Lidar조 종합설계1 발표 시작하겠습니다

발표는 종합설계 최종 설계 목표와 그 목표를 설정하게 된 계기를 먼저 말씀드리고 종합설계1을 진행한 결과 시연동영상을 먼저 짧게 보여드린 다음 각 단계별로 어떠한 작업을 진행하였는지 설명하는 순서로 진행하도록 하겠습니다.

저희의 종합설계 최종목표는 lidar와 camera의 calibration을 통한 ascc자율주행 시스템 구현입니다.

Ascc adaptive smart cruise control이란 카메라와 레이더를 이용해 앞차를 인식하고 일정한 간격을 유지하여 속도를 조절하며 stop&go가 가능한 자율주행 모드입니다. 현재 가장 상용화된 기술은 2단계로 아직까진 운전자의 통제가 필요한 수준의 기술이지만 최근 g90 등에 탑재될 예정인 교통신호와 도로 흐름을 스스로 인식하는 3단계 기술이 상용화될 예정이며 이 3단계 자율주행 차량이 등장하면서 기존 구형 레이더와 카메라 시스템의 성능 개선 및 lidar 센서 관련 기술이 급부상하고 있습니다.

그렇다면 레이더와 라이다의 차이점은 무엇이며 lidar를 이용한 자율주행 모드가 갖는 이점이 무엇일지에 초점이 쏠릴텐데요, 레이더는 전자기파를 사용해 객체의 거리와 속도를 감지하여 장거리에서도 객체를 감지할 수 있지만 작은 객체나 복잡한 형태의 객체, 비금속 객체를 상세히 감지하는 데는 한계가 있다는 단점이 있습니다. 반면에 라이다는 빛을 보내 반사된 정보를 분석하여 차량 주변 환경의 3d 이미지를 생성하여 물체의 거리 및 형상을 정밀하게 인식할 수 있으며 이를 통한 3d mapping 이 가능하기 때문에 레이더보다 더 정밀하고 상세하게 주변 환경을 파악할 수 있습니다. 이러한 측면에서 현재 자율주행 기술을 리드하는 기업들에게 라이다는 자율주행을 상징하는 센서로 인식되고 있습니다.

주제를 선정하게 된 배경을 간략히 말씀드리면, 우선 저희는 교수님이 올려주신 lidar & camera calibration 예제를 진행하였습니다. 예제를 진행하면서 calibration에 대한 전반적인 이해를 하고 주제를 색깔별로 3d mapping 한 후 보행자, 차량, 장애물 등의 우선순위를 정하는 방향전환 시스템으로 하면 어떨까 했으나 그 주제는 기존에 있는 시스템과 크게 차별점을 갖지 않았으며 종설 전체를 진행할 만한 주제가 아니라고 판단하여 lidar를 이용한 자율주행 모드를 설계해 보면 어떨까 생각하게 되었습니다. 이에 따라 자율주행에 관해 조사를 좀 해보니 기존의 radar를 이용한 자율주행에서 더 나아가 lidar를 이용한 자율주행 모드 기술이 급부상하고 있다는 뉴스를 접하게 되어 Targetcar만을 추적해 따라가는 반자율주행 크루즈 control system 설계를 진행하게 되었습니다. 하지만 targetcar만은 따라가는 시스템이 과연 필요할까하는 범용적 측면에서의 한계를 느껴 지금의 최종주제 lidar & camera calibration을 통한 ascc 자율주행 시스템 구현이라는 목표를 설정하게 되었습니다.

저희가 최종적으로 구현하고자 하는 결과물은 lidar와 camera의 calibration 센서퓨징 기술을 이용한 자율주행 시스템입니다. 앞서 설명드린 바와 같이 기본적으로 앞차를 targetcar로 인식해 따라가는 시스템이며 앞차가 바뀔때마다 targetcar를 변경과 동시에 주변 환경을 파악하여 자율주행 하도록 시뮬레이션 상에 구현하고, 이를 실제 트랙과 rc카로 구현 및 동작 확인을 할 계획입니다.

전반적인 종합설계 진행 절차입니다. 이번 학기 종합설계1에서는 설계 목표 및 계획 수립, yolo를 통한 targetcar 객체탐지, lidar & camera calibration, simulation 환경 구축 및 세부상황 구현의 초기단계까지 진행하였고 다음 학기 종합설계2에서는 simulation 상에 추가적이 세부상황 구현 및 차량 이외의 객체 탐지 구현과 실제 구현하기 위한 제어시스템 설계 및 실제 트랙과 rc카를 통한 시스템 구현 확인을 하려고 합니다.

각 단계별 세부진행 과정을 설명 드리기 전에 우선 짧게 종합설계1을 진행한 결과를 보여드리겠습니다. 왼쪽 동영상을 보시면 yolo detection과정을 거쳐 targetcar를 탐지하는 camera정보와 lidar의 3d정보를 sensor fusing한 칼리브레이션 결과를 바탕으로 상황을 인지하여 속도를 조절하는 것을 볼 수 있습니다. 왼쪽 아래 속도변경 과정을 보시면 앞의 차량과 가까워지면 속도가 느려지는 등의 과정을 확인할 수 있습니다. 오른쪽 동영상은 왼쪽동영상에서의 정보를 바탕으로 각 상황별로 저희가 입력한 명령 프롬프트를 통해 시뮬레이션을 진행하는 과정입니다. 그럼 이제 각 단계별 세부진행상황과 앞서 보여드린 이 결과가 나오기까지의 과정을 설명 드리겠습니다.

본격적인 설계 진행 전 lidar를 이용한 자율주행에 참고할 만한 여러 sensor fusing 예제를 시도해보았습니다. 하지만 기존 예제 중에는 radar와 camera를 이용한 예제만 있어 lidar & camera calibration을 이용한 자율주행 모드 설계에 대한 아이디어를 얻지 못했습니다. 그래도 왼쪽의 예제와 같이 radar sensor fusion을 이용해 충돌을 방지하는 과정을 보며 여러 상황에 대처하는 시스템을 어떻게 설계할지 전체적인 흐름을 정립할 수 있었습니다.

우선 YOLO를 통한 TARGETCAR 탐지부터 진행하였습니다. 왼쪽의 사진처럼 색 이진화를 통해 원하는 차량만을 탐지하는 Labelling algorithm을 만들어 이 차량의 200개의 사진 샘플에 자동으로 bounding box가 쳐지는 프로그램을 설계하였습니다. 그러면 오른쪽 사진처럼 탐지된 차량에 네모박스가 쳐지는데요 이 labelling된 데이터를 matlab상의 YOLO에 학습시킨 후 웹캠으로부터 실시간 프레임을 받아 그 학습시킨 데이터를 바탕으로 targetcar를 탐지하고 bounding box가 쳐지는 것입니다.

이 과정에서 애로사항이 발생하였는데, 왼쪽의 사진처럼 targetcar가 제대로 detection되지 않는 문제였습니다. 보시면 노란색 bounding box가 targetcar를 제대로 탐지하지 못한걸 볼 수 있습니다. 저희는 이 원인을 처음엔 50개의 샘플 사진을 학습시켜 진행하였기 때문에 훈련된 데이터양이 부족하여 제대로 탐지하지 못한다고 판단하였습니다. 그래서 200개의 샘플 사진을 추가로 학습시켰지만 여전히 detection이 향상되지 않았습니다. 이에 대한 문제점이 무엇인지 파악하던 중 처음 사용했던 4k 웹캠과 매트랩 사이의 연동 문제였다는 것을 알 수 있었습니다. 4k 웹캠의 높은 화질 프레임을 매트랩에서는 받아들일 수 없어서 생긴 문제였습니다. 그래서 웹캠을 변경하여 실행하였고 이 문제를 해결할 수 있었습니다.

Matlab Lidar & Camera Calibration 예제

. Calibration이란 다중 센서 여기서는 2D인 카메라와 3D인 lidar를 동일한 좌표계로 변환하는 방식입니다. 체커보드의 평면을 기반으로 calibration을 진행한 Matlab 예제입니다 해당 예제는 체커보드만을 디텍딩하기 위한 예제이기 때문에, 자동차의 3D에 대한 특징점을 미리 간주하는 함수가 필요하고, 그게 객체마다 다르기에, 다른 calibration 방식이 필요하여 github에서 region of interest방식을 사용한 calibration 예제를 확인할 수 있었습니다.

Github Lidar & Camera calibration 예제2

ROI는 Region of Interest의 약자로 관심영역을 제외한 나머지 부분을 소거하는 방식을 말합니다. 이를통해 라이다의 연산을 크게 줄일 수 있습니다. 이후 줄인 ldiar 데이터를 Rotation과 Translation 행렬을 사용하여 좌표축 calibration을 진행합니다. Rotation 행렬은 카메라의 2D 좌표축과 lidar의 3D 좌표축을 일치시키는 역할을 하고, Translation 행렬은, 라이다와 카메라의 현실 세계에서의 거리를 보정해주기위한 평행이동이라고 보시면 됩니다.

기존의 예제와 어떻게 다른가?

저희는 YOLO의 인식을 선행하여, 물체에 대한 인지를 카메라로 먼저 수행한 뒤, 그 물체에 대한 depth 값을 Calibration으로 뽑아내는 방식을 채택했습니다. 기존의 예제에 인지부분을 추가한 방식으로 볼 수 있습니다. 해당 아이디어를 통해, 다른 차량의 실질적인 위치를 더 자세히 알 수 있게 되었습니다. Stereo depth를 통한 카메라 거리 추정 방식보다 정밀한 라이다 데이터를 사용하였으므로, 위치에 대한 거리가 매우 중요한 차량상황에 알맞은 방식임을 알 수 있었습니다.

Lidar & Camera calibration <애로사항>

Calibration할 시에는 역시나 parameter가 중요한데, Rotation Matrix의 적당한 값을 찾기 위해 200회 정도의 실험을 통해 정확한 Rotation matrix를 찾을수 있었고, 이는 이론과는 많이 다른 모습임을 확인했습니다 항공대 드론센터에서 차량 한대를 가지고 수행한 모습입니다. 3D 값을 2D로 변환하였기에 입체적으로 라이다의 파란점이 보이는 모습입니다.

Lidar & camera calibration <애로사항2>

Calibration 정확도 문제가 발생하였습니다. 카메라 각도가 조금만 어긋나도 거리가 10m 이상이 되면 오차범위가 커지기 때문에, 이를 해결하기 위해 카메라와 라이다의 위치 및 각도를 정확하게 맞추어 재-실험하였습니다. 둘째로, bounding box 이내의 값들 중 원하는 군집만 특징점으로 간주하여서 정확도를 올렸습니다. Clustering 과정은 다음 장에서 설명드리겠습니다. 여기 사진에 보시면 가까우면 하얀색 점으로 멀면 파란색 점으로 시각적으로 다르게 표시되게끔 노력했습니다.

특징점 간주 clustering algorithm

가장 많은 값들이 들어있는 구간 내 depth 값을 특징점으로 정의하고 나머지는 소거하는 방식을 채택하였습니다. 이렇게 만들어진 특징점은 거리를 정확하게 인지하기에, depth 및 픽셀 값 별로 명령을 줄 수 있게 되었습니다. 영상 보시겠습니다. (일시재생 생각해야됌)

앞의 차량이 급정거하는 경우 fast speed에서 medium speed slow speed가 되고 이후에 너무 가까워지면 stop이 되는 모습을 보실 수 있습니다. 저희 조원이 나와있는 이 영상은 다른 차량이 끼어드는 상황을 가정한 것으로 yolo가 인식되지 않은 상황에서 2m 내의 거리에 있는 경우에 하얀색 점으로 표현하고 즉각적으로 멈추는 시나리오입니다. 픽셀값이 threshold를 우측으로 벗어난 경우 우회전한다고 가정하여 명령을 출력하는 영상입니다. 멈춰있던 차량이 출발하는 모습으로 slow speed에서 fast speed 까지 속도를 올리는 모습입니다.

Simulation 환경 구축

여러 가지 시나리오를 시도 및 수정하며 향상된 ASCC 자율주행 모드를 구현하기 위해 시뮬레이션이 필요합니다. 시뮬레이션의 상황 설정은 다음과 같습니다. 기본적인 4가지 상황설정, 장애물 및 도로상황 설정, 미세상황 제어등이 있습니다.

Simulation 환경 구축

시간 관계상 급정거 시나리오만을 영상으로 보여드리겠습니다. Driving scenario app을 통하여 제작하였고, 명령 프롬프트 별로 시뮬레이션을 진행하기위해서 simulink를 사용하려하였으나, connection오류가 발생하여 안타깝게도 다른 simulation 예제을 통해서 시도하였습니다.

Matlab Driving Scenario 예제

다른 시뮬레이션 예제이고 타원의 Road를 자동차가 계속해서 도는 서킷 예제입니다. 해당 예제에는 속도을 일정하게 제공하는 방식입니다. 이를 변경하여 저희는 시뮬레이션 토대를 구성하였습니다.

Matlab Driving scenario

저희가 만든 시뮬레이션을 설명 드리겠습니다. 영상 보시겠습니다. 해당 영상은 속도가 명령 프롬프트를 따라 변경되어가며 차선을 변경하는 예시입니다.직선형태로 도로를 바꾸었고, waypoints 마다의 속도를 명령 프롬프트에 따라 변경하도록 설계하였습니다. Waypoints 란 차량의 경로를 뜻합니다. Slowspeed로 입력이 발생할때, 속도를 느리게 주고, fastspeed로 입력이 발생할 때 속도를 빠르게 제공하였습니다. Trajectory를 변경하여 차선이동을 구현하였지만, 아직 명령 프롬프트에 따라 Way points를 동적으로 변경하여 실행하는 방식을 구현하지는 못하여 실시간으로 시뮬레이션 제공은 어렵습니다. 하지만 더욱 발전시킬 계획입니다..

종합설계2 진행계획

세부적인 simulation 상황 설정을 위해 현재 Target Car만을 detection하는 상황에서 다른 객체로 탐지 개수를 늘리고, 이를 시뮬레이션 환경에 구축하는 방식으로 설계하려합니다.

제어시스템 설계

Vehicle에 대한 wheel각도 계산 및, 물리학적으로 동작할 수 있는지를 판정하는 smoothing trajectory 등을 이해하고 제어하는 시스템을 설계하려고합니다. 시뮬레이션에 그치지 않고, 이를 실제 트렉으로 구현하고, RC 카를 이용하여 다양한 시나리오 상황에 있어서도 동작하는 ASCC 시스템을 구현하고자 합니다.

감사합니다