https://doi.org/10.7840/kics.2024.49.1.31

# 자율주행을 위한 포인트 클라우드 3D 객체 인식에 관한 연구

정 영 재◆, 전 우 민*, 이 성 진°

# Study on Point Cloud Based 3D Object Detection for Autonomous Driving

Youngjae Cheong◆, Woomin Jun*, Sungjin Lee°

요 약

자율주행 자동차의 상용화를 위해서는 정확한 3차원 공간 기반 상황인지 기술이 필수적이다. 이를 위해서 카메라만으로는 그 인식 성능에 한계가 있어 라이다 기반의 3차원 상황인지 기술 도입이 필수적이며, 이런 라이다 기반의 3D Object Detection 기술의 정확도를 최대화할 수 있는 포인트 클라우드 데이터 증식 방법에 대해 연구하였다. 이 포인트 클라우드 기반 데이터 증식 방법으로 Jitter, Uniform Sampling, Random Sampling, Scaling 기반의 방법을 사용하여 그 정확도를 분석하였으며 이들의 조합 통해 3D Object Detection의 정확도를 최대화 할 수 있는 방법에 대해 탐구하였다. 실험 결과 KITTI dataset 기준으로 정확도 AP가 약 0.5~0.8 정도 향상되는 것을 보였으며, Jitter기법이 성능향상에 가장 효과적이며 클래스마다 다른 데이터 증식을 적용하는 것이 더 좋은 결과를 얻을 수 있다는 것을 알아내었다.

**키워드 :** 자율주행, 포인트 클라우드, 3차원 객체 인식, 데이터 증식
**Key Words :** Autonomous Driving, Point Cloud, 3D Object Detection, Data Augmentation

ABSTRACT

For the commercialization of autonomous vehicles, precise perception based on three-dimensional (3D) spatial recognition is imperative. While cameras offer valuable insights, their perception capabilities are inherently limited for comprehensive 3D spatial awareness. Therefore, the integration of LIDAR-based spatial recognition technology is indispensable. This study delved into methods for augmenting point cloud data to maximize the accuracy of LIDAR-based 3D Object Detection. Through this point cloud augmentation approach, techniques such as Jitter, Uniform Sampling, Random Sampling, Scaling, and Translation were employed and analyzed for their impact on detection accuracy. Furthermore, we explored optimal combinations of these techniques to amplify the precision of 3D Object Detection. Experimental outcomes, benchmarked against the KITTI dataset, showcased an improvement in the average precision (AP) by approximately 0.5-0.8. In addition, it was discerned that adopting distinct augmentation techniques, in particular Jitter, for different classes yielded enhanced results.

※ 이 논문은 2023년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (2023 신산업특화선도전문대학 지원사업)
◆ First Author : Dong-Seoul University, Department of Electric Engineering, bluebull777@naver.com, 학생회원
° Corresponding Author : Dong Seoul University Department of Electronic Engineering, sungjinlee@du.ac.kr, 정회원
* Dong-Seoul University, Department of Electric Engineering, aplus912@naver.com, 학생회원
논문번호 : 202308-064-C-RU, Received August 27, 2023; Revised October 5, 2023; Accepted October 5, 2023

## Ⅰ. 서 론

최근 인공지능의 발전으로 촉발된 자율주행 기술의 비약적 진보는 단순히 인간의 운전이라는 노동으로부터의 해방이 아닌 교통사고율의 감소, 친환경 에너지로의 전환, 수많은 모빌리티 서비스로의 확대 등 관련 비즈니스는 혁명의 시대로 접어들게 한다[1,2].

특히 자율주행 기술의 핵심은 효율적이고 안전한 경로계획에 있으며 이를 위해서는 여러 센서를 통한 정확한 상황인지가 필수적이다. 하지만 이런 정확한 상황인지 기술은 안전한 주행 결정을 위해 실시간 처리 역시 중요하기 때문에, 주요 전기차 회사들은 이를 위해 카메라만을 이용한 2D, 3D 상황인지 기술개발을 진행하고 있다[1,2]. 하지만 최근의 컴퓨터 비전 기술의 비약적 발전에도 불구하고 카메라만으로는 탐지하지 못하는 다양한 희귀한 경우들이 있으며 가까운 미래에 이 어려움을 극복하기 힘들 것이라는 전망 또한 지배적이다.

이에 대부분의 자율주행차 회사들은 라이다 기반의 포인트 클라우드 데이터를 통한 3D 상황인지 방법을 대안으로 활용하며 관련 기술개발을 진행하고 있다[1,2]. 대표적인 기술로는 3D Object Detection을 위한 PointPillars[3], PointRCNN[4], VoxelNet[5] 등이 있으며 3D Semantic Segmentation을 위한 2DPASS[6], RangeFormer[7] 등이 있다. 하지만, 이런 기술들이 개발되었음에도 현재 정확도는 KITTI car moderate 데이터셋 기준으로 Object Detection 최대 AP (Averrage Precision)이 83% 정도 수준이고 semanticKITTI 데이터셋 기준으로 semantic segmentation 최대 mIoU가 74% 정도 수준이어서 자율주행의 안전을 보장할 만큼이라고 보기는 힘들다. 이렇게 영상인식 정확도가 만족스럽지 못한 수준인 이유는 모델이 아직 충분히 발전되지 않은 것도 있지만 Lidar 데이터를 획득하고 Labeling 하는 비용이 높아서 Labeling 된 데이터의 양이 충분치 못한 면도 있다. 그래서 이를 해결하기 위한 Unsupervised Learning[8], Semi-Supervised Learning[9,10] 이 도입되어 연구가 추진되고 있지만, 성능 향상 정도는 아직 상용화 수준으로 발전되지는 못한 것이 사실이다. 이에 본 연구에서는 데이터양을 좀 더 늘리기 위해 포인트 클라우드 데이터를 위한 다양한 데이터 증식기법들을 알아보고 이들의 개별 성능, 조합 성능들을 분석하여 최적의 포인트 클라우드 데이터 증식 전략을 제시하고자 한다.

## Ⅱ. 관련 연구

포인트 클라우드 데이터는 3차원 공간을 정해진 수의 레이저 센서의 반사 신호를 통해 파악해야 한다. 그러므로 그 분포가 매우 sparse 하며 비정규적인 패턴을 가지며 기하학적인 면에서 비 순서적 특성을 가진다. 결과적으로 기존 CNN 기반의 영상인식 딥러닝 기술들에 직접 적용하기 어려운 특성을 보이므로, 이를 해결하기 위한 3가지 접근법 즉, Voxel 기반 접근법, Point 기반 접근법, Point-Voxel 기반 접근법이 존재한다.

우선 Voxel 기반 접근법은 비정규적인 패턴의 Point 데이터를 2D/3D Grid들로 Voxel 화하여 Pixel 기반의 CNN과 유사하게 voxel 기반 CNN 기술을 적용하는 방식이다[3,5,11]. 연구 [5]에서는 Voxel 화 된 Point Cloud Feature 벡터에 3D CNN을 적용하였다. 하지만 해당 voxel 기반 접근법은 상당히 sparse 하게 분포하는 포인트 클라우드의 특성을 반영하지 못해 불필요한 공간까지 voxel 화하는 비효율성을 가지게 된다. 이에 연구 [11]에서는 sparse convolution을 적용하여 불필요한 3차원 공간 연산을 제거함으로써 효율적이며 좀 더 정확한 객체검출 성능을 달성하였다. 더 나아가 연구 [3]에서는 기존 Voxel 단위의 연산에서 축 단위로 길게 늘어뜨린 Voxel Pillar 단위로 연산함으로써 각 voxel이 가지는 sparse 한 특성을 극복한 representation을 통해 객체 인식 성능을 개선하고자 하였다.

반면 Point 기반 접근법은 포인트 클라우드에 Voxel 화 와 같은 전처리 없이 순서에 무관하며 어떤 양자화 처리 없이도 있는 그대로의 데이터들을 활용하여 객체 인식을 수행한다[4,12,13]. 연구 [4]에서는 2단계로 이루어지는데 1단계에서는 bottom-up 방식으로 encoder-decoder 구조를 통해 원본 포인트 클라우드 데이터에서 Point Feature Vector를 생성하고 이에 MLP를 통해 3D Proposal을 생성한다. 그리고 2단계에서는 생성된 3D Proposal에서 새로운 MLP와 encoder를 거쳐 Box Refinement를 수행하여 최종 객체 인식을 수행한다.

Point-Voxel 기반 접근법은 Point 기반 접근법과 Vogel 기반 접근법을 모두 사용하는 방법으로 각 방식의 장점들을 극대화할 수 있는 방법이다[14-16]. 연구 [14]에서는 원본 포인트 클라우드 데이터에서 Point 기반 접근법처럼 encoder-decoder 과 같은 신경망 구조를 통해 Point Feature Vector를 생성하고 여기에 Voxelization을 통해 생성된 Voxel에 다른 VoxelNet[5]

의 VFE (Voxel Feature Encoding) 방식을 사용하여 최종 객체 인식을 수행한다.

다음 그림 1에서는 KITTI Car moderate을 기준으로 한 각 3D object detection 기술별 성능을 나타내었다.

그림 1에서 확인할 수 있듯이, 이런 다양한 기술의 발전에도 불구하고 상용에서 쓸 수 있을 만큼의 만족스러운 성능은 나타내지 못하는 것을 알 수 있다. 이는 이런 포인트 클라우드 데이터의 객체 인식에 불리한 조건 때문인데, 즉, 고비용, 고사양 채널 (64ch, 128ch, 256ch) 라이다라 할지라도 그 취득 신호 양은 카메라에 비해 매우 적으며  심지어 비어 있는 공간도 많고 일정한 패턴을 지니고 있지도 않으며, occlusion으로 인해 손실되는 신호가 많으며 제조사에 따라 신호 특성이 달라지기 때문이다.

그러므로 이에 대한 적절한 데이터 전처리 및 Data Augmentation은 정확도 향상에 필수적이지만, 이런 Data Augmentation에 대한 연구는 지금까지 많이 연구되지는 않았다[17-20]. 연구 [17]에서는 포인트 클라우드 데이터에 평행 이동을 통해 얻어지는 증강데이터로 훈련했을 경우 원본 데이터와 증강데이터의 데이터 비율에 따른 성능을 분석하였다. 연구 [18] [19]에서는 다른 포인트 클라우드 데이터의 객체들 (예, 자동차, 사람)을 가져와서 또 다른 포인트 클라우드 데이터에 합성하는 방식의 Data Augmentation 방식을 제안하였으며 이를 통해 평균적으로 2-3 정도의 AP 성능 향상을 도출하였다. 연구 [20]에서는 원본 포인트 클라우드 데이터에서 다운샘플된 데이터를 통해 훈련할 경우 약 0.2 정도의 AP (Average Precision)

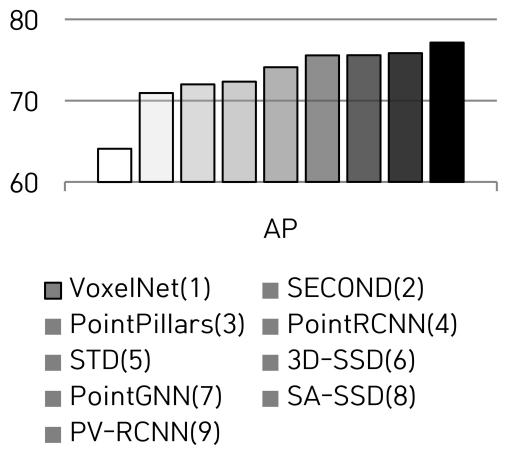



그림 1. 각 3D Objcet Detection 기술별 AP 성능 표
Fig. 1. AP performance table for each 3D Object Detection Models

성능 향상을 도출하였다.

이처럼 해당 연구들[17-20]은 일부 데이터 증식 방법들에 대한 성능을 분석하는 데 중점을 두고 있으며 포인트 클라우드 데이터 증식기법들의 개별 성능 기여도에 대한 정량적 분석이나 이들 증식기법의 조합을 통한 최적의 증식기법 전략에 관해 다룬 연구는 없었다. 본 연구는 이를 위해 고전적 데이터 증식기법들인 Jitter, Uniform Sampling, Random Sampling, Scaling 방식들과 데이터 합성 기반의 데이터 증식기법들을 사용하여 개별 성능 기여도, 조합 성능 기여도를 분석하여 최적의 포인트 클라우드 데이터 증식 전략을 제시하였다.

## Ⅲ. 시스템 모델

본 논문에서 사용된 시스템모델을 위해 3.1에서는 가장 널리 사용되는 3D Object Detection 모델 PointPillars에 대해 설명하고 3.2에서는 해당 모델로부터 도출되는 3D 객체 인식 정확도 지표 인 AP, mAP를 연산을 위한 3D BBox Label, 3D 좌표계, 3D IoU 계산법을 정의한다.

### 3.1 PointPillars

그림 2처럼 2번째 Backbone 단계에서 2D Conv를 사용하는데 이는 3D Conv보다 연산 시간을 줄이기 위해 사용된다. 이를 적용하기 위해 Pillar Feature Net 단계에서 Pointcloud를 sparse pseudo-image로 변환하고 2D CNN을 거쳐 고차원의 feature를 추출한다. 그리고 Single Shot Detection 단계를 거쳐 앞선 과정에서 생성된 2D feature map을 기반으로 object의 위치와 클래스를 detection하며, 이 정보를 사용하여 3D Pointcloud로 reconstruction 된다.

### 3.2 3차원 객체 인식 좌표계 및 정확도 계산

3차원 객체 인식을 위한 좌표계로써 KITTI 데이터셋의 좌표계를 기준으로 하였다. 다음 표 1에서는 KITTI GT의 Annotation을 세부적으로 정의한다.

표 1에 대한 예시로 그림3은 발견된 Object의 type 속성 정보부터 Truncation, Occlusion, ..., Rotation_Y 값을 세부적으로 나타낸다.

#### 3.2.1 Type

KITTI에서 제공한 3D Object Detection Dataset 중에는 다양한 Class들이 존재하며 Type들은 가장 첫 번째에 표시된다. Class로는 Car, Van, Truck,

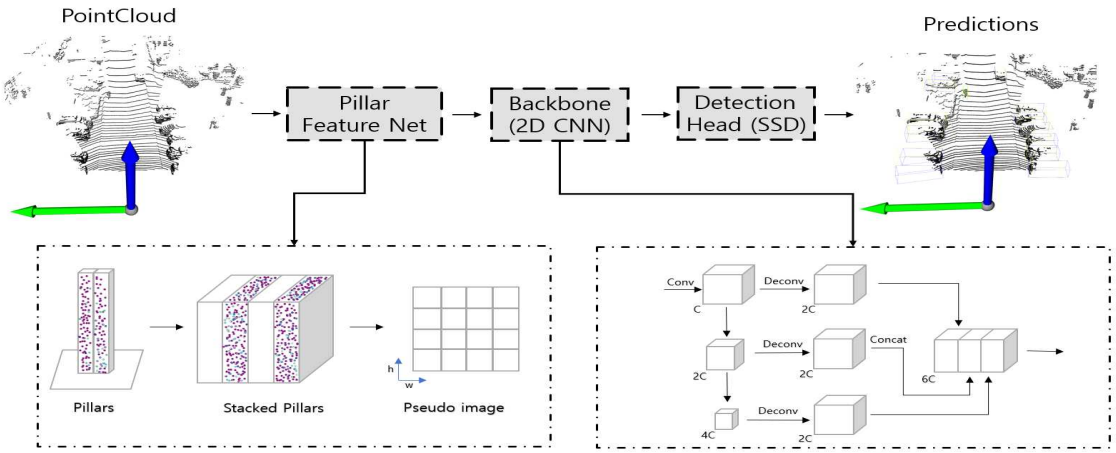



그림 2. 자율주행을 위한 포인트 클라우드 3차원 객체 인식 시스템 PointPillars 개요도 [3]
Fig. 2. System Architecture of Point Cloud 3D Object Detection, PointPillars, for Autonomous Driving

표 1. KITTI 정답 레이블 세부 정보
Table 1. KITTI GT Annotation Details.

| 총 자리수 | 속성 | 설명 |
|---|---|---|
| 1 | Type | 객체 클래스: 'Car', 'Van', 'Pedestrian', 'Person_sitting', 'Cyclist', 'Tram', 'Misc' or Dontcare' |
| 1 | Truncation | 0-1 범위 소수, 객체가 이미지 경계를 넘어간 정도 |
| 1 | Occlusion | Occlusion 정도를 0, 1, 2, 3의 정수로 표현 |
| 1 | Alpha | 객체의 관찰 각도, [π, -π] |
| 4 | 2D Box | 객체의 2D Bounding Box |
| 3 | Dimension | 객체의 3D 크기 (meter), height, width, length로 표현 |
| 3 | Location | 카메라 좌표계의 3D 객체 위치 |
| 1 | Rotation_Y | 카메라 좌표계에서 Y축 상의 회전 각도, [π, -π] |
| 1 | Score (생략가능) | Object Detection에서 객체의 존재 확률값 |

Pedestrian or Person, Bus…. , Dontcare 등이 있다.

3.2.2 Truncation

'Truncation'이란 3D 객체가 bbox의 경계 이탈 정도를 나타내며 값이 0.0이면 object가 bbox 안에 있고, 1.0이면 object가 bbox 경계를 벗어난 것이다.

3.2.3 Occlusion

'Occlusion'이란 3D 객체가 다른 객체나 장애물에 의해 얼마나 가려져 있는지를 나타내는 값으로 '0' = 완전 식별 가능, '1' = 부분 식별 가능, '2' = 절반 이하 식별 가능, '3' = 식별 불가 로 구분된다.

3.2.4 Alpha

'Alpha'는 3D 객체의 중심이 이미지에서 어디에 위치하는지를 나타내는 값이다. 이 값은 이미지의 좌측 끝에서부터 시계 방향으로 측정되며, '-π' ~ '+π'의 범위를 갖는다. 이를 통해 객체의 위치와 방향을 파악할 수 있다.

3.2.5 2D bbox

'2D bbox'는 3D 객체의 2D 투영 bbox를 나타내며 좌상단, 우하단 좌표값을 통해 나타낸다.

3.2.6 Dimension

'Dimension' 은 3D 객체의 Width, Height, Length를 나타내며 이를 통해 객체의 실제 크기를 예측할 수 있다.

3.2.7 Location

'Location'은 카메라 좌표값에서의 객체 위치를 나타낸다. 이를 통해 객체의 실제 위치를 예측할 수 있다.

3.2.8 Rotation_y

'Rotation_y'는 3D 객체의 방향 각도를 나타내며, 3D 공간에서 객체가 앞을 바라보면 '+' 부호, 마주 보는 방향이면 '-' 부호로 표현하며 Radian 단위로 나타낸다.

3.2.9 Score

'Score'는 Object Detection에서 객체의 존재 확률을 나타낸다.

다음 표 2에선 차량에 탑재된 센서의 X, Y, Z 축의 방향을 나타낸다. Camera의 X축은 우측, Y축은 하단, Z축은 정면을 나타내며, LiDAR와 GPS / IMU의 X축은 우측, Y축은 정면, Z축은 상단을 나타낸다. 연구 [21]에서는 차량에 2대의 Color Camera와 2대의 Gray scale Camera를 사용하였으며, 1대의 LiDAR, 1대의 GPS 센서를 탑재하였다. 이렇게 6대의 센서 간의 좌표값을 변환해 주어야 한다.

위 그림 4는 각 센서의 좌표 변환 과정을 나타낸 그림이다. 기준 카메라(cam_reference)와 Cam2처럼 카메라 간의 좌표값 변환을 해주며, 라이다와 카메라 간의 좌표값 변환도 해준다.

센서 간의 좌표값 변환의 세부 과정은 다음 표 3을 통해 각 인자의 역할을 알아야 한다. $P(i)$는 i번째 카메라에서 이미지로 투영되는 투영 행렬로 중심 카메라인 '0'번부터 '3'번 카메라까지 4대의 카메라에 대한 투영 행렬을 나타낸다. $R(0)$는 image rectification

표 2. 센서들의 각 Axis 방향
Table 2. Sensor Axis Direction.

| Sensor | X-Axis Direction | Y-Axis Direction | Z-Axis Direction |
|---|---|---|---|
| Camera | Right | Down | Forward |
| LiDAR | Forward | Left | Up |
| GPS/IMU | Forward | Left | Up |

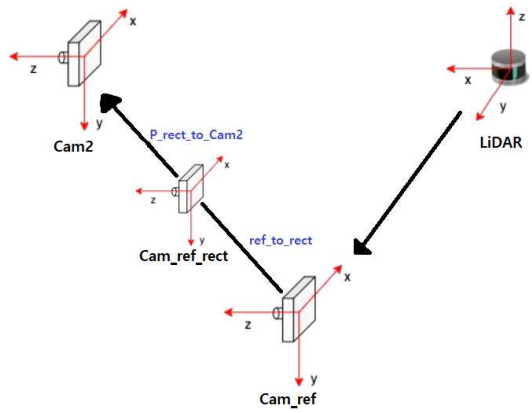
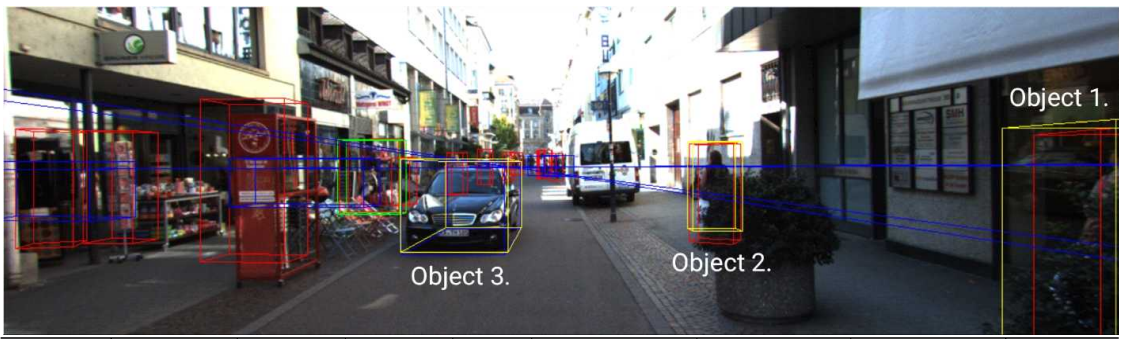



그림 4. 센서 간 좌표 변환 세부 정보
Fig. 4. Coordinate Transformation Details

행렬로 같은 객체에 대해 서로 다른 위치에서 촬영한 여러 이미지가 각각 다르게 인식되는데 이를 보정 하기 위한 행렬을 나타내며, $T_{L2C}$는 라이다 좌표에서 카메라 좌표로 변환하기 위한 행렬을 나타낸다. $x$는

표 3. 센서 좌표 변환을 위한 요소
Table 3. Elements for sensor coordinate transformation.

| | |
|---|---|
| $P(i)$ | i번째 카메라에서 이미지로 투영되는 Projection matrix |
| $R(0)$ | Rectification matrix : transformation process used to project images onto a common image plane |
| $T_{L2C}$ | 라이다 좌표에서 카메라 좌표로의 Transformation Matrix |
| $x$ | Matrix multiplication operation |




| 속성 | Type | Truncation | occlusion | alpha | 2D BBox | dimension | location | rotation_y |
|---|---|---|---|---|---|---|---|---|
| **Object 1.** | Pedestrian | 0.63 | 3 | -2.29 | 1104.62, 128.70, 1237.00, 373.00 | 1.16, 0.91, 0.91 | 3.78, 1.32, 4.33 | -1.61 |
| **Object 2.** | Pedestrian | 0.00 | 2 | -1.78 | 772.52, 152.22, 816.75, 255.07 | 1.65, 0.88, 0.49 | 3.04, 1.18, 11.82 | -1.54 |
| **Object 3.** | Car | 0.00 | 0 | 1.71 | 438.09, 172.68, 575.89, 279.55 | 1.49, 1.68, 4.08 | -1.44, 1.37, 12.14 | 1.59 |

그림 3. 객체별 속성 정보
Fig. 3. attributes by object

카메라 좌표와 라이다 좌표를 나타낸다.

다음 수식(1), (2), (3)은 Camera Calibration과 LiDAR & GPS/IMU Calibration의 과정을 나타낸다.

$$y = P_{rect}^{(i)} x$$
$$x = (x, y, z, 1)^T \rightarrow \text{카메라 좌표계},$$
$$y = (u, v, 1)^T \rightarrow \text{정규 좌표계}$$
$$P_{rect}^{(i)} = \begin{pmatrix} f_u^{(i)} & 0 & c_u^{(i)} & -f_u^{(i)} b_x^{(i)} \\ 0 & f_v^{(i)} & c_v^{(i)} & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad (1)$$

$f^{(i)}$= Focal Length, $c^{(i)}$= Centroid, $b_x^{(i)}$= Cam 0

$$y = P_{rect}^{(i)} R_{rect}^{(0)} x$$

카메라의 좌표를 통해 이미지 평면에 투영하려면 먼저 카메라 좌표계를 정규좌표계에 투영해야 한다. 여기서 $x$는 카메라의 3D 좌표를 나타내며, $P_{rect}^{(i)}$는 투영 3 by 4 행렬로 카메라의 intrinsic은 3가지로 표현한다. $f^{(i)}$는 카메라의 초점 거리, $c^{(i)}$는 카메라의 중심 좌표, $b_x^{(i)}$은 i번째 카메라를 뜻한다. $R_{rect}^{(0)}$는 각 카메라의 rectification을 위한 행렬을 나타낸다. 이를 통해 정규좌표계에 투영 후 이미지 평면에 투영하려면 행렬의 크기를 맞춰 주어야 하므로 $R_{rect}^{(0)}(4,4)$ = 1을 추가하여 4 by 4 행렬로 확장해야 한다.

$$t_{velo}^{cam} \in R^{1\times 3}, \quad R_{velo}^{cam} \in R^{3\times 3} \quad (2)$$
$$T_{velo}^{cam} = \begin{pmatrix} R_{velo}^{cam} & t_{velo}^{cam} \\ 0 & 1 \end{pmatrix}, \quad y = P_{rect}^{(i)} R_{rect}^{(0)} T_{velo}^{cam} x$$

다음 수식(2)은 라이다와 카메라로 측정한 객체의 3D location과 2D location을 맞춰 주기 위한 변환 과정으로 $R_{velo}^{cam}$은 회전 변환 행렬이고 $t_{velo}^{cam}$는 이동 변환 행렬로 라이다의 extrinsic 행렬을 나타낸다. 여기서 $x$는 라이다 좌표로 $T_{velo}^{cam}$를 통해 회전 및 이동 변환을 해주고 $R_{rect}^{(0)}$를 통해 라이다와 카메라로 촬영한 data를 보정 해준 뒤 이미지 평면에 투영한다.

$$y = P_{rect}^{(i)} R_{rect}^{(0)} T_{velo}^{cam} T_{imu}^{velo} x \quad (3)$$

위 수식(3)은 수식(2)과 같은 방법으로 자동차의 움직임을 보정해주는 과정을 나타낸다.

### 3.3 3차원 객체 IoU 및 AP 계산

KITTI 데이터셋의 3D Object Detection 성능 계산은 2D 와 마찬가지로 Precision-Recall 곡선을 사용하여 객체 검출 성능을 분석한다. 3D Object Detection에서는 3D IoU를 사용하여 예측된 bbox와 Ground Truth bbox의 겹치는 영역을 계산하고, 이를 통해 정확한 성능을 측정한다. 이를 위해 Ground Truth 레이블과 예측 결과에 대한 속성 정보 Type부터 Truncation, Occlusion, ..., Rotation_Y 값을 얻는다.

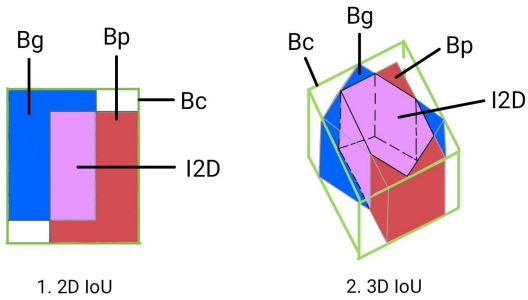



그림 5. 2D & 3D 바운딩 박스 교집합
Fig. 5. 2D & 3D IoU of bbox

위 그림 5는 기존의 2D IoU를 통해 3D IoU를 계산하는 그림을 나타낸다. $B_g$는 Ground Truth bbox를 나타내며 $B_p$는 예측된 bbox, $B_c$는 두 영역의 최소 enclosing box, $I_2D$는 두 bbox의 겹치는 영역을 나타낸다. 2D bbox에서 3D bbox를 생성할 때 Truncation, Occlusion 값에 따라 객체의 크기와 위치를 보정하고 Alpha 값을 통해 객체의 회전 각도를 반영하여 3D 평면에 투영한다. Dimension, Location, Rotation_y값을 통해 객체의 높이, 너비, 길이, 중심 위치, 회전 정보를 반영하여 3D Groun Truth bbox와 예측된 bbox의 IoU를 구할 수 있다.

그러면 이 IoU 값에 Threshold를 객체별로 설정하여TP, FN, FP, TN 값을 얻어내어 Precision-Recall 곡선을 계산하고 이를 통해 클래스 별 AP 및 모든 클래스에 대한 평균 mAP 값을 얻는다.

## Ⅳ. 데이터 증식

3D Object Detection의 정확도 향상을 위해 그림6의 4가지의 데이터 Augmentation 방법을 사용하였다. 이를 위한 세부 요소 기술은 다음과 같다.

### 4.1 Jitter

'Jitter'는 pointcloud의 좌표값에 변이를 주어 증식시키는 기법으로 각 좌표에 작은 노이즈나 랜덤한 변동을 추가함으로써 이루어진다.

### 4.2 Uniform DownSample

'Uniform DownSample'은 일정한 간격의 일부 점들을 선택하여 데이터의 크기를 줄이는 기법이다. 이를 통해 계산 부담을 줄이고 모델의 속도를 높일 수 있지만, 일부 점들 사이의 정보가 손실될 수 있다.

### 4.3 Random DownSample

'Random DownSample'은 일부 점들을 무작위로 선택하여 데이터의 크기를 줄이는 기법이다. 이 역시 계산 부담을 줄여주지만, 중요한 정보가 손실될 수 있다.

### 4.4 Scale

'Scale'은 pointcloud의 전체적인 크기를 변화시키는 기법으로 점들의 좌표값을 일정 비율로 확대하거나 축소하여 데이터를 다양하게 만든다. 이는 물체의 크기 변화에 대한 모델의 불변성을 해결하는 데에 도움을 줄 수 있다.

## V. 실 험

본 논문에서는 KITTI object detection benchmark Dataset을 사용하여 실험하였다. Original KITTI Dataset 은 7,518개의 test Dataset 과 7,481개의 train Dataset으로 나뉘어 있는데, PointPillars[3]에선 train Datase을 3,712개의 train samples와 3,769개의 validation samples로 나누어 실험하였다. Jitter,

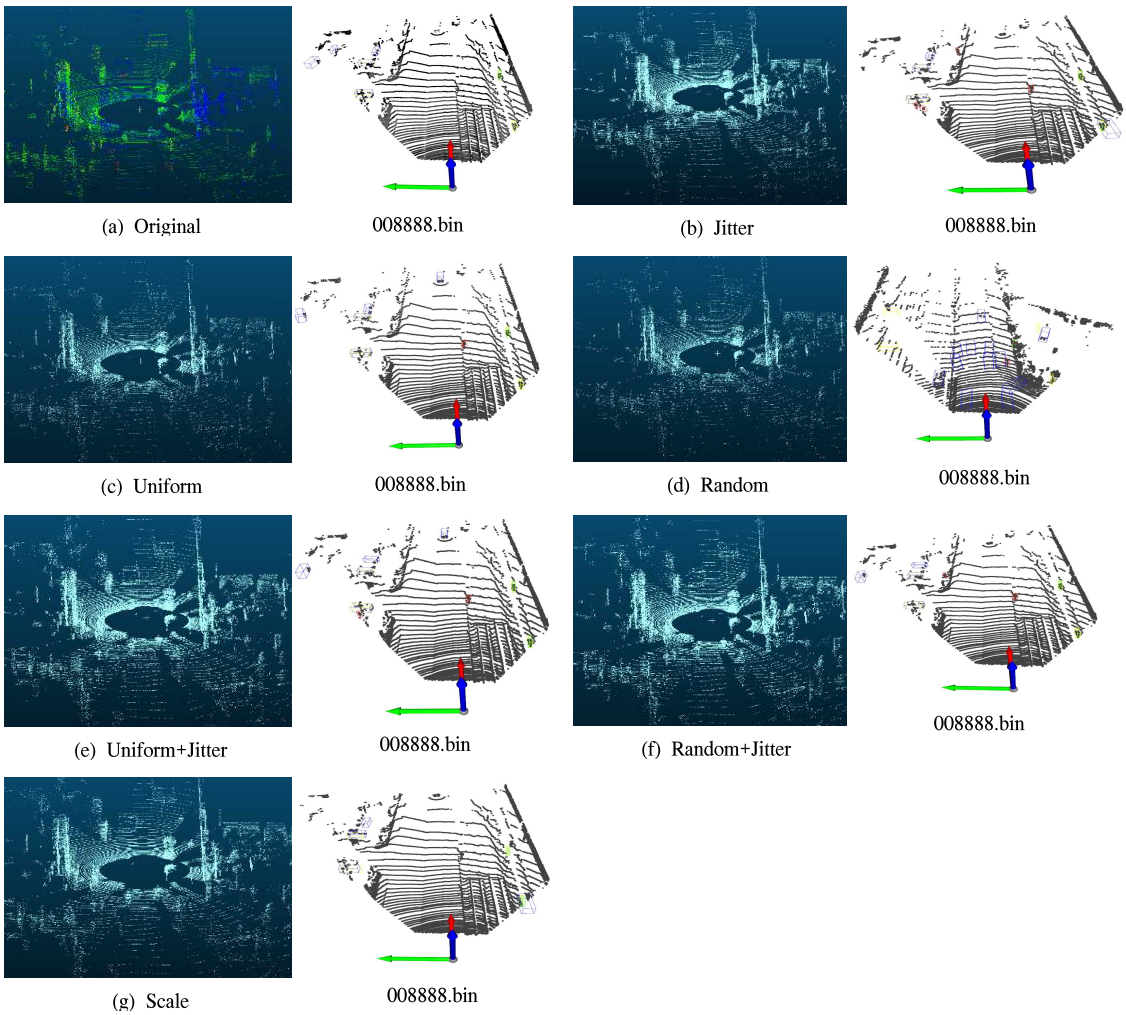



그림 6. 포인트 클라우드 3차원 객체 인식을 위한 데이터 증식
Fig. 6. Data Augmentation for Point Cloud based 3D Object Detection.

Scale, Uniform DownSample, Random DownSample, Uniform + Jitter, Random + Jitter 기법 모두 train samples를 1:1 비율로 설정하여 7,424개의 train samples와 3,769개의 validation samples로 나누어 훈련하였다.

표 4는 3가지 객체별 Easy, Moderate, Hard 부분에 대한 6가지 데이터 Augmentation의 성능을 나타낸다. 실험에서는 Point Cloud의 Augmentation 기술들(Jitter, Uniform DownSampling, Random DownSampling, Scale)을 적용하였을 때에 KITTI 데이터 셋의 클래스별 AP(Average Precision)를 측정하였다. 클래스별 data 수가 많은 Car는 IoU 값을 0.7로 설정하였고 비교적 적은 data 수의 Cyclist와 Pedestrian은 IoU를 0.5로 설정하여 실험을 진행하였다. Easy, Moderate, Hard의 기준은 truncation값과 occlusion으로 나눌 수 있다. Easy는 truncation 값인 예측 bbox가 정답 bbox 보다 벗어난 정도를 15%로 설정하였다. Moderate는 truncation 값을 30%, Hard는 truncation 값을 50%로 설정하였다. 또한, Easy의 occlusion은 '0'으로 완전 식별 가능으로 설정하였고 Moderate는 '1', Hard는 '2'로 설정하여 실험을 진행하였다.

Pedestrian에선 6가지 기법 모두 Easy, Moderate, Hard 부분에서 원본보다 성능이 낮아지는 것을 확인할 수 있는데, 모든 클래스에서 성능이 낮아진 기법은 Random DownSample과 Scale이 있었다. Randown sample은 무작위로 점들을 선택하여 데이터를 줄이는 방법으로 정보 손실이 발생하고 Scale은 포인트 간의 거리를 늘려 전체적인 맵을 확장 시키다 보니 Random DownSample과 동일하게 정보 손실이 발생하여 객체 인식을 제대로 하지 못하는 것을 알 수 있었다. 반면에 Cyclist에서 성능이 높아진 기법은 Jitter와 Uniform DownSample 기법이 있다. Uniform DownSample은 Moderate 부분에서 성능이 원본보다 0.8 정도 높아진 것을 확인할 수 있고 Jitter는 Easy, Moderate, Hard 부분 모두 성능이 0.5~0.6 소폭 향상된 것을 확인할 수 있다. 향상 원인을 분석해 본 결과 Uniform DownSample은 일정한 간격의 점들을 선택하여 데이터의 크기를 줄이는 방법으로 정보 손실이 일어날 수도 있다. 하지만, 무작위로 데이터양을 줄이는 Random DownSample에 비해 원본 data에 대한 정보 손실은 크지 않는 것을 알 수 있었다. Jitter는 점들의 개수를 원본의 2배 또는 그 이상으로 증가시키는 기법으로 객체를 더 선명하게 해주는 효과가 있어서 객체 인식에 도움이 되는 것을 알 수 있었다. Car에서 성능이 높아진 기법 역시 Jitter와 Uniform DownSample이다. Jitter는 Easy, Moderate 부분에서 0.5~0.6 향상되었고, Uniform DownSample은 Moderate 부분에서 0.02 소폭 향상된 것을 확인할 수 있다. 이를 통해 Point Cloud 3D Object Detection 기술을 위한 데이터 증식기법은 Jitter 와 Uniform DownSample을 위주로 하되 Car와 Cycle 클래스들에 한정적으로 적용하는 것이 전체 AP성능 향상에 도움이 될 수 있다는 것을 알 수 있다.

## Ⅵ. 결 론

본 논문에서는 향후 커스텀 데이터 셋을 구축할 때 센서들 간의 정보 변환을 위해 Calibration 과정을 세부적으로 파악하였고, 정확한 객체 인식을 위해 객체

표 4. KITTI 테스트 3D 검출 결과
Table 4. Results on the KITTI test 3D detection.

| BBOX_3D | Car(IoU = 0.7) | | | Cyclist(IoU = 0.5) | | | Pedestrian(IoU = 0.5) | | |
|---|---|---|---|---|---|---|---|---|---|
| | Easy | Moderate | Hard | Easy | Moderate | Hard | Easy | Moderate | Hard |
| Original | 86.6348 | 76.7511 | 74.1702 | 81.8677 | 63.6617 | 60.9022 | 51.4595 | 47.9421 | 43.8050 |
| Jitter | 87.2811 | 77.3158 | 74.1093 | 82.4805 | 64.3629 | 61.4332 | 47.4443 | 43.5501 | 39.0476 |
| Uniform (0.5) | 86.2287 | 76.7708 | 73.8777 | 81.0273 | 64.4466 | 60.3510 | 47.9338 | 43.5965 | 38.5858 |
| Random (0.5) | 85.3115 | 75.9889 | 73.4751 | 77.0455 | 61.3112 | 58.8526 | 50.4546 | 44.6814 | 40.5198 |
| Uniform(0.9) + Jitter | 85.9516 | 76.6039 | 74.0007 | 81.8220 | 63.6605 | 60.2763 | 45.3127 | 41.7275 | 37.1625 |
| Random(0.9) + Jitter | 86.1892 | 76.5855 | 73.7485 | 82.1111 | 62.8816 | 60.0747 | 47.8165 | 43.8841 | 39.0488 |
| Scale | 85.2639 | 75.9625 | 73.7198 | 80.6946 | 62.2883 | 59.3586 | 50.3248 | 45.3508 | 41.7913 |

의 속성 정보에 대해 알아보았다. 또한, 데이터양을 늘리기 위해 포인트 클라우드 기반의 다양한 데이터 증식기법들을 알아보고 각 기법의 개별 성능, 조합, 성능들을 분석하여 최적의 포인트 클라우드 데이터 증식 전략을 제시하였다. 여러 기법 중 성능이 낮아지는 기법이 있었는데, 이는 정해진 map 전체에 변화를 주는 기법으로 정보 손실이 발생하여 성능이 낮아지는 것을 알 수 있었다. 반면에 포인트 클라우드 개수 기반 기법인 Jitter가 가장 효과적이었으며 2가지 DownSample 기법 중 무작위로 포인트 클라우드의 크기를 줄이는 기법보다 균일하게 크기를 줄이는 기법이 정보 손실이 더 적어 미세한 성능 향상이 있었다는 것을 알 수 있었다.

## References


[1] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, “A survey of deep learning techniques for autonomous driving,” *J. Field Robotics*, vol. 37, no. 3, pp. 362-386, 2020. (https://doi.org/10.1002/rob.21918)

[2] D. Kwak, J. Yoo, M. Son, M. Park, D. Choi, and S. Lee, “Rethinking real-time lane detection technology for autonomous driving,” *J. KICS*, vol. 48, no. 5, pp. 589-599, 2023. (https://doi.org/10.7840/ kics.2023.48.5.589)

[3] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, “PointPillars: Fast encoders for object detection from point clouds,” *CVPR 2019*, pp. 12697-12705, 2019. (https://doi.org/10.48550/arXiv.1812.05784)

[4] S. Shi, X. Wang, and H. Li, “PointRCNN: 3D object proposal generation and detection from point cloud,” *CVPR 2019*, pp. 770-779, 2019. (https://doi.org/10.48550/arXiv.1812.04244)

[5] Y. Zhou and O. Tuzel, “VoxelNet: End-to-End learning for point cloud based 3d object detection,” *CVPR 2018*, pp. 4490-4499, 2018. (https://doi.org/10.48550/arXiv.1711.06396)

[6] X. Yan, J. Gao, C. Zheng, C. Zheng, R. Zhang, S. Cui, and Z. Li, “2dpass: 2d priors assisted semantic segmentation on lidar point clouds,” in *Eur. Conf. Comput. Vision*, pp. 677-695, 2022. (https://doi.org/10.1007/978-3-031-19815-1_39)

[7] L. Kong, Y. Liu, R. Chen, Y. Ma, X. Zhu, Y. Li, and Z. Liu, “Rethinking range view representation for lidar segmentation,” in *Proc. IEEE/CVF Int. Conf. Comput. Vision*, pp. 228-240, 2023. (https://doi.org/10.48550/arXiv.2303.05367)

[8] A. Xiao, J. Huang, D. Guan, X. Zhang, S. Lu, and L. Shao, “Unsupervised point cloud representation learning with deep neural networks: A survey,” *IEEE Trans. Pattern Anal. and Mach. Intell.*, 2023. (https://doi.org/10.48550/arXiv.2202.13589)

[9] Y. Su, X. Xu, and K. Jia, “Weakly supervised 3d point cloud segmentation via multi-prototype learning,” *IEEE Trans. Cir. and Syst. for Video Technol.*, 2023. (https://doi.org/10.48550/arXiv.2205.03137)

[10] H. Shi, J. Wei, R. Li, F. Liu, and G. Lin, “Weakly supervised segmentation on outdoor 4D point clouds with temporal matching and spatial graph propagation,” in *Proc. IEEE/CVF Conf. CVPR*, pp. 11840-11849, 2022. (https://doi.org/10.1109/CVPR52688.2022.01154)

[11] Y. Yan, Y. Mao, and B. Li, “Second: Sparsely embedded convolutional detection,” *Sensors*, vol. 18, no. 10, 3337, 2018. (https://doi.org/10.3390/s18103337)

[12] Q. Luo, H. Ma, L. Tang, Y. Wang, and R. Xiong, “3D-SSD: Learning hierarchical features from RGB-D images for amodal 3D object detection,” *Neurocomputing*, pp. 364-374, 378, 2020. (https://doi.org/10.1016/j.neucom.2019.10.025)

[13] W. Shi and R. Rajkumar, “Point-GNN: Graph neural network for 3d object detection in a point cloud,” *CVPR 2020*, pp. 1711-1719, 2020. (https://doi.org/10.48550/arXiv.2003.01251)

[14] C. Yuan, J. Lin, Z. Zou, X. Hong, and F. Zhang, “Std: Stabletriangle descriptor

for 3d place recognition," *IEEE ICRA*, pp. 1897-1903, 2023.
(https://doi.org/10.48550/arXiv.2209.12435)

[15] S. Shi, C. Guo, L. Jiang, Z. Wang, J. Shi, X. Wang, and H. Li, "PV-RCNN: Point-Voxel feature set abstraction for 3d object detection," *CVPR 2020*, pp. 10529-10538, 2020.
(https://doi.org/10.48550/arXiv.1912.13192)

[16] C. He, H. Zeng, J. Huang, X.-S. Hua, and L. Zhang, "Structure aware single-stage 3d object detection from point cloud," *CVPR 2020*, pp. 11873-11882, 2020.
(https://doi.org/10.1109/CVPR42600.2020.01189)

[17] X. Santos, P. Georgieva, P. Girão, and M. Drummond, "Data augmentation in 3D object detection for self-driving vehicles: The role of original and augmented training samples," *J. Physics: Conf. Series*, vol. 2407, no. 1, IOP Publishing, 2022.
(https://doi.org/10.1088/1742-6596/2407/1/012044)

[18] X. Hu, Z. Duan, and J. Ma, "Context-Aware data augmentation for lidar 3d object detection," *2023 IEEE ICIP*, pp. 11-15, 2023.
(https://doi.org/10.48550/arXiv.2211.10850)

[19] J. Fang, X. Zuo, D. Zhou, S. Jin, S. Wang, and L. Zhang, "LiDAR-Aug: A general rendering-based augmentation framework for 3d object detection," *CVPR 2021*, pp. 4710- 4720, 2021.
(https://doi.org/10.1109/CVPR46437.2021.00468)

[20] J. S. Hu and S. L. Waslander, "Pattern-aware data augmentation for lidar 3d object detection," *IEEE ITSC*, pp. 2703-2710, 2021.
(https://doi.org/10.48550/arXiv.2112.00050)

[21] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The Int. J. Robotics Res.*, vol. 32, no. 11, pp. 1231-1237, 2013.
(https://doi.org/10.1177/0278364913491297)

**정 영 재** (Youngjae Cheong)

2023년 2월 : 동서울대학교 전기정보제어과 졸업
2023년 3월~현재 : 동서울대학교 전자공학과 학사과정
<관심분야> 딥러닝, 영상인식, SLAM

**전 우 민** (Woomin Jun)

2021년 3월~현재 : 동서울대학교 전자공학과 전문학사과정
<관심분야> 딥러닝, 영상인식

**이 성 진** (Sungjin Lee)

2011년 8월 : 연세대학교 전자공학과 박사 졸업
2012년 9월~2016년 7월 : 삼성전자 DMC연구소 책임연구원
2016년 7월~현재 : 동서울대학교 전자공학과 조교수
<관심분야> 딥러닝, 영상인식, 3D Reconstruction,
[ORCID:0000-0003-3159-8394]