



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 원 저작자를 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 권리와 책임은 위의 내용에 의하여 영향을 받지 않습니다.

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

[Disclaimer](#)



석사학위논문

DBSCAN을 이용한
로봇 환경에서의 3차원 객체 검출 및 추적
3D Object Detection
and Tracking for a Robot Platform
Using DBSCAN

김 민 육

한양대학교 대학원

2023년 2월

석사학위논문

DBSCAN을 이용한
로봇 환경에서의 3차원 객체 검출 및 추적
3D Object Detection
and Tracking for a Robot Platform
Using DBSCAN

지도교수 최준원

이 논문을 공학 석사학위논문으로 제출합니다.

2023년 2월

한양대학교 대학원

인공지능학과

김민욱

이 논문을 김민욱의 석사학위 논문으로 인준함

2023년 2월

심사위원장 : 문준



심사위원 : 이형철



심사위원 : 최준원



한양대학교 대학원

차 례

차 례	i
그림 차례	iv
표 차례	vi
국문요지	vii
제1장 서 론	1
1.1 연구의 필요성	1
1.2 연구의 목표	3
1.3 논문의 구성	3
제2장 이론적 배경	4
2.1 2차원 객체 검출	4
2.1.1 R-CNN	5
2.1.2 Fast R-CNN	6
2.1.3 Faster R-CNN	7
2.2 YOLO	9
2.2.1 YOLOv1	9
2.2.2 YOLOv2	11
2.2.3 YOLOv3	12
2.3 3차원 객체 검출	15

2.3.1 센서(Sensor)	15
2.3.2 PointNet	18
2.3.3 VoxelNet	20
2.3.4 Pointpainting	22
2.4 군집화 알고리즘(Clustering algorithm)	23
2.4.1 K-평균 알고리즘(K-means clustering algorithm)	23
2.4.2 Mean-Shift 알고리즘(Mean-Shift Clustering)	24
2.4.3 DBSCAN 알고리즘	24
2.4.4 Expectation-Maximization 알고리즘	26
2.5 카메라 센서 보정(Camera-LiDAR Calibration)	28
 제3장 제안하는 기법	30
3.1 개요	30
3.2 제안하는 네트워크의 전체 구조	30
3.2.1 제안하는 센서 퓨전 알고리즘(Sensor fusion Algorithm) ·	36
3.3 학습 실험 환경	37
3.3.1 Aihub 인도보행 영상 데이터셋	38
3.3.2 하이퍼-파라미터 튜닝(Hyper-Parameter Tunning)	39
3.3.3 군집화 방식에 따른 센서 퓨전 알고리즘의 속도 비교	40
 제4장 실험 결과	41
4.1 개요	41
4.2 실험 환경	41

4.3 카메라-라이다 보정(Camera-LiDAR Calibration) 결과	42
4.4 제안하는 알고리즘의 실험결과	43
4.4.1 검출 모델에 따른 추론 속도 결과	43
4.4.2 센서 퓨전 알고리즘의 정량적 실험 결과	46
4.4.3 센서 퓨전 알고리즘의 정성적 거리 추정 결과	47
4.4.4 센서 퓨전 알고리즘의 정성적 2차원 객체 검출 및 추적 결과	49
제5장 결론	51
참고문헌	53
ABSTRACT	58
감사의 글	60

그림 차례

그림 1. 1-stage detector와 2-stage detector 모델의 비교	4
그림 2. R-CNN의 동작 과정	5
그림 3. Fast R-CNN의 전체 구조	7
그림 4. RPN(Region Proposal Network)의 구조	8
그림 5. YOLO의 전개과정	9
그림 6. DarkNet의 구조	10
그림 7. 앵커 상자(Anchor box)를 이용한 위치 예측 방식	11
그림 8. Feature Pyramid Network의 구조	14
그림 9. 카메라, 라이다, 레이더 센서	15
그림 10. PointNet의 구조	18
그림 11. (a) : 입력 변형(Input Transform)과 (b) : 특징 변형(Feature Transform)	18
그림 12. 복셀 특징 추출(Voxel Feature Encoding)의 과정	20
그림 13. Pointpainting의 전개 과정	22
그림 14. DBSCAN 알고리즘의 예시 (a) : DBSCAN 클러스터 조건이 성립되는 경우 (b) : DBSCAN 클러스터 조건이 성립되지 않는 경우	25
그림 15. Expectation-Maximization 알고리즘의 순서	26
그림 16. 카메라 좌표계와 월드 좌표계의 상관도	28
그림 17. 제안하는 알고리즘의 네트워크의 구조	32

그림 18. SORT 알고리즘	33
그림 19. IoU Matching 과정	34
그림 20. IoU(Intersection of Union)	35
그림 21. 제안하는 센서 퓨전 알고리즘(Sensor fusion Algorithm)의 순서도	36
그림 22. Aihub 인도보행 영상 데이터 셋	38
그림 23. 보정 작업에 사용된 체스판(Chessboard)	42
그림 24. 카메라-라이다 보정 과정 1, 샘플링된 체스판의 범선벡터 추정	43
그림 25. 제안하는 알고리즘의 정성적 거리 추정 결과	47
그림 26. 제안하는 알고리즘의 정성적 거리 추정 결과 2	48
그림 27. (a), (b), (c), (d), (e), (f), (g), (h) 시간 순에 따른 알고리즘의 정성적 객체 검출 및 추적 결과	50

표 차례

표 1. DarkNet-53의 구조	13
표 2. 카메라, 라이다, 레이더의 특징	16
표 3. 학습 실험환경	37
표 4. 하이퍼-파라미터에 따른 mAP 비교	39
표 5. 군집화 방식에 따른 센서 퓨전 알고리즘의 속도 비교	40
표 6. 추론 실험환경	41
표 7. 카메라-라이다 보정 결과	44
표 8. 검출 모델에 따른 추론 속도 비교	45
표 9. 센서 퓨전 알고리즘의 정량적 추론속도 비교 실험결과	46

국문요지

자율주행 알고리즘에서 센서를 사용한 주변 환경의 인지는 사람의 눈의 역할을 하는 만큼 중요한 과업(task)이라고 할 수 있다. 표지판, 신호등, 정지선과 같은 정적인 객체와 보행자, 자동차와 같은 동적인 객체의 인지를 통해 얻은 정보는 이후에 동작 되는 판단, 제어 알고리즘에 핵심 단서로 사용된다.

딥러닝 기반의 인지 알고리즘은 수동적인 네트워크의 설계로부터 진화되어 왔다. 데이터 셋을 이용한 학습(learning) 방식의 훈련은 컴퓨터의 발달, 센서 성능의 향상이 촉매 역할을 하여 자율주행 이외에 로봇, 의료영상 등과 같이 여러 분야에 응용되고 있다. 자동차, 로봇과 같은 플랫폼에서 사용 가능한 제원은 제한적이기 때문에 센서 선택과 신속 정확한 알고리즘 설계는 가장 중요한 해결과제라고 할 수 있다.

라이다는 카메라, 레이더와 같은 센서들에 비해 비싸다는 단점이 있다. 하지만, 최근 여러 라이다 업체들의 경쟁과 라이다 센싱 기술의 발달로 저렴해지고 있으며, 다른 센서들에 비해 라이다를 통해 습득되는 포인트 클라우드의 정확한 3차원 공간상의 위치 좌표는 매력적인 선택지이다.

이미지를 이용한 2차원 객체 검출은 라이다를 이용한 객체 검출에 비해 색감, 질감과 같은 의미론적 정보를 제공할 수 있다는 장점이 있다. 이에 본 논문에서는 2차원 객체 검출 및 추적을 기반으로 하는 센서 퓨전 알고리즘을 이용한 3차원 객체 검출 및 추적 알고리즘에 대해 소개한다.

제안하는 알고리즘은 YOLO [1]기반의 빠른 2차원 객체 검출 결과를 이용하여 SORT 기반의 추적으로 실시간 추론 성능을 개선한다. 딥러닝 기반의 객체 검출 모델을 훈련시키기 위해 Aihub 인도 보행 영상 데이터셋 [2]으로

학습시켜 파라미터 튜닝을 통한 최적의 학습지(checkpoint)를 사용한다.

검출된 객체의 경계상자(bounding box)는 객체 외부에 할당되는 포인트 클라우드를 제외해주는 필터 역할을 하며, 필터링된 점들은 DBSCAN 알고리즘에 의해 클러스터화 된다.

경계 상자(bounding box) 내부에 존재하는 클러스터 중 라이다 좌표계의 원점과 가장 가까이 있는 클러스터는 객체를 나타내는 점들의 클러스터이며, 이를 평균 풀링하여 3차원 공간 상의 객체를 나타는 위치 좌표로 사용하게 된다. 알고리즘의 실효성을 판단하기 위해 모델의 결과를 실측 거리와 비교하여 그 정확도를 계산하고, 정성적 평가를 통한 2차원 객체 및 추적 성능을 평가 한다.



제1장 서 론

1.1 연구의 필요성

최근 COVID-19로 인해 외식 문화가 줄어들고, 그로 인해 배달 시장이 급 속도로 커졌지만, 인건비 상승에 따른 판매자와 소비자의 물가 부담은 나날이 심화되고 있는 상황이다. 이를 해결할 방법으로 자율주행 배달로봇이 조명되었고 시장규모 또한 나날이 성장하고 있다. 시장조사업체 럭스 리서치에 따르면, 2030년에는 전체 배송물량 대비 배달로봇의 비중은 20%를 차지하고, 그 규모는 50조원에 달할 전망이다. 배달 생태계의 혁신이라고 생각되는 배달로봇이지만 경쟁력있는 서비스를 제공하기 위해서는 더 많은 발전이 필요한 상황이다. 그 예로, 최근 샌드박스 규제완화를 통해 실증을 진행중인 배달로봇들은 고객의 거주지가 아닌 거주지 근처의 장소에서 배달물을 전송하게 되는데, 이는 보통의 로봇들은 계단을 오르지 못하고, 건물 내부로 들어가게 되었을 때, 현관문과 엘리베이터와 같은 장애물요소를 지나는 과정이 어렵기 때문이다. 장애물의 종류에 따라 로봇의 행동 메커니즘이 달라야 하기 때문에 그 장애물이 무엇인지 잘 판단하는 것이 중요한 과제라고 할 수 있다.

배달로봇은 자율주행 자동차와 비슷한 구조를 가지기 때문에, 같은 메커니즘을 통해서 주행을 하게 되는데, 로봇에 탑재된 카메라, 라이다 그리고 레이더와 같은 센서들을 통해 습득된 데이터를 이용해서 주변 환경을 인지하고, 습득한 정보를 근거하여, 스스로 주행하게 된다. 주변 환경의 정보를 특정화하

는 방법에는 객체를 분류(classification)하고, 위치(localization)를 추정하는 과정을 포함한 객체 검출(object detection)이 선행적으로 진행된다.

이후, 검출 결과를 사용하여 객체 추적(object tracking) 그리고 경로 예측(path prediction) 알고리즘이 순차적으로 진행되고 각 알고리즘의 출력은 제어 파트로 전송되어 사용되게 된다. 즉, 객체 검출과 추적의 결과는 후행 알고리즘에 큰 영향을 미치는 알고리즘이며, 정확하고 빠른 객체 검출, 추적은 전체적인 자율주행 완성도를 높일 것으로 기대할 수 있다. 딥러닝을 이용한 알고리즘은 많은 GPU 메모리를 요구한다. 배달 로봇과 같이 자동차에 비해 크기가 작고 생산가를 고려했을 때, 비교적 낮은 사양의 GPU가 사용되기 때문에 주어진 자원에 맞는 딥러닝 모델을 개발하는 것이 중요하다. 널리 사용되고 있는 센서 중 라이다는 측정 지점으로부터 객체까지의 거리에 대한 정보를 포함하고 있기 때문에 매우 중요한 역할을 수행한다고 할 수 있다. 하지만 포인트 클라우드는 3차원 공간 상에 존재하는 무질서한 점들로 구성되어있고, 처리해서 모델 훈련(model training) 및 추론(inference)에 사용하기 위해서는 많은 연산량을 필요로 하는데, 이 때 연산량을 줄이는 작업을 필요로 하고, 그 해법으로 다음 소개하는 모델로 제시한다.

1.2 연구의 목표

자동차나 로봇에서의 자율주행은 딥러닝(deep learning)기반의 환경인지를 통해 수집한 정보를 단서로 차량을 제어하게 된다. 딥러닝 방식의 알고리즘은 많은 연산량을 동반하기 때문에, 알고리즘을 개발할 때에는 탑재될 모델의 연산 능력을 고려해 실시간 작동 가능성을 염두해야 한다. 본 논문에서 수행하게 될 환경인지 중 3차원 객체 검출 및 추적은 일반적으로 LiDAR(Laser Imaging Detection And Ranging)의 포인트 클라우드를 이용하지만 3차원 공간 상에 흩뿌려진 많은 점들을 처리하여 많은 연산이 필요하다. 이를 해결하기 위해 카메라를 이용한 2차원 객체 및 추적을 수행하고, 그 결과를 이용해 포인트 클라우드(point cloud)와의 센서 퓨전(sensor fusion)을 통해, 3차원 공간상의 위치로 확장하는 알고리즘을 소개한다

1.3 논문의 구성

본 논문은 다음과 같이 구성된다. 2장에서는 2차원 객체 검출, 3차원 객체 검출에 대한 설명과 모델을 소개하고, 클러스터링 알고리즘에 대한 동작 설명과 대표적인 메커니즘의 동작 과정을 설명한다. 3장에서는 제안하는 알고리즘의 전체적인 구조와 딥러닝 네트워크의 학습 및 데이터셋에 대해 소개하며, 4장에서는 정량적 정성적 비교를 통해 알고리즘의 효능을 입증한다. 마지막으로 5장 결론으로 논문을 마친다.

제2장 이론적 배경

2.1 2차원 객체 검출

2차원 객체 검출은 이미지가 입력되었을 때, 이미지에 등장하는 객체의 종류를 분류(classification)하고, 지역화(localization)를 수행한다. 객체 검출 모델의 구조는 네트워크 흐름의 방법에 따라 1-stage detector와 2-stage detector로 나뉜다.

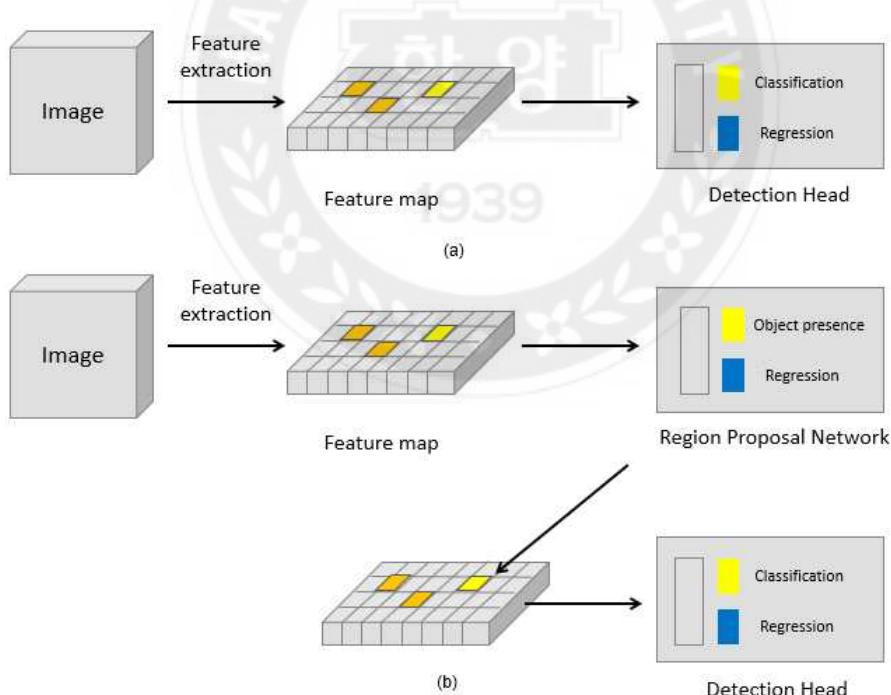


그림 1. 1-stage detector와 2-stage detector 모델의 비교

먼저, 1-stage detector는 이미지가 입력되면 컨볼루션(convolution)기반의 백본 네트워크(backbone network)를 이용해 이미지 특징 맵을 추출한다. 이후에, 추출된 특징 맵은 검출 헤드(detection head)에서 분류 및 회귀를 수행한다. 대표적으로 YOLO [1], RetinaNet [3]등이 있으며, 2-stage detector에 비해 비교적 빠르며, 정확도가 낮다는 특성이 있다.

2-stage detector는 1-stage와 비슷하게 진행되지만 추출된 특징 맵을 Region Proposal Network(RPN)에 사용된다. 기존에 1-stage에서는 이미지 내의 영역을 다양한 크기와 가로-세로 비율(aspect ratio)의 창(window)으로 탐색하여 시간이 오래 걸리지만 이를, Selective search [4]로 해결하였다

2.1.1 R-CNN

대표적인 2-stage detector 중 하나인 R-CNN[3]은 딥러닝(deep-learning)이 적용된 최초의 객체 검출 모델이며, 다음 그림 2는 R-CNN의 동작 순서이다. 먼저, 이미지가 입력되게 되면, Selective search 알고리즘을 통해서 객체가 존

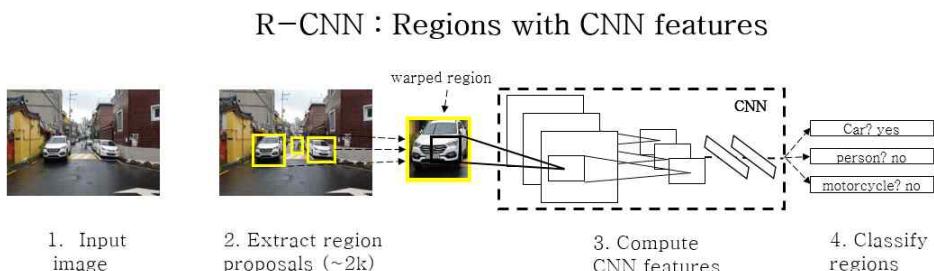


그림 2. R-CNN의 동작 과정

재할 가능성이 있는 후보 영역(region proposal)을 약 2,000개를 추출하고, 각기 다른 크기를 가진 특징 벡터를 하나의 통일된 크기로 일치시켜주기 위해, 그 크기를 227×227 으로 warp시켜준다. warp된 후보 영역(region proposal)을 학습된 AlexNet[4]에 입력하여, 2000×4096 크기의 특징 벡터를 추출한다. 추출된 특징 벡터는 각각 선형 SVM(Linear-SVM)과 경계 상자 회귀 모델(bounding box regressor)을 통하여, 선형 SVM(linear-SVM)에서는 입력된 특징 벡터를 분류하여, 해당 벡터가 어떤 물체 클래스에 속하는지 판단하게 된다. R-CNN은 후보 영역(region proposal)을 추출하게 되면, 물체의 대략적인 위치를 추정할 수 있다. 하지만 그 정확도가 매우 낮기 때문에, 특징 맵을 경계 상자 회귀 모델(bounding box regressor)에 사용하게 된다.

2.1.2 Fast R-CNN

Fast R-CNN[4]은 이전 모델인 R-CNN의 한계점을 보완했는데, 기존 R-CNN에서는 각 ROI(Region of Interest)를 CNN(Convolution Neural Network)을 이용해 연산을 하기 때문에, 속도가 느리고, 2-stage detector이기 때문에 모델을 end-to-end 학습을 하지 못한다는 단점이 있었다. 이를 해결하기 위해서 Fast-RCNN에서는 ROI-pooling과 객체의 분류와 회귀를 한번에 학습시키는 방법을 제안한다. 그림 3은 Fast R-CNN의 동작과정을 나타낸다. 먼저, Fast R-CNN은 기존 R-CNN과 마찬가지로 Selective search를 통해서 ROI(Region of Interest)를 다음, ROI(Region of Interest)를 특징 맵 크기에 맞춰서 정사영시킨 ROI에 대해서 ROI Pooling을 진행하여 고정된 크기의 특징

벡터를 얻는다. [4]

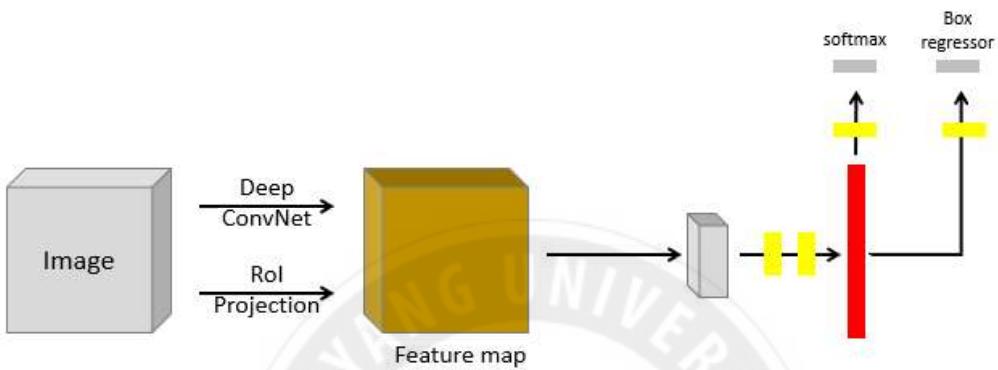


그림 3. Fast R-CNN의 전체 구조

2.2 Faster R-CNN

Faster R-CNN[5]은 선행 연구인 Fast R-CNN의 실시간 동작을 위해 제안된 모델이다. 기존 Selective search 알고리즘을 통해 계산하게 되는 Region Proposal 단계를 논문에서 제안하는 RPN(Region Proposal Network)로 대체하였다. 그림 4는 RPN의 동작 구조를 나타낸다. RPN(Region Proposal Network)의 입력 값은 이전 Convolution layer에서 출력된 특징 맵이다. 빨간색 상자로 나타낸 $n \times n$ 창(Window)은 특징 맵 위에서 움직이며 사전 정의된 각 앵커상

자에 객체가 존재할 확률과 객체의 위치를 나타내는 경계 상자의 중심점과 가로-세로의 길이를 계산하게 된다.

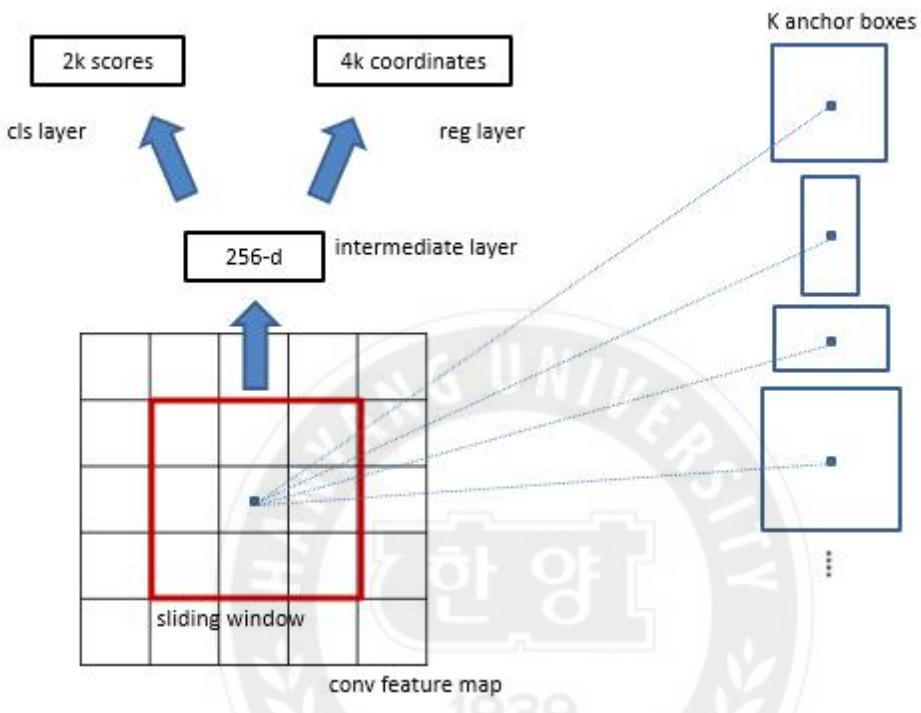


그림 4. RPN(Region Proposal Network)의 구조

Faster R-CNN에서는 3가지의 크기(scale)과 3가지의 가로-세로비(aspect ratio)를 사용해서 총 9개의 앵커 상자를 사용하였다. 기존 Fast R-CNN에서 사용했던 Selective search 알고리즘은 CPU를 사용하였지만, RPN으로 Region Proposal 계산함으로써 GPU를 이용한 연산이 가능해져, 연산속도가 향상되었다. [5]

2.2 YOLO

YOLO(You Only Look Once)[6]는 그림 5와 같이 이미지 전체에 대해서 하나의 신경망이 한 번의 계산을 통해, 분류(classification)과 지역화(localization)을 예측한다. 전체 파이프 라인이 하나의 신경망으로 구성되어 있으므로 end-to-end 방식이다. 이후 절에서는 YOLOv1에서 YOLOv2[7], YOLOv3[8]까지의 발전과정을 소개한다.

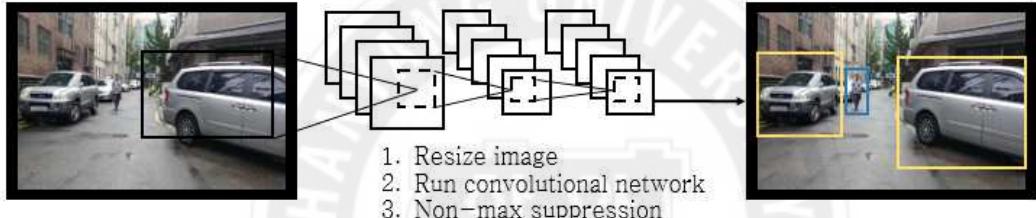


그림 5. YOLO의 전개과정

2.2.1 YOLOv1

YOLOv1[6]은 2-stage detector의 단점인 매우 느린 동작 속도를 해결하기 위해 제안된 방법이다. 2-stage detector와 달리 분류(classification)과 상자 회귀(box regression)을 한번에 진행하므로, 동작 속도가 빠르지만 정확도가 떨어진다는 단점이 있다. 반면에, 배경 이미지에 객체가 존재한다고 예측하는 배경 오차(background error)가 작아진다는 특징이 있는데, 이는 YOLOv1은 이미지 전체를 사용하지만, 비교 모델인 Fast R-CNN은 이미지를 작게 쪼개어 Selective search 알고리즘을 사용하기 때문에, 이미지를 볼 때 제한된 영역을

사용하는데 이는 배경을 객체로 예측한다는 단점이 있다. 이를 해결하기 위해, YOLOv1에서는 이미지 전체를 사용하였다. 모델은 이미지 특징 맵 추출과정을 거쳐 검출된 객체에 대한 경계 상자(bounding box)의 위치, 클래스를 예측한다.

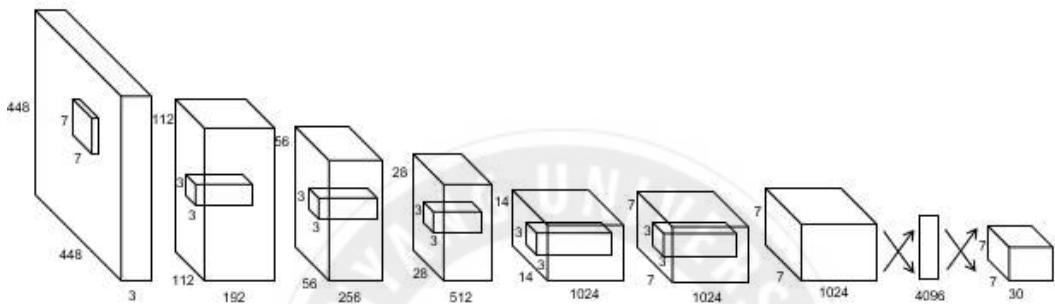


그림 6. DarkNet의 구조

그림 6은 DarkNet의 구조이다. 448×448 크기의 이미지를 입력 받아, 20개의 Convolutional layer를 거쳐 특징 맵을 계산한다. 검출 헤드(detection head)는 2개의 완전 연결 층(Fully-Connected Layer) 이후에, 3×3 Convolutional layer를 이용하여 $7 \times 7 \times 30$ 의 텐서(tensor)를 출력한다.

$7 \times 7 \times 30$ 의 텐서(tensor) 중 7×7 의 한 셀(cell)은 입력된 이미지 공간 중 64×64 를 나타내며 채널(channel)에는 검출된 객체의 경계 상자(bounding box) 2개에 대한 정보 즉, 상자의 중심좌표, 가로와 세로의 길이 값과 20개의 클래스(class)에 대한 클래스 점수(class score)를 포함하고 있다. [6]

2.2.2 YOLOv2

YOLOv2 [7]는 YOLOv1을 [6]계승한 모델이다. 실시간 동작을 위해 제안된 YOLOv1은 정확도가 낮다는 문제점이 있었는데 이를 보완한 YOLOv2에서는 기존 DarkNet에서 사용하던 완전 연결 층(Fully-Connected Layer) 대신 앵커 상자(anchor box)를 사용하였다. 이는 YOLOv1에서는 Convolution layer와 완전 연결 층(Fully-Connected Layer)만을 이용해 객체의 정확한 위치를 찾게 되어 난이도가 높은 반면, 사전 정의된 앵커 상자를 기준으로 객체의 위치를 찾기 때문에 더 쉽게 찾을 수 있다. 또한, YOLOv1의 최종 출력값 $7 \times 7 \times 30$ 의 텐서(tensor)는 최대 49개의 객체를 검출할 수 있는 반면 사전 정의된 앵커 상자에 의해 그보다 더 많이 객체를 검출할 수 있다.

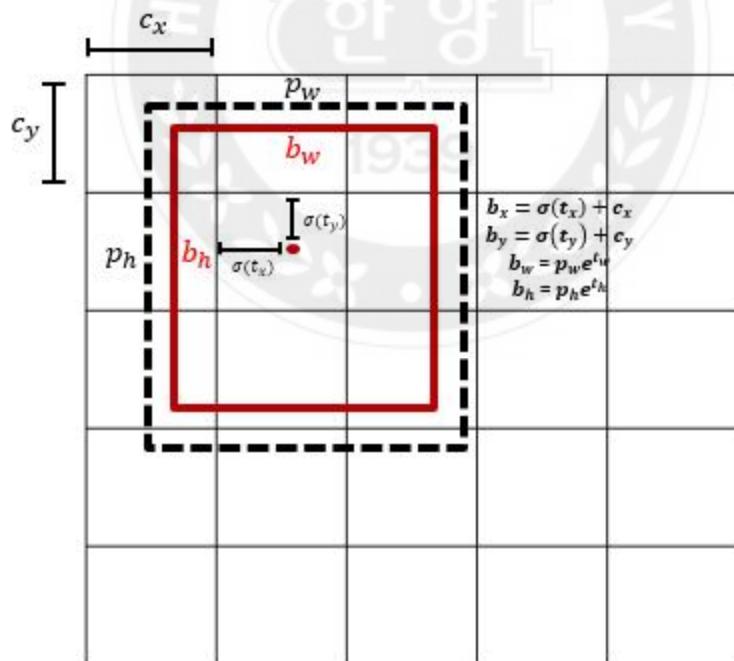


그림 7. 앵커 상자(Anchor box)를 이용한 위치 예측 방식

위 그림은 앵커 상자(anchor box)를 이용해 위치를 예측하는 방식에 대한 설명이다. 모델은 b_x , b_y , b_w , b_h 을 예측하게 되며, 이 값들은 앵커 상자(anchor box)와 예측한 객체 상자의 가로-세로, 높이-너비의 차이이다.

2.2.3 YOLOv3

YOLOv3 [8]는 YOLOv2 [7]에서 개선된 방식이다. 2-stage detector에 비해 1-stage detector의 정확도가 낮기 때문에 정확도를 올리려는 시도를 하였다. 기존 YOLOv2에서 사용했던 DarkNet 백본 네트워크 대신 DarkNet-53을 제안한다. DarkNet-53은 이미지 특징 맵을 더 많은 수의 Convolutional layer를 이용해 계산하게 되는데, 딥러닝 네트워크의 깊이가 깊어질수록 학습과정에서 발생되는 local minima에 수렴되는 문제가 생기는 것을 방지하기 위해 잔차 연결(residual connection)을 backbone 내부에 쌓았다. 추가로 구성된 Convolution layer는 feature pyramid를 생성하기 위한 이미지 특징 맵을 출력하여 두 번의 상향 샘플링(up-sampling)을 거치게 된다. 최종 이미지 특징 맵과 이전 2개의 이미지 특징 맵은 상향 샘플링된 이미지 특징 맵과 합치게 된다.

표 1. DarkNet-53의 구조

Type	Filters	Size	Output
Convolutional	32	3×3	256×256
Convolutional	64	$3 \times 3/2$	128×128
Convolutional	32	1×1	
Convolutional	64	3×3	
Residual			128×128
Convolutional	128	$3 \times 3/2$	64×64
Convolutional	64	1×1	
Convolutional	128	3×3	
Residual			64×64
Convolutional	256	$3 \times 3/2$	32×32
Convolutional	128	1×1	
Convolutional	256	3×3	
Residual			32×32
Convolutional	512	3×3	16×16
Convolutional	256	1×1	
Convolutional	512	3×3	
Residual			16×16
Convolutional	1024	$3 \times 3/2$	8×8
Convolutional	512	1×1	
Convolutional	1024	3×3	
Residual			8×8
Avgpool		Global	
Connected		1000	
Softmax			

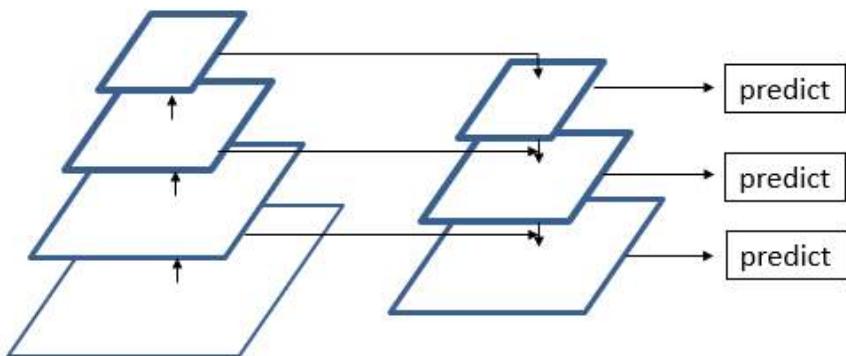


그림 8. Feature Pyramid Network의 구조

그림 10은 Feature Pyramid Network의 구조이다. 피라미드 형식의 네트워크를 구성하여 Convolutional 방식으로 비례된 크기의 이미지 특징 맵을 다중 레벨로 출력하여 다중 해상도 특징 맵(multi-scale feature map)을 사용해 여러 해상도의 이미지 특징 맵을 사용했으며, 검출 헤드(detection head)에서는 소프트 맥스(softmax) 대신 시그모이드(sigmoid)와 교차 엔트로피(cross-entropy)가 합쳐진 이진 교차 엔트로피(binary cross entropy)를 사용했다.

2.3 3차원 객체 검출

앞서 소개한 2차원 객체 검출(2d object detection)과는 3차원 객체 검출(3d object detection)에서는 2차원이 아닌 3차원 공간상에서의 위치를 추정한다. 이미지를 이용한 2차원 객체 검출과는 달리 3차원 객체 검출은 레이다, 라이다와 같이 거리 정보가 포함된 다양한 센서를 사용해 객체를 검출한다. 다음 절에서는 대표적인 3차원 객체 검출 모델 소개에 앞서 사용되는 센서들에 대한 특성에 대해 소개한다.

2.3.1 센서(Sensor)

2차원 객체 검출에서 사용되는 데이터는 그림11.(a)와 같이 카메라로 수집된 이미지를 사용했는데, 검출 결과에는 카메라로부터 객체까지 거리에 대한 정보를 포함하지 않는다. 이를 위해, 그림 (b), (c)의 라이다, 레이더와 같이 거리 측정이 가능한 센서들이 자율주행 목적의 연구나 개발에 많이 사용되었다.

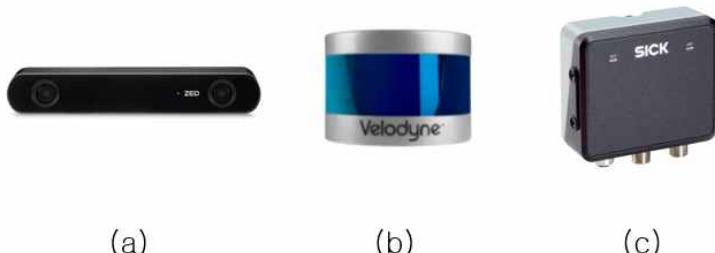


그림 9. 카메라, 라이다, 레이더 센서

아래 표 2는 각 센서들의 대한 특징을 서술한다.

표 2. 카메라, 라이다, 레이더의 특징

	카메라	라이다	레이더
방식	빛을 디지털 신호로 변환	빛을 발사하여 다시 되돌아오는 시간을 측정	전자기파를 발사하여 되돌아오는 전자파를 분석하여 거리를 측정
장점	1. 높은 해상도를 가짐 2. 색, 질감에 대한 정보를 가짐 3. 낮은 가격	1. 조도 변화에 대해 강건함 2. 거리에 대한 정보를 포함	1. 날씨 변화에 대해 강건함. 2. 낮은 가격
단점	1. 연산량이 큼 2. 거리, 속도에 대한 정보가 없음 3. 날씨에 민감함	1. 높은 가격 2. 습도에 민감함	1. 낮은 해상도를 가짐

이미지는 라이다, 레이더에 비해서 색과 질감과 같은 의미론적 정보를 포함하고 있기 때문에, 2차원 객체 검출에 주로 사용되지만, 이미지를 이용한 3차원 객체 검출에서 이미지는 거리에 대한 정보를 가지고 있지 않기 때문에 이미지를 이용한 3차원 객체 검출은 그 정확도가 낮고, 주로 단일 라이다 기반의 객체 검출 모델에 더해져 사용된다. 단일 라이다 3차원 객체 검출 모델의 성능이 높은 이유는 라이다로 취득되는 포인트 클라우드의 정확한 위치 정보가 매우 강력하다는 증거이다. 레이더는 라이다에 비해 가격이 저렴한 반면 이미지가 제공하지 못하는 거리 정보를 제공한다는 장점이 있다. 가격이 저렴하기 때문에 자율주행, 배달로봇을 개발하는 여러 기업에서는 카메라, 레이더를 사용하여 주변 데이터를 취득하게 된다. 하지만, 레이더는 해상도가 낮고 잡음이 많기 때문에 이를 보상하여 사용한다.

포인트 클라우드를 이용한 3차원 인지 모델은 전처리 방식에 따라 점 기반 방식, 복셀 기반 방식으로 나뉘는데, 다음 절에서는 포인트 클라우드, 복셀, 센서 퓨전을 이용한 3차원 검출 모델에 대해 소개한다.

2.3.2 PointNet

PointNet [9]은 포인트 클라우드를 직접적으로 딥러닝 모델에 사용한 3차원 객체 분할 모델(3d object segmentation)이다. 이전 연구에서는 포인트 클라우드를 3차원 복셀(voxel)로 분할하여 사용했는데, 이는 포인트 클라우드를 복셀화하면서 생기는 양자화 오류(quantization error)를 야기했다. PointNet은 이를 해결하고자 포인트 클라우드를 바로 Neural Network에 적용하였는데, 아래 그림은 PointNet의 구조이다.

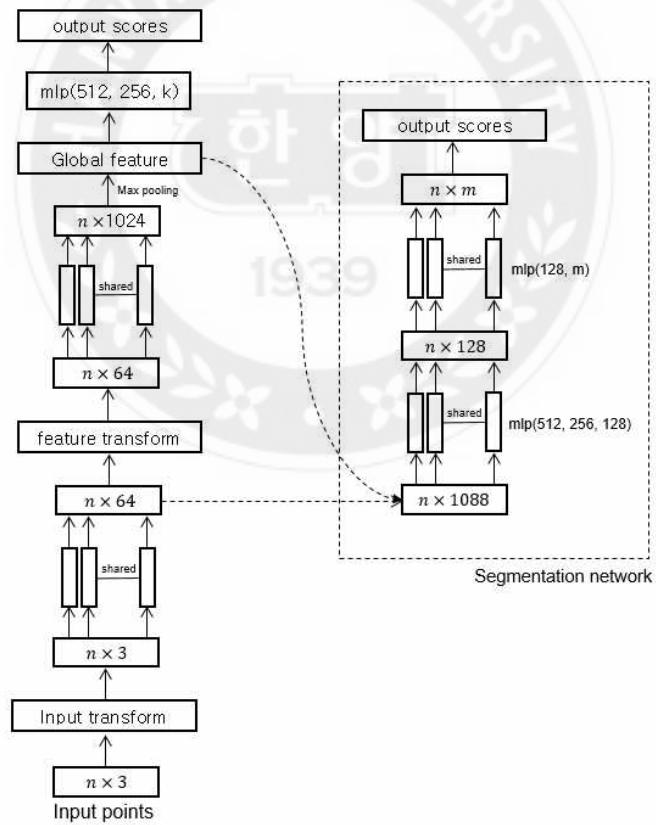


그림 10. PointNet의 구조

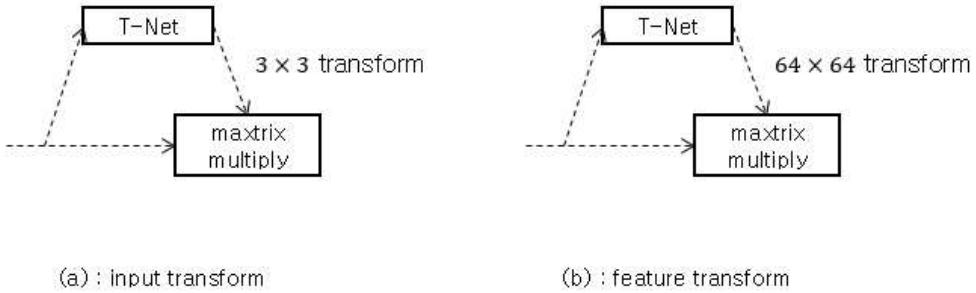


그림 11. (a) 입력 변형(Input Transform)과 (b) 특징 변형(Feature Transform)

포인트 클라우드는 이미지와 같이 정해진 형식을 가지지 않기 때문에, 네트워크에 직접적으로 사용하지 않고, 복셀이나 이미지 형태로 형식을 바꾸는 방식의 연구가 선행적으로 진행되었다. 그림11의 (a)는 입력된 포인트 클라우드를 canonical 공간으로 정렬한다. 이후에 PointNet은 포인트 클라우드의 불규칙한 형식을 최대 풀링(max pooling)을 사용해 점의 순서가 고정된 순서로 입력되지 않아도 항상 동일한 값을 출력하는 permutation invariant한 특성을 가져야 하며 rigid motion에도 invariant해야 한다. MLP(Multi-Layer Perception)을 거친 특징 맵들은 그림 11의 (b) 특징 변형을 통해 다시 canonical 공간으로 정렬된다. 정렬된 특징 맵들은 다시 MLP를 거쳐 global 특징 맵과 함께 각 점 당 특징 맵에 붙여져 사용하게 된다.

2.3.3 VoxelNet

VoxelNet [10]은 3차원 공간 상을 동일한 크기의 상자 형태로 나누어 복셀을 생성하고, 복셀 특징 맵 추출(voxel feature map)을 하는 것을 제안한다. 복셀은 $D \times H \times W$ 크기로 사전 정의되며, D는 깊이, H는 높이, W는 너비이다. 각 복셀의 내부에 있는 포인트 클라우드들은 서로 그룹화(grouping)되며, 한 복셀 내부에 있는 포인트 클라우드들을 대상으로 고정된 숫자만큼 무작위 샘플링(random sampling)을 수행한다. 이는 많은 포인트 클라우드들을 대상으로 계산했을 때, 많은 연산량을 필요로 하는데 고정된 점만 사용하여 이를 해결할 뿐만 아니라, 복셀 내부에 포인트 클라우드가 존재하지 않는 것과 존재하는 것에 대한 불균형(imbalance)문제를 해결할 수 있다고 설명한다.

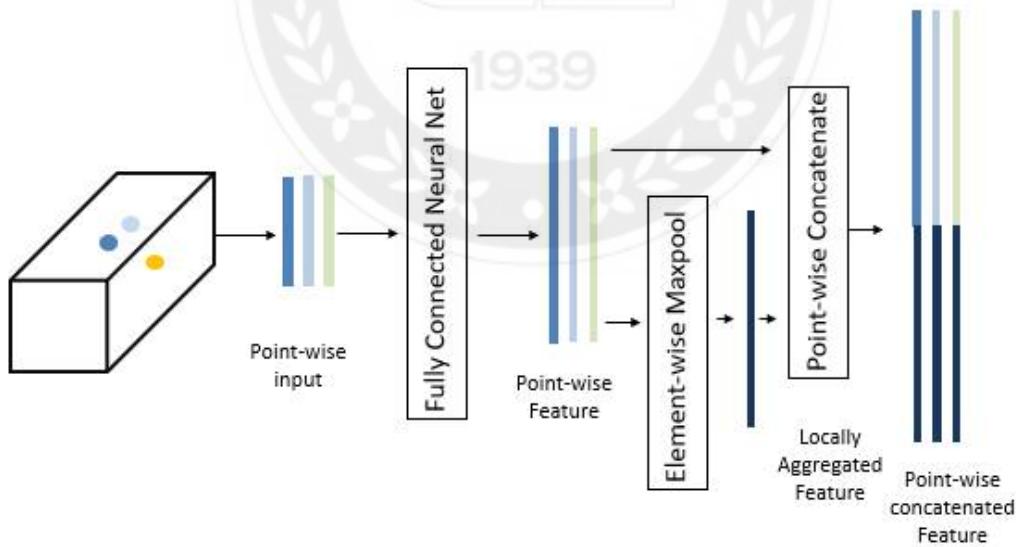


그림 12. 복셀 특징 추출(Voxel Feature Encoding)의 과정

그룹화를 거친 복셀들은 그 특징 맵 인코딩(voxel feature encoding)을 거치는데 위의 그림 12와 같이 나타낼 수 있다. 복셀 내부의 포인트 클라우드들은 포인트별로 완전 연결 층(Fully-Connected layer)를 이용해 특징 맵을 추출하게 되며 이를 포인트 당 특징 맵(point-wise feature)이라고 한다. 이후에는, 포인트 당 특징 맵 중 가장 큰 값을 선택하도록 최대 풀링(max pooling)하여 이전에 추출했던 포인트 당 특징 맵과 최대 풀링을 거친 포인트 당 특징 맵을 연결(concatenate)해준다. 그 결과, 네트워크는 지역적인(local) 특징 맵과 전체적인(global) 특징 맵의 값을 학습할 수 있다. 이후엔 생성된 복셀 특징 맵을 3차원 Convolutional middle layer로 더 고도화된 특징 맵을 학습시키는 것을 목표로 한다. 3차원 Convolutional middle layer는 3차원 Convolution, 배치-정규화 층(Batch Normalization layer) 그리고 ReLU 층으로 구성되어 있으며 검출 헤드로는 변형된 RPN(Region Proposal Network)를 사용했다.

2.3.4 Pointpainting

S Vora [11]가 제안한 Pointpainting은 카메라와 라이다에 대한 특성을 각 센서의 특징 맵이 보완하여 더 정확한 검출 예측이 가능하다고 시사한다. Pointpainting은 카메라로 취득되는 이미지를 DeepLabV3+ [12] 즉 의미 분할 네트워크(semantic segmentation network)에 입력하여 의미 분할 결과를 추출하고 포인트 클라우드를 그 위에 정사영하여 픽셀(pixel)당 클래스 스코어를 점에 매핑(mapping)하는 방식으로 각 센서의 데이터를 퓨전하는 방법을 제안한다.

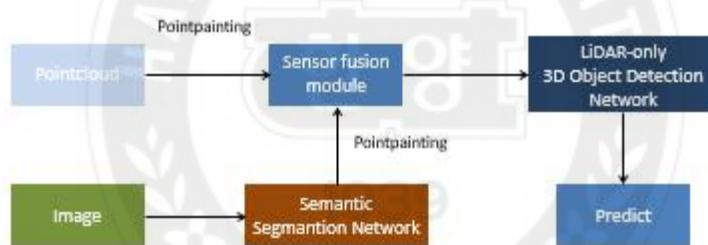


그림 13. Pointpainting의 전개 과정

각 점에 할당된 클래스 스코어는 점들의 3차원 좌표에 연결되어(concatenate) 라이다 기반 3차원 모델인 Pointpillar [24], Point R-CNN[26]을 사용해 객체를 검출한다. 그 결과 KITTI 검증 데이터 셋 기준 Point R-CNN에 비해 2.9 4% mAP 향상이 있었다. [11]

2.4 군집화 알고리즘(Clustering algorithm)

군집화란 데이터 집단을 정의해서 집단의 대표로 할 수 있는 점을 찾는 것으로 데이터 마이닝의 한 방법이다. 군집화의 결과로 데이터 집단 즉, 클러스터는 비슷한 특성을 가진 데이터들이 무리를 이루게 되며, 다른 특성을 가지면 다른 클러스터에 속해야 된다. 다음 절에서는 군집화 알고리즘의 각 종류에 대해서 설명한다.

2.4.1 K-평균 알고리즘(K-means clustering algorithm)

K-평균 알고리즘 [13]은 입력된 데이터를 k개의 클러스터로 분할하는 알고리즘으로, 각 클러스터 간 거리 차이의 분산을 최소화하는 방식으로 수행한다. d차원의 데이터 n개의 집합이 주어지면, 알고리즘은 n개의 데이터 집합들을 각 집합 내에 데이터 간의 분산을 최소로 하는 k개의 집합 S 로 분할한다. 즉, 수식으로 나타내면 μ_i 가 집합 S_i 의 중심점이라고 했을 때 각 데이터 집합 별로 중심점과 다른 점 간 거리의 제곱합을 최소로 하는 집합 S 를 찾는 것이다. [13]

2.4.2 Mean-Shift 알고리즘(Mean-Shift Clustering)

Mean-Shift 알고리즘 [14]은 데이터 간의 거리를 최소화 하는 방식으로 군집화를 진행하는 K-평균 알고리즘과 달리 중심을 밀도가 가장 높은 곳으로 이동시킨다는 점에서 다르다. 이 때, 밀도는 KDE(Kernel Density Estimation)로 계산하게 된다. KDE는 히스토그램 방법과 같은 기준에 데이터의 경계에서 불연속성이 나타난다는 문제점을 커널 함수를 사용하여 해결하였다.

$$KDE = \frac{1}{n} \sum_{i=1}^n K_k(x - x_i) = \frac{1}{nh} \sum_{i=1}^n K\left(\frac{x - x_i}{h}\right) \quad (2.1)$$

각 데이터의 일정 반경 내에 주변 데이터를 포함한 데이터의 분포도를 식(2.1)과 같이 KDE를 통해서 계산하고, 데이터 분포도가 높은 방향으로 데이터가 이동된다. [14]

2.4.3 DBSCAN 알고리즘

DBSCAN(Density-based spatial clustering of applications with noise) [15]은 Mean-shift 알고리즘과는 달리 한 점을 기준으로 해서, 거리 ϵ (epsilon) 과 점의 개수(minPts)이 주어지면 한 클러스터를 결정한다. 아래 그림은 DBSCAN의 동작 예시이다. 클러스터를 생성할 때 필요한 두 인자, 임의의 거리 ϵ 와 점의 개수가 4개라고 주어지고, 파란색 점들이 임의로 선정된 기준점이

라고 가정하면, (a) 같은 경우는 한 점인 파란색 점을 기준으로 반경 내에 점의 개수가 4개이므로, 하나의 클러스터로 묶여질 수 있다. 반면 (b)의 경우, 한 점인 파란색 점을 기준으로 반경 내에 점의 개수가 2개이므로, (b)의 예시는 클러스터로 묶여질 수 없다. DBSCAN은 클러스터의 총 개수를 미리 정해야 할 필요가 없고 이상치(noise)를 효과적으로 제외할 수 있다는 장점이 있다.

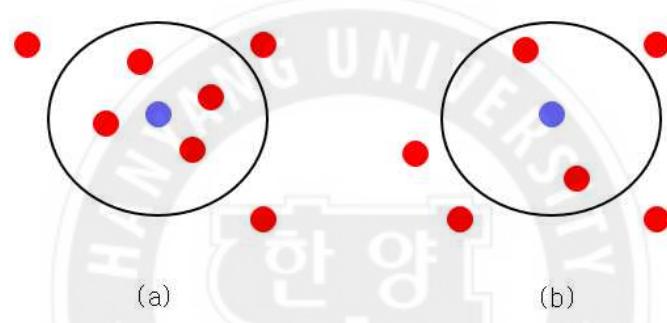


그림 14. DBSCAN 알고리즘의 예시

- (a) DBSCAN 클러스터 조건이 성립되는 경우
- (b) DBSCAN 클러스터 조건이 성립되지 않는 경우

2.4.4 Expectation-Maximization 알고리즘

Expectation-Maximization clustering 알고리즘 [16]은 데이터 집합이 가우시안 분포를 추종한다고 가정한다. 앞선 거리, 밀도를 고려한 알고리즘은 생성된 클러스터가 원형이라고 가정을 하는 반면, 타원형을 가정하기 때문에 가정에서 발생하는 오차를 줄일 수 있다. 가우시안 분포(gaussian distribution)를 추종하는 클러스터를 찾기 위해서는 가우시안 파라미터를 정의해야 하는데, 이를 Expectation-Maximization 알고리즘을 사용해서 추정하게 된다.

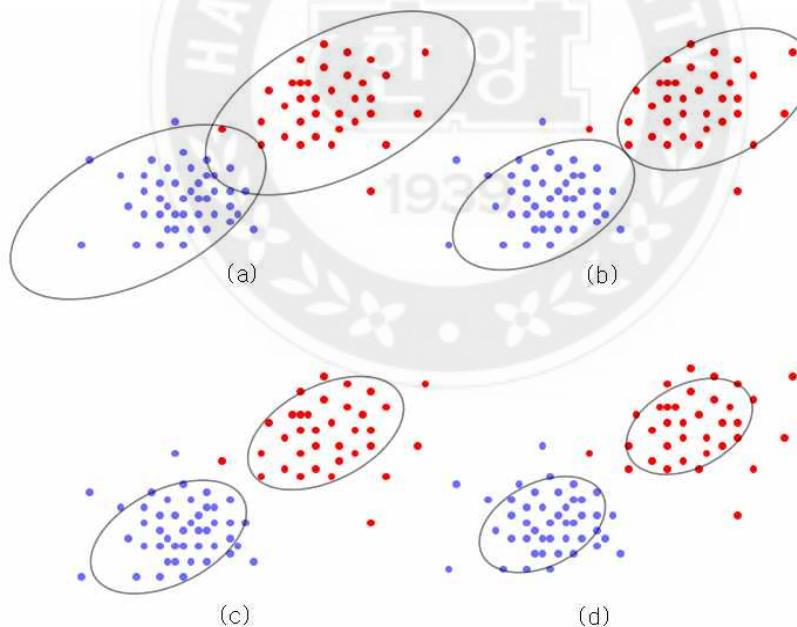


그림 15. Expectation-Maximization 알고리즘의 순서

추정 방식은 초기에 임의의 클러스터 수, 가우시안 파라미터를 정하고, 클러스터 내의 데이터 집합이 각각의 클러스터에 속할 확률을 계산하게 된다. 이 때, 조건부확률을 높이기 위해서 가우시안 파라미터를 재-정의하며 최적화하는 과정을 거치게 된다. 그림은 클러스터링 과정을 나타낸 것이다. 그림 (a)는 가우시안 파라미터를 초기화하여 클러스터를 구성하였는데, 최적화 과정을 거치면서 (b), (c), (d)와 같이 클러스터가 데이터가 밀집된 부분에 위치하고, 이 외 추정 방식은 초기에 임의의 클러스터 수, 가우시안 파라미터를 정하고, 클러스터 내의 데이터 집합이 각각의 클러스터에 속할 확률을 계산하게 된다. 이 때, 조건부확률을 높이기 위해서 가우시안 파라미터를 재-정의하며 최적화하는 과정을 거치게 된다. 그림은 클러스터링 과정을 나타낸 것이다. 그림 (a)는 가우시안 파라미터를 초기화하여 클러스터를 구성하였는데, 최적화 과정을 거치면서 (b), (c), (d)와 같이 클러스터가 데이터가 밀집된 부분에 위치하고, 이외에 동떨어진 데이터는 잡음(noise)이라고 판단하게 된다. [16]

2.5 카메라 센서 보정(Camera-LiDAR Calibration)

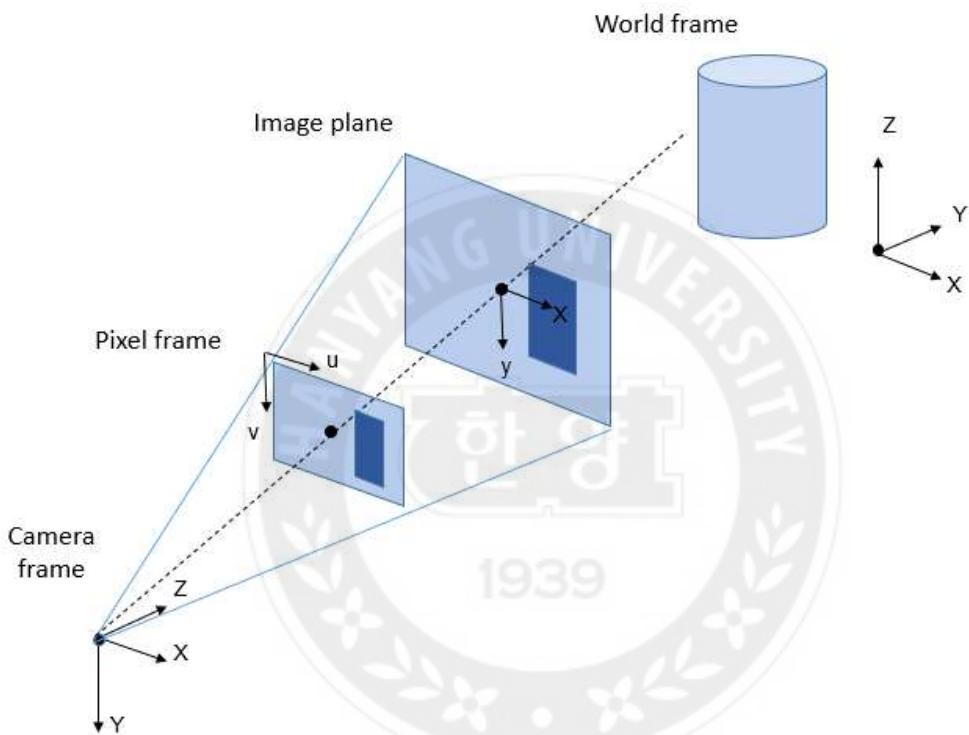


그림 16. 카메라 좌표계와 월드 좌표계의 상관도

데이터를 수집할 때 사용되는 센서들은 각기 다른 좌표계를 가진다. 센서 풍전을 위해서는 각 센서들의 좌표계를 하나의 기준 좌표계로 통일시켜야 한다. 라이다로 캡처된 포인트 클라우드들은 우리가 실제로 보는 World frame의 좌표계이다. 반면, 카메라로 촬영된 이미지는 Camera frame의 좌표계를 가지고 때문에 World frame을 Camera frame으로 변환시켜야 한다. [17]

변환을 할 때에는 homogeneous coordinate를 도입해 점의 변환(transformation)과 벡터의 변환(transformation)이 합쳐진 변환 행렬을 구해야 한다. 아래의 행렬식은 3차원 공간상의 한 점을 이미지 좌표계로의 정사영을 나타낸다. 식 (2.2)는 변환 행렬을 나타낸다.

$$s \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & skew & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = A[R|t] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (2.2)$$

x, y : image plane 좌표계에서의 좌표

c_x, c_y : Pixel frame에서의 원점 좌표를 image plane으로 정사영했을 때 위치

skew : 비대칭 계수

f_x, f_y : 초점 거리

$r_{11}, r_{12}, r_{13}, r_{21}, r_{22}, r_{23}, r_{31}, r_{32}, r_{33}$: 회전 행렬을 구성하는 인자(parameter)

t_1, t_2, t_3 : 이동 행렬 구성하는 인자(parameter)

우 항의 첫 번째 행렬은 내부 행렬(intrinsic matrix)이라고 하며, 카메라 자체의 스펙(specification)에 의해 결정된다. 또한, 두 번째, 세 번째 행렬의 곱을 외부 행렬(extrinsic matrix)이라고 하는데, 이는 카메라와 라이다 좌표계 간의 물리적인 위치와 방향에 의해 결정된다.

제3장 제안하는 기법

3.1 개요

본 3장에서는 논문에서 제안하는 로봇 환경에서의 객체 검출 및 추적 방법에 대하여 설명한다. 2절에서는 제안하는 모델에 대한 전체적인 구조와 구성 요소에 대하여 설명하고 3절에서는 카메라-라이다 간 센서 보정(Camera-LiD AR Calibration) 수행 내용과 4절에서는 2차원 검출 모델의 하이퍼 파라미터 튜닝과 검출 모델에 따른 실험에 대하여 서술한다.

3.2 제안하는 네트워크의 전체 구조

제안하는 네트워크의 전체 구조는 자율주행 로봇환경에서의 검출 및 추적 역할을 수행하게 된다. 전체적인 동작 과정은 YOLOv3의 검출 결과인 2차원 경계 박스(bounding box)의 위치 좌표를 이용해 비 딥러닝 추적 알고리즘인 SORT(Simple Online and Realtime Tracking)를 수행하여, 검출 결과에 추적 id (tracking identification)를 부여한다. 딥러닝 네트워크인 YOLOv3는 Aihub의 도로주행 데이터 셋을 사용하여 훈련(training)시켜, 학습률(learning rate)의 parameter-tunning을 거쳐 mAP(mean Average Precision)가 가장 높은 조합의 체크포인트(checkpoint)를 사용하였다. 이후, 3차원 공간 상의 포인트 클라우드를 2차원 이미지 공간으로 정사영시켜, 검출 결과인 경계 박스(bounding box)

내부에 있는 정사영된 점을 마스킹(masking)하게 된다. 이 때, 이미지와 포인트 클라우드의 시간적 동기화를 위해 마스킹 된 점들을 대상으로 군집화 과정을 거치는데, 군집화 알고리즘의 한 방법인 DBSCAN 알고리즘을 사용해 정사영된 점들 중 객체에 해당하는 점과 배경에 해당하는 점을 구분하게 된다.



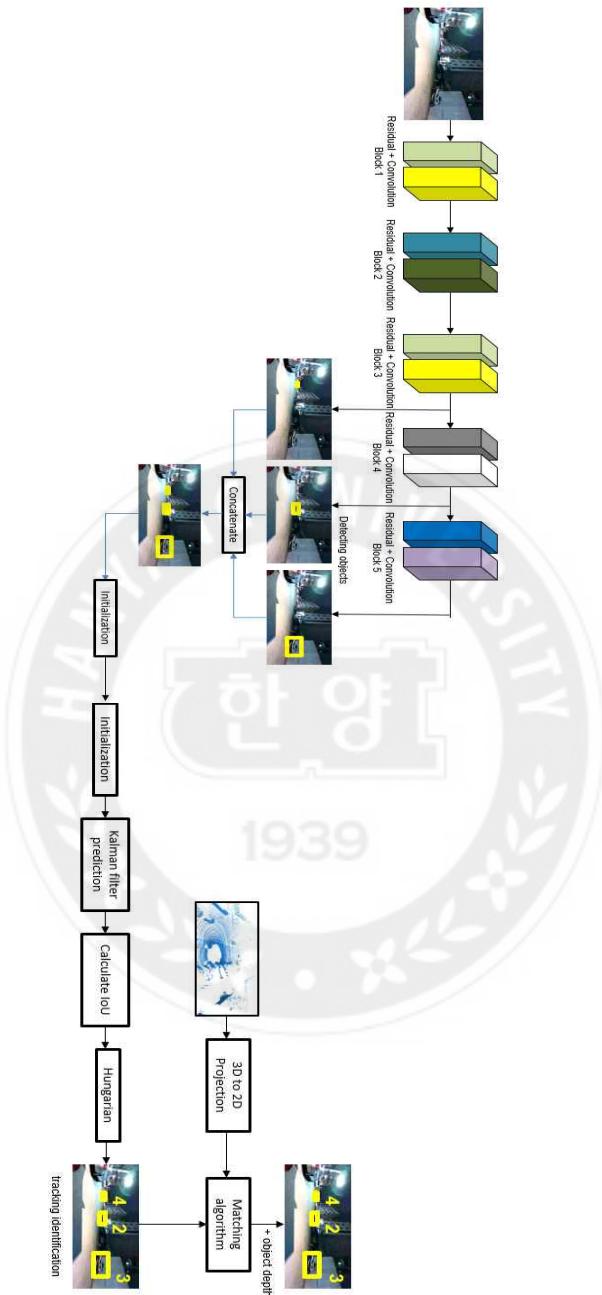


그림 17. 제안하는 알고리즘의 구조

정사영했을 때, 경계 박스(bounding box) 내부에 있는 점 즉, 객체에 해당한다고 판단된 점들은 위치 좌표를 평균(average)하여, 객체의 3차원 좌표로 정의하게 된다.

전체 네트워크의 흐름은 아래의 그림과 같다. Joseph Redmon[6]이 제안한 1-stage detector는 Residual Block과 Convolution Block이 조합된 네트워크로 구성되어 있으며, 다른 해상도를 가지는 통합된 블록들은 객체를 검출하는데 각각 사용되어 객체의 크기에 따라 검출하게 된다. 이후에, 각각의 검출된 객체의 정보들은 하나의 출력으로 통합된다. SORT(Simple Online Realtime Tracking) [18] 알고리즘은 아래 그림과 같이 구성된다.

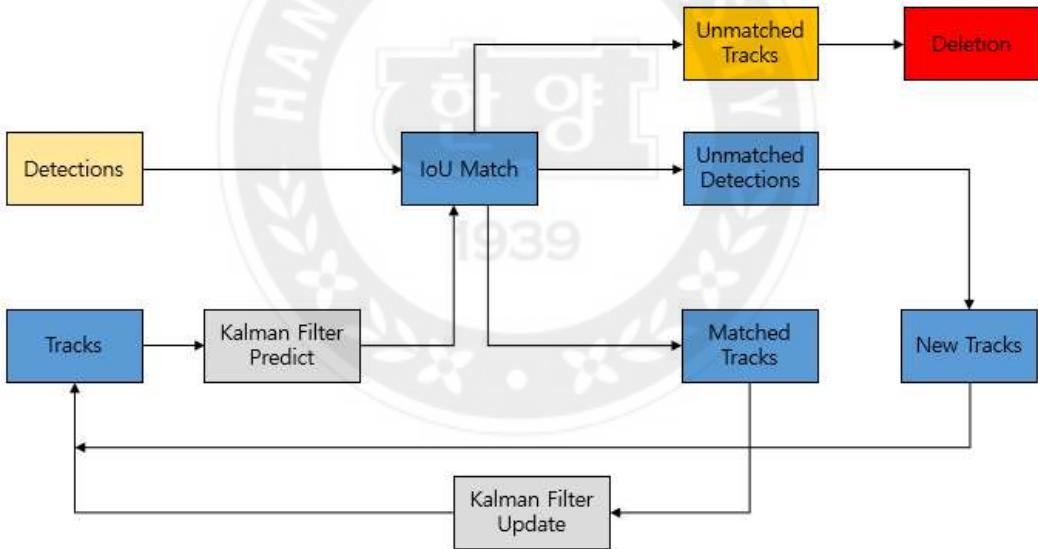


그림 18. SORT 알고리즘

SORT 알고리즘의 입력은 검출된 객체의 위치 좌표이며, 한 프레임(frame) 전에 검출된 객체에 부여된 추적 ID(tracking identification)과 현재 프레임(fra

me)에 입력된 객체의 위치좌표 간의 상관관계를 계산한다.

알고리즘이 처음 동작하게 되면 이전 프레임(frame)에 대한 정보가 없으므로, 초기화(initialization) 단계를 거치게 되며, 초기화(initialization) 단계는 검출된 객체에 모두 추적 ID(tracking identification)을 부여 한다.

이전 프레임의 객체 경계 상자(bounding box)는 Kalman filter에 의해 현재 프레임에서의 객체 위치를 예측하고 예측된 객체 경계 상자(bounding box)와 현재 프레임(frame)에서의 IoU Matching을 이용해 데이터 연결(data association)을 진행한다. 예측된 객체 경계 상자(bounding box)와 현재 프레임(frame)에서의 IoU matching을 이용해 데이터 접합(data association)을 진행한다.

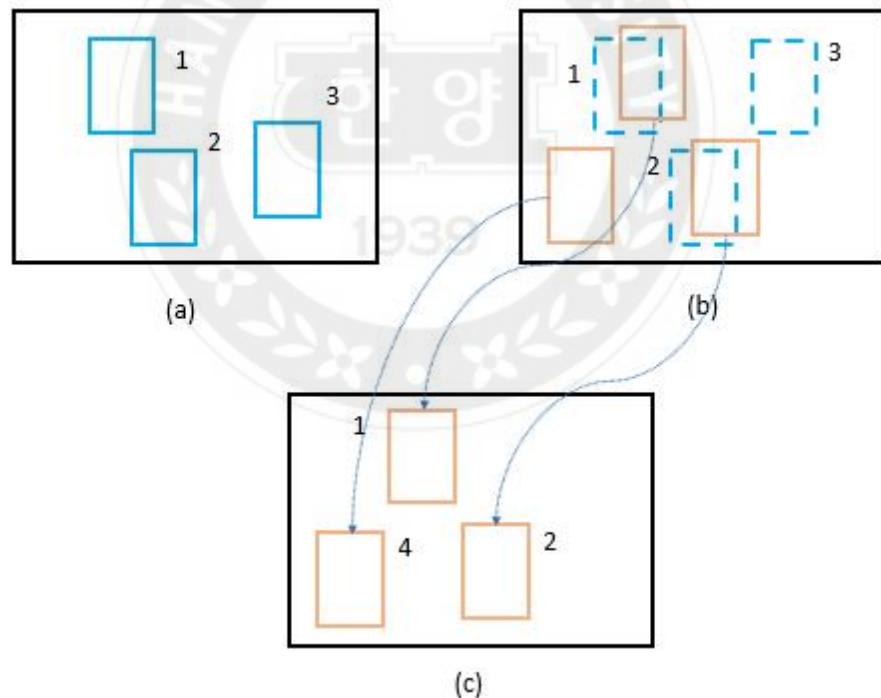
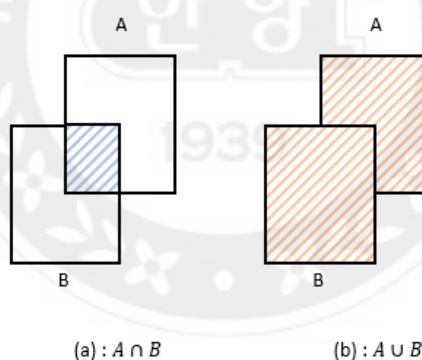


그림 19. IoU Matching 과정

위 그림은 IoU sensor fusion에 대한 설명이다. 진행 순서는 $(a) \rightarrow (b) \rightarrow (c)$ 이며, (a)는 이전 프레임에 대해 검출된 객체의 경계 상자(bounding box)와 부여된 추적 ID(tracking identification)을 나타낸다.

(b)의 파란색 점선 상자는 Kalman filter에 의해 예측된 상자의 위치를 나타낸다.

또한, (b)의 주황색 상자는 현재 프레임에서 검출된 경계 상자(Bounding Box)로써 파란색 점선 상자와의 IoU(Intersection of Union)을 계산한다. IoU(Intersection of Union)는 아래 그림의 예시와 같이 계산할 수 있다. 의미론적으로 두 상자간의 상관관계를 두 상자의 교집합과 합집합의 비율로 나타낼 수 있고, IoU가 1에 가깝다는 것은 두 상자가 동일한 객체를 나타낼 가능성이 높다. 또한 IoU가 0에 가깝다는 것은 두 상자가 다른 객체일 가능성이 높은 것을 의미한다.



$$IoU = \frac{A \cap B}{A \cup B}$$

그림 20. IoU(Intersection of Union)

3.2.1 제안하는 센서 퓨전 알고리즘(Sensor fusion algorithm)

제안하는 센서 퓨전 알고리즘은 2차원 경계 상자(bounding box)를 3차원 경계 상자(bounding box)로 확장시키는 것을 목표로 한다. 아래 그림은 센서 퓨전 알고리즘의 순서도이다.

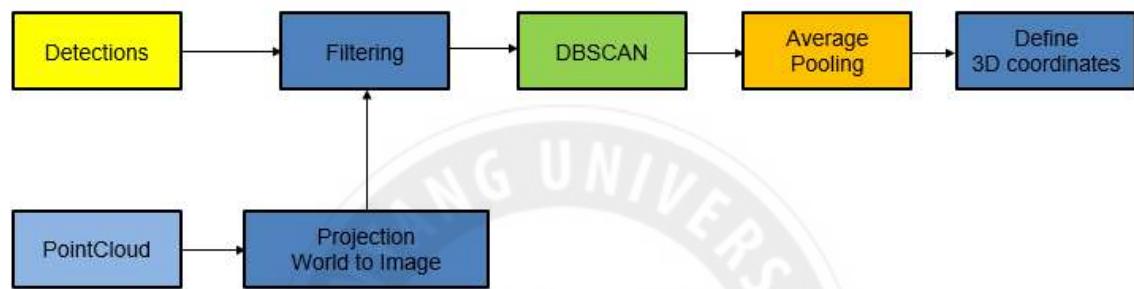


그림 21. 제안하는 센서 퓨전 알고리즘(sensor fusion algorithm)의 순서도

3.3.1절에서 설명될 카메라-라이다 보정(camera-liDAR calibration)을 수행하여 얻은 변환 행렬(transformation matrix)을 사용해 3차원 공간상의 포인트 클라우드들을 2차원 공간으로 정사영 한다. 검출된 경계 상자(bounding box) 외부에 정사영된 점들은 필터링(filtering)된다. 필터링된 점들 외의 점들은 군집화 알고리즘(clustering algorithm)인 DBSCAN(Density-Based Spatial Clustering of Applications with Noise)을 사용하여 군집화되며, 생성된 군집(Cluster)중 센서로부터 가장 가까운 클러스터를 대표 군집(cluster)으로 하여, 평균 풀링(average pooling)을 수행한다. 이에 의해 대표 군집(cluster)은 하나의 3 차원 좌표로 나타낼 수 있게 되고, 이를 객체의 3차원 좌표로 정의한다.

3.3 학습 실험환경

본 논문에서 제안하는 센서 퓨전 알고리즘을 이용한 3차원 객체 검출 및 추적 방법의 모델 학습의 실험환경은 아래의 표3과 같다.

표 3. 학습 실험환경

Hardware	CPU	Intel(R) Core(TM) i7 9800X
	GPU	Nvidia GeForce 1080Ti * 4
Software	Platform	Ubuntu 18.04.2 LTS
	Compiler	Python 3.6.9
	Deep Learning Framework	Pytorch 1.7.0
	GPU API and Library	CUDA 10.2
		cuDNN 7.6.5

3.3.1 Aihub 인도보행 영상 데이터셋

본 논문에서 제안하는 센서 퓨전 알고리즘을 이용한 3차원 객체 검출 및 추적방법의 모델 학습을 위해 Aihub 인도보행 영상 데이터셋을 학습에 사용하였다. Aihub 데이터셋은 한국지능정보사회진흥원의 사업결과이며, 휴대폰과 ZED사의 스테레오 카메라로 촬영되었으며 훈련 세트 229,753장과 검증 세트 9,8466장으로 구성되어 있다. [2]



그림 22. Aihub 인도보행 영상 데이터 셋

데이터 셋은 사전 레이블링된 Ground-Truth 2차원 경계박스(bounding box)가 annotation으로 제공되며, 이를 딥러닝 네트워크의 학습에 사용했다.

3.3.2 하이퍼-파라미터 튜닝(Hyper-Parameter Tuning)

본 절에서는 학습률(learning rate) 변화에 따른 mAP(mean Average Precision) 결과를 서술한다. 그 결과는 아래의 표 4와 같으며, mAP가 가장 높은 Case3의 학습지(checkpoint)를 사용하여 추론에 사용하였다. 각 mAP는 IoU 기반의 mAP 측정방법으로 계산되었다.

표 4. 하이퍼-파라미터에 따른 mAP 비교

	Learning rate	mAP
Case1	0.001	67
Case2	0.0005	68.7
Case3(Ours)	0.0001	69.7

3.3.3 군집화 방식에 따른 센서 퓨전 알고리즘의 속도 비교

본 절에서는 군집화 방식에 따른 센서 퓨전 알고리즘의 속도를 비교하였다. 비교 대상으로 K-MEANS, OPTICS [19]알고리즘을 사용했으며 그 결과는 아래의 표 4와 같다. DBSCAN을 사용한 센서 퓨전 알고리즘의 속도가 비교 모델에 비해 우수했으며, 클러스터의 개수를 제한하지 않고 사전 정의된 정해진 최소 점의 개수, 반경으로 측정되며 알고리즘 의존적인 계산에 의해 실행 속도가 빠른 것으로 판단했다.

표 5. 군집화 방식에 따른 센서 퓨전 알고리즘의 속도 비교

	Cluster	ms
Case1	K-MEANS+ Sensor fusion	318.67
Case2	OPTICS+ Sensor fusion	437.78
Case3	DBSCAN+ Sensor fusion(Ours)	66.54

제4장 실험 결과

4.1 개요

본 장에서는 실험을 통해 제안하는 객체 및 추적 알고리즘의 결과와 알고리즘의 객체까지의 거리 추정 성능을 서술한다. 또한 성능 비교를 위해, 제안하는 기법과 다른 기법간의 추론 속도를 비교하고, 실시간 구현이 가능한 인지 단계로서의 평가를 목표로 한다.

4.2절에서는 실험이 수행된 추론(inference) 환경과 사용한 데이터 셋에 대해 서술한다. 4.3절에서는 수행한 카메라-라이다 보정 결과와 과정을 설명하고 4.4절에서는 제안하는 알고리즘의 정량적, 정성적 평가를 서술한다.

4.2 실험환경

본 절에서는 실험이 수행된 추론(inference) 환경에 대해서 기술한다.

표 6. 추론 실험환경

Hardware		Jetson AGX Xavier 16GB
Software	Platform	Ubuntu 18.04.2 LTS
	Compiler	Python 3.6.9
	Deep Learning Framework	Pytorch 1.7.0
	GPU API	CUDA 10.2

4.3 카메라-라이다 보정(Camera-LiDAR Calibration) 결과

카메라-라이다 보정을 위해 아래 그림의 체스판(chessboard)를 사용, 카메라-라이다를 이용하여 위치와 방향을 바꿔 촬영한다. 촬영방식은 ROS(Robot Operating System)을 사용하여 포인트 클라우드와 이미지의 동시간대의 영상을 Rosbag 형식으로 취득한다.

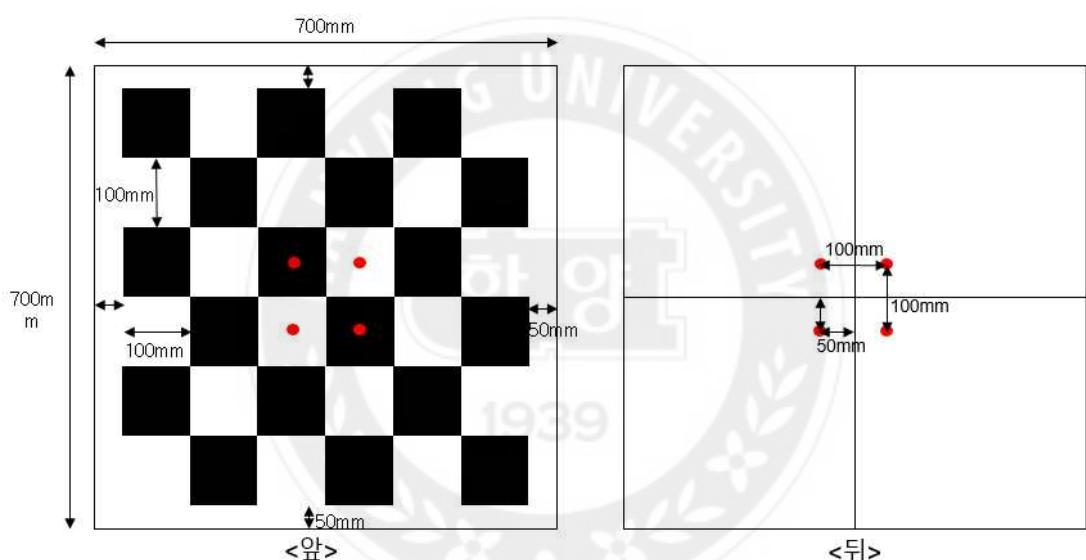


그림 23. 보정 작업에 사용된 체스판(Chessboard)

전체 체스판의 가로 길이 : 700mm

전체 체스판의 세로 길이 : 700mm

체스판의 격자무늬의 가로 길이 : 100mm

체스판의 격자무늬의 세로 길이 : 100mm

rosbag 형식으로 촬영된 파일은 체스판(chessboard)의 방향과 위치에 따라 캡쳐하여, 변환 행렬(transformation matrix)의 인자(parameter)를 찾기 위한 방정식의 해로 사용된다.

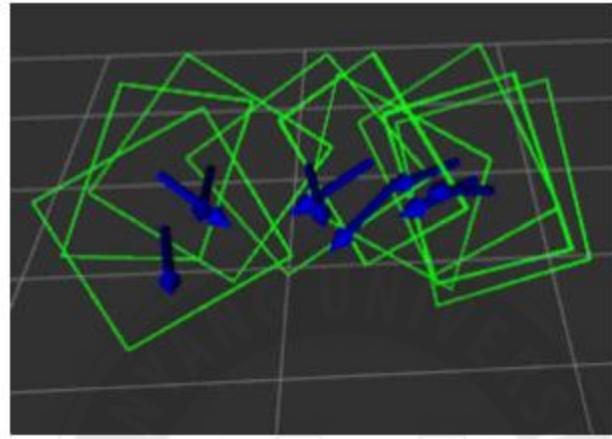


그림 24. 카메라-라이다 보정 과정 1, 샘플링된 체스판의 범선벡터 추정

위 그림 24는 촬영된 rosbag 형식의 영상을 캡쳐(capture)하여, 체스판(chessboard)의 위치와 방향의 추정을 나타낸다. 위치와 방향이 추정되면 체스판 상의 격자점을 주어진 가로, 세로 길이로 유추하고, 촬영된 이미지는 해리스 코너 격자점 검출을 사용하여 이미지 좌표계 상에서의 격자점에 대한 좌표를 구할 수 있다. 이에 의해, 동일한 격자점의 월드 좌표계 상에서의 좌표, 이미지 좌표계 상에서의 좌표를 구하여 변환 행렬의 각 인자(parameter)를 정립할 수 있다. 아래 그림은 그 결과를 나타낸다. 행렬 방정식을 풀기 위한 최소 데이터 수보다 많은 데이터를 조합하여, 각 경우에 대한 변환 행렬의 인자(parameter)를 정립하고, 평균-분산 최적화(mean-variance optimization) [20]에 의해 하나의 인자로 최적화를 진행한다. 변환 행렬을 구성하는 인자는 roll, pitch, y

aw, t_x , t_y , t_z 로 이루어지며, 카메라와 라이다 좌표계 간의 방향, 원점의 거리를 나타낸다. 표 7은 카메라-라이다 보정결과를 나타낸다.

표 7. 카메라-라이다 보정 결과

roll	pitch	yaw	t_x	t_y	t_z
1.60898	0.00276	1.56539	0.20108	0.00922	0.33953



4.4 제안하는 알고리즘의 실험 결과

4.4.1 검출 모델에 따른 추론 속도 결과

본 절에서는 2차원 검출 모델에 따른 추론 속도를 비교한다. Jetson AGX Xaiver는 일반 GPU에 비해 낮은 성능을 지니고 있기 때문에 상대적으로 느린 연산속도를 가진다.

표 8. 검출 모델에 따른 추론 속도 비교

Model	FPS(Frame Per Second)
Faster R-CNN	3
FCOS	3
RetinaNet	5
YOLOF	4
YOLOv3(Ours)	15

2-stage detector의 대표적인 모델인 Faster R-CNN과 1-stage detector의 대표적인 모델 YOLOv3를 비롯한 YOLOF [21], FCOS [22] 등을 대상으로 실험을 진행했으며 그 결과는 표 4와 같다.

4.4.2 센서 퓨전 알고리즘의 정량적 실험 결과

본 절에서는 제안하는 센서 퓨전 알고리즘의 정량적인 추론속도 비교 실험 결과를 서술한다. 정량적 비교 대상은 SECOND [23], Pointpillars [24], 3DSS D [25]이며 대표적인 라이다 기반의 3차원 객체 검출 모델이다. 실험은 Nvidia GeForce 1080Ti 1개를 사용했으며, KITTI [27] 데이터셋을 사용하여 추론 속도를 측정했다.

그리고, 제안하는 전체 알고리즘 중 추적 과정은 제외하고 실험을 진행하였다.

표 9. 센서 퓨전 알고리즘의 정량적 추론속도
비교 실험 결과

거리 범위(m)	추론속도(Frame Per Second)
SECOND	25.27
Pointpillars	29.42
3D-SSD	13.4
Ours	30.56

제안하는 알고리즘은 SECOND [20], Pointpillars [21], 3D-SSD [22]와 비교했을 때, 빠른 추론속도를 보이며 3D-SSD와는 두배 이상 추론 속도가 빠르다는 것을 확인했다.

4.4.3 센서 퓨전 알고리즘의 정성적 거리 추정 결과



그림 25. 제안하는 알고리즘의 정성적 거리 추정 결과

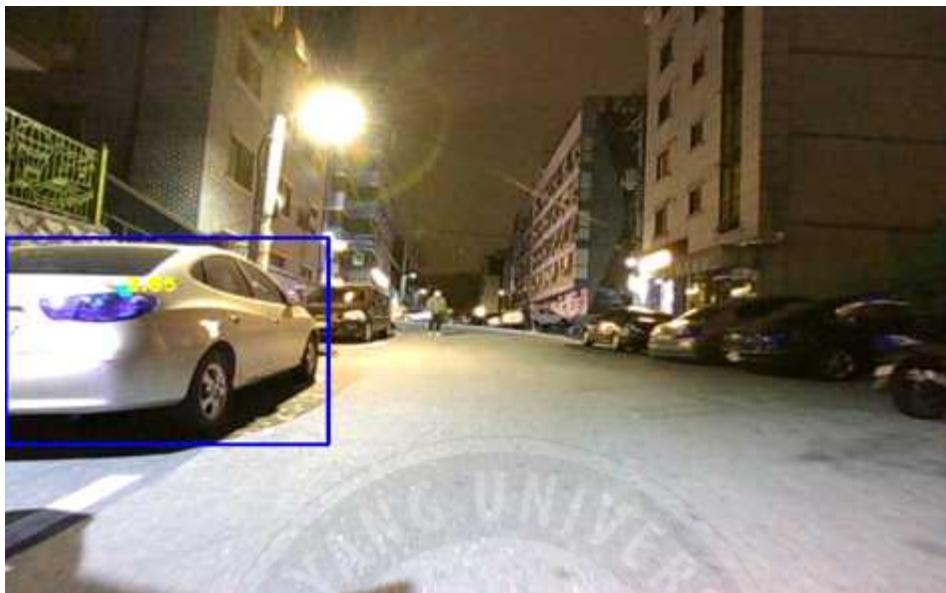
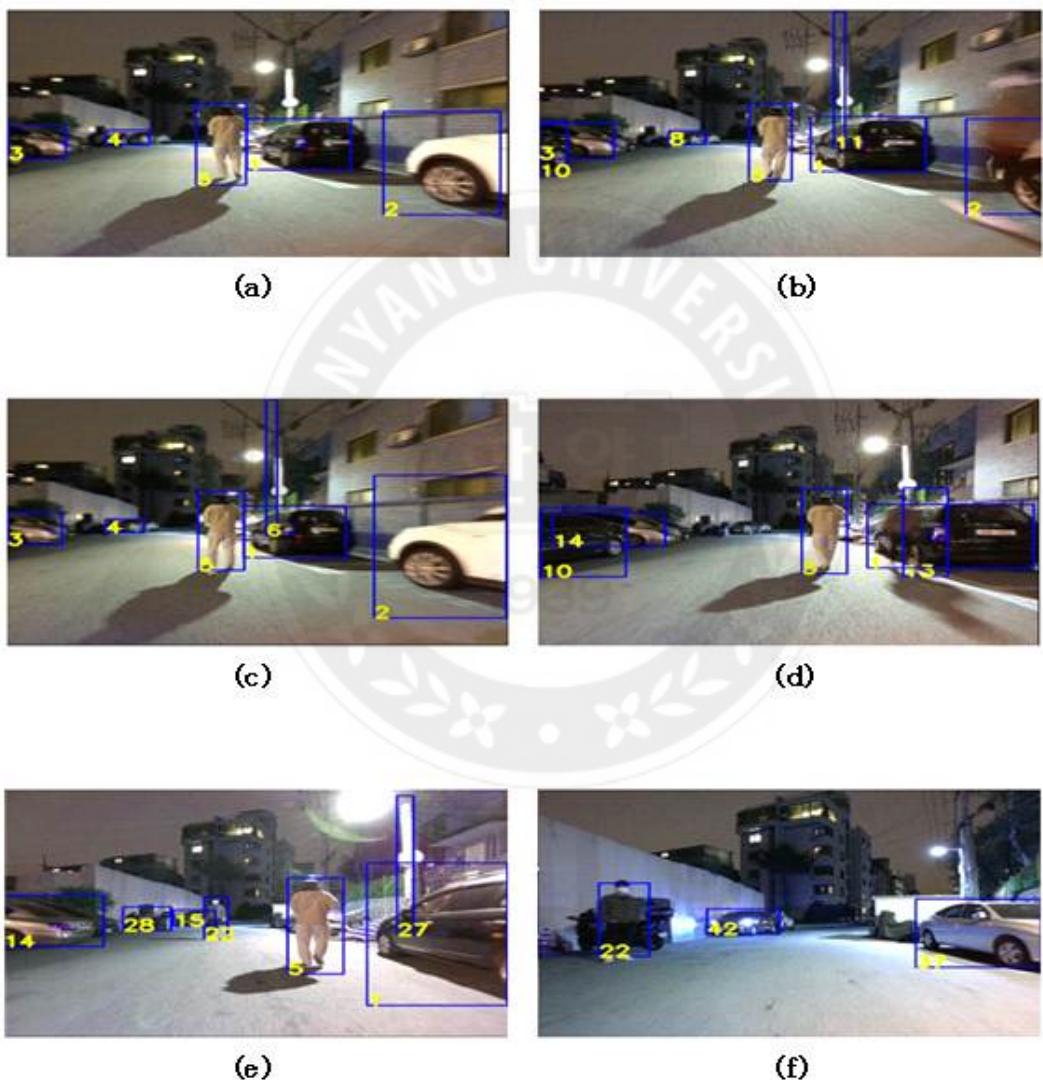
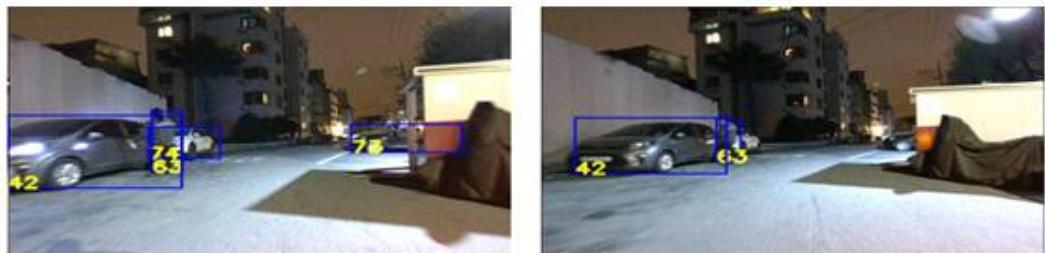


그림 26. 제안하는 알고리즘의 정성적 거리 추정 결과 2

4.4.4 센서 퓨전 알고리즘의 정성적 2차원 객체 검출 및 추적 결과





(g)

(h)

그림 27. (a), (b), (c), (d), (e), (f), (g), (h)
시간 순에 따른 알고리즘의 정성적 객체 검출 및 추적 결과

제5장 결론

인공지능은 분류(classification) 문제를 수동적인 네트워크 설계로 해결하는 것을 시작으로 데이터셋을 이용한 학습(learning)방식의 딥러닝(deep-learning)으로 진화되었다. 인공지능의 응용영역은 나날이 확장되어왔으며, DARPA Grand Challenge가 기폭제 역할을 하여 그 영역은 인공지능을 이용한 자율주행 기개발 그리고 자율주행 배달 로봇까지 이르렀다. 자율주행의 동작 순서는 인지, 판단, 제어로 나뉘며 센서와 인공지능을 이용한 인지 파트는 가장 먼저 실행되는 만큼 중요하며, 실시간 추론 속도와 정확성을 충족해야 한다.

카메라와 레이더는 라이다에 비해 가격이 저렴하여, 상용 가능성을 판단했을 때 선택할 수 있는 좋은 옵션이다. 하지만 라이다는 수집되어 취득되는 포인트 클라우드의 3차원 위치 좌표가 정확하여 매력적이고 업체들의 경쟁으로 인해 가격이 하락하고 있어 라이다도 좋은 선택지를 제공할 수 있다.

본 논문에서는 실시간 객체 검출 및 추적 알고리즘인 YOLO와 SORT의 2차원 객체 정보 결과를 3차원 공간상으로 확장하기 위해, DBSCAN clustering을 이용한 센서 퓨전 알고리즘을 제안하며 그 활용 가능성을 증명한다. 실제 응용을 위해 로봇에 흔히 탑재되는 Jetson AGX Xavier 상에 ZED Stereo Camera 그리고 Velodyne 16채널 라이다로 실시간 데이터를 취득할 수 있도록 ROS(Robot Operating System)으로 구현하고, 제안하는 알고리즘을 탑재하였다.

제안하는 알고리즘은 YOLOv3의 출력값인 2차원 경계 상자(2d bounding box)를 SORT 알고리즘으로 입력하여 2차원 객체 검출 및 추적을 수행한다. 멀티 쓰레딩(multi-threading)기법을 이용하여 이미지와 비동기화된 포인트 클라

우드의 정사영을 통해 2차원 경계 상자 외부의 점을 필터링하고, 남겨진 점들에 대해 DBSCAN 알고리즘을 이용하여 클러스터를 생성하였다. 클러스터에 포함된 점들은 평균화하여 대표 점으로 나타내어지고 이를 3차원 공간상의 좌표로 정의하였다. 제안하는 알고리즘은 실시간 추론이 가능했으며, 객체까지 실측 거리와의 비교를 통해 결과를 입증하였다. 그러나 2차원 객체 검출기에 의존된 알고리즘은 로봇의 횡이동, 객체의 빠른 움직임으로 나타나는 블러(blur) 현상에 취약했으며, 폐색(occlusion)된 객체는 검출하지 못하는 상황이 발생했다. 이를 개선하기 위해 향후 3차원 객체 검출기를 기반으로 하여 앞서 발생한 상황들을 해결하고 이미지 정보를 이용해 3차원 객체 검출기의 약점이라고 할 수 있는 분류(classification) 정확도를 보완하는 방법에 대한 연구가 필요하다.



참고문헌

- [1] Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 580–587).
- [2] <https://aihub.or.kr/aihubdata/data/view.do?currMenu=115&topMenu=100&aihubDataSe=realm&dataSetSn=189>
- [3] Uijlings, J. R., Van De Sande, K. E., Gevers, T., & Smulders, A. W. (2013). Selective search for object recognition. International journal of computer vision, 104(2), 154–171.
- [4] Girshick, R. (2015). Fast r-cnn. In Proceedings of the IEEE international conference on computer vision (pp. 1440–1448).
- [5] Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. Advances in neural information processing systems, 28.

- [6] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 779–788).
- [7] Redmon, J., & Farhadi, A. (2017). YOLO9000: better, faster, stronger. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7263–7271).
- [8] Redmon, J., & Farhadi, A. (2018). Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767.
- [9] Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). Point net: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 652–660).
- [10] Zhou, Y., & Tuzel, O. (2018). Voxelnet: End-to-end learning for point cloud based 3d object detection. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4490–4499).
- [11] Vora, S., Lang, A. H., Helou, B., & Beijbom, O. (2020). Pointpainting: Sequential fusion for 3d object detection. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 4604–4612).

- [12] Chen, L. C., Papandreou, G., Schroff, F., & Adam, H. (2017). Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv:1706.05587.
- [13] Krishna, K., & Murty, M. N. (1999). Genetic K-means algorithm. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 29(3), 433–439.
- [14] Carreira-Perpinán, M. A. (2015). A review of mean-shift algorithms for clustering. arXiv preprint arXiv:1503.0687.
- [15] Ester, M., Kriegel, H. P., Sander, J., & Xu, X. (1996, August). A density-based algorithm for discovering clusters in large spatial databases with noise. In kdd (Vol. 96, No. 34, pp. 226–231).
- [16] Erlich, I., Venayagamoorthy, G. K., & Worawat, N. (2010, July). A mean-variance optimization algorithm. In IEEE Congress on Evolutionary Computation (pp. 1–6). IEEE.
- [17] Bloomenthal, J., & Rokne, J. (1994). Homogeneous coordinates. The Visual Computer, 11(1), 15–26.
- [18] Bewley, A., Ge, Z., Ott, L., Ramos, F., & Upcroft, B. (2016, September). Simple online and realtime tracking. In 2016 IEEE international conference on image processing (ICIP) (pp. 3464–3468). IEEE.

- [19] Ankerst, M., Breunig, M. M., Kriegel, H. P., & Sander, J. (1999). OPTICS: Ordering points to identify the clustering structure. *ACM Sigmod record*, 28(2), 49–60.
- [20] Erlich, I., Venayagamoorthy, G. K., & Worawat, N. (2010, July). A mean-variance optimization algorithm. In *IIEEE Congress on Evolutionary Computation* (pp. 1–6). IIEEE.
- [21] Chen, Q., Wang, Y., Yang, T., Zhang, X., Cheng, J., & Sun, J. (2021). You only look one-level feature. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 13039–13048).
- [22] Tian, Z., Shen, C., Chen, H., & He, T. (2019). Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF international conference on computer vision* (pp. 9627–9636).
- [23] Yan, Yan, Yuxing Mao, and Bo Li. "Second: Sparsely embedded convolutional detection." *Sensors* 18.10 (2018): 3337.
- [24] Lang, A. H., Vora, S., Caesar, H., Zhou, L., Yang, J., & Beijbom, O. (2019). Pointpillars: Fast encoders for object detection from point clouds. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition* (pp. 12697–12705).

- [25] Yang, Z., Sun, Y., Liu, S., & Jia, J. (2020). 3dssd: Point-based 3d single stage object detector. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 11040–11048).
- [26] Shi, S., Wang, X., & Li, H. P. (2019, June). 3d object proposal generation and detection from point cloud. In Proceedings of the IEEE conference on computer vision and pattern recognition, Long Beach, CA, USA (pp. 15–20).
- [27] Geiger, A., Lenz, P., Stiller, C., & Urtasun, R. (2013). Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11), 1231–1237.

ABSTRACT

3D Object Detection and Tracking for a Robot Platform Using DBSCAN

Kim, Min Wook

Dept. of Artificial Intelligence

Graduate School of
Hanyang University

Recognition of the surrounding environment using sensors in autonomous driving algorithms can be said to be an important task as it serves as a human eye. Information obtained through the recognition of static objects such as signs, traffic lights, stop lines, and dynamic objects such as pedestrians and automobiles is used as a key clue to the judgment and control algorithms that operate later.

Deep learning-based cognitive algorithms have evolved from the design of passive networks. Learning-based training using datasets is being applied to various fields such as robots and medical images in addition to autonomous driving as the development of computers and the improvement of sensor performance act as catalysts. Sensor selection and rapid and accurate algorithm design are the most important challenges because the available specifications on platforms such as automobiles and robots are limited.

LIDAR has the disadvantage of being expensive compared to sensors such as cameras and radars. However, there have been many lida lately With competition from companies and the development of lidar sensing technology, the exact location coordinates of the point cloud acquired through lidar are attractive options compared to other sensors.

Two-dimensional object detection using images has the advantage of providing semantic information such as color and texture compared to object detection using lidar. Therefore, this paper introduces a three-dimensional object detection and tracking algorithm using a matching algorithm based on two-dimensional object detection and tracking.

The proposed algorithm improves real-time inference performance with SORT-based tracking using YOLO-based fast two-dimensional object detection results. To train an object detection model based on deep learning, we use the optimal study paper (checkpoint) through parameter tuning by learning with the Aihub sidewalk walking image dataset.

The bounding box of the detected object acts as a filter that excludes point clouds assigned to the outside of the object, and the filtered points are clustered by the DBSCAN algorithm.

Among the clusters present inside the bounding box The cluster closest to the origin of the coordinate system is a cluster of points representing an object, which is averaged and used as a position coordinate representing an object in a three-dimensional space. To determine the effectiveness of the algorithm, the results of the model are compared with the measured distances to calculate its accuracy, and the two-dimensional object and tracking performance are evaluated through qualitative evaluation.

감사의 글

SPA 연구실에서의 2년을 마무리하며 부족한 저를 위해 도와주신 많은 분들에게 감사의 말씀을 전하고 싶습니다. 먼저 저를 연구실의 일원으로 받아주시고 연구와 더불어 인생에 대한 조언을 아낌없이 해주신 최준원 교수님께 감사드립니다. 교수님의 가르침 덕분에 힘들다고 생각이 들 때도 연구와 졸업에 정진할 수 있었습니다. 졸업 후에도 교수님의 가르침을 항상 간직하겠습니다.

1년의 공백기로 인해 대학원 입학 후 잘 적응할 수 있을까 많은 걱정을 하기도 했습니다. 이런 제가 잘 적응할 수 있었던 것은 인턴과 논문 읽기 스터디를 진행해주신 선배님들이 계셨기 때문입니다. 또한, 연구실에 잘 녹아들 수 있도록 도와준 같은 학부를 졸업한 지송이 형, 민재 형에게 감사하다는 말을 전하고 싶습니다. 2년 동안의 ITBT에서의 생활과 추억은 절대 잊지 못할 것 같습니다. 옆자리에서 많이 도와준 건율이 형, 항상 밝은 창원이, 듬직한 종욱이 형, 코딩 테스트 공부를 도와준 정호, 건호, 세환이 형에게 감사합니다. 언급하지 못한 선배님들, 후배님들 감사드리고 모두 지금 하시는 연구가 잘되어 좋은 논문을 쓰시도록 기원하겠습니다.

그리고 2년 동안 물심양면으로 도와주신 아버지와 어머니, 동생에게 감사드립니다. 대학원에 가겠다고 했을 때 저를 믿어주셔서 그 믿음이 저의 연구실 생활에 많은 도움이 되었습니다. 앞으로 더욱더 보답하겠습니다.

추억이 깃든 학교와 연구실을 떠나게 되면서 돌아보면 아쉬움도 많고 한편으로는 좋기도 합니다. 마지막으로 연구실에서 2년의 시간을 양분 삼아 앞으로 사회에 좋은 영향을 주는 사람이 되겠습니다. 감사합니다.

2023년 2월 김민욱 올림

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.

2022년12월13일

학위명 : 석사

학과 : 인공지능학과

지도교수 : 최준원

성명 : 김민욱

김민욱(手寫)

한 양 대 학 교 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 13, 2022

Degree : Master

Department : DEPARTMENT OF ARTIFICIAL INTELLIGENCE

Thesis Supervisor : Jun-Won Choi

Name : KIM MINWOOK

(Signature)