

**국제 대학생 창작자동차 경진대회**

**자율주행자동차 기술보고서**

- 자동차번호 : 115
- 참가팀명 : MACARON
- 참가대학명 : 동국대학교
- 주요내용
  - 자율주행자동차 아키텍처
    - ▶ 플랫폼과 센서의 Layout
    - ▶ 센서 장착 설계
  - 자율주행자동차 제어 로직 개발
    - ▶ 인지(macaron\_sensing)
    - ▶ 계획(path\_planning)
    - ▶ 판단&제어(path\_planning)

- ※ 1. 제어로직 및 주요장치에 대해서는 전년도 출전자동차와의 차이점에 대하여 기술되어 있어야 한다.  
2. 작성형식은 자유이며 표지를 포함하여 총 30페이지 이내  
(휴먼명조체로 크기는 12포인트, 줄간격은 160%로 작성)

- 첨부
  - 자율주행자동차 제원표 1부 (양식 6)

2019. 08. 16.

MACARON 팀 팀장

홍 지 훈 (서명)

국제 대학생 창작자동차 경진대회 조직위원장 귀하

**국제 대학생 창작자동차 경진대회**

**자율주행자동차 제원표**

- 자동차번호 : 115
- 참가팀명 : MACARON
- 참가대학명 : 동국대학교

▶일반제원

구분	제원	보기
플랫폼 중량	250 Kg <sub>f</sub>	
플랫폼 최대 길이	1980 (mm)	
플랫폼 최대 폭	1180 (mm)	
플랫폼 최대높이	1100 (mm)	
기타		

▶ 자율주행자동차 제원

구분	제원	보기
LiDAR	SICK TIM571 1EA	시스템의 세부사항은 아키텍처 파트에서 기술
Camera	Logitech c930c 1EA	
Stereo camera	oCamS-1CGN-U	
IMU	stereo camera 내장	
GPS	zed f9p	
Controller	MSI GL73 9SE	
USB 확장허브	NEXT-UH308	
DC48-220AC 인버터	DK-4806	

2019. 08. 16.

MACARON 팀 팀장

홍 지 훈 (서명)

국제 대학생 창작자동차 경진대회 조직위원장 귀하

# 목차

## I. 서론(introduction & related works)

## II. 자율주행자동차 아키텍처(system overview)

1. 플랫폼과 센서의 Layout
2. 센서 장착 설계

## III. 자율주행자동차 제어 로직 개발(algorithm description&method)

1. 통합 알고리즘 개요
2. 인지(macaron\_sensing)
  - 가. Localization
  - 나. 차선 인식
  - 다. 충돌인지
  - 라. 표지판/신호등 인지
3. 계획(path\_planning)
  - 가. 근거리 노드 탐색 및 lookahead 설정
  - 나. 경로 생성
4. 판단&제어(path\_planning)
  - 가. 경로 선택
  - 나. 거동 판단
  - 다. 속도 제어

## IV. 실험계획 및 결과(experiment)

## V. 참고문헌(referencing & citation)

## I. 서론

2016년, 알파고와 이세돌의 대국을 통해 인공지능이 세계의 주목을 받았다. 현재 인공지능이 가장 두각을 나타낼 수 있는 산업분야가 바로 자율주행차이다. 인공지능이 탑재된 자율주행차는 인터넷, 스마트폰을 잇는 게임체인저가 될 것으로 기대되고 있다. 자율 주행기술은 차량 소유와 공유의 패러다임을 가져올 핵심적인 기술로, 최근 공유 자동차 업체들의 가치가 크게 상승하는 이유이기도 하다. 공유 자동차 업체인 우버를 필두로, 플랫폼 업체인 구글과 애플, 자동차 제조업체인 GM 등 많은 기업들이 자율자동차 기술 개발에 매진하고 있다.

국제자동차공학회(SAE)에 따르면 자율주행의 정의는 여섯 단계로 나뉜다. 0단계인 비자동화 단계에서 5단계인 완전자동화 단계까지 있으며, 현재 상용화된 자율주행은 2단계인 부분적 자율주행과, 3단계인 조건부 자율주행이다. 2018년 구글 웨이모가 4단계 고급 자율주행 개발에 성공하였고, 많은 업체들이 2020년대에 5단계 완전 자동화 단계에 들어설 것이라 예상하고 있다. 5단계 자율 주행차의 출현은 우리 삶에 많은 변화를 불러일으킬 것이다. 교통체증이 감소하여 쾌적한 교통 환경을 누릴 수 있게 되고, 교통사고도 크게 줄어들어 많은 인명 피해를 줄일 수 있을 것이다.

동국대학교 MACARON팀은 동국대 기계공학과 소모임 MAC(기계 자동화 소모임)에서 파생된 팀이며, 수준 높은 자동화 기술 연구 및 구현을 위해 본 대회에 참여했다. 우리 팀은 첫 출전으로, 자율주행에 대하여 직접적으로 배우지는 않지만, 그 동안 공부해온 로봇 관련 지식을 바탕으로 자율 주행을 구현할 것이다.

대회의 우승을 위해, k-city 탐사 후, macaron 차량은 다음과 같은 목표를 중점으로 설계했다.

- 9월 대회를 위한 방열성능 만족
- HD맵을 이용한 경로생성
- 방열성능과 PC 메모리성능 상한에서 메모리 스케줄링
- 도시환경에서의 local path planning
- 계층화된 노드를 이용한 협업 효율성 증대

k-city의 이번 미션은 HD맵과 차량이 가야할 경로가 있어 global planning에 대한 부분과, 주행로 내의 차량이 없어 local planning 파트에서 트래픽 예측 부분에 대한 메모리 부하를 줄일 수 있었다. 즉, 도시환경에서 주행에 필요한 알고리즘보다는 offroad에서 적용되는 차량주행 알고리즘으로 local

planning에 관련논문을 조사하며 대회를 준비했다. 범주를 크게 나누면, local planning에서 이용하는 알고리즘은 DWA, VFH+, grid-based approach, potential field, discrete optimization 등이 있었다.

DWA<sup>[1][2]</sup>의 경우, 현재 차량의 속도와 차량이 쓰로틀링을 최대로 했을 때, 차량이 낼 수 있는 속도와 각속도 영역을 설정한다. 이를 Dynamic Window 라고 하고 이 동적 창 내에서 모든 각속도와 선속도를 계산하여 목적함수가 가장 높은 속도를 선택한다. 대개 목적지에 대한 방향, 속도, 충돌에 대해서 가중치를 매겨 계산한다. 하지만 이 방식의 경우, 차량의 모터와 차량의 물성치를 모르면 차량의 속도/각속도를 계산하기 위해 필요한 연산량이 많아질 뿐더러 파라미터 튜닝하기에 명료하지 않은 단점이 있었다.

potential field 장애물로부터의 거리가 가까울수록 강한 척력을 만들고 목적지 방향으로는 인력을 생성하여 장애물을 자연스럽게 피해가게 만드는 알고리즘이다. 예를 들어, 장애물이 좌측에 있으면 스티어링은 장애물에 의한 가상의 척력에 의하여 오른쪽으로 꺾인다. 연산량도 적고 파라미터 튜닝이 명료하지만 국부최소화 문제를 피할 수 없다. 단순 차선추종 및 장애물 회피를 위해서 자율주행 알고리즘에 적용되는 사례도 있으나<sup>[3]</sup>, 양 차선이 잘 정의 되어 양쪽으로 척력이 적용되어야 하고, 지속적으로 인식해야 하기 때문에 교차로를 통과해야 하는 macaron에는 적절치 않다.

VFH<sup>+</sup><sup>[4][5]</sup>는 로봇 주변의 360도를 나눠서 polar histogram을 만들어서 그 히스토그램에 장애물이 있는지 판단한다. 특정임계치를 정해서 그보다 크면 갈 수 없는 방향이라고 판단하고, 임계값보다 작은 방향 중 점수가 가장 큰 방향으로 이동하는 방법이다. potential field는 매번 map에 대한 정보를 갱신하는데 반해, VFH는 gridmap에 이전 정보를 저장시켜서 다음 경로 선택에 참조 할 수 있는 장점이 있다. 또한 polar histogram이나 gridmap의 정보를 조절하여 로봇을 하나의 질점이 아닌 부피를 고려하여 설계를 할 수 있는 장점이 있다. 대신 이 방법도 탐지거리 설정에 따라 국부최소화를 피할 수 없는 문제가 발생하고, 충돌 외에도 고려할 게 많은 도심환경에서는 적용이 어려운 단점이 있었다.

grid-based approach로는 A\*와 D\* 등을 찾을 수 있었다. 시작점과 경유점까지 거리와 경유점과 도착점까지 거리의 합을 목적함수로 삼아서 목적함수의 합이 최저가 되는 경로를 탐색하는 알고리즘이다. 그러나 점유격자지도는 라이다값은 누적시켜 구하므로 이동하는 객체가 많은 환경에서는 이용이 어렵고 실시간으로 경로연산을 하므로 저속 환경에서는 적절할 수 있으나, 고속환경에서는 적용이 어려울 수 있다. 또한 차량의 거동은 nonholonomic 하

지만 gird-based approach에서는 이를 고려하기 어렵다.[6][7]

discrete optimization는 스탠포드의 로봇차 Stanley에 적용된 방법으로, 차량에 차가 갈 수 있는 경로 후보를 주고 그중 점수가 가장 높은 것으로 가는 방법이다. 이때, 경로후보는 3차식의 형태로 차량의 거동을 나타내고, 이 3차식의 길이, waypoint와의 거리, 곡률 등을 고려하여 최적의 경로를 정하는 방법이다. 경로의 개수가 정해져 있어 계산이 효율적이고, gird-based approach는 주차장이나 복잡한 경로에 강한 반면, 3차식의 구조는 도로에 유리하므로 해당 알고리즘을 macaron의 메인 알고리즘으로 선정하였다.[8] 이때, 주로 참조한 논문은 offroad 기반이기 때문에 장애물여부, 경로길이, 곡률, 기존 경로와의 차이 등으로 가중치를 결정했으나, 도로에서는 차선과 교통신호를 고려하여 가중치를 결정하는 방식으로 개선하였다. 그 결과는 3차식의 구현방법 및 실험 결과는 계획부분에서 후술한다.

## II. 자율주행자동차 아키텍처

### 1. 플랫폼과 센서의 Layout

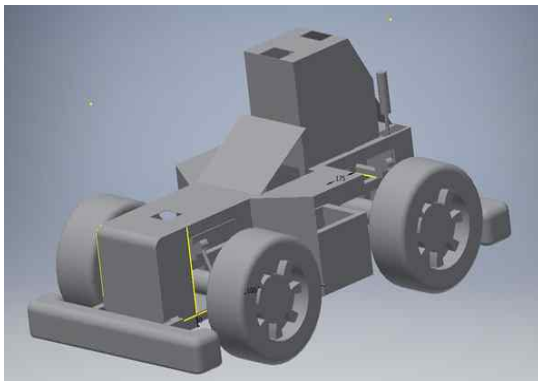


그림 1

카울설계는 (그림1), (그림2)와 같이 설계했다. 카울 디자인을 할 때에 가장 중요한 요소는 프로파일 및 센서의 위치였다. 카울이 센서의 동작에 영향을 주지 않도록 탐지 범위를 고려하여 작업하였다. 그리고 낮은 차고에 의해 신호등과 점선 차선이 잘 보이지 않는 현상을 방지하기 위해 높게 솟은 프로파일



그림 2

디자인 요소를 이용하고자 전기차 포물러 대회의 디자인을 참고하였다. 또한 주행 시 차량이 열을 받아 센서와 컨트롤러가 작동온도 이상의 열을 받는 현상을 방지하기 위해 통풍구와 경첩을 달고, 밝은색 계열로 도색했다.

프로파일 및 센서 마운트 설계에 있어서 가장 중요한 고려사항은

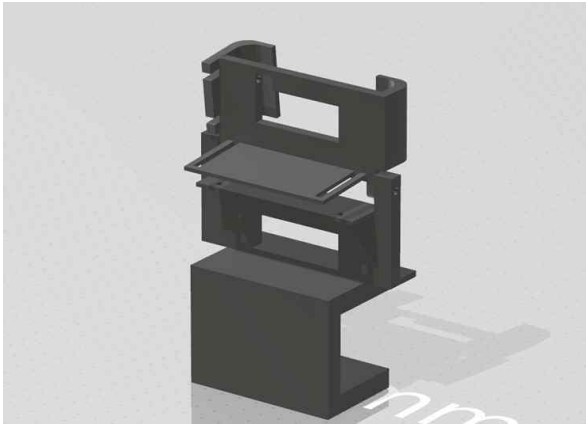


그림 3

센서의 개수, 종류 및 위치였다.

macaron은 차량에 웹캠 2개, 스테레오 카메라, GPS module, 라이다, 노트북, 인버터 등을 탑재시켰다.

웹캠(그림3)은 전방 감지와 후방 감지를 위해 두 개를 배치하였는데, 카메라의 화각을 고려하여 프로파일 최상단의 앞뒤에 배치하였다. 그리고 실제 주행을 통해 최적의 카메라 각도를 수시로 변경하기 위해

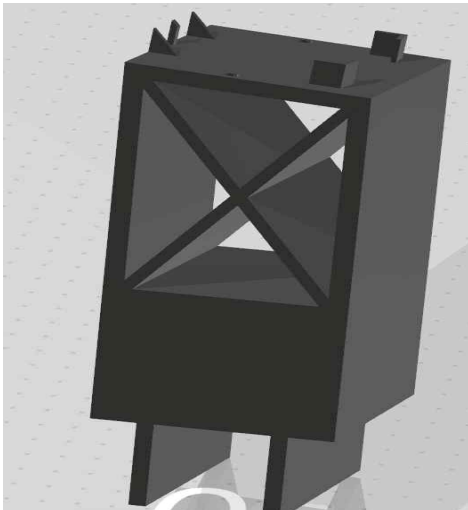


그림 4


서 웹캠이 프로파일에 단단히 고정되어있되, 각도를 쉽게 변경하고 고정할 수 있도록 센서마운트를 설계하였다.

스테레오 카메라(그림3)의 경우에는 내부에 INS가 내장되어 있고 정면에서 신호등 및 표지판을 구별해주는 역할을 담당하기 때문에 정면을 보도록 고정시켰다. GPS는 플랫폼의 회전 중심에 배치하였으며 카울 외부에 배치하여 외란을 최소화하였다.


그리고 라이다 마운트(그림4)의 경우에는 기존에는 플랫폼 가장 앞부분에 배치하였으

나 범퍼와 가까워 정면 충돌 시의 파손 가능성, 그리고 플랫폼에 가려 측면 감지가 불가능 하다는 점을 고려하여 플랫폼 위 앞부분으로 위치를 변경하여 설계하였다. 그리고 라이다가 10도 정도 아래를 보게 설계하여 라이다의 intensity 검출을 통한 차선인식에 사용할 수 있도록 하였다.

## 2. 센서 장착 설계

사진	제조사/ 모델명	주요 스펙	추가 구성품
	sick	<b>Accuracy:</b> 80mm <b>Resolution:</b> 0.33도 <b>Input voltage :</b> 9~28V <b>감지범위 :</b> 0.05m~25m, 270도 <b>Rate :</b> 최대 15Hz	이더넷 케이블 전원 케이블 micro 5핀 usb 케이블
	tim571		

지원받은 예산 내에서 3D 라이다의 구매가 불가능하여 2D 라이다를 구매하여 이용하기로 결정했다. 라이다의 주요 역할은 전면 장애물의 탐지와 지면의 intensity의 감지이므로, 2D 만으로도 충분히 주행이 가능하다고 판단했기 때문이다. 해당 제품의 Ros 파라미터 조절을 이용하여 센서값에 중간값 필터를 적용하여 outdoor 환경에서 강인하게 탐지할 수 있도록 설정했다. 라이다의 데이터는 추정경로상의 점만 추출하여 추정경로의 접근 가능성을 조사한다. 또한 도로의 intensity값을 얻기 위해 라이다의 각도는 10도 낮추기로 결정했다.

사진	제조사/ 모델명	주요 스펙	추가 구성품
	U-BLOX	<b>Accuracy:</b> 10mm 단위(X,Y,Z 3차원) <b>Convergence time :</b> 10초 내외 <b>Rate :</b> 최대 20Hz 5개의 통신 인터페이스 지원 짧은 time-to-first-fix 시간	C-type cable FTDI cable Balancer connector Interface cable Antenna ground plate Multi-band magnetic antenna
	GPS-RTK2 BOARD ZED-F9P		

U-Blox 사의 기존 제품 NEO-M8P에 비해 조사 가능한 자료는 없었으나, 출력Hz와 정확도가 거의 필터링이 없이도 이용 가능한 정도라고 판단하여 이 제품을 선택했다. RTK gps 단독으로 z방향에 대한 제어까지 하면 정확도가 부정확 할 수 있었으나, 이번 미션에서는 높이의 변화가 근소하여 HD맵과의 정확도 비교 후 이용하기로 결정했다. 그 외에 SMA to U.FL 케이블, 안테나 그라운드 플레이트를 함께 구매하여 GPS 위성 외의 다른 위성신호까지 받아서 정확도를 높였다. 특히 통신 인터페이스를 시리얼로 3개까지 지원



해서 한 개의 시리얼 포트로는 RTCM 데이터로 정확도를 보정하고 나머지 포트로는 ROS로 NMEA 데이터를 전송하도록 프로그래밍 했다.



		
SMA-to-U.FL 인터페이스 케이블 (Interface Cable SMA to U.FL)	GNSS 멀티밴드 GPS 안테나, SMA 자석, L1/L2 GPS (GNSS Multi-Band Magnetic Mount Antenna)	GPS 안테나 그라운드 플레이트 (GPS Antenna Ground Plate)


사진	제조사/모델명	주요 스펙	추가 구성품
	WITHROBOT	스테레오 영상이미지와 IMU 관성데이터 동시제공	화각렌즈 (4종 표준 M12 보드렌즈) USB 3.0 Micro B 케이블
	oCamS-1CGN-U	USB3.0 SuperSpeed 인터페이스지원 최대 2560x720 60fps 이미지 고속전송	

리얼센스 카메라는 3D 라이다로 취급되나 이 제품은 스테레오 카메라이므로 초기에 3D 라이다 사용을 고려할 때 부착했다. 리얼센스에 비해 거리에 대한 정확도는 낮으나, 렌즈를 교체할 수 있는 점이 메리트였다. 추가로 구매하여 부착할 수 있는 렌즈를 이용하여 화각을 조절해서 신호등, 판단에 필요한 정보가 탁 트이게 바꿀 수 있었다.


또한 한국제품이기 때문에 기술 지원이 가능 하고 ROS 환경을 지원하기 때문에 사용하기 편리했다. 해상도는 리얼센스보다 낮았지만, 대신 연산량의 관점에서 depth-camera의 영상을 사용하기 때문에 컴퓨터의 연산량 부하를 낮출 수 있었다.

사진	제조사/ 모델명	주요 스펙	추가 구성품
	logitech	최대 1920 × 1080 픽셀 30 fps 지원 90° 시야각 RightLight 2 기술 오토 포커스 1080p에서 4배 줌	외부 프라이버시 셔터
	C930E BUSINESS WEBCAM		


저렴한 가격에 고해상도의 영상을 얻을 수 있어서 이 제품으로 선정했다. 기존에 이용했던 logitech C525 제품에 비해 화각과 화질이 훨씬 넓었다. 작년 대회는 차선이 잘 정의되어 있어 C525의 69도 화각에 720p로 충분히 차선 인식이 가능했지만, 이번 대회에서는 차선 변경이 가능한 점선이 있고, 교차로에서는 차선인식이 거의 되지 않았다. 따라서 정말 화각이 넓은 카메라로 차량 뒤편에서 봐야 차선이 잘 검출될 것이라 판단하여 위치를 전술했던 곳으로 결정했다.

사진	제조사/ 모델명	주요 스펙	추가 구성품
	msi	코어i7-9세대(인텔) 커피레이크-R i7-9750H 2.6GHz(4.5GHz) 헥사 코어, RTX2060 32GB RAM USB 3.1, Type-C	마우스 180W 어댑터 충전기
	GL73 9SE		

데스크탑을 차량에 부착하면 낮은 가격에 더 좋은 작업 능력을 갖출 수 있으나 차량의 전력이 1900Wh로 충분하지 않은데 데스크탑은 높은 전력을 요구하므로 모바일에서 이용하기에 적절치 않다고 판단하여 노트북을 구매하기로 결정했다. 또한 뜨거운 k-city에서의 컨트롤러의 발열 문제에서도 비교적 자유로웠다. 특히 해당 제품은 쿨러를 수동으로 가동 시킬 수 있고, 데스크탑에 비해 아웃도어에 맞게 설계되어 해당 대회에 적절한 제품이었다. 또한 차량의 구동정보를 rosbag으로 저장시킨 후, k-city나 실험장소의 센서 데이터나 구동정보를 재생 시킬 수 있기 때문에 차량과 떼어서 데이터 실험을 할 수 있었다.

사진	제조사/ 모델명	주요 스펙	추가 구성품
	이지넷 유비쿼터스	속도 : 5Gbps 인터페이스 : USB3.0	USB 3.0 케이블 DC 5V/5A 전원아답터
	NEXT-UH308	포트수 : 8포트 연결방식 : 유전원(어댑터포함) 단자타입 : Type A 동작표시LED 고속충전기능	

노트북의 한계로 포트확장이 가능한 허브가 필요했다. 해당 제품은 USB 3.0 속도를 지원하고 총 7개의 포트로 비교적 보내는 데이터의 개수가 적은 센서를 부착하였다.(GPS RTCM, GPS NMEA, 라이다, ERP42, 무선랜카드, 노트북 쿨러) 그 외의 카메라와 ocams의 데이터는 남은 시리얼 포트에 받았다.

사진	제조사/ 모델명	주요 스펙	추가 구성품
	피엔케이 하이테크 DARDA	정격출력 : 600W 정격효율 : 91%	
	DK-4806	온도범위 : -35°C ~ 74°C 제품 무게 : 2.3kg	

노트북의 쿨러전원과 노트북 부스트 모드를 위한 전원공급을 위해 구매하였다.

### Ⅲ. 자율주행자동차 제어 로직 개발

#### 1. 통합 알고리즘 개요

본 연구팀은 가장 주축이 되는 개발환경으로 리눅스에서 작동하는 ROS (Robot Operation System)을 선택하였으며 이유는 다음과 같다.

##### (1) 안정성

리눅스는 윈도우의 블루 스크린과 같은 시스템 충돌에 대한 가능성 매우 낮아 다양한 센서와 연결하며 생길 수 있는 문제에 대한 해결하는 시간을 절약할 수 있다. 또한 시스템이 다운됐을 경우 즉시 복구할 수 있는 저널링 파일 시스템을 지원하여 작성한 코드가 사라지는 불상사를 미연에 방지할

수 있다.

## (2) 높은 호환성

리눅스와 ROS 모두 오픈소스로서 많은 개발자 층을 지니고 있기에 많은 하드웨어 회사에서 리눅스나 ROS에 호환이 가능한 프로그램을 지원하여 해당 하드웨어와 통신 프로토콜에 대해 적은 시간의 투자로 빠른 개발이 가능하다. 현재 사용 중인 라이다와 스테레오 카메라 등이 제공 중인 코드를 이용하여 빠르게 접목시킬 수 있었으며, 이외에도 OpenCV와 같은 다양한 알고리즘에 대한 작동 노드들이 ROS 커뮤니티에 공유되고 있어 개발된 모듈을 가져다씀으로서 아이디어를 실현시키는데 들어가는 개발 시간을 줄일 수 있다.

뿐만 아니라 각 기능을 모듈화 하여 각자 하나의 노드로 개발한 뒤, 합침으로서 분산된 기능 개발을 통해 빠르게 중요 기능을 완성시킬 수 있으며, 이를 통해 수정과 구조 파악에 용이하다.

## (3) 편리한 툴과 다양한 예제

ROS에서 제공하는 메시지 규약을 통해 노드(쓰레드) 간의 데이터 통신에 대한 구상을 별도로 할 필요가 거의 없으며 해당 메시지와 사용 예제를 봄으로써 빠르게 센서 값을 다루는 방법을 배울 수 있다. 또한 정해진 메시지 규약에 따라 시스템에 업로드 할 경우, RVIZ나 RQT와 같은 시각화 도구를 이용하여 정상 작동 여부와 데이터 값을 바로 확인할 수 있다.

## (4) 강력한 백업 기능

윈도우 상에서 얻은 데이터를 기록하기 위해서 txt파일로 변환하여 여러 개의 파일을 참조하고 timestamp와 센서 값을 매칭하는 프로그램을 만들어야 하는 것에 비해 ROS에서 제공하는 ROSBAG은 토픽명만 입력해주면 자동적으로 기록해 준다. 따라서 해당 백업 프로그램을 제작하는 개발 시간을 절약할 수 있으며, 용량만 충분하다면 필요로 하는 데이터를 하나로 묶어서 모두 기록할 수 있다는 장점이 있다.

이렇게 계측된 데이터는 Rviz를 통해 시각화하여 로봇의 판단과 주변 상황에 대한 재구성이 가능하여 한 번의 계측에 여러 알고리즘을 적용하고 최적화함으로써 비용과 시간, 체력을 모두 절약할 수 있었다. k-city에서의 직접 실험은 온도와 실험환경, 먼지 등 노트북에 우려가 되는 환경이었으나, rosbag을 해준 파일을 재생하면서 개발을 좀 더 편하게 할 수 있었다.

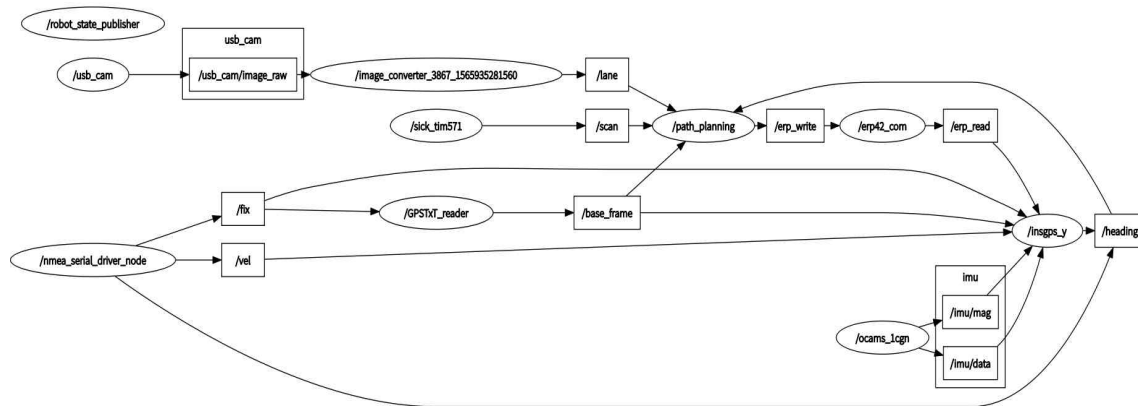


그림 5 - rqt 그래프

위의 그래프는 rqt 그래프로, 각각의 노드의 관계와 메시지의 형태를 표현한다.  
macaron\_deviceon : 센서켜기

nmea\_parser : GPS센서값 파싱(10Hz)

in : RTCM3, NMEA

out: 위도, 경도, 속도, heading, 데이터의 분산

erp42\_com : 차량 제어/상태 파싱(50Hz)

in : 속도, 조향각, 기어, 브레이크

out: 차량상태 출력 및 파싱

ocams\_ros : IMU값과 스테레오 카메라의 영상(10Hz & 100Hz)

out: 상향 카메라 raw\_image, imu\_data

sick\_tim571 : 라이다 데이터(15Hz)

out: 거리, 각도, intensity

usb\_cam : 하향 카메라 로우이미지 퍼블리싱(30Hz)

out: 하향 카메라 raw\_image

macaron\_sensing : 센서값을 받아서 유용한 값으로 변화

lanetracker : 하향카메라를 통해 얻을 수 있는 정보추출(30Hz)

in : usb\_cam이미지,

out: 차선중심간 거리, 좌우차선 곡률, 중앙선의 좌표

차선에러 유형, 차선 fairing 상태, 차선변경 가능성

insGPS : IMU데이터와 GPS 데이터의 융합(100Hz)

in : GPS, IMU, erp\_com data

out: 글로벌에서 빠른 주기의 좌표와 속도

GPSTxtT\_reader : 예상경로 추종 관련 정보 생성(10Hz)

in : 현재 차량의 tm 좌표

out: 현재 위치에서 가장 가까운 pre-defined 된 좌표탐색  
xy좌표의 에르미트 변환→x(s) & y(s) 좌표 생성  
lookahead point에 따른 10m 뒤 예상좌표 탐색

obstacle\_reader : 충돌관련 정보 생성(15Hz)

in : 라이다값, 현재위치

out: 로컬 좌표계에서 읽힌 라이다값을 글로벌 좌표계로 변형

macaron\_assemble : 유용한 값을 받아서 경로생성, 선택, 속도제어

path\_planning : macaron\_sensing을 이용한 경로 생성/선택(20Hz)

in : lanetracker, insGPS, gpstxt, obstacle\_reader

out: generate\_path, choose\_path

Mission\_identifier : 표지판 및 신호등인지(10Hz)

in : 상향카메라 raw\_data

out: 초록,빨강,교차로,좌/우회전,횡단보도,안전구역,방지턱,주차

vel\_planning : 선택한 경로로 차량이 주행할 수 있도록 조향각과 속도를 선택한다.(50Hz)

in : 선택한 경로로 진행하는 방향으로 달리되, Mission\_identifier의 정보를 이용하여 속도의 임계값 조절

## 2. 인지

### 가. Localization

Localization 로직은 현재 차량의 현재위치, 방향, 속도 등의 차량의 상태를 여러 개의 센서들을 활용하여 정확하게 측정할 수 있도록 해준다. 차량의 구동 및 제어에 있어서 차량의 현재 상태 값은 가장 중요한 입력 값 중 하나이기 때문에, 대회 준비기간의 상당시간을 이 로직에 투자하였다.

#### (1) GPS

##### - OS: Ubuntu16.04 (Linux)

GPS 데이터를 ZED-F9P를 이용하여 받기 위해서 GPS에 초기 setting을 넣어주어야 한다. 이를 위해 제조사에서 제공하는 프로그램인 U-center을 이용하여 GPS에 setting 값을 입력해주어 다음과 같은 화면을 통해 데이터 수신을 확인할 수 있다. GPS 에 입력되는 값들은 기본적으로 데이터 출력속도

(Hz)와 순수 nmea 신호가 있으며, ntrip client를 통해 위치 정확도를 높여주는 rtcm 보정 설정도 보낼 수 있다.

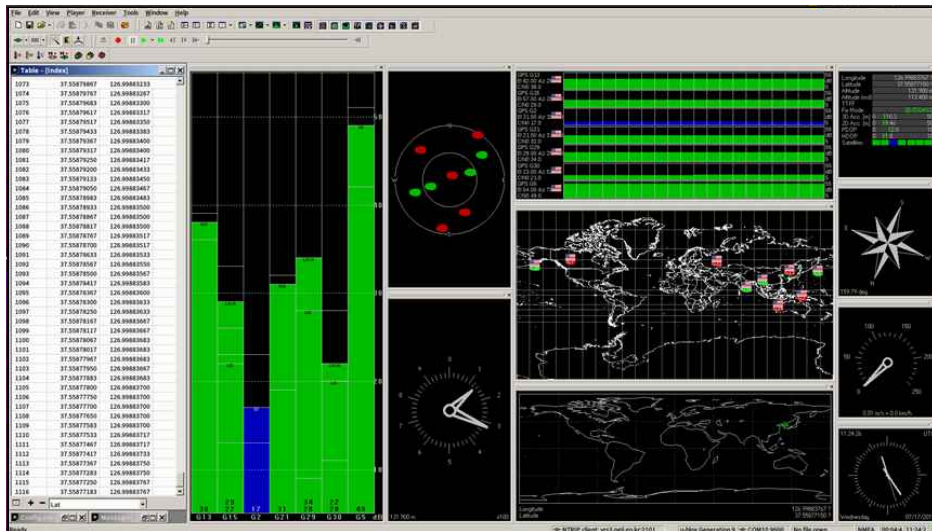


그림 6 - U-center 프로그램 세팅 화면

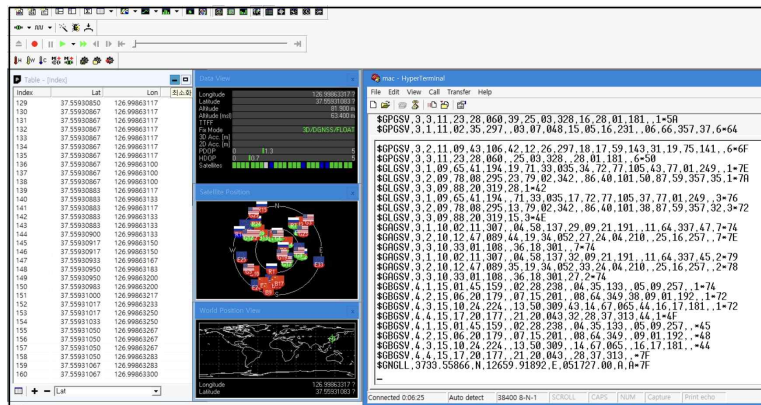
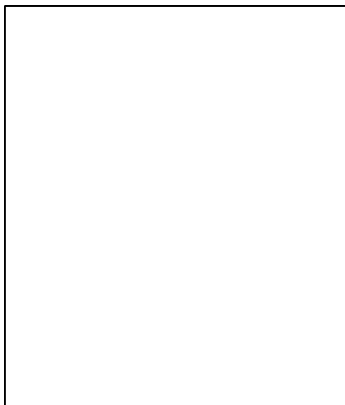


그림 7 - FTDI 케이블(좌),  
USB-C 케이블(우)

그림 8 - u-center (좌), 시리얼 통신 화면(우)

## - 포트 설정 : FTDI 케이블, USB-C 케이블

해당 프로그램이 아닌 ROS 플랫폼의 패키지를 이용할 경우, GPS setting 값의 업로드가 불가능하다. 제공 프로그램을 사용할 경우, 한 개의 usb 포트에 두 가지 프로그램이 접근할 수 없으므로 업로드는 USB-C type 케이블을 이용한 포트, 데이터 수신은 FTDI cable을 통한 시리얼 통신 포트에 이루어지며, 프로그래밍 된 serial노드를 통해 ROS 시스템에 데이터를 입력시킨다.

## (2) IMU

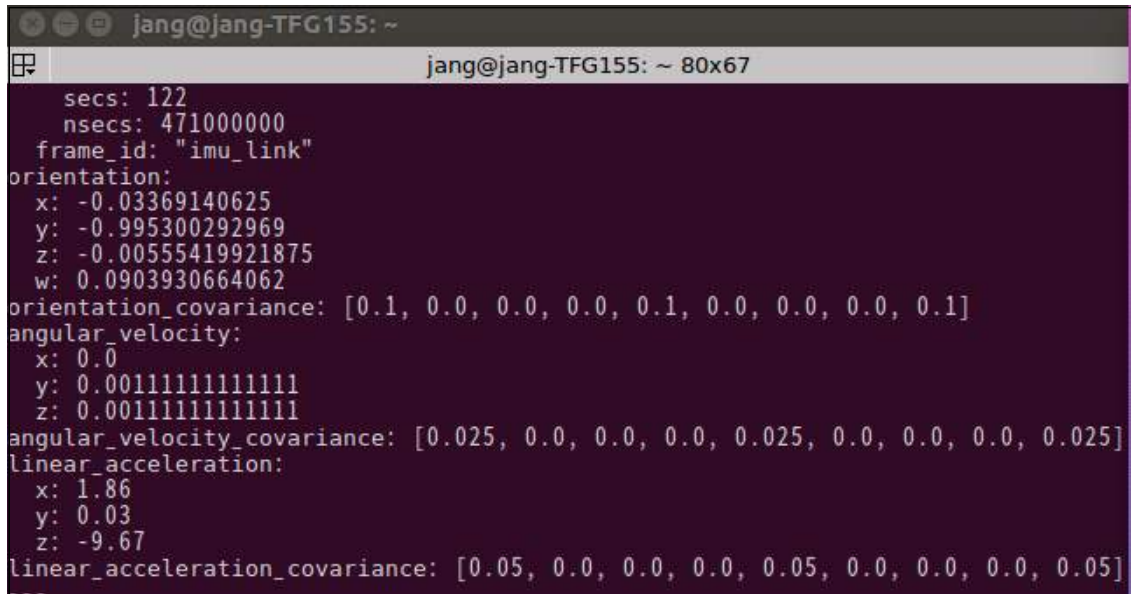
### - OS: Ubuntu16.04 (Linux), ROS

oCamS-1CGN-U는 ROS가 돌아가는 리눅스 환경에서 작동할 수 있는 패키지



지를 제공하며 이를 동작시키면 각각 세 개축의 Gyro 성분, Accl 성분, Magnetic 성분을 topic에 publishing 하게 된다. 스테레오 카메라와 IMU 센서 데이터가 같이 들어오기 때문에 USB3.0이 권장되며, 해당 포트의 데이터는 withrobot사의 드라이버를 통해 가상화 포트를 이용하여 분리하게 된다.

센서의 값은 Euler 각과 Quaternion 각을 세팅을 통해 받아볼 수 있으며 회전에 대한 자유도를 잃는 Gimbal-Lock 현상을 방지하기 위해 쿼터니언으로 ROS상에 받게 설정되어있다.



```

jang@jang-TFG155: ~
jang@jang-TFG155: ~ 80x67
secs: 122
nsecs: 471000000
frame_id: "imu_link"
orientation:
x: -0.03369140625
y: -0.995300292969
z: -0.00555419921875
w: 0.0903930664062
orientation_covariance: [0.1, 0.0, 0.0, 0.0, 0.1, 0.0, 0.0, 0.0, 0.1]
angular_velocity:
x: 0.0
y: 0.00111111111111
z: 0.00111111111111
angular_velocity_covariance: [0.025, 0.0, 0.0, 0.0, 0.025, 0.0, 0.0, 0.0, 0.025]
linear_acceleration:
x: 1.86
y: 0.03
z: -9.67
linear_acceleration_covariance: [0.05, 0.0, 0.0, 0.0, 0.05, 0.0, 0.0, 0.0, 0.05]
---
```

그림 9 - ros 패키지를 통해 출력한 imu 데이터

### (3) 주행 경로

정확한 경로 생성을 위해 국토지리정보원(<https://www.ngii.go.kr/kor/content.do?sq=210>)에서 제공하는 고정밀 지도(HD map, 그림10)를 활용하였다. HD map은 K-city 도로와 주변 시설 정보를 정확히 표현해주며 다양한 기준좌표계로 변환이 가능하여 유용하다. 이 중 GRS80(1980) 타원체를 기반으로 한 TM좌표계(평면 직각좌표계, Transverse Meractor)로 변환하여 CAD에서 사용하였다. (그림 11)

GRS80(1980)은 현재 지구를 가장 잘 나타내고 있는 타원체이며, 대한민국의 경우 TM 좌표계를 국가기본도의 기본체계로 하고 있으므로 활용성과 호환성을 고려하여 이를 선정하였다.

주행 경로 GPS data는 경위도 좌표계를 기반으로 하므로 HD map과 호환을 위해 GRS80 타원체면상의 경위도 좌표계를 TM좌표계로 변환 한다. 필요



한 투영식은 아래 그림12을 참고하였다. 중부원점은 127W, 38N이며 X축(North), Y축(East) 원점 가산값은 각각 600000, 200000이다.

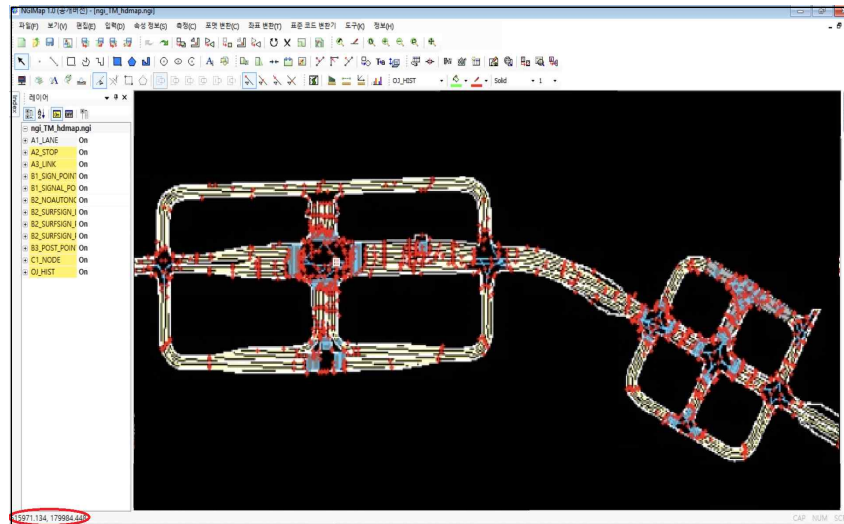


그림 10 - K-city 고정밀 지도(HD map)

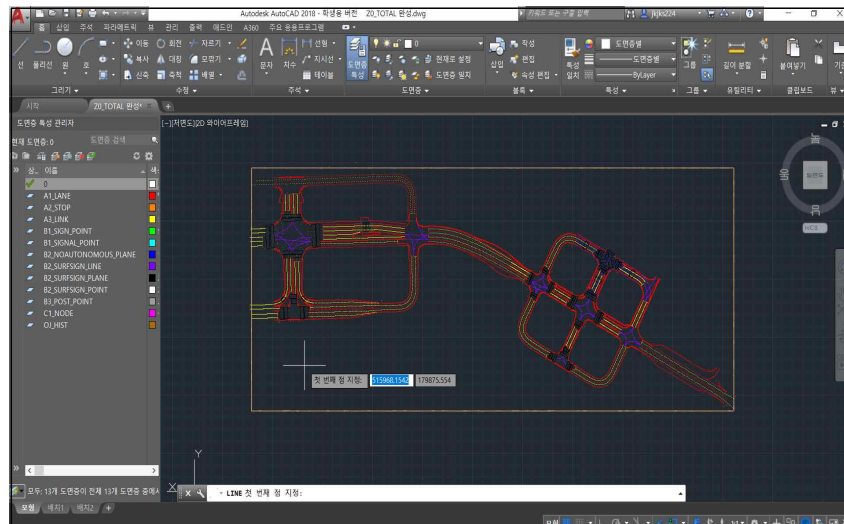


그림 11 K-city CAD 지도 (TM좌표)

## 세계측지계 변환에 필요한 투영식

$$Y(E) = \Delta Y + k_0 \cdot N \cdot \left[ A + \frac{A^3}{6} (1 - T + C) + \frac{A^5}{120} (5 - 18T + T^2 + 72C - 58e'^2) \right]$$

$$X(N) = \Delta X + k_0 \cdot \left\{ M - M_0 + N \tan \phi \cdot \left( \frac{A^2}{2} + \frac{A^4}{24} (5 - T + 9C + 4C^2) \right. \right. \\ \left. \left. + \frac{A^6}{720} (61 - 58T + T^2 + 600C - 330e'^2) \right) \right\}$$

여기서,

$$\textcircled{1} T = \tan^2 \phi$$

$$\textcircled{2} C = \frac{e^2}{1 - e^2} \cos^2 \phi$$

$$\textcircled{3} A = (\lambda - \lambda_0) \cos \phi \text{ (여기서, } \lambda \text{와 } \lambda_0 \text{는 radian값임)}$$

$$\textcircled{4} N(\text{위도 } \phi \text{에서의 모유선의 곡률 반경}) = \frac{a}{\sqrt{1 - e^2 \sin^2 \phi}}$$

$$\textcircled{5} M(\text{자오선장}) = a \cdot \left\{ \left( 1 - \frac{e^2}{4} - \frac{3e^4}{64} - \frac{5e^6}{256} \right) \phi - \left( \frac{3e^2}{8} + \frac{3e^4}{32} + \frac{45e^6}{1024} \right) \sin 2\phi \right. \\ \left. + \left( \frac{15e^4}{256} + \frac{45e^6}{1024} \right) \sin 4\phi - \frac{35e^6}{3072} \sin 6\phi \right\}$$

$$\textcircled{6} e^2(\text{제1이심률}) = \frac{a^2 - b^2}{a^2}$$

$$\textcircled{7} e'^2(\text{제2이심률}) = \frac{a^2 - b^2}{b^2}$$

그리고,

$\phi$ : 위도,  $\lambda$ : 경도,  $\phi_0$ : 투영원점 위도,  $\lambda_0$ : 투영원점 경도,

$a$ : 타원체 장반경,  $f$ : 편평률,

$b$ : 타원체 단반경( $=a(1-f)$ ),

$k_0$ : 원점축척계수,

$\Delta Y$ : Y축(East) 원점 가산값,

$\Delta X$ : X축(North) 원점 가산값

그림 12 - 세계측지계 변환에 필요한 투영식

### (4) 주행 실험

GPS data의 신뢰성을 확인하기 위해 실제 도로에서 플랫폼 실험주행 후, 순서대로 기록된 좌표 data (.txt)를 CAD 지도와 비교해 보았다. 사전 실험으로써 동국대학교 팔정도(그림13)에서 교내 주행을 시행하였고, 최종 실험으로써 K-city 예선(그림14), 본선 경로주행(그림15)을 각각 시행하여 data가 지도와 일치함을 확인하였다.

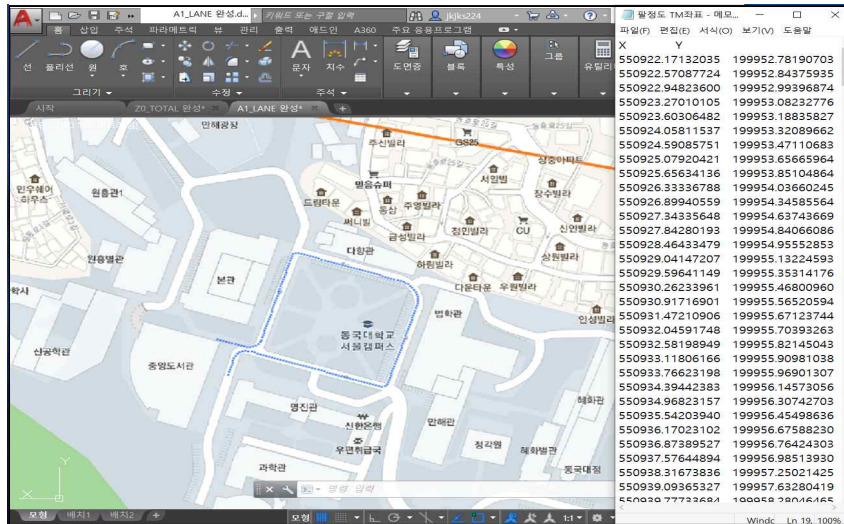


그림 13 - 교내 경로주행 결과

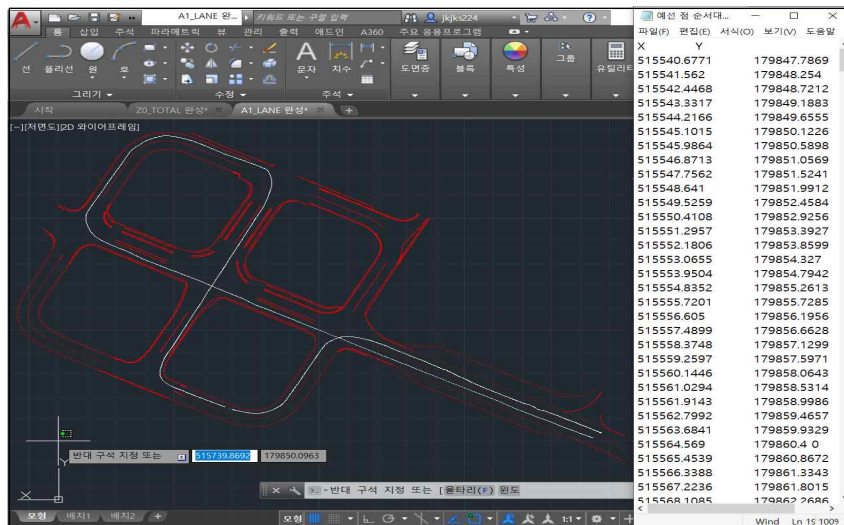


그림 14 - 예선 경로주행 결과

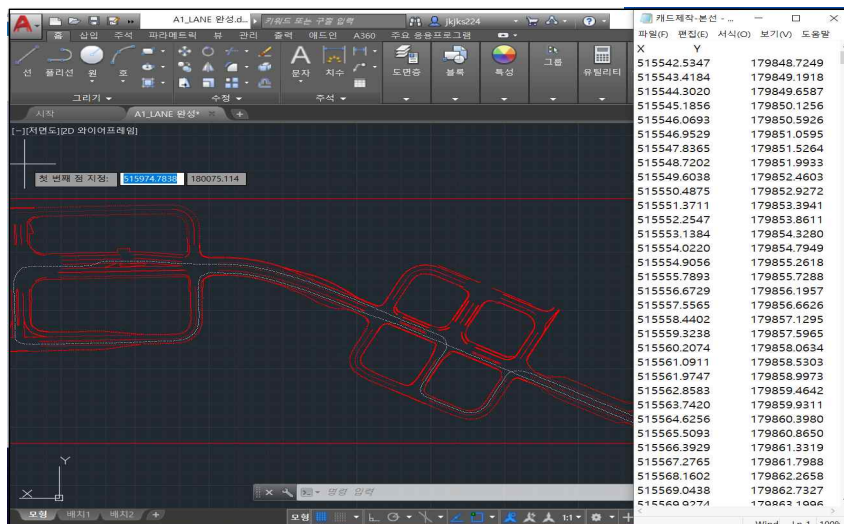


그림 15 - 본선 경로주행 결과

#### 4. Localization 알고리즘

(a) IMU sensor Raw data

##### - HDR (Heuristic Drift Reduction)

자이로 센서의 측정값은 잡음이 매우 적은 대신 미소한 오차가 일정하게 쌓이는 경향이 있다. 이러한 미소한 bias drift는 적분되어 결국 큰 오차로 이어지는 것을 막기 위해 이를 최소화하는 HDR 알고리즘을 적용하였다.

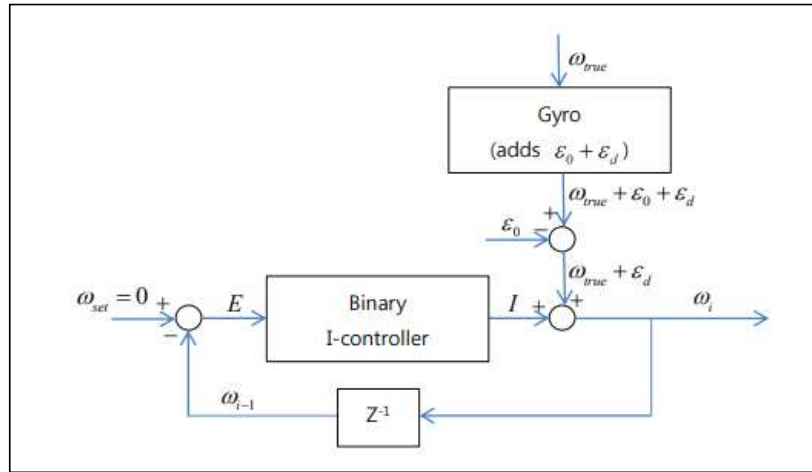


그림 16 - HDR 알고리즘 구성도

출처 : J. Borenstein “Heuristic reduction of gyro drift in gyro-based vehicle tracking”

##### - HSR(Heuristic Scale Regulation)

가속도 센서의 값에는 중력 성분이 포함되어 있다. 속도와 위치를 계산하기 위해서 중력을 제거할 경우, 미소 scale factor의 변화로 제대로 제거되지 않아 누적되면 큰 오차로 이어지기에 scale factor을 조절하기 위해 HSR알고리즘을 사용하였다.

##### HSR 알고리즘

$s$  = 중력가속도,  $\|\tilde{a}\| = \sqrt{\tilde{a}_x^2 + \tilde{a}_y^2 + \tilde{a}_z^2}$ ,  $i_c = 1.0001$

$$s \leftarrow \begin{cases} s i_c & \text{if } \|\tilde{a}\| > 1 \\ s / i_c & \text{if } \|\tilde{a}\| < 1 \end{cases}$$

( $\tilde{a}$  : 적용 전 가속도,  $a$  : 적용 후 가속도)

$$a = s \tilde{a}$$

이와 같이 두 알고리즘을 적용함으로써 Gyro와 Accl의 raw 데이터 잡음을 줄였다.

### (b) Attitude data (Roll, pitch)

센서에서 계측되는 자세 데이터는 사용하기에 정확도가 높은 값이나, 한 방향으로의 지속적인 미소 오차로 인해 중력 성분이 특정 방향으로의 가속도로 해석되는 문제를 해결하기 위하여 칼만 필터를 적용하였다.

필터는 Gyro 센서의 각 속도 측정값을 이용하여 다음 자세에 대한 추정값을 연산한 뒤, 가속도 센서의 데이터를 통해 Roll, Pitch에 대한 측정값을 입력하여 정확한 값을 얻어내었으며, 센서 계측값과 다르게 편향되지 않은 오차를 냄으로서 localization에 필요한 선형 속도를 얻어낼 수 있다.

### (c) Heading (Yaw)

Localizing 과정 중 가장 중요한 요소 중 하나로, Heading과 Yaw 두 가지로 분류되는데, Heading은 도북(grid azimuth angle)을 뜻하며, Yaw는 각 센서의 계측이 시작된다.[9]

우선순위	Heading	Yaw
1	GPS Heading (10Hz)	Line Yaw (10Hz)
2	HD map Heading (10Hz)	Encoder Yaw (50Hz)
3	magnetic Heading (100Hz)	IMU Yaw (100Hz)

Global 좌표계에 대한 자신의 위치와 자세에 대한 정보가 필요하므로 초기 Heading 은 magnetic heading을 이용하여 대략적인 자세를 추정하고 차선에 오를 경우, HD map의 Heading과 차선으로부터 이탈 각을 이용하여 자세를 추정한다. 주행하며 측정된 GPS Heading의 값이 안정되면 magnetic heading은 비활성화가 되며 해당 값을 기준으로 주행하게 된다.

그러나 차량의 장기간 정확한 진행 방향을 계측하는 것은 외란이 많은 현실 주행 환경 상, 많은 비용을 요구하게 되므로 GPS heading의 값이 부정확

해지면, HD map heading과 Yaw를 이용하여 보정해주며 주행하게 된다.

가장 중심이 되는 GPS Heading은 10Hz로 출력되며 Low pass filter을 이용하여 계측하면 안정되게 정확한 값을 얻을 수 있다. 정확도가 떨어질 경우, 이전 heading을 참조하게 되며 heading이 측정된 이후 IMU와 Encoder yaw에 대해 평균 필터를 통해 각의 차이를 계측하게 된다. GPS heading을 이용하여 주행할 경우 low pass filter을 Euler 각이 아닌 쿼터니언을 이용하여 잡음을 제거하며 정확도를 높였다.

IMU yaw의 경우 100Hz의 출력속도를 가지고 Yaw의 값을 출력하며 회전시, 감속에 따른 구심 가속도에 의해 다른 heading이나 Yaw에 비해 빠르게 돌아가 시간의 지연을 통해 사용해야하는 단점이 존재한다. Encoder Yaw는 no slip condition에서 정확하나, 도로 상황에 따라 부정확해질 수 있으므로 지연된 IMU yaw와 비교를 통해 사용된다.

#### (d) X-Y position

기본적으로 차량의 위치는 GPS와 속도, heading을 이용하여 추정하게 되며, GPS는 특정 시점의 좌표를, 속도와 heading은 rotational matrix를 이용한 변위를 기존 위치에 칼만 필터를 거쳐 추정하게 된다. 그러나 GPS에 수신되는 위치는 평균 10cm에서 최대 2m에 달하는 오차를 내포하고 있다. 칼만 필터를 이용하여 보정해주어도 결국 위치 정보에는 지속적인 GPS의 오차가 반영되므로 차선 위의 주행이 인지되면 경로 좌표를 통해 실제 도로의 좌표로 차량 좌표를 보정하게 된다.[10][11] 이때 경로 좌표는 주행 코스 중 heading이 일정 오차 안에 있는 최단 거리 노드를 이용하게 된다.

### 5. 노드 관계도

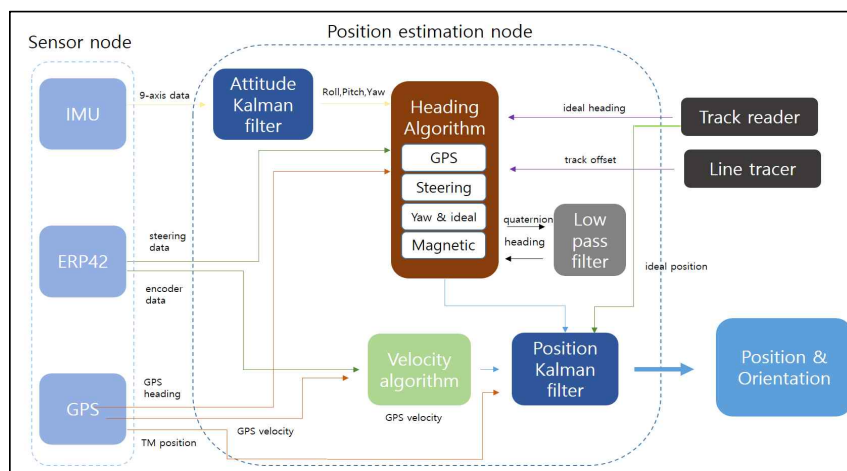


그림 17 - 노드 관계도

## 나. 차선 인식

### (a) 영상변환&투상변환

영상에서의 특징 추출이 더 용이하도록, 영상변환을 실시하였다. 위에서 바라본 시점으로 바꾼 후 차선이 영상에서 밝다는 점을 이용할 수 있도록 이진화된 영상을 검출하였다.

그 후, 변환된 영상의 축척을 100픽셀당 1미터가 되도록 바꾼 후 버드아이 뷰의 영상을 이용하여 실거리를 계산하였다. 투상변환은 일종의 좌표 변환이기 때문에 어떤 좌표로 바꿀 것인지 정해야 한다. 실제 차폭과 영상에서 보여지는 차폭의 비율을 통해 변환된 좌표를 구한다. 렌즈의 굴곡 때문에 카메라 렌즈의 바닥과 중심의 화각이 달라서 이에 따른 각각의 화면 차폭 비율을 구한다. 좌표를 구하는 순서는 다음과 같다.

카메라가 도로를 내려다 보는 각도  $\theta_{cam}$  은 영상에 나타낼 목표지점까지 거리  $d_{target}$  와 카메라가 설치된 높이  $h$ 로 구한다.

$$\theta_{cam} = \tan^{-1}\left(\frac{d_{target}}{h}\right) \quad (1)$$

카메라 화면중심에서 목표지점까지 직선거리  $d_1$ 는  $\theta_{cam}$  과  $d_{target}$  을 통해 구한다.

$$d_1 = \sqrt{\theta_{cam}^2 + d_{target}^2} \quad (2)$$

화면 중심에서의 비율  $R_2$ 은 실제 차폭  $w$ 와  $d_1$ , 화면 중심에서 가로 화각  $\theta_{m1}$ 을 통해 구한다.

$$R_2 = h * \tan\left(\theta_{cam} - \frac{\theta_{m2}}{2}\right) \quad (3)$$

카메라 화면바닥에 보이는 도로까지 직선거리  $d_2$ 는  $h$ 와  $\theta_{cam}$ , 화면 중심에서 세로 화각  $\theta_{m2}$ 을 통해 구한다.

$$d_2 = h * \tan\left(\theta_{cam} - \frac{\theta_{m2}}{2}\right) \quad (4)$$

화면 바닥에서의 비율  $R_2$ 은  $w$ 와 , 화면 바닥에서 가로 화각  $\theta_{b1}$ 을 통해 구한다.

$$R_2 = \frac{w}{d_2 * \tan\left(\frac{\theta_{b1}}{2}\right) * 2} \quad (5)$$

식 (3)과 식(5)를 통해  $R_1$ 과  $R_2$ 을 좌표를 구한 다음 투상변환한 좌표로 변환한 영상은 다음과 같다.



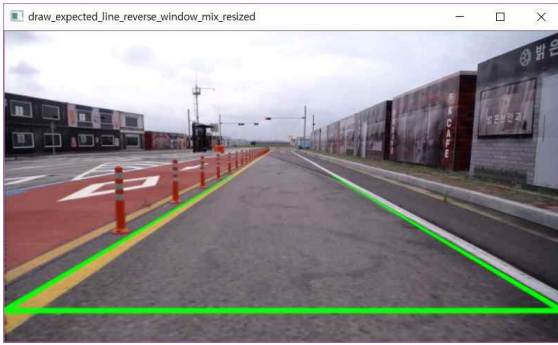


그림 18 - 투상변환 전 영상

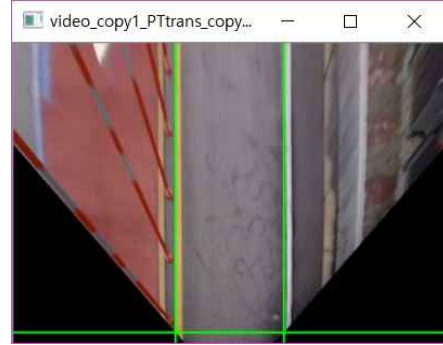
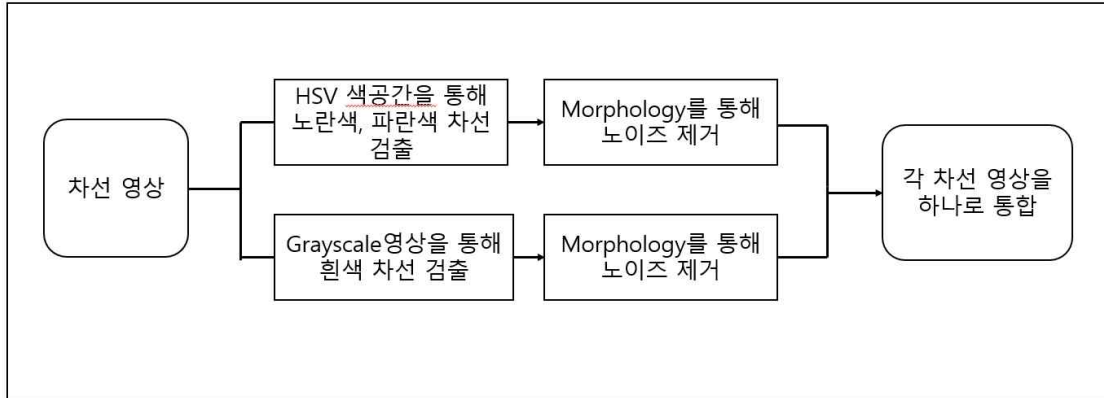


그림 19 - 투상변환 후 영상

## (b) 영상 이진화

차선 픽셀이 다른 픽셀에 비해 밝기 값이 높은 것을 이용하여, 영상을 이진화 한 후 차선과 나머지 부분을 구분하였다. 영상 이진화에 앞서, 노이즈 제거를 위해 Gaussian Filter를 이용하여 이미지에 블러 처리를 하였다. 차선의 색깔은 파란색, 노란색, 흰색이 있다. 영상을 이진화 하기 위해서는 임계값을 설정해주어야 하는데, 파란색과 노란색차선은 흰색차선과 임계값을 다르게 적용해야 잘 검출된다. 그렇기 때문에 파란색과 노란색 차선은 HSV 색 공간을 통해, 흰색 차선은 grayscale영상을 통해 검출하였다. 차선 검출 알고리즘을 요약하면 다음과 같다. [12][13]



태양의 밝기나 주변 그림자 및 환경의 변화에 따라 차선의 밝기나 색상, 채도가 달라지기 때문에 고정된 임계값으로는 다양한 환경에서 차선 검출이 어렵다. 이런 문제를 해결하기 위해 투상변환 화면에서 차선이 적절히 검출되었을 때의 차선의 픽셀 비율을 기준으로 두고, 검출된 차선 픽셀 비율과 기준 비율의 차이를 오차로 두어 PID 제어를 통해 상황에 따라 임계값을 찾아냈다.



### (c) 차선 검출

차선 인식을 위해 사용되는 값은 카메라가 차선 중심에서 떨어진 거리 (deviation), 차선의 기울기이다. 뿐만 아니라, 실선과 점선, 중앙선을 구분하였다.

#### - deviation, 차선의 기울기 계산

일정한 크기의 창으로 차선을 추적하는 Sliding-window를 이용하여 추세선을 구하고, 이를 통해 deviation과 차선의 기울기를 계산하였다. 순서는 다음과 같다.



그림 20 - 차선 이진화 영상

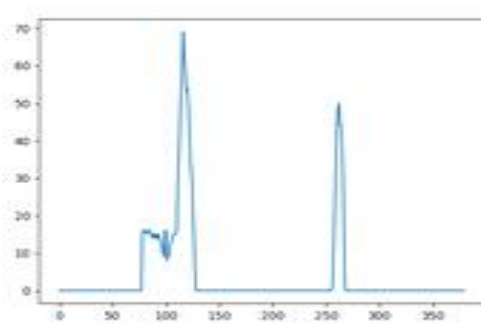


그림 21 - 차선에 대한 히스토그램

(b)에서 구한 이진화 영상의 첫 프레임 (그림 20)에서 2차원 배열의 세로 합을 구하여 히스토그램을 작성한다. (그림 21) 히스토그램에서 값이 가장 큰 좌표를 차선의 초기 x좌표로 지정한다.

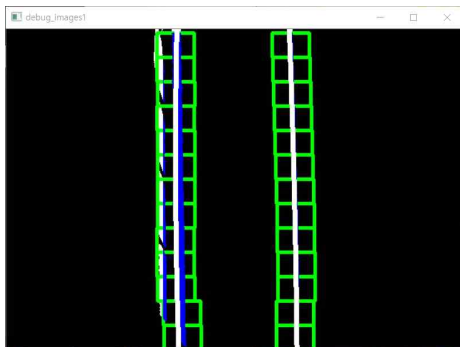


그림 22 추세선 검출 영상



그림 23 주요 추세선 검출 영상

초기 x좌표 지점으로부터 차선을 추적하는 Slide-window를 만들어서 추세선을 계산하기 위한 배열값을 감지한다.(그림 22)

배열값들의 좌표를 근사적으로 이어주는 초기 2차함수 추세선  $f_1(y)$ 을 계산한다.  $f_1(y)$ 으로부터 특정 거리 안으로 감지 영역을 지정해주고 이 영역 내의 배열값  $A_1$ 을 감지하여 주요 2차 함수 추세선  $f_2(y)$ 을 계산한다. 다음 프레임에 의 감지 영역 내 배열값  $A_2$ 값을 감지하여 현재 프레임의 차선 배열값  $A_3$ 을 통해 계산한 주요 추세선  $f_3(y)$ 을 구한다.

다음 프레임으로 넘어갈 때마다 마찬가지로 주요 추세선  $f_n(y)$ 을 구한다. (그림 23)각 프레임에서 얻어진 왼쪽, 오른쪽 차선의 이차함수 추세선을 통해 차선의 좌표를 구

한다. 구한 좌표로 deviation과 화면 전체에서의 차선 평균 기울기를 계산한다.(그림 24)



그림 24 - 차선 인식 및 차선 정보에 대한 결과 영상

#### - 실선, 점선 차선, 중앙선 구분

차선변경 미션이나 장애물 회피 수행하기 위해서 차선을 구분해야 한다. 이를 위해 투상 변환한 화면에서 왼쪽과 오른쪽 각각의 차선 영역만 검출된 영상의 2차원 배열의 가로 합을 구하여 1차원 배열  $M_1$ 을 구한다. 그 후,  $M_1$ 에서 0이 아닌 요소들은 모두 1로 변환하여 0(빈 공간)과 1(차선이 있는 공간)로 이루어진 배열  $M_2$ 로 변환시켰다.  $M_2$ 에서 1의 값들이 연속된 구간을 하나의 차선 마디로 간주하여 각 마디의 차선 길이를 계산한 후 가장 긴 차선 마디의 길이가 일정한 길이 이하일 경우 점선 차선으로 인식한다. 차선 이진화 영상에 HSV 색 공간을 통해 검출한 노란색 차선 영역이 포함되어 있으면 중앙선으로 인식한다.

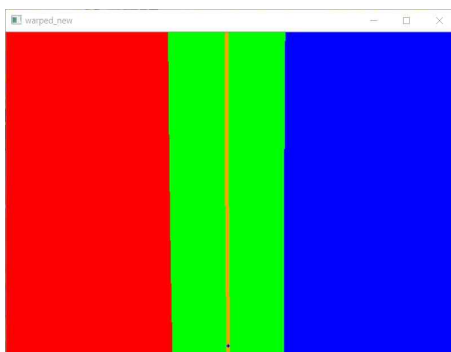


그림 25 - 영역 판단 및 분할



그림 26 - 분할 결과

(그림 25)와 같이 빨간색 영역은 가지 못하는 곳, 초록색은 지금 속한 곳, 파란색은 이동이 가능한 곳이다.

#### - 차선이 한 쪽만 읽히는 경우

장애물에 의해 한 쪽 차선이 가려지거나 차선이 퇴색되어 차선 영상이 검출되지 않을 경우 반대편 차선 영상을 이용하여 차선 영상이 검출 될 때까지 가상의 차선을 그려서 주행하게 하였다.

차선의 픽셀 비율이 특정값 이하일 경우 [한쪽 차선주행 모드]로 간주한다. 검출된 차선에서 구한 추세선  $f(y)$ 의 가장 위에 있는 점  $(x_1, y_1)$ 에서 기울기가 수직이 되는 직선  $l_1$ 을 구한다. 그 후,  $(x_1, y_1)$ 에서부터 차폭의 길이만큼 떨어진 직선  $l_1$  위의 점  $(x_2, y_2)$ 을 구한다. 이 때 구한  $(x_2, y_2)$ 로  $l_1$ 을 평행이동하여 가상의 차선을 그린다.

#### - 정지선

정지선을 인식하여 차가 멈출 수 있게 정지선까지의 거리를 구했다. (그림 27)에서 초록색영역의 가로 길이를 10픽셀씩 줄여서 차선에 포함되지 않게 한 후, (그림 28)과 같이 차선을 제외하고 정지선만 보이는 이진화된 영상  $B_1$  (2차원 배열)을 구한다.  $B_1$ 에서 x축의 값들을 모두 더하여 1차원 배열  $B_2$ 를 구한다.  $B_2$ 의 값이 제일 큰 좌표  $y_1$ 을 구한다. 임계값을 설정하여 값의 크기에 따라 화살표인지 정지선인지 빈 영역인지 구분한다.

$y_1$ 과 픽셀 당 거리 비율을 통해 정지선까지의 실제 거리를 구한다.



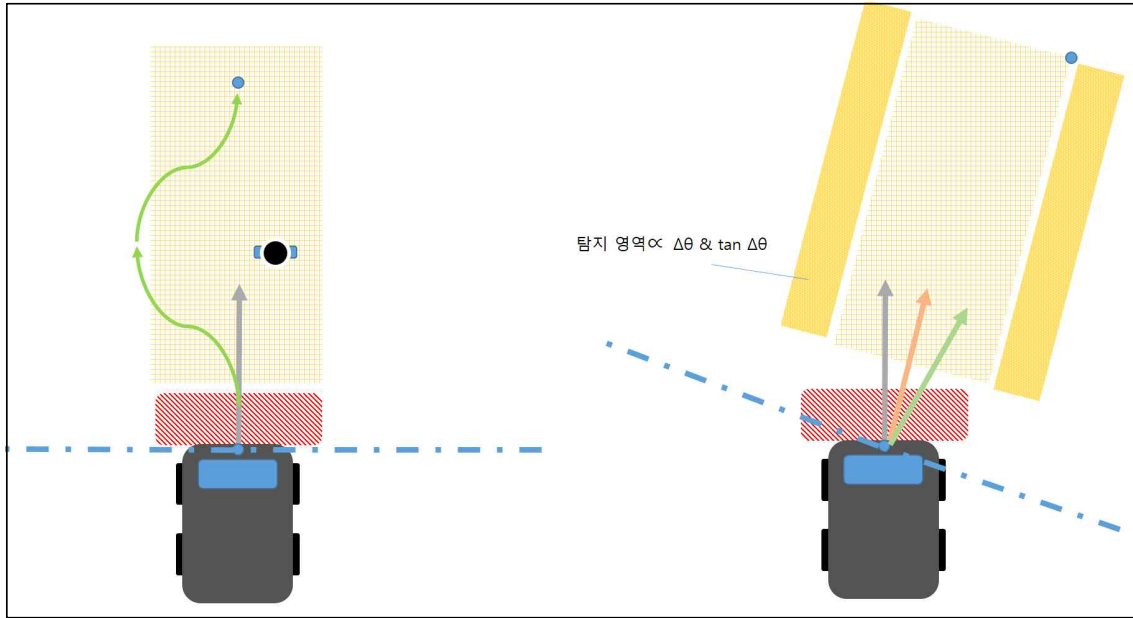


그림 29 - 라이다를 이용한 장애물 탐지

$$\Delta\theta = \frac{Heading + GoalHeading}{2}$$

#### - 경로 상 장애물 탐지

주행 알고리즘은 루프 당 총 15개의 경로를 생성하며, 해당 경로는 장애물을 감지함으로서 인근 경로를 배제하는 방식으로 작동된다. 경로는 목표점과의 Heading의 차이에 따라 폭이 넓어지며, 이에 맞게 장애물을 탐지하여 주행 알고리즘에 보내주는 영역을 확장시킴으로서 충돌을 미연에 방지한다. 이때 라이다에 인식되는 장애물은 모두 한 순간에 대한 인식이므로 객체 인식을 통한 동적 장애물을 탐지하여 탐지 영역 내에 진입하는 가상의 장애물을 다음 주행 알고리즘 작동에서 반영하게 된다.

#### - 동적 장애물 탐지

270도 인지 가능한 라이다를 이용하여 감지한 거리 데이터를 이용하여 일정 거리 이상 떨어져 있는 점들에 대해 별도의 개체로 인식한다. 전방 일정 범위 내의 장애물에 대해 직교 좌표계로 전환하여 중점과 차량 진행 방향에 대한 수직 길이를 측정한다. 한 번의 회전 후, 현재 측정된 개체와 이전 개체를 비교하여 추정된 뒤, 개체의 상대 속도를 감지하고 예측하여 다음 장애물 탐지에 반영하게 된다.



## 라. 표지판/신호등 인지

도시 내에서는 단순히 gps 노드나 차선과 같은 유형의 구속조건 외에도 표지판과 신호등과 같은 무형의 구속조건도 지키면서 주행해야 한다. 하지만 도심의 환경상 rule-base의 이미지 검출로는 다변하는 도시환경에서의 이미지 검출이 어렵다고 판단하여 yolo 알고리즘으로 도시의 표지판과 신호등을 검출시키기로 했다. yolo는 yolo\_mark 프로그램을 이용하여 라벨링이 편리하고, 학습시키기 위한 예제 또한 풍부하기 때문이다.

K-city의 완주를 위해서는 초록불, 빨간불, 교차로, 좌/우회전, 횡단보도, 안전구역, 방지턱, 주차 등 9가지의 인식이 필요하다 판단하여 이들 사진을 모았다. macaron의 이미지 인식에는 8/13일 기준, 구글에서 캡처한 이미지 325장, 팀원들이 도로를 지나다니면서 모은 사진 412장, rosbag을 이용하여 웹캠으로 모은 k-city의 영상을 캡처한 887장의 사진으로 총 9개의 라벨링으로 학습시킨 결과, k-city에 맞게 학습되어 k-city 내의 웹캠 영상으로 테스트 결과 우수한 성능을 보였다. 차후 대회에 맞는 표지판이 설치되면 추가로 촬영하여 학습에 반영할 계획이다.



그림 30 - 빨간불과 횡단보도



그림 31 - 방지턱과 횡단보도



그림 32 - 초록색 신호등



그림 33 - 빨간 신호등

### 3. 계획(path\_planning)

#### 가. 근거리 노드 탐색 및 lookahead 설정

##### - 근거리 노드 탐색

GPS노드들을 쉼스크라이브 : (x, y)

에르미트 보간을 기초로한 카디널 스플라인 생성 : (x(t), y(t))

최소자승법을 이용해 3차 다항식으로 회귀 : (S, L)

GPStxt\_reader 노드는 미리 주어진 GPS 점들에 대해 곡선경로를 생성한다. 주어진 점들을 부드러운 곡선으로 연결하는 곡선생성 방법으로는 보간법과 회귀법이 있다. 보간법은 점들 사이를 지나가는 곡선을 생성하며, 회귀법은 점은 곡선생성을 위한 제어 점으로 사용하여 점 근처를 근사하는 곡선을 생성한다. 본 연구팀은 주어진 GPS 점들에 대해 에르미트 보간(Hermite interpolation)을 기초로 한 카디널 스플라인(cardinal spline)을 적용한 후에, 최소자승법을 이용해 매개변수 t에 대한 x, y 좌표들을 3차 다항식으로 회귀시켜 SL좌표계의 곡선경로를 생성하였다.

에르미트 보간은 두 점과 두 점에서의 접선벡터가 주어진 경우 접선벡터에 영향을 받아 곡선을 생성 한다. 에르미트 보간을 이용하여 2차원 공간상에서 곡선을 표현하기 위해 파라미터 t를 정의하고, 다음과 같은 행렬로 표현 한다[14].

$$p(u) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \begin{bmatrix} 2 & -2 & 1 & 1 \\ -3 & 3 & -2 & -1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} p(0) \\ p(1) \\ p'(0) \\ p'(1) \end{bmatrix}$$

접선벡터는 주변 두 점 사이의 벡터를 이용하여 다음과 같이 접선벡터를 설정한다. 여기서  $\alpha$ 는 접선벡터의 가중치로 이 값에 따라 곡선의 장력이 결정된다.

$$p'_k = (1 - \alpha)(p_{k+1} - p_{k-1}) \quad \text{단 } , 0 \leq \alpha \leq 1$$

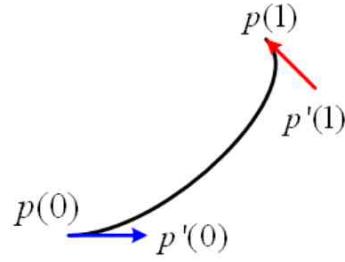


그림 35 - 에르미트 보간

에르미트 보간을 이용하여 임의의 장력으로 노드를  $t$ 의 매개변수로 나누어 준다. macaron 알고리즘에는  $t$ 는 10개의 노드로 분할시켰다. 이 알고리즘은 현재 위치에서 10m 앞의 gps노드의 기울기와 평행한 경로를 형성하므로, 이 함수를 구하기 위해서는 현재 위치와 가장 가까운 거리를 구할 필요가 있었다. 이를 위해 현재위치와 각각의 노드간의 거리를 목적함수로 경사하강법을 이용했다.[15] 이 시퀀스를 거리의 변화량이 일정량에 수렴할 때 까지 반복하고, 수렴이 안되는 정지시키는 방식으로 루프를 정지시켰다. 기존의 경사하강법은  $D'$ 만 이용하여 목적함수의 최소값을 수렴시킨 반면, 해당 방식은 직선주로와 같이 현재위치와 노드거리의 변화량이 급격한 경우는 천천히 수렴되게 해주고 특정 곡률을 가진 커브는 노드거리의 변화량이 거의 없으므로 수렴속도를 올려주도록 2계미분값으로 나눠주고 반복횟수가 커질수록 천천히 수렴하도록 반복식을 넣어주었다.

$$node^{m+1} = node^m - \frac{D(t)'}{D(t)'' + \nu^* \text{반복횟수}}$$

그러나 연산량과 오차의 누적문제로  $t$  노드에 대해서 경로와의 가장 가까운 노드를 구했으나,  $t$ 는 등간격이 아닌 임의의 매개변수이기 때문에 차후에 수정이 필요하다고 사료된다.

#### - lookahead 설정

에르미트 보간은  $x$ 와  $y$ 를 매개변수  $t$ 에 대해서 0~1사이의 값으로 매개변수화 시킨다. 즉 Arc-length parameterize가 필요하다.[16] 이를 위해  $t$ 노드에 대해서 adaptive quadrature로  $s(t)$ 를 구했다.[17] 적분값이 차이가 일정이상 수렴하면 다음노드를 적분하는 방식으로 vel\_planning가 제시하는 현재속도에 맞춰 lookahead point만큼의 gps노드까지의 거리를 구한 후, 그 때의  $x$ 와  $y$ 를  $s$ 에 대해 3차식에 대한 최소자승법을 이용하여 Arc-length parameterizing을 했다. 이 부분에서 [14]는 bisection method로  $s$ 와  $t$ 를 매칭 시켰으나, 각각의

원하는  $x$ 와  $y$ 에 대해 bisection method를 적용하는 것보다 연산의 반복없이 행렬식으로 구할 수 있는 최소자승법[18]이 더 연산량이 적다고 사료하여 이를 적용시켰다.

```

1. procedure integrate ( f, a, b,  $\tau$  )
2.    $Q \approx \int_a^b f(x) dx$ 
3.    $\varepsilon \approx \left| Q - \int_a^b f(x) dx \right|$ 
4.   if  $\varepsilon > \tau$  then
5.      $m = (a + b) / 2$ 
6.      $Q = \text{integrate}(f, a, m, \tau/2) + \text{integrate}(f, m, b, \tau/2)$ 
7.   endif
8.   return Q

```

그림 36

## 나. 경로 생성

path\_planning 노드에서는 GPStxt\_reader노드가 퍼블리시해주는 기본곡선 경로의 데이터를 이용해, 기본경로를 따라 주행하는 여러 개의 후보경로들을 생성하고, 가장 적합한 경로를 선택한다<sup>[8]</sup>. 여기서는 먼저, path\_planning 노드가 어떻게 후보경로를 생성하는지 다뤄보고자 한다.

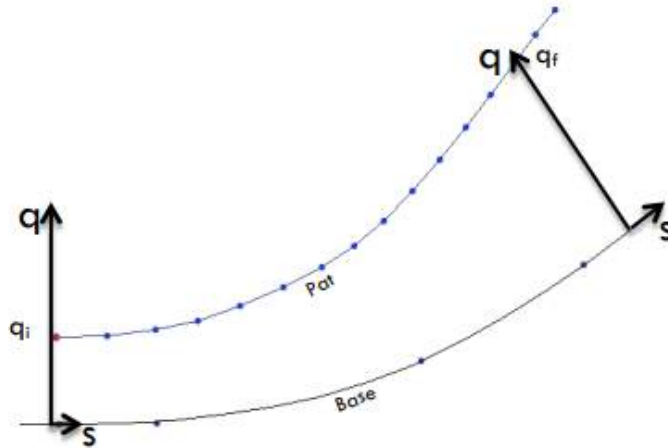


그림 37 - 기본경로를 S축으로 사용하는 SL좌표계

후보경로는 기본경로를 S축으로, 이에 반시계방향으로 수직인 축을 L축으로 사용하는 SL좌표계를 통해 생성된다. 이를 위해선 기본경로의  $x, y$ 좌표를 기본경로의 길이  $s$ 에 대한 함수로 표현하는 것이 매우 중요하다. 본 연구팀은 이를 위해서 앞서 기술한 해석적 방법을 사용하였다. 위의 첫 번째 그림은 SL좌표계를 통한 후보경로의 생성을 나타낸 것으로, 그림에선 문자 L대신  $q$ 를 사용하였다.

후보경로는 S축 방향으로 길이가  $\Delta s$ 이고, 끝점이 기본경로로부터 L축 방향으로  $q_f$ 만큼 떨어진 3차 다항식으로 정의된다. 즉,  $xy$ 좌표 상에서는 기본 경



로 투영한 길이가  $\Delta s$ 이고, 끝점이 기본경로에 내린 수선의 길이가  $q_f$ 인 경로가 되는 것이다. 또한, 차량의 부드러운 주행을 위해 차량의 현재 진행 방향을 경로생성 시에 고려해주어야 한다. 차량의 현재 진행방향과 가장 가까운 기본경로의 진행방향 사이의 차이를  $\theta$  라고 할 때, SL좌표계상의 후보 경로함수  $path(s)$ 는 다음을 만족시키는 3차 다항식이어야 한다.

$$\begin{aligned} path(0) &= q_i & path(\Delta s) &= q_f \\ path'(0) &= \theta & path'(\Delta s) &= 0 \end{aligned}$$

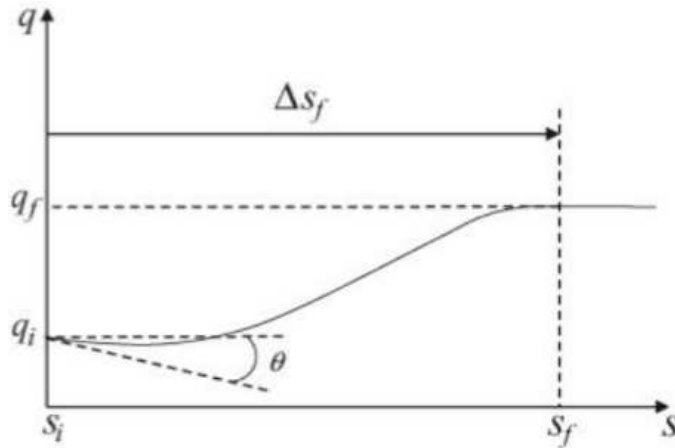


그림 38 - SL좌표계 상에서 3차 다항식으로 생성되는 후보경로

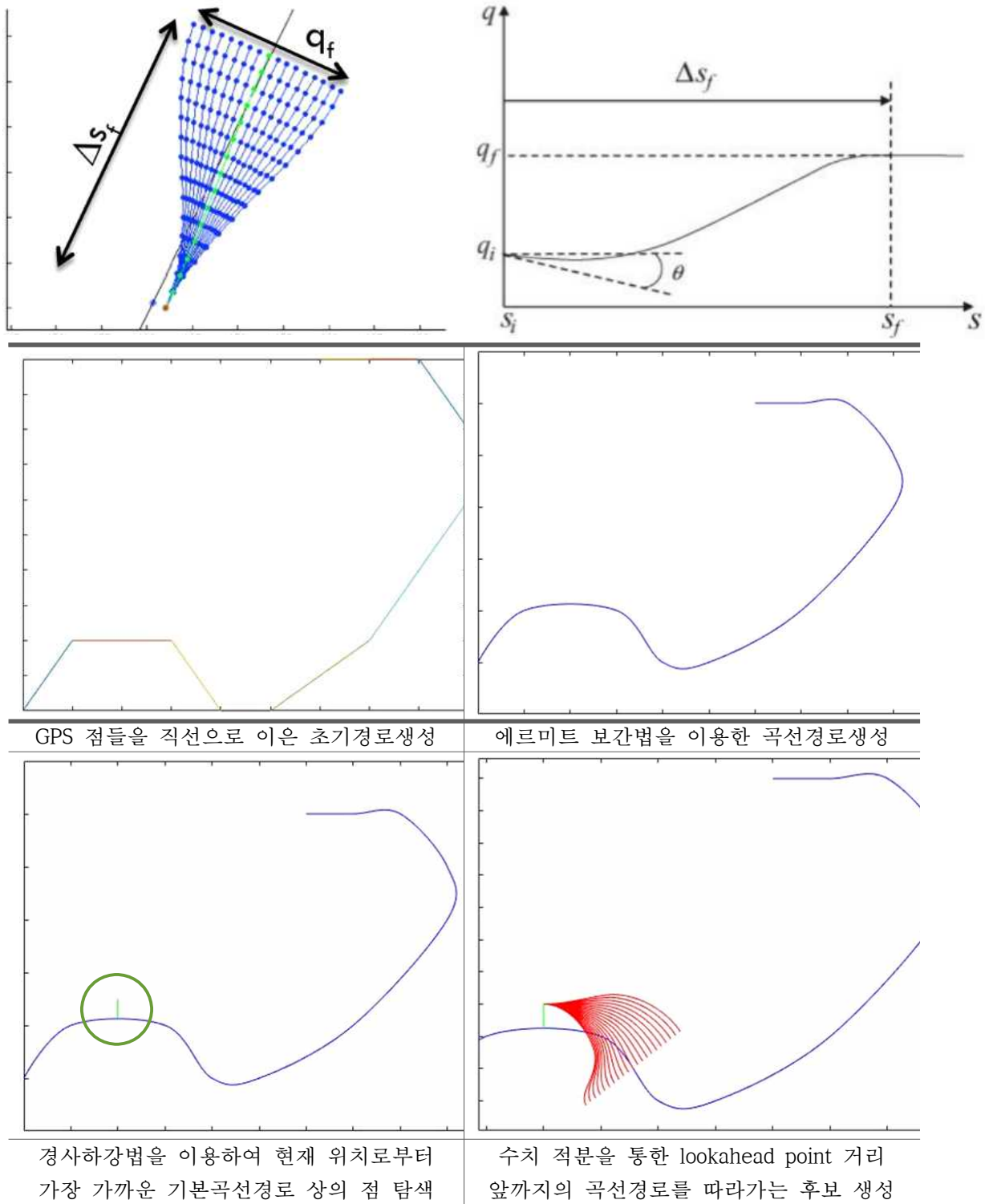
3차 다항식에 총 4개의 구속조건이 부여되었으므로, 몇 번의 짧은 계산을 통해 후보경로  $path(s)$ 를 구할 수 있다.

SL좌표계상에서 후보경로를 생성한 이후에는, 이를 xy좌표로 변환해주어야 한다. 이는 기본경로 위의 각 점들의 L축 방향, 즉 법선벡터 방향으로  $path(s)$ 만큼 오프셋시켜 얻을 수 있다. 이는 수식으로 다음과 같이 나타난다.

$$\begin{aligned} x_{path}(s) &= x_{base}(s) + path(s) \cdot \cos(\tan^{-1} \frac{dx}{dy_{base}}(s) + \pi/2) \\ y_{path}(s) &= y_{base}(s) + path(s) \cdot \sin(\tan^{-1} \frac{dx}{dy_{base}}(s) + \pi/2) \end{aligned}$$

아래의 그림은 계획단계의 전체적인 알고리즘을 ROS노드에 코딩하기 전,

디버깅을 위한 plot을 용이하게 하기 위하여 MATLAB에서 먼저 코딩을 진행한 후, plot해본 것이다.



#### 4. 판단&제어(path\_planning)

##### 가. 경로 선택

path\_planning 노드는 앞서 생성된 후보경로들에 각각 비용을 부과하고, 그

비용이 가장 적은, 즉 가장 적합한 경로를 선택한다<sup>[7]</sup>. 본 연구팀은 각각의 후보경로에 부과할 비용에 다음 4가지를 선정하였다.

1. 후보경로가 GPS 기본경로를 얼마나 잘 추종하는 지(offset cost)
2. 후보경로가 차선중심에서 얼마나 벗어나 있는지(lane cost)
3. 후보경로 상에 장애물이 존재하는지(obstacle cost)
4. 후보경로가 차량의 이전경로에서 얼마나 벗어나 있는지(consistency cost)

먼저, offset cost는 후보경로가 GPS 기본경로를 얼마나 잘 추종하는 지를 나타내는 값으로, 후보경로의 끝점으로부터 기본경로까지의 거리( $q_f$ )를 그 값으로 갖는다. 수식으로는 다음과 같이 표현된다.

$$C_{offset} = |q_f|$$

lane cost는 후보경로가 차선중심에서 얼마나 벗어나 있는지를 나타내는 값으로, 후보경로의 끝점으로부터 차선 중심까지의 거리를 그 값으로 갖는다. 참고한 논문<sup>[19][20]</sup>은 GPS 현재 위치에서 lookahead point의 거리 만큼의 deviation 들의 이산합으로 표현했으나, 도시에서는 gps노드와는 평행하게 따라가면서 차선에 구속되어야 하므로, 카메라에서 나오는 영상을 이용하여 구한 차선의 deviation과 gps노드를 이용하여 구한 경로들 간의 거리의 이산합으로 코스트를 매겼다. 그 수식으로는 다음과 같다.

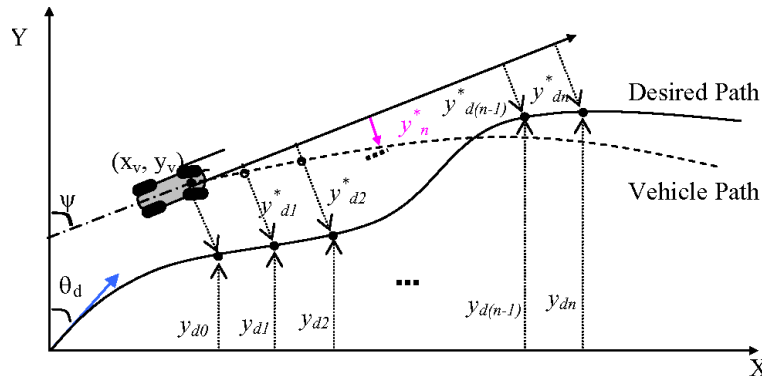


그림 39 - 공식의 이해를 돕기 위한 참고논문의 그림

$$C_{lane} = \sum_{i=2}^{lookahead} y_{di}$$

obstacle cost는 후보경로 상에 장애물이 존재하는지를 나타내는 값으로, 라이다를 통해 얻은 데이터들을 절대좌표 상의 데이터로 변환한 뒤, 후보경로 위에 장애물이 있는지 없는지를 판단한다. 장애물과의 거리를  $\frac{1}{r^2}$  한 값을 cost로 가지고 그 근처를 지나가지 않도록 gaussian을 적용시켜 각각의 경로에 대한 충돌dmf 그 값으로 갖는다. 수식으로는 다음과 같이 표현된다.

$$C_{obstacle}[i] = \sum_{k=0}^n \frac{1}{agv(k, r^2)} * \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(4/15*(i-k))^2}{2\sigma^2}}$$

consistency cost는 후보경로가 차량의 이전경로에서 얼마나 벗어나 있는지를 나타내는 값으로, 이전경로와 후보경로가 이루는 면적의 크기를 그 값으로 갖는다. 본 연구팀은 두 경로사이의 면적이 삼각형이라고 가정하여 식을 간소화하였다. 수식으로는 다음과 같이 표현되며, 여기서  $\Delta s$ 는 후보경로의 길이,  $d$ 는 이전경로를 계산한 시점으로부터 현재까지 차량이 기본경로 상에서 움직인 거리를 의미한다.

$$C_{consistency} = \frac{|q_f - q_{f_{before}}|}{2(\Delta s - d)}$$

total cost는 위의 4개의 비용을 모두 포함하는 비용으로, 각각의 비용에 가중치를 곱해 더한 값이다. path planning 노드는 후보경로들 중 total cost가 가장 낮은 경로를 최종적으로 선택한다. total cost는 수식으로 다음과 같이 포함된다.

$$C_{total} = C_{offset} \cdot w_{offset} + C_{lane} \cdot w_{lane} + C_{obstacle} \cdot w_{obstacle} + C_{consistency} \cdot w_{consistency}$$

## 나. 거동 판단

후보경로 중 가장 적합한 후보경로가 선택되면, 이제 차량이 그 경로를 따라 주행하도록 알맞은 조향 각을 입력해주어야 한다. 본 대회에 차량 플랫폼인 ERP42는 후륜부가 구동, 전륜부가 조향을 담당하므로 차량을 제어하기 위해서는 아래의 그림처럼 자전거 모델(Bicycle model)을 사용하여야 한다[21].

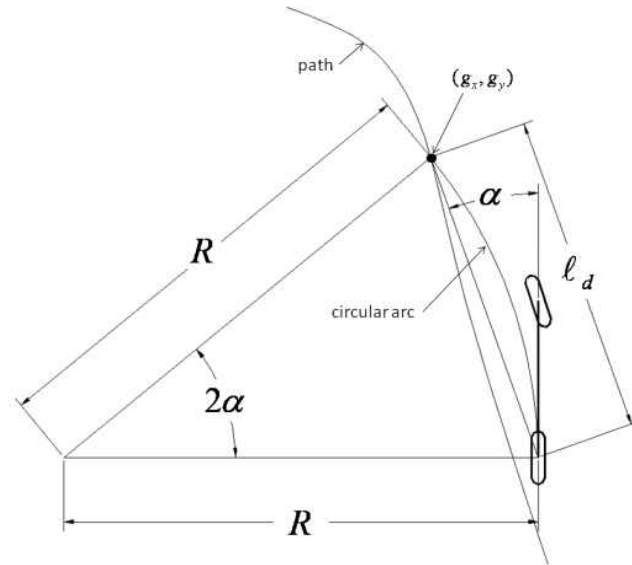


그림 40 - 차량제어를 위한 자전거 모델

먼저, 선택된 경로에서, 내 현재 위치로부터 look-ahead 거리(이하  $l_d$ )만큼 떨어진 지점을 목표지점으로 지정한다.  $l_d$ 가 길수록 차량은 더 먼 곳을 목표지점으로 놓고 주행하기 때문에 경로에 변화에 대해 더 일찍 반응하여 부드러운 주행을 할 수 있지만,  $l_d$ 가 너무 길면 차량이 바로 앞의 장애물에 대한 대응을 하지 못할 수도 있다. 따라서 여러 번 시험주행을 해보며 가장 적합한  $l_d$ 를 찾아야했다.

목표지점을 선택한 이후, 차량이 그 목표지점으로 이동하도록 특정 속도와 조향 각을 입력해주어야 한다. 속도는 이후의 속도제어에서 살펴보도록 하고, 여기서는 차량의 조향 각에 대해 알아보자. 차량의 윤거(wheel base)가  $L$ 이고, 현재위치로부터 목표지점까지의 방향과 현재 차량의 방향사이의 차이를  $\alpha$ 라고 할 때, 자전거 모델에 의거한 차량의 조향 각  $\delta$ 는 다음과 같다.

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

ERP42는 조향 각을 -2000부터 2000사이의 정수로 입력받으므로, 최종적으로 차량에 시리얼 통신을 통해 전송해주는 값은 아래와 같다. 또한,  $\delta_{ERP42}$ 에 저장된 값의 크기가 2000보다 클 때는, 그 크기를 2000으로 맞추어 주었다.

$$\delta_{ERP42} = \frac{180 \cdot 71}{\pi} \tan^{-1}\left(\frac{2L \sin \alpha}{l_d}\right)$$

#### 다. 속도 제어

차량에 항상 일정한 속도를 입력해 주어도 곡률이 큰 부분, 즉 크게 회전해야 하는 경로에선 속도를 줄여 준다. 이를 수식으로 나타내면 다음과 같다.

$$v = \frac{v_{std}}{\kappa}$$

### IV. 실험 계획 및 결과(experiment)

본 연구팀은 플랫폼을 용달하지 않고도 실험을 진행할 수 있도록 교내에서 찾을 수 있는 도로환경에서 실험주행을 진행하였다. 총 두 군데의 장소에서 실험을 계획하였는데, 각 장소에서 실험하기 적합한 미션을 찾아 실험하였다.

#### 1. 팔정도 실험

##### 가. 실험 계획

동국대학교는 남산에 위치해 있다는 지리적 특성상 교내에서 실험주행을 진행할 수 있는 평지가 많지 않았다. 그 중 가장 적합한 도로는 동국대학교 본관 앞쪽에 위치한 중앙광장 팔정도를 둘러싼 도로들이었다.



그림 41 - 동국대학교 팔정도

팔정도에서 본 연구팀이 계획했던 실험은 gps 기본경로와 후보경로, 그리고 차량의 현재 위치와 진행방향을 시각화하는 것과, 선택된 후보경로를 통해 차량이 팔정도의 도로를 따라 주행할 수 있는지를 검증해보는 것이었다. 또한, 정적·동적 장애물 회피, 어린이 보호구역 인지 등의 gps노드 및 판단노드에 대한 실험을 계획하였다.

## 나. 실험 결과

먼저, 본 연구팀은 계획한 여러 가지 실험들 중 경로의 시각화 실험과 차량의 gps경로 추종주행 실험을 진행하였고, 나머지 실험들은 이후에 진행하기로 하였다.

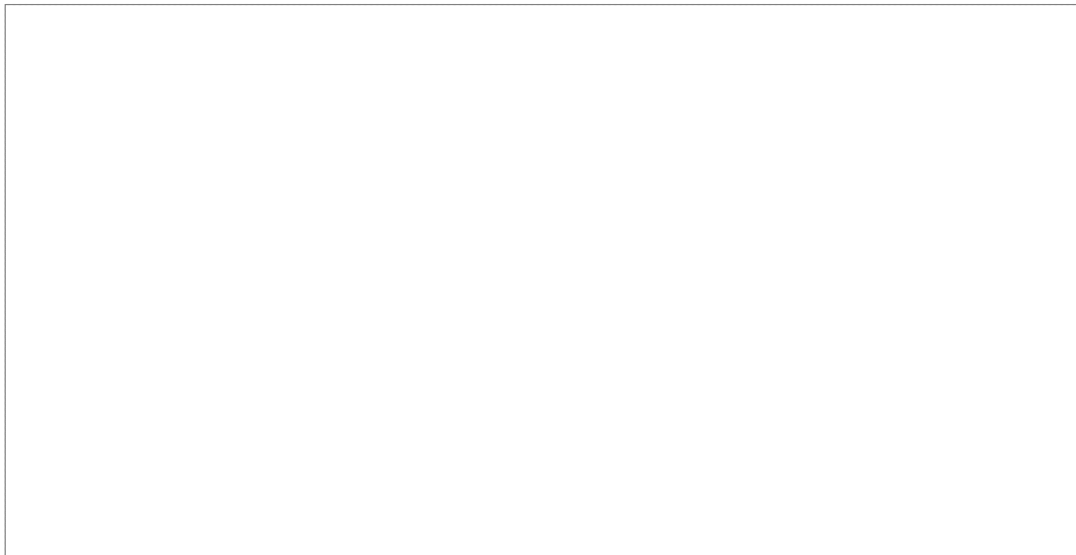


그림 42 - Rviz를 활용한 주행경로 및 차선의 시각화

경로의 시각화의 경우 위의 사진처럼 성공적으로 끝마칠 수 있었다. ROS에서 지원하는 시각화 툴 중 하나인 Rviz를 사용하였는데, 이를 통해 여러 센서가 감지한 주변 환경과 현재 차량의 위치정보, 그리고 경로를 하나의 창에서 시각화할 수 있었다. 오른쪽 Rviz 창에서 빨간색 경로는 차량이 지금까지 이동한 경로를, 파란색 경로는 사전에 주어진 gps 기본 경로를, 하얀 색 경로는 현재 차량이 선택한 후보경로를 의미한다. 또한 하늘색 화살표의 경우 현재 차량의 진행방향을 의미하며, 보라색 경로는 현재 카메라가 인식한 차선중심에 대한 위치정보를 의미한다.

차량의 gps경로 추종주행 실험 또한 성공적으로 진행되었다. path\_planning 노드에서 lane weight와 obstacle weight를 0으로 두어 차선과 장애물은 무시

하도록 하였고, offset cost와 consistency cost만을 이용하여 gps 기본경로를 부드럽게 따라가는 경로를 선택하도록 하였다. 첫 주행에서는 낮은 속도로 주행하여 차량이 gps경로를 추종할 수 있는지를 실험하였고, 차량은 20의 속도에서 gps경로를 부드럽게 따라 이동하였다. 그 후 계속된 실험 주행에서는 차량의 속도를 점차 높여보고, 차량의 거동이 매끄럽지 못하거나 gps경로를 제대로 추종하지 못할 때에는  $l_d$ 값과 후보경로의 길이를 바꾸어가며 실험을 진행하였다. 그 결과, 차량의 속도와 기본경로의 곡률,  $l_d$ , 후보경로의 길이 사이의 관계식을 유추해낼 수 있었다.



그림 43 - 팔정도 도로를 따라 자동으로 주행하는 차량

## 2. 원흥관 앞 도로 실험

본 연구팀이 두 번째로 실험을 계획한 장소는 동국대학교 공과대학 건물인 원흥관 앞에 위치한 도로이다. 앞서 언급했듯이 동국대학교 교내에는 평지가 많이 없을뿐더러, 양 옆 차선이 반듯하게 잘 그어져 있는 도로는 더욱 찾아보기 힘들었다. 교내의 여러 도로를 탐색해본 결과, 차선추종 및 차선변경 실험을 하기에 가장 적합한 도로로 원흥관 앞 도로를 선정하였다. 이 장소에서는 gps기본 경로가 차선 밖에 있을 때에도 gps경로를 추종하지 않고 차선을 따라 주행하는지, 날이 밝거나 흐려도 차선을 잘 인지하는지, 전방에 장애물을 인식했을 때 안정적으로 차선을 변경할 수 있는지 등을 실험할 계획이다.

원흥관 앞 도로실험의 경우, 아직 계획단계에 있고, 실제 실험은 8월 3주차에 진행할 예정이다.

그림 44 - 원흥관 앞 도로.  
차선이 잘 나타나 있다.



## V. 참고문헌(referencing & citation)

- 
- [1] Oliver Brock, High-Speed Navigation Using the Global Dynamic Window Approach(1991)
  - [2] 윤희상, 수정된 전역 DWA에 의한 자율이동로봇의 경로계획(2011)
  - [3] Kirsten L.R. Talvala, Pushing the limits: From lanekeeping to autonomous racing(2011)
  - [4] J.Borenstein, The Vector Field Histogram-Fast Obstacle Avoidance for Mobile Robots(1991)
  - [5] Iwan Ulrich, VFH+: Reliable Obstacle Avoidance for Fast Mobile Robots(1998)
  - [6] M. Montemerlo, Junior : The Stanford Entry in the Urban Challenge(2008)
  - [7] Keonyup Chu : local Path Planning for Off-Road Autonomous Driving With Avoidance of Static Obstacle.(2012)
  - [8] Thomas Braunl : Advanced path planning for an Autonomous SAE electronic race car(2014)
  - [9] Complete Triaxis Magnetometer Calibration in the Magnetic Domain
  - [10] 칼만 필터는 어렵지 않아 / 김성필
  - [11] Implementing a Tilt-Compensated eCompass using Accelerometer and Magnetometer Sensors by: Talat Ozyagcilar Applications Engineer
  - [12] 파이썬 opencv프로그래밍 / 김동근
  - [13] 컴퓨터비전 / 오일석
  - [14] V. B. Anand, computer graphics & geometric modeling for engineers, J. Wiley, 1993.
  - [15] Hongling Wang : Robust and Efficient Computation of the Closest Point on a Spline Curve(2003)
  - [16] Hongling Wang : Arc-length parameterized spline curves for real-time simulation(2002)
  - [17] [https://en.wikipedia.org/wiki/Adaptive\\_quadrature](https://en.wikipedia.org/wiki/Adaptive_quadrature)
  - [18] [https://en.wikipedia.org/wiki/Least\\_squares](https://en.wikipedia.org/wiki/Least_squares)
  - [19] Rs.shape(2010) a mathematical model for driver steering control , with design , tuning and tuning and performance results
  - [20] Muhammad aizzat zakaria(2016) Dynamic Curvature steering control for autonomous vehicle : Performance analysis
  - [21] Jarrod M.Snider, "Automatic Steering Methods for Autonomous Automobile Path Tracking", Robotics Institute Carnegie Mellon University Pittsburgh, Pennsylvania, 2009.