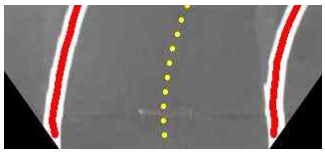
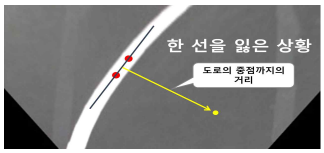
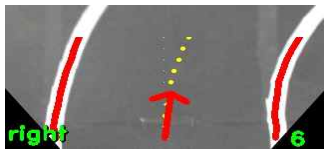
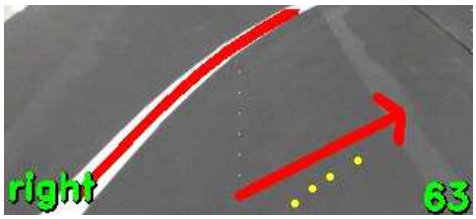
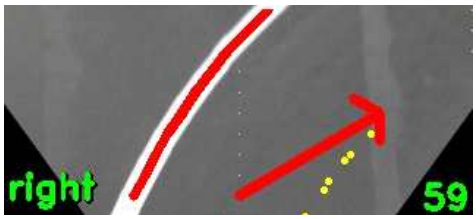
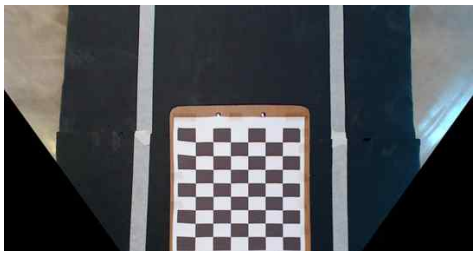
※ 접수번호	

종합설계 최종 과제보고서

전공	기계, 전자, 정보통신, 컴퓨터	팀명	Hermes

팀원 명	성명		성명	
		성현기		한형진
		민경규		문성주
		김진현		이힘찬
		최정은		장우진
		김시현		하선우
		이창혁		박성준
		박상훈		김광현
		배성현		임정혁
		박대희		

작품(과제)명	트럭 플래투닝 프로토타입
1. 개발동기 및 목적, 필요성	Hermes는 화물 운송의 비효율성과 물류비를 해결할 혁신적인 변화가 필요함을 느끼고 이를 공학적으로 해결하고자 한다. 육상, 해상, 항공 등 많은 운송 수단이 있지만 국내의 경우 지형 특성상 차량을 이용하여 운송을 하는 비중이 크다. 현재 국내 도로운송의 비중은 80%이상을 차지하며 최근 다년간 고속도로를 통행하는 화물차의 수가 증가하는 추세이다. 차량을 이용하여 원거리 운송 시 운임이 비싸고, 교통사고와 공해문제로 사회적 문제를 일으킬 수 있다. 또한, 중량에 제한이 많아 운송의 단위가 작아 대량화물의 운송에 부적합하며, 운송 중 사고발생률이 높다. 그리고 유류비 및 인건비의 상승으로 인해 물류비가 지속적으로 증가하고 있는 상황이다. 따라서 물류산업에서 차량의 운송 시스템이 현재와 같이 지속된다면 물류 운송의 효율성은 점점 감소될 것이다. 이에 대비하여 국내에서 화물운송의 비효율성과 물류비를 해결할 새로운 기술이 필요함을 확인할 수 있었다.
2. 과제 해결 방안 및 과정	Hermes는 자율주행 기술 중 하나인 트럭 플래투닝 기술을 프로토타입을 제작하여 구현하고자 한다. 트럭 플래투닝은 차량 간의 통신인 V2V로 이루어진 군집주행 기술이다. 선두 차량의 운전자가 운전하면 뒤의 차량들이 차선 및 차간 거리를 유지하며 따라가는 자율주행 기술이다. 선두 차량 하나를 운전함으로써 여러 대의 차량을 제어할 수 있는 시스템이다. 여러 대의 차량들이 하나의 그룹을 이루며 군집주행하게 되므로 교통, 연비의 효율과 후발차량들의 자율주행이라는 이점을 갖고 있다. Hermes 팀은 플래투닝 시스템의 프로토타입을 구현하기 위해 Arduino, Raspberry Pi를 활용한 RC카를 제작하였다. 차선인식(Lane Detection), 차간거리유지(Adaptive Cruise Control), V2V(Vehicle-to-Vehicle)통신, 객체 추적(Object Tracking) 기술을 개발했고 장애물이 있는 상황에서도 플래투닝 시스템을 유지할 수 있는 강인성을 보여주기 위해 장애물 회피(Obstacle Avoidance)기술도 추가하였다. 마지막으로 연비 효율을 위한 공기역학적 차체를 직접 설계하기 위해 UG-NX툴을 활용하였고 3D Printing을 통해 제작하였다.
3. 출품과제의 기술	1. Lane Detection : 선행 RC카의 라즈베리파이에 카메라를 연결하여 영상처리를 하도록 한다. 그리고 Pid Control을 이용하여 차선을 안정적으로 유지할 수 있도록 한다. 2. Car Tracking : 선행 RC카의 외형을 목표로 하는 데이터를 트레이닝 하여 이를 추적(Tracking)할 수 있도록 한다. 3. Communication : 후발 RC카는 선행 RC카의 속도 정보를 V2V통신을 통해 전달받아 실시간 감/가속 및 정지/출발이 가능하도록 한다. 4. Cruise Control : 후발 RC카는 설정한 차간 거리를 유지하며 선행 RC카의 속도로 주행한다. Fuzzy Logic Control을 이용하여 구현하였다. 5. Obstacle Avoidance : 초음파센서를 이용하여 장애물을 인지하고 이를 회피할 수 있도록 한다.

	6. Car Design : UG-NX 툴을 이용하여 차체를 설계하고 3D Printer로 출력하여 제작한다. 현재까지 선행 RC카의 차를 제작하였다.
4. 개념설계 및 상세설계(계산)	**1. Lane Detection** 자율 주행을 위해 외부상황에 따라, 조향방향을 결정해야 한다. Lane Detection은 이를 위해 카메라를 통해 도로의 영상을 받고, 이를 영상처리하여 조향에 필요한 각도 정보를 계산하는 부분을 맡고 있다. 영상처리의 과정은 다음과 같다. ① : 카메라로부터 들어온 이미지를 위에서 바라보는 Top View로 변환해준다. ② : Top View 상태의 이미지에서 본격적인 차선을 인식하기 위해, 전처리 과정을 거친다. ③ : ②의 이미지에서 차선을 Sampling하는 방식으로 점을 얻어 차선을 인식한다.  ④-1 : 얻어진 양쪽 차선의 점들을 이용하여 중선을 구한다.  ④-2 : 한 선이 보이지 않는 상황에서는 한 쪽선으로부터 중점까지의 거리를 계산하여 구한다.  ⑤ : 차량의 중심과 중선과의 차이를 통해 각도를 계산한다. 위의 과정을 각각 상세하게 설명하면 다음과 같다. ① : 카메라를 통해 들어오는 이미지는 카메라의 Perspective Projection에 의해 현실 세계의 3D 개체가 2D 이미지로 변환된 것이다. 따라서 이때의 이미지는 현실과는 다른 변형이 생기며, 이를 통한 도로의 각도를 구하는 것은 왜곡된 각도를 구하게 된다.   이런 문제점을 해결하기 위해, Checker Board를 통해 카메라의 행렬을 구하고 이를 이용하는 Warping을 수행하여 위에서 바라보는 Top View를 구함으로써, 현실 세계의 도로에서의 각도를 구할 수 있게 된다.  또한, Top View의 이미지를 사용하면, 한 선이 화면에 나오지 않는 상황에서 한쪽 선을 통해 중앙선을 추정할 수 있는 알고리즘이 근거가 된다. (④-2 참고) ② : 우리 알고리즘에서는 영상을 처리하기 위해서 원본 영상을 도로의 차선만을 가진 데이터로 전처리를 해준다. 이때의 전처리는 다음과 같은 과정을 거친다. 먼저, 장애물 인식의 거리와 차간 거리 등을 고려하여 ROI를 설정한다. ROI 내에서 도로의 차선과 배경을 구분하기 위해 RGB를 HSV로 변환한다. HSV의 Saturation과 Value의 값을 이용하며,, 적절한 Threshold 값을 설정하여 차선과 그 외 나머지로 구분되는 이진화 된 이미지를 얻을 수 있다. 이때의 적절한

Threshold 값은 다양한 시간대(오전, 오후)와 장소(실외-자연광, 실내-형광등)에서 얻은 영상에서의 실험적으로 얻은 결과이다.
③ : 이진화된 이미지에서는 차선이 나타나 있지만, 이를 사용하기 위한 Data로서의 변환이 필요하다. 선은 점의 집합이므로, 이미지에서 일정한 y축 간격마다 Sampling을 하는 방식으로, 차선의 중점을 따로 추출하였다. 중점을 추출하는 방법은 다음과 같다. 도로에서 차량이 달릴 수 있는 주행 부분에서 기준점을 생성하고, 이 기준점으로부터 처음 만나는 임계값과 두 번째 만나는 임계값의 중점을 차선의 중점으로 정의하고 구하였다. 이때의 기준점은 주행부분 내에서 빛이 거울에서 반사되듯이 지그재그로 판단하여 설정한다. 또한, 기준점을 생성하는 시작점은 이전의 도로의 중점을 임시로 저장하였다가 갱신하여 사용하는 방식으로 변화되는 상황에서도 강인하게 찾을 수 있으며, 도로의 밖으로 벗어나 주행하여도 회귀하도록 하는 각도를 구할 수 있게 하였다. 이 Sampling 방식의 장점은 차선을 점의 Data로 변환하여, 점 사이의 기울기를 구하면 차선인지 곡선인지 여부의 판단이 가능하며, 한 쪽 선만 있을 경우에도 계산을 통해 중점을 구하는 등의 선에 대한 처리에 대해 수식을 통해 구하는 방법보다 쉽고 빠르게 구하여 사용할 수 있다.
④-1 : 얻어진 차선의 점들을 통해 중점을 구하면 차량이 나아갈 목표인 도로의 중점을 얻을 수 있다.
④-2 급한 곡선 같은 경우에는 도로의 차선이 하나만 나오게 된다. 이 때는 양쪽 차선의 중점을 통해 도로의 중점을 구할 수 없기 때문에, 한 쪽 차선을 통해서 구한다. 이미 ①에서 Top View로 변환하였으므로, 왜곡이 없다. 따라서, 이 이미지에서 규정된 도로의 차선의 크기의 반을 한쪽 차선으로부터 계산하여 구하도록 하여 도로의 중점을 구하였다.
⑤ : 최종적으로 차량이 나아갈 방향의 각도를 구하기 이전에 정의를 하면 다음과 같다. 화면의 중점이 차량의 중심으로 정의하고, 이로부터 수직선이 각도의 기준선이 되도록 하였다. 이 차량의 중심점으로부터 도로의 목표점까지 직선을 형성하고, 이 방향이 차량이 나아갈 방향이며, 수직선으로 부터의 각도가 제어에 전달해줄 Servo의 각도로 하였다. 이때의 도로의 목표점을 구하는 방식은 Linear Regression을 이용하여 도로의 중점들에 의해 가장 잘 나타내는 직선을 구하고 이미지 상에서 직선의 중간에 위치한 점을 목표점으로 정의하였다. Linear Regression은 다음의 수식을 이용하여 Data들을 직선으로 표현하는 방법이다.

$$a = \bar{y} - b\bar{x} \quad b = \frac{n\sum yx - \sum x\sum y}{\left[n\sum x^2 - \left(\sum x\right)^2\right]} \quad y = a + bx$$

단순히 중점을 사용하지 않은 이유는 차선의 일부분이 인식이 안 되거나 잘못된 인식 등의 외부의 상황에서도 강인하게 계산하기 위해 사용하였다. 최종적으로 계산된 이 조향 각도를 제어에 전달한다.

2. PID Control

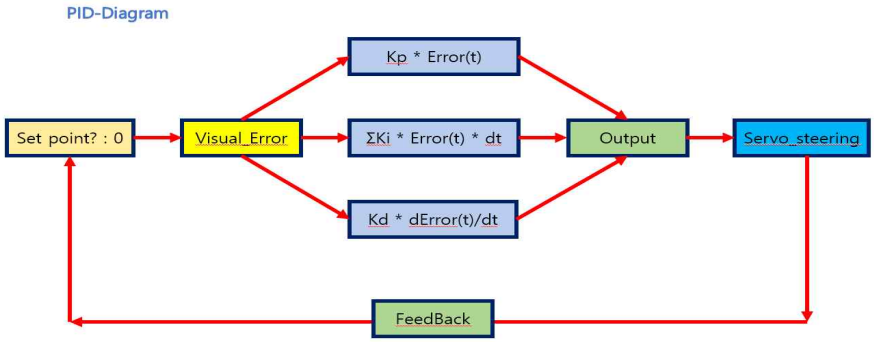
PID의 목표는 자율주행 트럭의 궤적이 차선의 중점의 궤적과 근사하게 만드는 것이었다. 영상 처리부에서 들어오는 각도와 서보모터의 조향각의 비례관계만을 이용하여 제어기를 설계할 시, zig-zag한 진폭을 가진 궤적은 필연적이며 이것은 안정감 있는 차량제어와는 거리가 멀 것이다. 영상 처리부에서 들어온 실시간 visual_error값을 사용했을 뿐만 아니라 이전 error을 실시간 제어에 사용하는 아두이노 코드를 구성함으로서 PID제어를 구현하였다.

최종 서보모터 제어를 위해 필요한 각도 값은

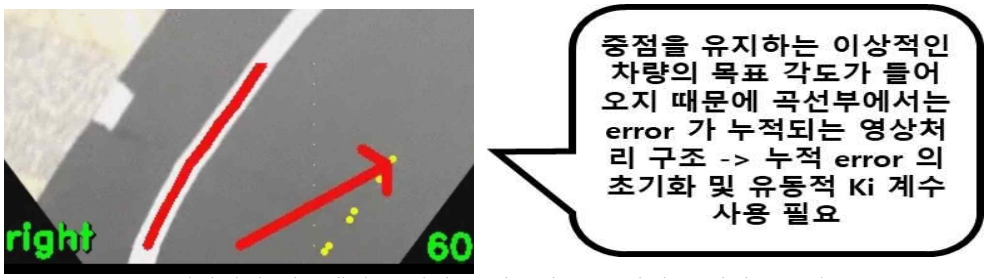
dT = 명령을 실행시키는 loop문의 간격 혹은 한 프레임이 처리되는 시간
integral_error(누적된 에러의 합) += error_previous(실행 이전 에러)
Servo_angle = Kp*error + Ki*(integral_error)*dT + Kd *(error-error_previous)/dT

$$Servo\ motor\ (\theta) = control\ value \pm K\rho\left(1 + \frac{1}{T_i s} + T_d s\right) \times Error[P.I.D\ Parameter]$$

라는 식으로 정의 되었으며 아두이노에서 계산을 통해 서보를 제어하였다.



PID 계수는 실험적인 방법으로 튜닝 하였으며 Kp계수를 조정을 통해 차량이 일정한 진폭으로 주행할 때를 기준으로 Ki, Kd를 찾아서 적용하였다. 처음에는 정해진 코스만을 통해 PID 튜닝을 하였는데 우리 팀의 목표는 open system 의 도로에서의 완벽한 주행이 목표였다. 그런데 visual error를 구하는 영상처리 특성상 목표로 하는 각도 값이 제어기에 적용되는데 곡선에서는 한쪽방향의 크기로 치중된 error의 편차를 보이며 이것은 누적되어 곡선 부를 벗어났을 시 주행에 영향을 미치는 현상이 있었다. 고정된 PID값을 사용하는 것이 아닌 곡선부에서는 PD제어를 사용, 직선부에서는 PID 값을 사용하며 경계구간에서 누적된 error를 초기화 시키는 식으로 제어기를 설계 하였다. 또한 실제 상황과 관계없을 error수치 및 잡음을 처리하는 예외처리를 해주었으며 현재제어기 동작이전의 명령을 사용하여 외란에 적응하도록 하였다.



영상처리 시스템의 특성상 – 편중된 Error값이 누적되는 모습

3. Car Tracking

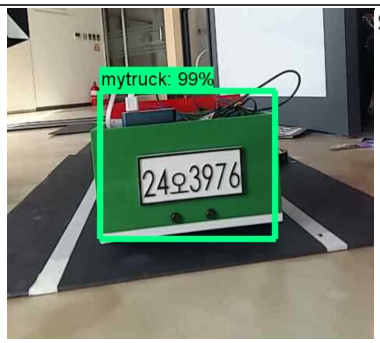
트레이닝을 위해 1200장의 선행차량의 이미지를 사진을 찍었다. 트레이닝에 사용된 이미지는 여러 환경에서 강인하게 인식을 하게하기 위해 실내, 실외, 다양한 각도, 다양한 상태에 대해서 사진을 찍어 트레이닝 하였다. annotation tool을 통해 선행 차량에 각각에 대한 ground truth를 COCO data set에 맞추어 지정해주었다. 그리고 몇 번의 시행착오를 통해 최적의 학습속도(learning rate)와 학습 속도 감소율(learning rate decay)를 구하여 12만 step의 학습을 시켰다. 이렇게 해서 객체를 찾아 추적을 할 때 객체가 탐지(object detection)되면 bound box가 화면상에 표시되는데 이렇게 해서 찾아진 객체가 완벽하게 선행차량이라는 보장이 없다. 따라서 여러 가지 조건(선행차량의 모습이 정사각형에 가까우므로 가로세로 비율 또는 세로가로 비율이 3보다 크거나 bound box를 아예 못 찾았거나 찾은 bound box가 화면에 꽉 차는 경우)에서 예외처리를 해주었다. 이 예외가 아닌 신뢰할 수 있는 bound box에 관하여 이 중앙 점과 화면자체의 하단 중앙 점사이의 각도를 구하여 객체를 추적하게 하였다.

객체와 후발 차량의 각도 계산식은 다음과 같다.

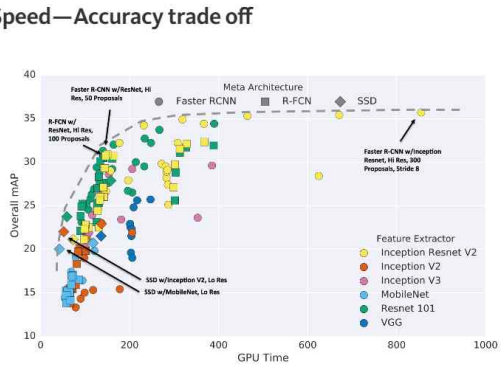
$$\theta = \arctan((x_{boundbox} - width/2)/(y_{boundbox} - height)*\pi/180)$$

$x_{boundbox}$ = bound box 에서 x좌표의 최소값

$y_{boundbox}$ = bound box 에서 y좌표의 최소값



트래킹을 하기 위해 딥러닝 기술을 이용하여 앞차를 인식한 사진.



평균 정확도 성능이 보장되고 실행속도는 가장 빠른 SSD mobilnet 모델.

4. Communication

군집 주행을 위해 후발 차량은 선행 차량의 주행 데이터가 필요하다. 따라서 두 차량은 하나의 통신망으로 연결되어야 한다. Hermes는 이를 MQTT 프로토콜을 활용하여 구현하고자 한다.

MQTT는 주로 IoT에 사용이 되는 통신 방식이며, 기존의 1대 1 통신이 아닌 1대 多 중계 방식이다. Client들은 Broker라는 Server를 통해 값을 주거나(Publish) 받을 수(Subscribe) 있다. 즉 선행 차량이 값을 Broker로 Publish하면 후발 차량들이 Subscribe함으로써 값을 받을 수 있다. MQTT는 저사양, 저전력 기기를 위한 메세징 프로토콜이므로 라즈베리파이나 아두이노와 같은 저사양 플랫폼에서도 원활하게 적용할 수 있다.

차량 간의 통신에서 큰 지연이 발생하는 것은 치명적이다. 시속 60km의 속력에서 1초의 지연시간은 16.7m의 지연거리를 발생시키기 때문이다. 따라서 차량의 플랫폼인 라즈베리파이 사이의 MQTT 통신의 속도를 측정하였다. 기준은 통신 값의 크기와 라우터와 플랫폼 사이의 거리이다.

Size	Latency(sec)	Distance Traveled with delay (Speed 60km/h = 16.7m/s)
3 bytes	0.036315	0.604 meters
6 bytes	0.046304	0.773 meters
9 bytes	0.052072	0.870 meters

- 통신 값의 크기에 따른 지연 시간

Distance	Latency(sec)	Distance Traveled with delay (Speed 60km/h = 16.7m/s)
3 meters	0.036315	0.604 meters
6 meters	0.043511	0.726 meters
9 meters	0.044538	0.743 meters

- 라우터와 플랫폼 사이의 거리에 따른 지연 시간

지연시간은 평균 0.05초 정도로 무시할 수 있을 정도의 지연 시간임을 측정할 수 있었다.

5. Cruise Control

후발 차량이 선행 차량을 따라갈 때 충돌하지 않으려면 일정 차간거리를 유지하며 주행해야 한다. 이를 위해 선행 차량과 간격이 가까우면 감속하고 멀면 가속하

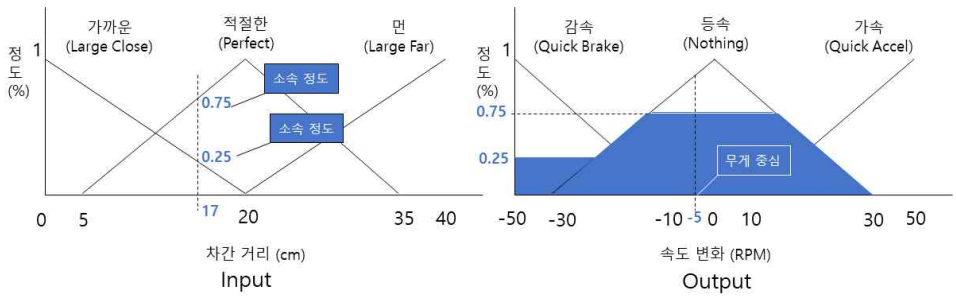
여 일정 간격을 유지하는 크루즈 컨트롤을 구현하였다.

후발 차량은 선행 차량과 같은 속력으로 주행하기 위해 선행 차량의 속력 값을 통신으로 받는다. 그리고 일정 차간거리를 유지하기 위해 초음파 센서를 이용하여 차간거리 값을 얻고 제어기에 입력하여 후발 차량의 가/감속 정도를 계산한다. 제어기는 Fuzzy Logic Controller를 사용하였고 후발 차량의 속력 RPM_follower은 다음과 같이 계산된다.

RPM_follower = RPM_leader + fuzzy_Output

즉 선행차량의 속력 RPM_leader에 Fuzzy 시스템의 계산 값 fuzzy_Output을 더하여 후발 차량의 속력 RPM_follower이 결정된다.

Fuzzy Logic Controller의 Fuzzy Set은 가까운(Large Close), 적절한(Perfect), 먼(Large Far) 3개의 Input set과 감속(Quick Brake), 등속(Nothing), 가속(Quick Accel) 3개의 Output set으로 이루어져있다. 각각의 Fuzzy Set은 Rule Base에 의해 대응된다. Fuzzy 시스템의 계산은 맘다니 형 추론을 따랐고 Output set의 통합된 영역의 무게 중심을 fuzzy_Output 값으로 설정했다.



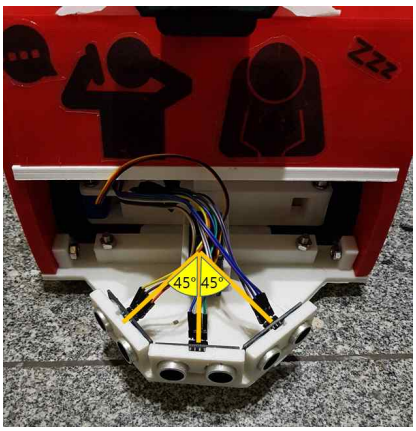
퍼지 Input, Output Set

IF 거리가 가까운(Close), Then 속도는 감속(Brake)
IF 거리가 적절한(Perfect), Then 속도는 등속(Nothing)
IF 거리가 먼(Far), Then 속도는 가속(Accel)

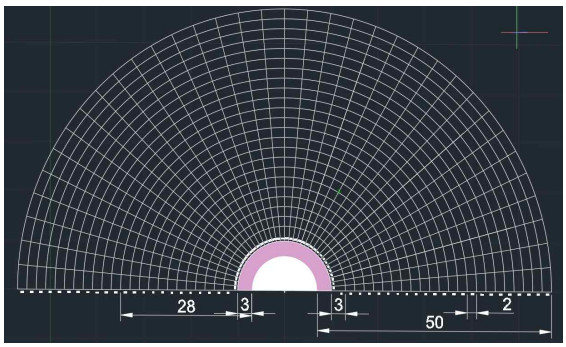
퍼지 Rule Base

6. Obstacle Avoidance

차량 주행 시에 갑작스런 장애물이 나타나면 회피할 수 있도록 초음파 센서를 장착하였다. 초음파 센서는 모두 3개로 각각 45도씩 차이가 발생한다. 초음파 센서가 탐지하는 영역은 행이 36개, 열이 23개인 이차원 배열로 구성되어 있다. 각도는 최대 180도를 5도씩 나누어 36개로, 거리는 최대 50cm에서 센서가 탐지하지 못하는 0~3cm는 제외하고 처음엔 3cm, 나머지는 2cm씩 하여 23개로 구역을 나누었다.

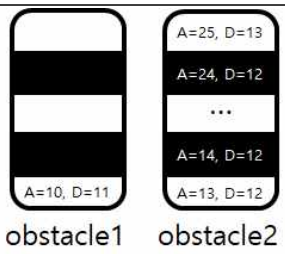


장착된 초음파센서



시각화된 2차원 배열

초음파 센서들은 서보모터에 의해 좌우로 45도씩 90도 움직이며 물체가 있으면 그때의 센서 방향과 그때의 거리 값에 해당되는 영역의 count값을 증가 시킨다. counting이 완료되면 count가 1번만된 곳은 정확도가 떨어지기 때문에 제외시킨 후, 연결된 곳은 하나의 물체로 labeling하게 된다. labeling된 배열을 탐색하며 label이 같은 곳

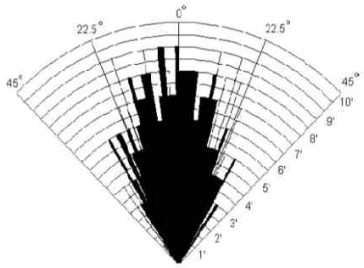


의 행과 열의 좌표를 스택에 각각 push 한다. 이때 앞뒤로 겹친 곳은 가까운 곳으로 선택한다. 초음파 센서의 감지각도는 거리에 따라 달라지지만 대략 좌우 15도씩 30도 정도이기 때문에 스택에서 pop을 하면서 처음 3개와 마지막 3개 (5도x3개=15도)를 제외하고 행과 열의 평균을 구하여 거리와 각도를 계산한다. 그리고 양쪽 끝의 거리와 각도로 제 2

스택

labeling 후 배열(축소된 모습)

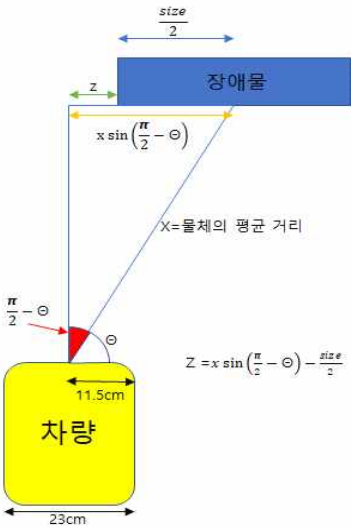
counting된 배열(축소된 모습)



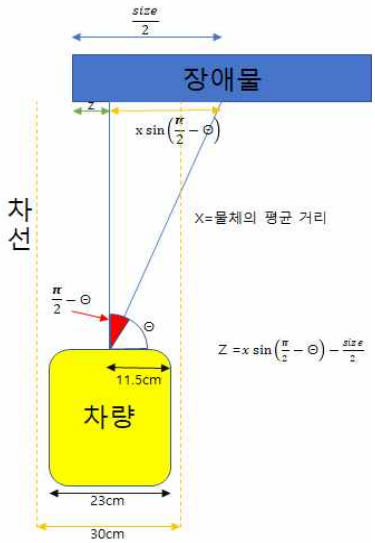
Practical test of performance,
Best in 30 degree angle

초음파센서의 감지 각도

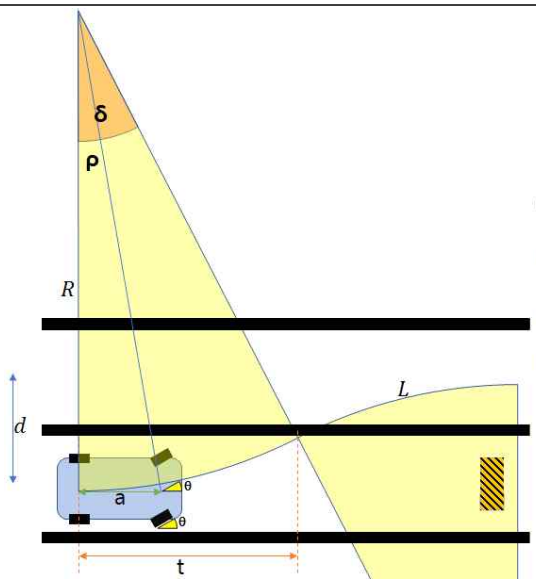
cosine법칙을 이용하여 물체의 크기를 구한다. 계산된 장애물의 크기, 각도, 거리를 바탕으로 z값을 구한다. 차체의 크기가 23cm 이기 때문에 z값이 11.5cm보다 크다면 그대로 다시 직진하며, 11.5cm보다 작으면 아두이노의 map()함수를 사용하여 z의 크기(0~11.5)를 15~30도로 mapping하여 회피 각도를 정한다. 만약 z값이 음수라면 장애물의 크기를 다른 차선을 넘지 않는다고 제한하고 –z(0~15)를 30~15도로 반비례 mapping하여 회피 각도를 정한다. 회피 시에는 차량의 조향각을 근사화하여 아래의 모델링처럼 계산하여 차량이 이동하게 했다.



장애물 회피 경우1



장애물 회피 경우2



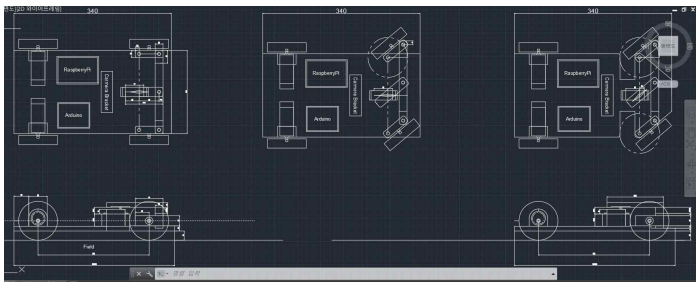
$$\cos\delta = \frac{R-\frac{d}{2}}{R} = 1-\frac{d\tan\theta}{2a}$$

$$R = \frac{a}{\tan\theta}\ (\rho \cong \theta)$$

$$L = R\times\delta = v\times t$$

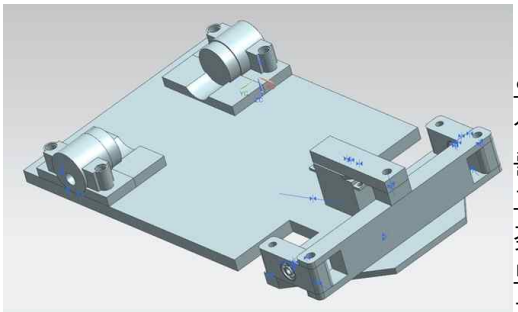
$$t = \frac{1}{v}\times R\times\delta = \frac{a}{v\tan\theta}\cos^{-1}\left(1-\frac{d\tan\theta}{2L}\right)$$

7. Car Design



CAD 프로그램을 통해 제작한 차량 초기 설계도면

초기 3D Print로 차체를 모두 제작해야 하므로 조향 시스템이 가장 중요했다. 공간 확보를 위해 Servo Motor와 링크를 직접 연결하여 제어하는 방식을 채택하였고 CAD 프로그램으로 초기 도면을 설계하였다. 조향각 θ에 따라 링크의 구멍과 구멍사이의 간격 $d=43mm$에 대해 계산하였다. 바퀴의 위치는 링크의 중앙에 위치하였고, Servo Motor와 직접 연결된 링크는 x축으로 $d(\cos\theta-1)$만큼 이동하며, y축으로 $dsin\theta$만큼 이동한다. 평균 실제 차량의 최대 조향각이 $30^\circ \leqq \theta \leqq 35^\circ$임을 감안하여, $\theta \leqq 35^\circ$로 설계하였고, 그에 맞추어 링크의 고정축을 중심으로 Reference Circle을 그려 차의 링크의 이동에 따른 바퀴 회전각을 계산하여 차를 설계하였다.

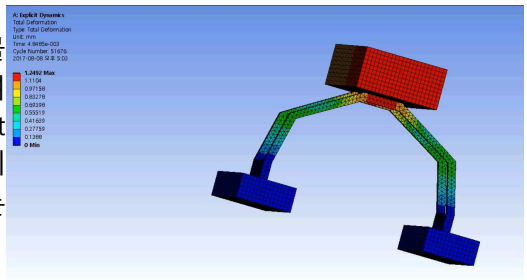


초기 설계도면을 토대로 NX로 3D 설계

초기 설계도면을 토대로 NX 프로그램으로 3D 설계 초기 CAD툴을 이용하여 설계한 그대로 NX프로그램으로 3D 설계를 하였다. 이 때, 3D Printing을 할 때 고려할 사항이 있었는데, 매끈한 표면을 갖기 위해 Supporter가 최소한으로 생기도록 하였다. Supporter가 발생하면 거친 표면을 갖게 되는데, 그렇게 되면 PIN에 있는 Hole의 Tolerance가 문제가 되어 PIN이 구멍에 헐거워지거나 맞지 않게 된다. 차체 대부분의 Hole에는 국제규격 $\varnothing=6mm$볼트를 사용하였고, 그에 맞는 구멍크기를 다수의 실험 끝에 $6.5mm$로 설계하였다. 3D Print는 교내 Finebot Print를 주로 사용하였는데, Nozzle Thickness가 $0.8mm$이며, Nozzle Temperature은 $200℃$, Bed Temperature은 $46℃$로 설정하였고, 일부 정밀성이 필요한 부품의 경우는 $0.4mm$ Thickness를 사용하였다. 또한 곡면이 아닌 외형의 Tolerance는 모두 실험을 통하여 측정하였고 대부분 $0.5mm$로 설계하였다. 현재 사용하려는 바퀴에 맞는 차축은 $4mm$원기둥 모양에 반원 모양으로 잘린 형상이다. 이를 위해 $4mm$철심을 구매 후 선반

가공을 통하여 축을 제작하였다.

모든 부품이 진동하거나 튀어나가는 것을 방지하기 위하여 각 부품에 대한 규격을 측정하여 틀을 제작하였다. 또한, 보조배터리 및 휴대폰 등의 Tolerance를 실험을 통하여 측정하고 진동 없이 탑재하도록 제작하였다. 차체 여러 부분의 Hole을 통해 전선들이 자유롭게 연결될 수 있도록 설계하였다.

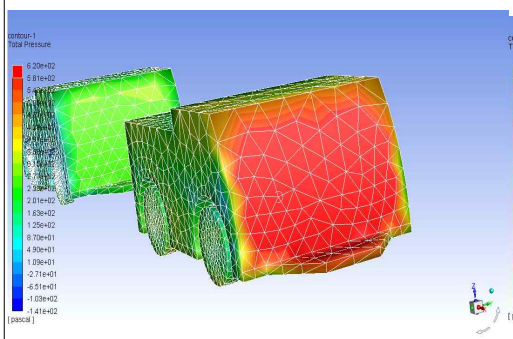


ANSYS를 이용한 Bumper Collision Test

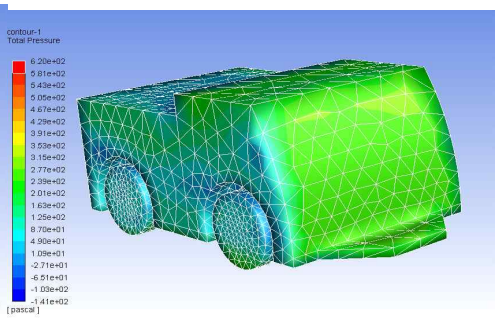
Sonar Sensor의 전면 돌출과 장애물 인식 장애로 인한 충돌을 고려하여 Bumper 제작 후 ANSYS explicit Dynamics를 이용하여 강철 벽에 실제 속도인 $0.34m/s$를 이용하여 충돌하는 Collision Test를 해석해 보았다.
3D Print에 사용된 PLA Filament 재질은 관련 논문에 의하여 $Density = 1.24g/cm^3$, $Elastic\ Modulus\,(E) = 3.5GPa$, $Shear\ Modulus = 1.287GPa$와 같은 물리적 특성을 이용하였다. 해석 결과 충돌 시 Bumper 전면부에 $1.24mm$의 Distortion이 발생하였고, 극한응력 $\sigma_y = 70MPa$을 이용하여 파괴 및 소성 변형 여부를 계산하였다. Hook's law를 이용하여 $\epsilon = 1.24/70 = 0.0177$을 대입하여 $\sigma = 3500^{*}0.0177 = 61.95MPa$ 로서 탄성변형으로 확인하였고, 안전하다고 판단하였다.

ANSYS를 이용한 Platooning의 유동학적 이득 계산

Truck 모형으로 Truck Platooning의 유체역학적 이득으로 어느 정도의 이득을 볼 수 있을지 ANSYS Tool의 Fluent를 사용하여 해석해 보았다. 현재 차량이 Prototype이며, 차량의 크기 및 속도가 낮지만, 형상과 차간거리만으로 적정 속도의 유체를 차량에 흐르게 하였다. 유체 모델은 일반적으로 사용되는 k-epsilon 모델을 사용하였다. 해석 방식은 두 차량을 차간거리로 설정한 후, Enclosure 및 Boolean으로 Control Volume과 해석 Target을 설정해주고, Inlet과 Outlet을 설정해 주었다.

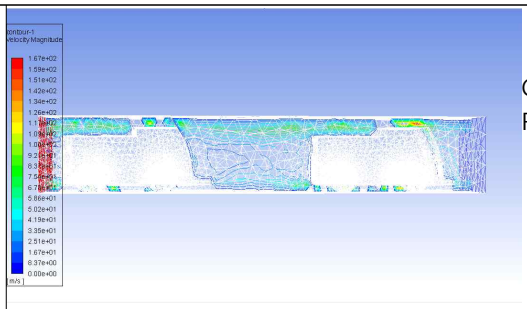
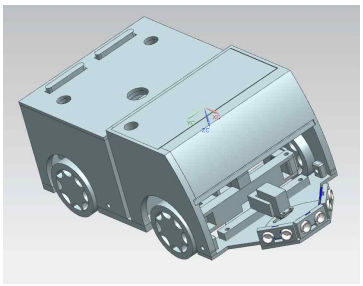
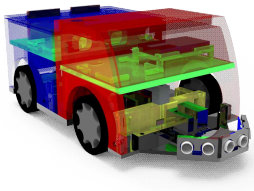
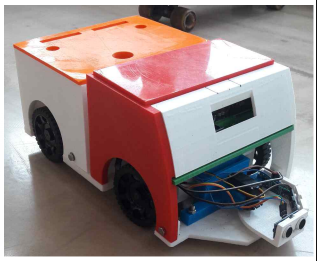


ANSYS Fluent로 계산한 앞차에 가해지는 유체의 높은 압력



앞차에 비해 큰 폭으로 감소한 뒷차의 유체 압력

첫차의 앞 표면에서의 평균 유동 압력은 Gauge Pressure로 $620Pa$, 뒷 차는 $337Pa$ 로 나타났고, 이는 앞차의 유체에 의한 압력보다 46% 감소한 효과를 볼 수 있다.

	 - ANSYS Fluent의 유동 해석, Velocity Contours로 확인해 본 눈에 띄게 줄어든 Rake Region.
5. 팀원 역할 내용	영상처리팀, 제어팀, 설계팀, 통신팀, 딥러닝팀 5개의 팀을 구성했고 업무를 분담하여 진행했다. 성현기 (팀장, 제어) : 전반적인 시스템 관리 민경규 (영상처리) : 차선인식 소프트웨어 설계 김진현, 배성현, 문성주 (회로, 제어) : 제어시스템 설계, 회로 구성, Arduino 프로그래밍 한형진 (통신, 프로그래밍) : MQTT Server-Client 통신, Python 프로그래밍 박대희, 이창혁, 김시현 (차체 설계, 3D Printing) : 차체 설계, 3D Printing 이힘찬, 최정은, 장우진, 박성준, 김광현, 하선우, 임정혁 (딥러닝) : 딥러닝 스터디 및 객체 추적 소프트웨어 설계
6. 기타사항	
7. 설계 작품 이미지	 NX 프로그램으로 설계한 최종 Assembly  Rendering한 모델 구상도  3D Printer로 출력하여 제작한 차체
8. 참고문헌	1. Torres, J., Cotelo, J., Karl, J., and Gordon, A. P. (2015) Multi-Objective Optimization of a Fused Deposition Manufactured Polylactic Acid under Torsion 2. Huang, Jonathan, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, et al. 2016. "Speed/accuracy Trade-Offs for Modern Convolutional Object Detectors." arXiv [cs.CV]. arXiv. 3. MQTT contributors. "MQTT." Mosquitto Source, W3 Consortium The W3 Consortium, 28 Aug. 2017. Web. Sep. 2017 4 김상겸 외 2명, *차선인식을 위한 무인자동차의 차량제어 및 모델링에 관한 연구*, 한국자동차공학회논문집, 한국자동차공학회, 2003 5. 안대원 외 2명, *사전정보를 활용한 실시간 차선검출의 알고리즘*, 한국멀티미디어학회 학술발표논문집, 한국멀티미디어학회, 2010 6. Joshu´e P´erez Rastelli외 3명, *Cascade Architecture for Lateral Control in Autonomous Vehicles*, IEEE Transactions on Intelligent Transportation Systems, IEEE, 2011.

2017. 09. 04

팀 장: 성현기 (인)