Project 4

Mapping and perception for an autonomous robot/ 0510-7591

By

Nimrod Curtis

Tel Aviv University

Spring 2023

Contents

Figures

1. Introduction

This project focuses on two fundamental components of computer vision:

- 3D object detection using PointPillars
- Multi-object tracking using BOT-SORT.

In Section A, we analyze the unfiltered output of PointPillars detector trained on KITTI dataset, studying objects like cars, cyclists, and pedestrians. We also explore an enhanced version that combines point cloud and image data for a comprehensive scene understanding.

In Section B, we install the BoT-SORT tracker repository and run an example on the MOT17 dataset. BoT-SORT is a powerful framework for multi-object tracking. By following instructions, we measure tracking performance using the TrackEval repository, gaining insights into interesting scenarios within the scene.

2. Solution

1. 3D Object Detection

a. Detection analysis based on PointPillars (PP) results on KITTI dataset

1. PointPillars Feature Net mechanism.

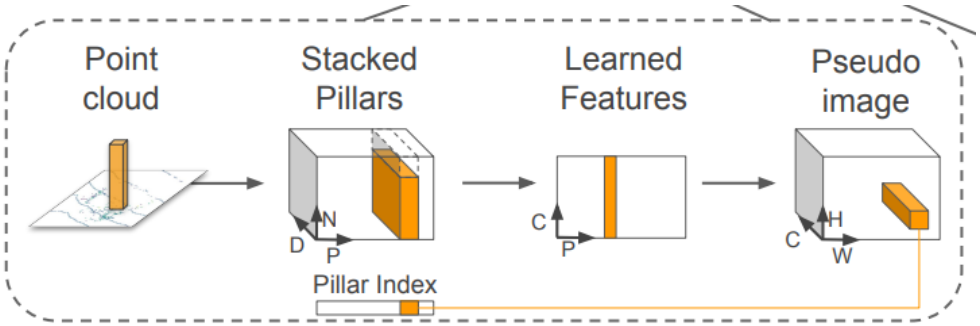


Figure 1 - Pillars Feature Net Architecture

In the PointPillars (PP) framework, the Feature Encoder, also known as the Pillar feature net, converts a 3D point cloud into a sparse pseudo-image. This conversion involves transforming the point cloud data into a grid structure of a specific size.

The mechanism used to convert the input from a 3D point cloud into a pseudo image involves several steps (see Figure 1) :

- Discretization (Voxelization) : In the initial stage, the point cloud undergoes a process of discretization where it is transformed into a grid with uniform spacing along the x-y plane. This transformation results in the creation of a set of pillars. Each pillar corresponds to a specific region in the x-y plane, and it encapsulates the points from the original point cloud that fall within that region.
- Embedding: Each point within a pillar is enhanced with additional coordinates and offsets, resulting in a 9-dimensional representation for further processing, to summarize – each augmented lidar point is D = [x,y,z,xc,yc,zc,xp,yp,r], where "c" indicates the distance to the mean of all points in the pillar, while the subscript "p" indicates the offset from the x, y center of the pillar , and r is the reflectance.
- Stacked Pillars: Next stage, is stacking the encoded tensors to a dense tensor based on the number of pillars per sample P and the number of points per pillars, creating a tensor of the size (D,P,N).
- Point-Net feature mapping: In the next step, a simplified version of PointNet is employed. For each point in the point cloud, a linear layer is applied, followed by BatchNorm and ReLU operations. This generates a tensor of size (C, P, N). Subsequently, a max operation is performed over the channels, resulting in an output tensor of size (C, P).

- Psuedo-Image: After the features are encoded, they are dispersed or scattered back to their original pillar locations. This process involves recreating a pseudo-image of size (C, H, W), where H and W represent the height and width dimensions of the canvas, respectively.

The pseudo-image generated by the Feature Encoder contains the encoded features extracted from the Point-Net. By representing the point cloud in the form of a pseudo-image, it becomes compatible with 2D convolutional neural networks (CNNs), allowing for efficient processing and analysis using standard image-based techniques.

2. Examples Analyzation

a. Example 1

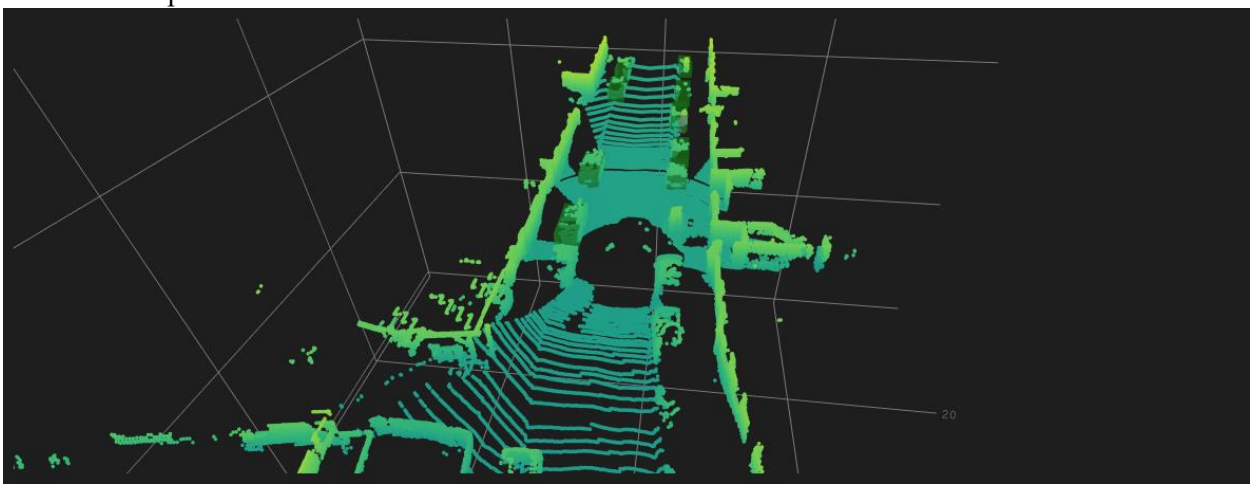


Figure 2 – Example 1 PCD

Figure 3 - Example 1 Image

The example depicts both the information contained in a point cloud and an accompanying image of a street. The image showcases various elements such as parked vehicles, buildings, and trees along the sides of the street.

When analyzing the detections of the network in relation to the results, several key things can be noticed:

- Successful detections of parked vehicles were achieved by considering factors such as distance and visibility. Vehicles that were close and not obscured received high confidence scores, indicating a

high likelihood of accurate detection. On the other hand, vehicles that were both distant and partially hidden, resulting in a lower and less organized number of points representing them, received relatively lower confidence scores. See the figure below for example:

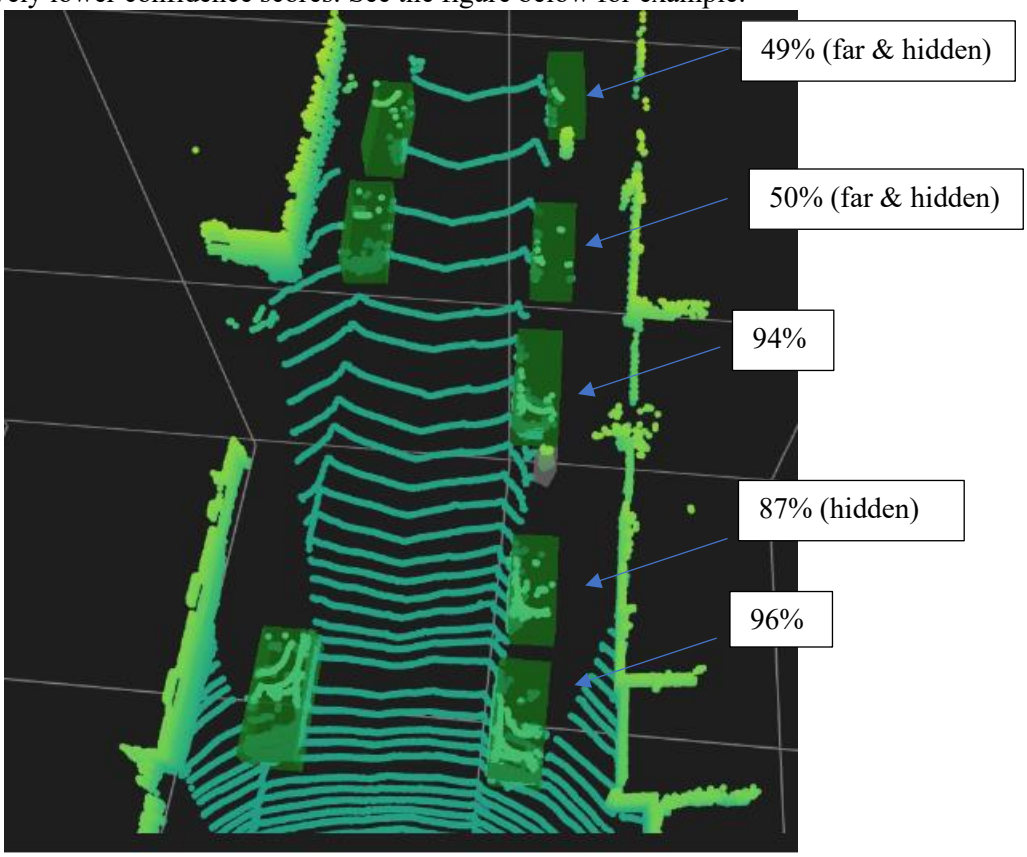


Figure 4 - Example 1 cars detection

- Regarding the identification of people, there was an instance where a misidentification occurred, mistakenly classifying a tree as a person (False Positive). This misidentification raised suspicion due to the low confidence score associated with it. This situation can arise because the point cloud representation of the tree is truncated at a specific height, likely due to filter parameter settings. Consequently, only the trunk of the tree is captured, and its thickness and height can lead to a misleading interpretation, potentially indicating the presence of a pedestrian. See the figure below.

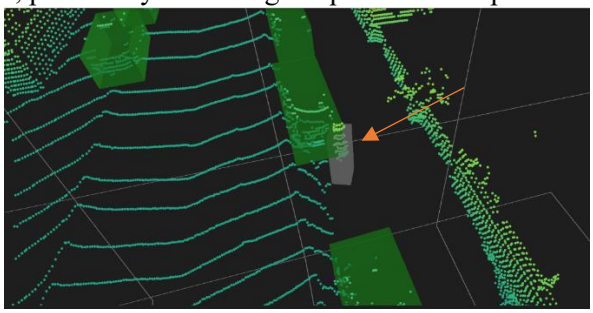


Figure 5 - Example 1 Pedestrian False Positive

Furthermore, there is an instance where a pedestrian on the left sidewalk of the street went undetected (False Negative), while he is still in the settings of search space (0-48meters, -20-

20meters) . Detect this pedestrian is particularly challenging, as even in the accompanying image, the pedestrian is obscured between two vehicles that are relatively far apart. Consequently, very few lidar points would be registered for this pedestrian, making their detection extremely difficult. See Figure 3 and the figure below.

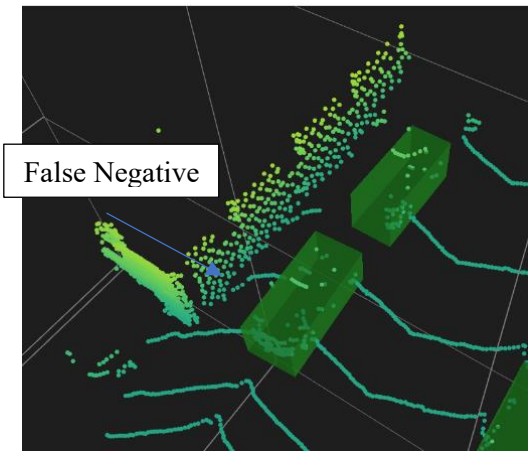


Figure 6 - Example 1 Pedestrian False Negative

b. Example 2

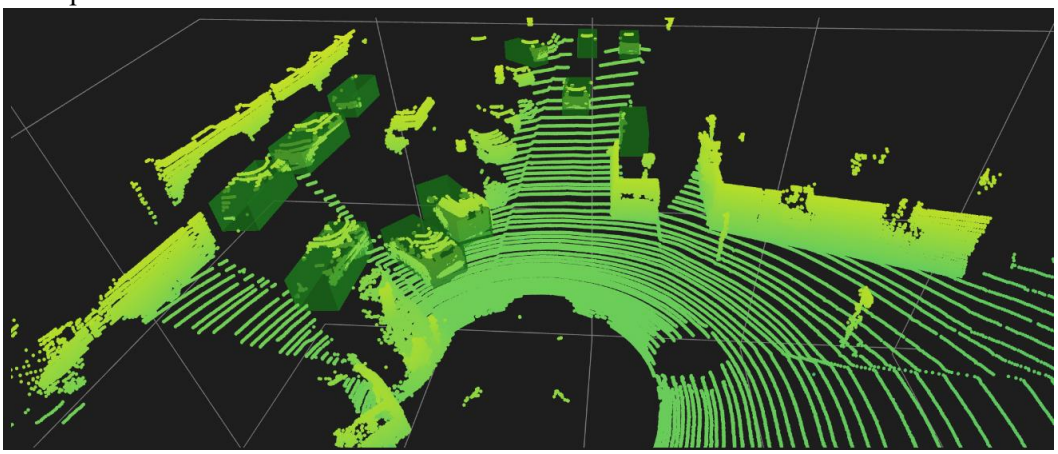


Figure 7 - Example 2 PCD

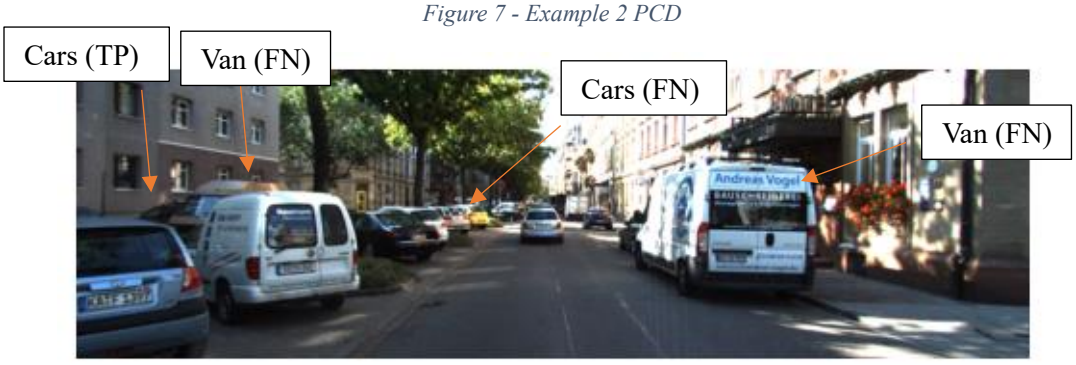


Figure 8 - Example 2 Image

Similar to the previous section, this scene also unfolds in a street with parked cars lining its sides, including some vans. However, unlike the previous scenario, there are no pedestrians present. In this scene, there are several main elements that are worth noting.

- The network encountered difficulties in identifying two vans, and this could be attributed to the training process with unfitted anchor size for vans. The network might not have been adequately trained to handle bounding boxes of certain sizes or ratios that are characteristic of these particular types of vehicles.

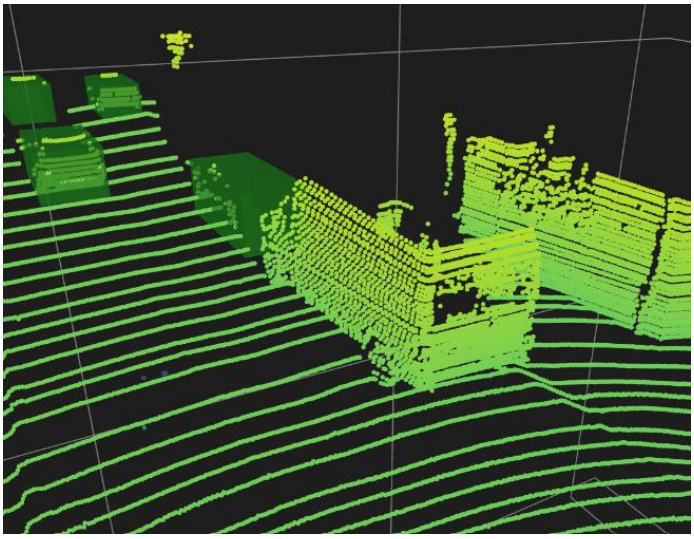
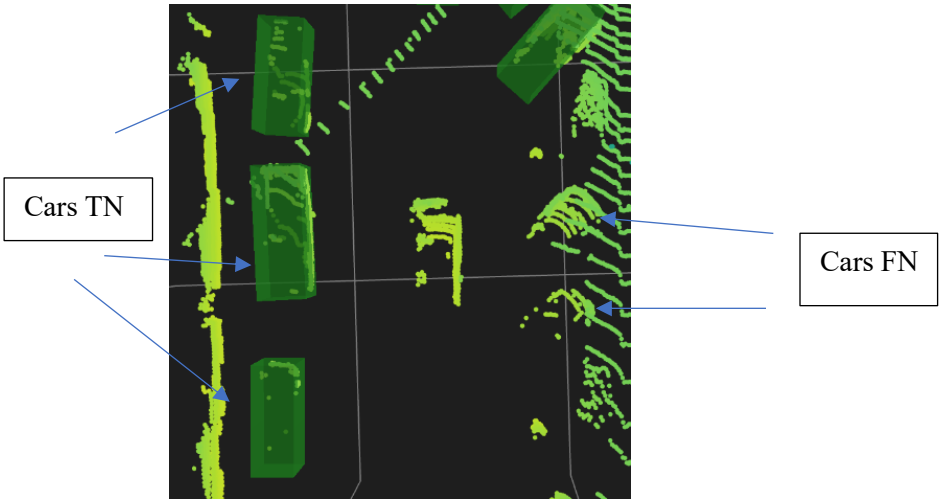


Figure 9 - Example 2 Van False Negative

If we want to include this van inside the car class, we need to configure, and tuned its settings of xyz ranges search space, anchors sizes and threshold appropriately.

- Additionally, there are instances of failures in detecting regular parked vehicles, primarily because they are located in the sensor's blind spot. It's interesting to note that despite receiving a larger number of lidar hits and having more information in relate to the vehicles parked further behind them near the buildings, the network was still able to detect the vehicles behind them. This can be explained by the point cloud distribution, where points representing the parked vehicles on the side are structured in a manner that better defines the bounding box for the specific vehicle class even if they have less points.



c. Example 3

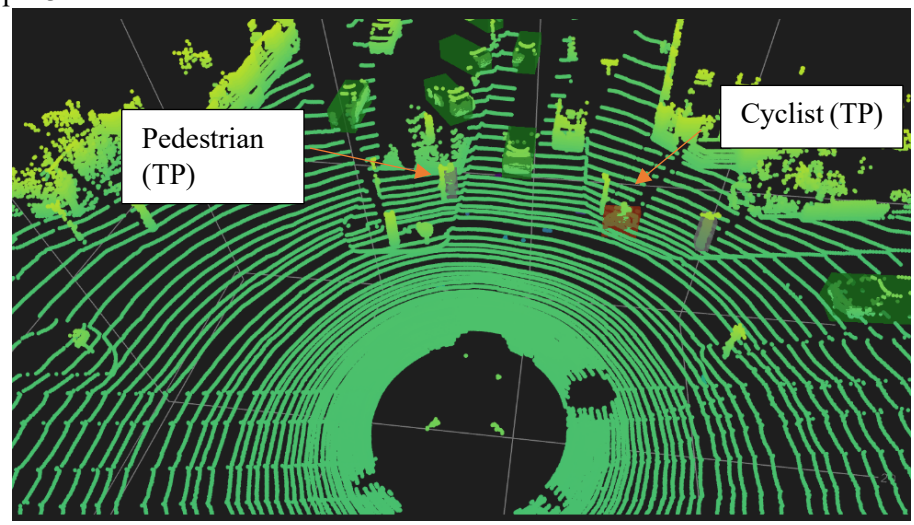


Figure 10 - Example 3 PCD

Figure 11 - Example 3 Image

In this example the vehicle is at an intersection, where there are detections received for a pedestrian, vehicles and a cyclist.

A few points to discuss:

- We can see True Positives of Person on the left, cyclist on the right and a few vehicles detections.

- On the right side, there are instances where vehicles that meet the defined criteria were missed. In these cases, it is evident that these vehicles are positioned behind a van, resulting in insufficient lidar points being captured for their detection. It is important to note that the first vehicle behind Huan is black in color, which absorbs the leader's rays, so in a rough estimate not a single point was taken for it. See figure below.

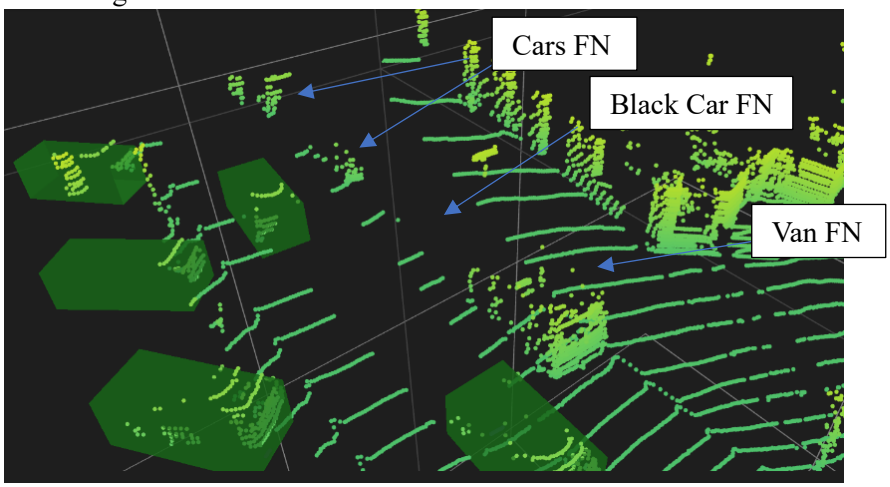


Figure 12 – Example 3 Cars False Negative

- Furthermore, there is a false positive detection of a pedestrian on the right side, whereas in reality, it is likely that a tree or pole is present in that location. See figure below.

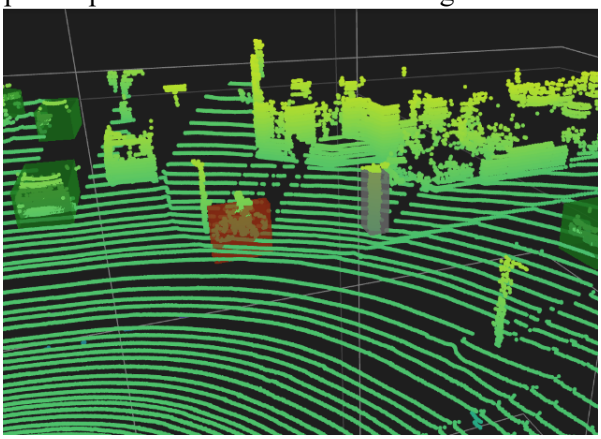


Figure 13 - Example 3 Pedestrian False Positive

There are potential avenues for improvement, such as utilizing a different feature extractor that can effectively differentiate subtle distinctions between objects. For instance, the PointNet++ network could be employed, as it is specifically designed for complex scenes and fine-grained patterns. Additionally, enhancing the network's performance can be achieved by training it with additional augmentations to further improve its capabilities.

d. Bonus

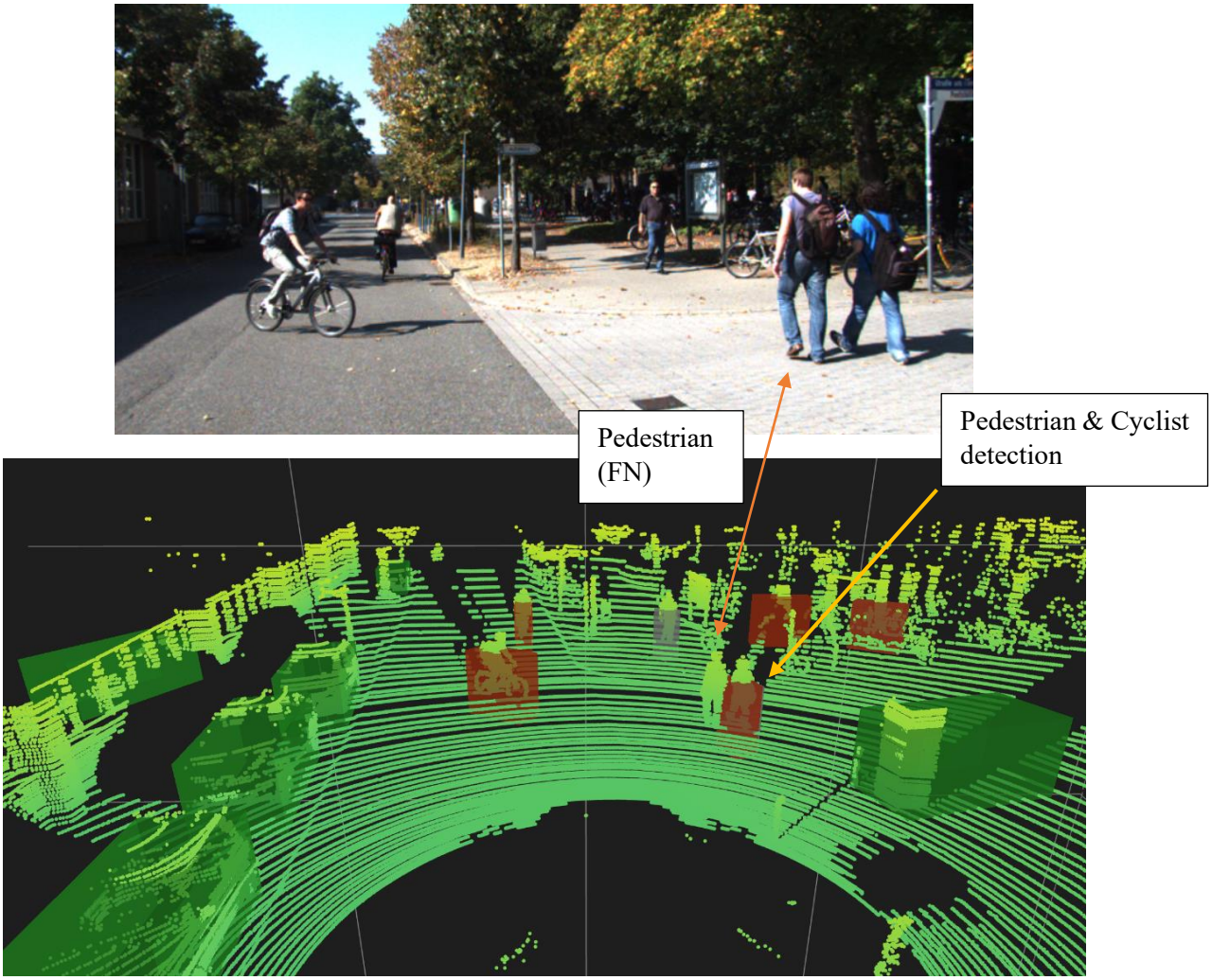


Figure 14 - Bonus scene image and PCD

Based on the picture provided in the bonus section, it is evident that the network has identified both a cyclist and a pedestrian in close proximity to each other. The detection associated the cyclist with a low probability and the pedestrian with a higher probability. This situation is likely influenced by the discretization phase carried out in PointPillars.

During the discretization phase, the space is divided into a coarse resolution grid. Since the two pedestrians are walking very close to each other, it's highly probable that the Pillar created for this grid includes

information from both individuals. Consequently, the features learned by PointNet may resemble those of a cyclist as well, leading to the mixed detection results.

To address this issue, one possible approach is to adjust the *voxel size* parameter in the configuration file while training the entire network. By doing so, it may facilitate better separation of features, potentially improving the network's ability to accurately distinguish between the cyclist and the pedestrian in such closely packed scenarios.

b. 3D detection based on point cloud and RGB image

1. **Multi-stage methods based on the paper: Frustum Convnet,Frustum PointNet and Frustum-PointPillars**

Frustum ConvNet is a multi-stage method described in the paper "Frustum ConvNet" that aims to detect objects in 3D using 2D bounding box information. It involves a two-stage process where the first stage performs 2D object detection on RGB images and extracts 2D bounding box proposals. The second stage utilizes 3D point clouds within the frustum of each 2D bounding box to estimate 3D object poses.

Frustum PointNet, as described in the paper "Frustum PointNet," is another multi-stage method that combines 2D and 3D information for object detection. It leverages PointNet to extract features from both RGB images and 3D point clouds within the frustum of each 2D bounding box. The extracted features are then used to classify and regress 3D object poses.

Frustum-PointPillars is an architecture introduced in the paper "Frustum-PointPillars" that combines the Frustum ConvNet and PointPillars techniques. It extends PointPillars by incorporating the frustum-based approach, allowing it to handle both 2D and 3D data. Frustum-PointPillars first utilizes PointPillars to generate a pseudo-image from the point cloud data, followed by frustum-based extraction of 2D and 3D features from the projected pseudo-image and RGB image. These features are then used for object detection and pose estimation.

2. **Advantages of the Frustum-PointPillars**

Advantages of the Frustum-PointPillars architecture compared to the PointPillars approach include:

1. Enhanced 3D context: Frustum-PointPillars leverages frustum-based techniques to incorporate 2D object detection information and project it onto the 3D point cloud data. This integration provides a richer 3D context, allowing for improved object detection and pose estimation.

2. Improved handling of foreground and background clutter: The frustum-based approach in Frustum-PointPillars helps in isolating foreground and background clutter. By considering the points within the frustum of each 2D bounding box, it focuses on the relevant region of interest and reduces the influence of surrounding clutter, leading to more accurate object detection results.

3. **Foreground and background clutter isolation**

a) The mechanism described in the paper to isolate foreground and background clutter uses Gaussian-based masking points within the frustum, filtering out points that are likely to be clutter or irrelevant to the object of interest. The region near the center of a 2d bounding box is more likely to be occupied by the object, the projected 3d points near the center region are also more likely to belong to the object instead of the background clutter. We define the likelihood of the points belonging to the object as a Gaussian function.

b) The frustum-based approach may struggle to isolate pedestrians on the sidewalks, in groups, and close to other buildings and objects. In the case of pedestrians, foreground and background clutter are not well Isolated.

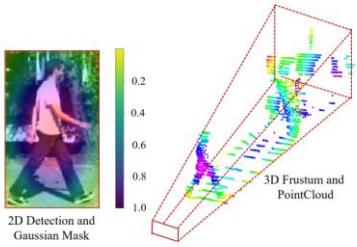


Fig. 4: Left: Illustration of the Gaussian Mask representing likelihood of the pixels belonging to the detected object. Right: Corresponding 3D frustum and masked point cloud.

c)

Figure 15 - Pedestrian foreground and background clutter filtering

In certain cases, as in Figure 15, where a pedestrian is seen extending its arm, the center of the 2D bounding box is slightly shifted and some of the background points are wrongly masked as more likely of belonging to the object

2. Multi Object Tracking

c. Running BoT-SORT on MOT17

1. Theoretical background for multi-object tracking

a. The main mechanism of the SORT (Simple Online and Realtime Tracking) algorithm is based on the principles of the Kalman filter and Hungarian algorithm. It operates in a frame-by-frame manner and aims to track multiple objects in a video stream. The algorithm consists of the following steps:

1. Detection: Objects are detected in each frame using an object detection algorithm, such as Faster R-CNN or YOLO. The detected bounding boxes are considered as potential object detections.

2. Data Association: The Kalman filter is used to predict the location of each tracked object in the current frame based on its previous state. The predicted locations are then compared with the detected bounding boxes to associate each detection with a specific object track. This is done by solving an assignment problem using the Hungarian algorithm, which minimizes the total distance between the predicted and detected locations.

3. State Update: After the data association step, the Kalman filter is used to update the state of each object track based on the associated detection. The state includes the position, velocity, and other relevant attributes of the object.

4. Track Management: Tracks are managed based on their age and the number of consecutive frames in which they have not been associated with any detection. If a track fails to receive any new detections for a certain number of frames, it is considered lost and removed from the tracking process.

b. The DeepSort algorithm enhances the SORT process by incorporating a deep appearance descriptor and a re-identification (RE-ID) technique. The main benefit of DeepSort is the ability to handle occlusions and maintain tracking identities more reliably.

DeepSort introduces a deep neural network, such as a Siamese network or a deep metric learning approach, to extract appearance features from the object detections. These features encode the unique visual characteristics of each object and are used to compute a similarity metric between detections and existing tracks.

The RE-ID aspect of DeepSort refers to the process of matching detections with existing tracks based on their appearance similarity. By comparing the appearance features of new detections with the features of existing tracks, DeepSort can associate occluded or partially visible objects with their correct identities more accurately. This is especially useful in crowded scenarios or when objects temporarily disappear from the field of view.

c. Some good additions to ByteTrack for DeepSort could include:

1. Appearance Modeling: ByteTrack can benefit from incorporating appearance modeling techniques similar to DeepSort. By utilizing deep neural networks to extract appearance features from object detections, ByteTrack can improve its ability to handle occlusions, maintain track identities, and handle re-identification challenges.

2. Track Confirmation: ByteTrack can introduce a track confirmation step to verify the accuracy of the assigned tracks. This can involve considering temporal consistency, motion patterns, and appearance consistency over time. By confirming the tracks, the system can reduce false positives and improve the overall tracking performance.

d. Some good additions to BoT-SORT compared to ByteTrack could include:

1. Efficient Feature Extraction: BoT-SORT can leverage efficient feature extraction methods, such as lightweight convolutional neural networks or feature compression techniques, to extract appearance features from object detections. This can improve the computational efficiency of the algorithm without sacrificing tracking accuracy.

2. Adaptive Track Initialization: BoT-SORT can incorporate an adaptive track initialization mechanism that dynamically adjusts the criteria for initializing new tracks based on the scene context. This can help handle challenging scenarios, such as object appearance changes, occlusions, or abrupt appearance variations.

3. Occlusion Handling: BoT-SORT can include advanced occlusion handling techniques, such as occlusion reasoning or trajectory prediction, to better handle occluded objects. These techniques can help maintain track continuity even when objects are partially or fully occluded by other objects in the scene.

2. Run BoT-SORT tracker

a. RUN

BoT-SORT was run on the five subsets and animation and final performance report were attached.

b. Comparison – CLEARMOT,HOTA,Identity,Detection,IDS

Table 1 - CLEARMOT

CLEAR: BoTSORT-pedestrian	MOTA
MOT17-02	55.584
MOT17-04	89.693
MOT17-09	83.506
MOT17-10	73.343
MOT17-13	81.827
COMBINED	79.331

Based on the information provided in the table, it is evident that the set MOT1704 demonstrated the highest MOTA performances using the BoT-SORT tracker compared to the other sets. In contrast, sets 1702 yielded considerably lower results, indicating a poor tracking quality in terms of MOTA with relate to the other sets.

Table 2 - HOTA

HOTA: BoTSORT-pedestrian	HOTA
MOT17-02	48.738
MOT17-04	78.821
MOT17-09	64.279
MOT17-10	59.315
MOT17-13	70.099
COMBINED	69.368

HOTA stands for Higher Order Tracking Accuracy, which is an extension of the traditional MOTA metric. While MOTA primarily focuses on object-level tracking accuracy, HOTA provides a more comprehensive evaluation by incorporating higher-order information and localization accuracy in addition to the object-level tracking performance. According to the information presented in the HOTA table, it can be observed that the BoT-SORT exhibited superior performance for the 1704 image set, while demonstrating poorer performance for 1702 set.

The reason for HOTA generally being lower than MOTA for the same set can be attributed to the additional criteria mentioned above, and stricter evaluation measures employed by HOTA compared to MOTA.

Table 3 - Identity

Identity: BoTSORT-pedestrian	IDF1
MOT17-02	57.353
MOT17-04	90.57
MOT17-09	76.018
MOT17-10	80.384
MOT17-13	89.368
COMBINED	81.787

In terms of Identity metric, a high IDF1 score suggests that there are fewer false positives (incorrectly identified objects) and false negatives (missed detections) in the tracking results, resulting in a more precise and reliable tracking performance. The higher scores of IDF1 received again for 1704 set, and 1713, in contrast to 1702 set.

Table 4 - IDS and Detections

Count: BoTSORT-pedestrian	Dets	GT_Dets	IDs	GT_IDs
MOT17-02	7223	9913	100	53
MOT17-04	23558	24226	99	69
MOT17-09	2462	2892	23	22
MOT17-10	4790	5946	48	36
MOT17-13	2760	3175	48	44
COMBINED	40793	46152	318	224

The table clearly illustrates that set 1704 (GT) had the highest number of identifications and IDs. This observation aligns with the video itself, which shows a significant presence of pedestrians. However, in comparison to the ground truth (GT), the tracker classified a much larger number of IDs, suggesting challenges in accurate identification.

In general, it can be concluded that the tracker performed better on set 1704. This particular set benefits from a wide camera coverage, a high shooting angle that aids in distinguishing individuals, no abnormal hiding of people, and the camera is static, which simplifies the tracking process.

In contrast, in video 1702, which yielded the lowest results, several factors contribute to the challenges faced by the tracker. Despite the presence of a considerable number of people in the video, the camera is positioned at the start of a street at eye level, with a horizontal angle. This perspective leads to the presence of various obstructions, such as people and objects, which hinder the visibility of individuals who are farther away.

Furthermore, the video reveals additional complexities, including a significant gathering of people near the entrance of a store with shaded areas, individuals sitting on a bench under a tree with shading as well, and the presence of cyclists navigating through the crowded environment. These conditions further complicate the tracking process and contribute to the lower performance observed in video 1702.

3. Results Analysis

a. Camera motion compensation

In 1713 and 1710 subsets where the camera is in motion rather than stationary, tracking poses additional challenges. One of the challenges in tracking is the dynamic camera situation, causing shifts in bounding box locations. This can lead to increased ID switches or false negatives. The proposed solution is to use image registration techniques to compensate for camera motion. By estimating the rigid motion of the camera onto the image plane through image registration between adjacent frames, background motion can be accurately estimated. This approach helps minimize the impact of bounding box shifts and reduces ID switches or false negatives.

b. ID switch

A good crossing objects can be obtained in 1709 set where ID12 and ID14 are crossing each other and the women keeps the same ID.

Figure 16 - Preventing ID switch set 1709

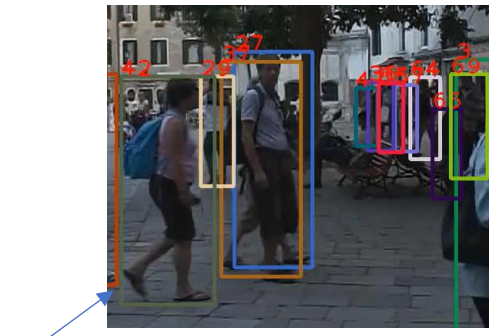


Figure 17 - Preventing ID switch set 1702

Another example can be obtained on 1702 set, where the tracker keeps ID42 while she crossing a few objects detected.

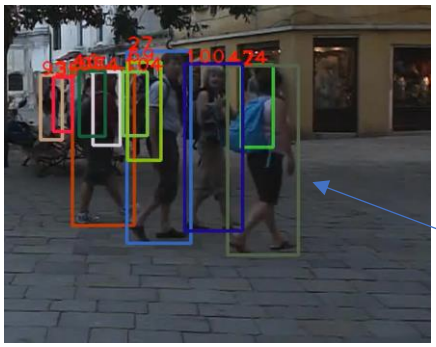


Figure 18 - Keeping ID set 1702

The BoT-SORT tracker employs three core functions to handle crossing objects and ID switches:

Below are the images that serve as an illustration of a poor example of ID tracking.

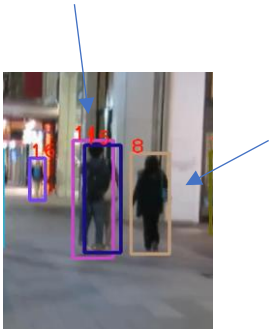


Figure 19 - ID switch set 1710

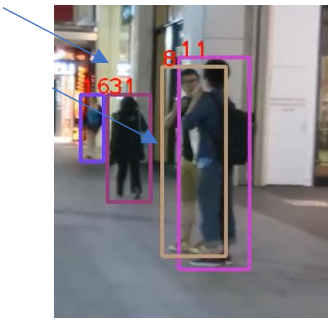


Figure 20 - ID switch set 1710 (after crossing)

It can be seen that the women that got ID8 switch ID to 31 after crossing the two standing persons, while one of them is getting her ID 8. In the initial part of the video, the man who got the ID of her appears to be poorly visible on the camera due to another person obstructing him almost entirely. As a result, there is limited information available about him. Moreover, the camera movement adds complexity to the image, potentially diverting attention. It is plausible that during the cross between the individuals, the tracker might mistakenly identify that person as teh woman with ID8, who may have paused or stopped momentarily.

c. Tracking on occlusion/truncated objects

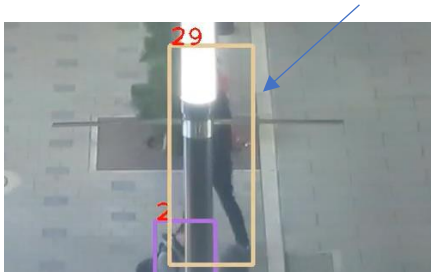


Figure 21 - Tracking on occlusion set 1704

By examining the image provided, you can observe a notable instance of effective tracking even when a person is obscured by a light pole. The individual is continuously tracked from the start of the video, capturing and retaining sufficient information and distinguishing characteristics that enable tracking to persist even when the person is truncated.

Figure 22 - Tracking on occlusion set 1713

Figure 21 above is another example for good tracking on occlusion case. Despite the camera's movement and the person's passage behind a stationary vehicle, the tracking remains intact in this scenario. This achievement could be attributed to a successful CMC (Camera Motion Compensation) which properly correct the KF in the tracking algorithm.

Figure 23 - Tracking on occlusion set 1704

MOT1704 video also includes a negative instance where the tracking fails for a person standing behind a light pole (see the figure above). The tracking loss could potentially be attributed to several factors. Firstly, the person is wearing a white garment and positioned behind bright lighting that blends with their clothing, making it challenging to distinguish the person from the surrounding illumination. Furthermore, the person is wearing a mask, and it is uncertain whether the model was trained using data from individuals wearing masks, which can further impede accurate tracking.

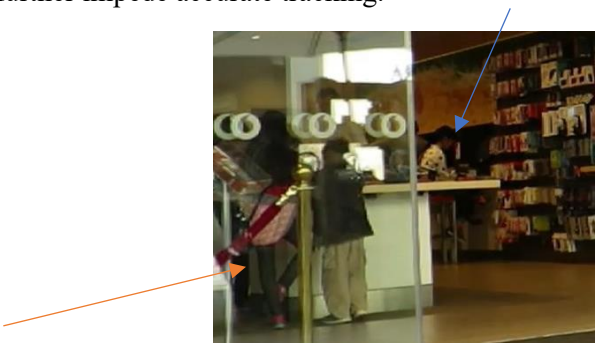


Figure 24 - Tracking on occlusion set 1709

In the provided figure above, we can observe another example where the algorithm fails to successfully detect and track a person (blue arrow). The person is seated, with a portion of their body obscured by a table. This failure could be attributed to multiple factors. Firstly, the inadequate lighting conditions might have played a role. Additionally, the sitting position of the person presents a challenge for identification since it is less common compared to standing or walking. It is worth noting that this difficulty in detecting individuals is evident in other videos as well.

Theoretical solutions can be attributed to training with relevant augmentations or using person segmentation which can enable proper occlusion handling by accurately delineating the boundaries of individuals within an image or video frame. This precise localization provides more reliable information about the position and extent of each person, even when they are partially occluded.

d. Detection performance

In Figure 24 you can additionally notice (orange arrow) a pair of people that the algorithm was unable to detect, the people are behind glass that affects the detection of the people in a bad way because of the reflection and the blurring of the image.

In the image below, we can observe another example of a False Negative, where the algorithm struggles to identify the children riding a bicycle. The dynamic nature of riding a bicycle makes it challenging for the algorithm to accurately recognize the individuals. Furthermore, the presence of the bicycle obstructs certain body parts, and the children's position deviates from typical standing or walking poses, further complicating the identification process.

Figure 25 - Set 1702 False Negative

Within the video of set 1702, there is an additional instance of a False Positive, where the algorithm incorrectly identifies a person and assigns the ID29, despite there being no actual person present. This misidentification occurred for one or two frames only when two individuals crossed paths, and it is crucial to take note of the mannequin displayed in the shop window located behind them. These factors, namely the crossing of two people and the presence of the mannequin, likely contributed to the erroneous identification, emphasizing their significance as potential causes of the error.

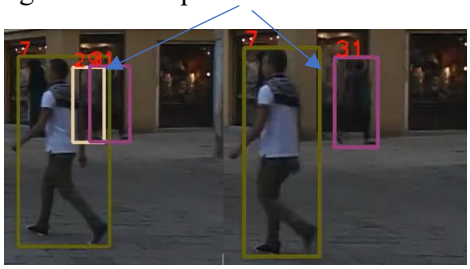


Figure 26 - Set 1702 False Positive

3. Summary

In summary, this project delved into two significant domains within the subject of visual perception in robotics and autonomous systems. We initially focused on implementing PointPillars 3D object detection in the first section, followed by BoT-SORT multi-object tracking in the second section. Throughout the project, we experimented with the mentioned state-of-the-art algorithms, carefully analyzing the obtained results. The outcomes we achieved were both captivating and enlightening, offering valuable insights into the capabilities of advanced algorithms in these areas.