제18회 임베디드SW경진대회 개발계획서

[자율주행 모형자동차]

□ 개발 개요

팀 명	FAFA (Five assembled for Autonomous)
목 표	카메라를 통한 영상인식으로 외부상황을 인지하고 센서로 주변 물체를 감지하며 자율주행하는 모형차의 임베디드 시스템을 개발한다.
개발 내용 (요약)	리눅스 운영체제의 Host PC와 제공 되는 TI TDA2x 보드를 이더넷 통신을 이용하여 모형차의 개발 환경을 구성한다. OpenCV를 활용해 영상인식으로 차선, 신호등, 표지판과 정지선의 정보를 처리하고 적외선 거리 센서로 주변 물체로부터의 값을 계산하여 외부 상황을 스스로 판단해 자율 주행하는 모형차의 알고리즘을 개발한다.
기술동영상	https://youtu.be/E9MgEd5yksw

1. 개발 목적 및 목표

1.1. 개발 목적

1) 카메라를 통한 영상 인식과 적외선 거리 센서를 통해 수집한 정보를 토대로 차량이 주행하는데 있어서 외부 상황을 판단하여 차량을 스스로 제어하는 모형차를 제작하는 것이 목적이다.

2) 도로주행, 주차, 신호등, 교차로, 추월구간, 돌발 상황 등 10가지 미션 코스를 임베디드 환경에서 신속하고 정확하게 수행 할 수 있게 개발한다.

1.2. 개발 목표

자율주행은 실시간으로 입력받은 정보를 빠르게 처리하여 상황에 맞는 동작을 해야 하는 신속성이 요구된다. 동시에 센서들로부터 값을 받아오고, 받아온 값들을 변수에 저장해 복잡한 동작을 구현하는 함수들이 서로 충돌하여 오류가 발생할 확률이 크다. 이러한 점을 보완하고자 함수를 기능별로 모듈화 하여 관리하고 변수의 수를 최소화 한다. 차량의 센서들로 부터 받아온 정보 중 필요한 정보만 선별하는 함수를 따로 구현하여 연산을 최대한 줄여 신속성을 높이는 것을 목표로 한다.

2. 향후 계획

영상 처리 기술과, 적외선 거리 센서를 이용한 자율주행 모형차를 개발한 경험을 토대로 향후 NVIDIA 플랫폼 임베디드 보드를 기반으로 자율주행 차량을 개발한다.

개발 중 발생할 수 있는 여러 상황을 빠른 연산속도로 최적화된 알고리즘 개발과 딥러닝을 통한 상황 예측/판단, Lidar 센서와 Radar 센서를 이용한 감지 기술로 해결하여 완전한 AI기반 자율주행 차량 개발을 목표로 한다.

□ 개발 방향 및 전략

1. 기술적 요구 사항

1.1 하드웨어 제어

차량 구동 제어를 위한 주어진 임베디드 보드의 기능 및 MCU Datasheet를 활용해 GPIO 제어법을 이해한다.

정밀한 모터 제어를 위한 PWM과 Timer 특성, Interrupt을 이해한다.

센서 값을 활용하기 위해 ADC의 특성을 이해하고 활용한다.

1.2 OpenCV를 활용한 영상 인식

특정 영역에 대해 영상을 처리하기 위해 ROI (Region of Interest)기술을 활용한다.

RGB 영상을 이진영상으로 변환하기 위해 Grayscale로 변환한다.

Grayscale을 거친 이미지를 Binarization을 통해 이진영상으로 변환한다.

이미지 노이즈를 효과적으로 검출하기 위해 Morphology 기법을 활용한다.

영상에서 노이즈를 제거하고 Edge를 찾기 위해 Canny Edge 검출 알고리즘을 사용한다.

검출된 Edge에서 정확한 경계선을 얻기 위해 Hough Transform을 적용한다.

신호등 색이나 차선 색을 검출하는데 HSV Filtering을 활용한다.

2. 개발 방법

2.1. 개발 환경 구축

Host PC의 개발환경 운영체제는 Linux(Ubuntu 16.04)를 이용한다.

영상 처리의 개발언어는 OpenCV를 사용하기 위해 C++을 이용하고 개발 도구로 Visual Studio를 사용한다.

실제 대회 구조물과 최대한 동일한 환경으로 Test Bed를 제작해 다양한 경우의 수에서 반복 테스트를 통해 수집된 데이터로 알고리즘과 차량의 움직임을 최적화한다.

팀원들이 작성한 코드는 버전 관리를 위해 형상 관리 도구로는 GitHub을 사용한다.

2.2. 모형자동차 OS 및 디바이스 개발

Host PC와 제공되는 임베디드 보드인 TI TDA2x를 이용하기 위해 UART와 Ethernet통신을 이용한다. Ethernet은 Host PC에서 Cross Compile된 파일을 보드로 전송하거나 모형차의 카메라에서 인식된 이미지를 Host PC로 전송하기 위해 사용한다.

2.3. OpenCV Library를 이용한 영상인식 알고리즘 개발

1) 차선 검출

기존 구현 사례 (1). 입력영상 하단 부분에 ROI를 적용 및 HSV변환으로 차선의 색 정보를 추출하고 이진화한다. Canny Edge Detection으로 차선 Edge를 검출하고 이미지에 가우시안 필터를 적용하여 노이즈를 제거한다. Hough Transform을 이용하여 정확한 경계선을 구한다.

기존 구현 사례 (2). 입력 영상의 왜곡을 보정하기 위해 카메라 캘리브레이션을 수행한다. 차선이 가진 색상 특징을 이용하여 차선을 검출하기 위해 카메라 영상에 HSV를 적용하고 이진화 영상으로 나타낸다. 카메라 뷰에서 탑뷰로 영상 시점을 변환시키는데 호모그래피를 이용하고, 시점 변환 후 가우시안 필터를 이용하여 노이즈를 제거한다.

2) 직진/곡선 코스

차선 검출로 얻은 Edge와 소실점의 움직임을 Tracking 하면서 차량의 방향을 정한다.

3) 신호등 분기점 코스

입력 영상에서 실제 신호등이 위치한 중앙과 상단 부분을 ROI영역으로 지정하고 노이즈 제거와 객체 추적에 사용되는 MSF 필터링 알고리즘을 적용한다. 조정된 영상에서 물체 외곽선을 Canny Edge 알고리즘으로 검출하여 후보영역을 지정하고 Hough Circle 알고리즘으로 신호등의 원을 검출한다. 검출한 원의 x좌표 값을 기준으로 좌, 우회전 신호등을 검출한다. 이미지를 이진화한 후 픽셀 값이 임계 값 이상이면 차량 조향을 한다.

4) 우선 정지 장애물 코스

입력 영상에서 빨간색을 추출하기 위해 HSV 변환을 한다. 변환된 이미지를 반복 테스트를 통해 빨간색으로 인식하는 HSV 값을 찾아낸 후 임계 값 이상이면 일시 정지 표지판으로 인식하고 정지한다.

5) 회전 교차로 코스

입력 영상에서 직진 차선과 수직한 정지선이 인식 되거나 라인 감지용 적외선 센서로 정지선이 감지되면 정지한다. 정지 후 곡선 차선이 인식 되면 회전 교차로로 진입한다.

6) 내리막 코스

입력 영상에서 카메라 뷰를 통한 차선 인식의 범위와 기울어진 각도를 보고 내리막 코스로 인식하고 진입한다.

7) 차로 추월 코스

입력 영상에서 카메라로 점선 차선을 검출하면 추월 차로 코스로 인식하고 진입한다. 차로 추월 코스 에서는 카메라를 일정 시간마다 좌우로 조향하면서 전방의 차량이 감지되면 차량을 제어하고 그렇지 않으면 직진 주행한다.

8) 종료 지점

입력 영상에서 안전지대를 나타내는 노란색, 흰색 빗금을 HSV, 이진화 변환을 통해 인식하고 종료지점으로 판단한다.

2.4. 주변 상황인지를 위한 센서 알고리즘 개발

1) 후진 주차, 평행 주차

후진 주차와 평행 주차 중 어느 주차가 필요 한지 판단하기 위해 적외선 거리 센서를 사용한다. 앞바퀴가 주차공간을 통과 했을 때 측면에 있는 적외선 거리 센서 2개의 측정값이 다르게 측정 되는 것을 이용한다. 주차공간에 있는 벽을 측정한 ADC값은 낮고 뒷바퀴에 위치한 ADC 값은 상대적으로 벽이 가까워 측정값이 높게 나오는 정보로 주차 종류를 판단한다. 그 후 설정된 알고리즘을 토대로 후진 주차와 평행 주차를 각각 구현한다.

2) 터널 코스

좌, 우측 적외선 거리 센서에서 비슷한 거리의 물체가 감지되면 터널코스로 인식하고 측면의 적외선 거리센서 값이 일정하도록 유지하며 주행한다.

3. 예상되는 장애요인 및 해결방안

Risk 1. 내리막길

내리막길에서 중력 가속도에 의해 급격한 속도 증가로 슬립이 발생할 가능성이 있다.

Solution 1.

슬립이 발생했을 때, VDC(Vehicle Dynamic Control)나 ABS(Anti Break System)을 적용시켜 차체 자세를 제어한다. 바퀴분해능의 PWM제어 기술을 이용하여 각속도를 측정한다. 급격한 속도 증가가 일어났을 경우 차량 속도를 제어한다. 반복 테스트를 통해 ABS 제어 방법을 구현한다.

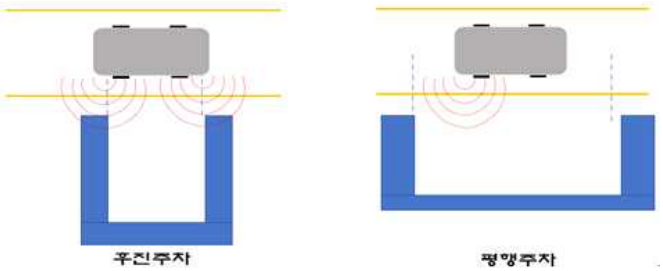
Risk 2. 주차

주차공간을 판단하지 못 할 가능성, 후진주차와 평행주차의 잘못 판단할 가능성, lane keeping을 실패하여 벽과의 충돌 가능성이 있다.

Solution 2.

주차 구조물을 인식하기 위해 측면 센서를 이용하여 주차 구조물인지 판단한다. 후진주차와 평행주차를 구분 할 때 주차 공간을 측정하기 위해 측면센서를 이용한다.

주차 구간 진입 시 미리 입력해놓았던 주차 공간의 깊이 값을 기준으로 후진 주차 공간과 평행 주차 공간을 구분한다. 주차 시 벽과의 충돌 가능성을 대비해서 차량 위치와 주차 공간 벽과의 거리를 측정한다. 측면 후방센서, 후방센서를 이용하여 자동차를 제어함으로써 주차와 정지, 출발을 구현한다.



Risk 3. 터널

터널을 잘못 판단할 가능성과 카메라 라인 인식이 아닌 적외선 거리 센서로 주행을 할 때 센서 스위칭 시 벽과의 충돌 가능성이 있다.

Solution 3.

터널 진입 시 양 측면의 적외선 거리 센서에 일정거리의 벽이 인지되면 터널 진입으로 간주한다. 진입 후 센서를 이용하여 주행하기 위해서 차선 유지 시스템(LKAS) 주행 기법을 활용한다. 측면 한 쪽 센서를 이용하여 벽을 지속적으로 인식하며 통과한다.

Risk 4. 회전교차로

전방 차량 인식 문제로 회전교차로의 차량과 충돌할 가능성이 있다.

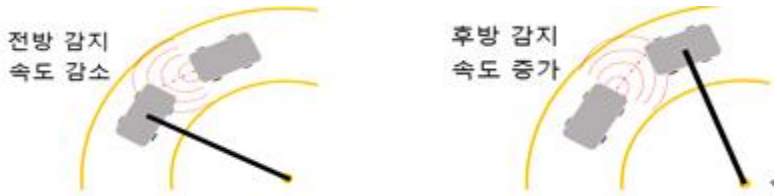
Solution 4.

교차로 진입 시 정지선에서 멈춰 적외선 거리 센서로 차량을 인지한다.

1) 정지선에서 차가 감지 된 경우 : 차가 감지되지 않았을 경우 진입하고 감지되면 정차한다.

2) 전방에 차량이 있는 경우 : 전방 장애물을 인지하며 일정거리 이상 가까워지지 않도록 최소 거리를 설정하여 속도를 제어한다.

3) 후방에 차량이 감지 된 경우 : 후방센서에 장애물이 감지되었을 경우 후방 차량이 가까워졌다는 의미이므로 속도를 올려 빠르게 교차로를 빠져 나간다. 교차로 회전 후 직선 lane에 우선순위를 두어 교차로를 빠져나오도록 한다.

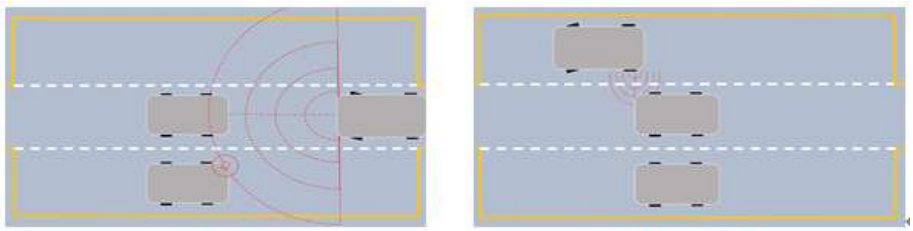


Risk 5. 추월차선

추월해야하는 차선의 좌, 우를 오판단하여 차량과 충돌하거나 차선 이탈의 가능성이 있다.

Solution 5.

먼저 추월 상황을 인지하기 위해서 전방 PSD센서로 장애물 인지와 HSV 필터링을 통해 추월 차선을 인지할 수 있게 해야 한다. 전방 장애물과 추월 차선이 인지가 된 경우 추월상황으로 판단하고 장애물이 감지되지 않은 반대쪽으로 조향하고, 측면 후방 적외선 거리 센서로 차량을 추월했다고 판단되면, 다시 반대 방향으로 조향 제어를 통해 원래 차선으로 돌아온다.



Risk 6. 우선 정지 장애물

정지 장애물 표지판을 인지하지 못 할 가능성이 있다.

Solution 6.

차선 주행 중 일정시간마다 일정 색을 검출하는 함수를 호출한다. HSV 필터링을 이용하며 표지판에 맞는 색상을 잘 검출할 수 있도록 실험을 통해 적절한 임계 값을 설정하도록 한다.

4. 예상 결과 작품이 활용될 분야 및 방법 제시

1) 자율주행 배달 로봇/자율 주행 이동 우체국, 물류센터 자율 주행 운반 로봇

적외선 거리 센서 또는 Lidar 센서로 주변 물체를 감지하거나 공간의 지도 및 현재 위치를 추정하는 slam기술을 추가하여 차선이 없는 공간에서도 자율주행이 가능하도록 한다.

2) 시각장애인을 위한 이동 로봇

앞선 방법으로 현재 위치를 인식하고 객체 인식기술을 적용하여 더 많은 사물의 종류를 판단하도록 하여 시각장애인이 탑승 가능한 이동 로봇으로 활용할 수 있다.

3) 공항/ 대형 마트에서 이용 가능한 자율주행 카트

앞선 방법으로 자율주행하고 사용자를 특정 객체로 인식하여 사용자를 따라 이동할 수 있도록 하여 무거운 물체를 자율주행 카트 로봇을 활용하여 운반할 수 있다.

□ 영상처리 및 센서 기술 활용 방안 및 공부 내용

1. 영상처리 기술 활용 방안 및 공부 내용

1) Grayscale Transform

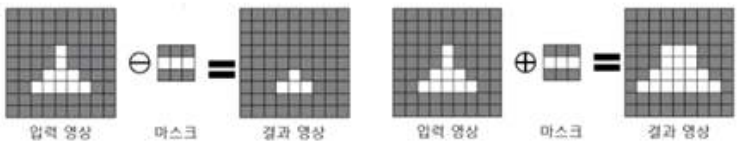
컬러영상으로부터 받아온 영상을 흑백으로 바꾸어 불필요한 연산량을 제거하여 최적화하는 과정이다. 컬러 영상의 경우 Grayscale 영상보다 외곽선 검출이 정확할 순 있지만, 실시간 처리에 있어 연산량이 부담되므로 흑백으로 변환한다.

2) 이진화

Grayscale Transform과정을 거친 영상의 명암을 제거하여 차선이 뚜렷이 인식되게 하기위한 과정이다. 8비트로 변환된 이미지를 이진화하여 0과 1만으로 표현해 연산량을 감소시켜 보드 환경에서 실시간 처리에 최적화 한다.

3) Morphology

Erode(침식), Dilate(팽창) 연산으로 이진화 처리된 영상의 노이즈를 제거하거나 증폭시킨다. Erode 연산은 각 픽셀과 mask를 비교하여 zero값을 가진 부분은 삭제시켜 결과적으로 이진영상의 노이즈 부분을 축소시킬 수 있으며, 반대로 Dilate 연산은 mask의 유효영역에 있는 픽셀들을 모두 밝게 만들어 노이즈를 증폭시킬수 있다. 이 방법을 통해 노이즈에 대한 제거나 검출을 효과적으로 기대 할 수 있다.



4) Canny Edge Detection

대부분 Edge추출 Mask는 잡음에 민감하므로 작은 잡음을 Edge로 간주하여 추출하는데 반에 Canny Mask는 잡음에 민감하지 않게 하여 강한 Edge를 추출하는 것에 목적을 둔다.

4.1. 노이즈 제거 (Noise Reduction)

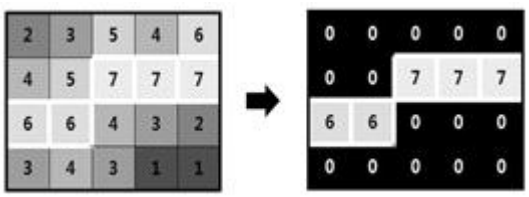
노이즈가 있으면 Edge판단이 어려우므로 Gaussian Mask를 통해 이미지의 노이즈를 줄인다.

4.2. 경사도 계산 (Finding Gradient)

이미지에 Sobel Mask를 수평방향, 수직방향으로 적용하여 x, y 방향의 Gradient 값 Gx, Gy을 획득한다. 얻은 Gx, Gy를 피타고라스 정리를 통해 경사 크기 G값을 도출한다. 그 다음 Gx와 Gy 방향을 통해 삼각함수 계산하여 경사도 θ를 얻는다. θ는 가장자리에 수직인 방향이기 때문에 방향성에 있어서 각도를 그룹화 하여 Edge를 구분한다.

4.3. NMS (Non-Maximum Suppression)

Image Processing에서 나온 Edge를 얇게 만들어주는 과정이다.

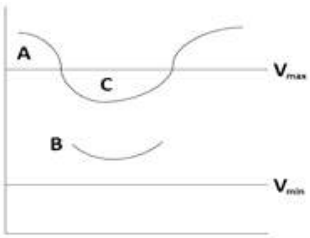


중심 픽셀을 기준으로 8방향의 픽셀 값들을 비교하여 중심픽셀이 가장 클 경우 그대로 두고, 아닌 경우에는 픽셀 값을 0으로 만드는 과정이다.

4.4. Double Thresholding

임계값을 기준으로 Low 값인 파란색 영역의 Edge들은 삭제시키고 주황색과 Hight 값인 빨간색 영역의 Edge는 값을 비교 구분하여 남겨둔다.

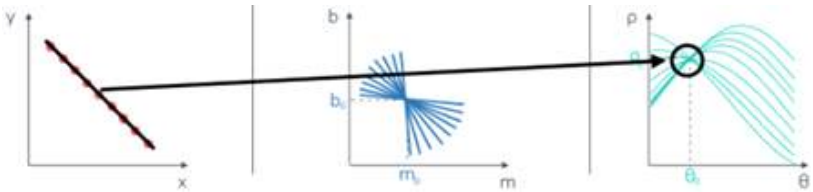
4.5. Edge Tracking by Hysteresis Threshold



강한 Edge만 최종 Edge로 남기기 위해 연관성 검사를 두 개의 임계점 Vmax와 Vmin을 설정하여 판별하게 된다. Vmax 이상에 있는 A는 강한Edge이므로 최종Edge에 추가된다. B는 주황색 영역에 있던 Edge이지만 강한Edge에 연결되어있지 않으므로 삭제된다. C는 강한Edge는 아니지만 A와 연결된 Edge이므로 최종Edge에 추가된다.

5) Hough Transform

이미지 상에서 직선과 원 같은 모양을 찾기 위한 과정이다. 본래 Image Space(y = mx + b)를 통해 생성할 수 있는 Parameter Space(m, b)는 기울기가 y축과 평행할 경우, m은 무한대의 범위이므로 문제가 생긴다. 이러한 문제점을 해결하기 위해 Hough Space(θ, ρ)로 표현된 공간으로 처리한다.



직선상에 있는 점들은 Hough Space에서 하나의 점으로 표현이 된다. 곧, Hough Space 상의 곡선들의 교점은 알고리즘 상의 임계 값과 비교되어 직선인지 아닌지 판단할 수 있게 된다.

6) HSV Filtering

H(Hue : 색조), S(Saturation : 채도), V(Value : 명도)로 이미지에 표현하는 방법이다. 흰색 차선은 무채색이므로 S와 V의 조건비교를 통해 만족시킨 값을 찾을 수 있으며, 노란색 차선은 V는 일정 값 이상이면서 H를 통해 노란색과 순수 색 값이 유사한 부분을 검출 할 수 있다.

7) ROI(Region Of Interesting) Selection

영상데이터 전체를 처리하기에는 시간이 오래 걸리고 판단하는데 불필요한 부분을 줄이기 위해 관심영역인 ROI 영역을 설정해서 그 부분만 작업을 처리하기 위해 요구된다. 차선은 영상의 하단 부분에 존재하기 때문에 하단부분을 관심영역으로 설정하여 처리한다.

8) Required Line Filtering

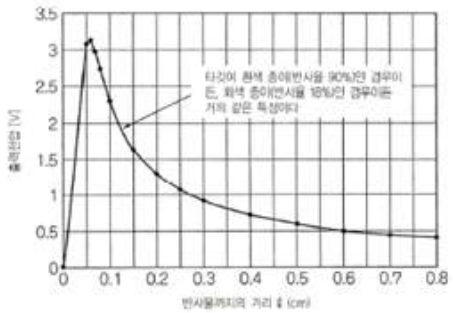
Hough Transform에서 구해진 직선들 중 기울기가 수직이거나 수평 혹은 길이가 매우 짧아 차선이 아님에 판단되는 직선들을 삭제하는 과정이다. 남겨진 직선들 중에서 차선으로 판단되는 기울기와 길이를 가진 직선을 선택한다.

9) 소실점 찾기

차선으로 판단된 두개의 직선을 토대로 연장선상에 위치한 소실점을 구한다. 소실점은 차량이 직선차선인지 곡선차선인지 판단하는 좌표 값으로 설정된다.

2. 센서 기술 활용 방안 및 공부 내용

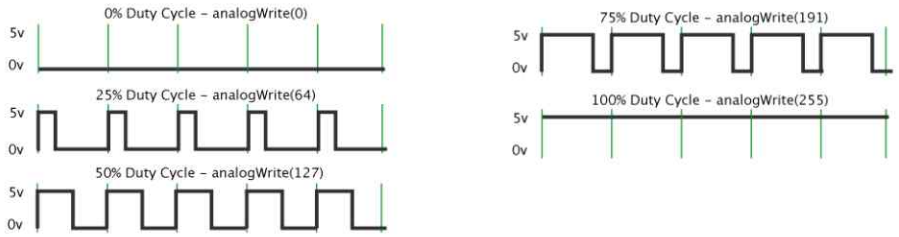
적외선 센서(IR 센서), 적외선 거리 센서(PSD 센서)를 통해 물체에 반사되는 적외선의 양에 따라 전류의 양이 달라지는 특성을 이용해 물체와의 거리를 측정한다.



적외선 센서는 색상 변화에 민감한 점을 이용해 흰색은 반사하고 검은색은 흡수하는 특징을 이용해 흰색, 검은색을 판별하는데 적합하다. 해당 값은 아날로그 값으로 데이터가 수집되고, 수집된 데이터가 실제 거리와 단위를 맞추어 주는 계산이 필요하다. 이에 따라 반복 측정을 통해 수집된 데이터의 통계를 토대로 실제 거리와의 오차를 줄이고 알고리즘을 작성한다. 적외선 센서를 통해 차량 주변의 구조물을 판단해 환경을 인지하거나 장애물을 판단해 차량 조향을 제어하고 다른 차량을 감지해 추월이나 차량 간격 유지 등의 동작에 필요한 데이터를 수집한다.

7. Step 모터 제어 기술 활용 방안 및 공부 내용

Step 모터, DC 모터를 제어하기 위해서 PWM(Pulse Width Modulation)제어 기술을 활용한다. PWM 제어는 임의의 펄스의 폭을 조절해 평균 출력 전압을 제어하는 기술이다.



출력 전압을 조정해 모터의 회전 속도를 조절할 수 있고, 이는 곧 차량 조향이나 카메라 방향 조절할 때 정밀한 제어를 할 수 있다. PWM 제어 기술의 장점으로는 출력 전압을 제어하는데 프로그래밍을 통해 제어를 할 수 있고, 전압 분배회로를 따로 배치하지 않아 전력 효율에도 도움이 된다. 또한 타이머 카운터를 이용하면 프로세서에서 부담 되는 작업량을 줄일 수 있어서 효율적이다.

□ 개발 일정

No	내용	2020年															
		6月				7月				8月				9月			
1	개발 환경 구축																
2	경기장, Test Bed구축																
3	출발, 정지 구현																
4	직선, 곡선 차선 인식																
5	내리막 인식																
6	우선정지 장애물 인식																
7	후진주차, 평행주차																
8	회전 교차로																
9	터널 주행																
10	추월 차로 코스																
11	신호등 분기점																
12	개발완료 보고서 작성																
13	시연 동영상 제작																
14	코드 최적화																
15	개발 완료																

□ 팀 구성 및 역량

No	구분	성명	팀 내 담당 업무	업무 관련 역량 (개발 언어, 프로젝트 경험 등)
1	팀장	백장현	- 영상 인식으로 기능 구현 전반 - 출발, 정지 동작 구현	○ 개발 언어 - C/ C++ /python ○ 프로젝트 경험 -matlab Simulink 기반 6개의 레이더 센서를 추가했다 가정하여 주행중 cut-in/ cut-out상황 모델링
2	팀원	김해린	- 영상 인식으로 사물 구분 기능 구현 - 우선 정지 장애물 인식 구현 - 신호동 분기점 코스 동작 구현	○ 개발 언어 - C/ C++ /python ○ 프로젝트 경험 - 시각 장애인을 위한 버스 보조 응용프로그램 개발(depth 카메라로 얻은 영상에 Faster rcnn검출기와 inception분류기를 적용하여 버스 내 하차벨, 빈좌석, 단말기, 입구를 인식하고 뎁스맵으로 카메라로부터의 거리를 측정하여 음성으로 출력하는 프로그램)
3	팀원	박찬종	- 터널 주행 구간 구현 - 회전 교차로 코스 구현 - 알고리즘 최적화	○ 개발 언어 - C/C++ ○ 프로젝트 경험 - 아두이노 보드를 활용한 스마트 미러(적외선 센서를 이용한 사용자 위치 인식 및 날짜와 시간 출력)
4	팀원	조일	- 후진주차, 평행주차 동작 구현 - 내리막 코스 구현 - 센서 데이터 활용 및 분석	○ 개발 언어 - C/C++ ○ 프로젝트 경험 - 센서 스위칭을 활용한 자율주행 RC카 제작(GPS수신이 가능한 지역에서는 GPS 위치를 인식하여 지정경로 pure pursuit 주행, GPS 수신이 불가능한 지역 (고가도로 및 터널) 에서는 Lidar, Radar 센서를 이용하여 LKAS주행)
5	팀원	허준영	- 추월 차선 코스 구현 - 구현 된 동작 오류 검출 - 알고리즘 최적화	○ 개발 언어 - C/C++ ○ 프로젝트 경험 - Hall 센서를 활용한 무선 충전 On/OFF 응용프로그램 개발(자기장의 변화로 Hall 센서가 차량을 감지해 무선 충전을 동작)