



## 저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 [이용허락규약\(Legal Code\)](#)을 이해하기 쉽게 요약한 것입니다.

[Disclaimer](#)

석사학위논문

거리측정을 위한 CNN 이미지 분석과 Lidar Point  
Cloud 기반 사물 인식 및 시각화

Object detection and visualization using CNN  
image analysis and Lidar point cloud for  
distance measurement

김 혜 진

한양대학교 공학대학원

2018 년 2 월

석사학위논문

거리측정을 위한 CNN 이미지 분석과 Lidar Point  
Cloud 기반 사물 인식 및 시각화

Object detection and visualization using CNN image  
analysis and Lidar point cloud for distance  
measurement

지도교수 조 인 휘

이 논문을 공학 석사학위논문으로 제출합니다.

2018 년 2 월

한양대학교 공학대학원

컴퓨터공학 전공

김 혜 진

이 논문을 김 혜 진의 석사학위 논문으로 인준함

2018 년 2 월

심사위원장 이 병 호



심사위원 조 인 휘



심사위원 손 규 식



한양대학교 공학대학원

## 국문요지

자율주행 보조 기능 목적의 ADAS 개발을 위해서는 다양한 센서로부터 취득한 대용량 데이터의 통합 분석한 결과가 필요하다. 2012년 이후로 고성능의 GPU를 이용한 대용량 데이터 분석이 가능해지면서 자동차 산업에도 고성능 GPU 시스템 기반 센서 데이터 분석 연구가 도입되었다.

본 연구에서는 GPU 기반 자율주행 개발 플랫폼을 이용하여 카메라 영상과 Lidar 데이터를 수집한 후 차량과 주변 사물과의 거리를 주행 중 실시간으로 표시하기 위한 연구 개발 소프트웨어를 개발하도록 하였다.

센서 분석 테스트는 6개 각각의 카메라와 4개 Lidar 를 이용하여 진행되었다. 카메라 데이터는 CNN 기반 사물 인지 모델인 DriveNet 에 의해 감지된 사물과 거리 분석을 진행하고, 4개 Lidar 의 3d point cloud data 는 방향에 맞게 3차원 transformation & rotation 과정으로 통합한 후 PCL 라이브러리로 사물을 인식하여 거리 데이터를 취득하였다.

테스트 결과는 실제 차량 간의 통신으로 거리를 계산하는 장비인 D-GPS 의 데이터를 기준으로 카메라 데이터 기반 거리와 라이다 데이터 기반 거리를 비교하여 오차율을 산정하였다.

카메라 데이터의 경우 전방 카메라에서 감지된 대상 차량은 1채널당 24Fps로 감지되며, Calibration 결과 이미지 데이터 분석 기반 대상과의 거리의 오차율은 주행 상태에서  $\pm 0.8$  m 이다.

Lidar 데이터의 경우 전방 Lidar 2개의 Integration 결과 데이터를 PCL 라이브러리 함수의 FPFH API를 이용하여 사물을 감지한 후 거리 데이터를 취득한 결과 거리 오차율이  $\pm 0.5$  m 로 나온다.

# 차 례

국문요지.....	i
차 례.....	ii
그림 차례.....	iv
테이블 차례.....	v
제1장 서론.....	1
1.1. 연구배경.....	1
1.2. 연구목적.....	3
1.3. 제안방식.....	5
제2장 관련연구.....	6
2.1. CNN 기반 보행자 감지.....	6
2.2. 결정레벨 Fusion 에 의한 사물 감지와 분류.....	8
2.3. FCN 을 이용한 3D 라이더의 차량 감지.....	10
2.4. 카메라와 Lidar Fusing 을 이용한 사물 인지와 트래킹.....	12
제3장 설계 및 구현.....	15
3.1. 개발환경.....	15
3.2. Flow Chart.....	17
3.3. 구현.....	18
3.3.1 CNN 기반 카메라 사물 인지 구현.....	19
3.3.2 OpenCV 기반 거리 계산 구현.....	25
3.3.3 Lidar 데이터 Point Cloud 분석.....	30
제4장 성능평가.....	33
4.1. 성능 평가.....	34

제5장 결론 및 향후 연구.....	3 8
참고문헌.....	3 9
ABSTRACT.....	4 1
감사의 글.....	4 3



## 그림 차례

Figure 1 GPU 기반 다중 센서 인식 테스트 시스템 구축.....	3
Figure 2 전통적인 CNN 구조 [2].....	7
Figure 3 물체 감지 및 의사 결정 융합 방법[3].....	8
Figure 4 (a)혼잡 도로에서 감지 결과 (b)원거리 차량의 감지 결과 [5] 1	0
Figure 5 LIDAR 3d point cloud 와 카메라 칼라 영상[8].....	1 2
Figure 6 카메라와 Lidar 의 입력 값에 대한 처리 프레임워크[8].....	1 3
Figure 7 센서 기반 ADAS 시험 차량.....	1 5
Figure 8 GPU 기반 다중 센서 데이터 수집 및 분석개발 플랫폼.....	1 6
Figure 9 센서 SW 와 GUI SW 의 플로우 차트.....	1 7
Figure 10 Classification + Localization과 Object detection.....	2 0
Figure 11 사물 감지를 위한 R-CNN.....	2 2
Figure 12 DetectNet 입력 데이터 표현.....	2 3
Figure 13 DetectNet Training & Validation.....	2 4
Figure 14 주행차량에서 6개 카메라로 DriveNet 으로 사물 인지한 화면	2 4
Figure 15 카메라 intrinsic / extrinsic 캘리브레이션.....	2 5
Figure 16 카메라 영상의 2차원 평면에 투사 변환 모델링.....	2 6
Figure 17 영상좌표계, 카메라 좌표계, 월드 좌표계.....	2 7
Figure 18 카메라로부터의 가상의 정규 이미지 평면에서의 좌표.....	2 8
Figure 19 차량에 부착된 4 방향 라이더의 배치도.....	3 0
Figure 20 행렬을 이용한 3D 데이터의 회전(상) 이동(하).....	3 1
Figure 21 4 방향 라이더 데이터의 회전 이동 전 후 Point Cloud 표시	3 1
Figure 22 차량 주행 중 카메라와 라이더 데이터 분석.....	3 4
Figure 23 DGPS, 카메라, Lidar에서 취득한 거리의 2D Map과 시간그래프	3



## 테이블 차례

Table 1 PC 개발 환경 테이블.....	1 6
Table 2 Region proposal 알고리즘 간의 성능 비교.....	2 1
Table 3 DGPS, 카메라, Lidar에서 취득하는 대상차량과의 거리 테이블..	3 6
Table 4 DGPS, 카메라, Lidar에서 취득하는 대상차량과의 거리 오차율..	3 6



# 제 1 장 서론

## 1.1. 연구배경

자율주행차는 자동차가 주행 중 환경을 센서 등으로 인식해 상황을 감지하고 위험에 대하여 판단하거나, 주행경로를 계획하여 자율적으로 안전하게 운행할 수 있는 시스템을 갖추어 운전자의 주행에 대한 관여를 최소화 시킬 수 있는 자동차이다(Autonomous Vehicle).

자율주행 보조 기능 목적의 ADAS 개발을 위해서는 다양한 센서로부터 취득한 대용량 데이터의 통합 분석한 결과가 필요하다. ADAS는 운전자가 운전을 위한 주의와 노력을 최소화하도록 도와 줌으로써 안전한 운전에 도움을 줄 수 있는 시스템이다(Advanced Driver Assistance System). 최종 자율주행 구현에 필요한 SW 기술로서 ADAS의 단계별 기능은 1단계 편의(Convenience)에서는 후방 카메라와 같은 단순 기능과 주행 중 정해진 속도를 유지시키는 크루즈 컨트롤이 있다. 2단계 보조(Assistance) 기능은 차선 이탈 방지나 전방 충돌 경고와 같이 차량 주행 방향의 차선이나 선행 차량과의 거리를 감지하며 일정 조건 에서 경고음, 비상 제동 등의 알림 기능을 가진다.

하지만 3단계 대리(Proxy) 의 경우 하위 단계의 ADAS와 더불어 차량 전체를 제어할 수 있는 폐쇄형 컴퓨팅 시스템이 필요하다. 이러한 시스템에서는 주차 조향, 차선 유지, 비상 제동과 같은 필요한 행동을 자동차가 스스로 수행할 수 있어야 하며, 여기서 핵심기술은 운전자의 판단 능력과 같거나 보다 나은 판단이 가능한 인공지능기술인 알고리즘이라 할 수 있다

ADAS 시스템을 이용한 물체 감지는 ADAS 의 주요 연구 주제이다. 그러나 보행자 감지 장치를 일반 차량에 설치하려면 비용을 줄이고 높은 정확성을 보장해야 한다. 많은 접근법이 개발되었지만 vision-based 보행자 감지 방

법이 이러한 요구 사항에 가장 적합하다고 보여진다. 우리는 다양한 분야의 연구에서 높은 정확도를 보여주고 있는 CNN (Convolutional Neural Networks) 기반으로 구현하고자 한다.

2012년 이후로 고성능의 GPU를 이용한 대용량 데이터 분석이 가능해지면서 자동차 산업에도 고성능 GPU 시스템 기반 센서 데이터 분석 연구가 도입되었다.

그래픽 카드로 유명한 GPU 는 그래픽 데이터를 고속으로 처리하는데 특화되어 있다. 이러한 GPU가 인공지능 개발을 위해 주목받게 된 이유는 연산을 병렬로 동시에 계산함으로 컴퓨터의 전체 연산속도를 획기적으로 높일 수 있기 때문이다. CPU의 경우 연산을 순차적으로 처리하기 때문에 연산속도를 높이기 위해서 CPU core 의 수를 multi-core 로 고도의 기술로 개발하여도 성능 향상 효과는 기대에 미치지 못하고 발열 문제 등을 고민해야 한다. 그러나 GPU의 경우에는 단순연산용 core를 다수 장착한 병렬처리 구조를 통하여 게임과 동영상 등과 같이 고도의 그래픽 성능을 필요로 하는 프로그램을 효율적으로 구동할 수 있으며, 같은 원리로 절대 다수의 연산이 절대적으로 필요한 최신 인공지능 기술에 최적이라고 볼 수 있다.

오늘날 딥러닝(deep learning)은 인공지능 분야의 핵심 파트를 담당하고 있는데 빠른 속도로 다수의 연산을 수행해야 하기 때문에 특히 GPU의 중요성이 절대적으로 높다고 할 수 있겠다.

본 연구에서는 Nvidia drivePX2 를 이용하여 카메라 영상과 Lidar 데이터를 수집한 후 차량과 주변 사물과의 거리를 주행 중 실시간으로 표시하기 위한 연구 개발 소프트웨어를 개발하도록 한다.

본 연구에서는 CNN 을 이용하여 영상 이미지에서 분석된 거리와 라이다에서 분석된 차량의 거리 정보를 실제 거리와 비교하여 2차원 맵과 타임 그래프를 실시간으로 표시하여, 주행 중 다중 센서를 이용한 차량 주변의 사물 인식 및 거리 측정 및 보정과 활용에 대한 연구를 할 수 있도록 한다.

## 1.2. 연구목적

센서 기반 지능형 차량 시스템에 의해 정확한 물체를 감지 및 분류를 위해 다중 센서 정보를 표현하고 인식하는 과정이 필요하다. 본 연구에서는 Nvidia drive PX2 를 이용하여 영상과 Lidar 데이터를 수집한 후 CNN 을 이용하여 차량을 인식한다. 인식된 차량을 정확한 거리를 실시간으로 시각화하여 보정한 영상 이미지에서 계산한 거리와 라이다에서 분석된 차량의 거리 정보를 매칭시켜서 연구할 수 있는 테스트 환경을 구축하고자 한다.

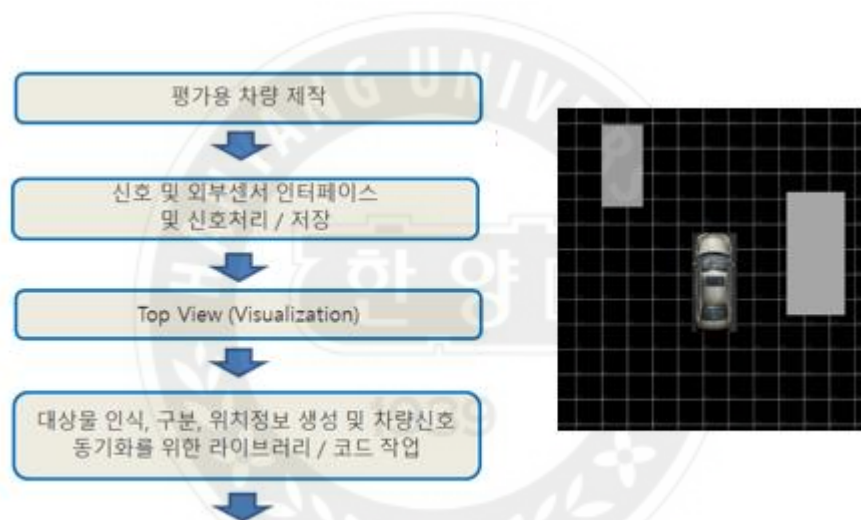


Figure 1 GPU 기반 다중 센서 인식 테스트 시스템 구축

운전 보조 시스템에서, 물체 감지는 일반적으로 LIDAR 또는 차내 카메라가 기록한 단일 또는 다중 프레임을 사용하는 vision-based approaches 을 사용하여 수행된다.

LIDAR(Light Detection and Ranging) 는 물체가 반사하는 빛으로부터 three dimensional point cloud 을 포착한다. 예를 들어 Google 자동 운전 시스템은 고품질의 깊이 정보를 기록하고 연구 분야에 적합하기 때문에

LIDAR를 사용한다. 그러나 LIDAR는 일반 차량의 경우 매우 비싼 기술이다.

차량용 카메라를 사용하는 방법에는 두 가지가 있다. 하나의 프레임을 기반으로 한 방법과 여러 프레임을 기반으로 하는 방법인데, 단일 프레임 방법은 실시간 탐지를 가능하게 하지만 다양한 상황에서 탐지 정확도를 향상시켜야 한다. 반면, 다중 프레임 기법은 모션 및 컨텍스트 정보를 사용하여 높은 탐지 정확도를 달성한다.

우리는 GPU 기반 시스템에서 이러한 다중 센서 기반 CNN 연구를 할 수 있도록 환경을 구성하는 것이 목적이다. 차량 센서 데이터를 통합 분석하기 위해 CNN 을 이용하여 영상 이미지에서 분석된 거리와 라이다에서 분석된 차량의 거리 정보를 실제 거리와 비교하여 2차원 맵과 타임 그래프를 실시간으로 표시하여, 주행 중 다중 센서를 이용한 차량 주변의 사물 인식 및 거리 측정 및 보정과 활용에 대한 연구를 할 수 있도록 하였다. 이 연구 결과물을 기반으로 자율주행차 개발을 위한 데이터 구축 및 센서 입력에 대한 차량 제어 테스트가 가능할 것이다.

### 1.3. 제안방식

본 연구에서는 다중 센서가 감지한 사물과의 거리의 정확도를 높이기 위해 D-GPS 라는 실제 두개 차량 간의 정확한 거리를 측정해주는 테스트 장비와 차량을 이용해서 거리 데이터를 얻으면서 카메라와 라이다에서 감지된 사물 간과의 거리를 계산하여 2D Map 와 그래프를 통해서 시각화 하여 주행 중 실시간으로 데이터의 추이를 살펴보고, 이를 통하여 보정과정을 거쳐 정확도를 높일 수 있도록 연구용 소프트웨어를 개발하도록 한다.



## 제 2 장 관련연구

### 2.1. CNN 기반 보행자 감지

HOG (Histogram of Oriented Gradients) 기능과 Support Vector Machines (SVM)의 결합은 일반적으로 보행자 탐지에 사용되었다. HOG 기능은 로컬 영역의 그래디언트에 중점을 두고 자세의 작은 변화에 강하다. Dalal 연구에 이어 몇 가지 관련 접근법이 제안되었다. 큰 자세 변화를 다루기 위해 Felzenszwalb는 전체 보행자 모델과 부분 영역을 가진 보행자를 탐지하는 변형 가능한 부품 모델 (DPM: Deformable Part Model) 을 제안했다. DPM은 보행자 감지 벤치 마크에서 최고의 성능을 보였다.[2]

Deep Convolutional Neural Networks (CNN)에 기반한 보행자 검출은 보행자 검출 벤치 마크로 높은 검출 정확도를 보였다. LeCun이 제안한 CNN이 주목을 받았는데, Krizhevsky는 CNN을 물체 인식에 적용했다. CNN은 ReLU (Rectified Linear Unit)를 활성화 함수로 사용하고 일반화를 위해 드롭아웃을 사용한다. 드롭아웃은 고정된 비율의 단위를 임의로 제거한다. 이 비율은 대개 50 %로 설정되고 선택한 단위의 응답 값은 0 이다. 트레이닝 과정을 반복할 때마다 다른 단위가 선택되었다. Dropout으로 교육된 네트워크는 일반화 성능이 향상되었지만 이 무작위 추출은 트레이닝 과정에만 적용된다.

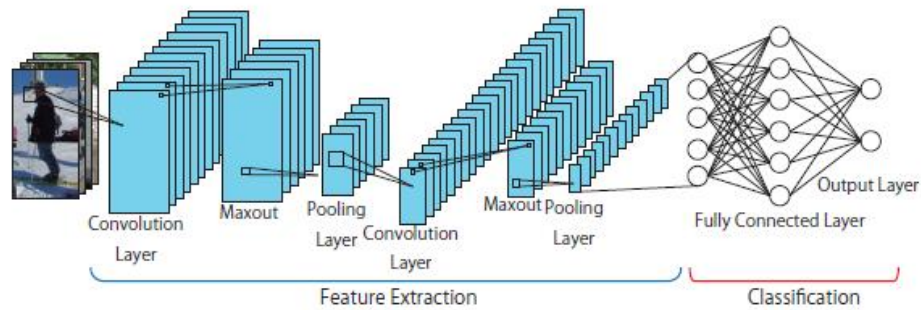


Figure 2 전통적인 CNN 구조 [2]

보행자 감지를 위해 CNN 기반의 방법 중 Random Dropout 및 Ensemble Inference Network (EIN)를 각각 훈련 및 분류 프로세스에 도입하는 방법을 제안하며 이 방법은 비록 제안 된 방법의 구조가 상당히 간단하더라도, 최첨단 기술에 필적하는 성능을 달성하였다. 이 연구에서는 탈락에 기초한 유닛의 무작위 선택에 의한 보행자 검출 개선을 위한 두 가지 기법을 제안했다.

랜덤 드롭 아웃 (Random Dropout)은 단위의 해당 값을 유연한 비율로 무작위로 설정한다. EIN은 완전히 연결된 계층에서 서로 다른 구조를 갖는 다중 네트워크를 최종 출력 결정 방식으로 구성한다.[2]



## 2.2. 결정레벨 Fusion 에 의한 사물 감지와 분류

센서 기반 지능형 차량 시스템에 의해 탐지된 물체의 탐지 및 분류 시, 정확한 물체 감지 및 분류를 위해 다중 센서 정보를 표현하고 인식하기 위해 의사 결정 단계에서의 퓨전 (decision-level fusion) 을 이용한 새로운 물체 검출 및 분류 방법을 제안하였다.[3]

센서 기반 지능형 차량 시스템에 의해 정확한 물체를 감지 및 분류를 위해 다중 센서 정보를 표현하고 인식하는 과정이 필요하다. 본 논문에서는 지능형 차량의 다중 센서 방식에 대한 물체의 정확한 분류를 위하여 물체 감지 및 의사 결정 융합 방법을 위한 새로운 물체 영역 제안 생성 방법을 제안했다.

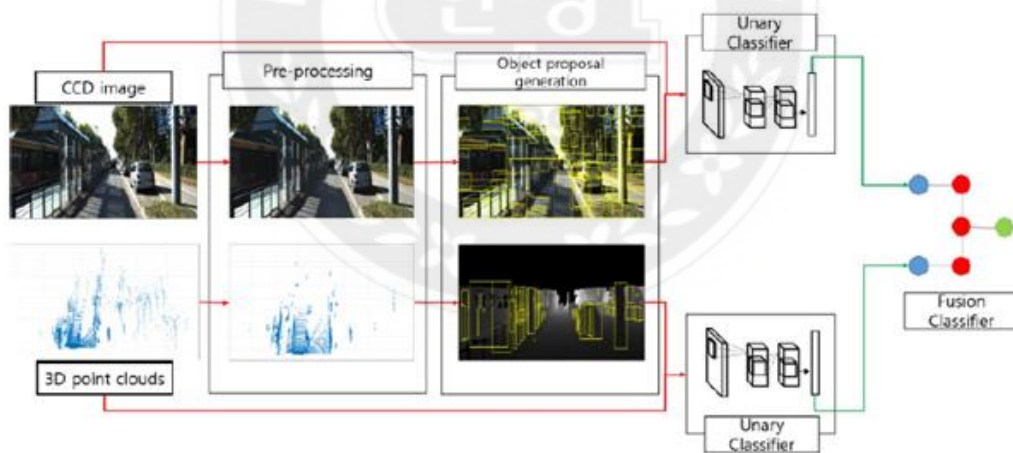


Figure 3 물체 감지 및 의사 결정 융합 방법[3]

이 연구에서는 3D 포인트 클라우드(point cloud) 및 컨볼루션 뉴럴 네트워크 (CNN)를 사용한 이미지 데이터와 같은 독립적인 단일 분류기(unary classifiers)의 분류 결과를 통합하였다.

두 센서에 대한 unary classifiers 는 5 개 레이어가 있는 CNN이며, 두 개 이상의 two pre-trained convolutional layers 를 사용하여 global features 에 대한 로컬을 데이터 표현으로 간주하였다.

convolutional layers 를 사용하여 데이터를 표현하기 위해 object proposal generation 을 사용하여 생성 된 object candidate regions 의 각 레이어 결과에 region of interest (ROI) 풀링을 적용하였다



### 2.3. FCN 을 이용한 3D 라이다의 차량 감지

Convolutional network 기술은 최근 비전 기반 detection 작업에서 큰 성공을 거두었다.[5]

본 논문에서는 fully Convolutional network 기술을 3D 범위 스캔 데이터의 detection 작업에 이식하였다. 구체적으로 시나리오는 Velodyne 64E 라이다의 범위 데이터로부터 차량 탐지 작업으로 설정되었다.

우리는 2D 포인트 맵에 데이터를 제시하고 객체의 신뢰도와 경계 상자를 동시에 예측하기 위해 단일 2D 종단 간 fully convolutional network 를 사용할 것을 제안하였다. 경계 상자 인코딩을 신중하게 설계함으로써 2D convolutional network 를 사용해도 3D 경계 상자 전체를 예측할 수 있다. KITTI 데이터 세트에 대한 실험은 제안 된 방법의 최첨단 성능을 보여주었다.

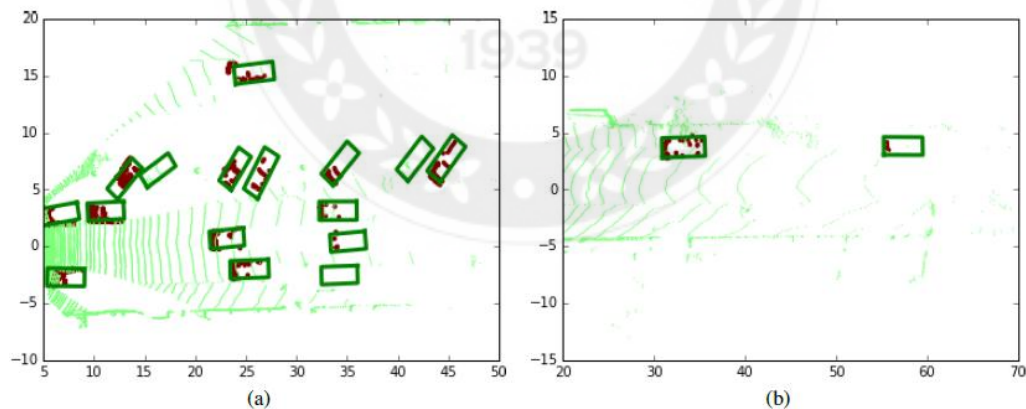


Figure 4 (a)혼잡 도로에서 감지 결과 (b)원거리 차량의 감지 결과 [5]

전체 객체 탐지 알고리즘은 point cloud 에서 후보를 제안한 다음 객체로 분류한다. 알고리즘의 일반적인 범주는 point cloud 을 클러스터로 분할하

여 후보를 제안한다.

후속 물체 감지(subsequent object detection)는 각 세그먼트를 분류하여 이루어지며 때로는 부정확한 세분화에 취약하다.

이 문제를 피하기 위해, Behley et al. 는 장면을 계층적으로 구분하고 다른 크기의 세그먼트를 유지할 것을 제안하였다. 다른 방법은 부정확한 분할을 피하기 위해 범위 스캔 공간을 직접 소모하여 후보를 제안하였다.

예를 들어, Johnson과 Hebert는 포인트 클라우드의 점들을 무작위로 표본으로 서 표본화 한다. Wang과 Posner 는 전체 공간을 슬라이딩 윈도우로 스캔하여 제안을 생성한다.

후보 데이터를 분류하기 위해 일부 초기 연구에서는 알려진 형상 모델을 가정하고 모델을 범위 스캔 데이터와 일치시킨다

## 2.4. 카메라와 Lidar Fusing 을 이용한 사물 인지와 트래킹

Google 및 Tesla와 같은 학계 및 업계에서 큰 성공을 거둔 자율 운전 및 고급 운전자 보조 시스템이 대중의 주목을 끌고 있다.

자율 주행 기술이나 ADAS의 다양한 모듈 중에서 DATMO (Detection and Tracking Multiple Object)는 교통 환경을 이해하는 데 가장 중요한 구성 요소 중 하나이다.[8]

지능형 주행 (또는 보조) 시스템의 입력에 대해 다양한 센서가 고려 될 수 있다. 일반적으로 많은 연구자들은 값 싸고 다기능 특성 때문에 컬러 카메라를 사용한다. 최근에, 일부 연구자들은 물체 감지 성능면에서 획기적인 발전을 이루었다. 특히, 보행자 감지 기술의 발전은 아직까지 계속되고 있다.

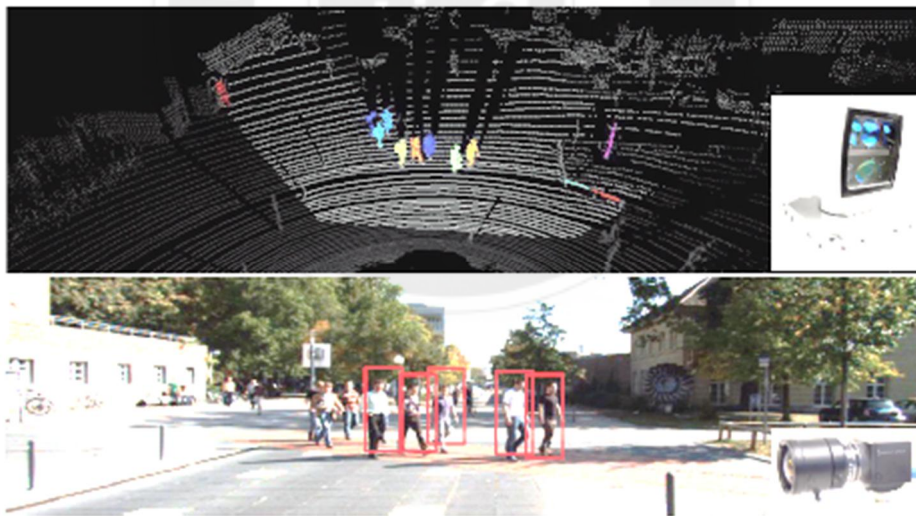


Figure 5 LIDAR 3d point cloud 와 카메라 칼라 영상[8]

컬러 카메라가 대중적인 선택이지만 센서에서 거리를 측정하기 위해 ToF (time-of-flight) 유형 센서를 사용하는 또 다른 연구 방향이 있다. 지능형

차량에 가장 널리 사용되는 센서는 LIDAR (위치 감지 및 범위 설정) 센서이다. 인간의 지각 시스템과 마찬가지로 물체와의 거리는 이 작업에서 유용할 수 있다.

센서 퓨전 접근법이라고하는 여러 센서를 최대한 활용하려고 시도한 연구도 있다. 센서 융합 접근 방식은 보완적인 특성(예: 빛의 파장 또는 독립적인 물리량)으로 인해 매우 정확한 인식 시스템을 구축하는 유망한 방향이다.

또한, 많은 학술 연구 또는 산업 제품은 멀티 센서 플랫폼을 기반으로 개발되었다. 그러나, 높은 계산 복잡성은 일반적으로 센서 퓨전 프레임워크에서 제한된다.

본 논문[8]에서는 여러 개의 독립적인 모듈로 구성된 객체 검출 및 추적을 위한 효율적인 프레임 워크를 제안한다. 또한 우리는 프레임 워크에서 큰 속도 향상을 이끌어내는 2D / 3D 조인트 객체 제안에 이어 그리드 기반 투영 DBSCAN을 포함한 새로운 세분화 기법을 설계하였다.

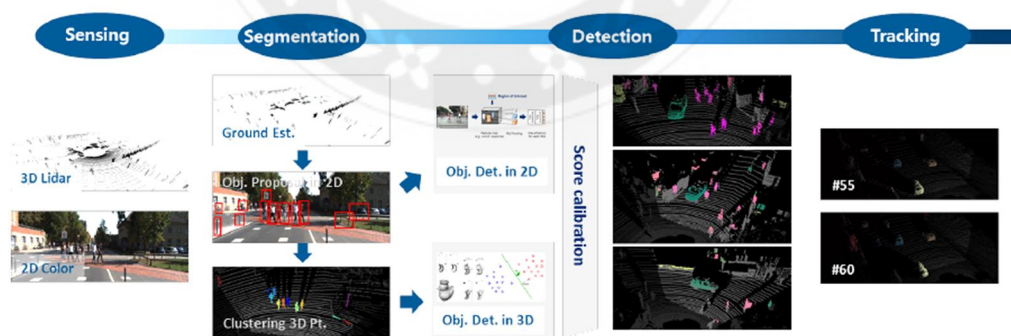


Figure 6 카메라와 Lidar 의 입력 값에 대한 처리 프레임워크[8]

또한 센서 퓨전 방식을 채택하여 성능을 향상시켰다. 보행자 및 차량과 같은 도로의 대표 타겟에 초점을 맞추었다.

이 연구에서는 여러 객체 (DATMO)를 탐지하고 추적하기 위한 효율적인 프레임 워크를 제안하였다. 그림 2에서 볼 수 있듯이, 제안 된 프레임 워크는 4 단계로 구성된다.

먼저 입력 데이터가 카메라와 3D LIDAR에서 입력된다. 두 번째로, 탐지 및 추적 단계에 대한 입력을 준비하는 분할 단계가 뒤 따른다. 그런 다음, 다른 방식, 즉 3D 포인트 공간 및 2D 이미지 공간으로부터의 2 개의 독립적인 오브젝트 검출기가 적용된다. 마지막으로 추적 기능의 계산 및 관련성을 포함하는 추적 절차가 제공되었다.



## 제 3 장 설계 및 구현

### 3.1. 개발환경

하드웨어 (테스트용 차량) : 자율주행차 개발을 위한 차량 주행 데이터 수집 및 분석이 차량 측위용 GPU 장비 NVIDIA Drive PX2와 그 외 센서 장비들 (gmsl camera(4~6), lidar(1~5), GPS/IMU, CAN 등) 을 차량에 장착한 후 실도로 주행을 통해 데이터를 수집하였다

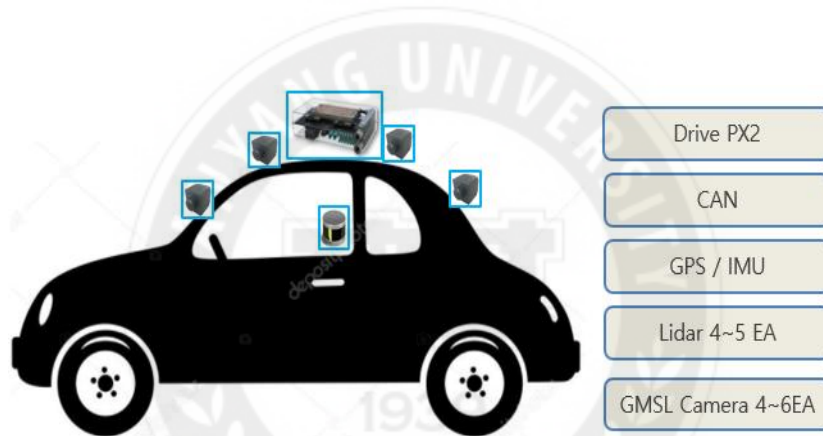


Figure 7 센서 기반 ADAS 시험 차량

각종 센서들로부터 수집된 데이터를 실시간으로 처리하기 위한 GPU 기기 기반 소프트웨어 플랫폼인 Nvidia Driveworks 에서 제공하는 기능을 이용하였는다. 분석을 위한 데이터 수집을 위해 동기화된 다중 센서 데이터 레코딩 기능을 이용하였으며, 차량, 보행자, 표지판 감지를 위하여 다중 Camera 영상 분석 네트워크인 DriveNet 를 이용하였으며, 영상 기반 거리 측정을 위한 camera calibration tool 을 이용하였으며 Lidar 데이터 분석을 위하여 3d point cloud player 기능을 이용하였다.



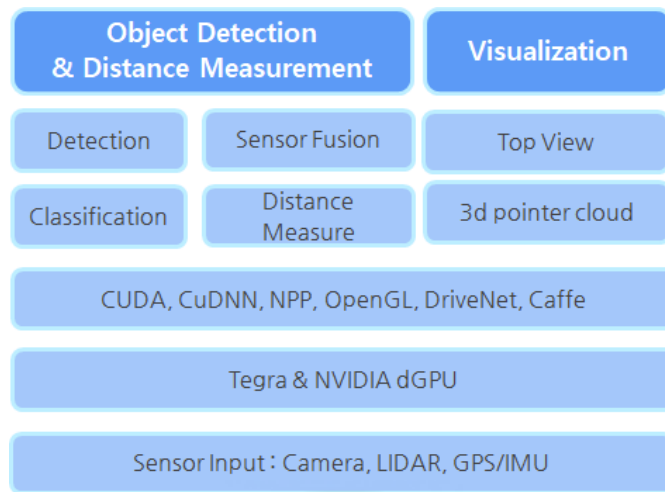


Figure 8 GPU 기반 다중 센서 데이터 수집 및 분석개발 플랫폼

개발은 ubuntu 16.04 리눅스 GPU PC 에 PC 용 Driveworks 를 설치하여 진행 하였으며 ARM 용 GCC로 빌드하여 진행하였다.

항목	버전
운영 체제	Ubuntu 16.04
컴파일러	gcc version 4.9.3 (Ubuntu 4.9.3-13ubuntu2)
cmake	cmake version 3.2.3
CUDA driver	nvcc: NVIDIA (R) Cuda compiler driver Cuda compilation tools, release 8.0, V8.0.72
Open GL	OpenGL version string: 4.5.0 NVIDIA 384.9
Open CV	3.1.0
QT	Qt version 4.8.7
Python	Python 2.7.12

Table 1 PC 개발 환경 테이블

### 3.2. Flow Chart

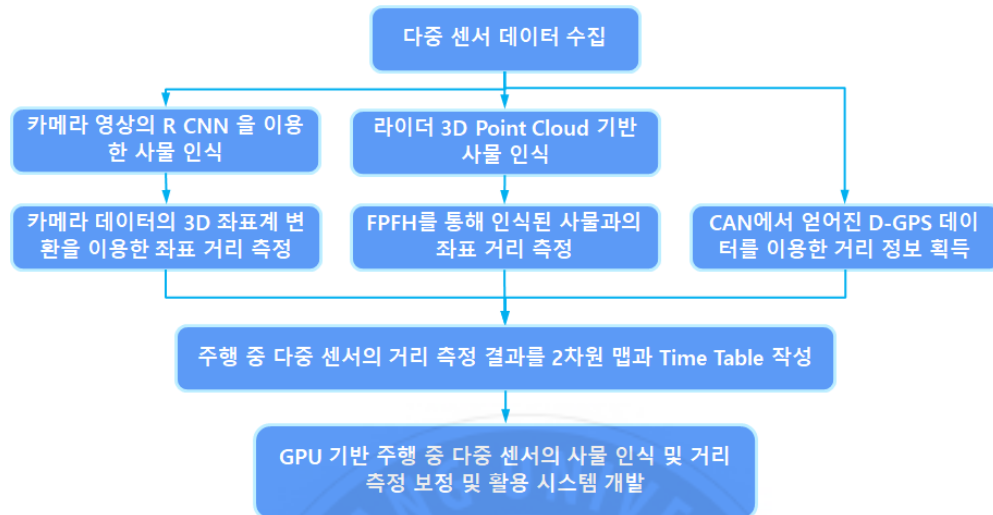


그림 3.3 소프트웨어 흐름 설명

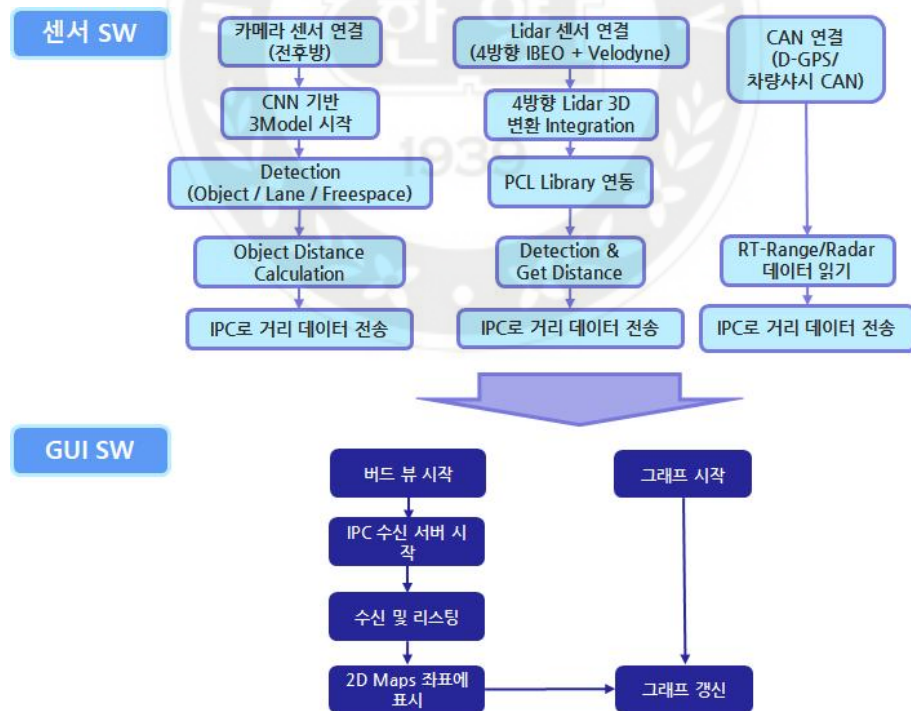


Figure 9 센서 SW 와 GUI SW 의 플로우 차트

### 3.3. 구현

다중 센서 데이터 수집 및 분석을 위해 다음과 같은 flow 로 소프트웨어가 동작한다. 카메라 센서를 연결한 후 수집되는 영상 데이터에서 CNN 기반 모델링으로 사물을 감지하며 감지된 사물을 대상으로 거리를 계산한다. Lidar를 연결한 후 수집되는 3D Point Cloud 데이터에서 PCL 라이브러리를 이용하여 사물을 감지하며 거리 정보를 획득한다. 이 두 가지 거리 데이터를 테스트 차량에서 들어온 정확한 거리 정보와 비교하여 시간에 따른 정확도를 관찰하기 위해 거리 2D Map 과 타임 그래프를 작성한다.



### 3.3.1 CNN 기반 카메라 사물 인지 구현

컨볼루션 뉴럴 네트워크 (Convolutional Neural Network, CNN)는 하나 이상의 컨볼루션 레이어 (종종 서브 샘플링 단계가 있음)로 구성되고 표준 멀티 레이어 뉴럴 네트워크와 같이 하나 이상의 완전히 연결된 레이어가 이어진다. CNN의 아키텍처는 입력 이미지 (또는 음성 신호와 같은 다른 2D 입력)의 2D 구조를 이용하도록 설계되었다. 이는 로컬 연결 및 연결된 가중치와 함께 변환 불변 기능을 발생시키는 형태로 이루어졌다. CNN의 또 다른 이점은 동일한 수의 숨겨진 유닛을 가진 완전히 연결된 네트워크보다 더 적은 수의 매개 변수를 가지고 훈련이 쉽다는 것이다.

CNN은 선택적으로 완전히 연결된 레이어가 뒤따르는 다수의 길쌈 (convolutional) 및 서브 샘플링 (subsampling) 레이어로 구성된다. 컨볼루션 층에 대한 입력은  $m \times m \times r$  화상인데, 여기서,  $m$ 은 화상의 높이와 폭이고,  $r$ 은 채널의 수이다. RGB 이미지는  $r = 3$ 이다. 컨볼루션 층은  $n \times n \times q$ 의  $k$  필터 (또는 커널)를 가지며, 여기서  $n$ 은 이미지의 크기보다 작고,  $q$ 는 채널의 수  $r$ 보다 작거나 같을 수 있으며 각 커널마다 다를 수 있다. 필터의 크기는 크기  $m-n+1$ 의  $k$  개의 특징 맵을 생성하기 위해 이미지와 각각 컨볼루션 된 국부적으로 연결된 구조를 발생시킨다. 각각의 맵은 보통 작은 이미지 (예: MNIST)의 경우  $p$ 가 2이고 큰 입력의 경우 보통 5보다 크지 않은  $p \times p$  연속 영역에서 평균 또는 최대 풀링으로 서브 샘플링 된다. 서브 샘플링 레이어 전후에 추가 바이어스 및 S자형 비선형 성이 각 피쳐 맵에 적용된다.

Object detection은 입력 영상이 주어졌을 때, 영상 내에 존재하는 모든 카테고리에 대해서 classification과 localization을 수행하는 것을 말한다.

CNN 기반의 object detection 중에서 높은 성능과 빠른 속도로 큰 주목을

받고 있는 것은 region based CNN 이다.

하지만, 탐색해야 하는 영역의 수가 너무 많기 때문에 classifier가 충분히 빠르지 않다면 연산시간이 오래 걸리는 비효율적인 방법이다. sliding window 방식의 비효율성을 개선하기 위해서, 입력 영상에서 ‘물체가 있을 법한’ 영역을 빠른 속도로 찾아내는 알고리즘을 region proposal 알고리즘이라 한다.

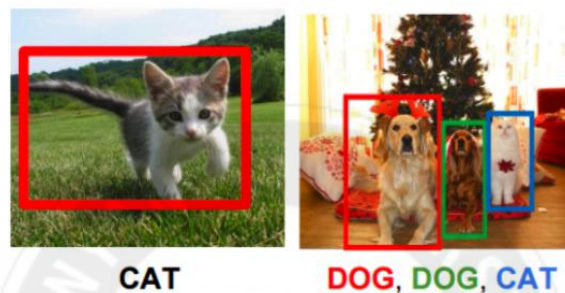


Figure 10 Classification + Localization과 Object detection

(출처: <https://blog.lunit.io/2017/06/01/r-cnns-tutorial/>)

Region proposal을 활용하면, sliding window 방식에 비해 search space가 확연하게 줄어들기 때문에 훨씬 빠른 속도로 object detection을 수행할 수 있게 된다.

R-CNN 은 Region Proposal 과 Convolutional Neural Network (CNN) 을 합한 개념이다.

R-CNN<sup>[14]</sup>은 Image classification을 수행하는 CNN과 이미지에서 물체가 존재할 영역을 제안해주는 region proposal 알고리즘을 연결하여 높은 성능의 object detection을 수행할 수 있음을 제시해주었다.

	Method	Low level	Output type	Score	#win-dows	Time	Repeatability	Quality	Detection
Pb	gPbUCM	Own	Sg	No	F	-	-	-	-
O	Objectness	sp	Bx	Yes	R	3	-	*	*
C	CPMC	Own	Sg	Yes	Fs	250	-	**	*
E	Endres2010	sp	Sg	Yes	Fs	100	-	**	**
SS	SelectiveSearch	sp	Sg	No	F	10	**	***	**
R1	Rahtu2011	sp	Sg	Yes	R	3	-	*	*
RP	RandomizedPrim	sp	Sg	No	LR	1	*	*	*
B	Bing	Own	Bx	Yes	Fs	0.2	***	*	*
M	MCG	C*	Sg	Yes	R	30	*	***	**
R4	Rantalankila2014	C+sp	Sg	No	F	10	**	*	*
R	Rigor	C*	Sg	No	LR	-	-	-	-
EB	EdgeBoxes	Own	Bx	Yes	Fs	0.3	**	***	**
U	Uniform	Ø	Bx	No	R	0	-	-	-
G	Gaussian	Ø	Bx	No	R	0	-	-	*
SW	SlidingWindow	Ø	Bx	No	R	0	***	-	-
SP	Superpixels	Own	Sg	No	F	1	*	-	-

Table 2 Region proposal 알고리즘 간의 성능 비교

(출처: <https://blog.lunit.io/2017/06/01/r-cnns-tutorial/>)

R-CNN의 수행 과정은 1) 이미지를 입력으로 받음. 2) 이미지로부터 약 2000개 가량의 region proposal을 추출함 (Selective search[2] 활용) 3) 각 region proposal 영역을 이미지로부터 잘라내고(cropping) 동일한 크기로 만든 후(warping), 4) CNN을 활용해 feature 추출함으로 진행된다.

R-CNN은 상향식 영역 제안과 컨볼루션 신경망에 의해 계산된 풍부한 기능을 결합한 최첨단 시각적 객체탐지 시스템이다. 출시 당시 R-CNN은 PASCAL VOC 2012의 이전 최상의 검색 성능을 평균 대비 40.9 %에서 평균 53.3 %로 30 % 향상 시켰다. 이전의 최상의 결과와 달리 R-CNN은 문맥 재검색이나 피쳐 유형의 앙상블을 사용하지 않고도 성능을 달성하였다.

R-CNN의 경우 미리 학습된 AlexNet을 변형시켜 만든 CNN 인데, 마지막 단계의 이미지 분류를 위해 SVM을 도입해 사용하였다. 그리고 오브젝트의 분류 결과인 boxing 좌표의 정확도를 높이기 위해 선형 회귀(linear regression) 모델을 사용한다.

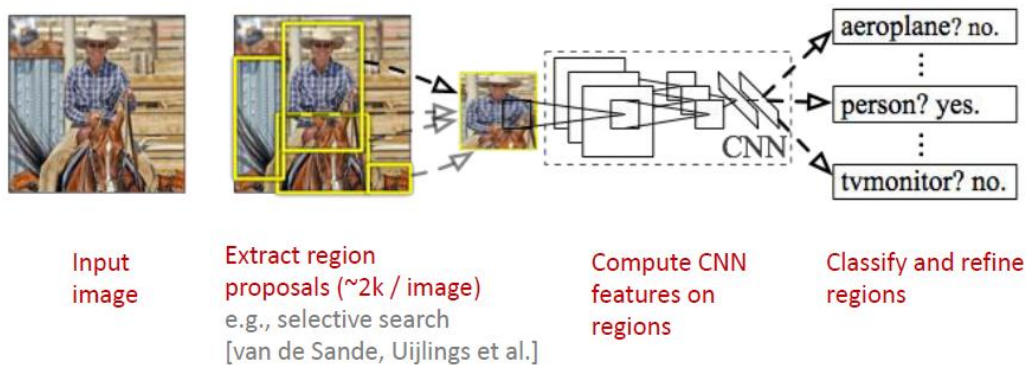


Figure 11 사물 감지를 위한 R-CNN

(출처: <http://tutorial.caffe.berkeleyvision.org/caffe-cvpr15-detection.pdf>)

R-CNN 은 계산 비용이 크지만, 컨볼루션을 공유함으로써 비용이 크게 감소되었다. 최근에는 Fast R-CNN 가 매우 깊은 네트워크를 사용하여 거의 실시간 속도를 달성한다.

이미지 분류 교육 데이터 샘플은 클래스 (일반적으로 정수 클래스 ID 또는 문자열 클래스 이름)로 레이블 된 이미지 (일반적으로 단일 이미지를 포함하는 작은 이미지 또는 패치)이다. 반면에 물체 감지는 훈련에 더 많은 정보가 필요하다. DetectNet 교육 데이터 샘플은 여러 개체가 포함된 더 큰 이미지이다. 이미지의 각 오브젝트에 대해 학습 레이블은 오브젝트의 클래스뿐만 아니라 경계 상자의 모서리 좌표를 캡처해야 한다. 학습 이미지간에 개체의 수는 다양 할 수 있기 때문에 다양한 길이와 차원의 레이블 형식을 순진하게 선택하면 손실 기능을 정의하는 것이 어려워진다.

DetectNet은 고정 된 3 차원 레이블 형식을 도입하여 다양한 크기의 이미지를 다양한 개체로 처리 할 수 있게 함으로써 이 핵심 문제를 해결한다.



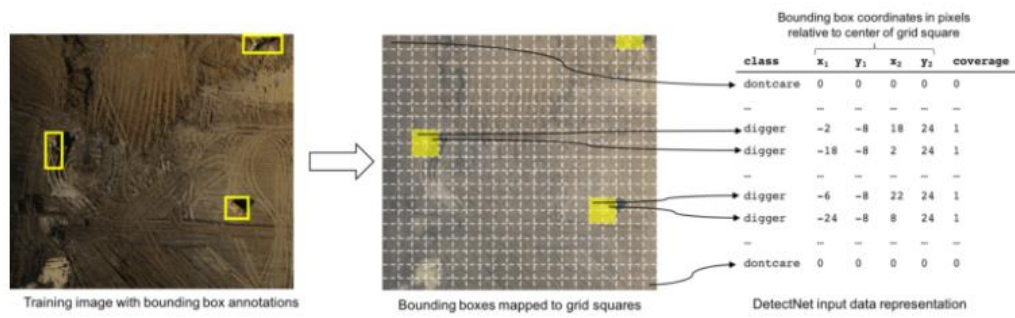
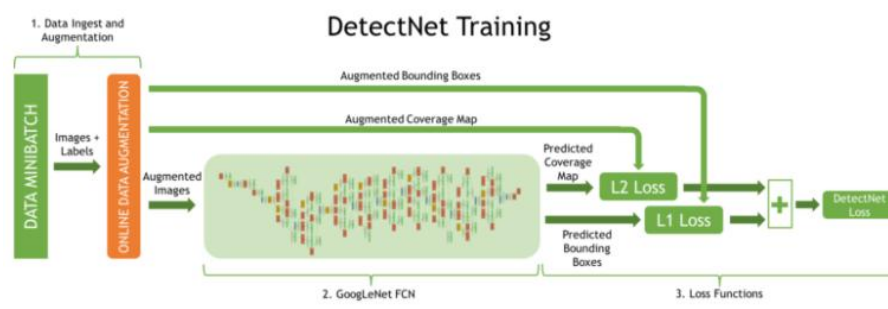


Figure 12 DetectNet 입력 데이터 표현

(출처: <https://goo.gl/hCS8U8>)

DetectNet 교육 데이터 샘플은 여러 개체가 포함 된 더 큰 이미지이다. 이미지의 각 오브젝트에 대해 학습 레이블은 오브젝트의 클래스뿐만 아니라 경계 상자의 모서리 좌표를 캡처해야 한다. 학습 이미지 간에 개체의 수는 다양 할 수 있기 때문에 다양한 길이와 차원의 레이블 형식을 순진하게 선택하면 손실 기능을 정의하는 것이 어려워진다.

DetectNet은 고정 된 3 차원 레이블 형식을 도입하여 DetectNet이 다양한 크기의 이미지를 다양한 개체로 처리 할 수 있게 함으로써 이 핵심 문제를 해결하였다.





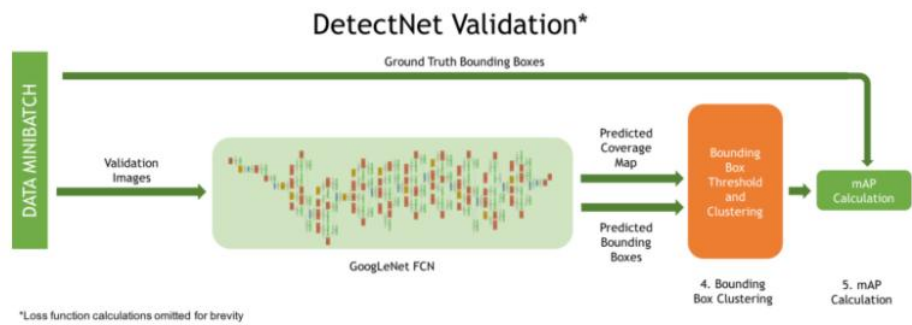


Figure 13 DetectNet Training & Validation

(출처: <https://goo.gl/v9tLq9> )

본 연구에서는 DetectNet 과 같은 사물 인지기능을 DrivePX2 시스템에서 차량 대상으로 개발된 DriveNet 을 이용하여 주행 중 차량 사물 인지를 수행하였다.



Figure 14 주행차량에서 6개 카메라로 DriveNet 으로 사물 인지한 화면

### 3.3.2 OpenCV 기반 거리 계산 구현

현재 카메라 기반 사물 인식 및 거리 계산을 진행하는데 반드시 필요한 센서에 의한 거리 계산에 필요한 카메라 캘리브레이션 (camera calibration) 은 본 연구를 위하여 반드시 요구되는 과정이다.

우리가 3차원의 실제 세상을 카메라로 찍어서 보게 되면 2차원의 이미지가 된다. 이 때 3차원 세상의 좌표 점들이 카메라 이미지에서 2차원 좌표 점으로 변환하는 것은 카메라의 위치와 각도 방향에 따라 기하학적으로 분석을 함으로써 가능해진다.

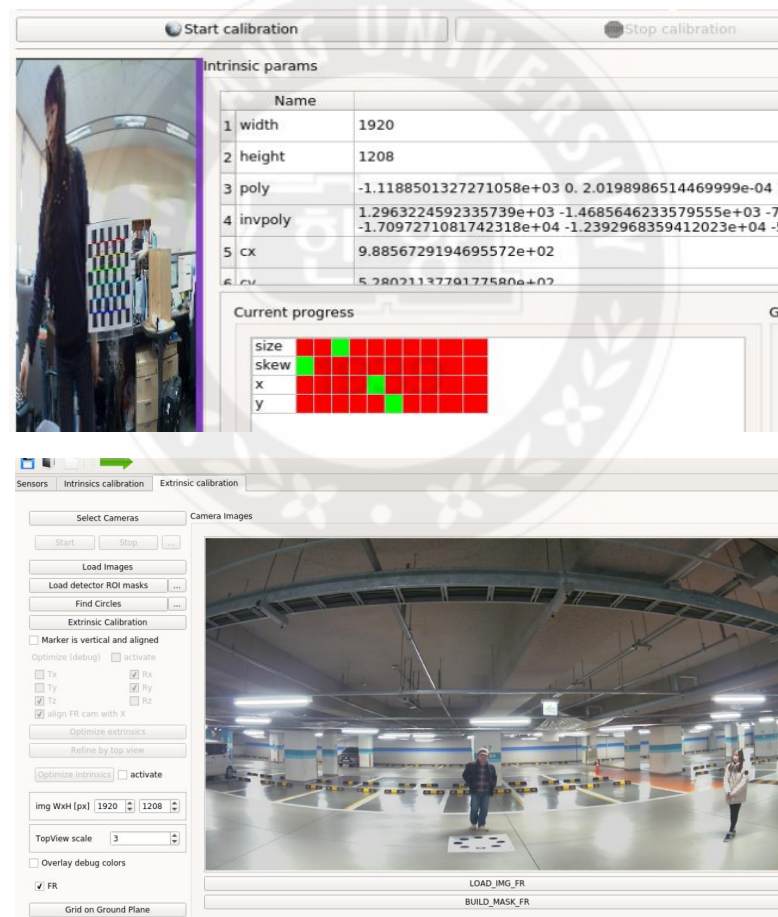


Figure 15 카메라 intrinsic / extrinsic 캘리브레이션

하지만 실제 2차원 이미지 좌표는 카메라의 기구적인 부분에 의해서도 영향을 받는데 카메라 내부의 기구적인 특성은 렌즈의 특성도 있고, 렌즈와 이미지 센서와 렌즈와의 각도와 거리를 말한다. 그러므로 카메라 캘리브레이션 과정이라고 함은, 3차원 세상의 좌표 점들이 카메라 이미지에서 2차원 좌표 점으로 변환하는 연산을 하기 위해서 위와 같은 카메라 내부의 기구적인 특성을 분석하는 과정이다. 즉, 이러한 내부 요인의 파라미터 값을 구하여 rig-configuration 데이터를 구하는 과정이 필요하다는 것이다.

핀홀(pinhole) 종류의 카메라에서 3차원 공간 좌표를 2차원 평면 좌표에 투사하는 변환(perspective projection)에 대한 모델링 한 관계식은 다음과 같다

$$s \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & \text{skew\_cf}_x & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$= A[R|t] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Figure 16 카메라 영상의 2차원 평면에 투사 변환 모델링

(출처: <http://darkpgmr.tistory.com/32>)

그림에서, 실제 공간의 좌표계에서 3D 점의 좌표는 X, Y, Z이고, 실제 공간 좌표계인 월드 좌표계를 카메라 좌표계로 변환시키기 위한 회전/이동 변환 행렬이  $[R|t]$ 이며 A는 camera 의 정보 matrix 이다.

위의 수식에서 보면 카메라 캘리브레이션 과정을 통하여 파라미터를 찾는

것이다.

여기서 카메라 정보 matrix  $A$ 는 카메라 intrinsic parameter 이고 좌표계로 변환시키기 위한 회전/이동변환 행렬  $[R|t]$ 는 카메라 extrinsic parameter 라고 한다. 내부 파라미터 intrinsic parameter 는 카메라의 내부적인 자체적인 특성을 표현한 수치인데 카메라의 중심점이나 초점 거리나 aspect ratio 등이 있다. 외부 파라미터 extrinsic parameter 는 카메라의 위치(높이, 좌표)나 방향(각도, 팬, 틸트) 와 같은 카메라의 좌표 정보를 주는 기하학적인 파라미터이다. 그림 18 에서 실제 공간의 좌표를 월드 좌표계라고 하면 카메라에서 보는 기하학적인 좌표를 카메라 좌표계라고 하며, 카메라에 맺힌 이미지의 2D 좌표를 영상 좌표계라고 한다.

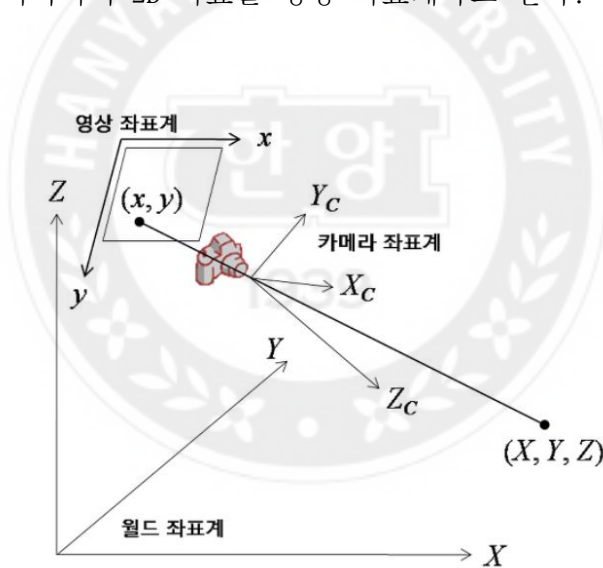


Figure 17 영상좌표계, 카메라 좌표계, 월드 좌표계

(출처: <https://goo.gl/bGhaf7>)

이러한 캘리브레이션 과정 결과로 영상 프레임의 픽셀 좌표를 실제 거리에 매핑하여 감지된 사물과의 거리를 계산한다.

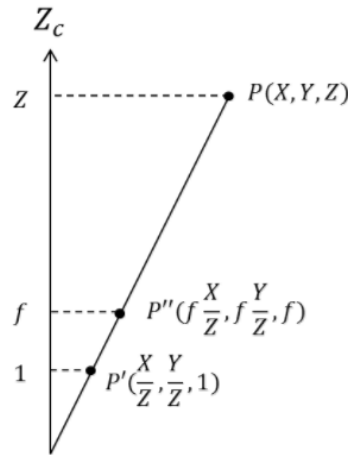


Figure 18 카메라로부터의 가상의 정규 이미지 평면에서의 좌표

(출처: <https://goo.gl/jdfEgz>)

카메라를 중심으로 3차원 좌표계(카메라의 광학축 방향이 Z축, 오른쪽이 X축, 아래쪽이 Y축)를 설정했을 때, Z만큼 떨어져 있는 점  $P(X, Y, Z)$ 의 좌표를 구해보도록 하자. P는 카메라로부터 1 만큼 떨어져 있는 가상의 2차원 이미지 평면 상의 좌표라고 가정한다면 삼각형의 닮은비의 식에 의해서 그 이미지 평면 상의 2차원 좌표는  $P'(\frac{X}{Z}, \frac{Y}{Z}, 1)$ 가 된다.

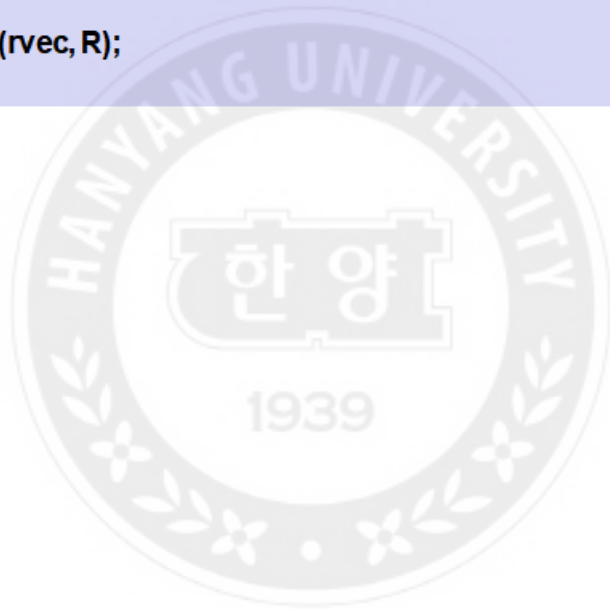
이 때 이미지 2차원 평면에 투사된 영상은 카메라 초점거리 f만큼 곱해줘야 하므로 실제 P에 대응되는 2차원 좌표는  $P''(f \frac{X}{Z}, f \frac{Y}{Z}, f)$ 라고 표현된다. 그래서 이  $P''$ 를 픽셀 좌표계 변환하면 최종 영상좌표를 얻을 수 있다.

카메라 캘리브레이션에서 카메라의 intrinsic parameters 인 왜곡계수( $f_x, f_y, c_x, c_y, k_1, k_2, p_1, p_2$ )를 알고 있으면 카메라의 위치 및 각도 자세를 알 수 있는 extrinsic parameter 를 계산할 수가 있는데 이때 특정 사물에 대한 4개 이상의 월드 좌표와 동일 사물에 대한 2D 이미지 좌표 쌍이 필요

하다.

따라서 특정 길이의 사각형 마크 이미지 보드를 바닥에 놓고 카메라로 이미지를 촬영하고 각각의 사각형 꼭지점들에 대한 2차원 좌표를 구한 후, 그 꼭지점들의 3D 월드좌표와 2D 영상좌표 쌍들을 solvePnP 함수를 이용하여 카메라의 위치 계산을 하면 그것이 extrinsic parameter 계산이 되는 것이고, 이 정보를 이용하여 그대로 카메라와 사물과의 거리를 구하였다.

```
Mat rvec, tvec; // rotation & translation vectors
solvePnP(objectPoints, imagePoints, A, distCoeffs, rvec, tvec);
Mat R, T;
Rodrigues(rvec, R);
T = tvec;
```



### 3.3.3 Lidar 데이터 Point Cloud 분석

Lidar 데이터 처리에서 Point cloud 라는 일종의 Visualization 과정을 거치는데, Point cloud는 좌표계의 데이터 요소 집합이다.

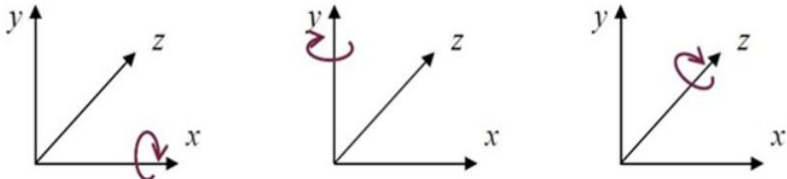
3 차원 좌표계에서 이 점은 일반적으로 X, Y 및 Z 좌표로 정의되며 대개는 객체의 외부 표면을 나타낸다. Point cloud는 측정한 점의 집합을 나타낸다. 3D 스캐닝 프로세스의 결과물 인 포인트 클라우드는 제조 부품, 계측 / 품질 검사 및 수많은 시각화, 애니메이션, 렌더링 및 대량 사용자 정의 응용 프로그램 용 3D CAD 모델을 만드는 등 다양한 목적으로 사용된다.

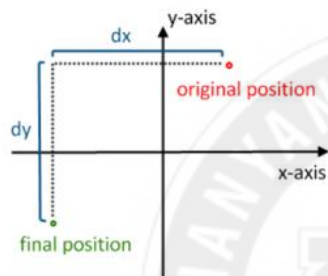
그림 20. 과 같은 4방향 라이더 데이터의 Integration 을 위해서는 3D 데이터의 Rotation & Translation 이 필요하다.



Figure 19 차량에 부착된 4 방향 라이더의 배치도



$$\mathbf{R}_x(\theta) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta & \sin \theta & 0 \\ 0 & -\sin \theta & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{R}_y(\theta) = \begin{bmatrix} \cos \theta & 0 & -\sin \theta & 0 \\ 0 & 1 & 0 & 0 \\ \sin \theta & 0 & \cos \theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{R}_z(\theta) = \begin{bmatrix} \cos \theta & \sin \theta & 0 & 0 \\ -\sin \theta & \cos \theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$




Translation matrix

$$\begin{bmatrix} 1 & 0 & 0 & dx \\ 0 & 1 & 0 & dy \\ 0 & 0 & 1 & dz \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x + dx \\ y + dy \\ z + dz \\ 1 \end{bmatrix}$$

Figure 20 행렬을 이용한 3D 데이터의 회전(상) 이동(하)

(출처: <https://blog.naver.com/hvl814/220597181980>)

(출처: <http://polymathprogrammer.com/2008/09/01/cartesian-coordinates-and-transformation-matrices/>)

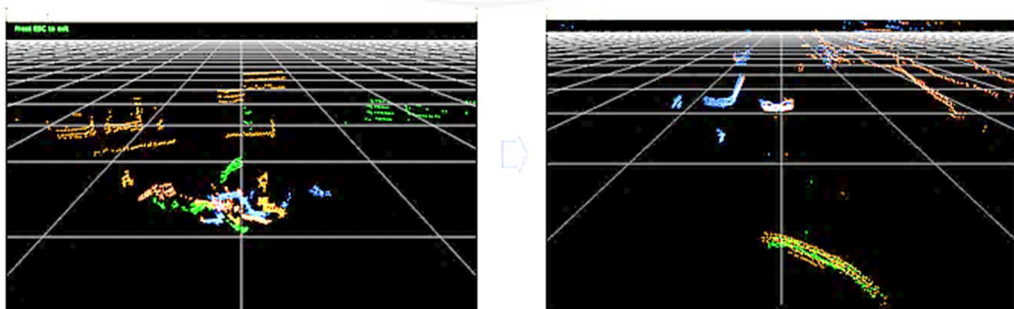


Figure 21 4 방향 라이다 데이터의 회전 이동 전 후 Point Cloud 표시



Point Cloud Library (PCL)는 3 차원 컴퓨터 비전과 같이 포인트 클라우드 프로세싱 작업 및 3D 지오메트리 프로세싱을 위한 알고리즘의 오픈 소스 라이브러리이다. 이 라이브러리는 피쳐 추정, 표면 재구성, 등록, 모델 피팅 및 분할을 위한 알고리즘을 포함한다. C ++로 작성되었으며 BSD 라이선스하에 배포된다.

이러한 알고리즘은 예를 들어, 로봇의 인지로 노이즈가 있는 데이터에서 이상치를 필터링하고, 3D Point cloud를 함께 연결하고, 장면의 관련 부분을 세그먼트로 만들고, 키포인트를 추출하고, 기하학적 모양을 기반으로 세계의 객체를 인식하는 데 사용된다.

Velodyne Lidar 의 360도 3D pointer cloud data 를 PCL (Pointer Cloud Library) 의 FPFH(Fast Point Feature Histogram) 방식의 API를 이용하여 object 를 detect 하였다. Detect 한 사물간의 거리는 Lidar 3D point Cloud Data 의 필드가 x, y, z 이기 때문에 좌표 계산으로 취득하였다.

## 제 4 장 성능평가

주행 중 주변 object 를 감지하여 반응할 수 있는 자율 주행차 개발을 위한 다중 센서 거리 측정 시스템을 GPU 기반 시스템과 CNN 을 이용하여 구현할 수 있다.

다중 센서 거리 측정 시스템을 통하여 각종 센서들로 부터 입력된 데이터로부터 감지된 object 까지의 거리에 대한 효율적인 보정 방안을 연구할 수 있다.

본 연구의 산출물인 다중 센서 거리 측정 시스템을 통해 향후에 다중 센서를 이용한 주행 차량 주변의 사물의 거리를 실시간으로 정확하게 감지하여 위험 거리인 경우 차량 steering을 제어하여 충돌을 피할 수 있는 시스템을 연구 및 개발할 수 있도록 한다.

#### 4.1. 성능 평가

주행 중 감지하고자 하는 1 대의 대상 차량에서 습득한 데이터를 가지고 연구개발된 소프트웨어 성능을 분석하였다.

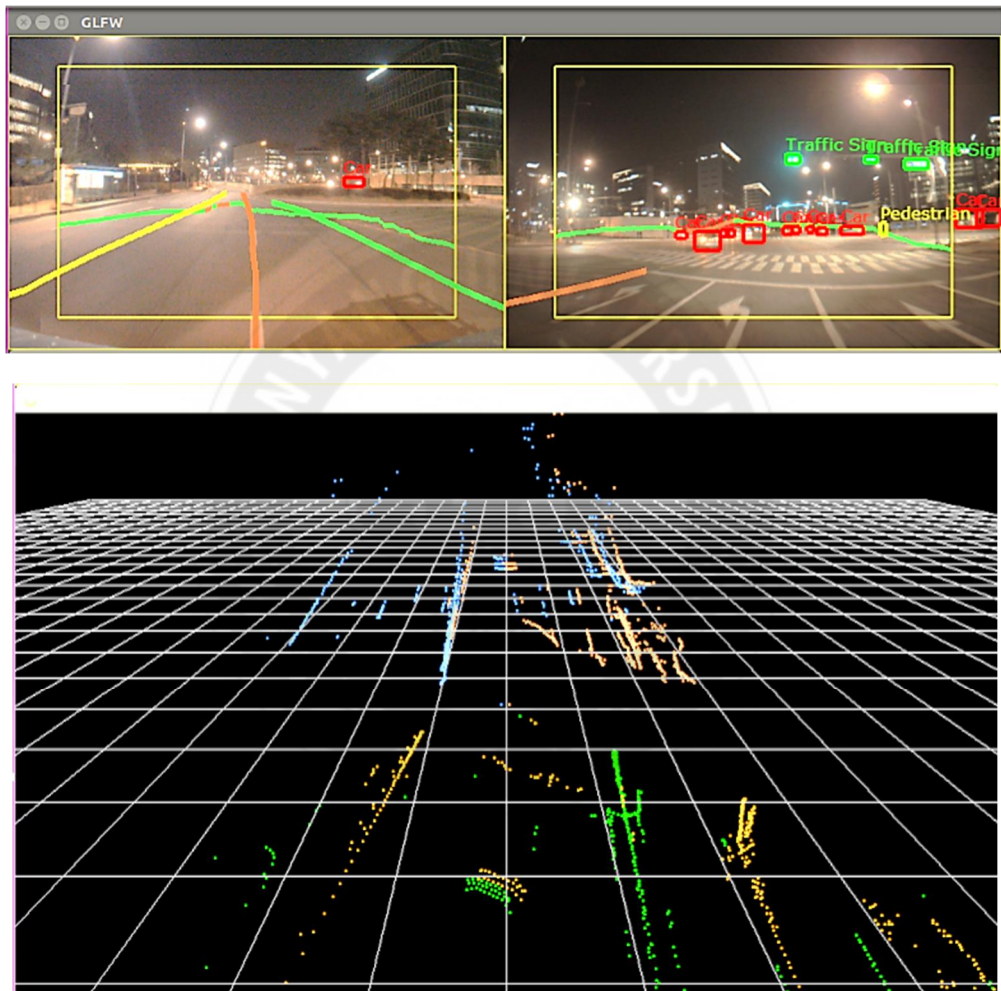


Figure 22 차량 주행 중 카메라와 라이다 데이터 분석

센서들의 습득된 대상 차량 간의 거리 데이터를 IPC를 통하여 open gl 로 개발한 2D Maps (5m 간격)와 python+qt로 개발한 Graph GUI 에 주행 중 나

타넨 결과 (Yellow : Camera, Red : Lidar, Blue : D-Gps) 이다.

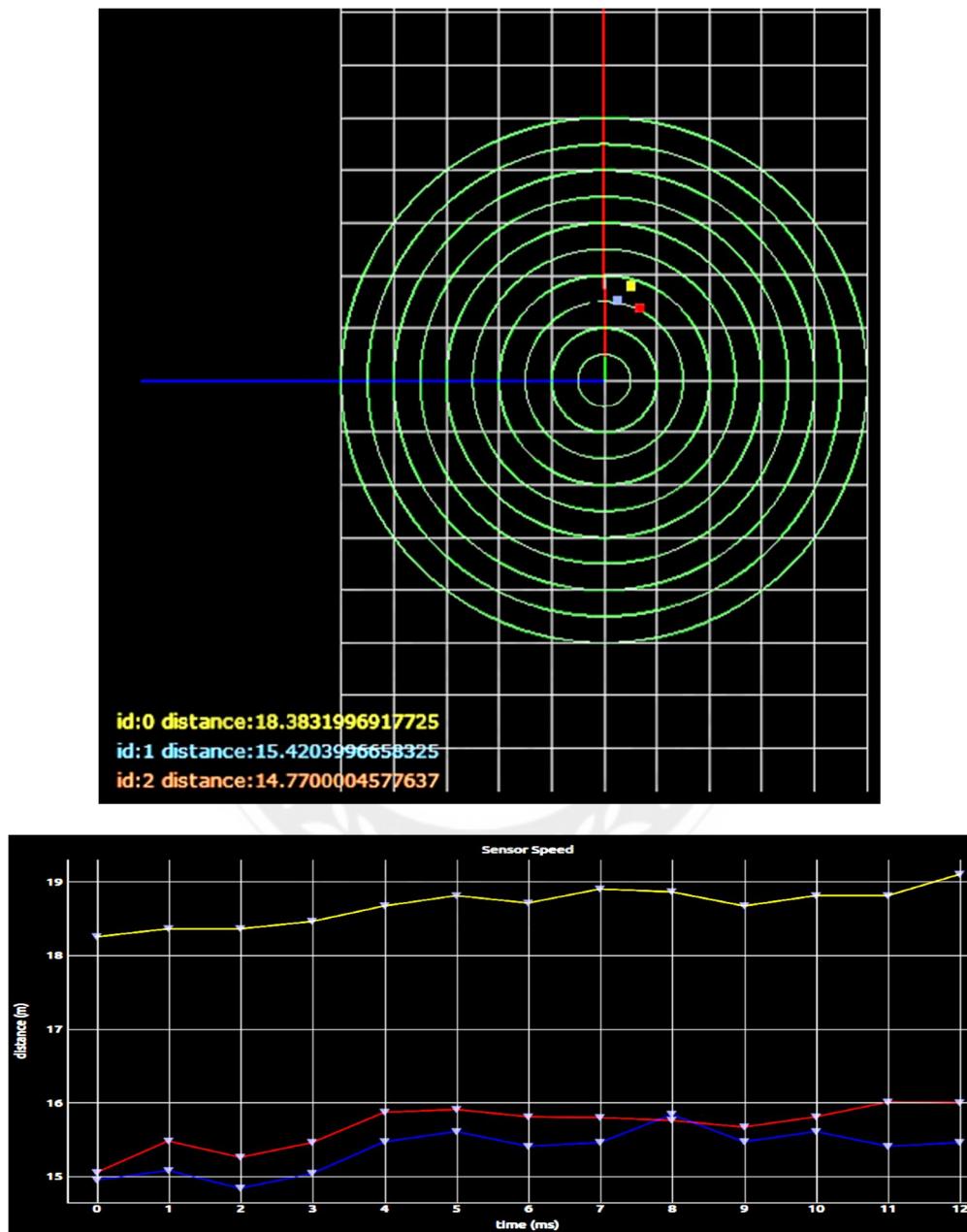


Figure 23 DGPS, 카메라, Lidar에서 취득한 거리의 2D Map과 시간그래프

msec	D-GPS	CAM	LIDAR
23.0	3.35027	3.517784	3.367021
23.5	3.91171	4.068178	3.950827
24.0	4.7164	3.77312	5.164458
24.5	4.76383	5.240213	5.264032
25.0	5.34062	5.233808	5.308576
25.1	5.74565	5.171085	5.602009
25.5	7.76939	9.323268	7.839315
26.0	7.5229	8.27519	7.530423
26.5	7.5988	6.45898	8.16871
27.0	8.5341	9.814215	6.912621
27.5	11.3518	11.46532	10.27338
28.0	14.3394	14.19601	14.32506
28.5	14.5653	15.00226	16.7501
29.0	14.9808	17.88708	14.51939
29.5	15.4864	13.93776	15.4895
30.0	13.1945	14.25006	13.26047
30.5	16.9346	14.05572	16.24028
31.0	12.7545	13.39223	14.09372
31.5	10.6928	10.90666	11.01358
32.0	12.6195	11.98853	12.6495
32.5	16.4343	16.53291	16.3843

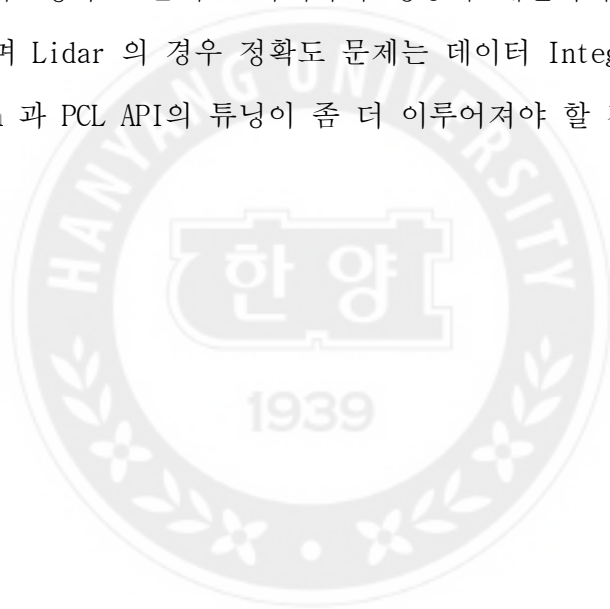
Table 3 DGPS, 카메라, Lidar에서 취득하는 대상차량과의 거리 테이블

msec	CAM	LIDAR
23.0	0.167514	0.016751
23.5	0.156468	0.039117
24.0	0.94328	0.448058
24.5	0.476383	0.500202
25.0	0.106812	0.032044
25.1	0.574565	0.143641
25.5	1.553878	0.069925
26.0	0.75229	0.007523
26.5	1.13982	0.56991
27.0	1.280115	1.621479
27.5	0.113518	1.078421
28.0	0.143394	0.014339
28.5	0.436959	2.184795
29.0	2.906275	0.461409
29.5	1.54864	0.003097
30.0	1.05556	0.065973
30.5	2.878882	0.694319
31.0	0.637725	1.339223
31.5	0.213856	0.320784
32.0	0.630975	0.03
32.5	0.098606	0.05
평 균	0.848358	0.461477

Table 4 DGPS, 카메라, Lidar에서 취득하는 대상차량과의 거리 오차율

카메라 데이터의 경우 전방 카메라에서 DriveNet 기반으로 감지된 대상 차량은 1채널당 24Fps로 감지되며, Calibration 결과 이미지 데이터 분석 기반 대상과의 오차율은 주행 상태에서  $\pm 0.8$  m 로 나오고 있다. Lidar 데이터의 경우 전방 Lidar 2개의 Integration 결과 데이터를 PCL 라이브러리 함수의 FPFH API를 이용하여 사물을 감지한 후 거리 데이터를 취득한 결과 거리 오차율이  $\pm 0.5$  m 로 나온다.

카메라의 경우 정확도 문제는 카메라의 성능과 계산식의 튜닝이 필요할 것으로 보이며 Lidar 의 경우 정확도 문제는 데이터 Integration 에 있어서 Calibration 과 PCL API의 튜닝이 좀 더 이루어져야 할 것으로 보여진다.



## 제 5 장 결론 및 향후 연구

본 논문에서는 주행 중 주변 object 를 감지하여 반응할 수 있는 자율 주행차 개발을 위한 다중 센서 거리 측정 시스템을 GPU 기반 시스템과 CNN 을 이용하여 구현하였다.

다중 센서 거리 측정 시스템을 통하여 각종 센서들로부터 입력된 데이터로부터 감지된 사물까지의 거리에 대한 효율적인 보정 방안을 연구할 수 있었다.

본 연구의 산출물인 다중 센서 거리 측정 시스템을 통해 향후에 다중 센서를 이용한 주행 차량 주변의 사물의 거리를 실시간으로 정확하게 감지하여 위험 거리인 경우 차량 steering을 제어하여 피할 수 있는 구조를 연구 및 개발할 수 있도록 한다.

향후 Multi-Camera 의 스테레오 비전 기반 거리 측정 데이터 도입과, 영상과 라이다 데이터를 CNN 기반 분석으로 사물 감지 및 거리 측정 보정 연구가 필요하며, 다중 센서의 거리 보정이 완성되면 CAN 을 통한 차량 steering 정보와 연동 및 제어할 수 있도록 CAN 데이터를 출력하여 자율주행 제어 입력 데이터로 활용될 예정이다.

## 참고문헌

- [1] ADAS 등 자율주행차 관련 기술 동향 - 교통과학연구원, 2016
- [2] Hiroshi Fukui, Takayoshi Yamashita, Yuji Yamauchi, Hironobu Fujiyoshi, Hiroshi Murase, Pedestrian Detection Based on Deep Convolutional Neural Network, Chubu University, 2015
- [3] Sang-Il Oh and Hang-Bong Kang, Object Detection and Classification by Decision-Level Fusion for Intelligent Vehicle Systems, Catholic University of Korea, 2017
- [4] Brody Huval, Tao Wang, Sameep Tandon, Jeff Kiske, Will Song, An Empirical Evaluation of Deep Learning on Highway Driving, Stanford University, 2015
- [5] Bo Li, Tianlei Zhang and Tian Xia, Vehicle Detection from 3D Lidar Using Fully Convolutional Network, Vehicle Baidu Research - Institute for Deep Learning, 2016
- [6] Joel Schlosser, Christopher K. Chow, and Zsolt Kira, Fusing LIDAR and Images for Pedestrian Detection using Convolutional Neural Networks, Georgia Tech Research Institute (GTRI) , 2016
- [7] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Praseon Goyal, End to End Learning for Self-Driving Cars, arXiv:1604.07316v1 [cs.CV] 2016
- [8] Soonmin Hwang, Namil Kim\*, Yookyung Choi, Seokju Lee and In So Kweon, Fast Multiple Objects Detection and Tracking, Fusing Color Camera and 3D LIDAR for Intelligent Vehicles, Korea Advanced Institute of Science and Technology, Ubiquitous Robots and Ambient Intelligence (URAI), 13th International Conference



on, 2016

- [9] Ross Girshick, Microsoft Research, Fast R-CNN Object detection with Caffe, ICCV 2015
- [10] 로버트 라가니에, OpenCV를 활용한 컴퓨터 비전 프로그래밍 기본 영상 처리부터 고급 컴퓨터 비전, acorn+PACKT, 2015
- [11] Bo Li, 3D Fully Convolutional Network for Vehicle Detection in Point Cloud, arXiv:1611.08069v2 [cs.CV] 2017
- [12] J. R. R. Uijings et al, Selective Search for Object Recognition, IJCV13, 2013
- [13] L. Zitnick et al, Edge boxes: Locating Object Proposals from Edges, ECCV14, 2014
- [14] R. Girshick et al, Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation, CVPR14, 2014

## ABSTRACT

### Object detection and visualization using CNN image analysis and Lidar point cloud for distance measurement

In order to develop ADAS for autonomous driving assistance function, it is required that integrated analysis results the massive data obtained from various sensors. Since 2012, massive data analysis using high-performance GPUs has become possible, and high-performance GPU system-based sensor data analysis research has also introduced to the automotive industry.

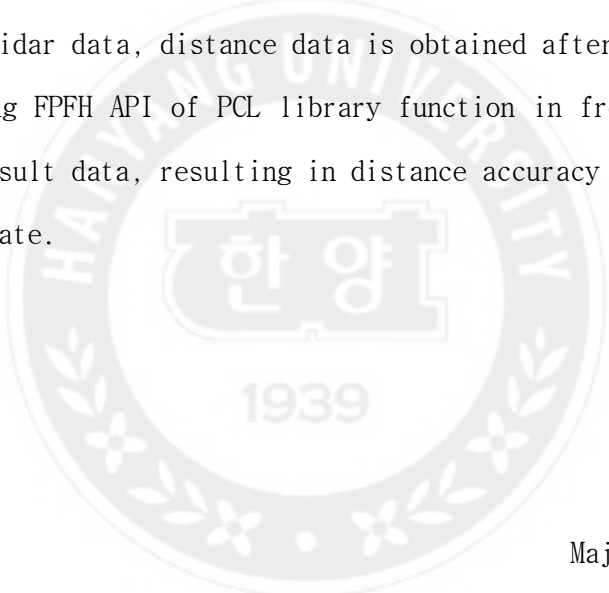
In this study, we developed the R & D software to display the distance between the vehicle and the surrounding objects in real time during the driving after collecting the camera image and Lidar data using the GPU based autonomous driving development platform.

The sensor analysis test consisted of six cameras and four lidars. The camera data is analyzed to detect object by DriveNet which is CNN-based object recognition model. The 3D point cloud data of four lidars are integrated into the 3D transformation and rotation process according to each direction and then obtain distance data.

The test results were calculated by comparing the distance based on the camera data and the distance based on the lidar data based on the data of the D-GPS which is the distance calculating device by the actual vehicle communication.

In the case of camera data, the target vehicle detected by DriveNet based on the front camera is detected as 24 fps per channel. As a result of calibration, the accuracy of the distance from the object based on image data analysis is  $\pm 0.8$  m in the running state.

In case of Lidar data, distance data is obtained after detecting the object by using FPFH API of PCL library function in front of 2 Lidar integration result data, resulting in distance accuracy of  $\pm 0.5$  m in the running state.

The seal of Hanyang University is a circular emblem. It features a central shield-like shape with the Korean characters '한양' (Hanyang) inside. Below the shield is the year '1939'. The entire seal is surrounded by a laurel wreath. The words 'HANYANG UNIVERSITY' are written in a circular path around the seal.

Kim, Hyejin  
Major in Computer  
Science and Engineering  
The Graduate School of  
Engineering  
Hanyang University

## 감사의 글

컴퓨터 공학 전공으로 좋은 논문을 꼭 써보고 싶다는 열망 하나로 입학을 하였지만 강의와 개발 업무로 야근과 주말근무가 많은 터라 대학원 졸업은 어려운 처지였던 제가 이렇게 논문을 완성하게 된 것은 조인휘 교수님의 차근차근 단계적으로 끝까지 끌어주신 좋은 지도와 103기 동기들의 도움 덕분입니다. 이렇게 감사의 글이라는 자리에서 깊은 감사를 드립니다.

같이 시작해서 언제나 좋은 일 곳은 일 같이 하며 누구보다도 도움을 많이 주었던 최광섭 기장, 채수정 총무 외 태영이, 진홍이, 영환이, 영탁이, 성민이, 영민이, 형민이, 람, 경락이와 뒤에 합류한 민희, 승현씨 현욱씨도 모두 감사합니다.

그리고 조직에 도움이 되겠다는 신념 하나로 시작한 저에게 대학원 진학 결심부터 오늘까지 믿음과 지원을 주신 한컴MDS의 오형관 상무님과 박성호 팀장님, 김종현 팀장님 그리고 같이 프로젝트 한다고 고생 많으신 김동규 책임님에게 감사드립니다.

2018년 2월

김혜진

## 연구 윤리 서약서

본인은 한양대학교 대학원생으로서 이 학위논문 작성 과정에서 다음과 같이 연구 윤리의 기본 원칙을 준수하였음을 서약합니다.

첫째, 지도교수의 지도를 받아 정직하고 엄정한 연구를 수행하여 학위논문을 작성한다.

둘째, 논문 작성시 위조, 변조, 표절 등 학문적 진실성을 훼손하는 어떤 연구 부정행위도 하지 않는다.

셋째, 논문 작성시 논문유사도 검증시스템 "카피킬러"등을 거쳐야 한다.

2017년12월14일

학위명 : 석사

학과 : 전기 · 전자 · 컴퓨터공학과

지도교수 : 조인휘

성명 : 김혜진

(서명)

한 양 대 학 교 공 학 대 학 원 장 귀 하

## Declaration of Ethical Conduct in Research

I, as a graduate student of Hanyang University, hereby declare that I have abided by the following Code of Research Ethics while writing this dissertation thesis, during my degree program.

"First, I have strived to be honest in my conduct, to produce valid and reliable research conforming with the guidance of my thesis supervisor, and I affirm that my thesis contains honest, fair and reasonable conclusions based on my own careful research under the guidance of my thesis supervisor.

Second, I have not committed any acts that may discredit or damage the credibility of my research. These include, but are not limited to : falsification, distortion of research findings or plagiarism.

Third, I need to go through with Copykiller Program(Internet-based Plagiarism-prevention service) before submitting a thesis."

DECEMBER 14, 2017

Degree : Master  
Department : DEPARTMENT OF ELECTRICAL ENGINEERING AND COMPUTER SCIENCE  
Thesis Supervisor : INWHEE JOE  
Name : KIM HYE JIN

(Signature) 