2017 지능형모형차 경진대회 보고서

학 교	한양대학교
팀 명	ACE
유 형	본 경기
팀 장	박성우 (미래자동차공학과)
팀 원	신현기 (미래자동차공학과) 박수현 (미래자동차공학과) 변효석 (미래자동차공학과) 변무경 (미래자동차공학과)

1. 개요

1.1 설계 배경

4차 산업혁명(Fourth Industrial Revolution)이라는 말이 대두하며 여러 산업에 변화의 바람이 불고 있다. 융합은 그 4차 산업혁명의 핵심적인 관건이다. 기존의 산업들은 ICT 기술과의 융합을 통해 혁신을 맞이하고 있다. 자동차 산업 역시 예외는 아니다. 지능형 자동차와 전기 자동차가 핫 토픽으로 떠오르고 뜨거운 감자가 된 것은 자연스러운 시대의 흐름인 것이다.

자율주행자동차는 그러한 지능형 자동차에서 다루는 주제 중의 하나로써 운전에 필요한 사람의 역할을 컴퓨터가 대신하여 스스로 목적지까지 주행할 수 있는 자동차를 말한다. 세계적인 IT업체인 구글은 각종 전문가들을 영입해 이 분야의 경쟁에 참여했으며 테슬라와 같은 전기자동차 업체 역시 공격적인 마케팅과 과감한 시도로 이름을 떨치며 빠른 성장세를 보이고 있다.

우리는 이번 기회를 통해 이러한 자율주행자동차를 직접 설계하고 제작함으로써 가지고있는 관련 지식을 시험하고 이해도를 높일 수 있는 계기로 삼고자 한다.

1.2 설계 목표

자율주행자동차가 스스로 주행하기 위해서는 주어진 경로와 주행상황에 따라 적절한 차량의 조향각을 생성할 필요가 있다. 여기에 사용되는 경로 추종(Path tracking)에는 여러 가지 방식이 있다. 우리는 그 중에서 복잡한 연산 없이 직관적으로 움직임을 이해할 수 있는 기하학적 경로 추종 방법의 하나인 Pure pursuit 경로 추종 제어를 사용해보고자 하였다. Pure pursuit 경로 추종 방법은 고정된 예견거리(Look ahead distance)를 사용하여 차량의 주행상황 변화에 따라 추종성능의 변화가 심하다. 따라서 MATLAB과 Simulink를 이용하여 이러한 예견거리를 결정하는데 중요한 역할을 하는 카메라가 감지하는 거리에 대한 최적화를 시행한다.

만약 속도제한구역 내에서 차량이 장애물을 만날 시에는 그것을 감지하고 차선을 변경하여 회피를 할 것이다. 이 때 이상적인 차선변경을 위해서 Zellner가 제안한 사인 함수 형태의 경로를 생성하고 그 경로를 따라 차선을 변경하는 알고리즘을 사용한다. AEB의 경우 PID를 통해 RPM을 제어할 것이다. 이때 차량의 단일계단함수 반응 그래프를 바탕으로 차량의 전달함수를 근사하고 이를 바탕으로 특정속도에서 원하는 특성을 가지고 제어할 수 있도록 PID auto tuner를 이용해 게인 값을 결정할 것이다.

2. 설계 내용

2.1 하드웨어 구성

2.1.1 센서부

< Line scan camera >

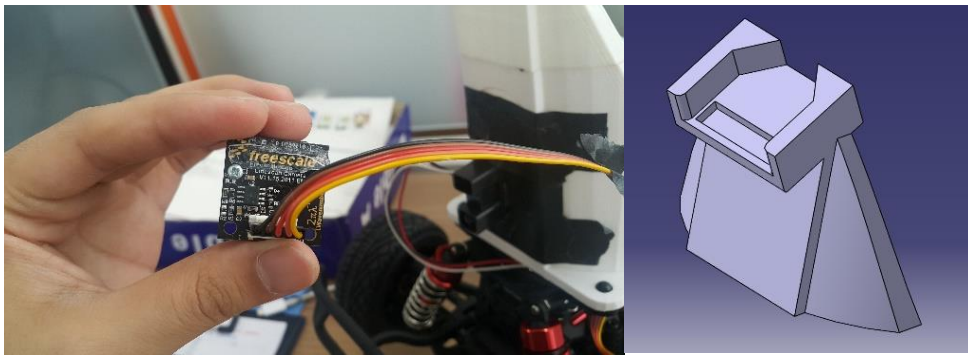

Figure 1. Line scan camera

카메라가 값을 제대로 잘 받는 것이 중요하기때문에 카메라를 고정시키는 부위를 제일 신경을 많이 썼다. 앞서 설명한 시뮬레이션 등을 통해 적당한 센싱 거리를 찾은 다음 카메라 고정부분을 3D 프린팅을 통하여 만들었다. 이전 대회에 참가했던 차량들을 보면 막대형상 위에 고정시킨 경우가 많았는데 이 경우 코너링을 하는 경우 흔들리는 현상이 일어날 것으로 예상되었다. 이를 보완하기 위해 최대한 튼튼하게 설계하였다.

< Infrared Ray Sensor >

Figure 2. Infrared Ray Sensor

우리가 원하는 위치에서 충분히 감지가 가능한 모델로 결정하였다.

2.1.2 구동부

< Encoder >

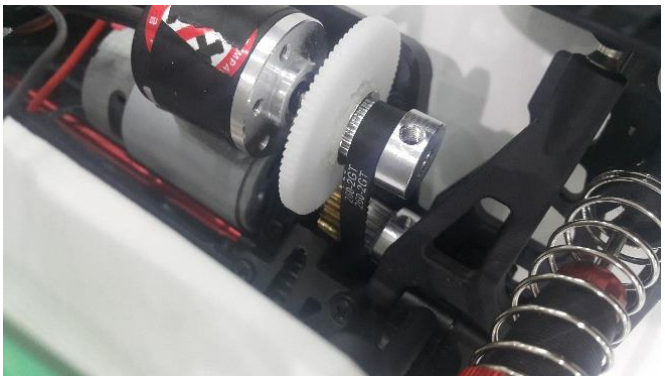

Figure 3. Encoder

엔코더를 사용하여 차량의 속도를 측정하기 위해서는 모터에 의해 회전하는 축과 기계적으로 연결될 필요가 있다. 당초에는 기어를 사용하려 했으나 엔코더에 장착한 기어의 치수가 차량 구동부의 기어 치수와 정확히 맞지 않는 문제가 있어 타이밍 벨트를 제작하여 사용하였다. 이 과정에서 기어를 덮고 있던 덮개를 제거하였다.

< 회로구성 >

엔코더로 회전에 대한 정보는 얻을 수 있으나 단순히 그것만 보아서는 방향성에 대한 정보까지는 알 수 없다. 따라서 정회전과 역회전의 구분을 위해 D Flip-flop 과 And Gate 를 사용하였다.

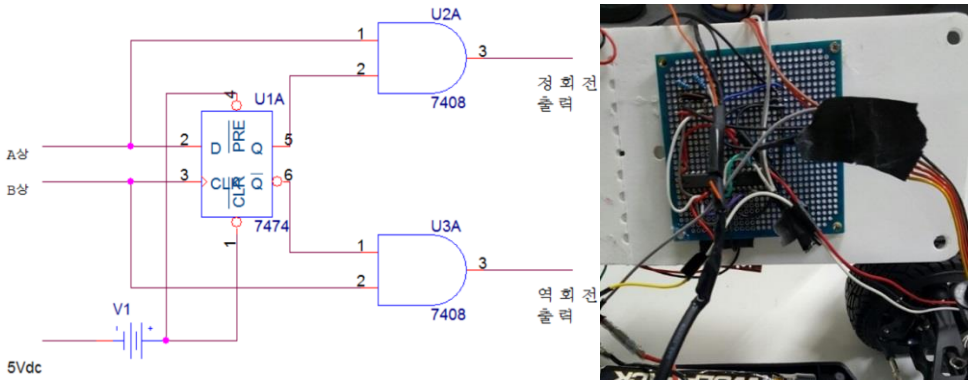

Figure 4. Circuit

A 상과 B 상의 출력 관계에 의해 정회전시 Q, 역회전시 $\bar{Q}$에 1 이 출력된다. 엔코더의 출력과 D Flip-flop 의 출력, 그리고 And gate 를 통해 정회전과 역회전의 출력을 구분하였다.

2.1.2 조향부

< Servo Motor >

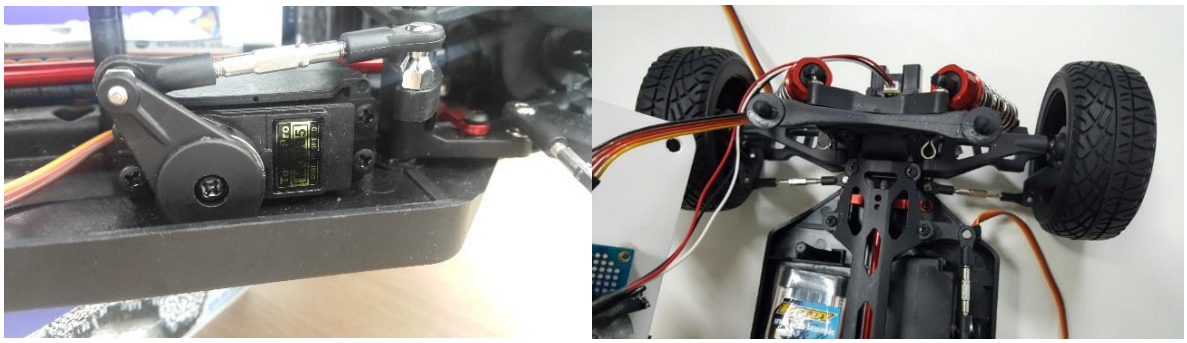

Figure 5. Servo motor

우리가 원하는 신호를 주었을 때 즉각 반응해줄 수 있는 출력이 좋은 제품을 찾는데 주력하였다.

2.1.3 전체 하드웨어 외관

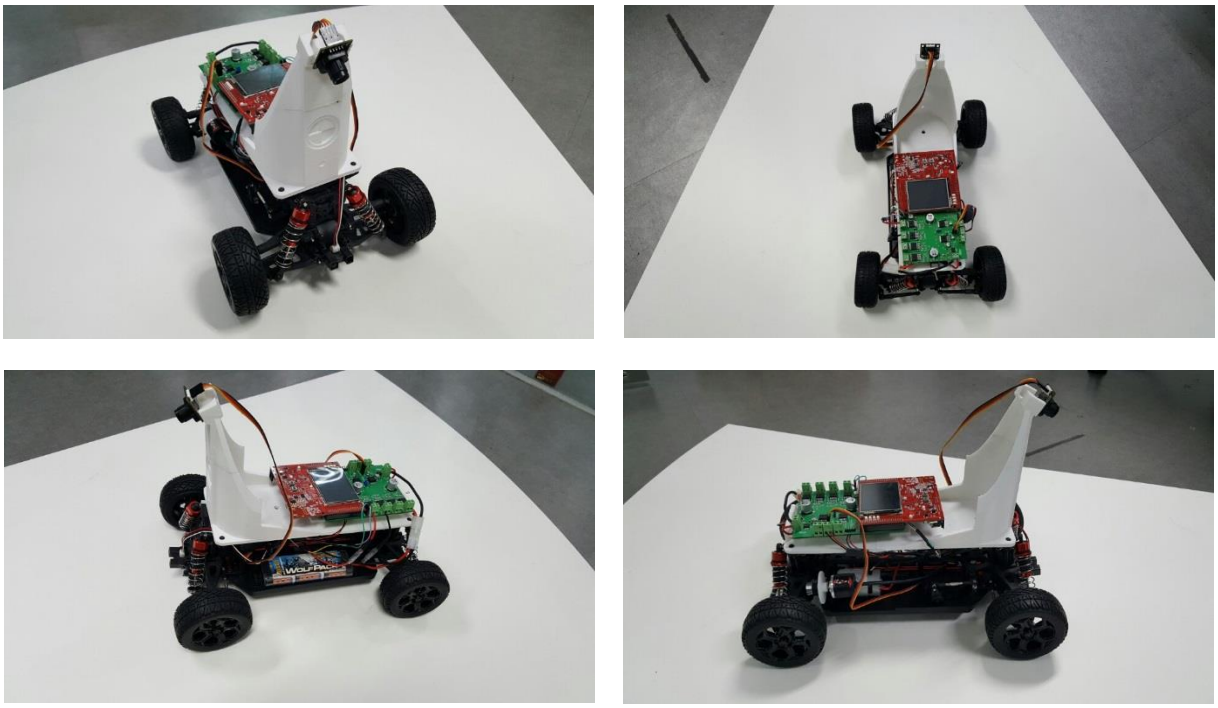

Figure 6. Full shot

2.2 소프트웨어 구성

2.2.1 카메라 신호처리

장치된 카메라는 일정시간마다 발생하는 interrupt에 의해 촬영을 하고 데이터를 MCU에 전달한다. 이 때 데이터의 정확도와 신뢰성을 위해 Median 필터와 미분필터가 사용된다.

2.2.2 경로 추종

기본적으로 기하학적 경로 추종 알고리즘인 Pure Pursuit을 사용한다. 만약 주행 중 속도제한구역에 진입하는 등에 의해 차량의 속도가 변화하는 경우 그에 맞추어 이론적으로 계산된 steering gain을 조향각에 보상한다. 또한 장애물회피를 해야 한다고 인식하게 되면 잠시 Pure Pursuit을 끄고 장애물 회피모드로 진입한다.

2.2.3 속도제한구역 인식

카메라로 도로 중앙부 일정 영역의 출력 값을 관찰한다. 이것을 저장해 두었다가 차량이 흰색 도로를 달리다 검은색 속도제한구역 표시선을 지나갔다는 것을 인식하면 속도를 줄인다. 다시 한 번 발생한다면 속도제한구역을 벗어났다고 이해하여 속도를 복구한다. 이때 도로 바깥 영역이 어둡다면 코너링 중에 속도제한구역을 인식할 수도 있으므로 차량의 현재각도로 제약조건을 걸어 둔다.

2.2.4 장애물 회피

만약 속도제한구역 내부에서 장애물을 관측한다면 회피 알고리즘을 실행시킨다. 차량은 잠시 Pure Pursuit을 끄고 차선변경 이론을 바탕으로 실험적으로 결정된 조향각으로 운행한다. 차선변경이 완료되었다고 인식되면 다시 Pure Pursuit을 작동시킨다.

2.2.5 AEB

만약 속도제한구역 외부에서 장애물을 관측한다면 긴급정지 알고리즘을 실행시킨다. 차량은 모터에 역회전을 걸어주는 식으로 감속을 하면서 내부에 설계 된 PID 제어기에 따라 제어되어 정지한다.

3. 주요 장치 이론 및 적용 방법

3.1 Filtering

필터링은 총 세 단계의 과정을 거친다. 일단 최우선적인 과제는 라인의 판별이고 따라서 edge detection kernel 을 적용했다. 하지만 edge detection 은 노이즈에 민감하기 때문에 우선 노이즈를 잡아주었다. 노이즈를 잡기 위해서는 Low pass filter 를 거쳤다. 이 때 사용한 것은 gaussian blur 이다. Median filter (box blur 라고도 한다.)와 다른 점은 box blur 의 경우 인접한 픽셀의 값을 단순히 평균을 내주는 반면 gaussian blur 의 경우에는 gaussian function(정규분포곡선)에 따라 가중치를 부여해준다는 점이다. 이 때 Edge detection 이 노이즈에 영향을 많이 받기 때문에 gaussian blur 시 표준편차를 크게 해주었다. 다시 말해 넓은 범위에 걸쳐 픽셀을 참고했다. 이번 경우에는 7 개의 픽셀을 사용했다. 실험을 해본 결과 7 개정도는 되어야 edge detection 시 edge 와 noise 를 뚜렷하게 구분하였기 때문이다. 이 때 부여한 가중치는 아래와 같다.

$\frac{1}{64}$	$\frac{6}{64}$	$\frac{15}{64}$	$\frac{20}{64}$	$\frac{15}{64}$	$\frac{6}{64}$	$\frac{1}{64}$

원래는 gaussian function 에 따른 continuous 한 수치를 사용해야 하지만 픽셀로 이루어져 불연속하기 때문에 discrete 하게 처리하였다. 상대적인 수치만 알면 되기 때문에 여기서 분자만을 따져 정수로 만들었다.

1	6	15	20	15	6	1

노이즈를 감소시킨 후 edge detection 을 시행했다. edge detection 을 할 때는 인접한 3 개의 픽셀을 사용했다. 세 픽셀에 가해지는 웨이트를 합해 0 이 되도록 웨이트를 정해주었는데, 좌우의 웨이트가 같아야 하므로 이 때의 웨이트는 각각 -1, 2, -1 으로 해주었다. 앞서 gaussian blur 를 통해 구한 계수에 적용하여 표로 나타내면 다음과 같다.

-1	-6	-15	-20	-15	-6	-1		
	2	12	30	40	30	12	2	
		-1	-6	-15	-20	-15	-6	-1

이것을 합해주면

-1	-4	-4	4	10	4	-4	-4	-1

여기 까지가 edge detection 이지만 생각보다 노이즈가 심하여서 최종적으로 한 번 더 gaussian filter 를 적용했다. 이번에는 인접한 5 개의 픽셀만 참고하였고 가중치는 아래와 같다.

1	4	6	4	1

이후 앞선 과정을 똑같이 거치면 최종 계수는 다음과 같다.

-1	-8	-26	-40	-15	48	84	48	-15	-40	-26	-8	-1

이 때 raw data 에서 라인의 수치가 음수로 나오기 때문에 직관성을 위해 -1 을 곱해주었다. 아래의 그래프들은 각각 raw data 와 필터링 후의 data 를 보여준다. 참고로 우리는 카메라를 뒤집어서 사용하기 때문에 좌우는 반전 되어있다.

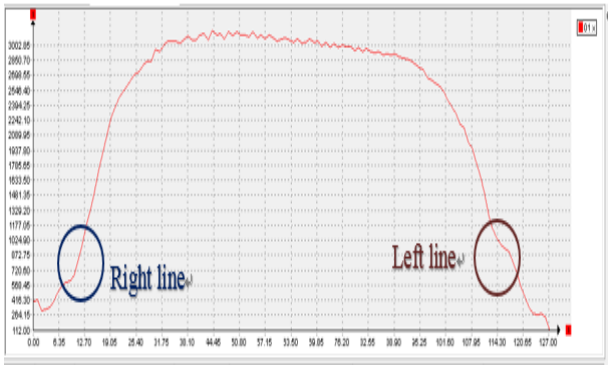

Figure 7. Raw data

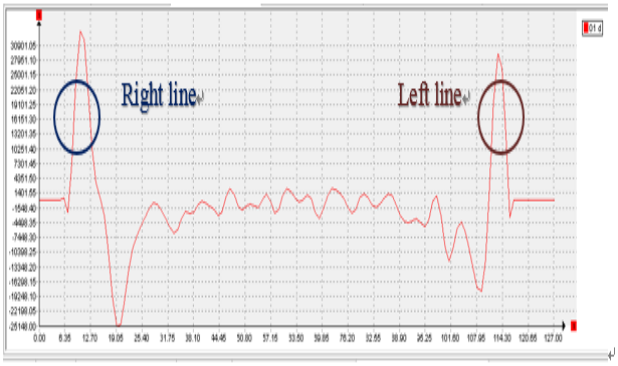

Figure 8. Filtering data

3.2 Pure Pursuit

3.2.1 Pure pursuit 경로 추종

Pure Pursuit 경로 추종은 기하학적 경로 추종 방법의 하나로써 현재 차량의 위치와 일정 거리 밖의 위치의 오차를 이용하여 기하학적으로 조향각을 발생시키는 알고리즘이다. 카메라가 일정 거리 밖의 Track 위의 어느 지점을 인식하면 그곳을 예견지점으로 잡은 후, 그 지점과 현재 차량의 위치 사이에 원호를 그리며 접근해간다.

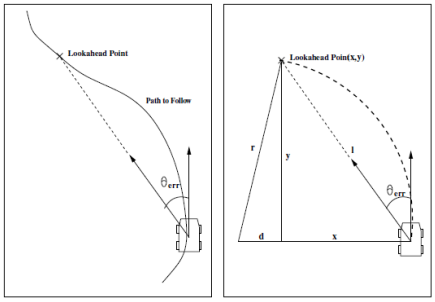

Figure 9. Pure pursuit

Fig.9 는 그러한 과정, 차량이 쫓아가야 할 도로 위의 한 점을 선정하고(좌) 그곳을 향해 나아갈 원호를 생성하는 모습(우)을 보여주고 있다. 이후 삼각형의 기하학적인 관계를 통해 다음과 같은 식 (2.1) ~ (2.3)을 도출한다.

$$R = d + x \quad (2.1)$$

$$R^2 = d^2 + y^2 \quad (2.2)$$

$$l^2 = y^2 + x^2 \quad (2.3)$$

이후 (2.2)에서 d와 y를 소거하면 (2.4)와 같은 결과가 나온다.

$$R = \frac{l^2}{2x} \quad (2.4)$$

3.2.2 애커만(Ackermann) 조향 기하학

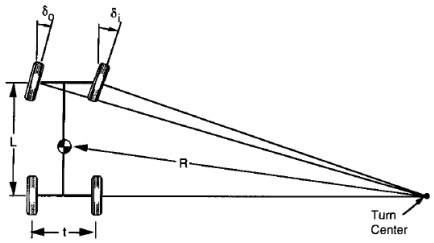

Figure 10. Geometry of a turning vehicle

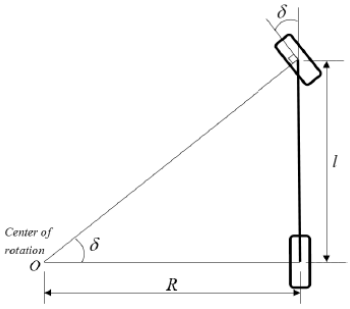

Figure 11. Bicycle model

Fig.10는 차량이 회전할 때의 기하학적인 모습을 나타낸 것이다. 이 때 동심원을 그리며 원활하게 회전하기 위해서는 양 바퀴의 회전각이 달라야 한다. 그러나 차가 저속이며 슬립이 일어나지 않는 구속 상태라 가정할 때 다음과 같은 자전거 모형으로 단순화가 가능하다. 이것은 Fig. 11과 같이 조향각을 회전반경과 축거의 관계로 표현할 수 있으며 수식으로는 다음과 같다.

$$\tan(\delta) = \frac{L}{R} \quad (2.5)$$

이 수식을 통해 앞서 도출해 낸 회전반경을 바탕으로 발생시켜야 할 조향각도를 계산할 수 있다.

3.2.3 카메라 감지 거리

앞서 Track을 감지하고 적절한 조향각을 계산하여 발생시킬 수 있음을 보였다. 이 때 이러한 조향각에 의한

주행성능은 차량이 어느 정도 멀리 떨어진 거리를 관측하고 있느냐 에 좌우될 수 있다. 카메라가 차량이 향한 방향으로 원뿔모양으로 경로를 감지한다고 보았을 때, 같은 오차를 감지하더라도 카메라가 멀리 보고 있었다면 예견거리 값이 길어져 (2.4)식에 의해 회전반경은 커지게 된다. 즉 결과적으로 완만한 steering을 하게 되어 안정적인 움직임을 보이지만 원하는 위치까지 도달하는 settling time은 길어지게 된다.

3.2.3.1 경향성 분석

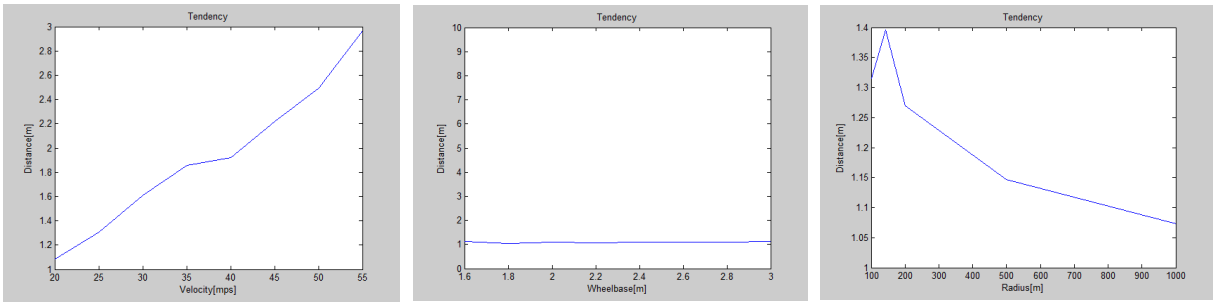

Figure 12. Tendency by velocity, wheelbase, radius

Fig.12 는 좌측에서부터 속도, 축간 거리, track 의 회전 반지름에 따른 최적 카메라 감지거리의 경향성을 나타낸 것이다. 보다시피 속도에는 비례하며 축간 거리에는 거의 영향을 받지 않고 track 의 회전 반지름에는 반비례하는 모습을 보였다. 속도에 비례하여 멀리 바라봐야 좋은 주행을 할 수 있는 것은 실제 차량 운전과 같아 납득하기 쉬웠다. 그러나 track 의 커브가 급해질수록 더 먼 감지거리를 지니는 것은 직관적으로 잘 이해되지 않았다. 감지거리가 짧다는 것은 오차의 발생에 대해 좀 더 즉각적인 반응을 한다는 것이고 그것은 급격한 커브변화에 대처하기 좋을 것이기 때문이다.

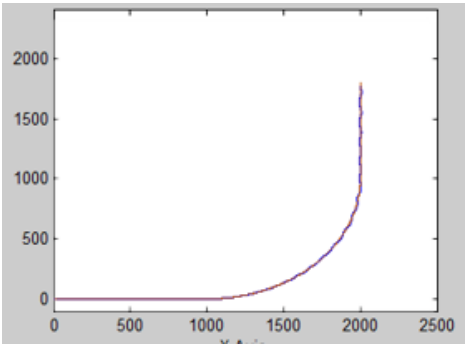

Figure 13. Trajectory of vehicle

답은 시뮬레이션을 통해 얻을 수 있었다. Fig.13 은 짧은 감지거리를 지닐 때의 차와 track 의 궤적을 나타낸 것이다. 붉은 선은 도로를 파란 선은 그것을 추종하여 나아가는 차의 궤적이다. 여기서 카메라의 감지

거리가 짧을수록 커브 시 발생한 오차에 의해 생기는 oscillation 이 감쇄되는 속도가 늦어지는 것을 발견했다. 거리가 짧아질수록 이 경향은 심해져 oscillation 이 끝까지 유지되는 경우도 있었고 더욱 짧은 거리를 볼 시에는 오히려 oscillation 이 발산해버리는 경우도 발생했다.

3.2.3.2 Feedforward 보상을 통한 주행성능 향상

속도 그래프는 MATLAB 의 Basic fitting tool 을 이용해 수식화 하였다. 그러나 이 수치에 따라 주행 중 카메라의 감지거리를 변경하는 것은 센서의 감도에 따른 threshold 나 line 인식 위치를 변경해야 하는 등 어려움이 많다. 때문에 그 대신 속도에 따른 적절한 조향각을 보상함으로써 주행성능을 개선해보기로 하였다. 이를 위해 이전의 식 (2.4)와 (2.5)를 이용해 다음과 같은 수식을 도출했다.

$$\tan(\delta) = \frac{2xL}{l^2} \tag{2.6}$$

이 때 차량의 속도가 미세하게 변화해간다면 감지거리와 조향각 역시 충분히 미세하게 돌아간다고 가정한다면 $\tan(\delta)$와 δ는 같다고 볼 수 있고 미세한 수직오차에서 카메라 감지거리와 예견거리를 같다고 볼 수 있으므로 다음과 같이 전개할 수 있다.

$$(\delta + \triangle\delta) - \delta = 2xL\left(\frac{1}{l^2 + \triangle l^2} - \frac{1}{l^2}\right) \tag{2.7}$$

$$\triangle\delta = 2xL\left(\frac{-\triangle l^2}{(l^2 + \triangle l^2)l^2}\right) \tag{2.8}$$

$$\triangle\delta = 2xL\left(\frac{-\triangle l^2}{l^4}\right) \tag{2.9}$$

$$\triangle\delta = \delta\left(\frac{-\triangle l^2}{l^2}\right) \tag{2.10}$$

식 (2.10)은 결과적으로 예견거리가 바뀌었을 때 조향각에 주는 영향을 수치화 한 것이다. 이것을 예견거리를 변화시키는 만큼 조향각에 보상함으로써 예견거리를 변화시키는 효과를 낼 수 있을 것이다.

회전 반경에 의한 영향은 대회규정대로 실제 track 을 구성하여 실험적으로 결정해 보상해주었다.

3.2.4 장애물 회피 시 경로생성

Zellner는 이상적인 차선 변경으로 다음과 같은 사인 함수 형태의 경로를 제안했다.

$$y(x) = y_s[\frac{x}{x_s} - \frac{1}{2\pi}sin(\frac{2\pi x}{x_s})]$$

$$x \doteq V_0 t$$

$$a_{y,peak} \doteq \frac{y_s}{x_s^2} 2\pi V_0^2 \quad (2.11)$$

이것을 이용하여 도로 폭과 허용 횡가속도가 주어졌을 때 차선을 변경하는 이상적인 궤도를 그릴 수 있다.

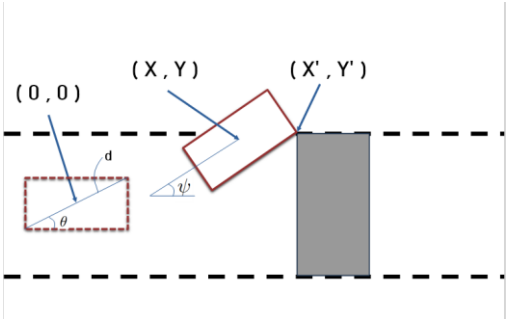

Figure 14. obstacle collision point

Fig. 14 는 장애물을 회피할 때 차량의 기하학적인 모습을 표현한 것이다. 이것을 이용해 좌표변환 한다면 앞서 구했던 차량의 질량중심이 움직이는 궤적에서 차량 최외각(最外殼)의 궤적을 구할 수 있고 차량이 코너링을 돌 때 이 부분에서 충돌이 일어난다고 가정한다. 그리고 이것을 이용해 장회물을 회피할 수 있도록 회피를 시작해야하는 최소거리를 구했다.

3.2.5 AEB

자동 긴급 제동(Autonomous Emergency Braking)의 경우 차를 제어하며 생기는 약간의 overshoot 나 steady state error 에도 충돌이 생길 수 있기 때문에 MATLAB 의 auto tuner 를 사용해 PID 제어의 제어특성을 원하는 대로 결정하고자 하였다. 이를 위해 차량의 속도제어에 대한 반응을 바탕으로 플랜트의 전달함수를 근사였다. 만약 플랜트에 적분기가 포함되어 있지 않고 주 극점이 복소근이 아닌 안정된 시스템이라면 개로 계단응답곡선의 그래프는 다음과 같이 S 형을 띄게 된다.

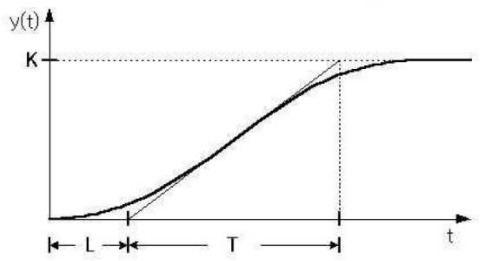

Figure 15. S shape response curve

이 때 지연시간(L), 시정수(T), 직류이득(K)의 세 가지 계수를 안다면 플랜트의 전달함수는 다음과 같이 근사할 수 있다.

$$\frac{Y(s)}{U(s)} = K\frac{e^{-Ls}}{Ts+1} \qquad (2.12)$$

3.2.5.1 AEB 를 위한 PID 게인값 결정

정지상태의 차량에 최종 속도를 입력해 거기까지 속도가 도달하는 모습을 그래프로 나타냈다.

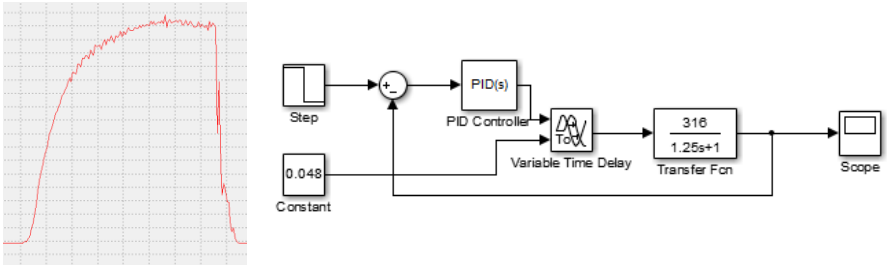

Figure 16. S shape response curve of car

그리고 2.1.1.5 에서 언급했던 이론을 사용하여 여기서 관찰되는 지연시간(L), 시정수(T), 직류이득(K)을 바탕으로 전달함수를 만들어 simulink 를 구성했다.

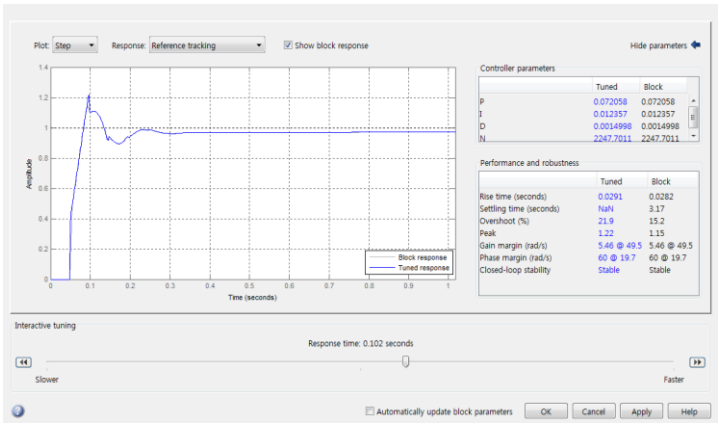

Figure 17. PID tuning

이후 MATLAB 내부의 튜너를 이용해 원하는 모양과 특성을 가지도록 제어했다. 게인 값이 생각보다 작게 나와서 실제 차량에 적용해도 될지 의구심이 생겼지만 실제로는 이론대로 잘 작동되는 모습을 보였다.

4. 결론 및 토의

4.1 결론

4.1.1 Path Tracking

카메라의 스펙이 한정되어 있는 상황에서 라인스캔의 안정성을 위해 카메라를 단단하게 고정하는 방식을 채택하였다. 그러나 우리가 사용한 Pure pursuit 경로 추종 알고리즘은 속도에 따라 가변적인 카메라 감지 거리를 가질 때 좋은 성능을 낼 수 있다. 때문에 고정된 카메라는 그대로 고정해두면서 속도에 따라 감지 거리를 바꾸는 대신 그 만큼의 적절한 steering 게인 값을 수식적으로 계산하여 보상하는 방식으로 주행성능을 개선하고자 하였다.

그것을 위해 시뮬레이션을 구성하여 카메라 감지거리의 경향성을 파악하고 수식화 하는 작업을 거쳤다. 이번 대회에서 우리 모형차의 경우 속도 제한 구역 밖과 안, 2 가지 속도를 가지고 주행하는데 이 중에서 빠른 속도를 기준으로 카메라를 고정하고 속도 제한 구역에서는 위에서 구한 게인 값을 보상해주는 식으로 코딩을 짰으며 성공적인 주행 성능 개선 효과를 얻을 수 있었다.

4.1.2 Lane Change

장애물을 회피하면서 차량은 차선을 변경하게 된다. 이 때 차량의 기하학적인 조건을 고려하면서 이상적인 차선 변경 곡선에 맞추어 회전을 할 수 있는 최소 회피거리를 구하였다. 이것을 차량에 적용하면서 구체적인 steering 각도는 실험을 통해 세밀 조정했다.

4.1.3 AEB

AEB 를 통해 제동을 할 때 settling time 이나 overshoot 등의 특성을 원하는 수치로 조절하여 미끄러지는 거리나 시간 등을 완벽히 제어하고자 하였다. 그것을 위해 차량에 개루프 계단함수의 입력을 주고 나오는 속도 반응곡선으로 시뮬레이션을 위한 전달함수를 근사하고 MATLAB 의 auto tuner 를 이용하여 특성치를 조절하여 모형차에 적용하였다. 결과로 나온 게인 값들이 생각보다 너무 낮아 걱정되었지만 생각 이상으로 잘 작동되었다.

4.2 토의

박성우 : 경진대회 같은 곳에 참여한 것은 이번이 처음이었다. 대학교에서는 이론만 공부했지 실제로 적용할 수 있는지는 감이 안 잡혔는데 이번 기회에 이것 저것 직접 만들어보면서 어느정도 가닥이 잡혔다. 물론 생각보다 쉽지는 않았다. 내가 잘못 생각했던 부분도 많아서 팀원들과 주변 사람들에게 도움을 받았다. 그리고 소프트웨어와 하드웨어 설계에 있어서 역량이 필요하다는 것을 새삼 또 느꼈다.

신현기 : 여러가지 알고리즘을 디자인하고 만드는 일은 생각보다 즐거웠다. 그러나 단순히 머리속으로 생각하는 것과 그것을 프로그래밍 언어로 표현하고 회로로 설계하는 것은 또 완전히 다른 일이라는 것을 알게 되었다. 예기치 못한 문제에 맞닥뜨렸던 경우도 많았다. 그럴 때마다 부족한 나에게 아낌없이 도움을 준 팀원들에게 감사를 전하고 싶다.

박수현 : 모형차의 주행시에 발생하는 여러 상황들을 한정된 센서의 정보만을 바탕으로 인식하는 것이 힘들었다. 하지만 이번에 실제로 그러한 고민들을 해보고 알고리즘을 짜는 경험을 통해서 좀 더 성장할 수 있었던 것 같다.

변효석 : 지능형 모형차를 설계하면서 직접 차량을 디자인 해볼 수 있는 기회가 되었다. 카메라를 고정시켜야 한다는 한계로 인해 원하는 디자인은 아니었지만 설계 시 고려해야할 것들을 한번 생각해 볼 수 있는 좋은 경험이었다.

변무경 : 하드웨어를 구성함에 있어 크고 작은 여러 문제를 맞닥뜨렸다. 지나고 나서 생각해보면 이렇게 접근했더라면 좋았겠다 하는 것들이 많다. 무언가를 제작하는 데에 있어 보다 효율적인 접근이나 원활한 작업을 위한 순서 등에 대한 지식을 경험적으로 습득하는 기회가 되었다.

Appendix – 부품 목록

제조사	부품 명	수량	사용 목적
Ready	AAK694 Wolf Pack 7.2V 3000mAh Ni-MH	1	전원공급
SHARP	GP2Y0A21YK	1	장애물 판단
TowerPro	MG-995	1	조향
Autonics	E30S4-3000-3-N-5	1	차량 속도 검출
TAOS	TSL-1401	1	차선인식
Hitachi	HD74LS74AP	1	D Flip-flop
NXP	74HC08N	1	And Gate
	1k ohm resistance	2	저항