Journal of Korea Robotics Society (2022) 17(2):198-208
https://doi.org/10.7746/jkros.2022.17.2.198 ISSN: 1975-6291 / eISSN: 2287-3961

세계 AI 로봇 카레이스 대회를 위한 자율 주행 시스템 구현

Implementation of an Autonomous Driving System for the Segye AI Robot Car Race Competition

최정현[1*]·임예은[1*]·박종훈[2]·정현수[2]·변승재[2]·사공의훈[1]·
박정현[1]·김창현[1]·이재찬[1]·김도형[1]·황면중[†]
Jung Hyun Choi[1*], Ye Eun Lim[1*], Jong Hoon Park[2], Hyeon Soo Jeong[2],
Seung Jae Byun[2], Ui Hun Sagong[1], Jeong Hyun Park[1], Chang Hyun Kim[1],
Jae Chan Lee[1], Do Hyeong Kim[1], Myun Joong Hwang[†]

Abstract: In this paper, an autonomous driving system is implemented for the Segye AI Robot Race Competition that multiple vehicles drive simultaneously. By utilizing the ERP42-racing platform, RTK-GPS, and LiDAR sensors provided in the competition, we propose an autonomous driving system that can drive safely and quickly in a road environment with multiple vehicles. This system consists of a recognition, judgement, and control parts. In the recognition stage, vehicle localization and obstacle detection through waypoint-based LiDAR ROI were performed. In the judgement stage, target velocity setting and obstacle avoidance judgement are determined in consideration of the straight/curved section and the distance between the vehicle and the neighboring vehicle. In the control stage, adaptive cruise longitudinal velocity control based on safe distance and lateral velocity control based on pure-pursuit are performed. To overcome the limited experimental environment, simulation and partial actual experiments were conducted together to develop and verify the proposed algorithms. After that, we participated in the Segye AI Robot Race Competition and performed autonomous driving racing with verified algorithms.

Keywords: Autonomous Driving System, Robot Racing Competition

1. 서 론

자율 주행 기술[1]은 차량시스템 뿐만 아니라 산업, 농업, 운송업 등 여러 분야에 걸쳐 활용되고 있고, 발전이 가속화되고 있다. 그에 발 맞추어 자율 주행 시스템 개발을 위해 무인 자동차가 목적지를 향해 주행하는 경진 대회 등이 다수 개최되고, 학계 및 산업계에서 활발히 참여하고 있다. 이러한 대부분의 자율 주행 경진 대회들은 목적지에 도달하는 시간을 단축시키거나 자율 주행을 통해 각종 미션을 빠르게 해결하는 등 속도에 초점을 맞추어 진행되고 있다[2].

그러나 최근 자율 주행 기술에서는 실제 동적인 환경에서 가져야 할 안전성 또한 요구하고 있다[3]. 따라서 이전의 미션 위주로 기록을 계측하는 자율 주행 경진 대회에서 다수의 차량이 동시에 주행하는 환경에서 빠른 속도로 목적지에 도달해야 하는 레이싱[4,5] 대회로 발전되고 있는 추세이다. 실제로 2021년 10월에 미국 IMS (Indianapolis Motor Speedway)에서 열린 Indy Autonomous Challenge 대회의 경우, 시뮬레이터로 자율 주행 시스템의 능력을 검증한 후 실제 트랙에서 일대일 자율 주행 레이싱을 진행하였다[6,7].

Received : Mar. 24. 2022; Revised : Apr. 13. 2022; Accepted : Apr. 19. 2022
※ This work was supported by Korea Foundation for Women In Science, Engineering and Technology (WISET) grant funded by the Ministry of Science and ICT (MSIT) under the team research program for female engineering students. This competition was hosted by Segye-Ilbo and managed by Unmanned Solutions. The hardware and simulator used in this work were provided from them.
* Jung Hyun Choi and Ye Eun Lim contributed equally to this work.
1 Undergraduate Students, Department of Mechanical and Information Engineering, University of Seoul, Seoul, Korea (2017430040, yeeun1410, yh04129, hyun75385, rlackdgus001, wocks2910, medist4230@uos.ac.kr)
2. Graduate Students, Department of Mechanical and Information Engineering, University of Seoul, Seoul, Korea (qkrwhdgns116, jeonghs, sjbyun15@uos.ac.kr)
† Associate Professor, Corresponding author: Department of Mechanical and Information Engineering, University of Seoul, Seoul, Korea (mjhwang@uos.ac.kr)

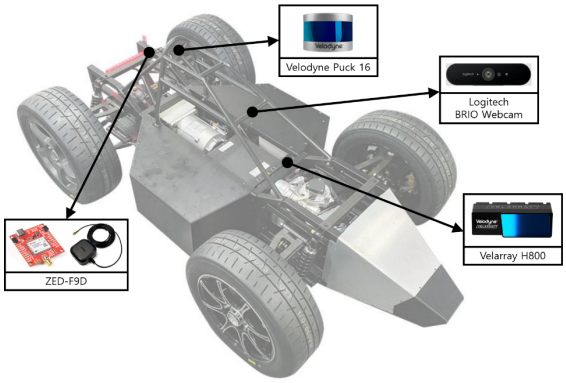
[Fig. 1] FMTC (Future Mobility Technical Center) of Seoul National University, Siheung campus

2021년 11월 27일 서울대학교 시흥캠퍼스 미래모빌리티기술센터에서 세계 AI 로봇 카레이스 대회[8]가 개최되었다. 이는 8대의 자율 주행 차량들이 동시에 출발하여 주행하는 국내 최초의 자율 주행 레이싱 대회였다. 본 대회의 목표는 자율 주행 기술을 이용하여 다른 차량과 충돌하지 않고 안전하게 추월하면서 속도를 경쟁하는 것이다. 이를 위해 본 대회에서는 배터리와 전기 모터의 구동력으로 주행하고, 실제 상용차에 적용되는 유압 브레이크와 독립 서스펜션 등을 사용한 차량 시스템을 활용하였다. 차량의 위치 추정 및 주변 환경의 인식 및 판단을 위한 실시간 이동 측위 위성 위치 확인 시스템(RTK-GPS)과 라이다(LiDAR), 카메라 센서 등을 활용하였다. [Fig. 1]과 같이 실제 차량이 주행하는 도로 환경과 유사한 서울대학교 시흥캠퍼스 미래모빌리티기술센터를 대회장으로 2차선 주행 경로를 설정하였다.

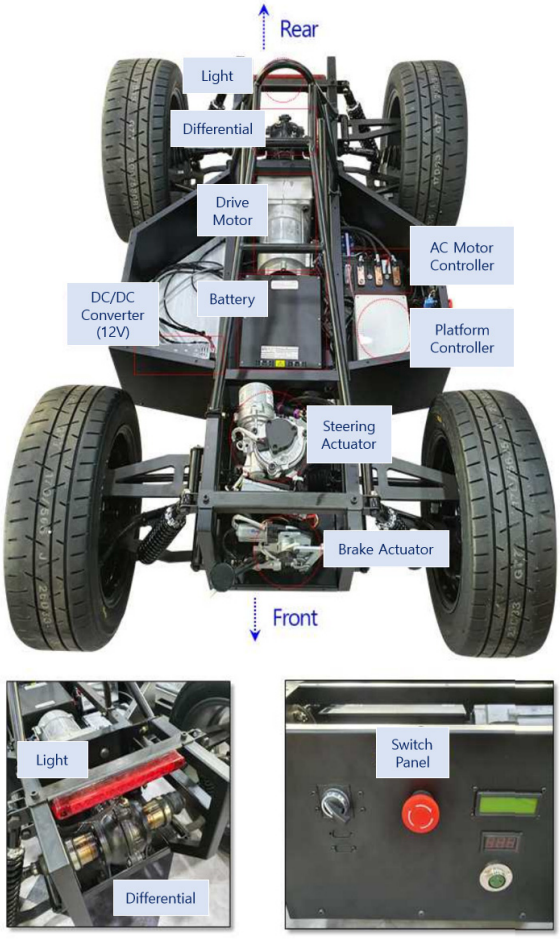
본 논문은 해당 대회에서 제공한 플랫폼을 활용하여, 인식-판단-제어 알고리즘을 통해 다수의 차량이 주행하는 환경에서 충돌하지 않고 주행 및 추월을 유연하게 하는 자율 주행 시스템을 제안한다. 먼저 하드웨어 구성에서 차량 시스템 및 각종 센서들에 대해 설명하고, 소프트웨어 구성에서 전체 자율 주행 프레임워크 및 인식-판단-제어 알고리즘을 설명한다. 또한 시뮬레이션 및 주행실험을 통해 구현한 알고리즘에 대한 검증 결과들을 제시한다.

2. 하드웨어 구성

하드웨어 시스템은 [Fig. 2]와 같이 언맨드 솔루션(Unmanned Solution) 사의 자율 주행 차량 플랫폼인 ERP-42 Racing 모델을 중심으로 위치 추정 및 주변 환경 인식을 위한 여러 센서들로 구성되어 있다. 센서는 크게 RTK-GPS, LiDAR, Camera로 구성되어 있으며, 각각 ZED-F9P 모델, Velarray H800과 Velodyne Puck 16 모델, Logitech BRIO Webcam 모델이다. 본 논문에서 제안한 자율 주행 시스템에서는 Camera를 활용하지 않았으므로 설명을 생략하였다.



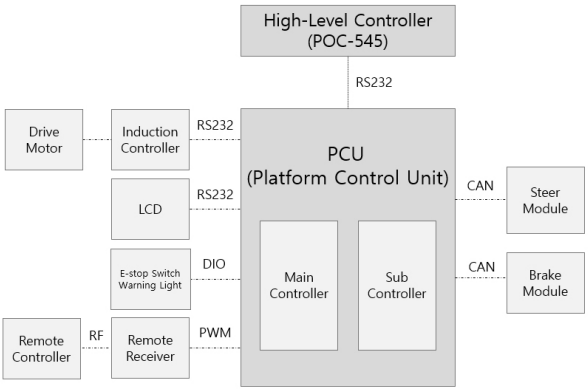
[Fig. 2] Hardware system



[Fig. 3] ERP-42 Racing model

2.1 ERP-42 Racing

대회에서 사용한 [Fig. 3]의 ERP42 Racing 모델은 실차량 대비 약 1/4 크기를 가지고 있다. 전륜의 조향과 3 kW급 AC 모터 1개에 의해 후륜 구동되며 최대 회전 속도는 3000 RPM이다. 차량 최대 속력은 40 km/h이며, 제어기 내부적으로 제한 속력인 20 km/h 이하에서 사용하였다.



HLV to PCU Packet (14 Byte)

BYTE NAME	S	T	X	AorM	ESTOP	GEAR	SPEED 0	SPEED 1	STEER 0	STEER 1	BRAKE	ALIVE	ETX 0	ETX 1
Value	0x53	0x54	0x58	0x00 or 0x01	0x00 or 0x01	0~2	0~1000		-2000~2000		0~100	0~255	0x0D	0x0A

[Fig. 4] Control architecture of ERP-42 Racing model

ERP42 Racing 모델의 제어 구조는 크게 [Fig. 4]와 같이 상위 제어기 POC-545와 플랫폼 제어기(PCU)로 구성된다. POC-545는 AMD Ryzen V1807B CPU와 Vega GPU로 구성되어 있고, RS232 프로토콜을 통해 명령이 플랫폼 제어기로 전달된다. 플랫폼 제어기는 myRIO1900를 메인 컨트롤러로 하며 추가적인 인터페이스 확장 보드가 포함되어 있다. 메인 컨트롤러에서 다시 내부 CAN 통신으로 제어 명령이 전달되며 이때 조향과 브레이크 제어가 이뤄진다. 구동 모터의 경우 직접적인 CAN 통신이 아닌 내부 컨트롤러을 거쳐 제어가 이뤄진다.

상위 제어기 POC-545에서 플랫폼 제어기로 전달할 주요 변수는 [Fig. 4]와 같다. 이 때 SPEED, STEER, BRAKE는 각각 속력, 조향각, 브레이크를 의미한다. SPEED는 0~1000의 값을 가지며, 1000은 차량의 최대 출력 속력인 20 km/h에 해당한다. STEER는 -2000~2000의 값을 가지며, 이는 실제 바퀴의 조향각 -20~20°와 같다. BRAKE는 0~100으로 브레이크를 밟는 세기의 정도를 의미한다.

2.2 라이다(LiDAR)

본 논문에서 제안한 자율 주행 시스템에서 활용한 라이다 센서는 Velarray H800[9]과 Velodyne Puck 16[10] 모델 두 가지이다.

[Fig. 5]의 Velarray H800 모델은 최신 발매된 Velodyne 사의 단방향 3D 라이다로 자율주행 및 주행 보조를 위한 고해상도의 포인트 클라우드(Point Cloud) 데이터와 빠른 데이터 전송을 지원한다. 데이터의 전송은 이더넷의 네트워크 패킷을 통해 전달하기 때문에 안정적이며 고밀도의 데이터를 빠르게 전송하는 것이 가능하다. 센서의 시야각은 수평 방향으로 120°, 수직 방향으로 16°이며 최대 200 m 거리의 장애물을 25 Hz의 속도로 인식 가능하다. 해당 모델은 단방향에 대해서 먼 거리

[Fig. 5] Velarray H800 LiDAR

[Fig. 6] Velodyne Puck 16 LiDAR

[Fig. 7] ZED-F9P RTK-GPS

까지의 고해상도 데이터를 받기 때문에 물체에 대한 정확한 인식이 가능하며 데이터의 오차나 손실이 적기 때문에 신뢰성 높은 데이터를 얻는 것이 가능하다[11].

[Fig. 6]의 Velodyne Puck 16 모델은 전방위 3D 라이다로 16개의 채널을 통해 비교적 높은 밀도로 100 m 이내의 포인트 클라우드 데이터 취득이 가능하다. 센서는 수평의 전방위에 대해 수직 방향으로 -15~15° 범위의 데이터 취득이 가능하다. 데이터는 수직 방향으로 2°의 간격을 두고 취득되며 수평 방향의 간격은 라이다 회전 속도에 따라 0.1~0.4°의 간격으로 취득된다. 마찬가지로 취득된 전방위의 고밀도 포인트 클라우드 데이터는 이더넷의 네트워크 패킷을 통해 전송되며 데이터의 오차가 작아 신뢰성이 높다.

2.3 RTK-GPS

GNSS (Global Navigation Satellite System)는 위성에서 발신한 정보를 이용하여 수신기의 측위 정보를 얻는 기술로 위성오차, 전리층과 대류층 오차, 다중경로 오차, GNSS 수신기의 오차 등이 발생하여 위치 정확도가 저하된다. 일반 GNSS의 경우 수평으로 약 10 m, 수직으로 약 20 m의 오차가 발생하며, 이를 자율주행 시스템에 활용하기에는 무리가 있다[12]. RTK-GPS (Real Time Kinematic-GPS)는 실시간으로 GNSS의 관측 값의 오차를 보정하여 위치 정확도를 향상시킨다. 기지점에 GNSS를 설치하여 기지점의 좌표와 위성에 의한 좌표의 차이로 보정 데이터를 생성한다. 이동국에서는 이를 수신해, GNSS 관측 값

과 보정 데이터를 합성하여 오차를 보정하는 방식이다.

사용한 RKT-GPS는 [Fig. 7]의 ublox사 ZED-F9P[13] 모듈을 사용한 SparkFun의 GPS-RTK-SMA 보드이고 안테나는 ANN-MB-00이다. ZED-F9P 모듈은 RTK를 사용한 위치정보를 최대 20 Hz로 연산할 수 있다. 위치의 정확도는 주변 건물들의 배치에 따라 영향을 받지만, 대회장과 비슷한 높은 건물이 거의 없는 상황에서는 수평방향 2 cm의 오차, 1 cm의 표준편차와 수직방향 3 cm의 오차, 3 cm의 표준편차로 높은 정확도와 정밀도를 보인다[14]. 이번 대회에서는 국토지리정보원에서 제공하는 인천 관측소를 기준국으로 하여 보정 데이터를 수신했다.

3. 소프트웨어 구성

3.1 프레임워크

자율 주행 시스템의 소프트웨어의 프레임워크는 [Fig. 8]과 같이 인식-판단-제어 세 단계로 구성되어 있다. RTK-GPS, Lidar 등의 센서들을 통해 받은 원본 데이터들을 가공하여 현재 차량의 위치 및 주변 상황을 인식하고, 인식한 정보를 바탕으로 주행 방식을 판단하고, 이를 바탕으로 차량을 제어한다.

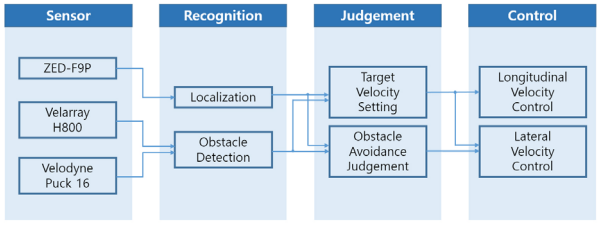
각 단계 사이의 통신은 ROS (Robot Operating System)를 기반으로 구현하였다. ROS는 미들웨어(Middleware)로 이기종 장치 사이의 데이터 통신을 노드와 메시지 형태로 구성하여 간단하게 연결해준다.

3.2 인식

인식 단계는 프레임워크의 첫 번째 단계로 센서로부터의 데이터를 통해 차량의 현재 위치 추정과 장애물 인식을 하는 과정이 포함된다. 먼저 본 대회로부터 제공된 주행 경로를 더 정확한 웨이포인트(Waypoint)로 변환하는 과정을 설명하고, 각각의 인식이 진행되는 과정을 설명한다.

3.2.1 웨이포인트 설정

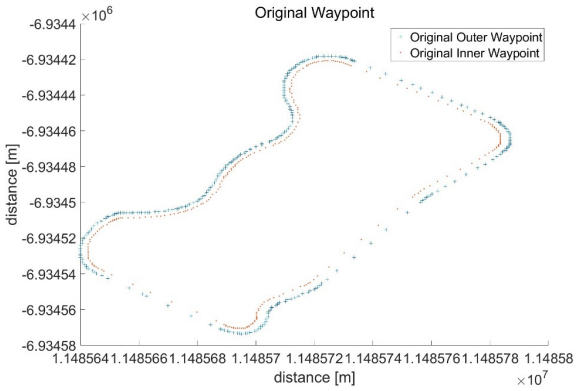
본 대회에서 제공한 주행 경로는 약 400 m 길이의 트랙에 대해 [Fig. 9]와 같이 1차선은 274개, 2차선은 257개의 위도/경도 기반 좌표로 이루어졌다. 해당 주행 경로는 각 좌표 사이의 거리가 곡선의 경우 약 1.14 m, 직선의 경우 약 4.918 m로 균일하지 않고, 멀게 형성되었기 때문에 정확한 주행 제어가 어려웠다. 따라서 정확한 경로 추종을 위해 경유지들 사이를 보간하여 새로운 웨이포인트를 설정하였다. 우선 약 20~30개의 일정 구간의 경유지들에 대해 기울기가 -2~2가 되도록 회전시킨 뒤, 구간의 모양에 따라 3~5차 다항식으로 근사하였다. 이후 근사한 다항식에 0.3 m 간격의 일정한 웨이포인트를 생성하였고, 회전시킨 점들에 대해서 역회전을 하여 원상태로 복구하였다. 그 결과 1차선은 약 1530개, 2차선은 약 1450개의 일정한 웨이포인트가 생성되었고, [Fig. 10]와 같은 결과가 나타났다.



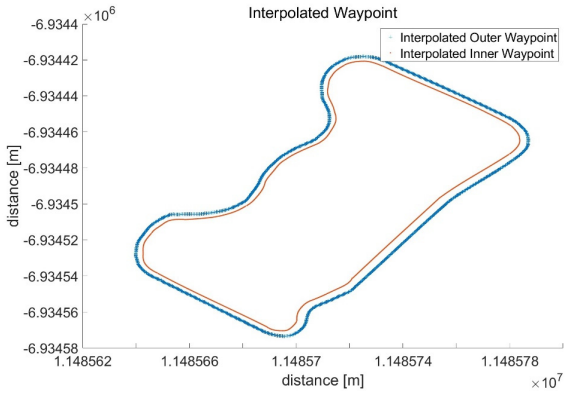
[Fig. 8] Software framework

3.2.2 차량 위치 추정(Localization)

RTK-GPS 센서로 받아오는 정보는 위도와 경도로 표현되는 현재 좌표이다. 하지만 위도/경도 좌표계는 지구, 즉 구 상에서의 위치를 표현하고 있고 각도 단위로 표현되기 때문에 이를 주행 알고리즘에 바로 적용하기에는 적합하지 않다. 따라서 3차원 위도/경도 좌표계를 2차원 평면상으로 나타낸 투영 좌표계로 변환시켜 사용해야 한다. 따라서 기상청에서 배포한 좌표 변환 코드를 사용하여 위도/경도 좌표를 직교 좌표



[Fig. 9] Provided original waypoints



[Fig. 10] Interpolated waypoints

계의 x, y 정보로 변환하였다.

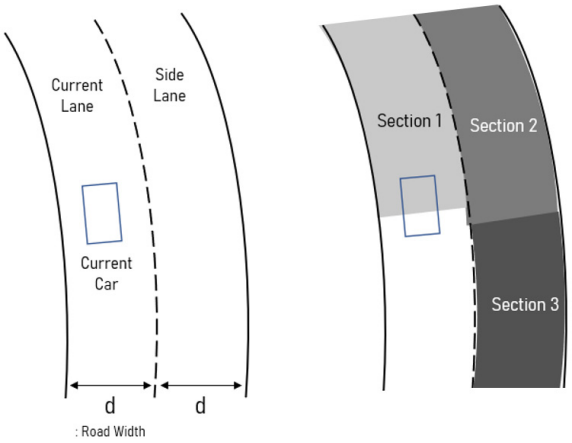
또한 차량의 위치 추정을 위해 현재 위치와 웨이포인트 사이의 거리를 계산하여 거리가 가장 작은 웨이포인트를 현재 차량이 위치한 웨이포인트로 설정하였다. 추가적으로 차량의 헤딩(Heading) 각을 구하기 위해 현재 위치와 이전 위치 사이의 벡터를 생성하여 계산하였다.

3.2.3 장애물 인식(Obstacle Detection)

본 대회는 다수의 차량이 동시에 주행하는 환경이기 때문에 안전한 자율 주행을 위해서는 도로 위의 다른 차량을 빠르게 인식하는 기술이 필수적이다. 본 논문이 제안한 자율 주행 시스템은 카메라를 사용하지 않고 두 개의 LiDAR만을 활용하여 도로 위의 객체를 인식하였다. 고밀도의 전방 포인트 클라우드를 반환하는 Velarray H800은 전방 장애물 인식를 담당하고, 전방위 포인트 클라우드를 반환하는 Velodyne Puck 16은 후방 및 측방 장애물 인식를 담당하였다. 이때 LiDAR의 ROI (Region Of Interest)는 웨이포인트를 따라 현재 차선의 전방과 옆의 추월 차선의 전후방에 대해 각각 20 m 길이의 일정한 간격을 갖는 영역으로 설정하여 [Fig. 11]과 같이 총 3개의 영역으로 구분하였다. 따라서 웨이포인트에 따라 LiDAR의 ROI를 설정하였기 때문에 해당 ROI에 감지되는 물체는 다른 차량 밖에 없으므로 트래킹(Tracking) 과정을 생략한 후 노이즈 제거만으로 다른 차량을 인식할 수 있다.

3.3 판단

센서에 들어온 데이터를 통해 상황을 인지한 후에는 차량이 이후에 어떻게 움직일지에 대한 판단 과정이 필요하다. 해당 판단 과정은 크게 두 가지로 차량의 목표 속도를 설정하는 과정과 추월 여부를 판단하는 과정을 설명한다.



[Fig. 11] ROI of LiDAR point cloud-based waypoints

3.3.1 목표 속도 설정(Target Velocity Setting)

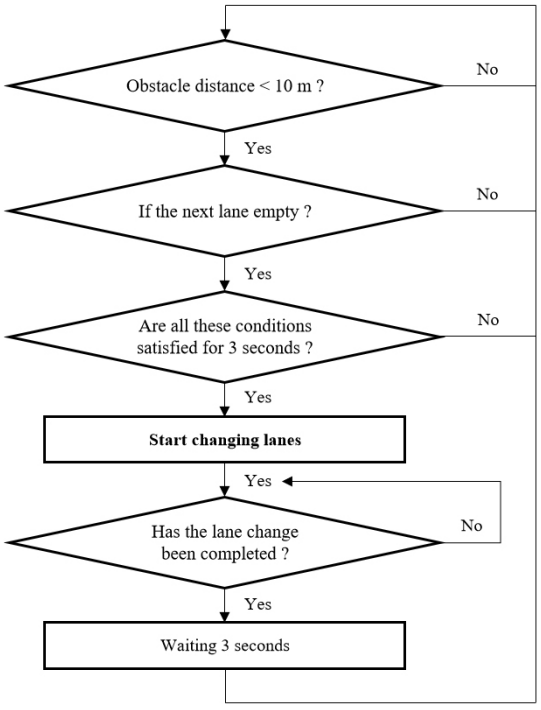
본 논문에서 제안한 자율 주행 시스템은 차량의 목표 속도를 판단할 때 두 가지를 고려한다. 첫 번째는 현재 차량이 위치한 구간의 직선/곡선 특성으로, 직선 구간에 비해 곡선 구간에서의 시야 확보가 느리고 주행의 안전성이 떨어지기 때문에 목표 속도의 최대 값을 다르게 설정하여 속도를 제한한다. 즉 직선 구간의 경우 18 m/s, 곡선 구간의 경우 14.5 m/s로 설정하여 곡선 구간에서 상대적으로 더 낮은 목표 속도를 갖도록 한다. 직선/곡선 구간 판단의 경우 현재 차량이 위치한 웨이포인트를 기준으로 전방의 일정한 간격으로 떨어진 2개의 웨이포인트와 이루는 원의 곡률을 계산하여 특정 임계값을 기준으로 직선/곡선 구간을 판단한다.

두 번째는 인식 단계에서 인식한 전방 장애물 사이의 거리이다. 이후 제어 단계에서 적응형 크루즈 제어(Adaptive Cruise Control)를 구현하기 위해 차량과 전방 장애물 사이의 유지할 목표 거리를 설정한 후에 해당 목표 거리를 유지할 수 있도록 목표 속도를 설정한다.

따라서 최종적으로 목표 속도 설정을 위해 직선/곡선 구간을 판별하여 최대 목표 속도 값을 다르게 제한하고, 인식한 전방 장애물과의 거리를 기반으로 목표 거리를 유지하는 방향으로 목표 속도 값을 결정한다.

3.3.2 추월 판단(Obstacle Avoidance Judgement)

앞서 목표 속도 설정에서 설명한 적응형 크루즈 제어만으



[Fig. 12] Flowchart of obstacle avoidance judgement

로 차량을 제어할 경우, 전방의 차량이 목표 거리를 유지한 채 정지하게 되면 차량이 주행을 멈추고 전방 차량이 움직일 때까지 정지 상태를 지속하는 문제가 발생한다. 따라서 이러한 상황을 해결하기 위해서는 전방의 차량에 대한 추월의 가능성을 판단하여 옆의 차선을 이용하여 추월을 시도해야 한다. [Fig. 12]는 추월 판단에 대한 순서도이다. 실제 사람이 운전을 하면서 추월을 할 때 판단을 내리는 순서를 참고하였다. 주행 시 해당 조건들을 계속해서 검토하여 추월 명령을 내리며, 추월 판단이 완료되기 전까지는 앞 차량에 대해 적응형 크루즈 제어로 주행한다.

추월을 판단하는 조건은 크게 세 가지가 존재한다. 첫 번째 조건은 [Fig. 11]의 Section 1 ROI 영역의 전방 차량까지의 거리가 10 m 이내일 조건이다. 전방의 차량이 더 가까운 상태에서 추월을 진행한다면 적응형 크루즈 제어에 의해 더 낮은 속도로 추월해야 하기 때문에 전방의 차량으로부터 10 m 떨어진 거리에서부터 추월 판단을 시작한다.

두 번째 조건은 [Fig. 11]의 Section 2, 3 ROI 영역에 다른 차량이 존재하지 않는다는 조건이다. 만약 옆의 차선의 후방인 Section 3 ROI 영역에 다른 차량이 존재한다면 추월할 때 해당 차량과 충돌할 위험성이 있고, 옆의 차선의 전방인 Section 2 ROI 영역에 다른 차량이 존재한다면 추월의 의미가 사라지기 때문이다. 따라서 추월의 경우 해당 두 조건이 모두 만족할 경우에만 추월이 가능하다고 판단하고, 이외의 경우에는 추월이 불가능하다고 판단한다.

마지막 세 번째 조건은 위에 제시된 두 조건이 모두 만족하는 상태를 3초 유지하는 것이다. 옆 차선의 장애물 여부 또는 전방 장애물까지의 거리가 불안정하여 조건 만족 여부가 수시로 바뀌게 된다면 추월에 대한 신뢰도가 낮다고 판단하였다. 따라서 3초라는 시간동안 주변 환경이 안정적으로 조건을 만족한다면 추월 명령을 전달하게 된다.

이후 추월하려는 차선의 목표 웨이포인트까지의 거리를 반복적으로 확인하여 현재 차선의 웨이포인트까지의 거리보다 짧아지면 차선 변경이 완료되었다고 판단한다. 차선 변경이 한번 완료된 후에는 연속적인 추월을 방지하기 위해 5초 동안 추월을 시도하지 않도록 한다.

3.4 제어

앞서 진행한 판단을 기반으로 차량을 제어한다. 차량의 제어는 구동 모터의 속도를 제어하는 종방향 속도 제어와 조향을 제어하는 횡방향 속도 제어로 나뉜다.

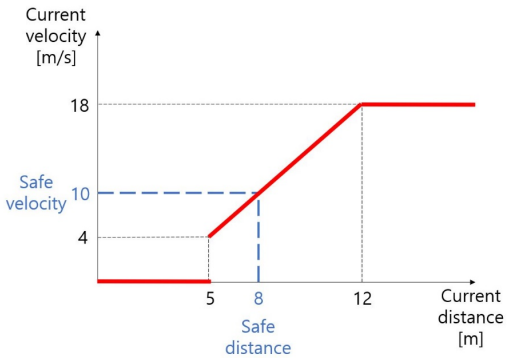
3.4.1 종방향 속도 제어(Longitudinal Velocity Control)

동일 차선의 앞 차량이 존재하는 주행 상황에서 안전과 효율을 위해서 적응형 크루즈 제어를 구현하였다. [Fig. 13]과 [Fig. 14]는 각각 직선 구간과 곡선 구간을 지날 때의 거리-속도 그래프이다. 이때, x 축인 Current distance는 라이다(LiDAR) 센서를 이용하여 측정한 가장 가까운 전방 차량까지의 현재 거리이고 y 축인 Current velocity는 차량의 현재 속도를 나타낸다. 또한 Safe distance는 차량 이 전방의 차량과 유지해야 할 안전 거리이며, 이때의 속도인 Safe velocity를 안전 속도라 한다.

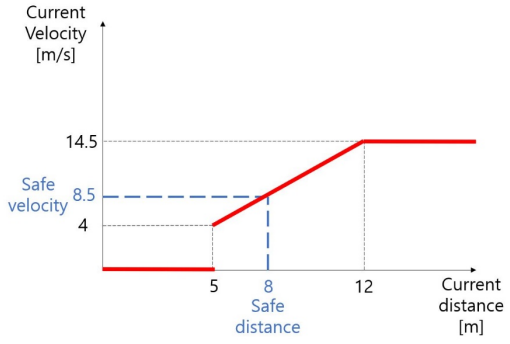
[Fig. 13]과 같이 직선 구간의 차량은 안전 거리 8 m, 안전 속도 10 m/s를 유지하는 것을 기본으로 한다. 또한 현재 거리가 5 m 이하인 경우에는 현재 속도는 0 m/s으로 급정지하도록 설정하였는데, 이는 차량의 가/감속 실험에서 최대 속도에서 도출한 안전 제동 거리(3 m)를 고려한 값이다. 현재 거리가 5~12 m 사이인 경우에는 안전 거리 8 m로 유지하기 위해 현재 속도가 4~18 m/s 사이에서 선형적으로 변화한다. 또한 현재 거리가 12 m 이상일 때는 최대 속도인 18 m/s로 유지하여 주행한다.

[Fig. 14]와 같이 곡선 구간에서는 직선 구간에 상대적으로 안전을 위한 감속이 필요하다. 즉 직선 구간에서의 최대 속도가 18 m/s인 것에 비해 곡선 구간에서는 12 m 이상에서 유지하는 최대 속도가 14.5 m/s로 제한된다. 또한 이에 따라 안전 거리 8 m일 때의 안전 속도는 8.5 m/s로 줄어든다.

따라서 최종적으로 안전 거리를 유지하기 위해 현재 속도를 선형적으로 변화시키는 구간을 안전 거리와 현재 거리 사



[Fig. 13] Distance-velocity graph in straight section



[Fig. 14] Distance-velocity graph in curved section

이의 오차에 대한 P (Proportional) 제어기로 구현하여 종방향 속도를 제어하였다. 또한 추가적으로 감속 또는 급정지 상황에서 속도 제어를 더 빠르게 하기 위해 브레이크를 사용하였다. 브레이크 제어 값의 경우에는 안전 속도와 현재 속도의 차이와 비례한 값에 일정한 값이 더해진 값을 사용하였다.

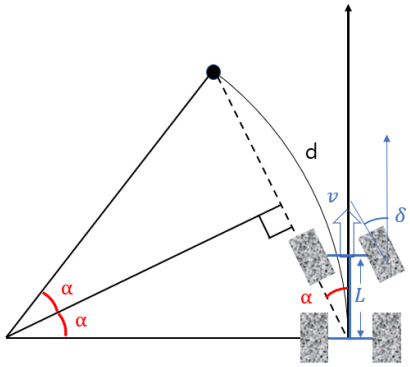
3.4.2 횡방향 속도 제어(Lateral Velocity Control)

횡방향 속도 제어에는 Pure-pursuit 알고리즘[15]을 활용하여 현재 차선 내의 주행과 옆의 차선으로의 추월을 구현하였다. 기본 Pure-pursuit 주행 알고리즘의 경로 추종 방법을 나타낸 계략도는 [Fig. 15]와 같다. 이 때 d는 전방 주시 거리(Look-ahead distance), α는 차량의 헤딩각과 전방 주시 거리에 존재하는 목표 사이의 각도, δ는 애커만 조향 각도(Ackerman-steering angle), L은 차량의 길이, v는 차량의 속도를 의미한다.

알고리즘의 설계 값으로 전방 주시 거리 d가 결정되며, 해당 거리의 시작은 후륜 축의 중앙에, 끝은 추종 목표 경로 상에 존재한다. 이 때 차량의 헤딩각과 전방 주시 거리에 존재하는 목표 사이의 각도 α를 측정하고, 전방 주시 거리에 존재하는 목표로 원운동을 하기 위한 애커만 조향 각도 δ를 계산한다. 즉 각도 α를 피드백 값으로 하여 경로 추종 제어를 진행한다. α의 각도 차이가 발생하였을 때, 목표 경로로 수렴하기 위한 차량의 애커만 조향 각도는 식 (1)과 같이 계산할 수 있다. 전방 주시 거리 d는 주로 속도에 비례하여 크기를 조절하므로 비례계수 k와 차량의 속도 v의 곱으로 나타낸다.

$$\delta = \tan^{-1}\left(\frac{2Lsin\alpha}{d}\right) = \tan^{-1}\left(\frac{2Lsin\alpha}{kv}\right) \quad (1)$$

추월은 기본 주행인 Pure-pursuit 주행을 유지하면서 진행된다. 안쪽과 바깥쪽의 2가지 웨이포인트 정보를 이미 알고 있기 때문에 단순히 Pure-pursuit 기반으로 차선만을 변경하여 추월이 가능하다. 따라서 현재 위치 점에서 추월 판단이 내려졌을 경우 추종하던 경로를 옆차선의 경로로 바꿔 앞 차량을 추월한다.



[Fig. 15] Pure-pursuit algorithm

4. 실험 및 검증

앞서 설명한 하드웨어 및 소프트웨어를 활용하여 본 대회를 위한 자율 주행 알고리즘을 구축하기 위해 실험을 진행하였다. 하지만 실험 과정에서는 안전상의 문제로 실제 대회와 같이 다수의 차량이 동시에 주행하는 환경을 구축하기 어렵고, FMTC 대회장을 활용할 수 있는 시간이 한정적이었기 때문에 실제 대회 환경에서의 충분한 실험을 할 수 없었다. 따라서 이러한 문제를 해결하기 위해 시뮬레이션 및 실제 실험을 함께 진행하는 전략을 사용하였다.

4.1 시뮬레이션

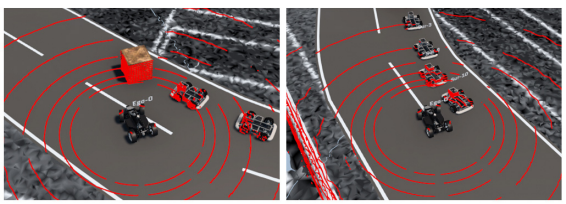
제한된 실험 환경을 극복하기 위해서 MORAI 자율주행 시뮬레이터[16]를 활용하여 실제 대회 환경과 유사하게 FMCT 대회장 맵에서 다수의 차량이 동시에 주행하는 환경을 구축하였다. [Fig. 16]과 같이 시뮬레이션 환경을 통해 먼저 GPS 센서를 이용하여 보간한 웨이포인트에 따른 종방향 및 횡방향 제어의 정확도를 높였다. 또한 정적 및 동적 장애물에 대해 LiDAR 센서를 이용한 인식 및 판단 알고리즘을 개발 및 검증하였고, 종방향 및 횡방향 속도 제어 알고리즘을 검증하였다. 최종적으로 다양한 속도를 가진 다수의 차량들을 시뮬레이터 환경에 포함시켜 자율 주행의 전체적인 인식-판단-제어 알고리즘을 반복적으로 검증하였고, 특정 조건에 발생하는 문제를 해결하여 알고리즘의 강건성을 높였다.

4.2 주행 실험

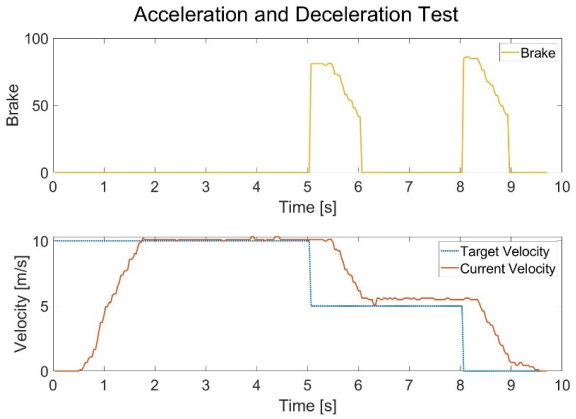
인식-판단-제어 알고리즘의 전체적인 개발은 시뮬레이션을 통해 진행할 수 있었지만, 지면의 마찰, 실제 차량의 모터 특성, 주행 도로 주변 환경 등 시뮬레이터에 정확하게 포함되지 않는 것들이 존재하기 때문에 정확한 검증이 필요한 몇 가지 실험에 대해서는 부분적으로 실제 실험을 진행하였다.

4.2.1 가/감속 제어 실험

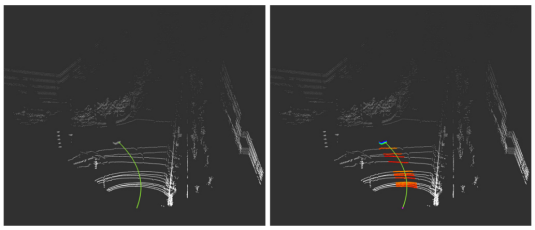
차량의 정확한 종방향 제어 또는 추월 제어를 위해서는 차량의 가/감속 시간과 그에 따른 제동 거리를 정확하게 파악해



[Fig. 16] Examples of using MORAI simulator



[Fig. 17] Example of acceleration and deceleration test



[Fig. 18] Example of point cloud ROI based waypoints

야 한다. 하지만 시뮬레이터 환경에서는 지면의 마찰과 차량의 실제 모터 특성 등이 정확하게 반영되어 있지 않으므로 부분적으로 실제 대회 환경과 동일한 아스팔트 환경에서 가/감속 제어 실험을 진행하였다.

[Fig. 17]과 같이 앞서 설정한 종방향 제어 알고리즘을 활용하여 각각의 목표 속도로 도달하는 가/감속 시간과 제동 거리를 측정하였고, 그 결과 차량의 최대 속도 기준 약 3 m의 제동 거리가 나타나는 것을 확인하였다. 따라서 해당 결과를 종방향 제어의 급정지 기준 거리에 반영하여 보다 안전한 속도 제어를 구현하였다.

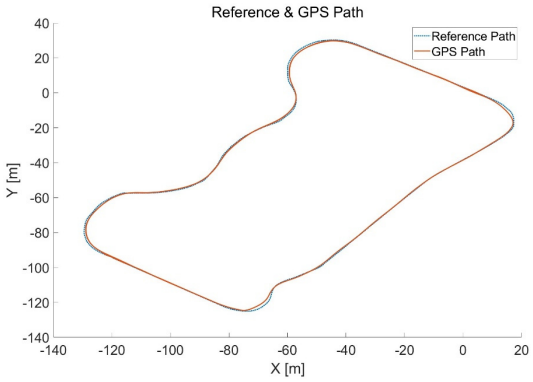
4.2.2 포인트 클라우드 ROI 설정 실험

시뮬레이터에 정확하게 반영되지 않은 실제 주행 도로 환경에 대해 장애물 인식 알고리즘의 정확도를 확인하기 위해서 실제 검증 실험을 진행하였다.

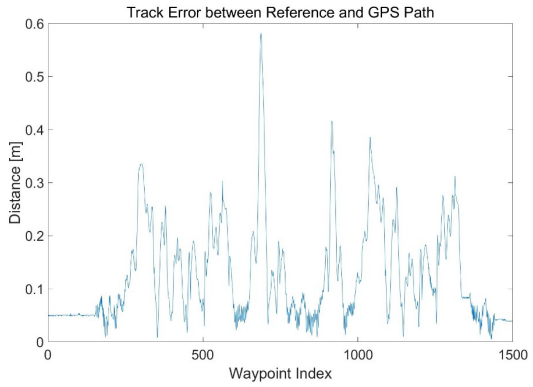
그 결과 [Fig. 18]과 같이 Velarray H800을 통해 들어온 원본 데이터와 웨이포인트를 기반으로 설정한 ROI 내의 데이터를 비교하였다. 해당 실험을 통해 실제로 미래에 주행할 경로에 라이다의 ROI가 정확히 설정되는 것을 확인하였고, 주행 경로 상의 장애물 또한 정확히 인식하는 것을 확인하였다.

4.2.3 종방향/횡방향 제어 실험

실제 대회 환경에서 Waypoint로 설정한 Reference Path에



[Fig. 19] Reference and GPS path



[Fig. 20] Track error between reference and GPS path

대해 앞서 설명한 종방향 및 횡방향 제어를 진행하여 실제로 주행한 GPS Path와 비교하였다.

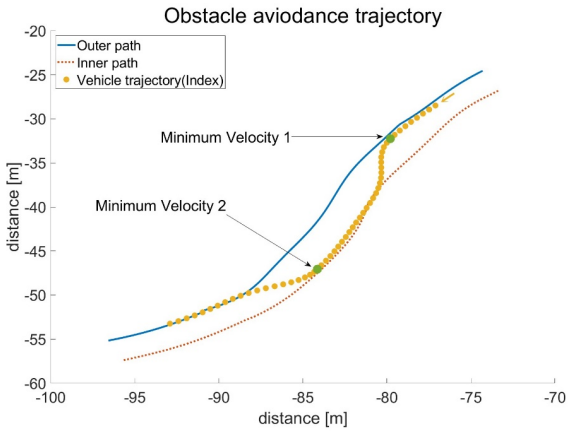
그 결과 [Fig. 19]와 같이 직선 및 곡률이 작은 곡선 구간은 실제 GPS Path가 Reference Path와 거의 일치하는 비교적 정확한 종방향 및 횡방향 제어가 이루어지는 것을 확인하였다. 하지만 곡률이 큰 곡선 구간은 두 Path의 차이가 비교적 크게 나타나는 것을 확인하였고, [Fig. 20]과 같이 최대 약 0.6 m의 오차가 발생하는 것을 확인하였다.

4.2.4 추월 실험

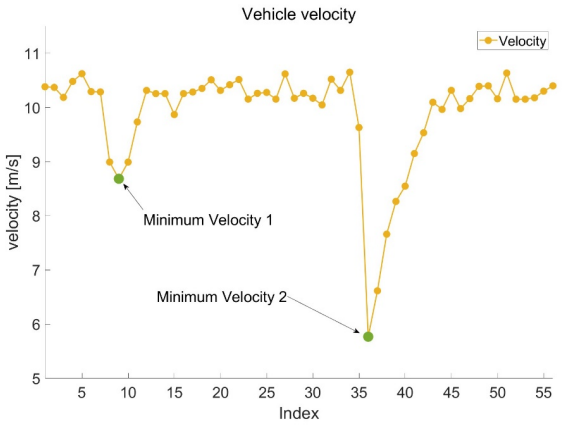
시뮬레이션 상의 추월에 대한 횡방향 제어는 비교적 정확하지 않기 때문에 실제 도로 환경에서 정적 장애물에 대해 추월 실험을 진행하였고, 그 결과를 통해 추월을 판단하는 기준 거리를 설정하였다.

[Fig. 21]과 [Fig. 22] 같이 18 m/s 이하의 속도에서 정적 장애물을 만났을 경우, 추월을 판단하는 기준 거리를 다르게 설정하여 실험을 통해 차선 변경을 통한 추월이 부드럽게 이뤄지는지 반복적으로 확인하였다. [Fig. 22]와 [Fig. 23]은 실제 실험에서 확인한 추월 경로와 그에 대한 속도를 나타낸 그림이다. 전방 주행 경로에 장애물이 나타나게 되면 차량의 크루즈

[Fig. 21] Experiment for obstacle avoidance



[Fig. 22] Path in obstacle avoidance



[Fig. 23] Velocity in obstacle avoidance

제어가 진행되고, 차량은 속도를 감소하면서 옆 차선을 확인한 이후 추월을 정상적으로 진행한다는 사실을 확인했다. 그 결과 약 10 m의 기준 거리에서 추월이 가장 안전하게 이루어진다는 결과를 도출하였고, 이를 판단 알고리즘에 반영하였다.

5. 결 론

본 논문은 세계 AI 로봇 카레이서 대회에서 제공한 차량 시스템 및 RTK-GPS와 LiDAR 센서를 활용하여 다수의 차량이 동시에 주행하는 환경에서 안전하고 빠르게 주행할 수 있는 자율주행 시스템을 제안한다. 인식-판단-제어 알고리즘을 통해 인식 단계에서는 차량의 위치 추정 및 웨이포인트 기반 라이다 ROI를 통한 장애물 인식을 진행하였다. 또한 판단 단계에서는 직선/곡선 구간 여부와 전방 차량 사이의 거리를 고려한 목표 속도 설정 및 추월 판단을 진행하고, 마지막 제어 단계에서는 안전 거리 기반 적응형 크루즈 종방향 속도 제어 및 Pure-pursuit 기반 횡방향 속도 제어를 진행하였다. 또한 제한된 실험 환경을 극복하기 위해 시뮬레이션 및 부분적인 실제 실험을 함께 진행하여 제안한 자율 주행 시스템의 개발 및 검증을 진행하였다.

본 논문의 한계점은 주행 경로가 고정되어 있기 때문에 웨이포인트 부근의 정보만을 활용하고 추가적인 객체 인식은 생략한 것이다. 실제 더 복잡한 환경에서는 추가적인 객체 인식이 필요하다. 또한 본 대회에서는 주변 객체에 대한 정보를 LiDAR 센서만을 활용하여 얻었지만, 이미지로써 많은 정보를 전달해주는 카메라 센서도 활용할 필요가 있다.

향후 연구 방향으로는 카메라 센서도 함께 활용한 객체 인식을 진행하여 인식에 대한 정확도 및 데이터 양을 증가시켜 Pure-pursuit 이외에 MPC 등의 다양한 제어 방식을 사용하는 자율 주행 시스템을 구현할 계획이다.

References

[1] Y. H. Kim and H. G. Kim, “Trends in Development of Autonomous Driving Vehicle,” *Information and Communications Magazine*, vol. 34, no. 5, pp. 10-18, 2017, [Online], http://www.koreascience.or.kr/article/JAKO2017 18054545396.page.

[2] J. H. Lee, K. J. Ahn, T. G. Lee, K. I. Min, O. S. Kwon, S. W. Baek, M. C. Park, D. H. Cho, J. R. Hwang, J. K. Kang, S. H. Lee, D. Y. Seo, J. H. Kim, and Y. S. Kang, “Development of control system for international autonomous driving competition using robot operating system,” *Journal of Institute of Control, Robotics and Systems*, vol. 25, no. 5, pp. 363-373, 2019, DOI: 10.5302/J.ICROS.2019.19.0022.

[3] S. W. Lee, “Overtake strategy for autonomous racing using model predictive control,” M.S thesis, KAIST, Daejeon, Korea, 2020, [Online], https://koasas.kaist.ac.kr/handle/10203/ 284962.

[4] D. Caporale, A. Settimi, F. Massa, F. Amerotti, A. Corti, A. Fagiolini, M. Guiggiani, A. Bicchi, and L. Pallottino, “Towards the Design of Robotic Drivers for Full-Scale Self-Driving Racing Cars,” *2019 International Conference on Robotics and Automation (ICRA)*, pp. 5643-5649, 2019, DOI: 10.1109/ICRA.2019.8793882.

[5] D. Caporale, A. Fagiolini, L. Pallottino, A. Settimi, A. Biondo, F. Amerotti, F. Massa, S. De Caro, A. Corti, and L. Venturini, "A Planning and Control System for Self-Driving Racing Vehicles," *2018 IEEE 4th International Forum on Research and Technology for Society and Industry (RTSI)*, pp. 1-6, 2018, DOI: 10.1109/RTSI.2018.8548444.

[6] G. Hartmann, Z. Shiller, and A. Azaria, "Autonomous Head-to-Head Racing in the Indy Autonomous Challenge Simulation Race," *ArXiv abs/2109.05455*, 2021, [Online], https://arxiv.org/abs/2109.05455.

[7] A. Wischnewski, M. Geisslinger, J. Betz, T. Betz, F. Fent, A. Heilmeier, L. Hermansdorfer, T. Herrmann, S. Huch, P. Karle, F. Nobis, L. Ögretmen, M. Rowold, F. Sauerbeck, T. Stahl, R. Trauth, M. Lienkamp, and B. Lohmann, "Indy Autonomous Challenge - Autonomous Race Cars at the Handling Limits," ArXiv, abs/2202.03807, 2022, [Online], https://arxiv.org/abs/2202.03807.

[8] Segye, *Homepage of Segye AI Robot Car Race Competition*, [Online], http://robotcar.segye.com/, Accessed: March 24, 2022.

[9] Velodyne LiDAR, *Maunal of Velarray H800*, [Online], https://velodynelidar.com/products/velarray-h800/, Accessed: March 24, 2022.

[10] Velodyne LiDAR, *Maunal of Velodyne Puck 16*, [Online], https://velodynelidar.com/wp-content/uploads/2019/12/63-9243-Rev-E-VLP-16-User-Manual.pdf, Accessed: March 24, 2022.

[11] N. Jayaweera, N. Rajatheva, and M. Latva-Aho, "Autonomous driving without a burden: View from outside with elevated LiDAR," *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*, Kuala Lumpur, Malaysia, 2019, DOI: 10.1109/VTCSpring.2019.8746507.

[12] N. H. Kim and C. H. Park, "A study on the advanced altitude accuracy of GPS with barometric altitude sensor," *Journal of the Institute of Electronics and Information Engineers*, vol. 49, no. 10, pp. 18-22, 2012, DOI: 10.5573/ieek.2012.49.10.018.

[13] ublox, *Maunal of ZED-F9P*, [Online], https://www.u-blox.com/en/ubx-viewer/view/ZED-F9P_IntegrationManual_UBX-18010802?url=https%3A%2F%2Fwww.u-blox.com%2Fsites%2Fdefault%2Ffiles%2FZED-F9P_IntegrationManual_UBX-18010802.pdf, Accessed: March 24, 2022.

[14] D. Janos and P. Kuras, "Evaluation of Low-Cost GNSS Receiver under Demanding Conditions in RTK Network Mode," *Sensors*, vol. 21, no. 16, 2021, DOI: 10.3390/s21165552.

[15] N. A. Shneydor, "Pure Pursuit," *Missile guidance and pursuit: kinematics, dynamics and control*, 1st ed. Elsevier, 1998, ch. 3, pp. 47-76, [Online], https://www.sciencedirect.com/book/9781904275374/missile-guidance-and-pursuit.

[16] MORAI, *Homepage of MORAI Simulator*, [Online], https://www.morai.ai/product, Accessed: April 10, 2022.

최 정 현

2017~현재 서울시립대학교 기계정보공학과 학사과정

관심분야: Autonomous Driving, Robotic Manipulation, Quadruped Locomotion

박 종 훈

2021 서울시립대학교 기계정보공학과(학사)

2021~현재 서울시립대학교 기계정보공학과 석사과정

관심분야: Trajectory Planning and Control of Mobile Manipulator, Autonomous Driving

임 예 은

2019~현재 서울시립대학교 기계정보공학과 학사과정

관심분야: Robotic Vision, Mobile Robot, Autonomous Driving

정 현 수

2021 서울시립대학교 기계정보공학과(학사)

2021~현재 서울시립대학교 기계정보공학과 석사과정

관심분야: Robotic Vision

변 승 재

2021 한국교통대학교 기계공학과(학사)

2021~현재 서울시립대학교 기계정보공학과 석사과정

관심분야: Robot Control, Design of Robot Gripper

사 공 의 훈

2017~현재 서울시립대학교 기계정보공학과 학석사연계과정

관심분야: Motion planning, Autonomous driving, Control theory

박 정 현

2019~현재 서울시립대학교 기계정보공학과 학사과정

관심분야: Computer Vision, Robotic Manipulation, Autonomous Driving

김 창 현

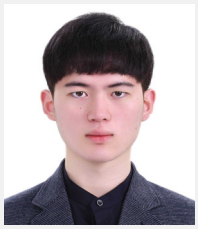
2017~현재 서울시립대학교 기계정보공학과 학사과정

관심분야: Robotic Manipulation, Visual Recognition

이 재 찬

2017~현재 서울시립대학교 기계정보공학과 학사과정

관심분야: Autonomous Driving, Automated Process System, Semiconductor Process



김 도 형

2018~현재 서울시립대학교 기계정보공학과 학사과정

관심분야: Robotic Manipulation, Embedded System

황 면 중

2001 한국과학기술원 기계공학과(학사)

2003 한국과학기술원 기계공학과(석사)

2007 한국과학기술원 기계공학과(박사)

2008~2009 Case Western Reserve University, Research Associate

2010~2013 삼성전자 생산기술연구소 책임연구원

2013~2015 한라대학교 기계자동차공학부 조교수

2015~2021 한국교통대학교 기계공학전공 조교수/부교수

2021~현재 서울시립대학교 기계정보공학과 부교수

관심분야: Robot Motion Planning, Motion Control, Field Robotics