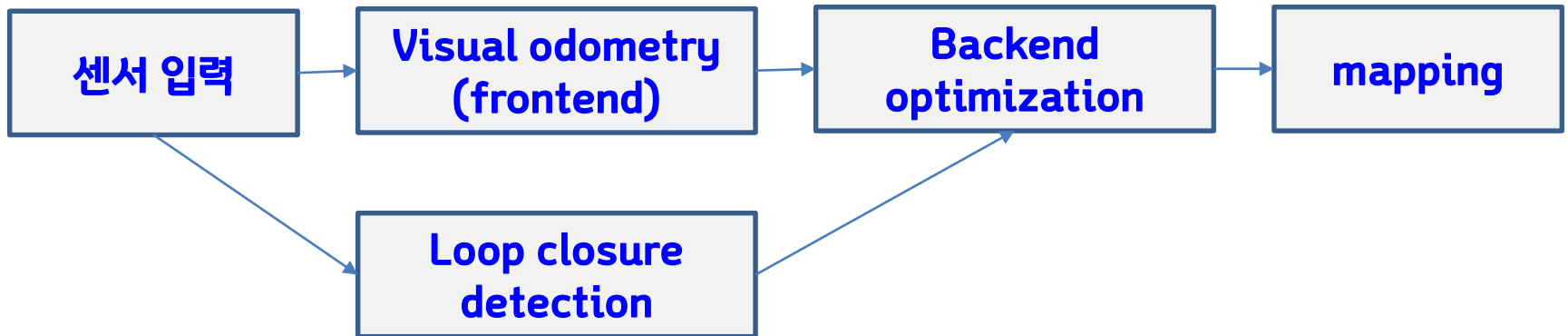
위치 추정, 지도 작성, 위치 업데이트



위치추정

❖ SLAM

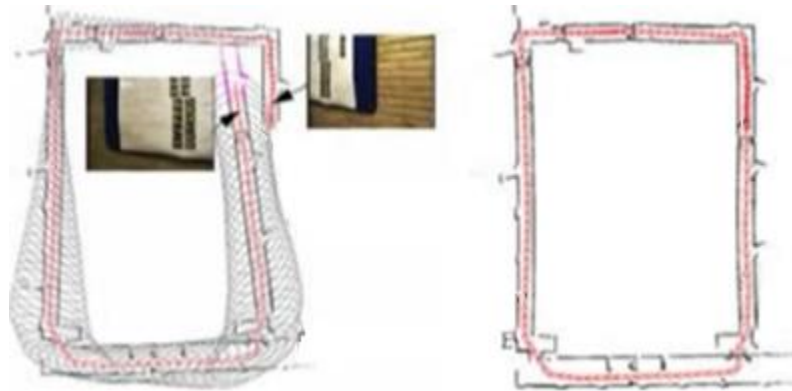
- 전형적인 SLAM의 구조



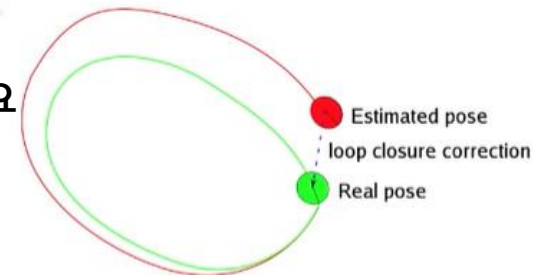
위치추정

❖ Visual Odometry (Frontend)

- 카메라 이미지를 분석하여 로봇의 위치와 방향 결정
- 인접한 이미지 사이의 카메라 움직임 계산
- 카메라 동작에 대한 정량적인 측정 필요
 - 회전 및 이동
- 카메라와 3차원 공간 점 사이의 기하학적 관계 이해 필요
- Drift error 발생
 - Visual odometry 만을 이용하여 궤적을 추정하면 오차 누적



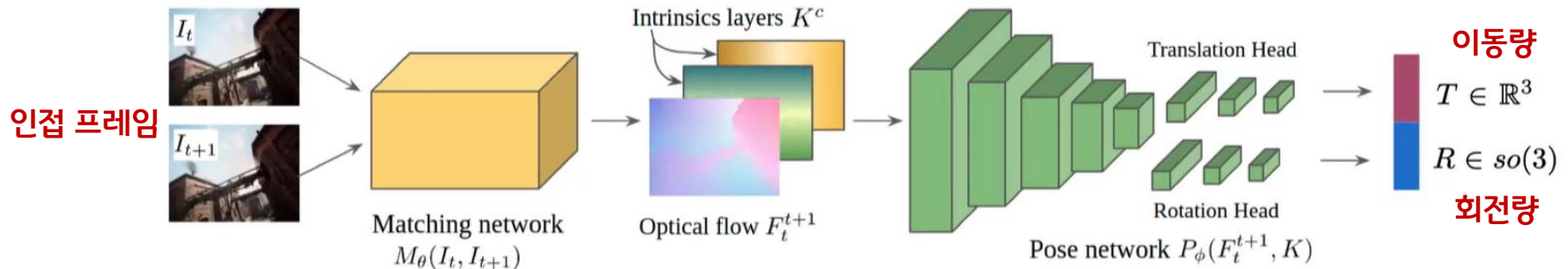
- 해결책
 - Backend optimization과 loop closure detection 필요
 - » 현재 위치가 이전에 방문한 곳인지 판단



위치추정

❖ Visual Odometry (Frontend) – cont.

- Deep learning 모델 적용
 - 인접 시점 이미지 → 이동, 회전 정보 추정



$$L = \lambda L_f + L_p = \lambda \|M_\theta(I_t, I_{t+1}) - F_t^{t+1}\| + \|P_\phi(\hat{F}_t^{t+1}) - \delta_t^{t+1}\| \quad , \text{ where } \delta_t^{t+1} = (T, R)$$

optical flow loss camera motion loss

$$L_p^{cos} = \frac{\hat{T} \cdot T}{\max(\|\hat{T}\| \cdot \|T\|, \epsilon)} + \|\hat{R} - R\|$$

$$L_p^{norm} = \left\| \frac{\hat{T}}{\max(\|\hat{T}\|, \epsilon)} - \frac{T}{\max(\|T\|, \epsilon)} \right\| + \|\hat{R} - R\|$$

위치추정

❖ Visual Odometry (Frontend) – cont.



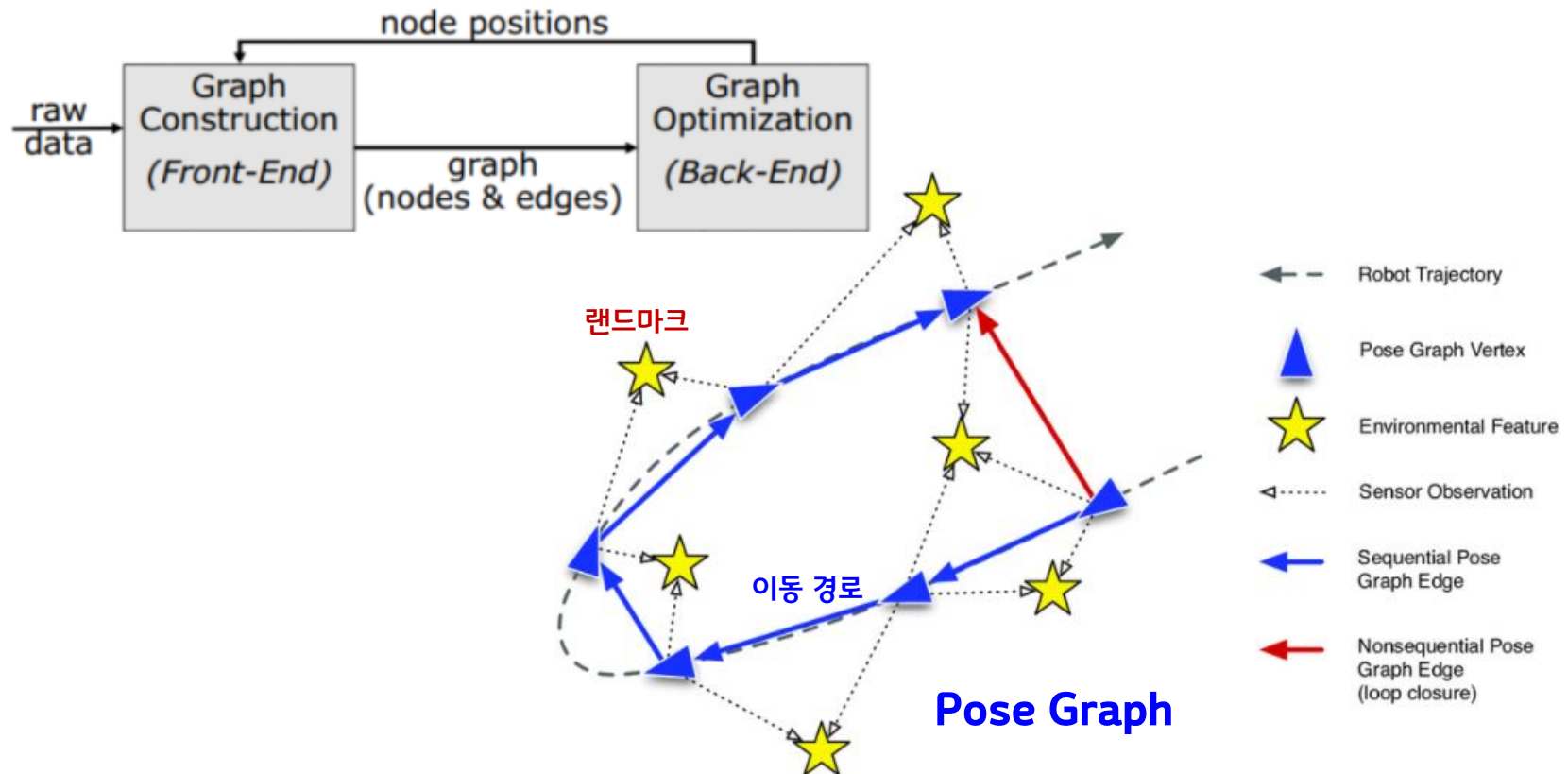
frame	1
key	0
keyframes	1
from start	0.001m
covered	0.000m
inliers	319
outliers	10
time per frame	34ms

FeatureDetectorFast
DescriptorSchemeSAD

위치추정

❖ GraphSLAM

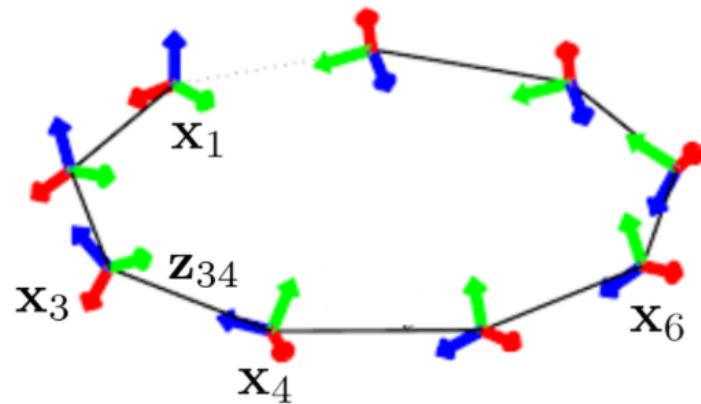
- Front-end : 카메라 등 **센서 입력**으로 받아 **Pose Graph (sub-map)** 생성. **다시 방문한 장소** 인식. **노드 간의 constraint** 생성
- Back-end : Front-end의 시간 경과에 따른 **누적된 오류 최소화(Pose Graph Optimization, PGO)** 를 통해 최적화된 노드 위치 결정.



위치추정

❖ GraphSLAM – cont.

■ Pose Graph

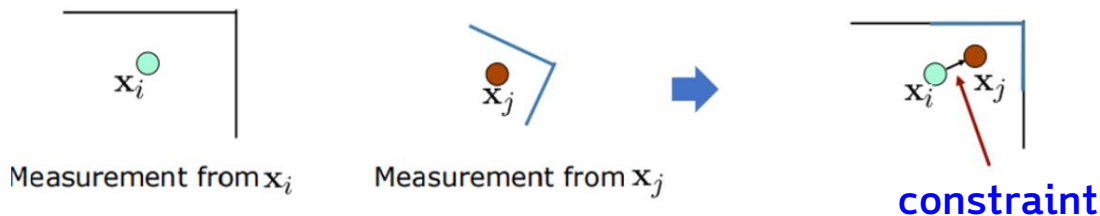


\mathbf{x}_i : i 번째 포즈

$$\begin{aligned} \text{(Node)} \quad \mathbf{x}_i &= [x_i \ y_i \ z_i \ \alpha_i \ \beta_i \ \gamma_i]^\top \\ &= \begin{bmatrix} \mathbf{R}_i & \mathbf{t}_i \\ \mathbf{0} & 1 \end{bmatrix}_{4 \times 4} \end{aligned}$$

\mathbf{z}_{ij} : 노드 i, j 사이의 상대포즈

$$\begin{aligned} \text{(Edge)} \quad \mathbf{z}_{ij} &= [x_{ij} \ y_{ij} \ z_{ij} \ \alpha_{ij} \ \beta_{ij} \ \gamma_{ij}]^\top \\ &= \begin{bmatrix} \mathbf{R}_{ij} & \mathbf{t}_{ij} \\ \mathbf{0} & 1 \end{bmatrix}_{4 \times 4} \end{aligned}$$



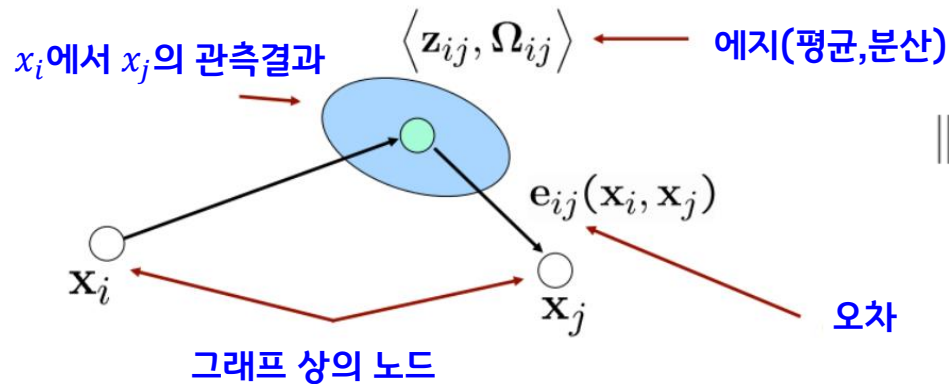
위치추정

❖ GraphSLAM – cont.

▪ Pose Graph Optimization (PGO)

- 오차가 최소화 되도록 모든 노드들의 위치를 최적화하는 과정

$$\mathbf{x}^* = \arg \min_{\mathbf{x}} \|\mathbf{z} - \hat{\mathbf{z}}\|_{\Sigma}^2 = \|\mathbf{e}\|_{\Sigma}^2 \quad \mathbf{z} : \text{관측값}, \hat{\mathbf{z}} : \text{예측값}$$



$$\|\mathbf{e}\|_{\Sigma}^2 = \mathbf{e}^T \Sigma^{-1} \mathbf{e} = \mathbf{e}^T \Omega \mathbf{e}$$

$$\mathbf{x} = [\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_i \quad \cdots \quad \mathbf{x}_n]^T : \text{로봇의 포즈 벡터}$$

$$\mathbf{x}_i = \begin{bmatrix} \mathbf{R}_i & \mathbf{t}_i \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{4 \times 4}$$

위치추정

❖ GraphSLAM – cont.

$$\mathbf{x}^* = \arg \min_{\mathbf{x}} \|\mathbf{z} - \hat{\mathbf{z}}\|_{\Sigma}^2 = \|\mathbf{e}\|_{\Sigma}^2$$

$$\mathbf{x}^* = \arg \min_{\mathbf{x}} \sum_i \|\mathbf{z}_{i,i+1} - \hat{\mathbf{z}}_{i,i+1}\|_{\Sigma_{i,i+1}}^2 + \sum_{i,j} \|\mathbf{z}_{i,j} - \hat{\mathbf{z}}_{i,j}\|_{\Sigma_{i,j}}^2$$

순차적인 노드들 간
관측값과 예측값의 오류

시간 순서와는 상관없는 노드들 간
관측값과 예측값의 오류

$$\mathbf{x}^* = \arg \min_{\mathbf{x}} \sum_i (\mathbf{e}_{i,i+1}^T \Omega_{i,i+1} \mathbf{e}_{i,i+1}) + \sum_{ij} (\mathbf{e}_{ij}^T \Omega_{ij} \mathbf{e}_{ij})$$

오류함수 \mathbf{e} : 회전과 관련된 cos, sin 성분을 포함하여
비선형성 → 비선형 최소제곱법 적용

Gauss-Newton(GN) 또는 Levenberg-Marquardt(LM)

위치추정

❖ GraphSLAM – cont.

▪ Gauss-Newton(GN) 방법

$$\mathbf{e}_{ij}(\mathbf{x} + \Delta\mathbf{x})^\top \Omega_{ij} \mathbf{e}_{ij}(\mathbf{x} + \Delta\mathbf{x})$$

Taylor 전개

$$\begin{aligned}\mathbf{e}_{ij}(\mathbf{x} + \Delta\mathbf{x})|_{\Delta\mathbf{x} \rightarrow \mathbf{x}} &= \mathbf{e}_{ij}(\mathbf{x}) + \mathbf{J}_{ij}(\mathbf{x} + \Delta\mathbf{x} - \mathbf{x}) \\ &= \mathbf{e}_{ij}(\mathbf{x}) + \mathbf{J}_{ij}\Delta\mathbf{x}\end{aligned}$$

$$\mathbf{J}_{ij} = \frac{\partial \mathbf{e}_{ij}(\mathbf{x} + \Delta\mathbf{x})}{\partial \Delta\mathbf{x}}$$

$$\mathbf{e}_{ij}(\mathbf{x} + \Delta\mathbf{x})^\top \Omega_{ij} \mathbf{e}_{ij}(\mathbf{x} + \Delta\mathbf{x}) \simeq (\mathbf{e}_{ij} + \mathbf{J}_{ij}\Delta\mathbf{x})^\top \Omega_{ij} (\mathbf{e}_{ij} + \mathbf{J}_{ij}\Delta\mathbf{x})$$

$$\begin{aligned}&= \underbrace{\mathbf{e}_{ij}^\top \Omega_{ij} \mathbf{e}_{ij}}_{\mathbf{c}_{ij}} + \underbrace{2\mathbf{e}_{ij}^\top \Omega_{ij} \mathbf{J}_{ij} \Delta\mathbf{x}}_{\mathbf{b}_{ij}} + \underbrace{\Delta\mathbf{x}^\top \mathbf{J}_{ij}^\top \Omega_{ij} \mathbf{J}_{ij} \Delta\mathbf{x}}_{\mathbf{H}_{ij}} \\ &= \mathbf{c}_{ij} + 2\mathbf{b}_{ij}\Delta\mathbf{x} + \Delta\mathbf{x}^\top \mathbf{H}_{ij} \Delta\mathbf{x}\end{aligned}$$

$$\mathbf{E}(\mathbf{x} + \Delta\mathbf{x}) = \sum_{ij} \mathbf{e}_{ij}^\top \Omega_{ij} \mathbf{e}_{ij} = \mathbf{c} + 2\mathbf{b}\Delta\mathbf{x} + \Delta\mathbf{x}^\top \mathbf{H} \Delta\mathbf{x}$$

$$\mathbf{H} = \mathbf{J}^\top \Omega \mathbf{J}$$

위치추정

❖ GraphSLAM – cont.

▪ Gauss-Newton(GN) 방법 – cont.

$$\mathbf{E}(\mathbf{x} + \Delta\mathbf{x}) = \sum_{ij} \mathbf{e}_{ij}^T \Omega_{ij} \mathbf{e}_{ij} = \mathbf{c} + 2\mathbf{b}\Delta\mathbf{x} + \Delta\mathbf{x}^T \mathbf{H} \Delta\mathbf{x}$$

- $E(x + \Delta x)$ 은 Δx 에 대한 **2차식**(Quadratic) 형태이고, $H = J^T \Omega J$ 는 positive definite 행렬이므로 $E(x + \Delta x)$ 를 **1차 미분**하여 **0**으로 설정한 값이 Δx 의 극소

$$\frac{\partial \mathbf{E}(\mathbf{x} + \Delta\mathbf{x})}{\partial \Delta\mathbf{x}} \simeq 2\mathbf{b} + 2\mathbf{H}\Delta\mathbf{x} = 0$$

$$\mathbf{H}\Delta\mathbf{x} = -\mathbf{b}$$

$$\Delta\mathbf{x} = -\mathbf{H}^{-1}\mathbf{b}$$

$$\mathbf{x} \leftarrow \mathbf{x} + \Delta\mathbf{x}$$

▪ Levenberg-Marquardt(LM) 방법

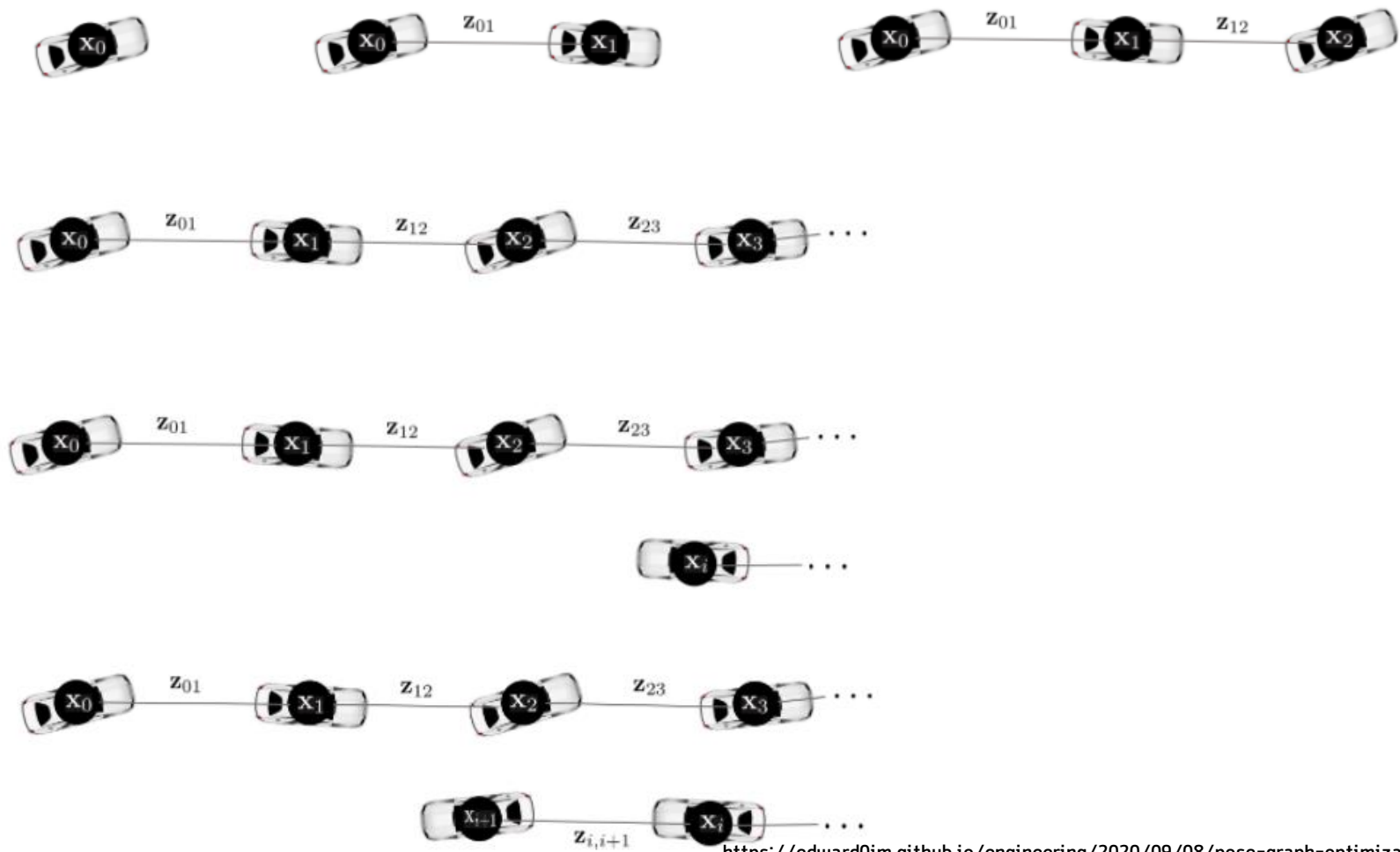
- damping factor λ 항 추가

$$(\mathbf{H} + \lambda \mathbf{I})\Delta\mathbf{x} = -\mathbf{b}$$

위치추정

❖ GraphSLAM – cont.

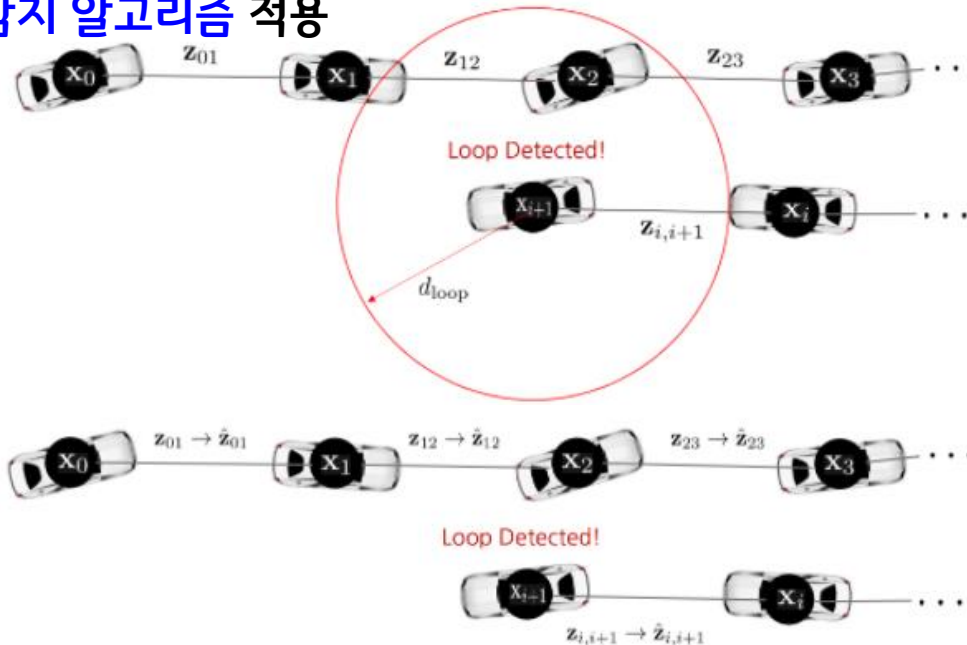
▪ Front-end 알고리즘



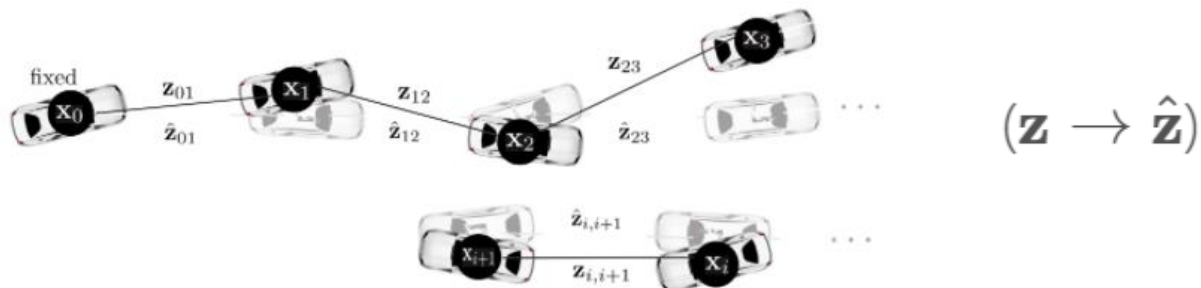
위치추정

❖ GraphSLAM – cont.

▪ loop 감지 알고리즘 적용



- **Loop closure 감지**: Front-end에서 **GICP**(generalized iterative closest point)를 통해 새로운 관측값 z_{ij} 를 구하고, 기존의 값들은 예측값 \hat{z}_{ij} 으로 전환

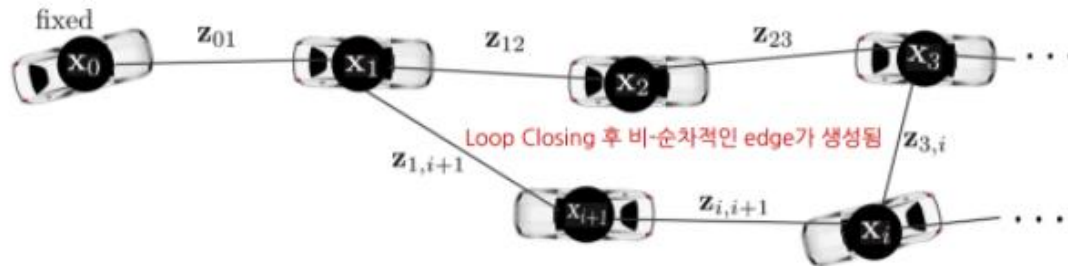


위치추정

❖ GraphSLAM – cont.

- 관측값과 예측값의 차이를 오차 e_{ij} 로 설정
- 관측값과 예측값은 상대포즈(Relative Pose)로 나타내므로, 둘의 차이는 $z_{ij} - \hat{z}_{ij}$ 이 아닌 $z_{ij}^{-1}\hat{z}_{ij}$

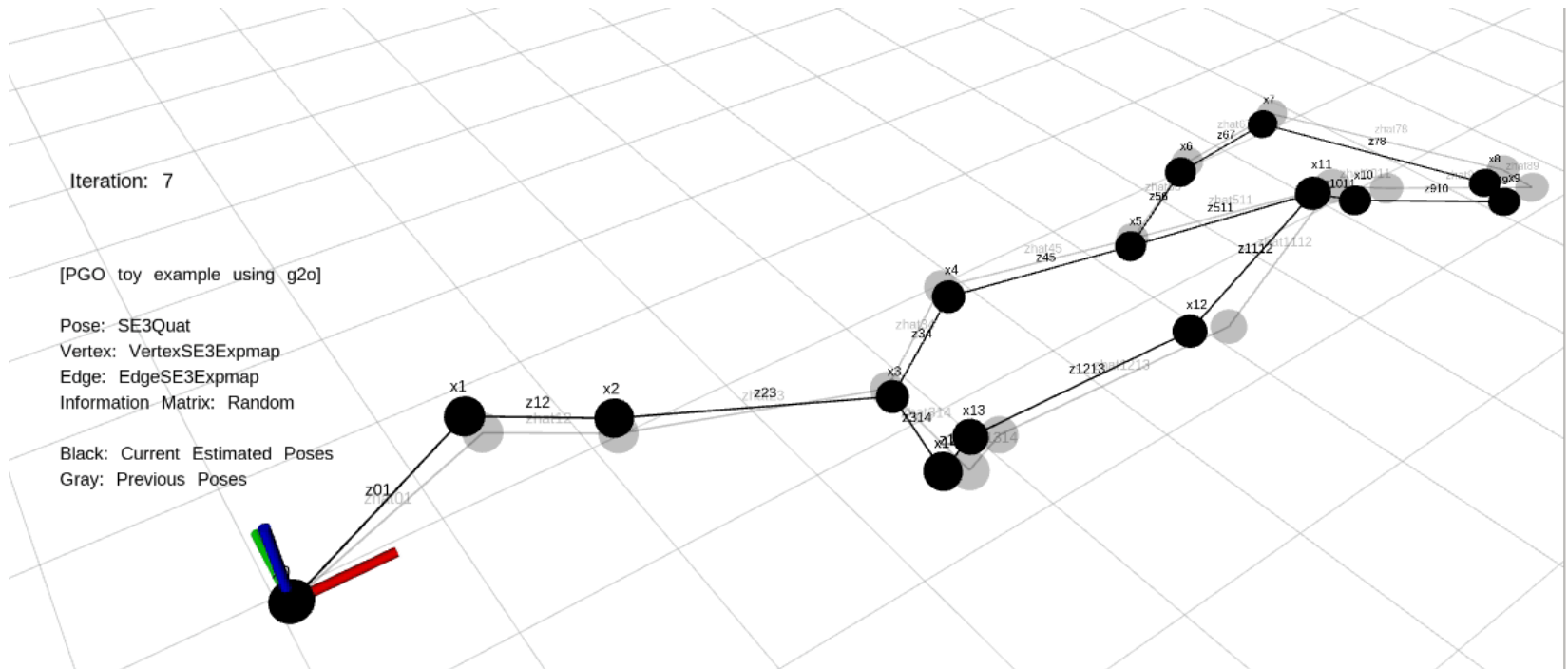
$$e_{ij} = z_{ij}^{-1}\hat{z}_{ij}$$



- 모든 노드의 오차가 최소화되는 차량의 포즈 x 계산
 - 비선형 최소제곱법(GN, LM)을 통해 반복적으로 오차가 최소가 되는 포즈 갱신

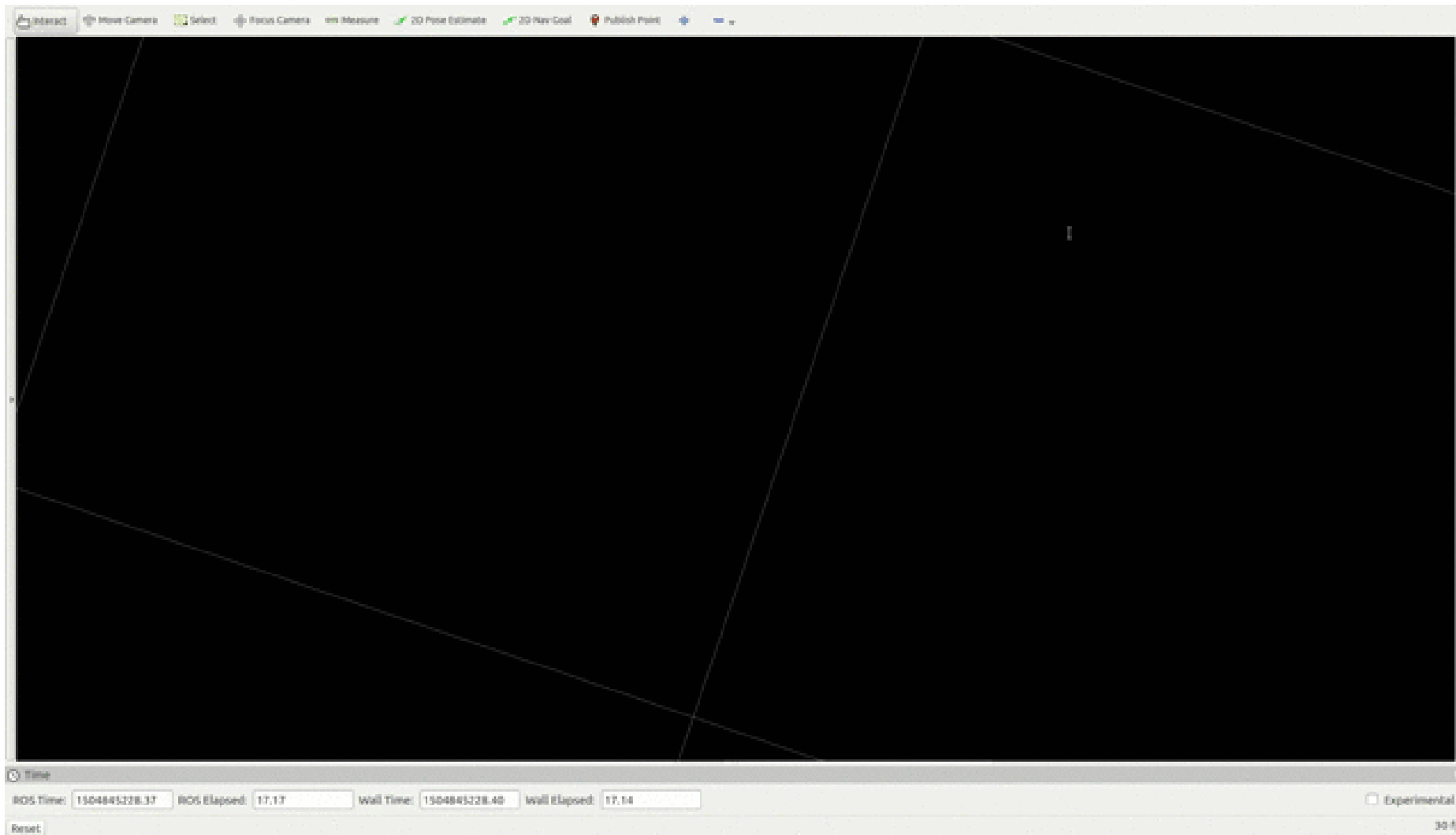
위치추정

❖ GraphSLAM – cont.



위치추정

❖ GraphSLAM – cont.



4. 계획 수립

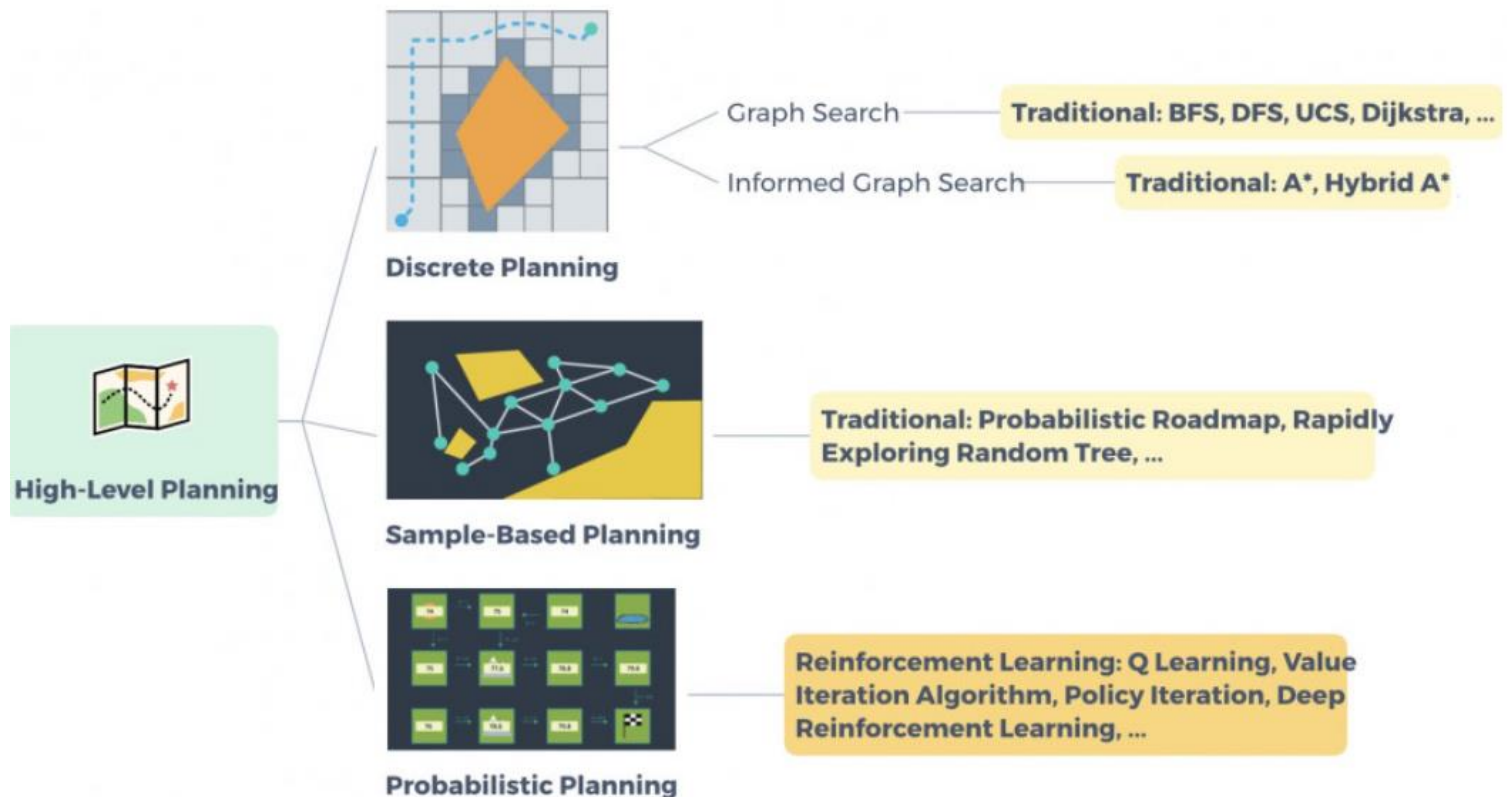
❖ 자율주행의 계획 수립 (planning)

- 상위/전역(high-level/global) 계획수립
 - 출발지에서 목적지로의 길(route) 결정
- 행동(behavioral) 계획수립
 - 타 차량, 보행자 등 장애물이 어떤 행동을 할 지 예측하고, 의사결정을 하는 것
- 경로/지역(path/local) 계획 수립
 - 장애물을 회피한 궤적(trajjectory) 생성

계획 수립

❖ 상위/전역(high-level/global) 계획수립

- **그래프 탐색 알고리즘** 적용 가능
 - A*, Dijkstra, DFS, BFS, ...
- **확률적(probabilistic) 계획수립**
 - 강화학습 적용



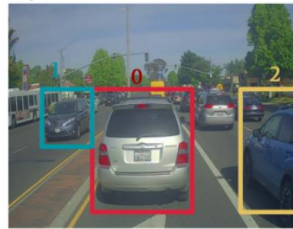
계획 수립

❖ 행동(behavioral) 계획수립

- **의도 예측** (intention prediction)
 - 객체 추적(tracking)
 - Hungarian Algorithm : 다중 객체 추적
 - » 프레임 간 경계상자 매칭



t=0



t=1

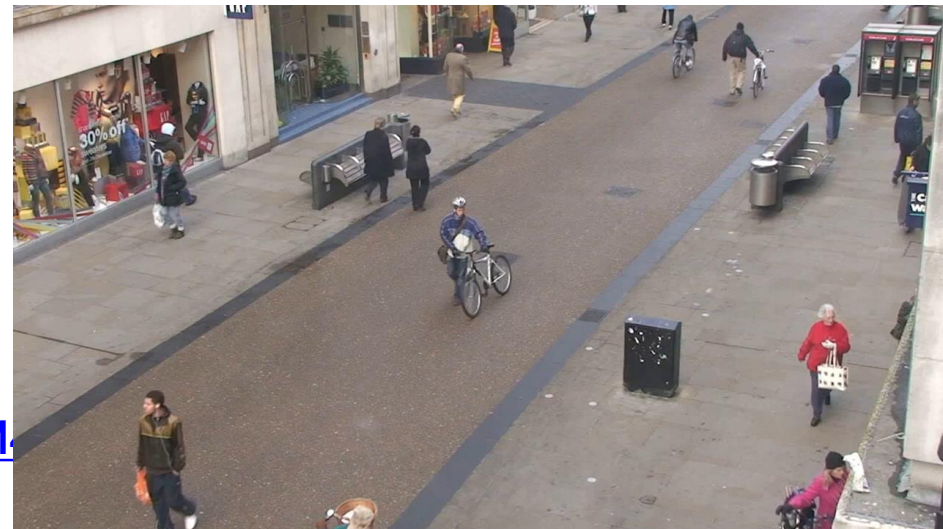


t=2

- 타 객체의 이동 예측
 - Kalman 필터 적용

- **의사결정**
 - 상황에 따른 행동 규칙 직접 제공
 - 강화학습을 통한 정책 학습

<https://youtu.be/bkn6M>

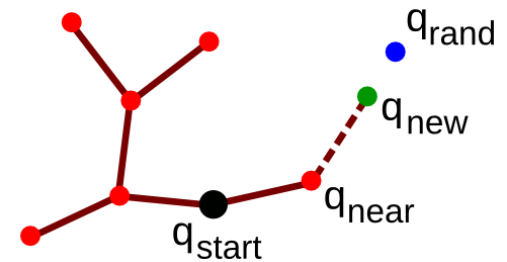
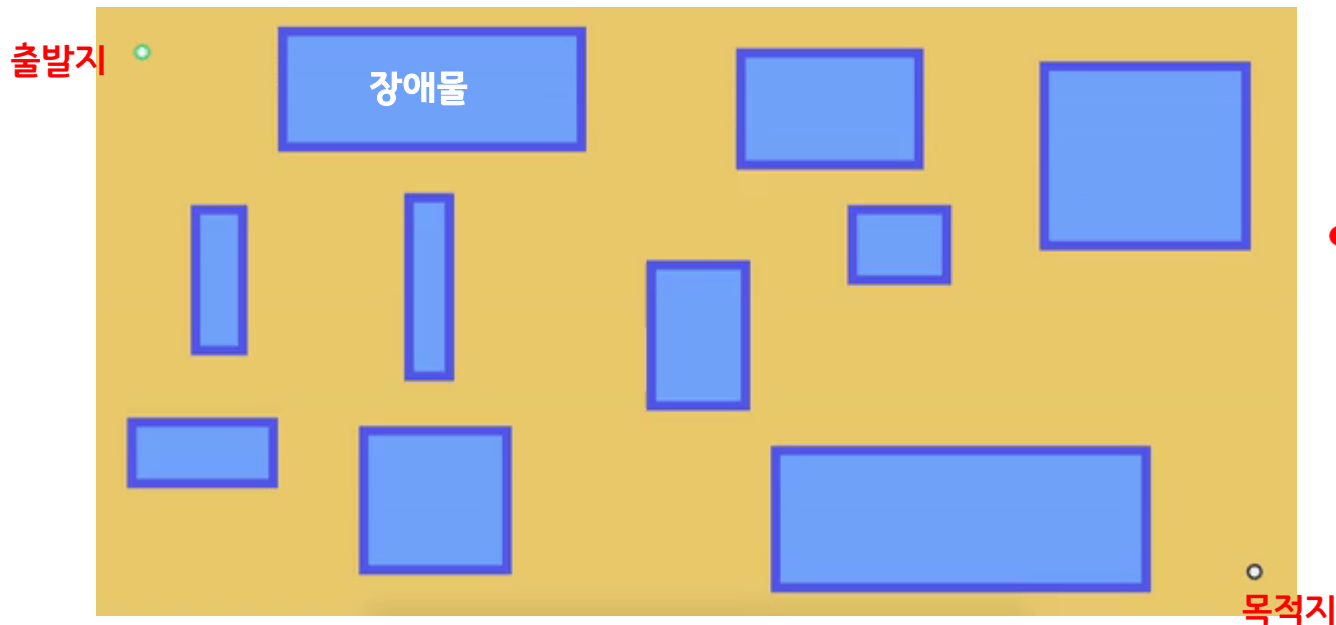


계획 수립

❖ 경로/지역(path/local) 계획 수립

- 장애물을 회피한 궤적(trajjectory) 생성
- Rapidly-exploring Random Trees (RRT), RRT*, Probabilistic RoadMaps (PRM), PRM*, ...

Rapidly-exploring Random Trees (RRT)

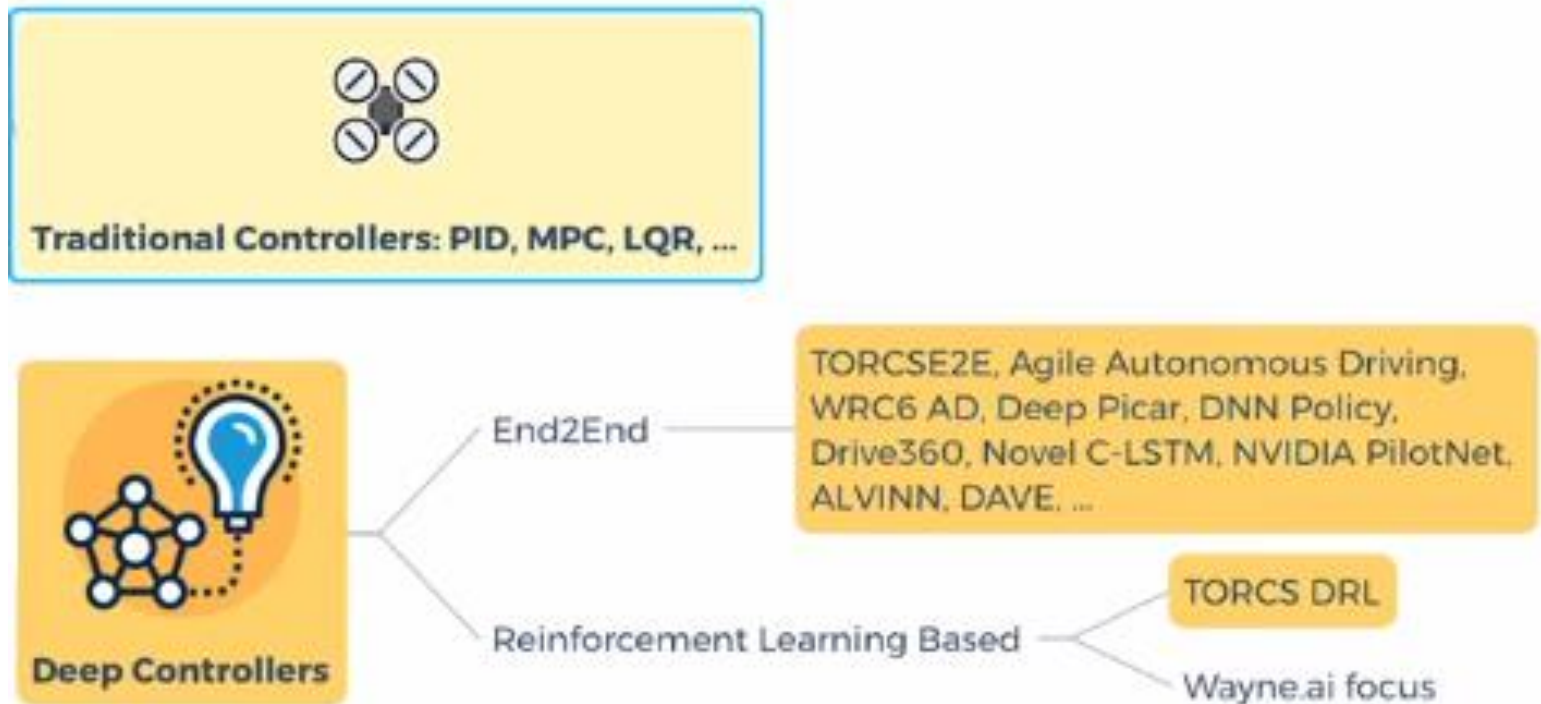


- q_{rand} : 무작위 선택 위치
- q_{near} : q_{rand} 의 최근접 노드(기존)
- q_{new} : 추가되는 노드
(q_{near} 거리 r 이내)
- q_{start} : 출발 위치

5. 제어

❖ 제어(control)

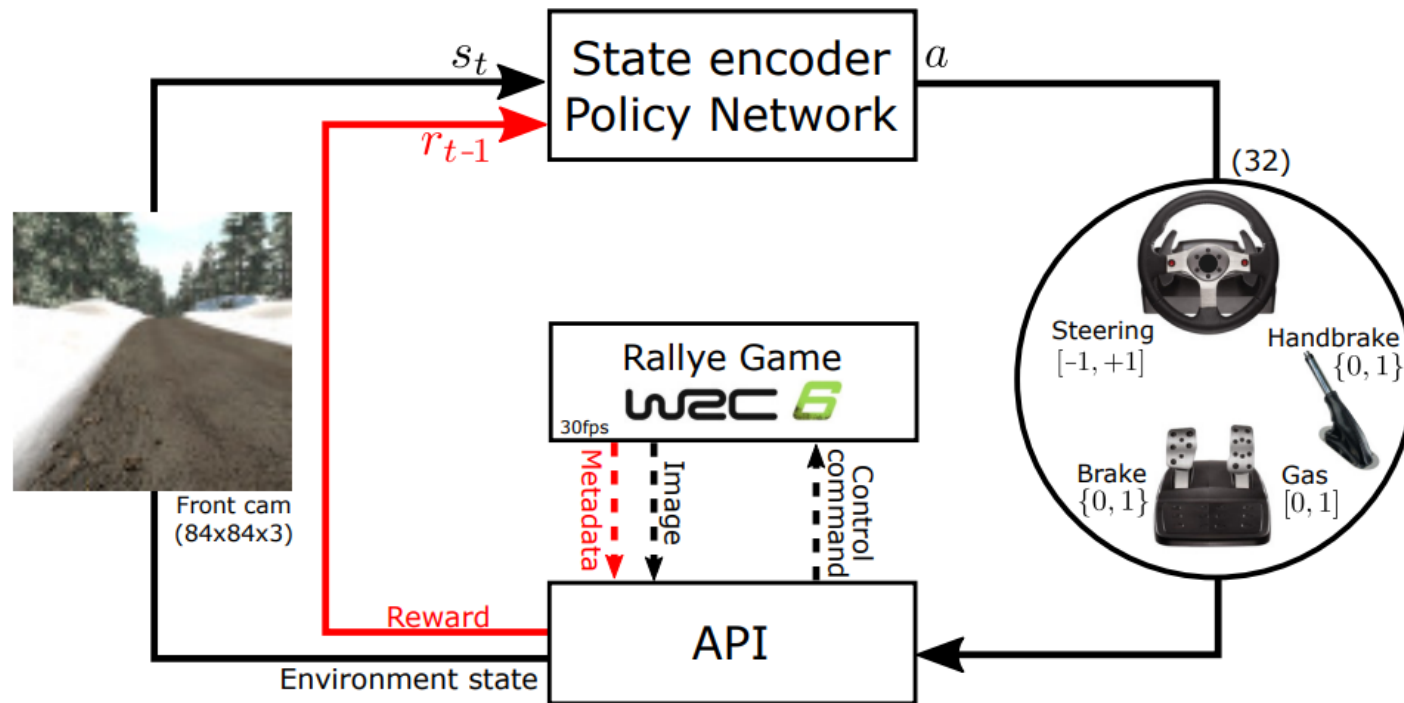
- 궤적을 따라 이동하기 위해 **조향** 및 **가속정보** 생성



제어

❖ 종단간(end-to-end) 주행 학습

- 인식단계 등이 없이 입력 영상 등으로 부터 주행 제어
- 강화학습 적용 사례



Quiz

❖ 자율주행에 대한 설명으로 옳지 않은 것은?

- ① 포인트 클라우드(point cloud) 데이터는 3차원 좌표들에 대한 정보를 포함한다.
- ② 포인트 클라우드 데이터는 순서(permutation) 불변이며 변환(transformation) 불변인 연산 사용하여 처리하는 것이 바람직하다.
- ③ 스테레오 비전은 하나의 사진으로부터 물체까지의 거리를 측정하는 기술을 의미한다.
- ④ 레이더 데이터는 각도, 거리, 도플러(Doppler) 정보를 포함한다.

❖ 센서 융합에 대한 설명으로 옳지 않은 것은?

- ① 여러 센서의 데이터를 결합하여 정확도 높은 추정을 하는 것이 센서 융합이다.
- ② 융합 시점에 따라 초기 융합(early fusion)과 지연 융합(later fusion)으로 나눌 수 있다.
- ③ 원 데이터를 처리하는 시작단계에 결합하는 것을 초기 융합이라고 한다.
- ④ 레이더와 카메라 데이터에 대한 융합은 일반적으로 초기 융합을 한다.

❖ 자율 주행 차량의 SAE 레벨 0은?

- ① 완전한 자동화
- ② 공유 제어
- ③ 자동화 없음
- ④ 선택적 운전

Quiz

❖ 자율 주행 차량 인식에 일반적으로 사용되지 않는 센서는?

- ① 카메라
- ② LiDAR
- ③ 레이더
- ④ 음향 센서

❖ 3D 객체 감지에서 RGB와 함께 사용되는 데이터 유형은?

- ① 초음파 데이터
- ② 적외선 데이터
- ③ 포인트 클라우드 데이터
- ④ 열화상 데이터

❖ 자율 주행 차량의 레이더 센서에서 제공되지 않는 데이터 유형은?

- ① 거리
- ② 방위각
- ③ 색상
- ④ 도플러

Quiz

❖ 자율 주행 차량에서 차선 감지에 현재 가장 적합한 방식은?

- ① GPS 매핑
- ② U-Net 기반 감지
- ③ 오디오 분석
- ④ 열화상 이미지

❖ LaneNet 네트워크는 주로 어떤 것에 중점을 두는가?

- ① 교통 신호 감지
- ② 차선 감지
- ③ 차량 속도 추정
- ④ 날씨 예측

❖ 자율 주행에서 객체 추적의 주요 목적은?

- ① 날씨 조건 예측
- ② 시간에 따라 동일한 객체의 지속적 추적
- ③ 도로 표지판 감지
- ④ 차량 속도 추정

Quiz

❖ 자율 주행의 장면 인식에서 사용되는 주요 방식은?

- ① 의미 분할
- ② 색상 감지
- ③ 소리 분석
- ④ 온도 측정

❖ 자율주행차의 주요 핵심 요소가 아닌 것은?

- ① 인식
- ② 위치 추정
- ③ 계획 수립
- ④ 운전자 감정 분석

❖ 자율주행차에서 '인식' 단계에서 주로 수행되는 작업은?

- ① 궤적 생성
- ② 위치 추정
- ③ 주변 환경 및 장애물 파악
- ④ 조향 및 가속 정보 생성

Quiz

❖ 자율주행차의 '계획 수립' 단계에서 주로 고려되는 요소는?

- ① 주변 환경 인식
- ② 속도 제어
- ③ 원하는 위치로 이동을 위한 궤적 생성
- ④ 배터리 수명 관리

❖ 자율주행차의 '제어' 단계에서 주로 수행되는 작업은?

- ① 주변 환경 인식
- ② 위치 추정
- ③ 궤적 따라 이동을 위한 조향 및 가속 정보 생성
- ④ 목적지 설정

❖ 자율주행차에서 '라이다' 데이터를 처리하는데 적합한 알고리즘은?

- ① YOLO
- ② SSD
- ③ PointNet
- ④ U-Net

Quiz

❖ 자율주행차에서 '레이더' 센서가 제공하는 정보는?

- ① 색상 인식
- ② 3D 이미지
- ③ 거리, 각도, 도플러(Doppler) 정보
- ④ 온도 측정

❖ 자율주행차에서 '스테레오 비전'의 주요 목적은?

- ① 속도 측정
- ② 물체까지의 거리 추정
- ③ 음성 인식
- ④ 내부 온도 조절

❖ 자율주행차의 SAE 레벨 중 운전자의 주의가 필요 없고 핸들이 선택적인 단계는?

- ① Level 3
- ② Level 4
- ③ Level 5
- ④ Level 2

Quiz

❖ 다음 중 자율주행차의 인식 센서가 아닌 것은?

- ① 카메라
- ② Lidar
- ③ Radar
- ④ GPS

❖ 자율주행차에서 2D 물체 감지를 위해 사용되는 기술은?

- ① YOLO
- ② GPS
- ③ RRT
- ④ SLAM

❖ 다음 중 자율주행차에서 사용되는 3D 물체 감지 알고리즘은?

- ① YOLO
- ② Frustum PointNet
- ③ U-Net
- ④ A*

Quiz

❖ PointNet 알고리즘의 특징 중 하나는?

- ① 순서 불변성
- ② 지리적 정확성
- ③ 시간적 연속성
- ④ 색상 인식 능력

❖ 자율주행차의 위치추정 단계에서 사용되는 기술은?

- ① SLAM (Simultaneous Localization And Mapping)
- ② GPS (Global Positioning System)
- ③ LiDAR (Light Detection and Ranging)
- ④ Radar (Radio Detection and Ranging)

❖ 자율주행차의 객체 추적(Object Tracking) 단계에서 많이 사용되는 알고리즘은?

- ① Hungarian 알고리즘
- ② Kalman 필터
- ③ Deep Learning 알고리즘
- ④ Hungarian 알고리즘과 Kalman 필터

Quiz

- ❖ 자율주행차에서 '학습 기반 접근법(learning-based approach)'이란?
 - ① 차량의 오디오 시스템 학습
 - ② 차량의 배터리 수명 최적화
 - ③ 데이터를 통한 의사 결정 알고리즘 개선
 - ④ 차량의 색상 변화 학습

- ❖ 자율주행차에서 '환경 매핑(environmental mapping)'의 중요성은?
 - ① 차량의 오디오 시스템 개선
 - ② 차량 주변 환경의 정확한 3D 모델 생성
 - ③ 차량의 연료 효율 향상
 - ④ 차량 내부의 온도 조절

- ❖ 자율주행차에서 '경로 계획 알고리즘(Path Planning Algorithm)'의 주요 목적은?
 - ① 차량의 오디오 시스템 조정
 - ② 최적의 경로 결정 및 탐색
 - ③ 차량의 색상 변경
 - ④ 차량의 연료 효율 향상

Quiz

❖ Hungarian 알고리즘은 주로 어떤 문제에 사용되는가?

- ① 최단 경로 문제
- ② 네트워크 플로우 문제
- ③ 다중 객체 추적 문제
- ④ 그래프 색칠 문제

❖ Hungarian 알고리즘은 어떤 유형의 데이터에 적용되는가?

- ① 시계열 데이터
- ② 이미지 데이터
- ③ 텍스트 데이터
- ④ 비용 행렬 데이터

❖ Kalman filter를 사용하는 이유는?

- ① 데이터 저장 공간 최적화
- ② 데이터 전송 속도 향상
- ③ 추적 대상의 행동 예측 및 가림(occlusion) 대응
- ④ 사용자 인터페이스 개선

Quiz

❖ 자율주행차에서 '통합 제어 시스템(Integrated Control System)'의 핵심 기능은?

- ① 차량의 색상 변경
- ② 차량 내부의 오디오 시스템 관리
- ③ 다양한 시스템과 센서의 조정 및 통합 제어
- ④ 차량의 연료 효율 관리

❖ GraphSLAM에서 Front-end의 역할은?

- ① 카메라 등 센서 입력으로 Pose Graph (sub-map)를 생성하고, 다시 방문한 장소를 인식하며, 노드 간의 제약을 생성한다.
- ② 오차를 최소화하기 위해 모든 노드들의 위치를 최적화한다.
- ③ Gauss-Newton 방법을 사용하여 비선형 최소제곱 문제를 해결한다.
- ④ damping factor를 추가하여 Levenberg-Marquardt 방법을 적용한다.

❖ GraphSLAM에서 Back-end의 주요 기능은?

- ① 카메라 등 센서 입력으로 Pose Graph를 생성한다.
- ② Front-end의 시간 경과에 따른 누적된 오류를 최소화하여 최적화된 노드 위치를 결정한다.
- ③ Levenberg-Marquardt 방법을 사용하여 최적화를 수행한다.
- ④ Pose Graph를 생성하여 장소를 인식한다.

Quiz

❖ Pose Graph Optimization (PGO)의 목적은?

- ① 카메라와 다른 센서로부터 데이터를 수집한다.
- ② Pose Graph를 통해 장소를 인식한다.
- ③ 오차가 최소화되도록 모든 노드들의 위치를 최적화한다.
- ④ damping factor를 추가하여 최적화 문제를 해결한다.

❖ GraphSLAM에서 Gauss-Newton 방법의 역할은?

- ① Pose Graph를 생성한다.
- ② Front-end의 오류를 최소화한다.
- ③ 모든 노드들의 위치를 최적화한다.
- ④ 비선형 최소제곱 문제를 해결하기 위해 사용된다.

❖ Levenberg-Marquardt 방법이 GraphSLAM에서 사용되는 이유는?

- ① 카메라 입력으로 Pose Graph를 생성하기 위해.
- ② Pose Graph를 최적화하기 위해.
- ③ damping factor를 추가하여 최적화 문제를 더 효과적으로 해결하기 위해.
- ④ Front-end에서 장소를 인식하기 위해.

Quiz

❖ 자율주행에서 계획수립(planning)의 주요 목적은?

- ① 차량의 위치를 추정한다.
- ② 주변 환경과 장애물을 인식한다.
- ③ 인식 내용 및 추정 위치를 토대로 원하는 위치로 이동을 위한 궤적을 생성한다.
- ④ 장애물을 탐지하고 분류한다.

❖ 자율주행에서 상위/전역(high-level/global) 계획수립에서 고려해야 할 주요 요소는?

- ① 차량의 속도와 방향을 조절한다.
- ② 출발지에서 목적지로의 길(route)을 결정한다.
- ③ 자동차와 보행자의 행동을 예측한다.
- ④ 실시간으로 위치를 추적한다.

❖ 자율주행에서 행동(behavioral) 계획수립의 핵심은?

- ① 출발지에서 목적지까지의 최적 경로를 계획한다.
- ② 타 차량, 보행자 등 장애물이 어떤 행동을 할지 예측하고 의사결정을 한다.
- ③ 센서 데이터를 분석하여 장애물을 탐지한다.
- ④ 주변 환경을 3D 지도로 구성한다.

Quiz

❖ 자율주행에서 경로/지역(path/local) 계획 수립의 주요 목적은?

- ① 차량의 실시간 위치를 정확하게 추적한다.
- ② 센서 데이터를 분석하여 주변 환경을 인식한다.
- ③ 장애물을 회피한 궤적(trajjectory)을 생성한다.
- ④ 차량과 보행자 사이의 거리를 유지한다.

❖ 자율주행에서 상위/전역 계획수립에 사용되는 기술은?

- ① 강화학습을 적용하여 확률적 계획수립을 한다.
- ② LiDAR 데이터를 사용하여 장애물을 탐지한다.
- ③ Kalman 필터를 사용하여 타 차량의 이동을 예측한다.
- ④ Rapidly-exploring Random Trees (RRT)를 사용하여 경로를 계획한다.

❖ Visual Odometry에서 카메라 이미지 분석을 통해 결정되는 것은?

- ① 로봇의 속도와 가속도
- ② 로봇의 위치와 방향
- ③ 카메라의 해상도와 감도
- ④ 로봇의 온도와 배터리 수준

Quiz

❖ **Visual Odometry에서 인접한 이미지 사이의 카메라 움직임을 계산하는 이유는?**

- ① 이미지의 화질을 향상시키기 위해
- ② 카메라의 위치 변화와 로봇의 이동 경로를 추정하기 위해
- ③ 데이터 전송 속도를 최적화하기 위해
- ④ 이미지 저장 공간을 최소화하기 위해

❖ **Visual Odometry를 사용하여 궤적을 추정할 때 발생할 수 있는 문제는?**

- ① 과속 감지
- ② 배터리 소모 증가
- ③ Drift error의 누적
- ④ 이미지의 해상도 감소

❖ **Visual Odometry에서 drift error를 해결하기 위해 필요한 것은?**

- ① 더 높은 해상도의 카메라
- ② Backend optimization과 loop closure detection
- ③ 데이터 압축 알고리즘
- ④ 강화학습 모델

Quiz

❖ Visual Odometry에서 deep learning 모델이 적용되는 부분은?

- ① 카메라의 실시간 이미지 향상을 위해
- ② 이미지 저장 알고리즘의 최적화를 위해
- ③ 인접 시점 이미지로부터 이동과 회전 정보를 추정하기 위해
- ④ 센서 데이터의 전처리를 위해

❖ 센서 융합(sensor fusion)이란?

- ① 여러 센서의 데이터를 결합하여 정확도 높은 추정을 하는 것
- ② 단일 센서의 데이터만을 사용하여 분석을 수행하는 것
- ③ 센서의 오류를 수정하기 위해 추가적인 센서를 개발하는 것
- ④ 센서 데이터를 클라우드에 저장하는 과정

❖ 초기 융합(early fusion)이란?

- ① 센서들 사이에서 데이터를 교환하는 과정
- ② 센서의 데이터를 시간에 따라 결합하는 방법
- ③ 원 데이터의 결합, 예를 들어 Lidar point cloud와 이미지의 픽셀의 결합
- ④ 센서 데이터를 머신 러닝 모델에 입력하는 것

Quiz

❖ Lidar와 Camera를 사용한 센서 융합에서 가능한 융합 형태는?

- ① 오직 초기 융합만이 가능하다.
- ② 오직 지연 융합만이 가능하다.
- ③ 초기 융합과 지연 융합 둘 다 가능하다.
- ④ 융합이 불가능하다.

❖ Lidar와 Stereo camera를 사용한 센서 융합에서 적용되는 융합 방식은?

- ① 초기 융합
- ② 지연 융합
- ③ 데이터 병합
- ④ 동시 융합

❖ 센서 융합을 통해 차량이 수행할 수 있는 기능은?

- ① 환경 내에서 차량 자신의 위치 추정
- ② 센서의 파손 여부 판단
- ③ 다른 차량의 연비 계산
- ④ 기상 조건 예측

Quiz

❖ 종단간 주행 학습이란?

- ① 인식단계를 포함한 여러 단계를 거쳐 주행을 제어하는 것.
- ② 인식단계 등이 없이 입력 영상 등으로부터 주행 제어를 하는 것.
- ③ 오직 Lidar 센서만을 이용하여 주행을 제어하는 것.
- ④ 주행 중 발생하는 모든 데이터를 저장하는 것.

❖ 종단간 주행 학습에서 인식단계의 제외의 의미는?

- ① 주행 중에는 인식을 수행하지 않는다는 것.
- ② 인식단계를 별도로 구축하지 않고, 입력으로부터 바로 주행 제어 정보를 추론한다는 것.
- ③ 인식과 관련된 센서가 고장 났을 때의 주행 방법을 의미한다.
- ④ 인식단계를 다른 차량과 공유하지 않는다는 것.

❖ 종단간 주행 학습에서의 주행 제어 정보는 어떻게 결정되는가?

- ① 전통적인 알고리즘에 의해 결정된다.
- ② 전문가의 판단에 의해 수동으로 조정된다.
- ③ 입력 데이터로부터 직접 추론하여 결정된다.
- ④ 사전에 프로그램된 경로를 따라 자동으로 결정된다.