

# Overcoming Domain Shift in Lidar-based 3D Object Detection in Urban Contexts

Jen Kun Darren Tsai

A thesis submitted in fulfilment  
of the requirements of the degree of  
Doctor of Philosophy



THE UNIVERSITY OF  
**SYDNEY**

This research reported in this thesis was supported by the award of  
a Research Training Program scholarship to the PhD Candidate.

Australian Centre for Robotics  
School of Aerospace, Mechanical and Mechatronic Engineering  
The University of Sydney

November 2023

# Declaration

This is to certify that to the best of my knowledge, the content of this thesis is my own work. This thesis has not been submitted for any degree or other purposes.

I certify that the intellectual content of this thesis is the product of my own work and that all the assistance received in preparing this thesis and sources have been acknowledged.

**Jen Kun Darren Tsai**

20 November 2023



# Authorship attribution

This thesis contains material published in:

- D. Tsai, S. Worrall, M. Shan, A. Lohr, and E. Nebot. "Optimising the selection of samples for robust lidar camera calibration," In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pp. 2631–2638. IEEE, 2021.
- D. Tsai, J. S. Berrio, M. Shan, S. Worrall, and E. Nebot. "See eye to eye: A lidar-agnostic 3d detection framework for unsupervised multi-target domain adaptation," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7904–7911, 2022.
- D. Tsai, J. S. Berrio, M. Shan, E. Nebot, and S. Worrall. "Viewer-Centred Surface Completion for Unsupervised Domain Adaptation in 3D Object Detection," In *2023 IEEE International Conference on Robotics and Automation (ICRA)*. pp. 9346-9353. IEEE, 2023.
- D. Tsai, J. S. Berrio, M. Shan, E. Nebot, and S. Worrall. "MS3D: Leveraging Multiple Detectors for Unsupervised Domain Adaptation in 3D Object Detection" In *2023 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2023. (accepted)
- D. Tsai, J. S. Berrio, M. Shan, E. Nebot, and S. Worrall. "MS3D++: Ensemble of Experts for Multi-Source Unsupervised Domain Adaptation in 3D Object Detection" *Submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*

I am the corresponding author of all the published material mentioned above.

Jen Kun Darren Tsai

30 September 2023

As supervisor for the candidature upon which this thesis is based, I can confirm that the authorship attribution statements above are correct.

Stewart Worrall

30 September 2023

# Abstract

3D Object Detection is a vital task in 3D scene understanding for many real-world applications such as autonomous driving, robotics and intelligent transport systems. It serves as a fundamental building block for critical downstream tasks such as object tracking and path prediction. However, the deployment of 3D detectors on different lidar types and environments is not a straightforward process. Despite remarkable accuracy on benchmarks, 3D detectors trained on one lidar dataset often suffer a significant drop in performance when tested on a separate lidar or operational environment. This performance degradation can be attributed to a domain shift between the training and testing datasets, stemming from variations in scan pattern across lidars, weather conditions or scenarios. Consequently, 3D detectors tested in unfamiliar testing environments might fail to detect densely observed objects, have misaligned confidence scores, and show an increase in high-confidence false positives. This compromised operational capability renders the detector unfeasible for use. For instance, applying a detector trained on a 32-beam lidar to a 64-beam lidar dataset results in many false positives and missed detections even within 30m range. This is a substantial barrier that hinders the widespread integration of the latest 3D detector research on different types of lidar technologies.

This thesis proposes two approaches that effectively leverage existing large-scale labelled datasets for addressing the domain shift: domain-invariant representation and auto-labelling. Firstly, we design a framework that enables a 3D detector to be lidar-agnostic by training and testing on a domain-invariant representation. Our proposed approach uses the camera as a complementary sensor with image instance segmentation to generate region of interest proposals. Within each region of interest, the object points are then transformed to a unified representation to enable 3D detectors to learn scan pattern invariant representations of objects. To enable optimal fusion of the two modalities, we propose a robust calibration pipeline and demonstrate its effectiveness in calibrating a camera with different lidar types. Furthermore, we propose a surface completion network that enables a consistent point representation of objects across different lidar types. Through our experiments, we demonstrate that our approach can obtain improved 3D detection performance for different lidar types without requiring re-training.

The second approach we propose is an auto-labelling framework. This involves utilizing automatically generated labels to re-train a 3D detector on an unlabelled dataset, commonly referred to as self-training. Our framework generates robust labels for a range of lidar types regardless of their point cloud density. To improve detector generalization for various lidar types, we utilize an ensemble of pre-trained detectors trained on different labelled source datasets. We fuse all predictions of our ensemble using our developed box fusion algorithm, which consistently outperforms the best individual detector. Subsequently, we use point cloud sequences and object characteristics to temporally refine our generated bounding box labels. Experiments demonstrate that 3D detectors trained with our generated labels achieve state-of-the-art performance, comparable to training with human-annotated labels in Bird’s Eye View (BEV) evaluation. Given that we only use labels for addressing the domain shift, we highlight that this approach is suitable for any 3D detector or data augmentation technique without impacting runtime performance.

# Acknowledgements

I am grateful to my supervisors Prof. Eduardo Nebot, Dr Stewart Worrall, Dr Mao Shan and Dr Stephany Berrio for their guidance and support throughout my PhD. Beyond the supervising of my PhD progress, they have also been influential in shaping my professional and personal growth. I would also like to thank all the ITS group members who have made my time in this PhD very enjoyable.

Last but not least, I am very grateful for my wife, Christine Tsai, who supported me through this PhD which has been full of ups and downs. I would not have been able to complete this PhD without her love and support.

# Contents

|                                 |            |
|---------------------------------|------------|
| <b>Declaration</b>              | <b>i</b>   |
| <b>Authorship attribution</b>   | <b>ii</b>  |
| <b>Abstract</b>                 | <b>iv</b>  |
| <b>Acknowledgements</b>         | <b>vi</b>  |
| <b>Contents</b>                 | <b>vii</b> |
| <b>List of Figures</b>          | <b>x</b>   |
| <b>List of Tables</b>           | <b>xvi</b> |
| <b>Nomenclature</b>             | <b>xix</b> |
| <b>Glossary</b>                 | <b>xx</b>  |
| <b>1 Introduction</b>           | <b>1</b>   |
| 1.1 Thesis Objectives . . . . . | 6          |
| 1.2 Contributions . . . . .     | 8          |
| 1.3 Publications . . . . .      | 10         |
| 1.4 Thesis Structure . . . . .  | 11         |

|          |                                                      |           |
|----------|------------------------------------------------------|-----------|
| <b>2</b> | <b>Literature Review</b>                             | <b>12</b> |
| 2.1      | 3D Object Detection . . . . .                        | 12        |
| 2.1.1    | Single-Frame 3D Detection . . . . .                  | 12        |
| 2.1.2    | Multi-Frame 3D Detection . . . . .                   | 14        |
| 2.1.3    | Multi-Modal 3D Detection . . . . .                   | 15        |
| 2.1.4    | Detection Head . . . . .                             | 16        |
| 2.2      | Unsupervised Domain Adaptation . . . . .             | 17        |
| 2.2.1    | Overview . . . . .                                   | 17        |
| 2.2.2    | Single-target, single-source . . . . .               | 18        |
| 2.2.3    | Multi-target . . . . .                               | 18        |
| 2.2.4    | Multi-source . . . . .                               | 19        |
| 2.2.5    | Source-free . . . . .                                | 19        |
| 2.3      | Target-based Calibration . . . . .                   | 20        |
| 2.4      | Real-world Point Cloud Completion . . . . .          | 21        |
| <b>3</b> | <b>Domain-Invariant Representation</b>               | <b>23</b> |
| 3.1      | Introduction . . . . .                               | 23        |
| 3.2      | Camera-Lidar Calibration . . . . .                   | 25        |
| 3.2.1    | Overview . . . . .                                   | 25        |
| 3.2.2    | Method . . . . .                                     | 29        |
| 3.2.3    | Experiments . . . . .                                | 33        |
| 3.3      | Scan Pattern Normalization Framework (SEE) . . . . . | 40        |
| 3.3.1    | Overview . . . . .                                   | 40        |
| 3.3.2    | Method . . . . .                                     | 41        |
| 3.3.3    | Experiments . . . . .                                | 46        |
| 3.4      | Viewer-Centred Completion Network . . . . .          | 55        |
| 3.4.1    | Overview . . . . .                                   | 55        |
| 3.4.2    | Method . . . . .                                     | 57        |
| 3.4.3    | Experiments . . . . .                                | 62        |
| 3.5      | Conclusion . . . . .                                 | 69        |

---

|          |                                                        |            |
|----------|--------------------------------------------------------|------------|
| <b>4</b> | <b>Auto-labelling</b>                                  | <b>72</b>  |
| 4.1      | Introduction . . . . .                                 | 72         |
| 4.2      | Multi-source Self-training Framework (MS3D) . . . . .  | 76         |
| 4.2.1    | Overview . . . . .                                     | 76         |
| 4.2.2    | Method . . . . .                                       | 78         |
| 4.2.3    | Experiments . . . . .                                  | 85         |
| 4.3      | Quantifying Cross-domain 3D Object Detection . . . . . | 94         |
| 4.3.1    | Overview . . . . .                                     | 94         |
| 4.3.2    | Experiment Setup . . . . .                             | 95         |
| 4.3.3    | Results and Discussion . . . . .                       | 96         |
| 4.3.4    | Conclusion . . . . .                                   | 105        |
| 4.4      | MS3D++ . . . . .                                       | 106        |
| 4.4.1    | Overview . . . . .                                     | 106        |
| 4.4.2    | Method . . . . .                                       | 108        |
| 4.4.3    | Experiments . . . . .                                  | 115        |
| 4.5      | Conclusion . . . . .                                   | 126        |
| <b>5</b> | <b>Conclusion</b>                                      | <b>129</b> |
| 5.1      | Summary . . . . .                                      | 129        |
| 5.2      | Contributions . . . . .                                | 130        |
| 5.3      | Future Research . . . . .                              | 132        |
|          | <b>List of References</b>                              | <b>134</b> |



# List of Figures

|     |                                                                                                                                                                                                                                                                                                                                                                                                                                                 |    |
|-----|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 1.1 | Scan patterns of different lidar datasets. . . . .                                                                                                                                                                                                                                                                                                                                                                                              | 2  |
| 1.2 | Illustration of in-domain and cross-domain settings for model deployment. . . . .                                                                                                                                                                                                                                                                                                                                                               | 4  |
| 1.3 | Challenges encountered by testing 3D detectors in a cross-domain setting. (1) Missed detections despite dense observations of objects due to the scan pattern gap. (2) Weather artefacts appear different across lidar types, causing missed detections. (3) Box height estimation is challenging for sparse lidar. (4) Lack of ego-motion compensation causes poor evaluation and transfer of pre-trained model to other test domains. . . . . | 5  |
| 1.4 | Scan pattern normalization of objects across various lidar types. . .                                                                                                                                                                                                                                                                                                                                                                           | 6  |
| 1.5 | Auto-labelling for re-training detectors on a new lidar dataset. . . .                                                                                                                                                                                                                                                                                                                                                                          | 7  |
| 3.1 | Sampling discrepancy of cars at 10m distance captured by different lidars. Top left: Baraja Spectrum-Scan, an adjustable scan pattern lidar; Top right: Waymo Open Dataset [Sun et al., 2020] (64-beam); Bottom left: KITTI Dataset [Geiger et al., 2012] (64-beam); Bottom right: nuScenes Dataset [Caesar et al., 2020] (32-beam). . . . .                                                                                                    | 24 |

|     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |    |
|-----|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 3.2 | Many existing methods often show the effectiveness of their approach by presenting a well-fitted target with low reprojection error (a), but not the whole scene (b). Both (a) and (b) have the same calibration parameters. The centre of the red tilted crosshair is the lidar's reprojected board centre, and green is the camera's board centre. The reprojection error in (a) and (b) is 1.8 pixels, which is good, however the calibration parameters don't fit the pillar well at the background because the algorithm only focuses on optimising the chessboard fit. (c) has better overall alignment of scene features than (b) despite having 2.7 pixel error vs 1.7 respectively, showing that same target pose can be fitted with multiple different calibration parameters. (d) and (e) compare the calibration result of 3 and 9 poses respectively, showing that an equally good calibration can be achieved on both. . . . . | 26 |
| 3.3 | <b>Our proposed calibration pipeline.</b> The entire pipeline (excluding the manual data collection phase) takes around 90s to assess the data and estimate calibration parameters. . . . .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  | 27 |
| 3.4 | Calibration result of Baraja Spectrum-Scan (top) and Velodyne VLP-16 (bottom) with our method. Projection is shown for the whole scene, to illustrate that our estimated calibration parameters are fitted to the scene. Points are coloured based on distance to the lidar, red for points closer than 3m and blue for points around 20m away. The chessboard for the Baraja Spectrum-Scan (top) has ground points projected on the board due to a large difference in perspective of the Baraja Spectrum-Scan and camera. . . . .                                                                                                                                                                                                                                                                                                                                                                                                          | 28 |
| 3.5 | Illustration of linear dependency. When the normal (blue arrows) of the target planes have enough variation, it outlines a distinct parallelepiped (stable) with a large determinant, which is the volume of this parallelepiped. However, if the normals have similar or exactly the same orientations, the parallelepiped is hard to define or non-existent, and the determinant is close to zero. . . . .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | 30 |
| 3.6 | Lidar range error may cause it to pick up a skewed board (left), or incorrectly fit a plane to the chessboard (right). Both of these affect the accuracy of feature extraction from the target. . . . .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | 32 |
| 3.7 | Sensor setup. Top-right shows the coordinate frames of the VLP-16 and camera, and the bottom-right shows the frames of the two Baraja Spectrum-Scan sensorheads and camera. The Baraja Spectrum-Scan sensorheads had to be calibrated with the camera separately. . . . .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    | 34 |
| 3.8 | Distance offset for the Baraja Spectrum-Scan in metres squared. Before the offset was added, the chessboard's area was consistently underestimated on every chessboard sample. . . . .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       | 35 |

|      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |    |
|------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 3.9  | Plots for VLP-16 data. (a) Demonstrates the effect of condition number on the stability of calibration results. (b) Shows that a decrease in board dimension error between the lidar-measured board dimensions and the physically measured board dimensions leads to a lower reprojection error mean and standard deviation. . . . .                                                                                                                                                                              | 36 |
| 3.10 | (a) Demonstrates the effect of the VOQ score on the stability of calibration results. (b) Mean reprojection errors for VLP-16 across 46 poses showing that a lower VOQ score leads to lower mean reprojection errors. After VOQ=200, the reprojection errors grow exponentially past 2000 and therefore have been excluded. . . . .                                                                                                                                                                               | 37 |
| 3.11 | Projection of point cloud onto image using estimated calibration parameters at the VOQ score indicated by the dotted lines in the top row. The left is the VLP-camera calibration, and the right is the Baraja-camera calibration. As the average VOQ score of the 50 sets increases, so too does the scene misalignment (observable in the background white pillar), and reprojection error mean and standard deviation. This correlation is observed in both the VLP-16 and Baraja Spectrum-Scan lidar. . . . . | 38 |
| 3.12 | <b>Overview of the SEE framework</b> , our lidar-agnostic unsupervised multi-target DA framework for 3D object detection. Our framework allows a 3D detector trained on one source domain to be used on multiple target domains without additional training. SEE is comprised of 3 phases: object isolation, Surface Completion (SC), and point sampling.                                                                                                                                                         | 40 |
| 3.13 | Background points due to viewpoint misalignment in KITTI dataset.                                                                                                                                                                                                                                                                                                                                                                                                                                                 | 42 |
| 3.14 | (a) Lidar ring separation with increasing distance. At 40m, the distance between the vertical rings of the nuScenes lidar is nearly 1m; (b) Velodyne HDL-64E (KITTI) ring separation at 10m, 20m, 30m and 40m. A KITTI car at 40m has a maximum of 5-6 rings on the car. . .                                                                                                                                                                                                                                      | 43 |
| 3.15 | The Baraja Spectrum-Scan dataset scan pattern is configured to foveate points at the horizon. The foveation pattern is visible on the SUV where the bottom half has a ring-like appearance whilst the upper half we describe as a uniform, interleaved scan pattern. . . . .                                                                                                                                                                                                                                      | 47 |
| 3.16 | Qualitative results for our SEE framework. . . . .                                                                                                                                                                                                                                                                                                                                                                                                                                                                | 51 |
| 3.17 | Minimum number of points using SEE-Ideal, evaluated on the Waymo → KITTI scenario with SECOND-IoU. Reported AP is at 0.7 IoU for (a) 3D (b) BEV. . . . .                                                                                                                                                                                                                                                                                                                                                          | 53 |
| 3.18 | VCN transforms objects from each lidar, into a unified surface representation. . . . .                                                                                                                                                                                                                                                                                                                                                                                                                            | 56 |

|      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |    |
|------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 3.19 | <b>SEE-VCN</b> integrates the SEE framework with our Viewer-centred Completion Network (VCN) to transform objects of interest to a unified point cloud representation. For the source domain, we use ground truth labels to transform objects into an object-centred (canonical) coordinates before completion (VCN-CN). For the target domain, we estimate the object pose to transform the viewer-centred object to object-centred coordinates before completing the object (VCN-VC).                                                                                                                                                    | 57 |
| 3.20 | Architecture of our Viewer-centered surface Completion Network (VCN).                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | 57 |
| 3.21 | Simulated training dataset.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                | 61 |
| 3.22 | Illustration of VCN's completion and post-processing for a single object                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   | 64 |
| 3.23 | SEE-VCN qualitative results                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                | 67 |
| 4.1  | Illustration of the self-training approach.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                | 73 |
| 4.2  | Domain generalization for unseen target domains can be improved by ensembling multiple pre-trained detectors from various source domains. Figure shows detector predictions on a point cloud from the Waymo dataset [Sun et al., 2020] in rainy weather. Top: PV-RCNN++ [Shi et al., 2023] trained on Lyft[Kesten et al., 2019], Middle: Ensemble of PV-RCNN++ and VoxelRCNN[Deng et al., 2020] trained on both Lyft and nuScenes[Caesar et al., 2020], Bottom: Fused detections with our Kernel-density estimation Box Fusion (KBF).                                                                                                      | 75 |
| 4.3  | <b>The MS3D self-training framework.</b> Given a set of $N$ pre-trained 3D detectors from multiple source domains $\mathbf{D}_{S-i}$ where $i = 1, 2, \dots, N$ , we generate detections for both 1-frame and multi-frame accumulated point clouds, and combine them with our KBF. Thereafter, we feed the sequence frames into a 3D tracker for generating trajectories of both 1-frame and multi-frame boxes. Finally, we refine static object boxes to get our final pseudo-labels for fine-tuning any pre-trained detector $\mathbf{D}_{S-i}$ . Visualized in this figure is our MS3D framework for Waymo/Lyft $\rightarrow$ nuScenes. | 78 |
| 4.4  | (a) Fused TTA predictions of 4 source detectors (multi-coloured boxes) with KDE Box Fusion (KBF); (b) Example of how we select the best rotation of multiple boxes with KDE.                                                                                                                                                                                                                                                                                                                                                                                                                                                               | 80 |
| 4.5  | 1-frame KBF boxes compared to 16-frame KBF boxes on the nuScenes dataset with detectors trained on Waymo/Lyft as the source domain. Visualized is a 1-frame point cloud.                                                                                                                                                                                                                                                                                                                                                                                                                                                                   | 82 |

|      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |     |
|------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| 4.6  | Accumulated frames of a parked car and its assigned ground-truth boxes from all frames of a Waymo sequence (197 frames/boxes total). <b>Light green</b> , <b>dark green</b> , <b>pink</b> , <b>red</b> and <b>orange</b> represent accumulated points in intervals of 40 frames in order where light green and orange represents frames 0-40 and 160-197 respectively. The bottom left annotation illustrates the registration misalignment, especially between light green (1) and red (4) outlines on the car's rear, potentially caused by ego-pose drift. Best viewed in colour. . . . .                                                                                                                                                                                                                                            | 84  |
| 4.7  | Evaluation of MS3D's pseudo-labels on different target domains. 1F MS3D refers to using only KBF on 1-frame detections from the 4 source domains. 1+16F MS3D refers to our final pseudo-labels after refining static vehicles. . . . .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  | 90  |
| 4.8  | Qualitative results for MS3D. Red circles highlight the areas where MS3D detected true positive vehicles that were missed by the "Source-only" models. . . . .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          | 92  |
| 4.9  | 3D Detectors tested on cross-domain data encounter various issues. From left to right: (1) Scan pattern discrepancy causes densely observed vehicles and pedestrians to be missed, (2) Weather artefacts such as reflections on the ground due to rain, appear different across various scan patterns, causing missed detections or false positives, (3) Sparse lidar is challenging for box height estimation, (4) Poor (or lack of) motion compensation for the ego-vehicle causes artefacts, inaccurate labels, and therefore, prediction "errors". In the image, the upper predicted box may not meet the IoU=0.7 threshold for a true positive detection. Predictions for image (1-4) were made with PV-RCNN++ (centerhead) and trained on nuScenes (1), Lyft (2) and Waymo (3-4). We elaborate on these in Section 4.3.3. . . . . | 103 |
| 4.10 | Our <b>multi-source self-training framework, MS3D++</b> . Given a set of $M$ pre-trained, multi-frame 3D detectors from multiple source domains $\mathbf{D}_{s,m}$ where $m = 1, 2, \dots, M$ , we generate predictions for a varying number of accumulated point cloud frames which are fused with KBF and tracked. Our temporal refinement uses object characteristics to ascertain class labels and refine bounding box localization. The set of pseudo-labels is iteratively improved with each update, utilizing the re-trained detectors from the previous round in its ensemble. . . . .                                                                                                                                                                                                                                         | 107 |
| 4.11 | Ensembling with Varied Multi-frame Inference (VMFI) obtains an extended range of detections compared to using 16-frames for inference. This could be attributed to the exacerbated scan pattern domain gap when feeding 16-frames to a pre-trained multi-frame detector of a different source domain. . . . .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           | 110 |

|      |                                                                                                                                                                                                                                                                                                                                                                                              |     |
|------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| 4.12 | Iterative label refinement of MS3D++: In the first round of self-training, we employ a range of detectors from various source domains. The pseudo-labels are used to re-train the 3D detectors with multi-frame accumulation. We use the re-trained detectors to create a new ensemble for the next round. . . . .                                                                           | 114 |
| 4.13 | Visualization of MS3D++ pseudo-labels on Waymo and nuScenes point clouds. We show that we can obtain well localized boxes for sparse observations on both datasets. For a 64-beam lidar (Waymo), we show that MS3D++ can generate accurate pseudo-labels for up to 80m range.                                                                                                                | 117 |
| 4.14 | Missing labels on the Lyft dataset affects accurate evaluation of domain adaptation performance. Additionally, this contributes to the poor performance when testing Lyft-trained detectors on Waymo or nuScenes due to the difference in labelling guidelines. . . . .                                                                                                                      | 120 |
| 4.15 | Iterative label refinement on Waymo as the target domain with VoxelRCNN-Anchorhead (VX-A) and VoxelRCNN-Centerhead (VX-C). We show the improvement in detection rate of VX-A and VX-C over multiple rounds of self-training. All detectors are trained on 4-frame point cloud accumulation. The 0th round refers to the detectors without any domain adaptation (i.e., Source-Only). . . . . | 124 |

# List of Tables

|     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |    |
|-----|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 3.1 | Results for the different DA scenarios. We report the AP for the car class on KITTI dataset in bird’s-eye-view ( $AP_{BEV}$ ) and 3D ( $AP_{3D}$ ) at the 0.7 IoU threshold on the moderate case. “Improvement” denotes the AP increase from source-only to the various approaches in the format $AP_{BEV} / AP_{3D}$ . The best results are indicated in bold (with the exception of SEE-Ideal as it uses ground truth boxes). All models are evaluated on ground truth boxes with 50 points or more. Publicly available ST3D models are evaluated with this same criterion. . . . . | 50 |
| 3.2 | Results for the Baraja Spectrum-Scan dataset. “Improvement” denotes AP increase from Source-only to SEE. Reported AP is at the 0.7 IoU threshold. We evaluate using the same trained models in Table 3.1. .                                                                                                                                                                                                                                                                                                                                                                           | 52 |
| 3.3 | Point sampling strategies: VRES, Surface-Area (SA), and Virtual Lidar (VL). The reported AP shows SEE-Ideal/SEE at 0.7 IoU using SECOND-IoU on the Waymo $\rightarrow$ KITTI task. Methods are evaluated on ground truth boxes with 50 or more points. . . . .                                                                                                                                                                                                                                                                                                                        | 54 |
| 3.4 | Comparison of BPA with Alpha Shapes and PCN [Yuan et al., 2018]. Table shows SEE-Ideal/SEE on the nuScenes $\rightarrow$ KITTI task using SECOND-IoU for $AP_{3D}$ at 0.7 IoU. . . . .                                                                                                                                                                                                                                                                                                                                                                                                | 54 |
| 3.5 | Overview of the sensor setup across datasets used in this section. . . .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              | 62 |
| 3.6 | Evaluation of viewer-centred surface completion of cars with at least 30 points. c: centre estimation, r: rotation estimation, l: lidar simulation used. R is median rotation error ( $^{\circ}$ ) and t is mean translation error (m). BEV/3D refers to the IoU of the completed car’s box compared with the ground truth box size. CD is evaluated with the L2 norm and scaled by 1000 (following [Yu et al., 2021]). . . . .                                                                                                                                                       | 63 |
| 3.7 | Results for the different DA scenarios, reported with an IoU threshold of 0.7. “Improvement” denotes the AP increase from source-only to the various approaches in the format $AP_{BEV} / AP_{3D}$ . “Ideal iso” refers to the scenario where car points are ideally isolated in testing. All models are evaluated on ground truth boxes with 50 points or more. . . . .                                                                                                                                                                                                              | 65 |

|      |                                                                                                                                                                                                                                                                                                                                                            |     |
|------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| 3.8  | We demonstrate that SEE-VCN on the Baraja Spectrum-Scan <sup>TM</sup> dataset with an IoU threshold of 0.7. . . . .                                                                                                                                                                                                                                        | 66  |
| 3.9  | IoU of VCN for cars with varying number of points on lidar datasets. . . . .                                                                                                                                                                                                                                                                               | 67  |
| 3.10 | IoU results for our post-processing step using SECOND-IoU on the Waymo $\rightarrow$ KITTI setting. . . . .                                                                                                                                                                                                                                                | 68  |
| 3.11 | Evaluation of SEE-VCN on ground truth boxes with 0, 30 and 50 points or more. . . . .                                                                                                                                                                                                                                                                      | 68  |
| 4.1  | Overview of each dataset. Country refers to the country in which the dataset was collected. . . . .                                                                                                                                                                                                                                                        | 85  |
| 4.2  | UDA from Waymo/Lyft to <b>nuScenes</b> . We report $AP_{BEV}/AP_{3D}$ . Best results for each source-target pair are highlighted in <b>bold</b> . GT-Trained uses GT labels to fine-tune the pre-trained detectors with no frame accumulation. MS3D is compared with SN [Wang et al., 2020], ST3D [Yang et al., 2021] and L.D [Wei et al., 2022] . . . . . | 87  |
| 4.3  | UDA from nuScenes/Lyft to <b>Waymo</b> . We report $AP_{BEV}/AP_{3D}$ . MS3D is compared with SN [Wang et al., 2020] and ST3D [Yang et al., 2021]. . . . .                                                                                                                                                                                                 | 88  |
| 4.4  | UDA from nuScenes/Waymo to <b>Lyft</b> . We report $AP_{BEV}/AP_{3D}$ . GT-Trained uses Lyft labels to fine-tune the pre-trained detector with no frame accumulation. MS3D is compared with SN [Wang et al., 2020] and ST3D [Yang et al., 2021]. . . . .                                                                                                   | 88  |
| 4.5  | Ablation study of box fusion methods for multiple detector predictions on Waymo as target domain. KBF is compared with NMS [Neubeck and Van Gool, 2006], WBF-C and WBF-P [Solovyev et al., 2021]. . . . .                                                                                                                                                  | 91  |
| 4.6  | Ablation study of our static box labelling using a rolling window of $H$ frames for KBF on Waymo as target domain. For $H = 0$ we replace static boxes with their 16-frame detections. For "Single box", we use KBF of all observed frames. . . . .                                                                                                        | 93  |
| 4.7  | Overview of datasets used for this study. . . . .                                                                                                                                                                                                                                                                                                          | 96  |
| 4.8  | Multi-frame vs single-frame detectors on cross-domain data. We show $AP_{3D}$ gain ( $AP_{multi} - AP_{single}$ ) on official validation datasets. . . . .                                                                                                                                                                                                 | 98  |
| 4.9  | Pre-training SECOND-IoU with 4 frames on the Waymo dataset generally outperforms ST3D when testing on nuScenes data. Source and Target are given as Dataset / $T_{\Delta MAX}$ . . . . .                                                                                                                                                                   | 98  |
| 4.10 | Varying the number of inference frames for multi-frame PV-C on cross-domain data. The source and target columns are written as dataset / $T_{\Delta MAX}$ . Results reported with $AP_{3D}$ at IoU=0.7 and 0.5 for "Vehicle" and "Pedestrian" respectively. . . . .                                                                                        | 100 |



|      |                                                                                                                                                                                                                                                                  |     |
|------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----|
| 4.11 | Evaluation of 3D detectors from different sources on <b>Waymo</b> as the target domain. Results reported with $AP_{3D}$ at $IoU=0.7$ and $0.5$ for "Vehicle" and "Pedestrian" respectively. . . . .                                                              | 101 |
| 4.12 | Evaluation of 3D detectors from different sources on <b>Lyft</b> as the target domain. Results reported with $AP_{3D}$ at $IoU=0.7$ and $0.5$ for "Vehicle" and "Pedestrian" respectively. . . . .                                                               | 102 |
| 4.13 | Evaluation of 3D detectors from different sources on <b>nuScenes</b> as the target domain. Results reported with $AP_{3D}$ at $IoU=0.7$ and $0.5$ for "Vehicle" and "Pedestrian" respectively. . . . .                                                           | 103 |
| 4.14 | Performance reduction when comparing pre-trained detectors on different source domains $AP_{CD}$ to training from scratch on target domain data $AP_{oracle}$ where $AP_{\% \text{ Reduction}} = \frac{AP_{CD} - AP_{oracle}}{AP_{oracle}} \times 100$ . . . . . | 104 |
| 4.15 | <b>Waymo</b> as the target domain for unsupervised domain adaptation. Detectors for all methods are trained on 4-frame accumulation ( $T_{\Delta MAX} = 0.3$ ). . . . .                                                                                          | 118 |
| 4.16 | <b>nuScenes</b> as the target domain for unsupervised domain adaptation. . . . .                                                                                                                                                                                 | 119 |
| 4.17 | <b>Lyft</b> as the target domain for unsupervised domain adaptation. Detectors for all methods are trained on 3-frame accumulation ( $T_{\Delta MAX} = 0.4$ ). . . . .                                                                                           | 120 |
| 4.18 | VMFI and TTA for ensembling. Precision (P) and Recall (R) range breakdown on Waymo point clouds. Source is given as Training dataset / $T_{\Delta MAX}$ / Detector. . . . .                                                                                      | 121 |
| 4.19 | Ensemble combinations for Lyft (L) and nuScenes (N) detectors on Waymo point clouds. We report $AP_{3D}$ in the ranges 0-30m, 30-50m and 50-80m. . . . .                                                                                                         | 122 |
| 4.20 | Label breakdown of KBF and temporal refinement in MS3D++. Precision (P) and Recall (R) breakdown by range for Waymo point clouds. Delta $\Delta$ is the difference between Ensemble and Refined. . . . .                                                         | 123 |
| 4.21 | Iterative Label Refinement. Precision (P) and Recall (R) range breakdown of the pseudo-label quality for each self-training round. Delta $\Delta$ is the difference between round 4 and round 1. . . . .                                                         | 125 |

# Nomenclature

## List of Acronyms

|               |                                 |
|---------------|---------------------------------|
| <b>UDA</b>    | Unsupervised Domain Adaptation  |
| <b>FOV</b>    | Field Of View                   |
| <b>SOTA</b>   | State-Of-The-Art                |
| <b>STDA</b>   | Single-Target Domain Adaptation |
| <b>MTDA</b>   | Multi-Target Domain Adaptation  |
| <b>MSDA</b>   | Multi-Source Domain Adaptation  |
| <b>Lidar</b>  | Light Detection And Ranging     |
| <b>RANSAC</b> | RANdom SAmple Consensus         |
| <b>CD</b>     | Chamfer Distance                |
| <b>KDE</b>    | Kernel Density Estimation       |
| <b>KBF</b>    | KDE Box Fusion                  |
| <b>TTA</b>    | Test-Time Augmentation          |
| <b>SC</b>     | Surface Completion              |
| <b>AP</b>     | Average Precision               |
| <b>IoU</b>    | Intersection Over Union         |
| <b>BEV</b>    | Bird's Eye View                 |
| <b>NMS</b>    | Non-Maximum Suppression         |
| <b>MLP</b>    | Multi-Layer Perceptron          |
| <b>BPA</b>    | Ball Pivoting Algorithm         |
| <b>VRES</b>   | Vertical Resolution             |

# Glossary

**domain:** A general term that refers to the characteristics of a dataset such as the sensor setup used, location it was recorded in, weather, post-processing methods etc.

**in-domain:** Training and testing a model on the same sensor setup and deployed in a fixed testing environment.

**cross-domain:** Training a model on one sensor setup and environment and deploying it on a completely different sensor setup and/or environment.

**source domain:** The labelled dataset that a model is trained on

**target domain:** An unlabelled dataset that we would like to adapt a pre-trained model to

**supervised:** Training a model with human annotated labels.

**unsupervised:** Training a model without human annotated labels.

**pre-trained:** A model that has been trained on a labelled source dataset.

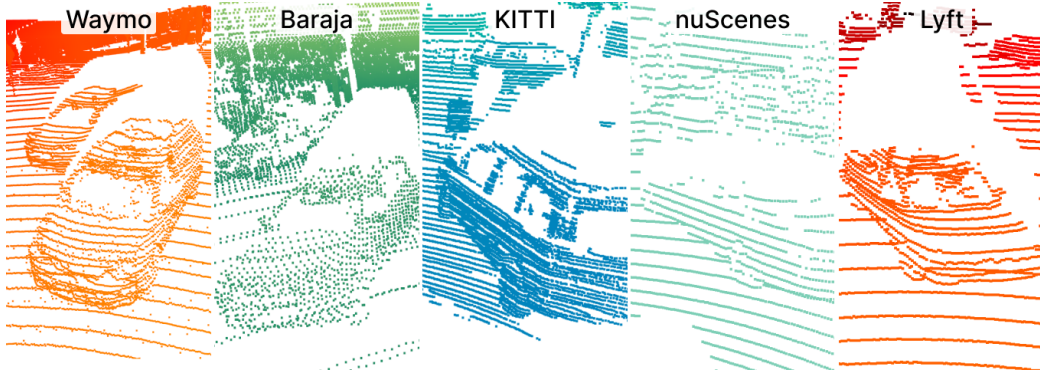
**pseudo-labels:** Labels that are generated automatically, and not from manual human labeling.

# Chapter 1

## Introduction

3D object detection on point clouds aims to localize and recognize 3D objects from a set of 3D points. It is a fundamental 3D scene understanding task and is extensively adopted in many real-world applications such as autonomous driving, robotics and intelligent transportation systems. Light Detection And Ranging (Lidar) has grown in popularity over the years, establishing itself as an indispensable sensor for achieving precise 3D object detection due to their accurate range measurements. In recent years, lidar-based 3D object detection has made remarkable progress, benefiting from advancements in detection architectures and the availability of large-scale autonomous vehicle datasets [Sun et al., 2020, Geiger et al., 2012, Caesar et al., 2020, Kesten et al., 2019, Alibeigi et al., 2023]. However, despite their success, 3D detectors trained on lidar datasets often suffer from a significant performance drop when tested on a different lidar type or operational environment. This is largely attributed to the unique distribution of points in a 3D scene for different lidar types which cause drastic differences between the training and testing data. This domain shift between the training and testing domain can be caused not only by different lidar types used for training and testing, but also weather conditions, and geographical locations. This domain shift has been demonstrated to result in a significant 70-90% drop in detection performance [Tsai et al., 2023]. We illustrate the scan pattern difference across lidars in Fig. 1.1, where cars at similar ranges can appear significantly different

across datasets, therefore affecting the generalizability of detectors.



**Figure 1.1** – Scan patterns of different lidar datasets.

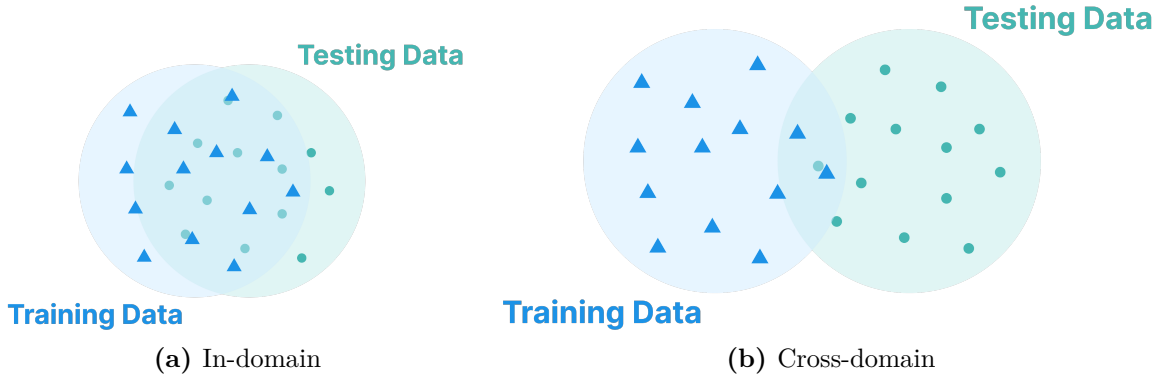
In real-world applications, model degradation from the domain shift can occur when deploying a pre-trained detector to upgraded lidar sensors, necessitating the collection and annotation of massive datasets specific to each new sensor. Additionally, mass-produced robots and vehicles typically use lidars with lower point density compared to large-scale public datasets commonly used for 3D detector training, therefore also suffering from model degradation. In both cases, collecting and manually annotating data for re-training detectors can be costly and impractical. Therefore there is a strong need for approaches that leverage existing labelled datasets (source domain) to enable 3D object detection on a range of lidar types and operational environments (target domains). We refer to this task as Unsupervised Domain Adaptation (UDA).

Currently, mechanically rotating lidars are the predominant type of lidar in large-scale autonomous driving datasets that 3D detectors are developed and trained on. These use a rotating array of laser emitters to form ring-like point clouds. The number of lasers in the rotating array, often referred to as “beams” or “rings”, as well as the beam divergence caused by the horizontal and vertical spacing between lasers, contribute significantly to the unique point distribution of the point cloud. KITTI [Geiger et al., 2012] is a pioneering dataset for 3D object detection in autonomous driving with a 64-beam lidar for data collection. However, KITTI only captures 22 scenes and provide 3D annotations that are in the Field Of View (FOV) of the front-facing camera. Objects are also labelled only if they are taller than 25 pixel height,

therefore limiting the progress of 3D detector advancements for 360° FOV, as well as occluded and far-range objects. Recent years have seen a shift in focus towards 3D detector research and development on nuScenes [Caesar et al., 2020] and Waymo [Sun et al., 2020] datasets, both of which are large-scale datasets with 1000 scenes containing bounding box annotations for 360° FOV. The quality of labelling is also improved compared to KITTI, with a significant number of labelled ground-truth boxes for sparsely observed objects (less than 5 points). nuScenes uses a 32-beam lidar with a large vertical angular resolution, leading to a larger beam divergence, while Waymo uses a 64-beam lidar that densely concentrates its rings on the horizon. To date, these two datasets remain the prominent benchmark for new State-Of-The-Art (SOTA) 3D detectors [Wang et al., 2023, Chen et al., 2022, Liu et al., 2023].

This thesis aims to address the domain shift and facilitate the integration of cutting-edge 3D detector research across a range of lidar technologies. One strategy to alleviate the performance drop involves acquiring the same sensors as those used by large-scale datasets. However, this is not always feasible. For example, lidars used by Waymo are not commercially available and the nuScenes lidar has only 32-beams which has limited functionality due to its sparsity. Furthermore, this strategy inadvertently narrows the consideration of alternative lidar solutions that could be better suited for specific tasks or environments. Another solution involves manually annotating data for each new lidar and environment to retrain the 3D detector. However, such an approach can be prohibitively expensive and time-consuming. Concurrent with advancements in 3D detector research, the lidar manufacturing landscape has continued innovating, with 128-beam lidars now being commonplace among manufacturers like Ouster, Velodyne and RSLidar. Some lidar manufacturers companies [Pulikkaseril and Langdale-Smith, 2020] have also adopted software-definable scan pattern strategies for configuring high-resolution lidar. This increasing prevalence of different lidar types on robotic platforms further highlights the need for UDA approaches to bridge the gap between rapidly advancing 3D detector research and the expanding array of lidar technologies.

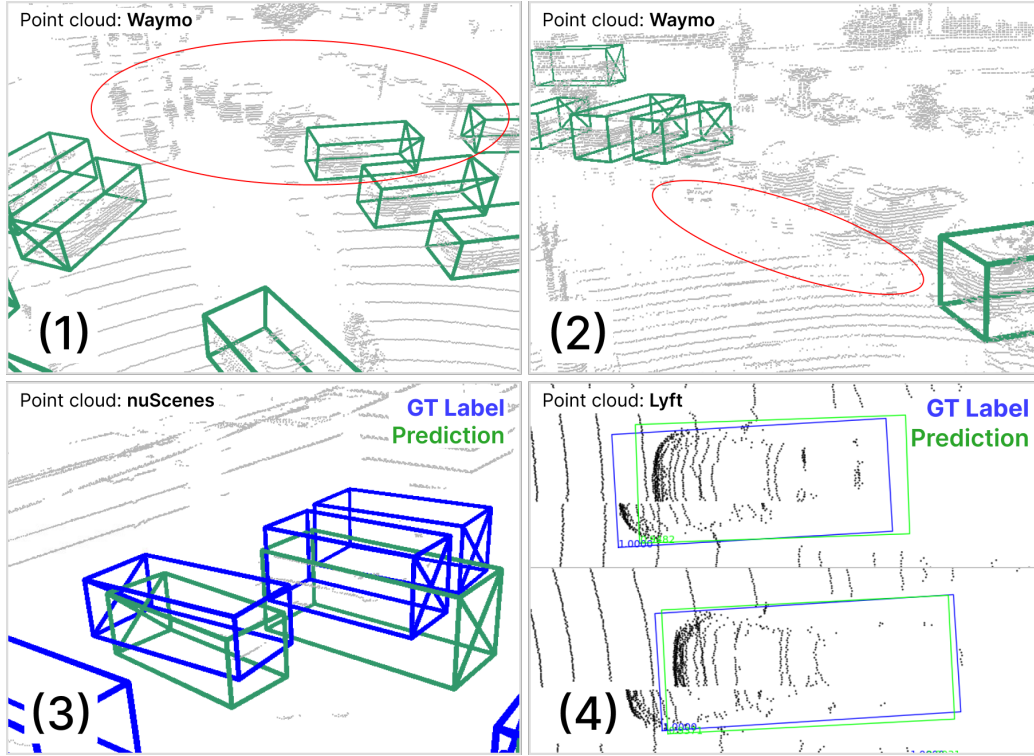
We first establish a clear distinction between the in-domain and cross-domain settings



**Figure 1.2** – Illustration of in-domain and cross-domain settings for model deployment.

to clarify the UDA task, as illustrated in Fig. 1.2. The in-domain setting encompasses the majority of 3D detector research where models are trained and tested on the same sensor setup, and deployed in the same environment. In this setting, the training dataset is often an adequate representation of the testing domain. This leads to a strong performance on the test set due to the model’s exposure to similar samples in its training dataset, with exceptions arising mainly when faced with samples that are infrequently encountered within the training dataset. This can include unfamiliar weather artefacts, obscure road objects or rare events such as a car crash. In the cross-domain setting, models are trained on one sensor setup and environment, but deployed in a different sensor setup or environment. Deploying 3D detectors in such a setting leads to significantly lower baseline performance. In this case, the training data is often a poor representation of the testing domain, especially in the case of training and testing on different lidar types. While differences in scan pattern across lidar types are a large component of the domain gap, the performance degradation could also be affected by other factors such as object size statistics across countries [Wang et al., 2020], labelling guidelines, and unfamiliar weather conditions [Zhang et al., 2021]. As a result, 3D detectors tested on cross-domain data exhibit various idiosyncrasies such as missing or poorly localized detections for densely observed objects, unreliable confidence scores and an increase in false positives. We visualize a few of the key cross-domain challenges in Fig. 1.3.

Specific to lidar-based 3D object detection, the largest factor of the domain gap for



**Figure 1.3** – Challenges encountered by testing 3D detectors in a cross-domain setting.

(1) Missed detections despite dense observations of objects due to the scan pattern gap. (2) Weather artefacts appear different across lidar types, causing missed detections. (3) Box height estimation is challenging for sparse lidar. (4) Lack of ego-motion compensation causes poor evaluation and transfer of pre-trained model to other test domains.

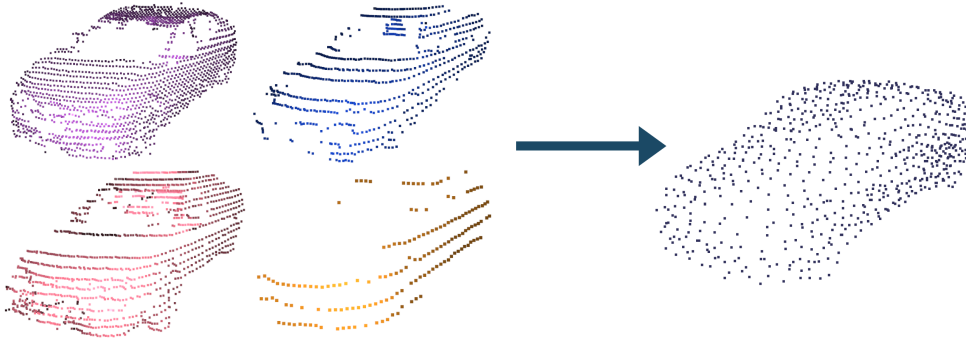
cross-domain adaptation is the scan pattern discrepancy across lidar types. Unlike images which are structured in a 2D grid representation, point cloud data is unstructured when representing objects in 3D space. Each lidar has their own specific vertical/horizontal angular resolution, number of beams, methods for handling of artefacts, noise profile and range capabilities. This scan pattern difference across lidars leads to a representation of objects that is unique for each lidar. This can cause 3D detectors to be confused in the identification of object classes when trained on one lidar and tested on a different type of lidar. For example, a detector trained on a 32-beam lidar predicts many false positives for pedestrians and vehicles when tested on a 64-beam lidar, particularly at farther ranges. This can be attributed the fact that detectors trained on a 32-beam lidar are optimised to make highly confident



predictions from sparse observations (e.g., a single lidar ring) for objects beyond 30m range. In a cross-domain context, this leads to noisy predictions due to a different distribution of points between source and target domains, particularly for semi-occluded or far-range structures.

## 1.1 Thesis Objectives

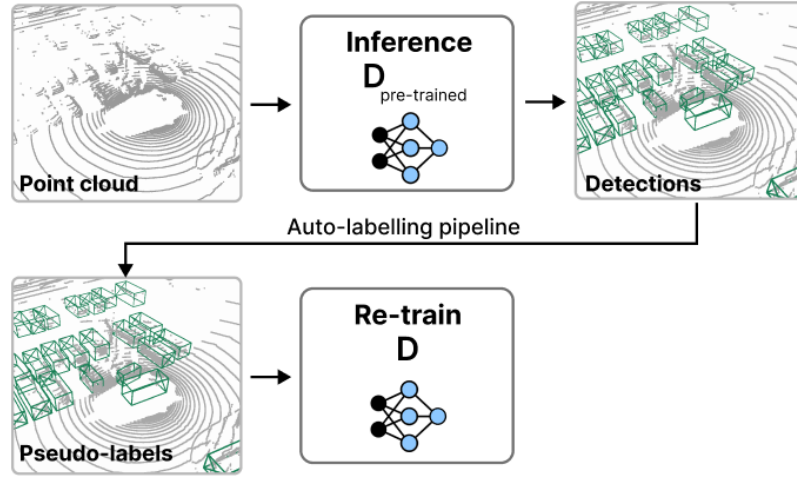
The continuous advancement of lidar manufacturing in robotics constantly outpaces lidars used in existing large-scale research datasets due to the significant effort required to produce such datasets. This thesis aims to bridge this gap by introducing UDA approaches that enable companies and research groups to effectively utilize state-of-the-art 3D detector research with various lidars technologies. As a result, our approaches focus on being universally applicable for a range of lidar scan patterns and to account for various factors of the domain gap. We adopt two distinct approaches to tackle the domain gap: Domain-invariant representation and Auto-labelling.



**Figure 1.4** – Scan pattern normalization of objects across various lidar types.

The **domain-invariant representation** approach, as illustrated in Fig. 1.4, focuses on enabling 3D detectors to be deployed on a range of lidar types without requiring detector re-training. In general our approach obtains a stronger baseline performance across different lidar types, and has better object size estimation due to consistent point cloud representations. This approach focuses on scan pattern normalization with surface completion using camera as a complementary sensor to bridge between

various lidar types. To effectively use the camera as a bridge between lidar types, accurate extrinsic calibration is required between the camera and lidar. Therefore, we first investigate the various issues present in camera-lidar calibration and propose a calibration pipeline that obtains accurate extrinsic parameter estimation with a simplified data collection process.



**Figure 1.5** – Auto-labelling for re-training detectors on a new lidar dataset.

The second approach uses automatic labelling, i.e. **auto-labelling**, to generate labels for new lidar datasets in an offline setting. The generated labels are subsequently used to re-train 3D detectors on the new dataset by replacing human annotated labels in a supervised training setup as shown in Fig. 1.5. In this approach, we place a heavy emphasis on domain generalization to support the automatic labelling of any type of lidar data, both now and in the future. To achieve this, we introduce a framework that leverages existing pre-trained 3D detectors to automatically label new lidar datasets through detector ensembling and temporal refinement.

Through extensive experiments, we demonstrate that our approaches achieve strong 3D detection performance across a range of lidar types. We believe that our proposed approaches can provide a strong foundation for further development of lidar-agnostic UDA approaches, facilitating the deployment of cutting-edge 3D detector research on the latest lidar innovations.

## 1.2 Contributions

The main contributions of this thesis are as follows. We also reference our corresponding publication for each contribution. The enumerated publications follow the list order of Section 1.3.

- Development of a calibration pipeline that automates expert-knowledge to provide a more robust estimation of calibration parameters. To fully leverage the benefits of both camera and lidar, it is essential to have an accurate extrinsic calibration. However, the calibration process is often complicated and requires expertise in the selection of the optimal calibration poses. We propose a pipeline that quantifies the calibration expert’s knowledge of pose selection in order to make the calibration process less arduous, less time-consuming, and easily accessible. This contribution was presented in publication 1 [Tsai et al., 2021].
- Development of a lidar-agnostic 3D detection framework, presented in publication 2 [Tsai et al., 2022]. Each lidar has a representation of objects specific to its scan pattern. We propose an unsupervised multi-target domain adaptation framework in Chapter 3 that works on both fixed and adjustable scan pattern lidars without needing to train a model for each new scan pattern. Our approach focuses on transforming objects in their lidar-specific representation to a unified scan pattern representation with surface completion. We subsequently use this for training and testing any SOTA detector and in so doing, we alleviate the bias towards a specific lidar’s object representation.
- We present a surface completion network for improved scan pattern unification. We demonstrate that the use of our network in our lidar-agnostic framework results in improved object size estimation and overall detection performance. In Chapter 3, we discuss how we develop a lidar-agnostic surface completion network that does not require the pre-canonicalization of objects unlike existing works. This contribution was presented in publication 3 [Tsai et al., 2023a].

- In chapter 4, we present an effective box fusion method for fusing predictions from multiple detectors using Kernel Density Estimation (KDE). Through our experiments, we observe that our fused boxes consistently outperforms the best individual detector’s predictions. This box fusion enables detector ensembling in both supervised and unsupervised scenarios. This contribution was presented in publication 4 [Tsai et al., 2023b].
- Development of an auto-labelling framework for adaptation of 3D detectors to unfamiliar target domains in Chapter 4, and published in publication 4 [Tsai et al., 2023b]. We utilize our box fusion method to demonstrate that fusing pre-trained detectors from multiple source domains can improve detector performance on unseen target domains. Experiments demonstrate that detectors re-trained with our automatically generated labels achieve high performance on new target domains, including improved far-range 3D detection capabilities. Furthermore, our developed framework is able to generate high quality labels for a variety of scan patterns, resulting in efficient 3D detector adaptation for a range of lidar types.
- In Chapter 4, we present an in-depth study on the baseline performance of pre-trained 3D detectors in a cross-domain setting and demonstrate how these insights can be incorporated into auto-labelling. Our experiments show that simply accumulating multiple frames in pre-training significantly boosts the domain generalization of detectors. This contribution was presented in publication 5 [Tsai et al., 2023].
- We incorporate multi-frame pre-trained detectors into an auto-labelling framework with iterative label refinement through multi-stage self-training for both vehicles and pedestrians. This is elaborated on in Chapter 4 and presented in publication 5 [Tsai et al., 2023]. The incorporation of multi-frame detectors and multi-stage training brings a further improvement in performance for 3D detectors on new target domains, achieving comparable performance to training with human-annotated labels.

## 1.3 Publications

The contributions of this thesis have been presented at relevant conferences and journals in the field of autonomous vehicles:

1. D. Tsai, S. Worrall, M. Shan, A. Lohr, and E. Nebot. "Optimising the selection of samples for robust lidar camera calibration," In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pp. 2631–2638. IEEE, 2021.
2. D. Tsai, J. S. Berrio, M. Shan, S. Worrall, and E. Nebot. "See eye to eye: A lidar-agnostic 3d detection framework for unsupervised multi-target domain adaptation," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7904–7911, 2022.
3. D. Tsai, J. S. Berrio, M. Shan, E. Nebot, and S. Worrall. "Viewer-Centred Surface Completion for Unsupervised Domain Adaptation in 3D Object Detection," In *2023 IEEE International Conference on Robotics and Automation (ICRA)*. pp. 9346-9353. IEEE, 2023.
4. D. Tsai, J. S. Berrio, M. Shan, E. Nebot, and S. Worrall. "MS3D: Leveraging Multiple Detectors for Unsupervised Domain Adaptation in 3D Object Detection" In *2023 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2023. (accepted)
5. D. Tsai, J. S. Berrio, M. Shan, E. Nebot, and S. Worrall. "MS3D++: Ensemble of Experts for Multi-Source Unsupervised Domain Adaption in 3D Object Detection" *Submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*

## 1.4 Thesis Structure

This thesis is organized as follows. Chapter 2 presents a comprehensive review of the related works for the task of unsupervised domain adaptation for lidar-based 3D object detection, including supplementary topics such as shape completion and extrinsic calibration which are used in our UDA approaches. Chapter 3 introduces our domain-invariant representation approaches, using camera as the bridge between lidar types. In this chapter, we first address the challenges in the extrinsic calibration of lidar and camera, and subsequently introduce our scan pattern normalization approaches to enable lidar-agnostic 3D detection. We demonstrate that our approach obtains a higher baseline performance on various lidar types without requiring re-training for each new lidar. Chapter 4 focuses on our auto-labelling framework which uses an ensemble of detectors, trained on various source domains to automatically label lidar datasets. The versatility of this approach lies in its capacity to tailor the 3D detector ensemble to a specific target domain making it an adaptable solution for a wide range of lidars and scenes. We show that 3D detectors trained with our generated labels obtain strong 3D detection performance, even comparable to training with human-annotated labels. Lastly, Chapter 5 concludes the thesis with a summary of all contributions and an outline of avenues for future research.

# Chapter 2

## Literature Review

This chapter presents a review of related work on unsupervised domain adaptation for lidar-based 3D object detection in the autonomous driving context, as well as supplementary knowledge for our proposed domain adaptation approaches.

### 2.1 3D Object Detection

3D Object Detection is the task of estimating an oriented 3D bounding box for objects in a scene. The bounding box can be parameterised with its centroid  $(c_x, c_y, c_z)$ , dimensions  $(l, w, h)$  and heading angle  $\theta$ . The approaches in 3D Detection can be categorised as Single-frame, Multi-frame, and Multi-modality approaches.

#### 2.1.1 Single-Frame 3D Detection

A significant portion of the current research in 3D detection is concentrated on single-frame detection. Due to the sparse and irregular nature of point clouds, there are various approaches that aim to efficiently and accurately encode spatial information from raw point clouds. The representation of point clouds can be largely categorized into three main categories: voxel-based, point-based and hybrid representations.

Voxel-based [Yan et al., 2018, Deng et al., 2020, Zhou and Tuzel, 2018, Chen et al., 2023] methods partition the point cloud into voxels that can be processed by mature Convolutional Neural Network (CNN) architectures and robust 2D detection heads. This representation offers regularity and efficient feature encoding of multi-scale features due to its strong memory locality, which also facilitates fast neighbour querying [Deng et al., 2020, Liu et al., 2019]. However, voxelization requires a high resolution (i.e., small voxel sizes) in order to preserve location information which is computationally expensive. At lower resolutions, multiple points may be assigned to the same voxel, resulting in a loss of structural details.

Point-based representations [Zhang et al., 2022, Yang et al., 2020, Shi et al., 2019] on the other hand, process raw points with PointNet [Qi et al., 2017a, Qi et al., 2017b], using symmetric functions for point feature extraction to tackle the unordered nature of point cloud data. These methods use sampled points as key points and aggregate local regions (i.e., grouping) around them, encoding precise position information. However, constructing local region sets by finding neighboring points often involves costly nearest neighbor searches due to irregular point distribution in 3D space.

Hybrid representations [Shi et al., 2020, Yang et al., 2019, Shi et al., 2020, Noh et al., 2021, Chen et al., 2017, Guan et al., 2022] explore combining multiple representations of the point cloud. For example, MV3D [Chen et al., 2017] and MVF [Zhou et al., 2020] proposes to combine BEV and range view, whilst others explore fusing points, voxels and BEV with a transformer [Guan et al., 2022]. A prominent state-of-the-art approach, PV-RCNN [Shi et al., 2020], along with its successor [Shi et al., 2023], focus on combining the strengths of both point and voxel-based methods, to achieve efficient multi-scale feature encoding while preserving precise location information. This approach alleviates the grouping inefficiency of the point-based methods by sampling fewer keypoints, and aggregating neighbouring voxel features instead of points.

A recent trend is the use of Transformers [Vaswani et al., 2017], or more specifically, its self-attention mechanism, in 3D object detection due to their success in natural language processing and 2D detection. The pioneering work of Vision Transformer



(ViT) [Dosovitskiy et al., 2020] in 2D detection applies the Transformer architecture directly on overlapping medium-sized image patches for image classification. Swin-transformer [Liu et al., 2021] refines the transformer architecture for image detection by using self-attention on local patches with a shifted window approach. This effectively tackles the issue of ViT’s quadratic increase in complexity with image size, therefore maintaining accuracy and presenting a scalable transformer-based image backbone. In 3D object detection, many approaches propose innovative uses of the self-attention module [Fan et al., 2022, Wang et al., 2023, Sheng et al., 2021] SST [Fan et al., 2022] applies self-attention on a set of voxels with a shifting window mechanism, similar to Swin-transformer. Their backbone design preserves the number of voxels and their spatial locations, therefore having the benefit of minimal information loss without expensive computation cost. DSVT [Wang et al., 2023] optimizes the design of SST for real-world applications through proposing a sliding window approach that does not require custom CUDA operations. Overall, transformer-based methods have shown improved performance over existing voxel or hybrid methods in 3D object detection, especially in the detection of smaller object classes.

### 2.1.2 Multi-Frame 3D Detection

Simple concatenation of a short sequence of point cloud frames as input to a 3D detector was popularised by the nuScenes dataset [Caesar et al., 2020]. They proposed to accumulate multiple historical point clouds by transforming them into the current ego-vehicle frame. This densified the 32-beam point cloud tremendously and gave a substantial increase in performance due to increased point resolution on all objects. This method however starts to deteriorate for dynamic objects as more frames are accumulated due to motion trails. This was similarly observed in the Waymo dataset where Yang et al. [Yang et al., 2021] demonstrated that concatenating 4 point cloud frames had the optimal performance boost before it started to deteriorate with more frames. However, they observe that for static and slow objects, accumulating more than 4 frames continued to improve detection performance. To overcome this, [Yang et al., 2021] proposed a learnable module to align relevant features between new frame

proposals and historical proposals stored in a memory bank. With their approach, they demonstrate that accumulating more than 4 frames can continue to boost model performance even on dynamic objects.

Another set of works in multi-frame 3D detection focus on offline 3D detection. Offline 3D object detection [Qi et al., 2021, Ma et al., 2023, Fan et al., 2023a] typically focuses on the use of pre-trained detectors to label new, unlabelled data for fine-tuning existing detectors. This is also known as auto-labelling for the in-domain setting. In an offline setting, 3D detectors are not constrained by runtime performance or memory. This allows methods to use high performing 3D detectors and trackers that may not necessarily be efficient. 3DAL [Qi et al., 2021] showed that accumulating 100+ frames of a sequence can be effectively utilized by separating static and dynamic vehicle box refinement. They propose separate networks to encode static and dynamic objects. For static objects, 3DAL accumulates all the observations of the object and uses a PointNet to segment foreground points and refine a single box in world coordinate for the object. For dynamic objects, they propose to additionally encode the trajectory of all frames to propose a box for the current frame. CTRL [Fan et al., 2023b] extends 3DAL with a bidirectional tracking module to address missing detections, which are then fed into a track-centric learnable module for refinement.

### 2.1.3 Multi-Modal 3D Detection

A growing trend is the fusion of image and point clouds for 3D object detection. These methods have shown that incorporating the high-resolution shape and texture information from cameras can improve the performance of lidar-only detectors especially on smaller object classes such as pedestrians or cyclists. MVP [Yin et al., 2021] and PointPainting [Vora et al., 2020] demonstrate how decorating point clouds with image information can lead to improved detection performance. These methods however, rely on externally trained camera feature extractors which can have additional computation costs and sub-optimally extracted features. To address this, DeepFusion [Li et al., 2022] proposes to integrate camera and lidar features in an end-to-end manner

with a cross-attention module to align deep features from the two modalities. DeepFusion has also been shown to be more robust to point cloud noise such as poor lidar reflections. BEVFusion [Liu et al., 2023] adopts a novel approach that achieves strong performance without decorating lidar points in the input [Vora et al., 2020] or feature space [Li et al., 2022]. Instead, BEVFusion maps all sensor modalities to a shared BEV space, making it a generic multi-task, multi-sensor perception framework.

### 2.1.4 Detection Head

The final layers of a detection network are referred to as the detection head. These are often combined with various point representations and transformer-based methods above and are responsible for processing feature maps into bounding box and object class predictions. These can be classified into two categories: anchor-based and anchor-free. Anchor-based detection heads such as the Region Proposal Network (RPN) [Yan et al., 2018, He et al., 2017] place predefined anchors at uniform intervals on the feature map. The model is thereafter trained to regress an offset from these anchor box priors to the final bounding box. Anchors sizes are often selected based on the mean sizes of objects in a training dataset. The benefit of anchor-based methods are that axis-aligned boxes can be a very strong prior for regressing vehicles due to the nature of driving scenes where vehicles are often axis-aligned. The disadvantage however, is that objects come in various sizes and orientations, making it challenging to robustly regress accurate object size and orientation [Yin et al., 2021]. Anchor-free [Yin et al., 2021, Shi et al., 2019, Hu et al., 2022, Zhang et al., 2022] approaches focus on predicting foreground points [Shi et al., 2019, Zhang et al., 2022] or centroids [Yin et al., 2021] of objects for regression of the bounding boxes. This addresses the bias of anchor-based methods towards axis-aligned predictions, as well as removes the burden of choosing IOU thresholds for the anchor target assignment for new classes or datasets.

## 2.2 Unsupervised Domain Adaptation

### 2.2.1 Overview

UDA focuses on adapting a model trained on a labelled dataset (source domain), to a new, unlabelled dataset (target domain). UDA works can be categorised into domain-invariant representation [Yi et al., 2021, Tsai et al., 2022, Tsai et al., 2023a], adversarial methods [Qin et al., 2019, Saito et al., 2019], self-training [Yang et al., 2021, Yihan et al., 2021, Tsai et al., 2023b, Saltori et al., 2020, Luo et al., 2021, Peng et al., 2023], amongst others. Domain-invariant feature representation is a common approach where explicit distance measures such as maximum mean discrepancy [Rozantsev et al., 2018] and correlation alignment [Rozantsev et al., 2018] are used to learn feature representations that are invariant between the two domains. Domain-invariant features can also be learnt with adversarial training [Ganin and Lempitsky, 2015, Saito et al., 2018, Qin et al., 2019, Sankaranarayanan et al., 2018]. Instead of aligning the feature space, some methods align the data representation. This could be in the form of using adversarial networks to align the appearance of the source to resemble the target domain [Isobe et al., 2021, Hong et al., 2018], or in transforming all input data into a domain-invariant representation [Yi et al., 2021, Tsai et al., 2022, Li et al., 2019]. Another category of works explore self-training [Yang et al., 2021, Zou et al., 2018, Lian et al., 2019, Li et al., 2020] to generate pseudo-labels for training on the target dataset. Within the UDA task, we define a few key terms: (1) single-source refers to the use of a single source domain for adaptation; (2) multi-source is the use of multiple source domains; (3) single-target is the adaptation to a single target domain; (4) multi-target (aka. Domain Generalization) is the adaptation to multiple target domains; (5) source-free approaches do not make use of any source data or their labels in adaptation. UDA in 3D detection encompasses a range of domain gap factors including scan pattern differences, weather, geographical locations and object sizes.

### 2.2.2 Single-target, single-source

Most UDA approaches for 3D object detection focus on single-source to single-target adaptation, which we also refer to as Single-Target Domain Adaptation (STDA). ST3D [Yang et al., 2021] proposed a self-training framework with a thresholding method that selects high-confidence detections as pseudo-labels, however this can lead to the discarding of correct labels with low-confidence. CL3D [Peng et al., 2023] proposed the use of domain-invariant features, such as geometry and temporal motion, for template matching. This involved matching target domain geometry and motion features to the learned source geometry and motion features. This was subsequently used to reweigh the confidence score of pseudo-labels for higher quality labels. Some works address specific components of the domain shift such as SRDAN [Zhang et al., 2021] which focuses on time-of-day, weather and synthetic-to-real domain adaptation through learning object geometry for a domain-invariant representation. [Wang et al., 2020] highlighted that the main domain gap issue for adaptation of US-based datasets [Sun et al., 2020, Caesar et al., 2020] to German-based datasets [Geiger et al., 2012] is in the sizes of the cars.

### 2.2.3 Multi-target

Multi-Target Domain Adaptation (MTDA), or more generally, domain generalization, aims to train solely on the source data and annotations, to learn a representation that enables more robust performance on a wide range of domains. For point cloud tasks, a few works propose to transform the entire input point cloud into a domain-invariant representation by completing its surface [Yi et al., 2021, Li et al., 2019]. However, transforming the entire point cloud is computationally expensive. Our work, SEE [Tsai et al., 2022, Tsai et al., 2023a], as presented in Chapter 3, is a multi-target domain adaptation work that proposes a partial point cloud transformation by leveraging image detectors to transform objects into a domain-invariant representation selectively. However, the proposed approach adds additional steps at inference, therefore affecting the real-time capability of adapted 3D detectors.

### 2.2.4 Multi-source

Multi-source domain adaptation (MSDA) approaches use a bag of source domains, where each domain is correlated to the target domain by different amounts. This task involves not only robustly fusing the knowledge of multiple models but simultaneously preventing the chance of negative transfer [Ahmed et al., 2021]. For example, [He et al., 2021] uses style transfer to reduce the discrepancy of multiple source image styles in a collaborative learning framework. [Yao et al., 2021] proposes to align the feature space of multiple sources through an exponential moving average parameter combination for object detection. In cross-domain 3D detection, Uni3D [Zhang et al., 2023] proposes to train a detector on multiple source datasets with statistics-level alignment in the feature space to learn data-agnostic features and prevent negative transfer. Our proposed auto-labelling framework, MS3D [Tsai et al., 2023b] is a source-free, MSDA approach that improves domain generalization on a target domain with an ensemble of pre-trained detectors, and temporal refinement for pseudo-labelling within a self-training framework. MS3D is further elaborated on in Chapter 4.

### 2.2.5 Source-free

Source-free domain adaptation is another category of works [Tsai et al., 2023b, Saltori et al., 2020, Ahmed et al., 2021] where only the pre-trained detectors and unlabelled target data are available. In a real-world setting, source data may not be available for a variety of reasons, such as privacy and security. It may also not be practical to transfer or store large-scale datasets on different platforms for adaptation [Ahmed et al., 2021]. SF-UDA [Saltori et al., 2020] is a source-free self-training approach that uses a tracker to re-score pseudo-labels. Our MS3D framework also falls under this category as a source-free approach that does not require source labels or data for adaptation.

## 2.3 Target-based Calibration

Combining 3D depth information with 2D colour and texture information allows autonomous vehicles to overcome the implicit limitations of these sensor modalities for more robust accurate sensing capabilities. To fully leverage the benefits of both camera and lidar, it is essential to have accurate extrinsic calibration to obtain the rotation and translation parameters between these two sensor frames. Target-based methods estimate calibration parameters by simultaneously observing the same calibration target from both image and pointcloud domains for feature alignment. The types of targets can be categorised into (1) Planar, such as a chessboard [Zhang and Pless, 2004, Zhou and Deng, 2012, Geiger et al., 2012, Verma et al., 2019], triangles [Park et al., 2014], and fiducial markers [Dhall et al., 2017, Beltrán et al., 2022]; and (2) Non-planar types such as trihedrons [Pusztai and Hajder, 2017], and spheres [Toth et al., 2020]. Of the existing targets, the chessboard is the most popular as its pattern provides distinct edges which make it easily identifiable in the image domain. Additionally, the chessboard is straightforward to prepare and easily accessible as it is widely used in intrinsic camera calibration, a pre-requisite of any extrinsic calibration process.

In existing target-based methods, common features extracted from a target are the plane of the target [Zhou et al., 2018, Verma et al., 2019, Pandey et al., 2010, Pusztai and Hajder, 2017, Beltrán et al., 2022], centres and vertices [Verma et al., 2019, Alismail et al., 2012, Pusztai and Hajder, 2017], and edge lines [Zhou et al., 2018]. A popular choice for fitting a line or plane in the lidar domain is RANdom SAMple Consensus (RANSAC) [Verma et al., 2019, Pandey et al., 2010, Alismail et al., 2012, Park et al., 2014, Pusztai and Hajder, 2017] as it is robust in managing the outliers from noisy lidar returns. Once the plane of the target is obtained, the normal to the plane can be used to represent the plane, often for estimation of the rotation parameters. The corners of the target, and the centre point, can also be obtained in a similar fashion by finding the intersection of the lines that are fitted to the edges of the target. When target features are extracted, the transform parameters between the lidar and camera can be estimated by computing linear systems of equations

and optimised with techniques such as Levenberg-Marquardt [Levenberg, 1944, Zhou et al., 2018, Park et al., 2014], Gauss-Newton [Geiger et al., 2012], Iterative Closest Point (ICP) [Alismail et al., 2012, Geiger et al., 2012, Dhall et al., 2017] and Genetic Algorithm [Verma et al., 2019]. The designed cost functions are usually variants of a least-squares error [Zhou and Deng, 2012, Pusztai and Hajder, 2017, Verma et al., 2019] for feature alignment and the reprojection error [Pandey et al., 2010, Verma et al., 2019, Geiger et al., 2012].

## 2.4 Real-world Point Cloud Completion

Point cloud completion approaches focus on recovering complete point clouds from partial point clouds. Most approaches [Yu et al., 2021, Yang et al., 2018, Groueix et al., 2018, Sun et al., 2021, Park et al., 2019] are proposed for synthetic 3D CAD objects [Chang et al., 2015, Wu et al., 2015] to allow evaluation with ground-truth object shapes. However, simulated sparse point clouds are not representative of the noise and sparsity of the point cloud of objects as captured from a real-world sensor. A few approaches [Yuan et al., 2018, Stutz and Geiger, 2018, Duggal et al., 2022] tackle completion for real-world lidar data, however, they rely on using ground truth bounding boxes to transform each object into object-centred coordinates. This access to ground truth annotations is infeasible in real-world applications. Conversely, there are few existing point cloud completion approaches for objects in viewer-centred coordinates. Viewer-centred coordinates represent the object in the coordinate frame aligned to the viewing perspective of the sensor. Traditional methods [Bernardini et al., 1999, Edelsbrunner et al., 1983, Kazhdan et al., 2006] are typically able to complete viewer-centred objects, though the meshes are often inadequate for sparse observations, and require heavy parameter tuning for each object. Whilst deep learning models for object-centred completion have displayed great performance, adapting these models to the viewer-centred context is non-trivial. Gu et al. [Gu et al., 2020] estimates object pose and shape given an object in viewer-centred coordinates. Their training setup uses multiple views of the object in a Siamese training setup, with the



ground truth surface as the accumulation of points from multiple views of that object. Our proposed SEE-VCN [Tsai et al., 2023a] framework, presented in Chapter 3, uses simulated lidar data to enable viewer-centred surface completion through joint pose and surface completion. They further demonstrate that completing the surfaces of objects can be used in a UDA setting to bridge between lidar types, leading to improved 3D box localization and size estimation.

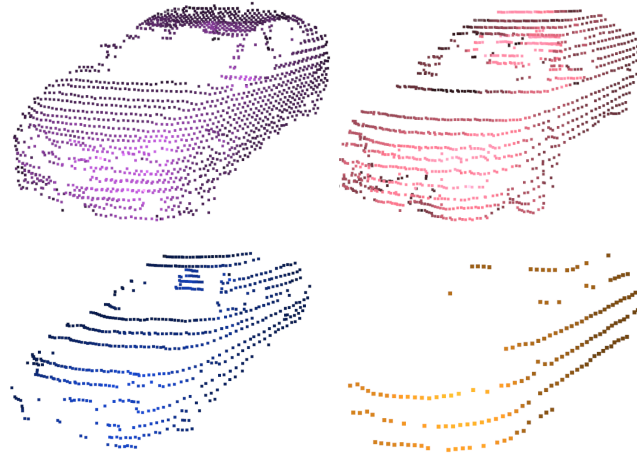
# Chapter 3

## Domain-Invariant Representation

### 3.1 Introduction

Each lidar has a distinct scan pattern, leading to a representation of objects specific to its scan pattern (Fig. 3.1). This characteristic of lidars results in performance degradation when models trained for one lidar are tested on other types of lidars. Existing works [Yang et al., 2021] use self-training to generate high-quality pseudo-labels to re-train a 3D detector on a different type of lidar. This approach has shown to be highly effective. For adjustable scan pattern lidars such as the Baraja Spectrum-Scan however, this would require training a new detector for each scan pattern which is impractical. The motivation for the approaches presented in this chapter is to therefore enable a trained 3D detector to obtain strong 3D detections on any lidar scan pattern.

In this chapter, we propose an unsupervised multi-target domain adaptation framework designed to work on both fixed and adjustable scan pattern lidars without the need to train a model for each new scan pattern. Our framework enables the performance of SOTA models, trained on single lidar dataset, to be adapted for multiple unlabelled lidar datasets of various, distinct scan patterns. Our approach focuses on transforming objects in their lidar-specific representation, to a unified scan pattern representation. We subsequently feed this into any SOTA detector and in so doing, we



**Figure 3.1** – Sampling discrepancy of cars at 10m distance captured by different lidars. Top left: Baraja Spectrum-Scan, an adjustable scan pattern lidar; Top right: Waymo Open Dataset [Sun et al., 2020] (64-beam); Bottom left: KITTI Dataset [Geiger et al., 2012] (64-beam); Bottom right: nuScenes Dataset [Caesar et al., 2020] (32-beam).

alleviate the bias towards a specific lidar’s object representation. Our approach uses the camera as a complementary sensor modality, acting as a bridge between various scan patterns by proposing image regions through instance segmentation. The idea is to enable an image detector, trained in a supervised setting, to act as an anchor for a lidar that has the capability to change its scan patterns in real-time. This approach is also valuable when different lidar types are swapped out for comparison purposes.

Recently, Segment Anything Model [Kirillov et al., 2023] demonstrated strong performance on unseen target domains and tasks without any domain adaptation, achieving competitive or even superior results compared to a fully supervised model. This is part of a recent trend in computer vision towards foundation models, which have strong generalization performance to a wide variety of deployment environments. Currently, image instance segmentation models still suffer a performance drop when deployed on cross-domain data, however we believe that strong generalizable instance segmentation models will also be developed in the future. This would therefore allow our approach to be used effectively on any camera-lidar setup without requiring an image detector to be trained in a supervised setting.

To effectively use the camera as a bridge across various scan patterns, it is important to ensure that the camera and lidar setup are accurately calibrated. Inaccurate calibration between these sensors could result in the extraction of erroneous point cloud clusters from designated image regions, consequently undermining the efficacy of our approach.

In summary, this chapter presents a new approach to address the domain gap across lidar scan patterns, and also addresses the challenge of accurate extrinsic calibration between the camera and lidar setup. The main contributions are as follows:

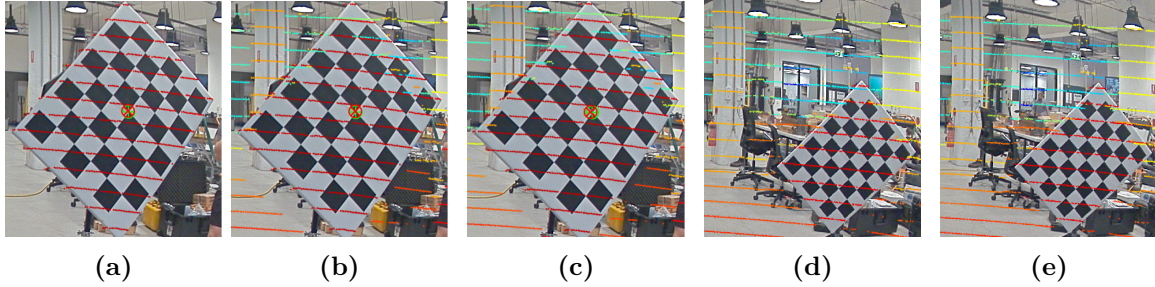
- Camera-lidar calibration pipeline that optimises the selection of calibration samples for the estimation of extrinsic parameters that optimally align the sensors to the entire scene. The code for this calibration tool is available at [https://github.com/acfr/cam\\_lidar\\_calibration](https://github.com/acfr/cam_lidar_calibration).
- An unsupervised multi-target domain adaptation framework that unifies the scan pattern representation of lidars. We release the code at <https://github.com/darrenjkt/SEE-MTDA>.
- A viewer-centred completion network that improves unified scan pattern representation through lidar-agnostic surface completion. We release the code at <https://github.com/darrenjkt/SEE-VCN>.

## 3.2 Camera-Lidar Calibration

### 3.2.1 Overview

The fundamental theoretical problem in calibration is the computation of the transform between two frames (camera and lidar in our case), which is solved in theory but challenging, and often underestimated in its application. The primary challenge is attaining robust and accurate estimation of the calibration in practical applications which is often affected by human, environmental and sensor errors.

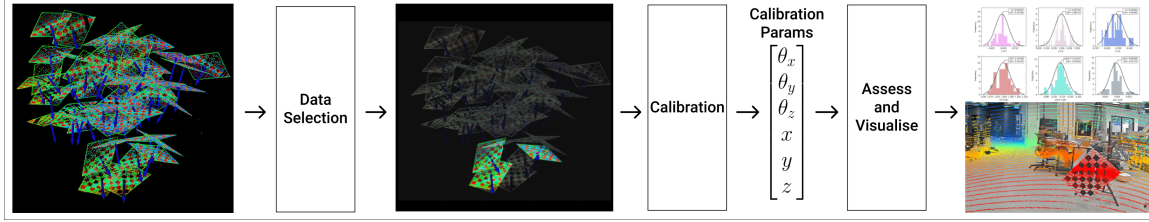
Many successful target-based methods have been proposed for the extrinsic calibration of camera and lidar, however, in practice, their calibration process is often rife with trial and error of finding the best poses for the perfect calibration. Extrinsic calibration parameters are very sensitive, as small errors in rotation or translation estimation can drastically affect the usability of the calibration result [Huang and Grizzle, 2019]. This sensitivity can most commonly be attributed to noise in the feature extraction process due to errors in sensor measurement which leads to uncertainty in plane-fitting [Park et al., 2014, Verma et al., 2019, Huang and Grizzle, 2019] or line-fitting of edges [Park et al., 2014, Verma et al., 2019].



**Figure 3.2** – Many existing methods often show the effectiveness of their approach by presenting a well-fitted target with low reprojection error (a), but not the whole scene (b). Both (a) and (b) have the same calibration parameters. The centre of the red tilted crosshair is the lidar’s reprojected board centre, and green is the camera’s board centre. The reprojection error in (a) and (b) is 1.8 pixels, which is good, however the calibration parameters don’t fit the pillar well at the background because the algorithm only focuses on optimising the chessboard fit. (c) has better overall alignment of scene features than (b) despite having 2.7 pixel error vs 1.7 respectively, showing that same target pose can be fitted with multiple different calibration parameters. (d) and (e) compare the calibration result of 3 and 9 poses respectively, showing that an equally good calibration can be achieved on both.

It is challenging to find the perfect calibration as there is no ground truth. To assess the quality of the calibration result, the reprojection error on the alignment of a target is often computed. However, while these methods craft clever cost functions to optimise the alignment of the target, it often results in an overfitted target and an underfitted scene as shown in Fig. 3.2a and Fig. 3.2b. This is because while the target might be fitted at closer ranges, the errors of estimated parameters grow with distance, becoming more noticeable on objects that are farther away. Many existing works [Zhou et al., 2018, Toth et al., 2020, Pusztai and Hajder, 2017, Kim

and Park, 2019] may often only show Fig. 3.2a and report a low reprojection error for it, however this is misleading for those who attempt to use such methods, thinking that they would get a good calibration for the entire scene. Furthermore, there are a range of possible parameters that can fit the target well, but not all of them fit well to the scene as can be seen when comparing Fig. 3.2b and Fig. 3.2c.

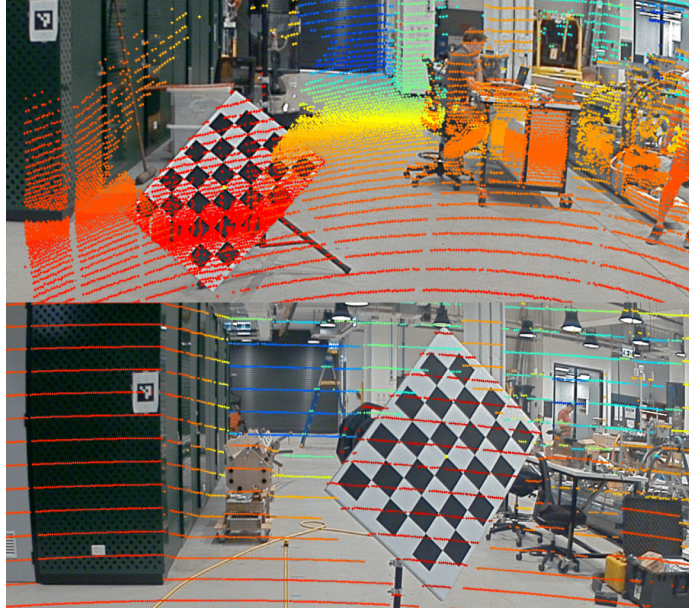


**Figure 3.3 – Our proposed calibration pipeline.** The entire pipeline (excluding the manual data collection phase) takes around 90s to assess the data and estimate calibration parameters.

The target-based calibration pipeline commonly includes 3 sections: data collection, feature extraction and the calibration algorithm. Many authors typically focus on the feature extraction and algorithmic aspect of calibration, however, less attention has been given to the crucial data collection aspect. The importance of good data is commonly stressed in many machine learning algorithms, but not often so in extrinsic calibration methods. In this work, we demonstrate that the estimated calibration parameter’s proper alignment of scene features and their uncertainty are highly influenced by the quality of data that is fed into the calibration algorithm. We adopt an existing target-based method [Verma et al., 2019] for calibration, as we believe that the existing algorithms are effective, but suffer from user error due to the challenges of collecting good quality samples.

In section, we propose a calibration pipeline, shown in Fig. 3.3, that fits well to the scene and can be used in an out-of-the-box fashion by any user regardless of calibration expertise. Our method allows the user to simply record many poses with as much variation as possible, and let our algorithm automatically select the best poses to calibrate with. In effect, our work seeks to quantify the calibration expert’s knowledge of pose selection in order to make the calibration process less arduous, less time-consuming, and easily accessible. With new lidar technologies surfacing, it

is important to have lidar-agnostic algorithms. Therefore, we demonstrate that our method performs well on both the commonly used Velodyne VLP-16 and a novel adjustable scan pattern lidar, Baraja Spectrum-Scan, as shown in Fig. 3.4.



**Figure 3.4** – Calibration result of Baraja Spectrum-Scan (top) and Velodyne VLP-16 (bottom) with our method. Projection is shown for the whole scene, to illustrate that our estimated calibration parameters are fitted to the scene. Points are coloured based on distance to the lidar, red for points closer than 3m and blue for points around 20m away. The chessboard for the Baraja Spectrum-Scan (top) has ground points projected on the board due to a large difference in perspective of the Baraja Spectrum-Scan and camera.

In particular, we propose the Variability of Quality (VOQ) metric to assess the calibration data, and give an indication of the generalisation ability and uncertainty of the estimated calibration parameters. The features used in this metric are common features used in most other target-based methods. We argue that calibration quality is not about how well a target is fitted, but rather, how well the calibration parameters fit the entire scene. To assess this we propose a robust evaluation of our proposed calibration pipeline that computes the mean and standard deviation of the reprojection error of poses that are evenly spread out in the calibration area.

### 3.2.2 Method

We adopt the calibration method in [Verma et al., 2019] which extracts the centre point and plane normal  $\mathbf{n}_L$  of a chessboard target. The rotation matrix  $\mathbf{R}_C^L$  rotates a matrix of camera normals  $\mathbf{N}_C = [\mathbf{n}_C^0, \mathbf{n}_C^1, \mathbf{n}_C^2]^T$  to align with the corresponding lidar normals  $\mathbf{N}_L = [\mathbf{n}_L^0, \mathbf{n}_L^1, \mathbf{n}_L^2]^T$  as shown in Eq. (3.1), which we can solve for, given that we know  $\mathbf{N}_C$  and  $\mathbf{N}_L$ .

$$\mathbf{R}_C^L \mathbf{N}_C = \mathbf{N}_L \quad (3.1)$$

Verma et al. [Verma et al., 2019] modified this equation to allow more normals to be included in  $\mathbf{N}_L$  and  $\mathbf{N}_C$  to improve the robustness of the algorithm by overconstraining the problem, hence side-stepping the linear dependency issues. Including more samples in the computation is a common approach in existing methods [Park et al., 2014]. However in practice, it was found that while more poses was more robust, it was also prone to overfitting the chessboard. In this work, instead of using more poses for a single calibration result, we chose to take the approach of using multiple sets of 3 poses, to obtain a robust estimate of the calibration parameters with uncertainty. Using 3 poses fully constrains Eq. (3.1), and also makes  $\mathbf{N}_C$  and  $\mathbf{N}_L$  a square matrix, which allows for more meaningful analysis of the samples. Comparing Fig. 3.2d and Fig. 3.2e shows that a good result can be obtained with sets of 3 and 9 on our baseline method if the correct poses are chosen.

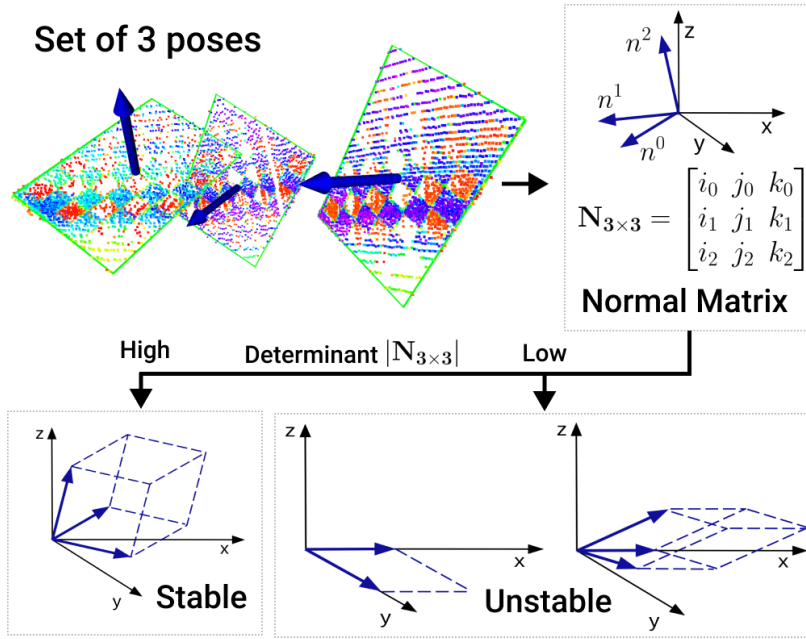
#### Linear dependence in the normal matrix

To avoid redundant poses, it is important to be able to identify linear dependence in the normal matrix as we need to compute  $\mathbf{N}_C^{-1}$  when we solve for  $\mathbf{R}_C^L$  in Eq. (3.1). When two or more rows are close to being linearly dependent in the normal matrix, the computation of the inverse becomes unstable and can adversely affect calibration results. This issue is often side-stepped when an expert in calibration knows which poses to use, however for an inexperienced user, it can be challenging to know how



to do so.

The normal matrix  $\mathbf{N}_{3 \times 3}$  outlines a parallelepiped whose signed volume can be computed by taking the determinant  $|\mathbf{N}_{3 \times 3}|$  which is the volume scaling factor of the linear transformation of  $|\mathbf{N}_{3 \times 3}|$ . When the determinant is zero, there is little to no definition of the parallelepiped, its volume can't be well computed and hence the computation of its inverse is unstable as illustrated in Fig. 3.5.



**Figure 3.5** – Illustration of linear dependency. When the normal (blue arrows) of the target planes have enough variation, it outlines a distinct parallelepiped (stable) with a large determinant, which is the volume of this parallelepiped. However, if the normals have similar or exactly the same orientations, the parallelepiped is hard to define or non-existent, and the determinant is close to zero.

The linear dependency of the matrix can also be computed by looking at the rank, or the condition number  $\kappa$  of the matrix [Zhou and Deng, 2012, Zhou et al., 2018].

$$\kappa(\mathbf{N}) = \|\mathbf{N}\| \|\mathbf{N}^{-1}\| \quad (3.2)$$

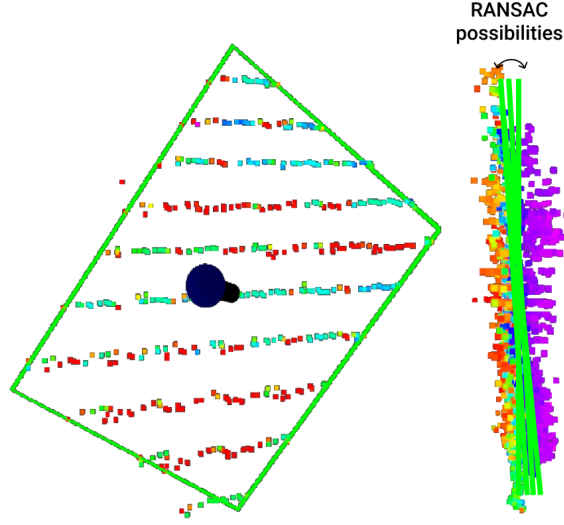
$$\kappa_{LC} = \max\{\kappa(\mathbf{N}_L), \kappa(\mathbf{N}_C)\} \quad (3.3)$$

The condition number of the normal matrix  $\kappa(\mathbf{N}_{3 \times 3})$ , shown in Eq. (3.2), gives an indication of the stability of its inverse  $\mathbf{N}_{3 \times 3}^{-1}$  where most stable result gives  $\kappa = 1$  and the least stable result giving a value that grows towards  $+\infty$ . In theory, the condition number of  $\mathbf{N}_C$  should be exactly the same as  $\mathbf{N}_L$  as the relative orientation of the poses in the set should be the same whether in point cloud or image domain. However, this is not always the case and therefore we take the worst case scenario of the two condition numbers for a more robust measure. We compute the condition number of the normal matrices using the frobenius norm to assess linear dependency.

### Error in lidar measurements

To get a good estimation of calibration parameters for target-based methods, an accurate feature extraction of the target is expected. However in reality, lidars and cameras both have errors in measurements that affect the accuracy of the feature extraction process. Lidars tend to have errors in their range accuracy; Velodyne VLP-16 for example documents a range accuracy of  $\pm 3$  cm. Our baseline method first determines the edge lines of the target with RANdom SAmple Consensus (RANSAC), then finds the intersection of edge lines to identify the corners of the chessboard. This estimation of chessboard corners is heavily influenced by the number of points that lie on the board, and the range accuracy of those points as illustrated in Fig. 3.6.

It was found empirically that the errors of chessboard measurements had a greater error at certain distances from the sensor pair (such as being too close or too far), and also when the board faced certain directions. In our approach, the chessboard corners are estimated using the intersection of the lidar-measured board edges. These corners are then used to determine the centre of the board. If the board is too close or too far, the estimated board lengths are often inconsistent, and if the board is oriented too much away from the sensors, the board length estimation also suffers. Therefore, to assess the accuracy of sensor measurements, the sum of dimension errors  $e_{dim}$  for the chessboard was used as described in Eq. (3.4) where  $l_{L,i}$ ;  $i = 0, 1, 2, 3$  are the lidar's measurement of the chessboard's edge lengths, and  $l_{P,i}$ ;  $i = 0, 1, 2, 3$  are the physically measured chessboard edge lengths. For each set of poses, we get 3 errors



**Figure 3.6** – Lidar range error may cause it to pick up a skewed board (left), or incorrectly fit a plane to the chessboard (right). Both of these affect the accuracy of feature extraction from the target.

$e_{dim,j}$ ;  $j = 0, 1, 2$ , and the mean error of the 3 poses are taken as the set's average board error  $e_{be}$  in Eq. (3.5) in millimetres.

$$e_{dim} = \sum_{i=0}^3 |l_{L,i} - l_{P,i}| \quad (3.4)$$

$$e_{be} = \frac{1}{3} \sum_{j=0}^2 e_{dim,j} \quad (3.5)$$

By assessing the error of the board measurements, we can have a quantitative measure of the accuracy in chessboard corner and centre estimation. As the translation vector is computed from aligning the centre of the chessboard in the camera and lidar domain, an error in centre estimation will lead to errors in calibration results.

### Variability Of Quality (VOQ) metric

For a set of 3 poses, we can assess the quality of the rotation parameters using the condition of the matrix, and the translation parameters using the errors in chessboard

measurements. We combine these metrics into a single equation defined in Eq. (3.6), called Variability of Quality (VOQ).

$$VOQ = \kappa_{LC} + e_{be} \quad (3.6)$$

A low VOQ score for a set of poses, leads to better generalisation of the calibration parameters to the scene as well as less variability in calibration results. On the other hand, a high VOQ score leads to overfitting and a higher standard deviation of estimated calibration results.

To estimate the calibration parameters, we propose to use 50 sets of 3 with the lowest VOQ scores out of all possible combinations of the given data. For this reason, our approach thrives when more data is provided. The calibration parameters are calculated by taking the mean of the 50 sets after filtering results that are outside 2 standard deviations from the mean using a z-score. This standard deviation of the resultant sets can also be used as an indication of the certainty of the estimated calibration parameters.

### 3.2.3 Experiments

#### Chessboard and sensor setup

Experiments were conducted using a chessboard with a 7x5 internal corners, of square length 95mm and board dimensions of  $610 \times 850$  mm. The sensors used were a Nvidia 2Mega SF3322 GMSL camera, Velodyne VLP-16 and Baraja Spectrum-Scan lidar. The VLP-16 has 16 rings with a  $360^\circ$ horizontal and  $30^\circ$ vertical FOV. The Baraja Spectrum-Scan has a  $110^\circ$ horizontal and  $25^\circ$ vertical FOV with adjustable point cloud resolution, allowing us to focus the points on regions of interest within the FOV.

The experimental setup is shown in Fig. 3.7, where it can be seen that the lidar-camera placements of the Velodyne and Baraja differ vastly; VLP-16 being close to the camera whilst Baraja sits further away. For calibration, we configured the Baraja

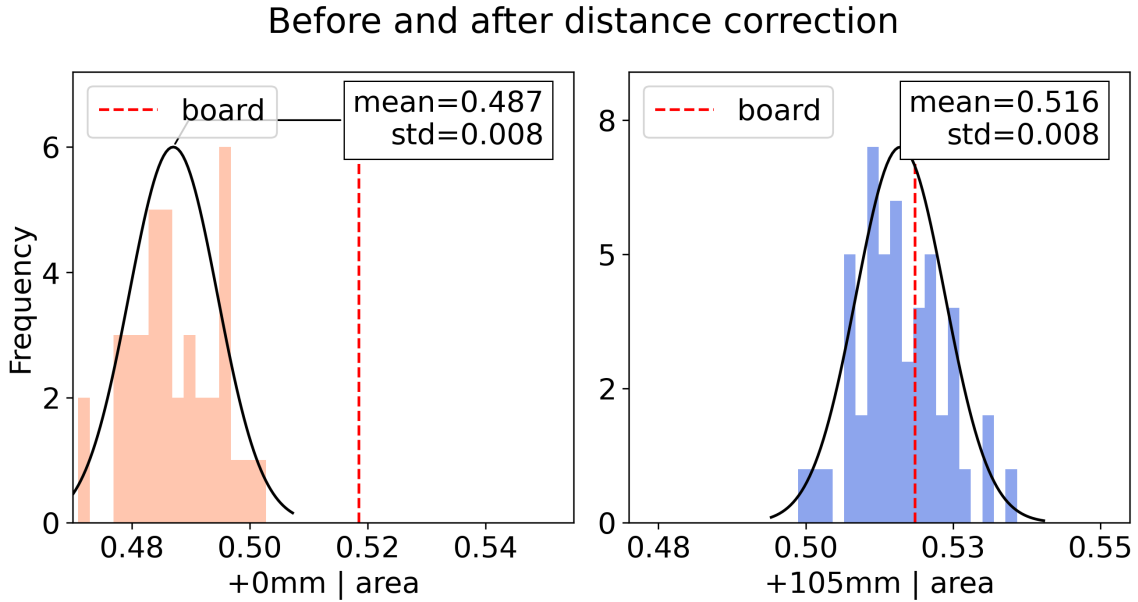


**Figure 3.7** – Sensor setup. Top-right shows the coordinate frames of the VLP-16 and camera, and the bottom-right shows the frames of the two Baraja Spectrum-Scan sensorheads and camera. The Baraja Spectrum-Scan sensorheads had to be calibrated with the camera separately.

Spectrum-Scan such that the points were always the most dense on the chessboard, which led to lower board dimension errors. We assume that the pre-requisite step of intrinsic camera calibration has been undertaken. We compared the brute-force approach of computing the VOQ for every possible combination of poses, with a clustering approach where we selected normals from clusters to minimise the condition number  $\kappa$ . However as the number of combinations are not very large for sets of 3, we adopted the brute-force approach which was faster. Our proposed calibration pipeline takes around 90s (excluding the manual data collection process).

### Data collection

A total of 50 poses were collected for both pairs of VLP-16-camera and Baraja Spectrum-Scan-camera, which generated  $\binom{50}{3} = 19600$  combinations. When capturing these poses, we ensured that there was variation in the pitch and yaw axes of the target. For the roll axis, we kept the chessboard at a  $45^\circ$  angle to form a diamond shape to better detect chessboard edges with a low resolution lidar. We noticed that the sensor measurements of both the VLP-16 and Baraja Spectrum-Scan lidars were consistently under/over estimated and therefore converted each point to its polar form and added a distance offset to it as shown in Fig. 3.8.

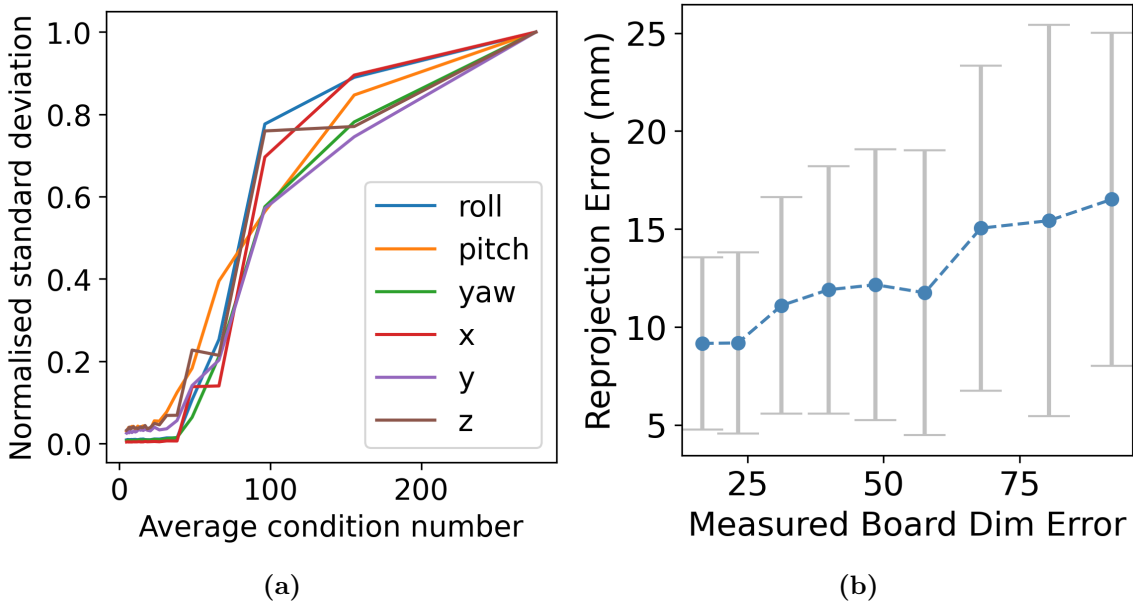


**Figure 3.8** – Distance offset for the Baraja Spectrum-Scan in metres squared. Before the offset was added, the chessboard’s area was consistently underestimated on every chessboard sample.

The VLP-16 and Baraja Spectrum-Scan lidars we used required a distance offset of -30mm and +105mm respectively. If the internal distance bias of the lidars are not accounted for, it is hard to obtain a perfect calibration for the sensor pair as all the target measurements will be inaccurate. For each set of 3 generated, we computed a calibration result with our baseline method. These 19600 sets took around 6 hours to compute with an Intel® Core i7-9700T CPU on a Dell Optiplex 7070 desktop.

## Results

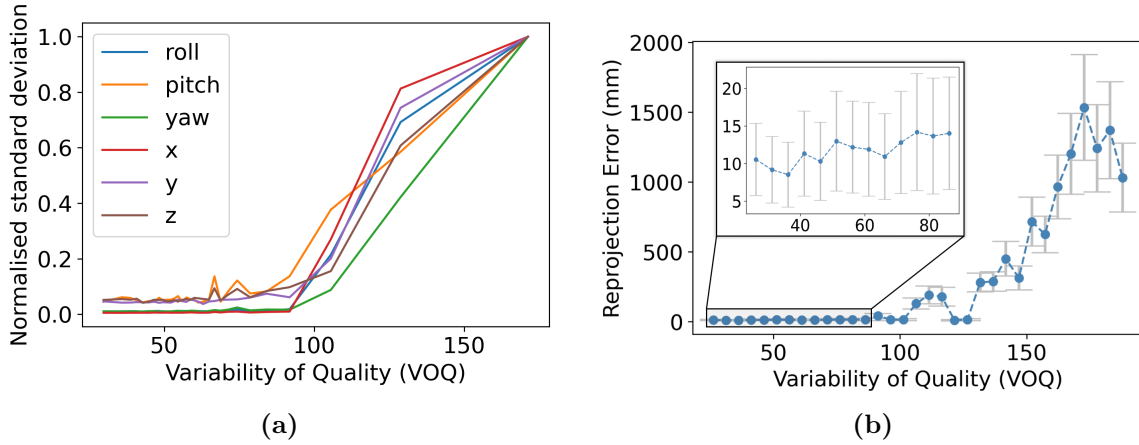
Often when assessing the quality of the calibration parameters, existing methods would project the corners or centre of the target into the image to compute the reprojection error for that particular pose. To assess the efficacy of our metric, we computed the reprojection error over 46 VLP-16 poses that are evenly distributed in the range of 1.7m to 4.5m away from the sensor pair. We did not record poses that were farther away as the low resolution of VLP-16 posed a challenge for RANSAC to accurately outline board edges. By computing the reprojection error over all 46 poses, we can get a more robust assessment of how well the estimated calibration parameters fit the entire scene, along with the standard deviation of the reprojection error.



**Figure 3.9** – Plots for VLP-16 data. (a) Demonstrates the effect of condition number on the stability of calibration results. (b) Shows that a decrease in board dimension error between the lidar-measured board dimensions and the physically measured board dimensions leads to a lower reprojection error mean and standard deviation.

The stability of the computation of rotation matrix can be gauged using  $\kappa_{LC}$  which we assess by grouping the 19600 sets and their corresponding calibration results in equal sized bins. For each bin, the mean  $\kappa_{LC}$  and standard deviation of calibration results

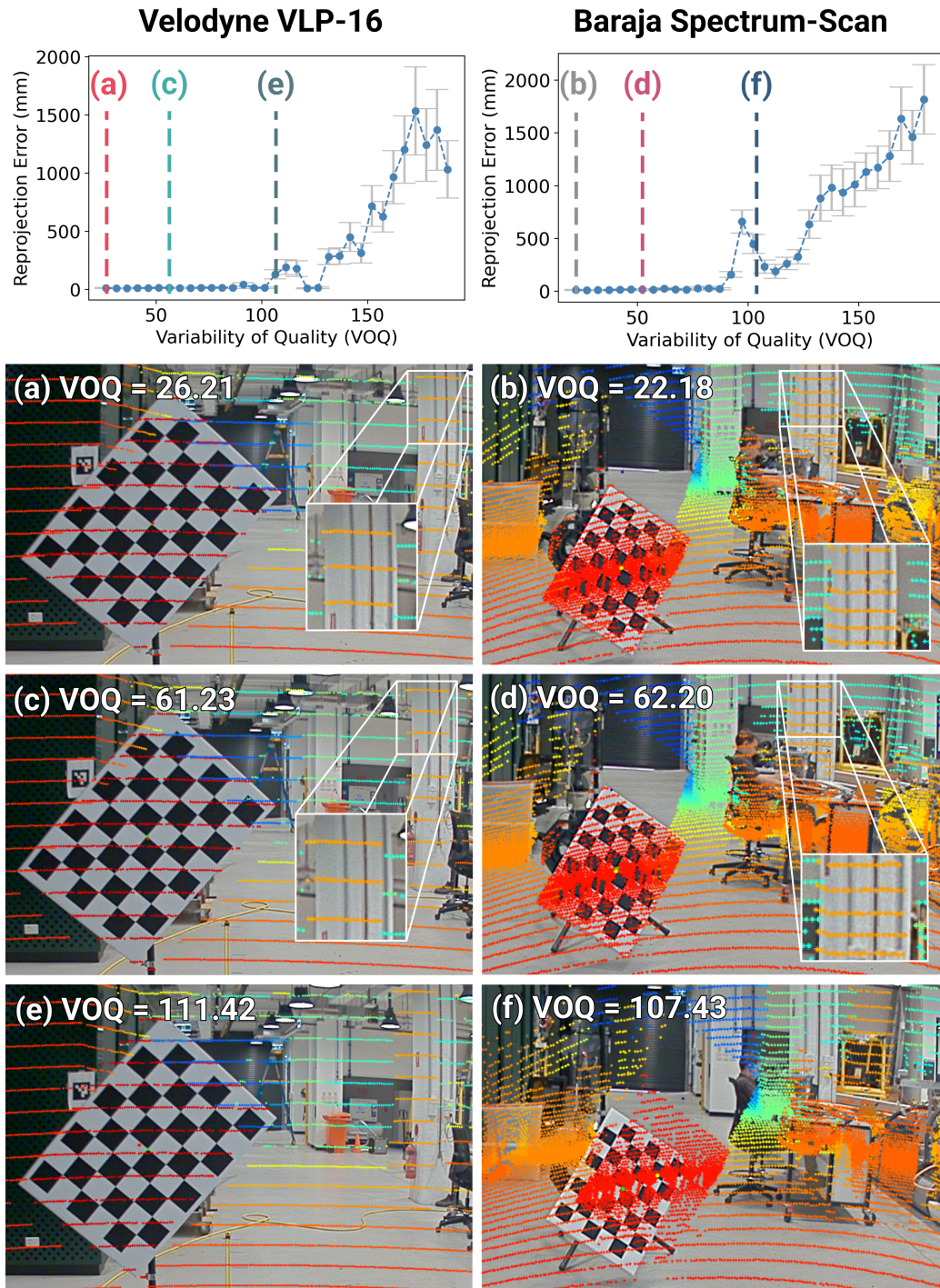
were computed with results shown in Fig. 3.9a. Here we observe that the condition number of the normal matrix starts to affect the stability of calibration results at  $\kappa_{LC} \approx 50$ . Board error was assessed by first filtering out all sets with a condition number higher than 20, then plotting board error with the respective reprojection error as shown in Fig. 3.9b. We filtered out the condition numbers higher than 20 as they would greatly affect the stability of estimated calibration parameters, and hence affect the assessment of the board dimension error. The smallest average reprojection error achieved on the test dataset was around 1cm. This reprojection error, along with its standard deviation steadily grows as the board dimension errors increase. We provide an easy set of guidelines to facilitate the data collection process in our code repository.



**Figure 3.10** – (a) Demonstrates the effect of the VOQ score on the stability of calibration results. (b) Mean reprojection errors for VLP-16 across 46 poses showing that a lower VOQ score leads to lower mean reprojection errors. After VOQ=200, the reprojection errors grow exponentially past 2000 and therefore have been excluded.

VOQ was assessed in a similar manner to the condition number and board errors. In Fig. 3.10a it can be seen that the uncertainty of calibration results grows as the VOQ score increases. Fig. 3.10b shows that an increase in VOQ doesn't simply result in more variability in calibration results, but also a higher average reprojection error of the scene, and a growing standard deviation. For VLP-16 and Baraja Spectrum-Scan lidars, the lowest average reprojection errors achieved were both in the range of 1-1.2cm, with a standard deviation of 0.4-0.6cm.





**Figure 3.11** – Projection of point cloud onto image using estimated calibration parameters at the VOQ score indicated by the dotted lines in the top row. The left is the VLP-camera calibration, and the right is the Baraja-camera calibration. As the average VOQ score of the 50 sets increases, so too does the scene misalignment (observable in the background white pillar), and reprojection error mean and standard deviation. This correlation is observed in both the VLP-16 and Baraja Spectrum-Scan lidar.

In Fig. 3.11, the lidar point clouds for both VLP-16 and Baraja Spectrum-Scan have been projected onto the images at specific VOQ scores for a visual comparison to validate the effectiveness of our approach. We emphasise on using both an average reprojection error in combination with a visualisation of the entire scene to assess calibration quality.

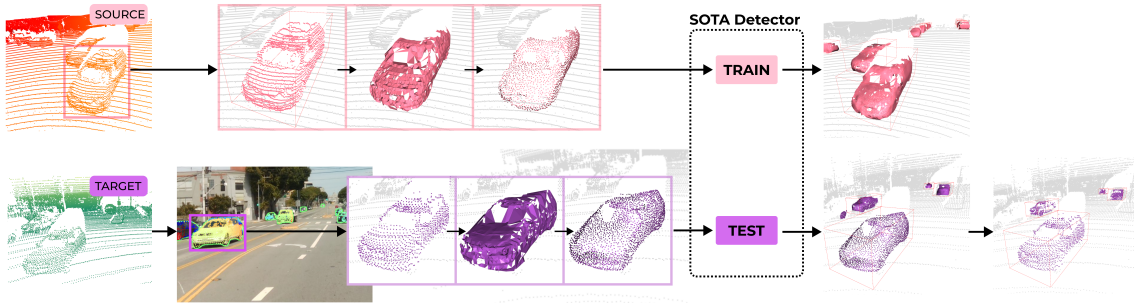
It can be seen that a lower VOQ score is correlated with a calibration estimation that fits well to the scene. When the VOQ score is slightly increased, it can be seen in the 2nd row that the reprojection error is still low, but the background is not well fitted due to the small calibration errors becoming more noticeable at a distance. Cross-referencing Fig. 3.11 with Fig. 3.10b shows us that when VOQ increases roughly from 20 to 60, both the mean and standard deviation of the reprojection error grows. As the VOQ score increases, the instability of a high condition number for  $\mathbf{N}$  starts to affect the estimated parameters, resulting in an unreliable and highly uncertain estimation. There is a spike in reprojection error at  $VOQ \approx 100$  which is caused by the sharp increase in uncertainty of the  $\kappa_{LC} \approx 100$ . The decrease that follows could be attributed to a mix of sets with high board errors but a low condition number. There is a correlation between VOQ and reprojection error, however it is only an indication of background alignment as we do not have a ground truth. We therefore conclude that it is more important to obtain a result within a low VOQ range ( $VOQ < 30$ ), than the specific VOQ value itself.

In summary, we presented a simplified calibration pipeline by quantifying expert knowledge to optimise the data selection process. We demonstrate that our calibration pipeline robustly estimates extrinsic parameters for different lidar types and lidar-camera placements. Having an accurate calibration between lidar and camera enables approaches to effectively use both sensor modalities in a common frame of reference, which we demonstrate in the following sections.

### 3.3 Scan Pattern Normalization Framework (SEE)

#### 3.3.1 Overview

In this section, we introduce our unsupervised multi-target domain adaptation framework, See Eye to Eye (SEE), shown in Fig. 3.12. Our approach uses image instance segmentation on camera images as region proposals for selective scan pattern normalization of object regions. Due to the use of the camera as a complementary modality, a strong camera-lidar calibration which we demonstrated in Section 3.2, is a key pre-requisite. We validate the efficacy of our framework with extensive experimentation using two SOTA detectors, SECOND-IoU [Yan et al., 2018] and PV-RCNN [Shi et al., 2020], on multiple public datasets as well as a manually labelled, Baraja Spectrum-Scan dataset. We demonstrate the transferability of SECOND and PV-RCNN from Waymo and nuScenes to the Baraja Spectrum-Scan to highlight SEE’s ability to adapt to unseen scan patterns. We note that our framework is applicable for single-source to both single-target and multi-target UDA settings.



**Figure 3.12 – Overview of the SEE framework**, our lidar-agnostic unsupervised multi-target DA framework for 3D object detection. Our framework allows a 3D detector trained on one source domain to be used on multiple target domains without additional training. SEE is comprised of 3 phases: object isolation, SC, and point sampling.

SEE is a simple and effective framework that enables easy adoption of SOTA detectors for any kind of scan pattern. SEE has the following advantages:

- **Scan Pattern Agnostic:** Significant improvements achieved when transferring

the performance of 3D detectors across lidars regardless if the lidar scan pattern is fixed or adjustable.

- **Unified Detector Comparison:** Any 3D detector can now be lidar-agnostic and easily benchmarked on multiple point cloud datasets.
- **Detection Performance:** SEE achieves SOTA for the Waymo  $\rightarrow$  KITTI task on all evaluated models.
- **Lidar Innovation:** With SEE, novel scan patterns from innovative lidars can now be used with any detector; these previously needed expensive manual labelling, or significant hardware resources for fine-tuning.

### 3.3.2 Method

For this section, we denote the point cloud as a set of unordered points  $\mathbf{X}_i = \{p_j |_{j=1}^{N_j}\}$ , where  $p_j$  is the  $j$ -th point in the  $i$ -th point cloud  $\mathbf{X}_i$  with  $N_j$  number of points. Each labelled point cloud has a set of annotations  $\mathbf{Y}_i = \{y_k |_{k=1}^{N_k}\}$  for  $N_k$  objects, where  $y_k$  is the  $k$ -th bounding box, parameterised by the center  $(c_x, c_y, c_z)$ , dimensions  $(l, w, h)$  and rotation  $\theta_z$  around the world frame Z-axis. The point set of all object instances in the point cloud can be represented as  $\mathbf{O}_i = \{o_k |_{k=1}^{N_k}\}$ , where  $o_k$  refers to the points of the  $k$ -th object. In this UDA scenario, we denote the source domain as  $\chi^S = \{(\mathbf{X}_i^S, \mathbf{Y}_i^S) |_{i=1}^{N^S}\}$ , where each  $i$ -th point cloud  $\mathbf{X}_i^S$ , has a corresponding annotation  $\mathbf{Y}_i^S$  for all  $N^S$  frames in the source domain. For the target domain, we are given a set of unlabelled point clouds  $\chi^T = \{\mathbf{X}_i^T |_{i=1}^{N^T}\}$ , where  $\mathbf{X}_i^T$  is the  $i$ -th point cloud in the target domain with  $N^T$  point clouds.

SEE explicitly bridges the domain gap between distinct scan patterns by transforming only the object instances into a normalised representation. We call this resultant point cloud a semi-canonical domain where only the objects  $\mathbf{O}_i^S$  and  $\mathbf{O}_i^T$  reside inside a canonical domain. This can be represented by  $\mathbf{M}_i^S = \psi(\mathbf{O}_i^S)$  and  $\mathbf{M}_i^T = \psi(\mathbf{O}_i^T)$  where  $\mathbf{M}_i^S$  and  $\mathbf{M}_i^T$  represent the  $i$ -th semi-canonical point cloud. We train the 3D object detector  $\rho(\mathbf{M}_i^S)$  on this semi-canonical source domain using labels  $\mathbf{Y}_i^S$ ,

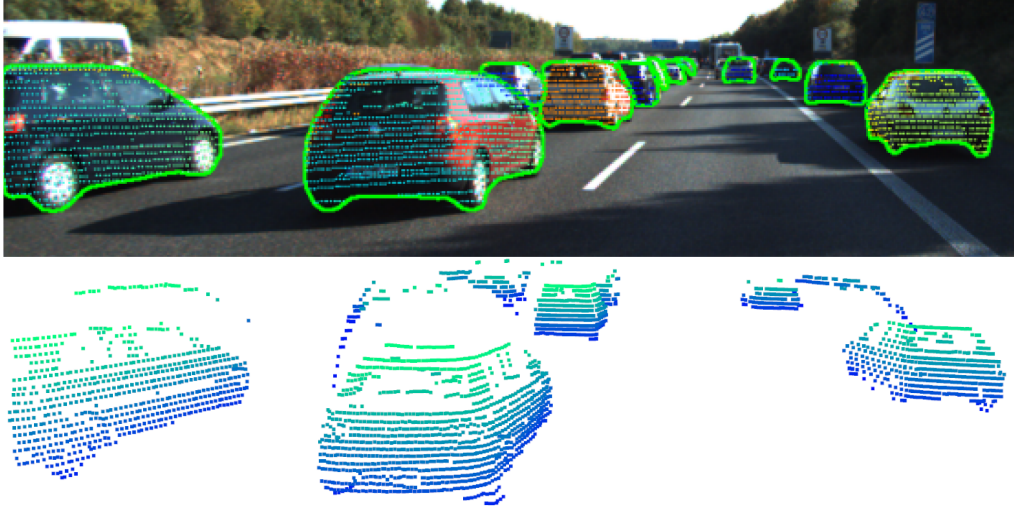


and subsequently, in testing, apply it to the semi-canonical target domain  $\rho(\mathbf{M}_i^T)$  to obtain bounding boxes which can then be projected back into the target domain  $\mathbf{X}_i^T$ .

### Isolating the object point cloud

The first phase of SEE depicted aims to isolate the object points  $\mathbf{O}_i^S$  and  $\mathbf{O}_i^T$  in the point cloud. Where labels are available in the source domain, this can easily be accomplished for each  $k$ -th object by cropping the point cloud with their bounding box  $y_k$ . In the target domain, we utilize instance segmentation to isolate object instances  $\mathbf{O}_i^T = \sigma(\mathbf{x}_i^T)$ .

**Instance segmentation.** Using image instance segmentation, we can obtain a mask that outlines the object shape within the detected bounding box. We first project the point cloud into the image plane using the homogenous transformation matrix  $T_{\text{camera} \leftarrow \text{lidar}}$  and the camera matrix. Thereafter, in the image plane, we isolate the points within each mask.

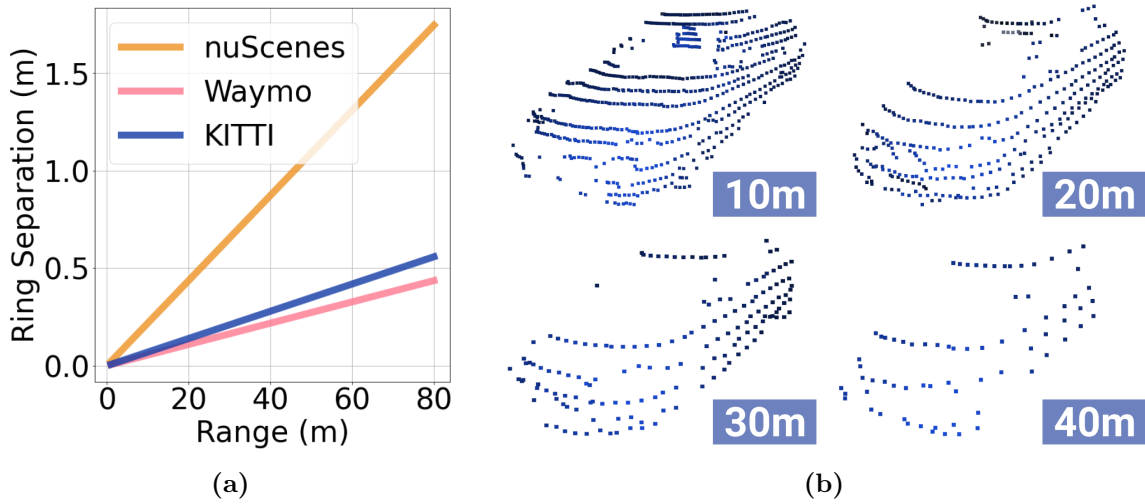


**Figure 3.13** – Background points due to viewpoint misalignment in KITTI dataset.

Instance segmentation masks allow for tight bounds on the object, giving better isolation of object points over bounding boxes; but this is not without its issues [Berrio et al., 2021]. Viewpoint misalignment of camera and lidar, loosely fitted

instance masks, as well as calibration errors, discussed in Section 3.2, can lead to the inclusion of points within the mask that do not belong to the object, shown in Fig. 3.13. Additionally, due to the lack of depth in images, instance segmentation may occasionally pick up cars or people in posters and billboards [Gomez-Donoso et al., 2020], which is not of interest to 3D object detection for autonomous driving. If these are transformed to the canonical domain and passed to the 3D detector for training, the model would learn the idiosyncrasies and failures specific to a particular instance segmentation model used for generating training data. To bypass this, we opt to use instance segmentation only in the target domain which in turn gives our framework flexibility in our choice of instance segmentation models. By using instance segmentation only in the target domain, we ensure that the source domain model is trained with clean data.

Our results show that for a well-calibrated lidar and camera pair with minimal view-point misalignment, a simple clustering strategy is sufficient to isolate object points  $\mathbf{O}_i^T$  from the set of points in the mask  $\mathbf{S}_i^T = \{s_k^T |_{k=1}^{N_k}\}$ , where  $s_k^T$  is the set of mask points for the  $k$ -th object.



**Figure 3.14** – (a) Lidar ring separation with increasing distance. At 40m, the distance between the vertical rings of the nuScenes lidar is nearly 1m; (b) Velodyne HDL-64E (KITTI) ring separation at 10m, 20m, 30m and 40m. A KITTI car at 40m has a maximum of 5-6 rings on the car.

**VRES-based Clustering.** To find the points of the  $k$ -th object  $\mathbf{o}_k^T$  from the point

set  $s_k^T$  of the  $i$ -th point cloud, we employ a DB-Scan [Ester et al., 1996] clustering strategy using an estimated Vertical Resolution (VRES) of the lidar. Beam divergence is a characteristic of lidars where the separation between points widens with distance. For ring-based lidars, such as KITTI, this widening is more prevalent in the vertical separation between points of different rings as illustrated in Fig. 3.14. Therefore, a good clustering distance  $\epsilon$  needs to be adaptive to the widening of vertical point distances. We can estimate a VRES  $\phi_v$  with  $\phi_v = \frac{\theta_F}{N_R}$ , where  $\theta_F$  is the vertical FOV of the lidar and  $N_R$  is the number of rings. For a non-ring based lidar,  $N_R$  was approximated based on the observed resolution. The clustering distance  $\epsilon$  can be set by  $\epsilon = \alpha d_v$ , where  $\alpha$  is a scaling factor for the vertical point distance  $d_v = d_o \tan(\phi_v)$  and  $d_o = \|p_k^T\|_2$  is the centroid distance of the mask points. We select the largest cluster of  $p_k^T$  as the set of our object points  $o_k^T$ .

### Surface completion

To recover the geometry of the isolated object points  $o_k^S$  and  $o_k^T$ , we use the Ball Pivoting Algorithm (BPA) [Bernardini et al., 1999] to interpolate a triangle mesh. The core idea of BPA is simple. A set of spheres are rolled around the point set. If any ball touches three points and doesn't fall through, it forms a triangle. We estimate surface normals for the object point set by orienting them towards the lidar origin. BPA uses these surface normals to ensure that triangles are only generated on the surface of the object; this was effective in recovering an accurate geometry of the object. The added benefit of using an SC method is that we address the issue of partial occlusion, where objects are split into two components, most commonly in driving datasets by poles, signs or trees. With BPA we can connect the two disjointed components into a single object. BPA can be easily swapped out with other SC methods such as Alpha Shapes [Edelsbrunner et al., 1983] or Poisson Surface Reconstruction [Kazhdan et al., 2006], which do not require pose normalization [Yuan et al., 2018, Yu et al., 2021]. We explore this further in Section 3.3.3.

### Point sampling

Objects at closer ranges typically have more confident detections due to the high density of points; this can be observed in the Waymo leaderboard [Sun et al., 2020] when evaluating by range. In the last phase of SEE, we choose to upsample every object’s triangle mesh to increase the confidence of 3D detectors. We sample the triangle meshes using Poisson disk sampling [Yuksel, 2015], where each point is approximately equidistant to neighbouring points.

To increase the confidence of detections at a further range, we upsample the meshes to emulate the point density of objects at a closer range (5-10m). Smaller point distances are typically correlated with a higher density of points. We define an ideal vertical point distance  $d_{\text{ideal}}$ , as our desired density of points. The relationship between  $d_{\text{ideal}}$  and vertical point distance  $d_v$  can be represented as  $d_{\text{ideal}} = \frac{d_v}{\beta}$ , since  $d_v$  increases with distance. By rearranging this equation, we can solve for our upsampling factor  $\beta$ . Whilst more points are generally better, densely sampled points may accentuate mistakes in the object isolation phase. We sample each  $i$ -th point cloud object according to this strategy to obtain our semi-canonical representation  $\mathbf{M}_i^S$  and  $\mathbf{M}_i^T$ .

### Object detection in the semi-canonical domain

We train 3D object detectors on the semi-canonical domain  $\rho(\mathbf{M}_i^S)$  using labels in the source domain  $\mathbf{Y}_i^S$ . We demonstrate the framework on two SOTA detectors SECOND-IoU [Yang et al., 2021] and PV-RCNN, which required no architectural modifications. Given a point cloud in the test domain  $\mathbf{X}_i^T$  we isolate the objects  $\mathbf{O}_i^T$ ; convert to the semi-canonical domain  $\mathbf{M}_i^T$ ; apply the detection network  $\rho(\mathbf{M}_i^T)$ ; and finally, project detections back into the target domain  $\mathbf{X}_i^T$ .



### 3.3.3 Experiments

In this section, we first validate SEE on public datasets and subsequently demonstrate its efficacy on the novel Baraja Spectrum-Scan dataset.

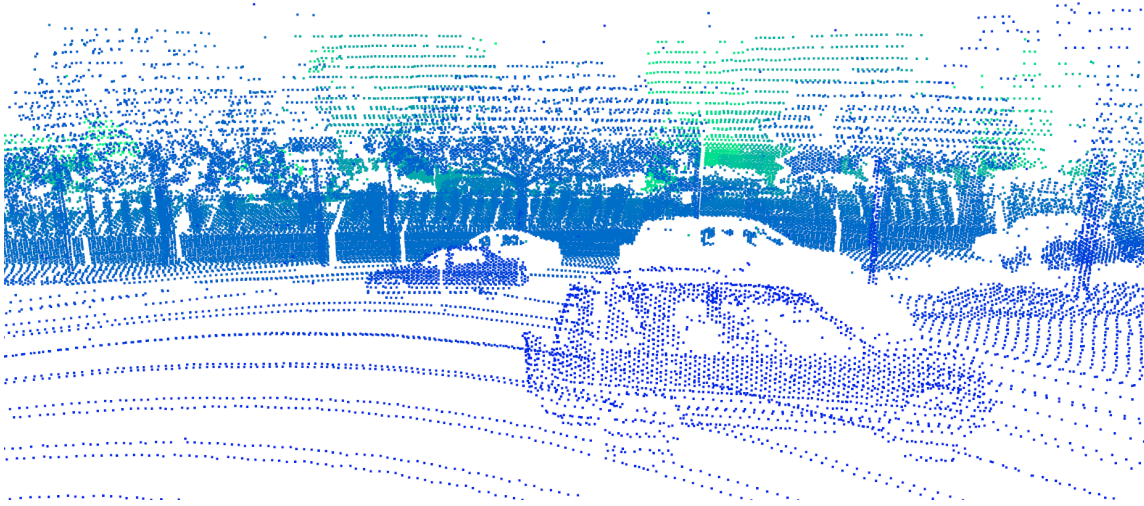
#### Datasets

We evaluate our framework on the "Car" class using three public datasets: Waymo, KITTI and nuScenes; and a manually annotated Baraja Spectrum-Scan dataset. We demonstrate the concept across multiple scenarios: (1) Differing lidar ring numbers (i.e., nuScenes  $\rightarrow$  KITTI); (2) Multiple concatenated point clouds to a single point cloud (i.e., Waymo  $\rightarrow$  KITTI); (3) Ring-based to a uniform, interleaved scan pattern (i.e., Waymo/nuScenes  $\rightarrow$  Baraja). We evaluate all scenarios on SECOND-IoU and PV-RCNN.

**KITTI.** The KITTI dataset [Geiger et al., 2012] provides 7,481 training samples and 7,518 testing samples. The entire training dataset contains 27,459 cars. The sensor suite includes 2 colour cameras and a Velodyne HDL64E lidar with 360° horizontal FOV (HFOV) and a narrow 26.8° vertical FOV (VFOV) with 64 equally spaced angular subdivisions (approximately 0.4°). We follow the popular 50/50 training/validation split [Chen et al., 2017, Yan et al., 2018], giving 3,712 training samples and 3,769 validation samples.

**nuScenes.** The nuScenes dataset [Caesar et al., 2020] provides 28,130 training, 6,019 validation and 6,008 testing samples. The nuScenes sensor suite includes 6 cameras and a Velodyne HDL32E lidar with 360° HFOV and a wide 40° VFOV with 32 equally spaced angular subdivisions (approximately 1.25°). For fair comparison across datasets, and due to limited hardware resources, we created a subset of nuScenes comprising of 4025 training samples, and 3980 validation samples. This subset contains a similar number of cars and frames as in KITTI. To overcome the surface completion challenges caused by point cloud sparsity, we truncated the point cloud range in the training process to [-24, 24]m for X and Y, and [-2,4]m for Z axis.

**Waymo.** The Waymo Open Dataset [Sun et al., 2020] contains 1,000 scenes, each with 200 frames. The dataset provides a top-mounted long-range lidar and 4 short-range lidars, with the point cloud truncated to a maximum of 75m for the top-lidar and 20m for the short-range lidars. The 360° HFOV top-lidar has a narrow VFOV of 20° with 64 non-uniformly spaced subdivisions where points are denser at the horizon. We concatenate all 5 lidar scans into a single point cloud. Similar to nuScenes, we create a subset of the Waymo dataset. Selecting one frame from each scene leads to a subset of 1,000 training samples with a similar number of cars (30,405 cars) to the KITTI dataset. We trained the models on the "Vehicle" class with a point cloud range of  $[-75.2, 75.2]$ m for X and Y axis, and  $[-2, 4]$ m for Z axis.



**Figure 3.15** – The Baraja Spectrum-Scan dataset scan pattern is configured to foveate points at the horizon. The foveation pattern is visible on the SUV where the bottom half has a ring-like appearance whilst the upper half we describe as a uniform, interleaved scan pattern.

**Baraja Spectrum-Scan.** Our manually labelled dataset uses Baraja Spectrum-Scan, a novel, high-resolution lidar that can dynamically increase point cloud resolution around key objects [Pulikkaseril and Langdale-Smith, 2020]. Baraja Spectrum-Scan has 120° HFOV and 30 ° VFOV. It is able to adjust its vertical (from 0.025°-0.2°) and horizontal (from 0.05°-0.2°) angular resolution for user-defined, non-uniformly spaced subdivisions. This allows it to have a scan pattern that can achieve up to 1000 subdivisions per beam channel. Our collected dataset uses 128 non-uniform

subdivisions per channel, with the points foveated to focus on the horizon as shown in Fig. 3.15. In our particular configuration, the foveated region is what we describe as a uniform, interleaved distribution of points, where the vertical and horizontal angular resolution is approximately equal. The top and bottom of the vertical FOV, however, is configured similarly to the beams of KITTI, Waymo, and nuScenes. Our camera, Intel RealSense D435i, was mounted directly below the lidar. Using the labelling tool, SUSTechPOINTS [Li et al., 2020], we followed the KITTI annotation scheme and labelled cars only if they are visible in the image FOV. In the image extremities, we only label cars if more than 50% of the car is visible in the image. We adopt the same dataset limitations as Waymo in truncating the point cloud to a maximum of 75m, and we include side mirrors within the annotated bounding box. The dataset was collected in San Francisco in dry weather and contains 100 frames, with a total of 928 cars.

### SEE results on public datasets

**Comparisons.** SEE was compared in the UDA scenario where no prior information about the target domain is known. We adopt a similar convention to [Yang et al., 2021] and compare SEE with (1) Source-only, where no DA strategies are used; (2) ST3D [Yang et al., 2021], the SoTA UDA method on 3D object detection using self-training. Trained ST3D models were obtained from their open-sourced codebase; (3) SEE-Ideal, where we use the ground truth annotations to isolate the target domain objects; (4) Oracle, the fully supervised detector trained on the target domain. For (3), the ideal case highlights the potential of this framework with further improvements to object isolation, viewpoint alignment and image instance segmentation in the target domain.

**Evaluation Metric.** We evaluate SEE on the “Car” class in the KITTI validation dataset, similar to other UDA methods [Yang et al., 2021, Wang et al., 2020]. We follow the official KITTI evaluation metric and report the Average Precision (AP) over 40 recall positions at the 0.7 Intersection Over Union (IoU) threshold for 3D and Bird’s Eye View (BEV). We report the performance increase of models trained with

SEE, over the Source-only. We evaluate all methods only on ground truth annotations where there are 50 points or more within the 3D bounding box. This is because when there are less than 50 points, it is often insufficient to generate a significant triangle mesh [Stutz and Geiger, 2018].

**Implementation Details.** We use the Hybrid Task Cascade (HTC) instance segmentation network [Chen et al., 2019a] provided by the mmdetection codebase [Chen et al., 2019b] which was pretrained on the COCO dataset [Lin et al., 2014]. In our SEE framework, we set  $d_{\text{ideal}} = 0.05$ ,  $\alpha = 5$ , 20 BPA spheres with upper radius 1.155, and we shrink instance segmentation masks by 2%. We use the Open3D [Zhou et al., 2018] implementation of BPA. For cross-dataset training and evaluation, we adopted the settings of the ST3D [Yang et al., 2021] codebase which was based on OpenPCDet [Team, 2020]. We add an offset to the z coordinates of nuScenes (+1.8m), KITTI (+1.6m), and Baraja Spectrum-Scan (+2.2m) to shift the origin to the ground plane. For training, we use the widely adopted data augmentation strategies: random flipping, random world scaling, random world rotation, and random object scaling (proposed by [Yang et al., 2021]). All models were trained on an RTX 2080 Ti.

**Results.** From Table 3.1, we show that SEE improves the performance of both detectors by a large margin. Due to the nature of our approach, having more points leads to a better-interpolated geometry. This is observed in the Waymo  $\rightarrow$  KITTI task, where our approach outperforms ST3D with both SECOND-IoU and PV-RCNN by 2.71 and 6.07 AP<sub>3D</sub> respectively. From this, we demonstrate that SEE, an MTDA method, is able to achieve similar or higher performance than the SoTA single target DA method, ST3D. For the 64-beam KITTI/Waymo lidars, 20-50 points on an object often outlines 3D geometry as the points are spread across more than two rings. In nuScenes however, due to having less beams and large vertical angular resolution, 20-50 points are often in the form of 1-2 rings which reduces the point set’s dimension to a line, or a plane. Whilst each ring may be dense horizontally and have around 50 points for 2 rings on a car, a line or plane does not give any 3D information. The lack of 3D geometry in the object points poses a challenging task when nuScenes is

| Task             | Method      | SECOND-IoU                           |                 | PV-RCNN                              |                 |
|------------------|-------------|--------------------------------------|-----------------|--------------------------------------|-----------------|
|                  |             | AP <sub>BEV</sub> / AP <sub>3D</sub> | Improvement     | AP <sub>BEV</sub> / AP <sub>3D</sub> | Improvement     |
| Waymo → KITTI    | Source-only | 47.67 / 11.92                        | -               | 67.42 / 41.54                        | -               |
|                  | ST3D        | 78.93 / 62.81                        | +31.26 / +50.88 | 89.94 / 73.32                        | +22.53 / +31.78 |
|                  | SEE         | <b>88.09</b> / <b>65.52</b>          | +40.42 / +53.60 | <b>90.82</b> / <b>79.39</b>          | +23.41 / +37.85 |
|                  | Ideal-SEE   | 88.93 / 70.48                        | +41.26 / +58.56 | 93.17 / 84.49                        | +25.75 / +42.95 |
|                  | Oracle      | 86.07 / 81.73                        | -               | 92.17 / 89.32                        | -               |
| Waymo → nuScenes | Source-only | 21.38 / 07.29                        | -               | 29.59 / 15.09                        | -               |
|                  | ST3D        | <b>38.48</b> / <b>21.48</b>          | +17.10 / +14.19 | <b>38.70</b> / <b>19.10</b>          | +09.11 / +04.01 |
|                  | SEE         | 21.28 / 10.51                        | -00.11 / +03.22 | 23.84 / 14.72                        | -05.75 / -00.36 |
|                  | Ideal-SEE   | 24.27 / 12.20                        | +02.88 / +04.91 | 27.36 / 18.41                        | -02.23 / +03.32 |
|                  | Oracle      | 26.15 / 22.37                        | -               | 26.39 / 24.83                        | -               |
| nuScenes → KITTI | Source-only | 48.74 / 16.39                        | -               | 68.97 / 48.03                        | -               |
|                  | ST3D        | 76.82 / <b>58.58</b>                 | +29.62 / +47.71 | <b>88.89</b> / <b>83.86</b>          | +24.23 / +40.59 |
|                  | SEE         | <b>77.00</b> / 56.00                 | +28.26 / +39.61 | 85.53 / 72.51                        | +16.56 / +24.49 |
|                  | Ideal-SEE   | 83.80 / 66.82                        | +35.06 / +50.43 | 89.94 / 80.29                        | +20.98 / +32.26 |
|                  | Oracle      | 86.07 / 81.73                        | -               | 92.17 / 89.32                        | -               |

**Table 3.1** – Results for the different DA scenarios. We report the AP for the car class on KITTI dataset in bird’s-eye-view (AP<sub>BEV</sub>) and 3D (AP<sub>3D</sub>) at the 0.7 IoU threshold on the moderate case. “Improvement” denotes the AP increase from source-only to the various approaches in the format AP<sub>BEV</sub> / AP<sub>3D</sub>. The best results are indicated in bold (with the exception of SEE-Ideal as it uses ground truth boxes). All models are evaluated on ground truth boxes with 50 points or more. Publicly available ST3D models are evaluated with this same criterion.

the source or target domain, causing failed or poor interpolation of object geometry. The result of this is minor performance gain or even degradation in the Waymo → nuScenes task. This is also apparent in nuScenes → KITTI, where SEE achieves comparable performance to ST3D with SECOND-IoU. We additionally note that there is a substantial gap when comparing SEE to SEE-Ideal, which indicates the potential of our chosen SC method, BPA, with improvements to the object isolation phase and future instance segmentation models. We believe that the gap between SEE and SEE-Ideal can be further closed if the camera and lidar viewpoint alignment is minimised, as this reduces background points. Whilst we do not explicitly deal with the bounding box distribution gap [Wang et al., 2020], it is an inherent benefit of SEE as the surface completed objects provides the detector with better prior estimates, leading to better car size predictions. We show qualitative results in Fig. 3.16.

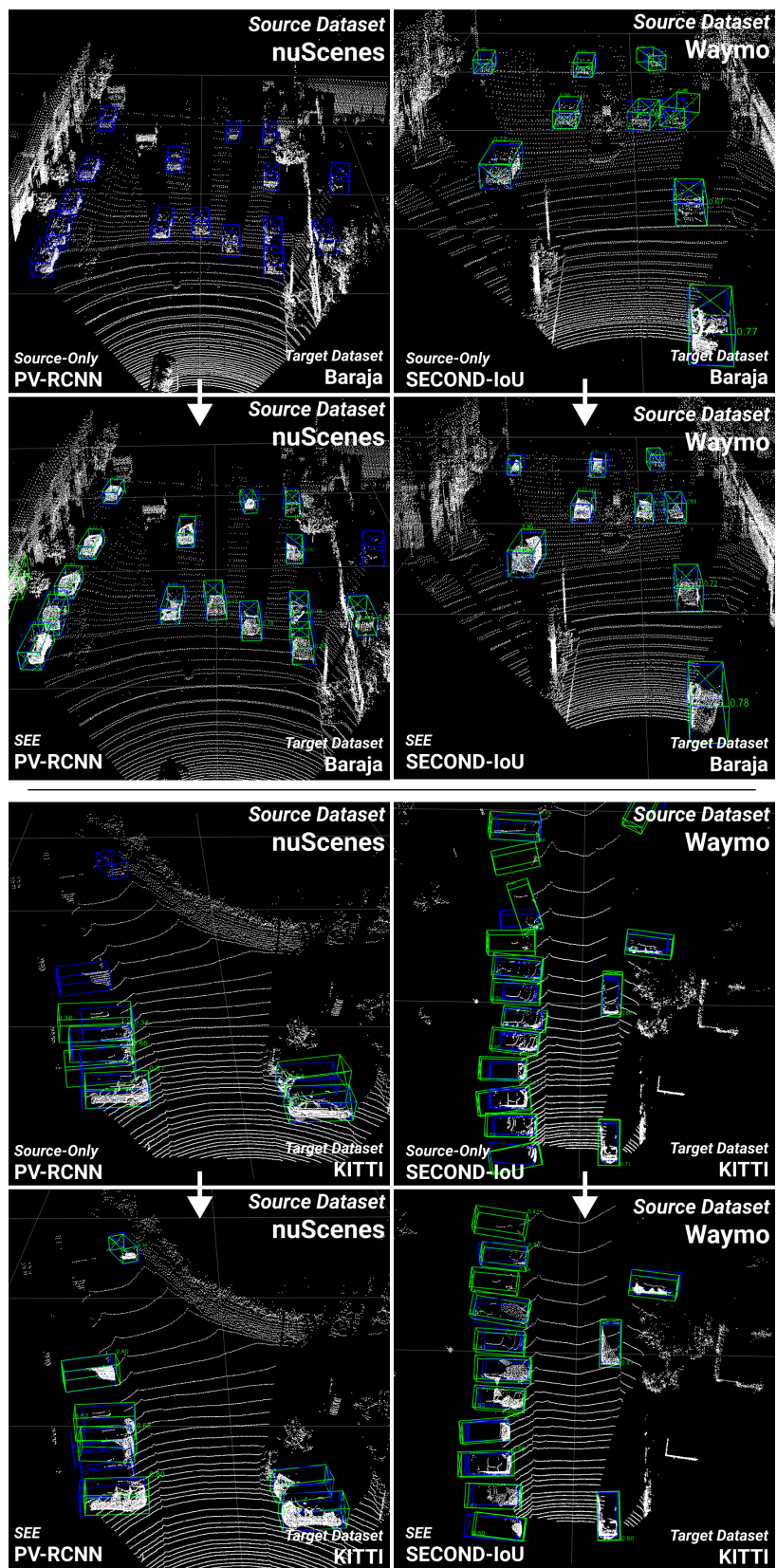


Figure 3.16 – Qualitative results for our SEE framework.

| Source   | Method      | SECOND-IoU    |               | PV-RCNN       |               |
|----------|-------------|---------------|---------------|---------------|---------------|
|          |             | 3D            | BEV           | 3D            | BEV           |
| nuScenes | Source-only | 1.02          | 5.12          | 10.85         | 13.74         |
|          | SEE         | 34.54         | 58.45         | 64.34         | 77.73         |
|          | Improvement | <b>+33.52</b> | <b>+53.33</b> | <b>+53.49</b> | <b>+63.99</b> |
| Waymo    | Source-only | 49.96         | 74.64         | 76.14         | 84.10         |
|          | SEE         | 73.79         | 84.74         | 79.13         | 87.79         |
|          | Improvement | <b>+23.84</b> | <b>+10.10</b> | <b>+2.98</b>  | <b>+3.69</b>  |

**Table 3.2** – Results for the Baraja Spectrum-Scan dataset. “Improvement” denotes AP increase from Source-only to SEE. Reported AP is at the 0.7 IoU threshold. We evaluate using the same trained models in Table 3.1.

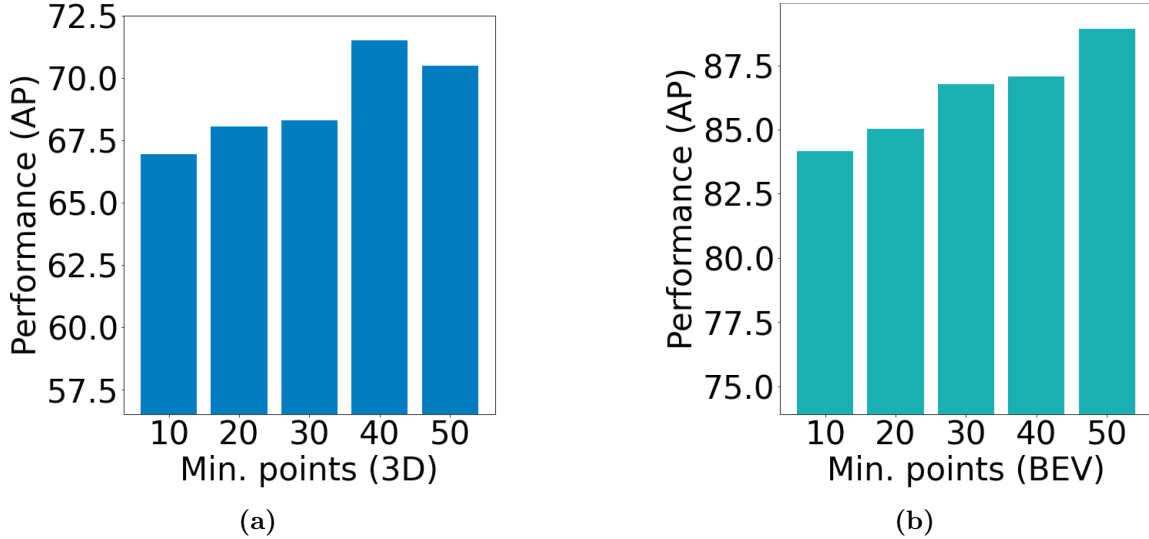
### SEE for Baraja Spectrum-Scan

In Table 3.2, we highlight SEE’s adoption of novel lidar technology by evaluating on the Baraja Spectrum-Scan dataset. We demonstrate that the same trained models in Table 3.1 can attain high performance in an unseen dataset with a distinct scan pattern (see Fig. 3.15). When comparing with nuScenes, due to its ring-like nature and sparsity of beams, the scan pattern of the lidar is vastly different. Even at a close range, a car in nuScenes bears little resemblance to any car perceived by Baraja Spectrum-Scan. In Table 3.2, this is observed in the dramatically low performance of models trained on nuScenes as the source domain, obtaining 1.02 and 10.85 in  $AP_{3D}$  for SECOND-IoU and PV-RCNN respectively. When the same models are trained with SEE, the performance is drastically improved with an increase of  $1.02 \rightarrow 34.54$  and  $10.85 \rightarrow 64.34$  in  $AP_{3D}$ . For Waymo, lidar beams are clustered in the middle of vertical angles, leading to high resolution at the horizon and sparse rings in the top and bottom of the FOV, similar to Baraja Spectrum-Scan. This scan pattern similarity leads to a higher performing “Source-only”. We observe that for the Waymo “Source-only” PV-RCNN attains a high  $AP_{3D}$  for the Baraja dataset, however the same “Source-only” model does not perform well on KITTI as shown in Table 3.1. The single SEE-trained PV-RCNN model however, can obtain high performance on both KITTI and Baraja Spectrum-Scan. In Table 3.2 we show that SEE further pushes SECOND-IoU and PV-RCNN’s performance to 73.79 and 79.13  $AP_{3D}$  respectively.

## Ablation Studies

In this section, we conduct ablation studies using the SECOND-IoU model.

**Minimum number of points.** To interpolate the underlying geometry, we need to have a sufficient amount of points to interpolate from. We investigate the performance of SEE on objects containing less than 50 points and adjust the evaluation accordingly. For example, if the minimum number of points for the SC phase is 40, we also evaluate only on ground-truth boxes that have 40 points or more. It can be seen in Fig. 3.17 that more points leads to higher quality meshes, and hence, better detection performance.



**Figure 3.17** – Minimum number of points using SEE-Ideal, evaluated on the Waymo → KITTI scenario with SECOND-IoU. Reported AP is at 0.7 IoU for (a) 3D (b) BEV.

**Point Sampling Strategy.** We compare our VRES-based sampling with two other strategies: (1) Surface-area (SA) based sampling, where we sample a certain number of points per  $m^2$  of the triangle mesh surface area; (2) Virtual lidar (VL), where we adopt the strategy of [Yi et al., 2021] mimicking the scan pattern of the target domain by comparing the spherical coordinates of the point clouds. In Table 3.3 we can see that a increasing the density of point sampling in  $d_{\text{ideal}} = 0.03$  and  $\frac{\text{pts}}{m^2} = 700$  causes a decrease in performance, as it potentially over-accentuates mistakes of the



SC method. On the flip side, fewer points lead to a weaker presence of the object, which also decreases the performance.

| Strategy                        | Config | 3D            | BEV           |
|---------------------------------|--------|---------------|---------------|
| VRES ( $d_{\text{ideal}}$ )     | 0.07   | 64.13 / 58.70 | 85.70 / 82.36 |
|                                 | 0.05   | 70.96 / 65.52 | 89.70 / 88.09 |
|                                 | 0.03   | 68.62 / 57.16 | 89.50 / 87.94 |
| SA ( $\frac{\text{pts}}{m^2}$ ) | 300    | 70.86 / 58.70 | 89.67 / 80.68 |
|                                 | 500    | 75.52 / 61.41 | 91.17 / 80.34 |
|                                 | 700    | 73.86 / 61.80 | 89.63 / 81.09 |
| VL                              | -      | 70.02 / 64.23 | 86.71 / 85.57 |

**Table 3.3** – Point sampling strategies: VRES, Surface-Area (SA), and Virtual Lidar (VL). The reported AP shows SEE-Ideal/SEE at 0.7 IoU using SECOND-IoU on the Waymo  $\rightarrow$  KITTI task. Methods are evaluated on ground truth boxes with 50 or more points.

**Surface completion.** In Table 3.4, we compare another SC method, alpha shapes [Edelsbrunner et al., 1983] and a deep learning shape completion method, PCN [Yuan et al., 2018]. We notice that alpha shapes tend to join the interior object points, leading to irregularly shaped cars, which is shown in the performance degradation in  $AP_{3D}$  when comparing with BPA. With PCN, we demonstrate the potential of our framework when using deep learning shape completion methods. In this table, we only use PCN with SEE-Ideal as we require the ground truth boxes to normalise the pose and scale of each car for completion. However, we note that PCN is not directly applicable to our approach in the non-ideal setting which we elaborate in the following discussion.

| Method                  | 3D          | BEV         |
|-------------------------|-------------|-------------|
| Ball Pivoting           | 66.81/55.78 | 82.67/76.80 |
| Alpha Shapes            | 56.31/49.07 | 68.57/65.17 |
| PCN [Yuan et al., 2018] | 83.89/ -    | 85.60/ -    |

**Table 3.4** – Comparison of BPA with Alpha Shapes and PCN [Yuan et al., 2018]. Table shows SEE-Ideal/SEE on the nuScenes  $\rightarrow$  KITTI task using SECOND-IoU for  $AP_{3D}$  at 0.7 IoU.

## Discussion

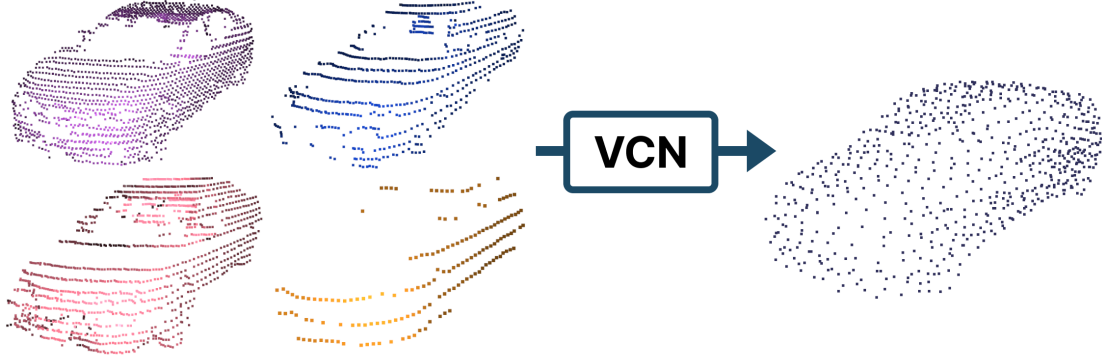
The SC phase of SEE runs at 0.33FPS for a single camera-lidar pair on an Intel(R) Core(TM) i7-6700K CPU which is not in real-time. However, as the primary intent for SEE is for offline unified detector evaluation, these processing times are acceptable. We showed in Figure 3.17 that our method performs better with more points on the object. This is also reflected in the high performance of SEE across lidars with at least 64 beams, and lower performance when using 32-beam lidars. However, we consider this a minor limitation as we argue that the future direction of lidar manufacturing is headed towards higher resolution point clouds. The results in Table 3.4 indicates the importance of the SC phase and leads us to believe that more robust SC methods can further improve the domain adaptation capabilities of SEE. We believe that there is potential in our framework to adopt deep learning shape completion methods [Yu et al., 2021, Yuan et al., 2018] in our SC phase. The difficulty in naively adopting these methods, however, is that existing approaches require the use of ground truth boxes to normalise the object pose and scale before surface completion. Incorrect normalization can cause the shape completion network to generate spurious cars in odd rotations and scales which is detrimental to the detector. In the following section we address this issue by proposing a deep learning network that can complete the surface of objects without requiring ground truth boxes for pose normalization.

## 3.4 Viewer-Centred Completion Network

### 3.4.1 Overview

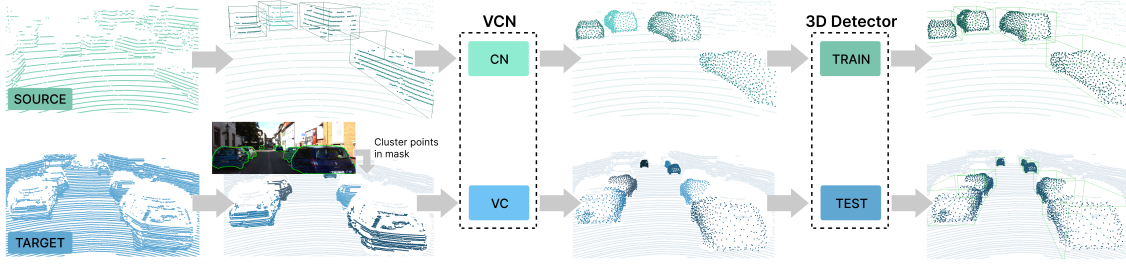
In this section, we incorporate our lidar-agnostic, viewer-centred surface completion network (VCN) with the SEE framework to improve the SC phase. We name our approach SEE-VCN. Similar to the ball pivoting surface completion in SEE, VCN extrapolates the underlying surface of each point to transform the lidar-specific to a unified representation, shown in Fig. 3.18. However, the ball pivoting algorithm used in the work is slow and leads to inconsistent car meshes. Instead, VCN is fast

(0.32ms/car) and generates consistent surface representations for cars from different lidars. Overall, SEE-VCN outperforms MTDA approach SEE, and STDA approach ST3D [Yang et al., 2021] in multiple domain adaptation settings.



**Figure 3.18** – VCN transforms objects from each lidar, into a unified surface representation.

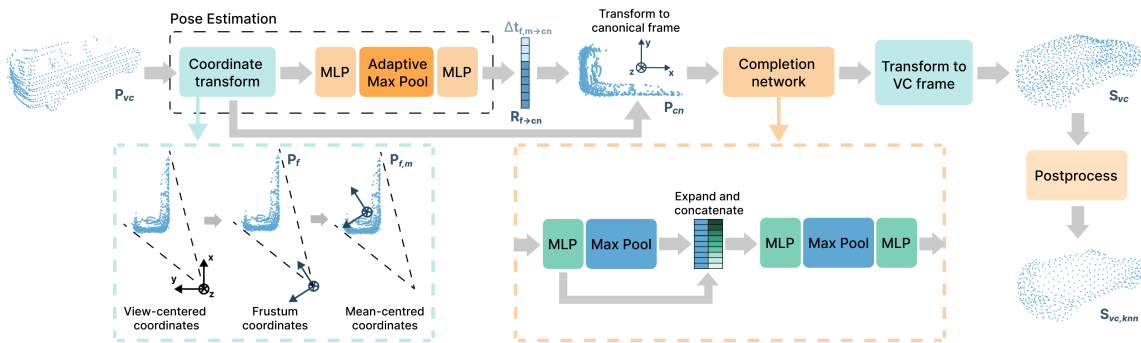
Deep learning point cloud completion approaches [Yuan et al., 2018, Yu et al., 2021, Groueix et al., 2018, Yang et al., 2018] are typically applied in object-centred coordinates, where ground truth orientation and dimensions are required to pre-canonicalize objects to a unit-bounding box with a pose that is centred and aligned to the Euclidean frame axes. Conversely, there are few existing point cloud completion approaches for objects in viewer-centred coordinates. Viewer-centred coordinates represent the object in the coordinate frame aligned to the viewing perspective of the sensor. Traditional methods [Bernardini et al., 1999, Edelsbrunner et al., 1983, Kazhdan et al., 2006], such as BPA used in SEE, are typically able to complete viewer-centred objects, though the meshes are often inadequate for sparse observations, and require heavy parameter tuning for each object. While deep learning models for object-centred completion have displayed great performance, adapting these models to the viewer-centred context is non-trivial. Our proposed VCN is a deep learning model that can estimate the pose and complete the surface of an object without pre-canonicalization. We adopt a synthetic-to-real approach, which enables VCN to accommodate viewer-centred surface completion for various lidar scan patterns.



**Figure 3.19** – SEE-VCN integrates the SEE framework with our Viewer-centred Completion Network (VCN) to transform objects of interest to a unified point cloud representation. For the source domain, we use ground truth labels to transform objects into an object-centred (canonical) coordinates before completion (VCN-CN). For the target domain, we estimate the object pose to transform the viewer-centred object to object-centred coordinates before completing the object (VCN-VC).

### 3.4.2 Method

SEE-VCN addresses the lidar scan pattern discrepancy through training and testing any 3D detector on a unified point cloud representation of objects. This is achieved through our viewer-centred surface completion network (VCN). For each object  $\mathbf{P}_{vc}$ , VCN transforms the object to a canonical frame  $\mathbf{P}_{cn}$  before completing the surface. To adapt it for the UDA setting, we adopt the MTDA framework, SEE [Tsai et al., 2022] and integrate it with VCN as shown in Fig. 3.19. In this thesis, the object’s canonical frame (aka. object-centred coordinates) is defined as the coordinate frame where the object is centred at the origin, and aligned to the positive x-axis. Unlike ShapeNet [Chang et al., 2015] models, our canonical frame does not involve normalizing the objects to a unit-bounding box.



**Figure 3.20** – Architecture of our Viewer-centered surface Completion Network (VCN).

### Viewer-centered Surface Completion Network (VCN)

**Pose Estimation.** To estimate the object pose, we follow [Qi et al., 2018] by first transforming the object points  $\mathbf{P}_{vc}$  to the frustum frame coordinates  $\mathbf{P}_f$  to constrain the search space and improve rotation-invariance. The frustum angle is computed with  $\theta_f = \tan^{-1}(\frac{\bar{\mathbf{P}}_y}{\bar{\mathbf{P}}_x})$  where  $\bar{\mathbf{P}}_x$  and  $\bar{\mathbf{P}}_y$  are the mean x and y-axis coordinates of the object. Thereafter, we transform the input point cloud to the frustum frame  $\mathbf{P}_f$  by applying rotation matrix  $\mathbf{R}_y$  with angle  $-\theta_f$  along the y-axis  $\mathbf{P}_f = \mathbf{P}_{vc}\mathbf{R}_y(-\theta_f)$ .

We centre  $\mathbf{P}_f$  on the mean of the points in the frustum frame with  $\mathbf{P}_{f,m} = \mathbf{P}_f - \bar{\mathbf{P}}_f$  to get the frustum-mean frame coordinates  $\mathbf{P}_{f,m}$  as illustrated in Fig. 3.20.

We feed the  $\mathbf{P}_{f,m}$  into our pose estimation component of VCN to get  $\mathbf{R}_{f \rightarrow cn}$ ,  $\Delta \mathbf{t}_{f,m \rightarrow cn} = \psi_{\text{pose}}(\mathbf{P}_{f,m})$ . VCN’s pose estimation,  $\psi_{\text{pose}}$ , is comprised of 2 Multi-Layer Perceptron (MLP) blocks with Leaky ReLU activation, and an adaptive max pooling in between.  $\psi_{\text{pose}}$  outputs a 9D vector which we split into a 3D translation delta  $\Delta \mathbf{t}_{f \rightarrow cn}$ , and a 6D rotation representation that we convert to a  $3 \times 3$  matrix,  $\mathbf{R}_{f \rightarrow cn}$ , following [Zhou et al., 2019] to ensure orthogonalization and rotation continuity. The point cloud in frustum frame  $\mathbf{P}_f$  can now be transformed to a canonical frame with  $\mathbf{P}_{cn} = (\mathbf{P}_{f \rightarrow cn} - \mathbf{t}_{f \rightarrow cn})\mathbf{R}_{f \rightarrow cn}$  where  $\mathbf{t}_{f \rightarrow cn} = \bar{\mathbf{P}}_f + \Delta \mathbf{t}_{f,m \rightarrow cn}$  and  $\mathbf{R}_{f \rightarrow cn}$  is the translation and rotation from frustum frame  $\mathbf{P}_f$  to the canonical frame  $\mathbf{P}_{cn}$  respectively. The pose from viewer-centred frame to canonical frame can be recovered by  $\mathbf{R}_{vc \rightarrow cn} = \mathbf{R}_{f \rightarrow cn}\mathbf{R}_y(\theta_f)$  and  $\mathbf{t}_{vc \rightarrow cn} = \mathbf{t}_{f \rightarrow cn}\mathbf{R}_y(\theta_f)$ .

We use Geodesic loss for  $\mathbf{R}_{vc \rightarrow cn}$ , shown in Eqs. (3.7) and (3.8) with estimated rotation,  $\mathbf{R}_{\text{pred}}$ , and ground truth rotation,  $\mathbf{R}_{\text{gt}}$ . We use Smooth- $l_1$  (huber) loss for  $\mathbf{t}_{vc \rightarrow cn}$ . Combined, pose loss is  $L_{\text{pose}} = \alpha L_{\mathbf{t}} + L_{\mathbf{R}}$  with weight  $\alpha$ .

$$\mathbf{R}'' = \mathbf{R}_{\text{pred}}\mathbf{R}_{\text{gt}} \quad (3.7)$$

$$L_{\mathbf{R}} = \cos^{-1}((\mathbf{R}_{00}'' + \mathbf{R}_{11}'' + \mathbf{R}_{22}'' - 1)/2) \quad (3.8)$$

**Surface Completion** For surface completion, we adopt PCN’s encoder [Yuan et al., 2018], which is a variant of PointNet [Qi et al., 2017a]. Each PointNet module consists

of an MLP module, followed by Max Pooling. PointNet uses an MLP for each point,  $p_i \in \mathbf{P}_{cn}$ , to transform  $\mathbf{p}_i$  into a feature vector  $\mathbf{f}_i$ . These feature vectors are fed into a point-wise max pooling to obtain a global feature vector  $\mathbf{g}$  that can intuitively be perceived as a summary of the information in  $\mathbf{P}_{cn}$ . For the output of the first PointNet, we concatenate the global vector  $\mathbf{g}_1$  to each point feature vector  $\mathbf{f}_i$  and feed it to a second PointNet module. The output of the second PointNet is our final global vector  $\mathbf{g}_2$ , which we feed into another MLP to reshape the global vector into the final object points  $\mathbf{S}_{cn}$  of shape  $N \times 3$ .

The output of the encoder network is commonly identified as “coarse” or “key” points in shape completion literature [Yuan et al., 2018, Yu et al., 2021, Qi et al., 2017a].  $\mathbf{g}_2$  is commonly fed into a decoder network to generate a more fine-grained object shape. For our use case, the coarse form is sufficient as we aim to generate a general shape outline of cars with a unified representation across scan patterns.

To obtain our completed car in viewer-centred coordinates, we first transform the object back to the frustum coordinates, then undo the frustum rotation as shown in Eq. (3.9).

$$\mathbf{S}_{vc} = (\mathbf{S}_{cn}\mathbf{R}_{cn \rightarrow f} + \mathbf{t}_{cn \rightarrow f})\mathbf{R}_y(\theta_f) \quad (3.9)$$

In synthetic point cloud completion approaches, objects are typically pre-canonicalized into a unit-bounding box [Sun et al., 2021]. However, in our scenario, we do not have the knowledge of object dimensions beforehand for pre-canonicalization. Therefore, to generate cars with more accurate sizes, we add a smooth- $l_1$  loss for the object dimension error  $L_{\text{dim}}$  between the ground truth and predicted bounding box. When an object is in the canonical frame, it is trivial to estimate a 3D bounding box by taking the maximum and minimum bounds of the object points in each axis.

To generate car-like surfaces, we follow existing works [Yuan et al., 2018, Yu et al., 2021] to calculate the distance between each generated surface point  $s \in \mathbf{S}_{vc}$ , and the ground-truth surface point  $g \in \mathbf{G}_{vc}$  using Chamfer Distance (CD) as our loss  $L_{\text{complete}}$ , shown in Eq. (3.10).

$$\begin{aligned} \text{CD}(\mathbf{S}_{vc}, \mathbf{G}_{vc}) = & \frac{1}{|\mathbf{S}_{vc}|} \sum_{s \in \mathbf{S}_{vc}} \min_{g \in \mathbf{G}_{vc}} \|s - g\|_2 + \\ & \frac{1}{|\mathbf{G}_{vc}|} \sum_{g \in \mathbf{G}_{vc}} \min_{s \in \mathbf{S}_{vc}} \|g - s\|_2 \end{aligned} \quad (3.10)$$

**Post-Processing** Viewer-centred coordinates encodes the pose of the object. For certain partially observed objects, the generated 3D surface by VCN is good, but the pose is misaligned. This causes confusion for the 3D detector. For example, a common distribution of points on a car in KITTI [Geiger et al., 2012] is where there are points on the rear, but none on the side, top or front. Given this set of points, it is difficult to determine a ground truth orientation; there could be multiple possible orientations. Dimensions are also challenging to estimate, since the rear of a sedan can appear similar to the rear of a limousine. We observe, however, that the surfaces generated by VCN are most accurate near the object’s points  $\mathbf{P}_{vc}$ . Therefore, for each car, we retain the nearest  $N$  generated surface points  $s_i \in \mathbf{S}_{vc,knn}$  to each input object point  $p_i \in \mathbf{P}_{vc}$ . To improve the quality of  $\mathbf{S}_{vc,knn}$ , we add another CD loss  $L_{knn}$  to minimise the distance between  $\mathbf{S}_{vc,knn}$  and  $\mathbf{G}_{vc,knn}$ . Lastly, we jointly optimise our network over the overall loss function shown in Eq. (3.11).

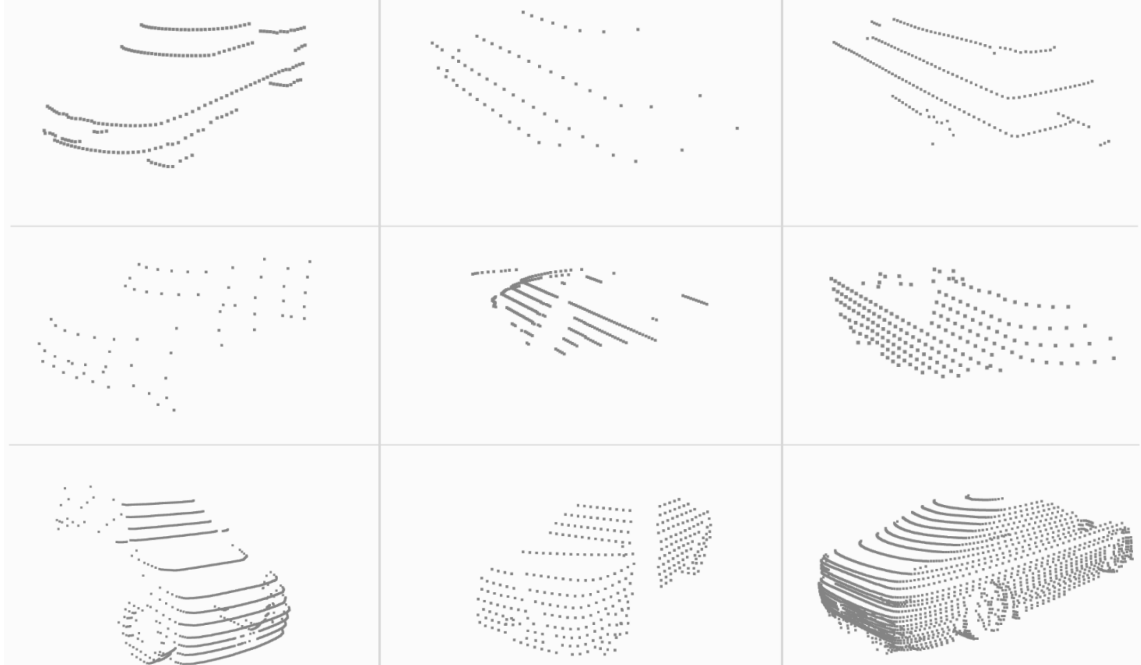
$$L = L_{\text{complete}} + L_{\text{knn}} + L_{\text{dims}} + L_{\text{pose}} \quad (3.11)$$

## Training Data

VCN was trained in a supervised manner, where partial  $\mathbf{P}_{vc}$  and complete  $\mathbf{G}_{vc}$  synthetic cars were prepared with ShapeNet [Chang et al., 2015] car models. To obtain the complete ground truth surface points  $\mathbf{G}_{vc}$  for each car model, we applied raycasting from multiple views. The surface points from each viewpoint were concatenated together. 16384 points were thereafter sampled using farthest point sampling. To generate realistically positioned and occluded partial cars  $\mathbf{P}_{vc}$ , we used “Vehicle” labels from the Waymo dataset to position and size our car models in a raycasting scene. Using the location of “signs” from the Waymo labels, we placed poles of various heights. Once the scene was set up, we raycasted from the location of the lidar,

to obtain partial cars in viewer-centred coordinates.

To ensure that VCN is not biased to a particular scan pattern, we simulate various sampling patterns for training. In the dataset generation, we densely raycast each partial car. This enables us to selectively downsample the partial cars to simulate different sampling patterns. We first convert each point into spherical coordinates  $(r, \theta, \phi)$ , and discretize vertical angle  $\phi$  into  $N$  bins, each representing a lidar ring. For training, we randomly select every  $N^{\text{th}}$  ring, and every  $k^{\text{th}}$  point within the ring for each car as the input partial car. In Fig. 3.21 we show some examples from our simulated dataset.



**Figure 3.21** – Simulated training dataset.

### Augmented Point Clouds for 3D Object Detection

For training and testing, we use two configurations of VCN: (1) VCN-CN is our object-centred configuration where we use ground truth labels to first canonicalize the car, before completing the surface. Pose estimation is not necessary for this setting. (2) VCN-VC is our viewer-centred configuration where the pose and surface of the car



are both estimated. For training, we isolate the cars with the provided ground truth bounding boxes, and use VCN-CN to complete cars in the source domain. This augmented point cloud dataset is used for training the 3D detector. For testing, we follow [Tsai et al., 2022] in isolating point cloud car instances  $\mathbf{P}_{vc}$  by keeping the points within an image instance segmentation mask. These points are further clustered with DB-Scan [Ester et al., 1996] before passing to VCN-VC for surface completion of each car. Each car instance  $\mathbf{P}_{vc}$  is replaced with the completed, and post-processed car  $\mathbf{S}_{vc,knn}$  to form our final augmented point cloud for inference.

### 3.4.3 Experiments

We evaluate SEE-VCN on the "Car"/"Vehicle" category across multiple datasets, summarised in Table 3.5. We assess the following scenarios: (1) Same number of beams (Waymo  $\rightarrow$  KITTI). Despite having the same number of beams, there still remains a performance degradation. This is in part due to object size [Wang et al., 2020], but also from scan pattern variations [Tsai et al., 2022]; (2) Low  $\rightarrow$  high beam (nuScenes  $\rightarrow$  KITTI). We show that we can attain good performance by upsampling objects of interest instead of the entire point cloud [Li et al., 2019, Yifan et al., 2019, Yi et al., 2021]. (3) High  $\rightarrow$  low beam. (4) Lidar with adjustable scan pattern (Waymo/nuScenes  $\rightarrow$  Baraja Spectrum-Scan<sup>TM</sup>).

| Datasets | VFOV | HFOV | Beams      | Sensors                                               | Vertical Resolution (°) |
|----------|------|------|------------|-------------------------------------------------------|-------------------------|
| KITTI    | 26.8 | 360  | 64         | 1x Velodyne HDL64E, 2x cameras                        | 0.40                    |
| nuScenes | 40   | 360  | 32         | 1x Velodyne HDL32E, 6x cameras                        | $\sim 1.25$             |
| Waymo    | 20   | 360  | 64         | 1x long-range lidar, 4x short-range lidar, 6x cameras | $\sim 0.31$             |
| Baraja   | 30   | 120  | Up to 1000 | 1x Baraja Spectrum-Scan <sup>TM</sup> , 1x camera     | 0.025 - 0.2             |

**Table 3.5** – Overview of the sensor setup across datasets used in this section.

### Viewer-Centred Surface Completion

We first evaluate the completion ability of VCN for cars from both synthetic and real datasets in this section.

**VC-ShapeNet.** Our generated viewer-centred data for VCN consists of 3394 car models, split into 3054 for training and 340 for testing. For the training set, we have 20 different partial views per car model, giving 61,080 total training samples. For the testing set, we have a total of 5100 samples (15 partial views per car), sampled with our simulated lidar.

**Lidar Data.** We form a nuScenes, KITTI and Waymo test dataset, each consisting of 5000 cars with at least 30 points.

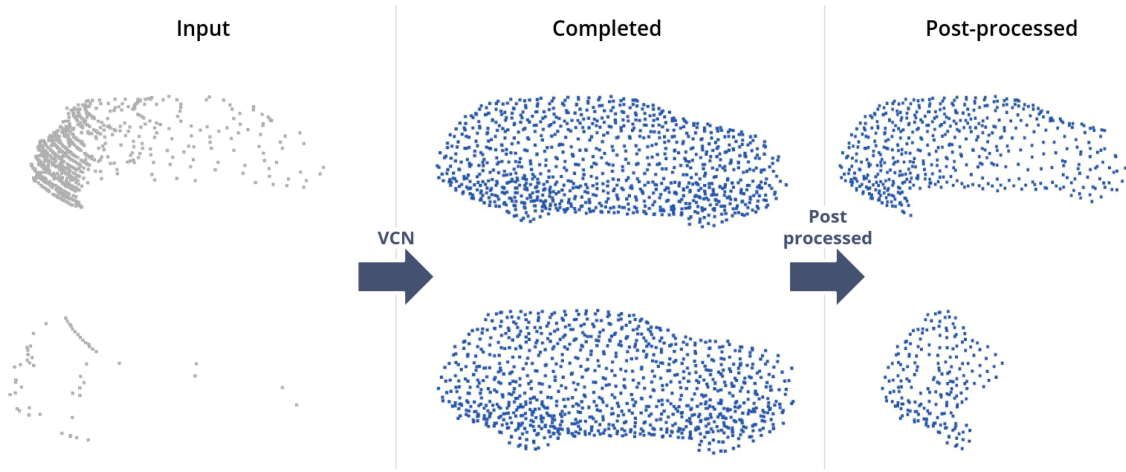
**Evaluation.** To evaluate the completed car, we use two metrics. (1) 3D and BEV IoU of the completed car with the ground truth car bounding box for both synthetic and real datasets; (2) CD with the ground truth car surfaces of the synthetic dataset. For comparison, we pick two methods, SCiW and Atlas-VC. SCiW [Gu et al., 2020] performs single-frame surface completion for viewer-centred objects using a Siamese training setup. As they did not release their code, we therefore compare with our implemented adaptation of it. We adapted SCiW to train on VC-ShapeNet instead of real data for fair comparison. Atlas-VC is based on AtlasNet [Groueix et al., 2018]. We use VCN’s pose estimation to estimate pose, followed by AtlasNet’s patch-based approach to complete the surface.

| Model    | Config | VC-ShapeNet  |                           |             |              | KITTI              |             |              |  | Waymo                     |             |              |  | nuScenes           |             |              |  |
|----------|--------|--------------|---------------------------|-------------|--------------|--------------------|-------------|--------------|--|---------------------------|-------------|--------------|--|--------------------|-------------|--------------|--|
|          |        | CD           | BEV / 3D                  | R (°)       | t (m)        | BEV / 3D           | R (°)       | t (m)        |  | BEV / 3D                  | R (°)       | t (m)        |  | BEV / 3D           | R (°)       | t (m)        |  |
| PCN      | -      | 7296.31      | 0.02 / 0.01               | -           | -            | 0.02 / 0.01        | -           | -            |  | 0.02 / 0.01               | -           | -            |  | 0.02 / 0.01        | -           | -            |  |
| VCN      | -      | 7575.57      | 0.04 / 0.03               | -           | -            | 0.04 / 0.03        | -           | -            |  | 0.05 / 0.01               | -           | -            |  | 0.06 / 0.05        | -           | -            |  |
| VCN      | c      | 49.45        | 86.9 / 79.8               | -           | 0.083        | <b>81.6</b> / 72.5 | -           | 0.101        |  | 80.1 / 68.1               | -           | 0.146        |  | 80.5 / 68.9        | -           | 0.107        |  |
| VCN      | c,r    | 47.24        | 87.3 / 80.1               | <b>1.73</b> | 0.091        | 81.1 / 72.9        | 2.96        | 0.108        |  | 80.4 / 68.9               | <b>3.36</b> | 0.155        |  | <b>80.5</b> / 69.5 | <b>1.85</b> | 0.113        |  |
| VCN      | c,r,l  | <b>39.12</b> | <b>88.1</b> / <b>81.5</b> | 1.77        | <b>0.080</b> | 80.6 / <b>74.3</b> | 3.30        | <b>0.099</b> |  | <b>80.9</b> / <b>71.2</b> | 3.57        | <b>0.149</b> |  | 80.3 / <b>72.0</b> | 2.06        | <b>0.102</b> |  |
| Atlas-VC | c,r,l  | 56.17        | 83.4 / 76.1               | 2.48        | 0.094        | 78.4 / 69.8        | 2.57        | 0.106        |  | 76.6 / 65.3               | 4.60        | 0.164        |  | 77.5 / 68.1        | 2.82        | 0.111        |  |
| SCiW     | c,r,l  | 65.03        | 81.2 / 73.1               | 1.95        | 0.113        | 74.8 / 66.1        | <b>2.31</b> | 0.125        |  | 75.3 / 63.2               | 3.37        | 0.177        |  | 75.8 / 63.9        | 2.44        | 0.134        |  |

**Table 3.6** – Evaluation of viewer-centred surface completion of cars with at least 30 points. c: centre estimation, r: rotation estimation, l: lidar simulation used. R is median rotation error (°) and t is mean translation error (m). BEV/3D refers to the IoU of the completed car’s box compared with the ground truth box size. CD is evaluated with the L2 norm and scaled by 1000 (following [Yu et al., 2021]).

**Results.** Table 3.6 shows that a direct implementation of object-centred networks such as PCN [Yuan et al., 2018] fails to generate meaningful complete shapes. We show that VCN without pose estimation also suffers similarly. Simply estimating the centre (c) allows VCN to begin generating meaningful surfaces and more accurate

car dimensions. By adding a rotation estimation  $(c,r)$ , we can further decrease the CD and increase IoU. When we simulate the lidar scan patterns  $(c,r,l)$ , we further boost each real lidar by 2-3% in 3D IoU box estimation, as well as decrease CD. VCN outperforms Atlas-VC and SCiW in both CD and IoU metrics. We also evaluate the median rotation and mean translation error and show that for object pose estimation, VCN can obtain a median rotation error of  $2-3^\circ$  and a translation error of 0.10-0.14m. Fig. 3.22 illustrates the completion and post-processing for each object in the scene.



**Figure 3.22** – Illustration of VCN’s completion and post-processing for a single object

### Unsupervised Domain Adaptation

In this section, we evaluate SEE-VCN for UDA on two 3D detectors, SECOND-IoU [Yan et al., 2018] and PV-RCNN [Shi et al., 2020]. For training of Waymo and nuScenes, we subsample the data to use 5267 frames for Waymo and 4025 frames for nuScenes.

**Baraja Spectrum-Scan.** We build upon the dataset collected by [Tsai et al., 2022] to increase the number of frames and scan pattern variations. The dataset has 408 frames, with 3 different types of scan configurations: (1) 64-beam, dense at the centre of the vertical FOV and sparse at the extremities; (2) 128-beam uniformly distributed across the vertical FOV; (3) Horizon-tracking (96-beam), where the scan pattern is adjusted whilst driving to have higher density at the horizon.

**Evaluation.** We evaluate results on Average Precision (AP) at the 0.7 threshold for 3D and BEV IoU. For KITTI, we report the moderate AP. We compare SEE-VCN with (1) Source only, where no DA is used; (2) ST3D [Yang et al., 2021], the SOTA DA method for single-frame UDA; (3) SEE [Tsai et al., 2022], the only existing MTDA method; (4) Oracle, the model trained on the target domain. We additionally include the ideal isolation scenario ("Ideal iso") where we use ground truth boxes to isolate the car points for input to VCN-VC. This shows the upper bound of SEE-VCN if we improve on the isolation of points for inference.

| Task             | Method              | SECOND-IoU                           |                 |  | PV-RCNN                              |                 |  |
|------------------|---------------------|--------------------------------------|-----------------|--|--------------------------------------|-----------------|--|
|                  |                     | AP <sub>BEV</sub> / AP <sub>3D</sub> | Improvement     |  | AP <sub>BEV</sub> / AP <sub>3D</sub> | Improvement     |  |
| Waymo → KITTI    | Source-only         | 71.82 / 38.33                        | -               |  | 69.06 / 39.97                        | -               |  |
|                  | ST3D                | 78.93 / 62.81                        | +07.11 / +24.48 |  | 89.94 / 73.32                        | +20.88 / +33.35 |  |
|                  | SEE                 | <b>88.09</b> / 65.52                 | +16.27 / +27.19 |  | <b>90.82</b> / <b>79.39</b>          | +21.76 / +39.42 |  |
|                  | SEE-VCN             | 87.19 / <b>69.27</b>                 | +15.37 / +30.94 |  | 89.00 / 74.68                        | +19.94 / +34.71 |  |
|                  | SEE (Ideal iso)     | 88.93 / 70.48                        | +17.11 / +32.15 |  | 93.17 / 84.49                        | +24.11 / +44.52 |  |
|                  | SEE-VCN (Ideal iso) | 93.25 / 79.02                        | +21.43 / +40.69 |  | 94.08 / 85.84                        | +25.02 / +45.87 |  |
| Waymo → nuScenes | Oracle              | 86.07 / 81.73                        | -               |  | 92.17 / 89.32                        | -               |  |
|                  | Source-only         | 81.54 / 57.99                        | -               |  | 80.23 / 65.26                        | -               |  |
|                  | ST3D                | <b>85.85</b> / <b>66.71</b>          | +04.31 / +08.72 |  | <b>85.59</b> / <b>71.93</b>          | +05.36 / +06.67 |  |
|                  | SEE                 | 74.37 / 44.93                        | -07.17 / -13.06 |  | 75.63 / 55.05                        | -04.60 / -10.21 |  |
|                  | SEE-VCN             | 69.03 / 39.28                        | -12.51 / -18.71 |  | 76.51 / 50.98                        | -03.72 / -14.28 |  |
|                  | SEE (Ideal iso)     | 82.47 / 53.80                        | 00.93 / -04.19  |  | 85.60 / 67.38                        | 05.37 / 02.12   |  |
| nuScenes → KITTI | SEE-VCN (Ideal iso) | 82.57 / 56.30                        | +01.03 / -01.69 |  | 87.33 / 52.45                        | +07.10 / -12.81 |  |
|                  | Oracle              | 86.05 / 71.92                        | -               |  | 85.03 / 70.94                        | -               |  |
|                  | Source-only         | 47.20 / 10.87                        | -               |  | 64.66 / 43.27                        | -               |  |
|                  | ST3D                | 76.82 / 58.58                        | +29.62 / +47.71 |  | <b>88.89</b> / <b>83.86</b>          | +24.23 / +40.59 |  |
|                  | SEE                 | 77.00 / 56.00                        | +29.80 / +45.13 |  | 85.53 / 72.51                        | +20.87 / +29.24 |  |
|                  | SEE-VCN             | <b>85.23</b> / <b>71.02</b>          | +38.03 / +60.15 |  | 86.88 / 76.01                        | +22.22 / +32.74 |  |
|                  | SEE (Ideal iso)     | 83.80 / 66.82                        | +36.60 / +55.95 |  | 89.94 / 80.29                        | +25.28 / +37.02 |  |
|                  | SEE-VCN (Ideal iso) | 94.74 / 83.54                        | +47.54 / +72.67 |  | 92.78 / 85.92                        | +28.12 / +42.65 |  |
|                  | Oracle              | 86.07 / 81.73                        | -               |  | 92.17 / 89.32                        | -               |  |

**Table 3.7** – Results for the different DA scenarios, reported with an IoU threshold of 0.7. "Improvement" denotes the AP increase from source-only to the various approaches in the format AP<sub>BEV</sub> / AP<sub>3D</sub>. "Ideal iso" refers to the scenario where car points are ideally isolated in testing. All models are evaluated on ground truth boxes with 50 points or more.

**Results.** Table 3.7 shows that SEE-VCN outperforms ST3D and SEE in Waymo → KITTI and nuScenes → KITTI for SECOND-IoU. For PV-RCNN in nuScenes → KITTI, SEE-VCN outperforms SEE, but underperforms in Waymo → KITTI. However, when assessing the ideal isolation scenario, we observe that when VCN is fed with the correctly isolated points, SEE-VCN is able to achieve a significant 10-17% increase over SEE in AP<sub>3D</sub>. This improvement highlights the potential of SEE-VCN

when the isolation of points is improved. In the Waymo  $\rightarrow$  nuscenesc scenario, SEE-VCN suffers due to the low resolution in nuScenes dataset. Due to the low resolution, any noise would exacerbate the errors of VCN. In the future, we expect this to be less of an issue as lidar manufacturing is headed towards higher point cloud resolution. Overall, SEE-VCN performs comparatively to ST3D, with the advantage of being an MTDA approach. We demonstrate this on the Baraja dataset without re-training in Table 3.8, where SEE-VCN obtains higher performance than source-only and SEE in multiple DA settings. If we can obtain perfect isolation of points (ideal iso), SEE-VCN can further increase the performance by around 12-25% AP, which is 18-33% better than SEE (ideal iso). In Fig. 3.23 we show qualitative results for our SEE-VCN framework.

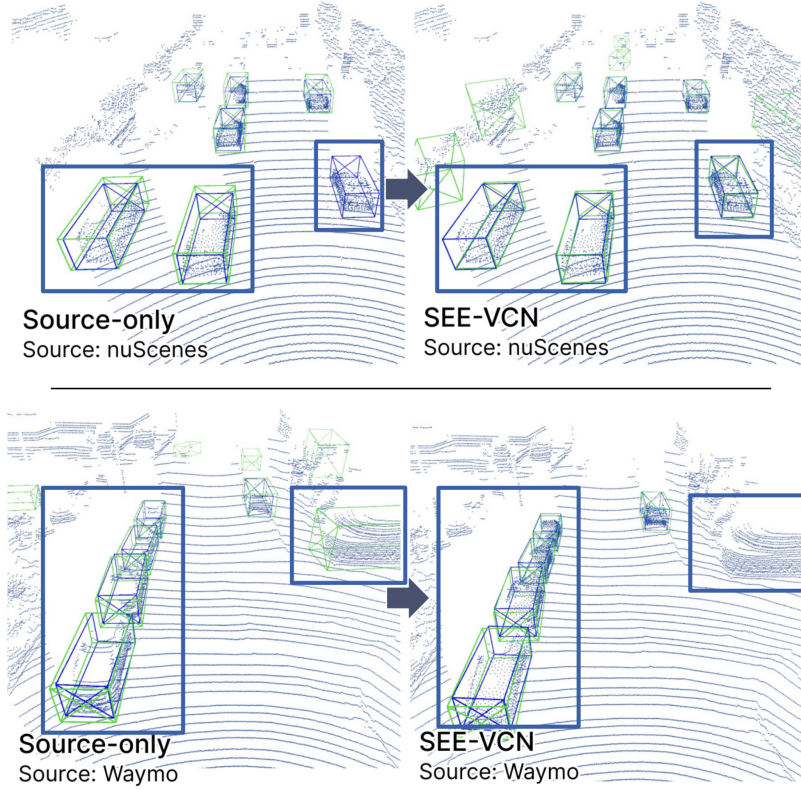
| Source   | Method              | SECOND-IoU                           | PV-RCNN                              |
|----------|---------------------|--------------------------------------|--------------------------------------|
|          |                     | AP <sub>BEV</sub> / AP <sub>3D</sub> | AP <sub>BEV</sub> / AP <sub>3D</sub> |
| Waymo    | Source-only         | 83.52 / 71.98                        | 83.27 / 76.76                        |
|          | SEE                 | <b>87.39 / 77.10</b>                 | 82.77 / 74.60                        |
|          | SEE-VCN             | 86.10 / 75.12                        | <b>88.34 / 80.50</b>                 |
|          | SEE (Ideal iso)     | 87.09 / 70.99                        | 85.52 / 79.75                        |
|          | SEE-VCN (Ideal iso) | 94.27 / 88.93                        | 93.66 / 90.53                        |
| nuScenes | Source-only         | 16.88 / 12.00                        | 18.31 / 15.30                        |
|          | SEE                 | 52.23 / 34.63                        | 71.19 / 62.15                        |
|          | SEE-VCN             | <b>66.63 / 47.45</b>                 | <b>78.04 / 63.75</b>                 |
|          | SEE (Ideal iso)     | 52.97 / 36.73                        | 72.36 / 65.54                        |
|          | SEE-VCN (Ideal iso) | 85.59 / 70.86                        | 90.16 / 82.14                        |

**Table 3.8** – We demonstrate that SEE-VCN on the Baraja Spectrum-Scan<sup>TM</sup> dataset with an IoU threshold of 0.7.

## Ablation

**VCN at varying number of points.** Table 3.9 shows that when there is above 200 points, VCN’s completion performs well across datasets. When there is less than 30 points however, the performances degrades significantly.

**Post-processing.** In Table 3.10, we show that directly using VCN causes more sensitivity to the errors in surface completion, leading to lower detector performance.

**Figure 3.23** – SEE-VCN qualitative results

|          | IoU <sub>BEV</sub> / IoU <sub>3D</sub> |             |             |             |
|----------|----------------------------------------|-------------|-------------|-------------|
|          | > 201                                  | 81 - 200    | 31 - 80     | 5 - 30      |
| KITTI    | 84.0 / 78.5                            | 79.8 / 73.9 | 76.1 / 67.9 | 62.8 / 49.7 |
| nuScenes | 81.3 / 75.6                            | 81.2 / 73.9 | 78.7 / 67.8 | 65.2 / 50.3 |
| Waymo    | 85.5 / 77.8                            | 80.6 / 70.4 | 75.9 / 64.0 | 58.8 / 46.1 |

**Table 3.9** – IoU of VCN for cars with varying number of points on lidar datasets.

When we retain the points nearest to the input points (KNN), we are able to increase the performance by 6.34% in 3D IoU. In certain scenarios, car points can be far apart, due to occlusion or noise. Therefore, to reduce the influence of noise, we further cluster  $\mathbf{S}_{vc,knn}$  to retain only the generated surface points that are close to each other (KNN+DB).

**Minimum number of points.** We provide results for SEE-VCN for evaluation at various number of minimum points in Table 3.11 for comparison with existing and

|                    | $AP_{BEV} / AP_{3D}$ |
|--------------------|----------------------|
| KNN + DB-Scan      | 87.19 / 69.27        |
| KNN                | 85.54 / 68.81        |
| No post-processing | 81.16 / 62.47        |

**Table 3.10** – IoU results for our post-processing step using SECOND-IoU on the Waymo  $\rightarrow$  KITTI setting.

future methods.

|          | Waymo $\rightarrow$ KITTI |               | nuScenes $\rightarrow$ KITTI |               |
|----------|---------------------------|---------------|------------------------------|---------------|
| Min pts. | SECOND-IoU                | PVRCNN        | SECOND-IoU                   | PV-RCNN       |
| 0        | 73.35 / 56.31             | 75.22 / 61.38 | 72.79 / 58.47                | 76.98 / 64.01 |
| 30       | 82.96 / 63.40             | 84.77 / 69.13 | 81.92 / 65.90                | 84.33 / 71.61 |
| 50       | 87.19 / 69.27             | 89.00 / 74.68 | 85.23 / 71.02                | 86.88 / 76.01 |

**Table 3.11** – Evaluation of SEE-VCN on ground truth boxes with 0, 30 and 50 points or more.

## Discussion

The run time of the surface completion of SEE-VCN is 0.32ms (3125 cars/s) for VCN on a 2080Ti, and 20.1ms (50 cars/s) for post-processing on an Intel Core™ i7-6700K CPU. We observe that there is around 10% difference in 3D IoU between SEE-VCN and SEE-VCN (Ideal iso), indicating that incorrectly isolated object points can negatively affect the completions of VCN. This could be due to misaligned calibration [Tsai et al., 2021], occlusions or incorrect/missed image detections. For instance, the image detector would occasionally include a trailer as part of a car, leading to ill-formed completed cars. In other cases, bushes, or poles may be very close to the car, causing it to be clustered as one object. Regardless, overcoming these challenges in object isolation for SEE-VCN can lead to better performance as shown in the ideal isolation upper bound.

## 3.5 Conclusion

This chapter proposed a domain-invariant approach to addressing the lidar scan pattern domain gap. We demonstrate that the scan pattern discrepancy is a large component of the domain gap and that unifying the representation of objects allows for more accurate detections. Additionally, we address key challenges in the fusion of camera-lidar information to enable optimal use of the camera as a bridge between various lidar scan patterns.

In this chapter, we first presented a novel and robust approach to the calibration task that is heavily focused on optimising the data selection process. We provide transparency in our results by projecting the entire point cloud and evaluating our estimated calibration parameters on their ability to fit the entire scene instead of their ability to fit a single pose well. Our method achieves an average reprojection error of 1-1.2cm, with a standard deviation of 0.4-0.5cm, and takes 90s for the whole pipeline. This approach carries with it the added benefit of simplifying the calibration process for practitioners of any calibration expertise, who only needs to focus on providing lots of varied data to the algorithm. Our process has been demonstrated to robustly estimate calibration parameters for different lidar technologies and lidar-camera placements, which further demonstrates that the method is not reliant on a specific lidar or lidar-camera placement.

We present a framework for lidar-agnostic 3D object detection, SEE, to enable any 3D detector to be lidar-agnostic without manual labelling or fine-tuning of the model. Our approach interpolates the underlying geometry and normalises the scan pattern of objects from different lidars before passing them to the detection network. We demonstrated the effectiveness of SEE on public datasets, achieving state-of-the-art results, and additionally provide quantitative results on the Baraja lidar to demonstrate the use of novel lidars in our framework.

To improve the surface completion of SEE, we proposed a synthetic-to-real and lidar-agnostic, viewer-centred surface completion network, VCN, to estimate pose and generate object surfaces. We incorporate VCN into the SEE framework to complete the



surfaces of objects of interest. With SEE-VCN, we obtain a more robust, unified representation of objects across datasets, allowing the network to focus on learning geometry, rather than overfitting on scan patterns. Through extensive experiments we show that SEE-VCN outperforms SEE in most DA settings. We also outperform SOTA single-target domain adaptation work on multiple settings without needing to tune for a specific target domain.

The proposed approaches for domain adaptation in this chapter resulted in improved performance across multiple lidar types, however, the common limitation across all methods is the surface completion, and subsequently, accurate localization of objects with few lidar points. In autonomous driving datasets, there are often numerous objects in a point cloud frame with sparse observations due to objects being at far-range or occluded. For sparse lidar, this issue is more pronounced, as evident in the suboptimal domain transfer of the SEE approaches when tested with nuScenes (32-beam) point clouds in Table 3.1 and Table 3.7. In the following chapter, we address this by presenting an auto-labelling approach that significantly improves 3D object detection on various lidar types, including the detection of objects with few lidar points.

The contributions in this chapter have been presented in the following publications:

- D. Tsai, S. Worrall, M. Shan, A. Lohr, and E. Nebot. "Optimising the selection of samples for robust lidar camera calibration," In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pp. 2631–2638. IEEE, 2021.
- D. Tsai, J. S. Berrio, M. Shan, S. Worrall, and E. Nebot. "See eye to eye: A lidar-agnostic 3d detection framework for unsupervised multi-target domain adaptation," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7904–7911, 2022.
- D. Tsai, J. S. Berrio, M. Shan, E. Nebot, and S. Worrall. "Viewer-Centred Surface Completion for Unsupervised Domain Adaptation in 3D Object Detection,"

---

In *2023 IEEE International Conference on Robotics and Automation (ICRA)*.  
pp. 9346-9353. IEEE, 2023.

# Chapter 4

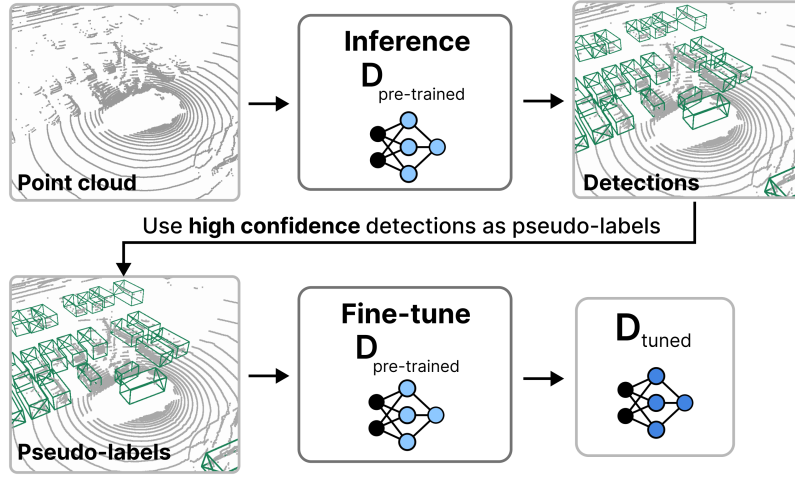
## Auto-labelling

### 4.1 Introduction

The field of 3D object detection has benefited greatly from the availability of large-scale autonomous vehicle datasets [Geiger et al., 2012, Sun et al., 2020, Caesar et al., 2020, Kesten et al., 2019], which have fueled research in the development of 3D detectors. Unfortunately, directly applying these 3D detectors to new datasets results in suboptimal performance due to domain shift caused by variations in lidars, geographic regions, scenarios, and sensor setups. Traditionally, the approach to improving detectors for new datasets is to manually annotate the data in the new domain to re-train the 3D detector. However, this approach is both costly and labour-intensive.

Self-training, shown in Fig. 4.1, is a promising UDA approach that follows the traditional approach of re-training the detector to a new domain, except with the use of automatically generated labels (aka. pseudo-labels). The key concept is to use the predictions of a pre-trained detector on a new target domain as pseudo-labels for re-training. Therefore, we also refer to self-training in this chapter as automatic labelling, or auto-labelling.

Many works have demonstrated the effectiveness of self-training for domain adaptation [Yang et al., 2021, Yihan et al., 2021, Tsai et al., 2023b, Saltori et al., 2020, Luo



**Figure 4.1** – Illustration of the self-training approach.

et al., 2021, Peng et al., 2023]. However, existing self-training approaches typically focus on adapting a single detector from one source domain to a new target domain [Yang et al., 2021, Luo et al., 2021, You et al., 2022]. With this approach, the source dataset and detector choice can significantly impact the domain adaptation performance. For example, adapting a Waymo-trained detector to KITTI achieves a far higher performance than adapting a nuScenes-trained detector to KITTI [Hu et al., 2023, Yang et al., 2021] due to Waymo and KITTI both having similar lidar resolution. In practice, with the ongoing development of novel lidar types, evaluating and identifying the optimal source dataset or detector for a particular scan pattern may be challenging due to the lack of labelled data on target domains.

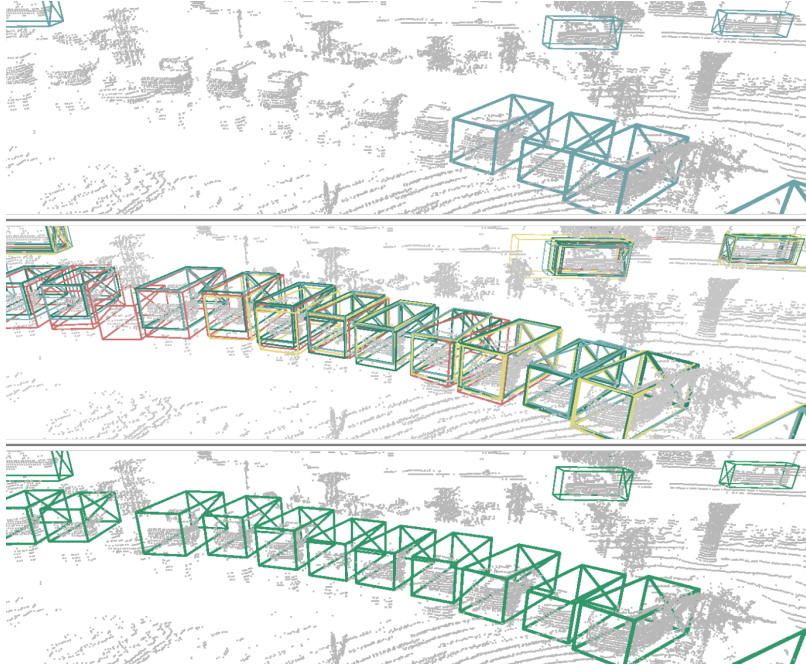
In this chapter, we propose a multi-source self-training framework, Multi-Source 3D (MS3D), as a general framework to enable the 3D detectors to be deployed on a range of lidar types. Through multi-source detector ensembling and temporal refinement, we demonstrate that 3D detectors trained with MS3D achieve state-of-the-art performance on both low and high density lidars, even comparable to the BEV detection results from detectors trained with human-annotated labels. In contrast to the single-source, multi-target SEE framework presented in Chapter 3, MS3D is a multi-source, single-target framework. The focus on a single-target domain allows MS3D to tailor solutions to achieve optimal 3D detection performance on any given target domain.

As a result, MS3D obtains a much farther detection range than the SEE framework, and higher performance for a given target domain.

A key component of our MS3D framework is our proposed KDE Box Fusion (KBF) method. Our KBF algorithm allows the effective fusion of multiple 3D bounding boxes by estimating a density function for each box parameter. We demonstrate through various experiments that this outperforms existing box fusion methods such as Weighted Box Fusion [Solovyev et al., 2021] or Non-Maximum Suppression [Neubeck and Van Gool, 2006]. By effectively fusing boxes with KBF, we can improve box localization and increase the recall rate of our pseudo-labels. This enables MS3D to have a strong initial set of pseudo-labels and accurate localization refinement of static vehicles.

In real-world scenarios, Multi-Source Domain Adaptation (MSDA) is a highly feasible approach for adaptation to a new target domain. The availability of multiple large-scale datasets and detector architectures effectively forms a vast database of detector and source domain pairings that can be used for domain adaptation. MS3D demonstrates that ensembling detectors from various source domains with different architectures can lead to improved domain generalization for a given target domain as shown in Fig. 4.2.

The versatility of MS3D lies in its capacity to tailor the 3D detector ensemble to include detectors with different architectures, source domains, hyperparameters, or class-specific detectors to a specific target domain making it an adaptable solution for a wide range of lidars and scenes. Moreover, MS3D presents a simple approach to domain adaptation by using the generated set of labels as a replacement for human-annotated labels in a supervised setting. This enables easy integration with any 3D detector architecture and data augmentation technique, making MS3D highly practical for adapting existing and future 3D detectors while preserving their inference runtime. This sets MS3D apart from other UDA works that require detector architecture modifications [Zhang et al., 2021], are limited to detectors with Bird’s Eye View (BEV) feature extraction [Yihan et al., 2021, Zhang et al., 2023], or involve additional inference steps [Tsai et al., 2022, Tsai et al., 2023a].



**Figure 4.2** – Domain generalization for unseen target domains can be improved by ensembling multiple pre-trained detectors from various source domains. Figure shows detector predictions on a point cloud from the Waymo dataset [Sun et al., 2020] in rainy weather. Top: PV-RCNN++ [Shi et al., 2023] trained on Lyft[Kesten et al., 2019], Middle: Ensemble of PV-RCNN++ and VoxelRCNN[Deng et al., 2020] trained on both Lyft and nuScenes[Caesar et al., 2020], Bottom: Fused detections with our Kernel-density estimation Box Fusion (KBF).

In this chapter, we first present the MS3D framework for multi-source domain adaptation. MS3D focuses on vehicle auto-labelling through multi-source ensembling, tracking and temporally consistent box refinement of static vehicles. To improve this framework, we conducted a comprehensive analysis of the performance of pre-trained detectors in a cross-domain context. This study provided valuable insights that were used to improve the MS3D framework. In particular, our improved framework, MS3D++, proposes a strategy for the effective use of multi-frame pre-trained detectors in the ensemble. Additionally, MS3D++ introduces an approach for auto-labelling pedestrians; this was not supported by the initial MS3D framework due to the non-rigid nature of pedestrians. Through our experiments, we demonstrate that MS3D++ outperforms MS3D with strong vehicle and pedestrian detection. At the time of publication, both MS3D and MS3D++ achieved SOTA performance, outper-

forming all existing works on multiple lidar types.

We summarise the main contributions of this chapter as follows:

- Multi-Source 3D (MS3D) Framework to combine multiple pre-trained detectors for auto-labelling a given target domain
- Quantitative analysis of pre-trained detector performance in a cross-domain context for better ensembling.
- An improved MS3D framework, MS3D++, leveraging insights from the quantitative cross-domain analysis to better auto-label both vehicles and pedestrians

All code and experiments for this chapter are released at <https://github.com/darrenjkt/ms3d>.

## 4.2 Multi-source Self-training Framework (MS3D)

### 4.2.1 Overview

In this section, we introduce MS3D, a multi-source domain adaptation approach for 3D object detection that leverages multiple source domain experts and temporal information to generate high quality pseudo-labels for self-training. The benefits of our approach are:

- **Robust:** By combining the expertise of different detectors trained on various source domains, MS3D is more robust against domain shift.
- **Versatile and Scalable:** MS3D is easily combined with multiple and various types of pre-trained 3D detectors to further boost pseudo-label quality.
- **Improved Pseudo-labels:** MS3D self-trained models can obtain state-of-the-art UDA performance regardless of the source pre-trained model chosen for fine-tuning. This is because our fused pseudo-labels consistently outperform the best individual source domain detector.

- **Speed:** MS3D does not add any processing latency during inference, ensuring it can be used in real-time applications.
- **Detection Performance:** MS3D achieves state-of-the-art on all UDA settings presented.

MS3D is built upon two fundamental observations. Firstly, we recognize that different detectors trained on various datasets possess distinct expertise. For instance, a detector trained on the Waymo dataset may be better suited for a specific novel dataset compared to a detector trained on nuScenes, and vice versa. Secondly, we observe that by utilizing temporal point clouds (i.e., multi-frame accumulation), we can aggregate various perspectives to gain a more comprehensive understanding of static objects’ complete geometry, leading to more accurate bounding boxes [Yang et al., 2021, Qi et al., 2021, Caesar et al., 2020, Chen et al., 2022].

A naive approach to combining multiple source-trained detectors on a new domain is to select the detection box produced by the most confident detector. However, this method neglects potentially valuable information provided by the other detectors, and it is possible that the most confident detector is not necessarily the most accurate. Instead, we propose KDE Box Fusion (KBF), a robust box-fusion strategy that utilizes KDE to consider each detector’s proposal and enhance pseudo-label quality. We note that KBF can also be applied for detector ensembling in a supervised setting.

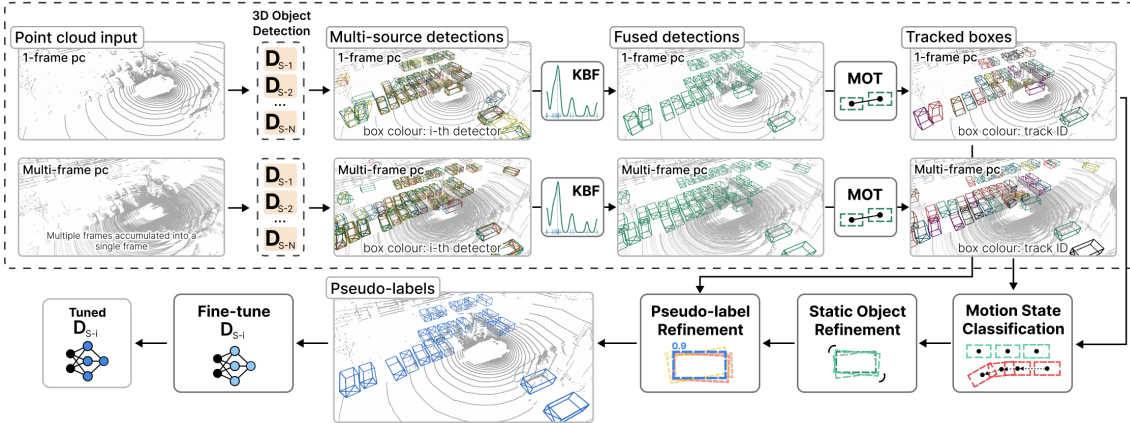
Point cloud accumulation has gained popularity as a promising direction for improving single-frame detection through densification. Yang et al. [Yang et al., 2021] demonstrated that concatenating 4 point cloud frames had the optimal performance boost before it started to deteriorate with more frames for the Waymo dataset. However, they observe that for static and slow objects, accumulating more than 4 frames continued to improve detection performance. Qi et al. [Qi et al., 2021] showed that accumulating 100+ frames of a sequence can be effectively utilized by separating static and dynamic vehicle box refinement. To capitalize on these findings, MS3D uses multi-frame detection to refine the box localization of static objects. Furthermore, by extending the static boxes of parked vehicles in both directions temporally,



we obtain precise pseudo-labels for objects located beyond 50m from the ego-vehicle.

### 4.2.2 Method

In MSDA, we have multiple  $M$  labelled source domains  $S_1, S_2, \dots, S_M$  and a single unlabelled target domain  $T$ . In our context, each  $i$ -th source  $S_i = \{(p_{S,i}, L_{S,i})\}_i^N$  is a detector that has been trained on a point cloud dataset  $p_{S,i}$  with annotations  $L_{S,i}$ . This source-trained detector's box predictions on the  $k$ -th frame of the unlabelled target dataset is denoted as  $\{B_{T,i}^k\}_i^N$ . Our goal is combine box predictions from each source-trained detector and utilize temporal information to robustly localize and classify objects in 3D for each frame. We assume that the vehicle ego-pose is known in world coordinates for ego-motion compensation. Fig. 4.3 illustrates our proposed framework, which we will introduce in the following sections.



**Figure 4.3 – The MS3D self-training framework.** Given a set of  $N$  pre-trained 3D detectors from multiple source domains  $D_{S-i}$  where  $i = 1, 2, \dots, N$ , we generate detections for both 1-frame and multi-frame accumulated point clouds, and combine them with our KBF. Thereafter, we feed the sequence frames into a 3D tracker for generating trajectories of both 1-frame and multi-frame boxes. Finally, we refine static object boxes to get our final pseudo-labels for fine-tuning any pre-trained detector  $D_{S-i}$ . Visualized in this figure is our MS3D framework for Waymo/Lyft  $\rightarrow$  nuScenes.

## Fusing Multiple Sources

**Initial Pseudo-labels.** In order to enhance the resilience of our model against domain shifts in the unlabelled target domain, we use different types of detectors and source datasets for a greater range of knowledge. For example, with Waymo as our target domain, we employ both nuScenes and Lyft source domains, with two types of detectors, namely SECOND [Yan et al., 2018] and CenterPoint [Yin et al., 2021]. We notice that Lyft’s lidar scan pattern is similar to that of Waymo, which could give Lyft-trained detectors an edge over nuScenes-trained detectors in certain scenarios. Additionally, different detectors possess varying areas of expertise. For instance, SECOND is adept at estimating object dimensions for axis-aligned objects due to its anchor boxes, whereas CenterPoint performs better with objects that are not axis-aligned [Yin et al., 2021].

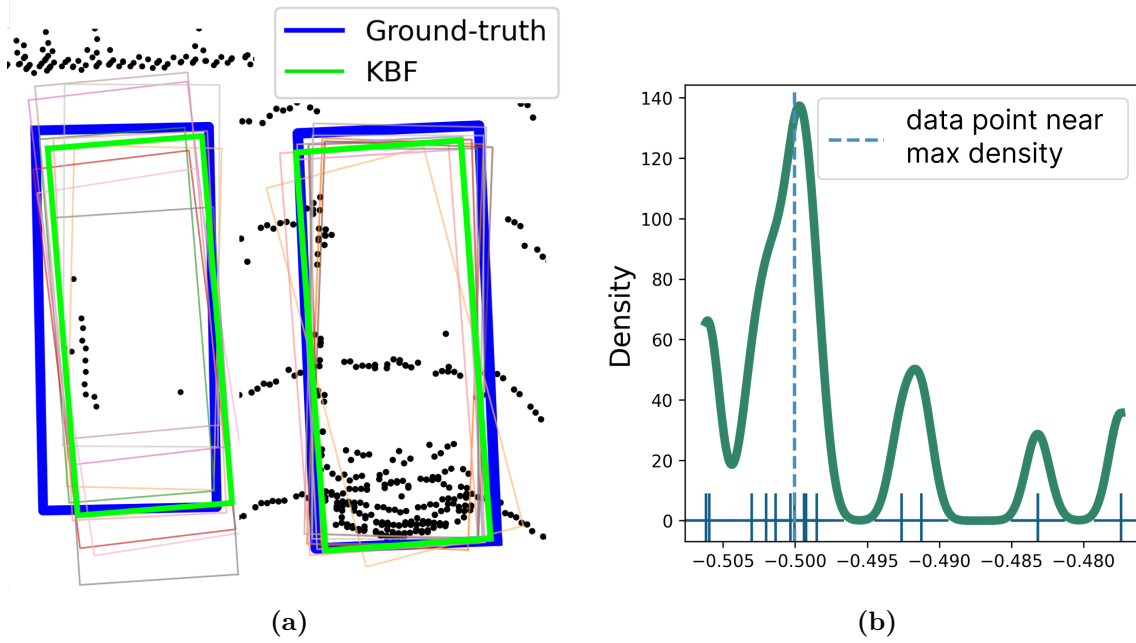
We take advantage of a detector’s full expertise by using two augmentations for Test-Time Augmentation (TTA) - random world flip (RWF) across x and y axes, and random world rotation (RWR) in the range  $[-\pi, \pi]$ . For each source detector we run four permutations: no TTA, RWF, RWR, RWF+RWR, which gives bounding boxes  $\{B_{T,i}^k\}_i^4$  for the  $i$ -th source detector on the  $k$ -th frame which we then fuse with KBF as follows.

**Kernel density estimation Box Fusion (KBF).** The backbone of our KBF method,  $\kappa(\cdot)$ , is Kernel Density Estimation (KDE) [Parzen, 1962], which is capable of estimating a probability density function (PDF) using a kernel function shown in Eq. (4.1). KDE places a kernel  $K$  at each data point  $x_i$  which sums to give the final PDF estimate  $\hat{f}(x)$  as shown in Fig. 4.4b. A weight  $w_i$  for each data point and a kernel bandwidth  $h$  can be adjusted for fine-tuning the PDF estimate.

$$\hat{f}(x) = \frac{1}{h} \sum_{i=1}^N w_i K\left(\frac{x - x_i}{h}\right) \quad (4.1)$$

In 3D object detection, a predicted box is parameterised by its centre  $(c_x, c_y, c_z)$ , dimensions  $(l, w, h)$ , heading  $\theta$  and a confidence score  $s$ . Regardless of the localization accuracy, each box carries potentially valuable information. For example, we

observe that a box with poor heading estimation may have accurate object dimensions. Therefore to combine the information of every box proposal, we opt to fuse the centre, length, width, height and heading separately.



**Figure 4.4** – (a) Fused TTA predictions of 4 source detectors (multi-coloured boxes) with KDE Box Fusion (KBF); (b) Example of how we select the best rotation of multiple boxes with KDE.

To merge boxes, we first create a KD-Tree with the box centroids. We identify matching boxes by querying for sets of boxes that are within a certain radius of each other. We compute KDE separately for the centroid, dimensions and score, then select the peak value of each of their PDF. We take the sine of the heading,  $\sin(\theta)$ , to ensure rotational continuity before applying KDE. We found empirically that selecting rather than combining box headings produces the best result. Therefore, we choose the box heading with the highest density based on the PDF. For each box parameter we use a Gaussian kernel and use the predicted box confidence score as weights. Given the set of 1-frame detections from  $i$  source detectors, we fuse them with KBF to obtain boxes  $B_{1f} = \kappa(\{B_{i,1f}\})$  for each frame  $k$ .

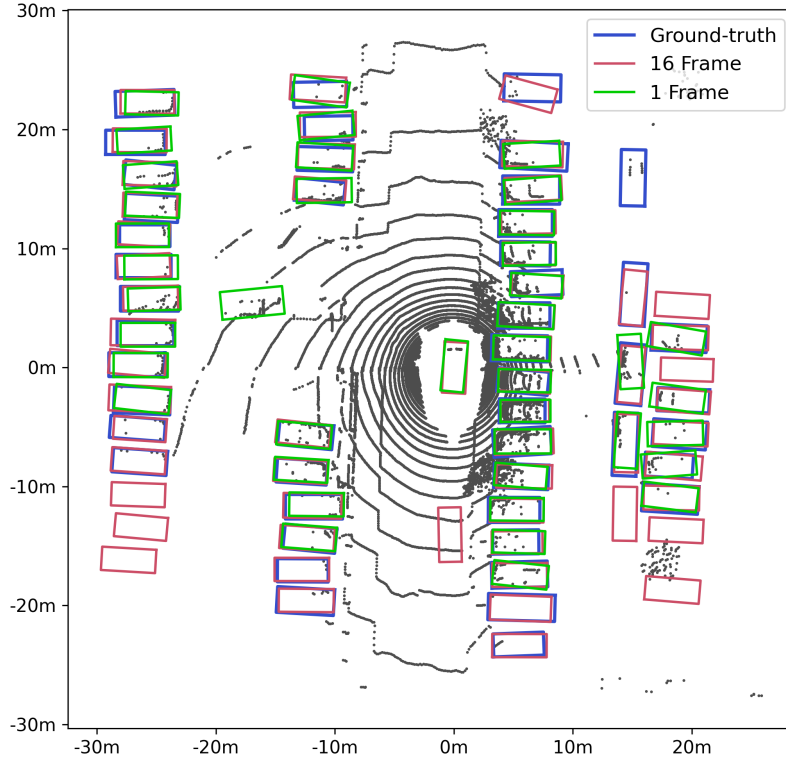
### Utilizing Temporal Information for Static Vehicles

In urban city environments, parked vehicles are common and often are completely static for long periods. Car parks in particular are often present in datasets and provides an abundance of viewpoints for various types of vehicles. Furthermore, due to the rigid-body property of vehicles, accurate static vehicle labelling would enable our detector to generalize to dynamic vehicles as well. To improve static vehicle localization, we use multi-frame detection to leverage the additional viewpoints provided by accumulated point clouds. For multi-frame detection, the input point cloud  $p_t^k$  for frame  $k$  is the stacking of  $N$  historical point clouds with the point cloud at time  $t$  where each historical point cloud has been transformed to the ego-frame at time  $t$ . With a 10Hz lidar, each point cloud is 0.1s apart, and the accumulated point cloud can be represented as  $p_t^k = \{p_0, p_{0.1}, p_{0.2}, \dots, p_{N(0.1)}\}$ . By feeding the source detector with  $p_t^k$ , it can utilize the increased point density to improve the accuracy of box localization, particularly for static objects that may be occluded or located at a distance as shown in Fig. 4.5. In particular, we follow [Chen et al., 2022] to use  $N = 16$  frames for detection for a trade-off between TTA inference time and point cloud density. We use the same TTA permutations and combine box predictions with KBF,  $B_{16f} = \kappa(\{B_{i,16f}\})$ , as in Section 4.2.2.

### Motion State Classification

To assign temporally consistent boxes for static vehicles, we associate detection boxes across frames to classify the motion state of the object. To accomplish this, we transform all boxes to world coordinates with known sensor poses. Thereafter, we utilize a tracking-by-detection, Kalman Filter-based tracker [Pang et al., 2021] which is an implementation variant of [Weng et al., 2020] to generate box trajectories for both 1-frame ( $T_{1f}$ ) and 16-frame ( $T_{16f}$ ) predictions. Following [Qi et al., 2021], we classify the tracklets as either static or dynamic based on pre-defined thresholds of begin-to-end distance and box centre variance.

Nonetheless, this is not sufficient in classifying  $T_{16f}$  due to predictions on the motion



**Figure 4.5** – 1-frame KBF boxes compared to 16-frame KBF boxes on the nuScenes dataset with detectors trained on Waymo/Lyft as the source domain. Visualized is a 1-frame point cloud.

trails of dynamic objects in 16-frame detection. We observe that when two dynamic vehicles are driving in the same lane at a similar speed, their motion tails often overlap, leading to a false 'static' classification. A simple approach to replacing static boxes would be to rely solely on the motion classification of  $T_{1f}$  and replace static objects with their corresponding  $T_{16f}$  boxes. However, this approach is suboptimal since 1-frame detection often detects less static objects at a distance than 16-frame detection for certain target datasets as shown in Fig. 4.5, causing us to overlook valuable 16-frame boxes as a result.

To improve the accuracy of  $T_{16f}$  motion classification, we make a key observation: in 3D space, dynamic objects cannot occupy the same space as an object that is static throughout the sequence. This means that the trajectory of a dynamic object will never overlap with that of a parked vehicle. To take advantage of this, we use  $T_{1f}$  to identify all dynamic objects in frame  $k$ , and project their entire trajectory  $T_{1f}$  into

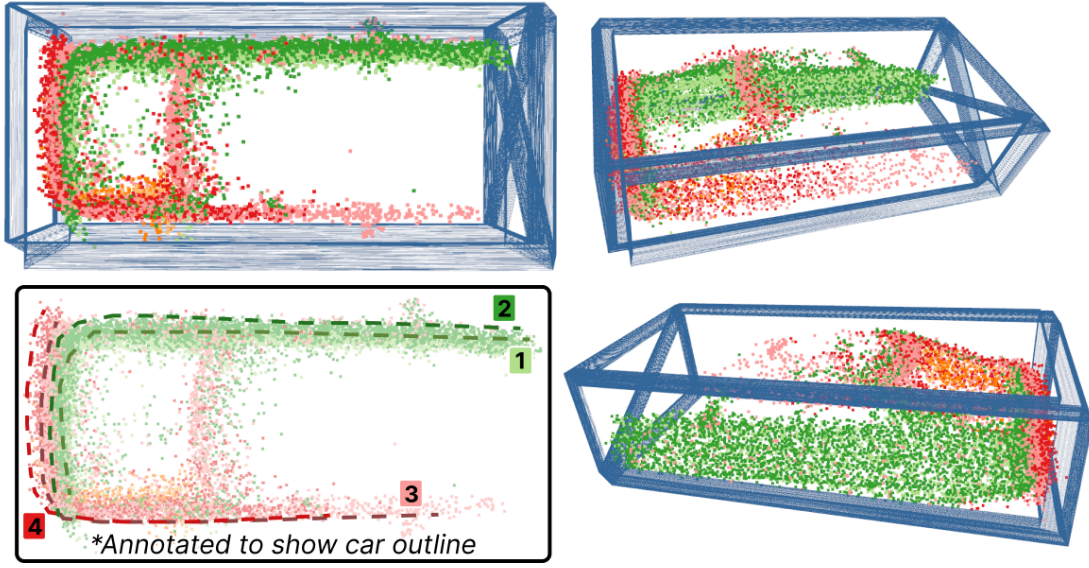
frame  $k$ . This effectively identifies drivable areas and enables us to filter out wrongly classified static boxes through comparing IoU-matched  $T_{1f}$  and  $T_{16f}^k$  boxes. If either  $T_{1f}$  or  $T_{16f}^k$  is labelled as dynamic, we classify the object as dynamic. While this might lead to more objects falsely classified as dynamic, it prevents harmful extrapolation of vehicles that are only temporarily stationary.

### Multi-frame Static Object Refinement

We anticipate that a parked car’s position in the real world should remain constant. However, we have observed that attempts to assign a single bounding box to a parked car for an entire sequence often leads to the exclusion of some car points in certain frames. To investigate this, we accumulated points and ground truth boxes from all frames in a sequence and noticed many parked cars had poor registration across frames. Moreover, the bounding box positions were inconsistent because of the apparent motion of static vehicle points over time. Fig. 4.6 shows that human annotators assign multiple boxes to a parked car over 197 frames due to poor registration. This finding is supported by the nuScenes labelling guidelines where human annotators are explicitly told to annotate a separate cuboid for stationary objects in each frame due to localization errors. Therefore, selecting any ground truth box for the car may lead to the partial or complete exclusion of object points. This highlights the fact that relying on a single bounding box to label a parked vehicle throughout a sequence can be unsatisfactory.

This poor registration of parked vehicles was observed in multiple datasets such as nuScenes, Waymo, Lyft and KITTI with some datasets worse than others. To address this, we propose to use KBF on  $H$  historical observation frames,  $\kappa(\{T_{16f}\}_{k-H}^k)$ , to obtain a static box for the  $k$ -th frame instead of using all frames. This gives us a static box  $\{B_{\text{static}}^k\}_k^N$  for each  $k$ -th frame with a score determined by  $\max(\alpha, s_{\text{static}}^k)$  where  $\alpha$  is a minimum score threshold and  $s_{\text{static}}^k$  is the fused score of  $\{B_{\text{static}}^k\}_k^N$ .

To extend the range of pseudo-labels, we assume that static vehicles with sufficiently long tracks will remain static throughout the sequence. Many pre-trained detectors



**Figure 4.6** – Accumulated frames of a parked car and its assigned ground-truth boxes from all frames of a Waymo sequence (197 frames/boxes total). **Light green**, **dark green**, **pink**, **red** and **orange** represent accumulated points in intervals of 40 frames in order where light green and orange represents frames 0-40 and 160-197 respectively. The bottom left annotation illustrates the registration misalignment, especially between light green (1) and red (4) outlines on the car’s rear, potentially caused by ego-pose drift. Best viewed in colour.

can only reliably identify vehicles within a closer range for a new target domain. Therefore, we propagate the first and last KBF boxes of static vehicles through historical and future frames, respectively, under the assumption that they remain static. To reduce the increase in false positives resulting from this propagation, we decay the confidence score of the propagated boxes by  $\beta$  for each frame that it is propagated. For certain datasets, we found that updating the pseudo-labels in the self-training process to refine these propagated boxes can give a bump in performance.

### Pseudo-Label Refinement

In the UDA setting, higher point density with 16-frame accumulation does not necessarily lead to improved detection at close ranges compared to 1-frame detections. This is partly because densified point clouds may accentuate differences in scan patterns, and also because static boxes can be susceptible to ego-pose drift, which negatively

affects localization accuracy. To address this issue and combine all box proposals, we employ Non-Maximum Suppression (NMS) to allow confident single frame detections to replace 16-frame static boxes. We project the following into the scene and apply NMS for filtering: (1) 1-Frame KBF boxes; (2)  $T_{1f}$  tracked boxes; and (3) Static object boxes,  $\{B_{\text{static}}^k\}_k^N$ . Finally, we remove any pseudo-labels with less than 1 point inside their box. By including  $T_{1f}$  tracked boxes, we can interpolate missing detections and extrapolate for farther ranges in a similar manner to [You et al., 2022].

### 4.2.3 Experiments

#### Setup

**Datasets.** Many works have addressed the object size domain gap and improved adaptation to the KITTI dataset [Yang et al., 2021, Wang et al., 2020, Tsai et al., 2023a, Luo et al., 2021, You et al., 2022]. However, other factors such as scan pattern, weather and geographical differences have a significant impact on the domain gap. To focus on this, we select datasets with similar object sizes: Waymo [Sun et al., 2020], Lyft [Kesten et al., 2019] and nuScenes [Caesar et al., 2020], summarised in Table 4.1, for our experiments.

| Dataset  | Lidar Beams       | Size    | Rain | Night | Country        |
|----------|-------------------|---------|------|-------|----------------|
| nuScenes | 1×32              | 34,149  | Yes  | Yes   | USA, Singapore |
| Waymo    | 1×64 + 4×200      | 198,068 | Yes  | Yes   | USA            |
| Lyft     | 1×40 or 64 + 2×40 | 22,680  | No   | No    | USA            |

**Table 4.1** – Overview of each dataset. Country refers to the country in which the dataset was collected.

**Methods.** We compare MS3D with the following methods: (1) Source-only, which is the direct evaluation of a source-trained detector on the target dataset; (2) SN as a baseline to observe how much of the domain gap is caused by size differences; (3) ST3D, the state-of-the-art self-training approach for 3D object detection; (4) Lidar Distillation, an approach that builds on ST3D for the Waymo to nuScenes setting;



(5) GT-Trained, the best performance we can achieve when we fine-tune the source-trained detector on the target ground-truth labels. We compare methods (1)-(4) with GT-Trained as it is a more realistic scenario of what is achievable in the presence of perfect pseudo-labels, as opposed to careful selection of the voxel sizes and anchor box dimensions for an "Oracle" supervised benchmark [Yang et al., 2021]. We fine-tune all models with no frame accumulation. We only compare Lidar Distillation for the target-nuscenes setting as it is designed specifically for the high beam to low beam UDA setting. Though we use SECOND-IoU and CenterPoint for generating pseudo-labels, we only focus on the validation results of SECOND-IoU from different source domains to highlight domain discrepancies.

**Evaluation Metric.** We follow [Yang et al., 2021, Wang et al., 2020, Wei et al., 2022] and adopt the KITTI evaluation metric of Average Precision (AP) over 40 recall positions, in Bird’s Eye View (BEV) and 3D, denoted as  $AP_{BEV}/AP_{3D}$ . We assess all models on the "Vehicle" category. For Lyft and nuScenes, we map "truck" and "bus" to the "car" category. We report the AP at IoU thresholds of 0.7 and 0.5 i.e. a vehicle is considered a true positive detection if the IoU of the predicted and ground truth box is over 0.7 or 0.5. We present results at IoU=0.5 as well to highlight that our method greatly improves the number of correctly detected vehicles.

**Implementation Details.** We train SECOND-IoU [Yan et al., 2018] and CenterPoint [Yin et al., 2021] on all datasets and select the best pre-trained model based on evaluating with the source dataset labels. We re-use the same source pre-trained detectors for adapting to all settings. We argue that this is more realistic to real-world applications as we may not have sufficient labels to evaluate and tune the best model for adaptation to a new target dataset. All our models were trained with the OpenPCDet [Team, 2020] codebase with tweaked settings for voxel size of (0.1m, 0.1m, 0.15m), detection range of  $[-75, 75]$ m for x,y axes and  $[-2, 4]$  for z-axis, and we shift the point cloud frame to the ground plane. Lyft and nuScenes pre-trained models were trained with 5 and 10 frame accumulation respectively, following popular practice [Kesten et al., 2019, Caesar et al., 2020].

For MS3D, we adopt the settings of ST3D [Yang et al., 2021] and fine-tune all pre-

| Target: nuScenes |        |             |                      |                      |
|------------------|--------|-------------|----------------------|----------------------|
| Method           | Source | Detector    | IoU=0.7              | IoU=0.5              |
| Source-Only      | Waymo  | SECOND      | 32.91 / 17.24        | 43.32 / 37.58        |
|                  |        | CenterPoint | 32.10 / 17.77        | 40.83 / 35.77        |
|                  | Lyft   | SECOND      | 24.15 / 13.47        | 30.52 / 26.79        |
|                  |        | CenterPoint | 23.71 / 11.31        | 30.45 / 26.17        |
| SN               | Waymo  | SECOND      | 33.23 / 18.57        | 43.19 / 37.74        |
|                  | Lyft   | SECOND      | 27.51 / 17.00        | 33.32 / 29.92        |
| ST3D             | Waymo  | SECOND      | 35.92 / 20.19        | 43.03 / 38.99        |
|                  | Lyft   | SECOND      | 29.88 / 18.37        | 33.18 / 30.67        |
| L.D              | Waymo  | SECOND      | 42.04 / 24.50        | - / -                |
| MS3D             | Waymo  | SECOND      | <b>42.23 / 24.76</b> | <b>52.33 / 46.65</b> |
|                  | Lyft   | SECOND      | <b>41.64 / 23.46</b> | <b>51.47 / 45.74</b> |
| GT-Trained       | Waymo  | SECOND      | 44.39 / 29.46        | 55.61 / 50.83        |

**Table 4.2** – UDA from Waymo/Lyft to **nuScenes**. We report  $AP_{BEV}/AP_{3D}$ . Best results for each source-target pair are highlighted in **bold**. GT-Trained uses GT labels to fine-tune the pre-trained detectors with no frame accumulation. MS3D is compared with SN [Wang et al., 2020], ST3D [Yang et al., 2021] and L.D [Wei et al., 2022]

trained detectors with no frame accumulation, for 30 epochs at a learning rate of  $1.5 \times 10^{-3}$  with the Adam one-cycle scheduler. We use pseudo-labels with score threshold of  $s > 0.6$  as the ground-truth labels for fine-tuning. For fusing proposals with KBF, we consider a vehicle as present if there are more than 4 box proposals with their centroids within 2m of each other. We use  $H = 16$  historical frames for static object refinement and propagate them with a decay of  $\beta = 0.95$  with a minimum score  $\alpha = 0.7$  if they have more than 7 tracked detections. To focus on static vehicles, we sort Waymo and nuScenes scenes by number of cars and select the top 150 and 190 scenes from nuScenes and Waymo respectively for training; scenes with many cars tend to be car parks. For an unlabelled target domain, this can similarly be done by counting the number of fused 1-frame detections per scene, or by intentionally collecting data from scenes with many parked vehicles. Additionally, this suggests that our model can be further improved with a second round of self-training with the remaining scenes which we leave for future work.

| Target: Waymo |          |             |                      |                      |
|---------------|----------|-------------|----------------------|----------------------|
| Method        | Source   | Detector    | IoU=0.7              | IoU=0.5              |
| Source-Only   | Lyft     | SECOND      | 52.14 / 34.40        | 61.66 / 58.71        |
|               |          | CenterPoint | 50.56 / 31.02        | 62.32 / 58.97        |
|               | nuScenes | SECOND      | 50.30 / 24.70        | 61.40 / 53.73        |
|               |          | CenterPoint | 49.39 / 24.33        | 59.52 / 52.64        |
| SN            | Lyft     | SECOND      | 53.39 / 39.22        | 62.42 / 59.55        |
|               | nuScenes | SECOND      | 50.69 / 28.86        | 61.38 / 53.90        |
| ST3D          | Lyft     | SECOND      | 56.06 / 39.17        | 65.16 / 62.39        |
|               | nuScenes | SECOND      | 55.67 / 28.83        | 67.19 / 61.63        |
| MS3D          | Lyft     | SECOND      | <b>61.25 / 42.88</b> | <b>72.34 / 69.30</b> |
|               | nuScenes | SECOND      | <b>61.39 / 42.76</b> | <b>72.47 / 69.45</b> |
| GT-Trained    | nuScenes | SECOND      | 66.76 / 52.50        | 75.16 / 72.40        |

**Table 4.3** – UDA from nuScenes/Lyft to **Waymo**. We report  $AP_{BEV}/AP_{3D}$ . MS3D is compared with SN [Wang et al., 2020] and ST3D [Yang et al., 2021].

| Target: Lyft |          |             |                      |                      |
|--------------|----------|-------------|----------------------|----------------------|
| Method       | Source   | Detector    | IoU=0.7              | IoU=0.5              |
| Source-Only  | Waymo    | SECOND      | 49.68 / 38.94        | 56.22 / 54.63        |
|              |          | CenterPoint | 43.83 / 32.98        | 49.94 / 48.42        |
|              | nuScenes | SECOND      | 42.22 / 23.62        | 48.45 / 45.51        |
|              |          | CenterPoint | 45.98 / 26.36        | 52.41 / 49.43        |
| SN           | Waymo    | SECOND      | 71.61 / 56.13        | 80.65 / 78.52        |
|              | nuScenes | SECOND      | 63.11 / 39.60        | 72.27 / 68.25        |
| ST3D         | Waymo    | SECOND      | 73.86 / 56.33        | 82.90 / 80.81        |
|              | nuScenes | SECOND      | 67.33 / 41.82        | 77.17 / 72.99        |
| MS3D         | Waymo    | SECOND      | <b>76.48 / 61.23</b> | <b>83.36 / 81.49</b> |
|              | nuScenes | SECOND      | <b>75.02 / 59.01</b> | <b>83.46 / 81.52</b> |
| GT-Trained   | Waymo    | SECOND      | 81.10 / 66.76        | 91.12 / 88.69        |

**Table 4.4** – UDA from nuScenes/Waymo to **Lyft**. We report  $AP_{BEV}/AP_{3D}$ . GT-Trained uses Lyft labels to fine-tune the pre-trained detector with no frame accumulation. MS3D is compared with SN [Wang et al., 2020] and ST3D [Yang et al., 2021].

## Results

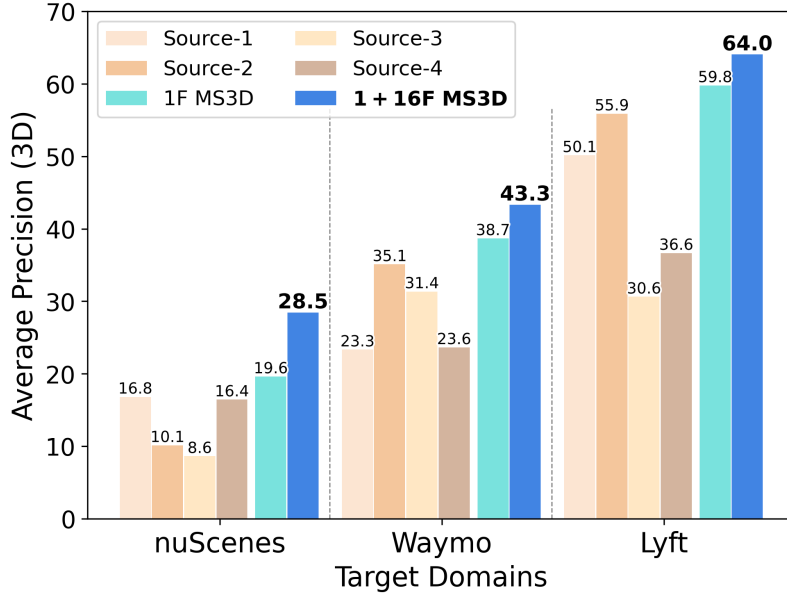
As shown in Tables 4.2 to 4.4, MS3D outperforms all the other methods regardless of which source domain is used. With nusenes as target domain, Table 4.2, MS3D not

only improves detections with accurate box localizations ( $\text{IoU}=0.7$ ) and outperforms existing approaches but also greatly increases the number of correct object proposals ( $\text{IoU}=0.5$ ) by a large margin. It can be seen that whilst SN and ST3D improved box localization with  $\text{IoU}=0.7$ , they did not greatly increase the number of true positive detections as  $\text{AP}_{\text{BEV}}$  at  $\text{IoU}=0.7$  remains relatively unchanged. In contrast, MS3D obtains both a high performance for both  $\text{AP}_{3\text{D}}$  and  $\text{AP}_{\text{BEV}}$ . In particular, we highlight that for  $\text{AP}_{\text{BEV}}$ , MS3D is only  $\sim 2\text{AP}$  less than the GT-Trained approach, indicating that we can nearly detect all vehicles in the scene. This is also apparent with Waymo as target in Table 4.3, where MS3D obtains a high 42.88  $\text{AP}_{3\text{D}}$  and 61.25  $\text{AP}_{\text{BEV}}$  for  $\text{IoU}=0.7$  which is  $\sim 5\text{AP}_{\text{BEV}}$  less than GT-Trained. With Lyft as the target domain, Table 4.4, MS3D also outperforms other methods in 3D and BEV for both  $\text{IoU}$  criterions. From the above results, we have demonstrated that MS3D is able to adapt detectors from low (e.g. nuScenes) to high (e.g. Waymo) beam lidar datasets and vice-versa, unlike ST3D [Yang et al., 2021] and Lidar Distillation [Wei et al., 2022]. In Fig. 4.8 we show qualitative results of MS3D for all target domains.

For each target domain, we show the detection performance from two different source domains. We demonstrate that with MS3D, regardless of the pre-trained detector’s source dataset, the detector can be fine-tuned to a similar state-of-the-art performance. In contrast, ST3D’s choice of pre-trained detector’s source dataset can have a significant impact on performance, with a 10.34  $\text{AP}_{3\text{D}}$  difference at  $\text{IoU}=0.7$  between selecting nuScenes or Lyft. MS3D’s robustness is especially valuable in real-world scenarios where acquiring sufficient labelled data for evaluation can be costly.

## Ablation

**Pseudo-label Quality.** In Fig. 4.7, we show that our pseudo-labels consistently improve the detections of every source-trained detector on the target domain. When we combine all 4 sources with KBF, we can already surpass the best source pre-trained detector’s  $\text{AP}_{3\text{D}}$ . By adding 16-frame detections for refining of static vehicles, we can further boost the detection by an impressive amount for all target domains. For example with nuScenes as the target domain, MS3D overcomes the detection range



**Figure 4.7** – Evaluation of MS3D’s pseudo-labels on different target domains. 1F MS3D refers to using only KBF on 1-frame detections from the 4 source domains. 1+16F MS3D refers to our final pseudo-labels after refining static vehicles.

limitations of a sparse-beam lidar with our multi-frame static refinement, leading to +8.84AP<sub>3D</sub> increase from 1F MS3D to 1F+16F MS3D.

**KDE Box Fusion.** We compare KBF with NMS [Neubeck and Van Gool, 2006] and two variants of Weighted Box Fusion [Solovyev et al., 2021] in Table 4.5. WBF-C is the official 3D implementation using opposite 3D box corners, and WBF-P is weighted average of box params ( $c_{xyz}, l, w, h, \theta, s$ ). We show that KBF outperforms other methods at all ranges, with the 50-80m range having the most improvement over existing methods. With KDE, setting a high bandwidth is similar to taking the average of the data points. Being able to tune the bandwidth for different parameters allows us to better capture the nuances of the box fusion process. For example, a common error in box prediction is the heading orientation being incorrect by an error of  $\pi$ . Setting a high bandwidth for rotation estimation would therefore be detrimental as we may select an in-between orientation instead of one that corrects the orientation by  $\pi$ .

**Static Object Refinement.** In Table 4.6, we show that using 16-frame detections

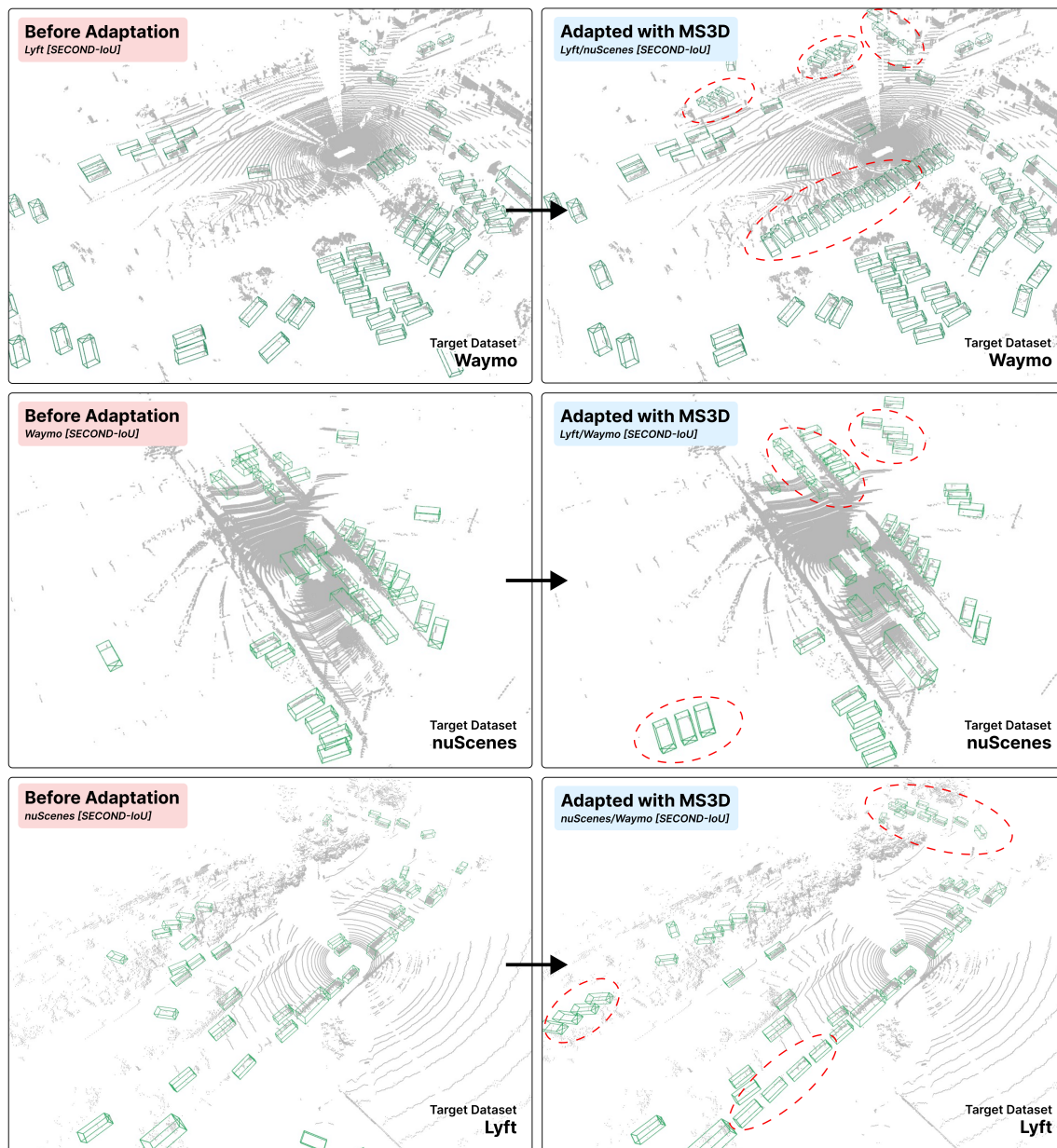
| Method          | IoU=0.7                             | RANGE (L2/APH) |              |              |
|-----------------|-------------------------------------|----------------|--------------|--------------|
|                 | AP <sub>BEV</sub> /AP <sub>3D</sub> | [0,30)         | [30,50)      | [50,+inf)    |
| N / CenterPoint | 49.49 / 23.33                       | 58.80          | 17.56        | 1.89         |
| N / SECOND      | 50.74 / 23.64                       | 60.07          | 15.56        | 1.61         |
| L / CenterPoint | 49.26 / 31.35                       | 61.53          | 30.97        | 9.88         |
| L / SECOND      | 50.83 / 35.09                       | 46.98          | 19.11        | 6.93         |
| NMS             | <b>57.93</b> / 34.88                | 57.13          | 24.01        | 7.59         |
| WBF-C           | 53.27 / 36.55                       | 67.17          | 32.74        | 8.33         |
| WBF-P           | 54.44 / 36.75                       | 67.43          | 33.32        | 8.92         |
| KBF (Ours)      | 57.22 / <b>38.67</b>                | <b>68.89</b>   | <b>33.93</b> | <b>10.40</b> |

**Table 4.5** – Ablation study of box fusion methods for multiple detector predictions on Waymo as target domain. KBF is compared with NMS [Neubeck and Van Gool, 2006], WBF-C and WBF-P [Solovyev et al., 2021].

for static objects ( $H = 0$ ) gives a slight improvement over 1F MS3D (i.e., using KBF for 1-frame detections). Applying KBF on  $H > 0$  historical boxes of a static object can further improve the localization of the box. As explained in Section 4.2.2, using KBF on all historical boxes is inadequate. This is shown in the lower performance of “Single Box” compared to using  $H = 16$  boxes. Quantifying the extent of this difference is challenging because comparing  $H = 16$  and “Single box” to a GT label may only differ by a few cm, which can still meet the IoU=0.7 threshold. Errors in point cloud alignment over long sequences becomes problematic when there are sparse number of points on the car because localization errors can result in the car’s points being outside the box entirely. Therefore, based on qualitative and quantitative assessment, we chose to use  $H = 16$  boxes for estimating the bounding box of a parked vehicle.

## Discussion

In our experimentation with various source to target domain pairs, we observed that while 16-frame accumulation increased point density, it did not necessarily help detectors localize well even when the entire car shape is visible. For example, when testing nuScenes/Waymo detectors on KITTI, we found that for a fully visible, 16-frame accumulated car at 5m away from the ego vehicle, detectors consistently predicted a



**Figure 4.8** – Qualitative results for MS3D. Red circles highlight the areas where MS3D detected true positive vehicles that were missed by the “Source-only” models.

| Method                 | IoU=0.7                             | RANGE (L2/APH) |              |              |
|------------------------|-------------------------------------|----------------|--------------|--------------|
|                        | AP <sub>BEV</sub> /AP <sub>3D</sub> | [0,30)         | [30,50)      | [50,+inf)    |
| nuScenes / CenterPoint | 49.49 / 23.33                       | 58.80          | 17.56        | 1.89         |
| nuScenes / SECOND      | 50.74 / 23.64                       | 60.07          | 15.56        | 1.61         |
| Lyft / CenterPoint     | 49.26 / 31.35                       | 61.53          | 30.97        | 9.88         |
| Lyft / SECOND          | 50.83 / 35.09                       | 46.98          | 19.11        | 6.93         |
| 1F MS3D                | 57.22 / 38.67                       | 68.89          | 33.93        | 10.4         |
| $H = 0$                | 59.19 / 38.73                       | 69.49          | 34.47        | 10.63        |
| $H = 4$                | 62.06 / 40.89                       | 71.74          | 38.83        | 14.11        |
| $H = 16$               | <b>62.32 / 43.33</b>                | <b>72.23</b>   | <b>41.72</b> | 17.63        |
| Single box             | 62.32 / 43.12                       | 72.02          | 41.30        | <b>17.88</b> |

**Table 4.6** – Ablation study of our static box labelling using a rolling window of  $H$  frames for KBF on Waymo as target domain. For  $H = 0$  we replace static boxes with their 16-frame detections. For “Single box”, we use KBF of all observed frames.

box that was 0.5-1m longer. This was also observed when using a KITTI detector on nuScenes/Waymo where it would predict a smaller box that did not encapsulate the entire object. This is likely due to the scan pattern domain gap as 16-frame accumulated point clouds appear quite distinct depending on the lidar used. To address this gap when the target dataset statistics deviate significantly from the source dataset, we recommend incorporating Random Object Scaling [Yang et al., 2021] or SN [Wang et al., 2020] in the self-training process.

MS3D obtains a high performance across various types of lidars. However, given the versatility of the ensembling step, we believe that the initial set of pseudo-labels can be further improved through a proper understanding of the performance of pre-trained detectors in a cross-domain setting. In the following section, we design a comprehensive study to quantify the performance of different 3D detector architectures on various lidar datasets in order to better tailor our ensemble for any given target domain.



## 4.3 Quantifying Cross-domain 3D Object Detection

### 4.3.1 Overview

Cross-domain 3D object detection is the task of applying a 3D detector, trained on a labelled source dataset, to a separate target dataset where factors such as sensor setup and geographical location may vary significantly. In this section, we aim to provide a deep understanding of the nature of pre-trained detectors in a cross-domain context in order to improve our ensembling of pre-trained detectors to achieve optimal 3D detection on a wide variety of lidar types.

Many existing works propose methods to address cross-domain adaptation, however the study of cross-domain performance without adaptation (aka. “Source-Only” or “Direct Transfer”) of 3D detectors is often overlooked. We believe that a comprehensive understanding of the baseline cross-domain performance and domain gap idiosyncrasies will enhance the community’s ability to obtain higher domain adaptation performance, and overall, better generalization to any given target domain. For example, most UDA approaches adapt a 32-beam (nuScenes) to 64-beam with a detector trained on 1-frame of nuScenes as the pre-trained model [Yang et al., 2021, Hu et al., 2023, Chen et al., 2023]. However, we demonstrate in Section 4.3.3 that simply pre-training with 10 accumulated frames can increase the vehicle detection rate of nuScenes on Waymo point clouds by  $+21\text{AP}_{3D}$  compared to just pre-training on 1-frame. We show in Table 4.9 that a multi-frame pre-trained Waymo detector can even outperform a prominent self-training UDA work, ST3D [Yang et al., 2021, Yang et al., 2022], when testing on nuScenes. Overall, we demonstrate that multi-frame pre-training presents a stronger baseline performance on various target domains without requiring complicated source-target alignment approaches. The findings from this section are subsequently used to improve our MS3D framework which we explain in more detail in Section 4.4.

### 4.3.2 Experiment Setup

#### 3D Detector Selection

To investigate the performance of various point cloud representations in the cross-domain setting, we select five detectors that are representative of the various point cloud representations and detection heads. For point cloud representation we select VoxelRCNN [Deng et al., 2020] (voxel-based), IA-SSD [Zhang et al., 2022] (point-based) and PV-RCNN++ [Shi et al., 2023] (hybrid). For detection heads, we select Anchorhead [Yan et al., 2018] (aka. RPN, anchor-based) and Centerhead [Yin et al., 2021] (anchor-free) as our representative methods. Except for IA-SSD, we select two-stage 3D detectors, PV-RCNN and VoxelRCNN, due to their superior performance over single-stage detectors. For PV-RCNN and VoxelRCNN, we conduct experiments with both anchor and centerhead. In summary, we conduct this study on five detectors: (1) PV-RCNN with Anchorhead (PV-A), (2) PV-RCNN with Centerhead (PV-C), (3) VoxelRCNN with Anchorhead (VX-A), (4) VoxelRCNN with Centerhead (VX-C), and (5) IA-SSD. Following existing cross-domain adaptation works [Yang et al., 2021, Tsai et al., 2023b], each voxel-based detector for every dataset is trained with a voxel size of  $[0.1, 0.1, 0.15]$ m, detection range of  $[-75, 75]$ m for x,y and  $[-2, 4]$ m for z-axis, and we shift the lidar frame to the ground plane. We train detectors with point clouds  $\{P_i \in \mathbf{R}^{n_i \times C}\}$ ,  $i = 1, 2, \dots, N$  with the  $i$ -th frame point cloud  $P_i$  consisting of  $n_i$  points with  $C$  channels for each point, for  $N$  total frames. For single frame 3D detection, each point in  $P_i$  has channels  $C = 3$  consisting of the  $XYZ$  point coordinates. For multi-frame 3D detection, each point has  $C = 4$  channels corresponding to  $XYZT$  where  $T$  is the timestamp which we elaborate on in Section 4.3.3. We follow default settings from [Team, 2020] for other hyperparameters.

#### Dataset Selection and Evaluation

We focus on three popular datasets for this study: (1) Waymo Open Dataset [Sun et al., 2020] with a 64-beam lidar, (2) nuScenes [Caesar et al., 2020] with a 32-beam

**Table 4.7** – Overview of datasets used for this study.

| Dataset  | Lidar Beams       | Size    | Rain | Night | Country        |
|----------|-------------------|---------|------|-------|----------------|
| nuScenes | 1×32              | 34,149  | Yes  | Yes   | USA, Singapore |
| Waymo    | 1×64 + 4×200      | 198,068 | Yes  | Yes   | USA            |
| Lyft     | 1×40 or 64 + 2×40 | 22,680  | No   | No    | USA            |

lidar, and (3) Lyft [Kesten et al., 2019] with a 40-beam and 64-beam lidar. These are large-scale datasets, summarised in Table 4.7, with over 20,000 frames of labelled data and containing variations in scan pattern, geography, and weather. We note that the frame rate of each of these datasets are different: Waymo is given at 10Hz, nuScenes has 2hz key frames with 20Hz sweeps, and Lyft is at 5Hz. As a result, the historical accumulation of 10 frames in nuScenes (0.45s) is not an equivalent time period to 10 frames in Waymo (0.9s). We report the 3D Average Precision ( $AP_{3D}$ ) for IoU at 0.7 using Waymo’s range evaluation of L2/AP for each table on “Vehicle” and “Pedestrian” classes. “Vehicle” evaluation in Lyft and nuScenes includes the car, truck and bus subcategories.

### 4.3.3 Results and Discussion

#### Training with Accumulated Point Clouds

First we explore the popular trend of point cloud accumulation in the cross-domain context, specifically for short sequence accumulation (i.e., less than 0.5s of historical frames). Accumulation is implemented by transforming past point clouds to the coordinate system of the current frame and appending a scalar timestamp  $T$  indicating the time delta in seconds from the current frame. We denote the maximum time delta as  $T_{\Delta MAX}$ . For instance, for a 10Hz lidar where each point cloud is 0.1s apart, the accumulated point cloud at time  $t$  can be represented as  $P_t = \{P_0, P_{0.1}, P_{0.2}, \dots, P_{N(0.1)}\}$  for  $N$  historical frames with each point represented as  $XYZT$  with  $T_{\Delta MAX} = N(0.1)$ . Directly concatenating the points from multiple historical frames has been demonstrated to improve performance for multiple 3D detectors [Yin et al., 2021, Shi et al., 2020, Liu et al., 2023, Chen et al., 2022, Yang et al., 2021] across various datasets

[Caesar et al., 2020, Sun et al., 2020] due to increased point density of objects.

Currently, most domain adaptation works for 3D object detection focus on single-frame detector adaptation [Yang et al., 2021, You et al., 2022, Tsai et al., 2022, Zhang et al., 2023, Hu et al., 2023]. However, in Table 4.8 we demonstrate that simply training a detector on short sequence, accumulated point clouds can significantly improve domain generalization across multiple target domains for both vehicles and pedestrians. Furthermore, Table 4.9 shows that multi-frame pre-training of SECOND-IoU on the Waymo dataset can even outperform ST3D [Yang et al., 2021] on the nuScenes dataset without any domain adaptation. For training  $\mathbf{D}_{\text{multi}}$ , we use  $T_{\Delta\text{MAX}} = 0.45$  for nuScenes,  $T_{\Delta\text{MAX}} = 0.3$  for Waymo and  $T_{\Delta\text{MAX}} = 0.4$  for Lyft. For testing on different target domains, we match the same  $T_{\Delta\text{MAX}}$  as used in training. Empirically, the sweet spot for frame accumulation across the datasets is around 0.4s which strikes a balance between dynamic and static objects, minimal motion tail overlap, and point cloud processing time.

Table 4.8 indicates that training the same detector on accumulated point clouds performs significantly better on most cross-domain pairs compared to single-frame pre-training. In particular, multi-frame pre-training is highly beneficial for nuScenes-trained detectors, achieving up to 30.5AP<sub>3D</sub> improvement when tested on Lyft/Waymo compared to single-frame pre-training. This is predominantly because accumulating sparse 32-beam point clouds can give variation in scan pattern representations of objects in the training dataset, especially with different ego-vehicle motion (straight roads/turning corners). This can include particular vehicle point representations that are common at different ranges in the Waymo or Lyft dataset, leading to an increased AP<sub>3D</sub> overall. Similarly, we show that multi-frame pre-training can also improve performance of Waymo detectors on Lyft point clouds. Notably, this improvement is not solely attributed to the non-rigid nature of vehicles, as pedestrians likewise achieve a notable boost in AP<sub>3D</sub> across most source-target pairs.

$\mathbf{D}_{\text{multi}}$  however does not always improve cross-domain performance. The ability to leverage multi-frame point cloud training is detector-specific. Point-based IA-SSD shows mixed results when comparing  $\mathbf{D}_{\text{multi}}$  and  $\mathbf{D}_{\text{single}}$ . This might be attributed

**Table 4.8** – Multi-frame vs single-frame detectors on cross-domain data. We show  $AP_{3D}$  gain ( $AP_{\text{multi}} - AP_{\text{single}}$ ) on official validation datasets.

| Detector | Source   | Target   | AP <sub>3D</sub> Gain |       | AP <sub>3D</sub> Gain (%) |        |
|----------|----------|----------|-----------------------|-------|---------------------------|--------|
|          |          |          | Veh.                  | Ped.  | Veh.                      | Ped.   |
| PV-A     | Waymo    | nuScenes | +3.1                  | +2.8  | 20.6%                     | 57.1%  |
|          |          | Lyft     | +0.6                  | +2.9  | 1.3%                      | 8.6%   |
|          | nuScenes | Lyft     | +12.8                 | +8.2  | 58.4%                     | 92.2%  |
|          |          | Waymo    | +21.0                 | +3.3  | 141.9%                    | 154.9% |
|          | Lyft     | nuScenes | +1.3                  | +2.8  | 8.8%                      | 70.7%  |
|          |          | Waymo    | -0.7                  | -3.8  | -1.4%                     | -13.6% |
| PV-C     | Waymo    | nuScenes | +3.3                  | +3.1  | 21.8%                     | 60.4%  |
|          |          | Lyft     | +1.4                  | +1.0  | 2.8%                      | 2.8%   |
|          | nuScenes | Lyft     | +15.8                 | +11.5 | 85.0%                     | 137.1% |
|          |          | Waymo    | +23.0                 | +5.2  | 172.0%                    | 332.5% |
|          | Lyft     | nuScenes | +0.9                  | +2.2  | 6.4%                      | 94.6%  |
|          |          | Waymo    | -0.7                  | -6.2  | -1.5%                     | -23.4% |
| VX-A     | Waymo    | nuScenes | +2.4                  | +1.9  | 17.2%                     | 53.0%  |
|          |          | Lyft     | +1.9                  | +3.5  | 4.0%                      | 11.4%  |
|          | nuScenes | Lyft     | +15.5                 | +11.0 | 111.3%                    | 183.3% |
|          |          | Waymo    | +25.0                 | +7.3  | 348.4%                    | >500%  |
|          | Lyft     | nuScenes | +1.3                  | +2.1  | 9.6%                      | 63.8%  |
|          |          | Waymo    | +1.2                  | +5.9  | 2.5%                      | 20.1%  |
| VX-C     | Waymo    | nuScenes | +2.2                  | +1.8  | 14.4%                     | 36.5%  |
|          |          | Lyft     | +0.0                  | +2.9  | 0.1%                      | 8.4%   |
|          | nuScenes | Lyft     | +20.4                 | +13.8 | 185.5%                    | 236.5% |
|          |          | Waymo    | +30.5                 | +11.5 | >500%                     | >500%  |
|          | Lyft     | nuScenes | +1.0                  | +2.2  | 7.9%                      | 90.0%  |
|          |          | Waymo    | +1.7                  | +7.0  | 3.7%                      | 22.3%  |
| IA-SSD   | Waymo    | nuScenes | +1.1                  | -0.2  | 10.0%                     | -4.6%  |
|          |          | Lyft     | +0.3                  | -7.5  | 0.6%                      | -24.0% |
|          | nuScenes | Lyft     | -2.9                  | -1.3  | -10.6%                    | -13.3% |
|          |          | Waymo    | +5.0                  | +0.2  | 23.0%                     | 6.3%   |
|          | Lyft     | nuScenes | -0.7                  | +0.9  | -8.3%                     | 41.7%  |
|          |          | Waymo    | -1.9                  | -9.3  | -5.5%                     | -52.7% |

**Table 4.9** – Pre-training SECOND-IoU with 4 frames on the Waymo dataset generally outperforms ST3D when testing on nuScenes data. Source and Target are given as Dataset /  $T_{\Delta\text{MAX}}$ .

| Method                     | Source  | Target  | Vehicle           |                  | Pedestrian        |                  |
|----------------------------|---------|---------|-------------------|------------------|-------------------|------------------|
|                            |         |         | AP <sub>BEV</sub> | AP <sub>3D</sub> | AP <sub>BEV</sub> | AP <sub>3D</sub> |
| Source-Only                | W / 0.0 | N / 0.0 | 32.10             | 17.77            | 7.32              | 5.01             |
| ST3D [Yang et al., 2021]   | W / 0.0 | N / 0.0 | 35.90             | 20.20            | 5.75              | 5.11             |
| ST3D++ [Yang et al., 2022] | W / 0.0 | N / 0.0 | 35.73             | <b>20.90</b>     | 12.19             | 8.91             |
| Source-Only                | W / 0.3 | N / 0.3 | <b>40.36</b>      | 18.81            | <b>15.90</b>      | <b>9.05</b>      |

N: nuScenes, W: Waymo

to the learnable center-aware sampling being overfitted to the scan pattern of the training dataset, including how the accumulated points are distributed. There are also mixed results when feeding a Lyft-trained PV-RCNN<sub>multi</sub> with Waymo data, potentially due to the ball-query sampling of accumulated historical points causing a discrepancy in object representations across domains. Due to the overwhelmingly positive improvement of  $\mathbf{D}_{\text{multi}}$  compared to  $\mathbf{D}_{\text{single}}$ , we adopt multi-frame pre-training of all detectors for ensembling in MS3D++. Point cloud accumulation has the added benefit of greater point density at farther ranges, therefore increasing the box proposal recall at farther ranges as well.

### Varying Multi-frame Inference

When using multi-frame detectors for cross-domain data however, a pertinent question arises: Should we prioritize matching  $T_{\Delta\text{MAX}}$  or point density for achieving optimal performance? For instance, accumulating 0.45 seconds of nuScenes data results in around 350k points, while for Waymo, 0.4s corresponds to 850k points, thereby further amplifying the scan pattern domain gap. In Table 4.10, we show that varying  $T_{\Delta\text{MAX}}$  at inference can lead to different results across ranges and classes. When testing a nuScenes-trained PV-C on Lyft, the optimal performance for vehicles at 30-80m is without point cloud accumulation, whereas pedestrians gain up to +10AP<sub>3D</sub> from  $T_{\Delta\text{MAX}} = 0.4$  compared to no accumulation. Overall, it is challenging to select the optimal  $T_{\Delta\text{MAX}}$  as it is difficult to predict how a detector trained on one scan pattern will generalize to another scan pattern.

### Cross-Domain Results for Each Target Domain

Tables 4.11 to 4.13 presents a comprehensive assessment of the overall performance of each detector in a cross-domain setting, evaluated on Waymo, Lyft and nuScenes datasets. We observe that 3D detectors perform reasonably well on close range (0-30m) detection across various target domains, irrespective of their source domain. This can be attributed to the fact that beam divergence across different lidars is less

**Table 4.10** – Varying the number of inference frames for multi-frame PV-C on cross-domain data. The source and target columns are written as dataset /  $T_{\Delta\text{MAX}}$ . Results reported with  $\text{AP}_{3\text{D}}$  at  $\text{IoU}=0.7$  and  $0.5$  for “Vehicle” and “Pedestrian” respectively.

| Source   | Target  | Vehicle     |             |             | Pedestrian  |             |             |
|----------|---------|-------------|-------------|-------------|-------------|-------------|-------------|
|          |         | 0-30        | 30-50       | 50-80       | 0-30        | 30-50       | 50-80       |
| N / 0.45 | L / 0.0 | 71.6        | <b>27.5</b> | <b>3.9</b>  | 35.9        | 4.4         | 0.2         |
|          | L / 0.2 | 71.9        | 25.7        | 3.4         | 43.3        | 10.3        | 1.6         |
|          | L / 0.4 | <b>72.6</b> | 24.9        | 3.1         | <b>45.8</b> | <b>12.3</b> | <b>1.8</b>  |
|          | W / 0.0 | <b>74.6</b> | <b>29.5</b> | <b>5.1</b>  | 14.0        | 4.4         | 0.4         |
|          | W / 0.1 | 74.6        | 28.5        | 4.7         | 13.4        | 4.5         | 0.6         |
|          | W / 0.2 | 74.3        | 28.1        | 4.6         | 14.2        | 4.2         | 0.6         |
|          | W / 0.4 | 73.7        | 27.3        | 4.4         | <b>14.9</b> | <b>4.6</b>  | <b>0.7</b>  |
|          |         |             |             |             |             |             |             |
| W / 0.3  | L / 0.0 | <b>80.3</b> | 55.0        | 15.4        | 51.7        | 23.3        | 7.1         |
|          | L / 0.2 | 79.5        | <b>56.2</b> | <b>17.6</b> | <b>55.7</b> | <b>33.2</b> | <b>20.7</b> |
|          | L / 0.4 | 79.3        | 55.1        | 17.1        | 51.5        | 27.7        | 18.4        |
|          | N / 0.1 | 49.6        | 1.9         | 0.0         | 20.7        | 0.2         | 0.0         |
|          | N / 0.2 | 51.3        | 2.7         | 0.1         | 23.3        | 0.4         | 0.0         |
|          | N / 0.3 | <b>52.0</b> | 3.4         | 0.1         | <b>24.1</b> | 0.8         | 0.1         |
|          | N / 0.4 | 51.7        | <b>3.7</b>  | <b>0.2</b>  | 22.4        | <b>1.1</b>  | <b>0.1</b>  |
|          |         |             |             |             |             |             |             |

N: nuScenes, W: Waymo, L: Lyft

prominent at closer ranges, therefore having less of a domain shift. However, as the range increases, the scan pattern domain gap is exacerbated due to differences in beam divergence, leading to drastic performance degradation in detectors.

Interestingly, no single detector or detection head stands out as the highest performer, indicating that existing state-of-the-art 3D detectors are far from domain generalization on point cloud data. However, we observe that voxel-based backbone networks (PV-RCNN++ and VoxelRCNN) perform better than IA-SSD on all target domains, potentially due to the limitations of IA-SSD’s learnable sampling module in domain generalization.

In Table 4.14, we breakdown the degradation of each detector when adapting from different source domains to a given target domain, compared to training from scratch on that target domain. From this table, we find that the drop in detector performance is more impacted by the source-target domain pairing rather than the detector architecture itself. In general, source-target pairs that are more similar to each other

**Table 4.11** – Evaluation of 3D detectors from different sources on **Waymo** as the target domain. Results reported with  $AP_{3D}$  at  $IoU=0.7$  and  $0.5$  for “Vehicle” and “Pedestrian” respectively.

| Detector | Source   | Vehicle     |             |             | Pedestrian  |             |             |
|----------|----------|-------------|-------------|-------------|-------------|-------------|-------------|
|          |          | 0-30        | 30-50       | 50-80       | 0-30        | 30-50       | 50-80       |
| PV-A     | Waymo    | 90.4        | 68.2        | 40.7        | 67.7        | 56.7        | 37.0        |
| PV-C     | Waymo    | 91.1        | 70.1        | 42.3        | 75.9        | 68.1        | 53.0        |
| VX-A     | Waymo    | 90.8        | 69.8        | 43.6        | 68.9        | 61.0        | 46.6        |
| VX-C     | Waymo    | <b>91.1</b> | <b>71.7</b> | <b>45.5</b> | <b>78.5</b> | <b>71.7</b> | <b>60.3</b> |
| IA-SSD   | Waymo    | 86.7        | 59.7        | 31.3        | 60.9        | 55.4        | 42.6        |
| PV-A     | nuScenes | 73.7        | 28.9        | 4.7         | 12.2        | 3.5         | 0.5         |
|          | Lyft     | 75.7        | 46.9        | 23.1        | 40.3        | 29.3        | 14.9        |
| PV-C     | nuScenes | 74.6        | 29.5        | 5.1         | 14.9        | 4.6         | 0.7         |
|          | Lyft     | 73.9        | 43.7        | 21.4        | 35.2        | 27.7        | 16.6        |
| VX-A     | nuScenes | 71.2        | 21.5        | 3.9         | 21.5        | 2.7         | 0.5         |
|          | Lyft     | <b>75.7</b> | <b>48.1</b> | <b>24.8</b> | 46.1        | 37.0        | 22.0        |
| VX-C     | nuScenes | 71.5        | 21.8        | 4.2         | 27.5        | 6.1         | 1.4         |
|          | Lyft     | 74.0        | 46.8        | 24.5        | <b>46.8</b> | <b>40.8</b> | <b>28.3</b> |
| IA-SSD   | nuScenes | 52.1        | 22.5        | 4.9         | 5.8         | 4.2         | 1.3         |
|          | Lyft     | 59.9        | 31.9        | 12.5        | 23.1        | 19.0        | 10.9        |

such as Lyft and Waymo, which both contain 64-beam point cloud data have less of a performance drop compared to Waymo/Lyft to nuScenes.

To delve deeper into the potential reasons for the significant performance decrease, we visualize the predictions and present a summary of several key findings in Fig. 4.9. Despite dense observations of objects, pre-trained detectors may struggle to obtain detections, indicating that higher point density alone is insufficient for effective cross-domain detection. We observe that a nuScenes and lyft detector misses many densely observed vehicles in Fig. 4.9 (1,2) potentially due to weather artefacts exhibiting different point distributions across lidar types. For Fig. 4.9 (2), it could also be attributed to the peculiar labelling guidelines of Lyft where labels do not often include non-traffic participants, therefore causing a Lyft-trained detector to have many missed detections of parked cars on other datasets.

In Fig. 4.9 (3), we illustrate a prominent issue when adapting from dense to sparse lidars. In the nuScenes dataset, the 32-beam lidar’s large vertical angular resolution



**Table 4.12** – Evaluation of 3D detectors from different sources on **Lyft** as the target domain. Results reported with  $AP_{3D}$  at IoU=0.7 and 0.5 for “Vehicle” and “Pedestrian” respectively.

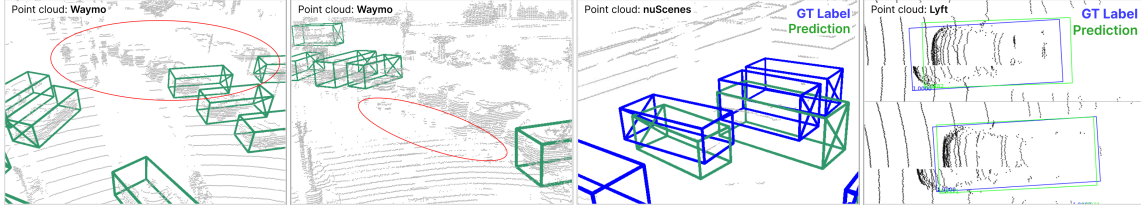
| Detector | Source   | Vehicle     |             |             | Pedestrian  |             |             |
|----------|----------|-------------|-------------|-------------|-------------|-------------|-------------|
|          |          | 0-30        | 30-50       | 50-80       | 0-30        | 30-50       | 50-80       |
| PV-A     | Lyft     | <b>90.3</b> | <b>73.3</b> | <b>29.0</b> | 57.0        | 33.4        | <b>19.2</b> |
| PV-C     | Lyft     | 88.2        | 71.1        | 27.8        | 53.9        | 33.1        | 17.5        |
| VX-A     | Lyft     | 88.0        | 70.7        | 26.2        | 57.8        | <b>36.6</b> | 18.9        |
| VX-C     | Lyft     | 88.0        | 70.4        | 26.3        | <b>59.5</b> | 34.4        | 18.9        |
| IA-SSD   | Lyft     | 82.6        | 58.7        | 17.6        | 28.9        | 18.9        | 12.2        |
| PV-A     | nuScenes | 73.1        | 27.7        | 3.8         | 39.1        | 11.0        | 1.4         |
|          | Waymo    | 78.8        | 55.8        | 17.2        | 57.1        | <b>34.0</b> | 18.2        |
| PV-C     | nuScenes | 71.6        | 27.5        | 3.9         | 45.8        | 12.3        | 1.8         |
|          | Waymo    | <b>79.5</b> | <b>56.2</b> | <b>17.6</b> | 55.7        | 33.2        | <b>20.7</b> |
| VX-A     | nuScenes | 68.0        | 18.3        | 2.2         | 39.6        | 9.9         | 1.6         |
|          | Waymo    | 78.0        | 53.9        | 16.5        | 55.4        | 30.9        | 16.2        |
| VX-C     | nuScenes | 69.9        | 22.1        | 2.4         | 45.5        | 11.9        | 1.6         |
|          | Waymo    | 75.9        | 54.1        | 16.0        | <b>57.8</b> | 33.6        | 18.6        |
| IA-SSD   | nuScenes | 52.0        | 19.4        | 3.1         | 17.3        | 6.2         | 2.0         |
|          | Waymo    | 74.8        | 48.9        | 12.0        | 54.3        | 28.7        | 11.1        |

results in sparse information for vehicles beyond 30m, making accurate height localization challenging. While many existing works focus on the number of beams, we argue that the scan pattern domain gap is more related to the angular resolution. A 32-beam lidar with small vertical angular resolution can also result in dense vehicle appearances that are similar the 64-beam lidar in Waymo, albeit with a smaller vertical field of view. Additionally, the Lyft dataset does not appear to perform ego-motion compensation, resulting in misaligned lidar points during a single revolution due to vehicle ego-motion[Berrio et al., 2022]. This introduces challenges for accurate detection and labelling consistency as depicted in Fig. 4.9 (4). In general, these various domain idiosyncrasies substantially affects predicted confidence scores. Much of the previous works in 3D detectors place a large importance on IoU-aware refinement to enable the confidence score to reflect the accuracy of box localization [Hu et al., 2022, Yang et al., 2021, Shi et al., 2020, Yin et al., 2021, Jiang et al., 2018] because this scoring itself has a large impact on the AP metric. However, in the cross-domain setting, this relationship between confidence score and object localization becomes

**Table 4.13** – Evaluation of 3D detectors from different sources on **nuScenes** as the target domain. Results reported with  $AP_{3D}$  at  $IoU=0.7$  and  $0.5$  for “Vehicle” and “Pedestrian” respectively.

| Detector | Source   | Vehicle     |             |            | Pedestrian  |             |            |
|----------|----------|-------------|-------------|------------|-------------|-------------|------------|
|          |          | 0-30        | 30-50       | 50-80      | 0-30        | 30-50       | 50-80      |
| PV-A     | nuScenes | <b>72.6</b> | <b>20.8</b> | <b>2.6</b> | <b>44.0</b> | 13.8        | 1.4        |
| PV-C     | nuScenes | 68.9        | 18.9        | 2.2        | 42.2        | <b>14.8</b> | 1.4        |
| VX-A     | nuScenes | 69.8        | 17.2        | 2.1        | 42.7        | 12.3        | 0.9        |
| VX-C     | nuScenes | 66.6        | 17.5        | 1.9        | 43.2        | 14.8        | <b>1.7</b> |
| IA-SSD   | nuScenes | 57.0        | 10.2        | 0.8        | 31.5        | 8.9         | 0.7        |
| PV-A     | Waymo    | 51.0        | 3.2         | 0.1        | 22.2        | 0.8         | 0.0        |
|          | Lyft     | 44.9        | 2.5         | 0.1        | 19.1        | 0.8         | 0.0        |
| PV-C     | Waymo    | <b>52.0</b> | <b>3.4</b>  | <b>0.1</b> | <b>24.1</b> | <b>0.8</b>  | <b>0.1</b> |
|          | Lyft     | 40.8        | 2.0         | 0.0        | 13.4        | 0.3         | 0.1        |
| VX-A     | Waymo    | 46.1        | 2.6         | 0.0        | 16.4        | 0.3         | 0.0        |
|          | Lyft     | 42.6        | 1.8         | 0.1        | 16.0        | 0.3         | 0.0        |
| VX-C     | Waymo    | 48.6        | 2.8         | 0.1        | 19.0        | 0.8         | 0.0        |
|          | Lyft     | 41.3        | 1.7         | 0.1        | 14.0        | 0.2         | 0.0        |
| IA-SSD   | Waymo    | 35.0        | 0.7         | 0.0        | 12.2        | 0.1         | 0.0        |
|          | Lyft     | 23.0        | 0.8         | 0.0        | 8.5         | 0.5         | 0.0        |

misaligned and unreliable, therefore further affecting  $AP_{3D}$ .



**Figure 4.9** – 3D Detectors tested on cross-domain data encounter various issues. From left to right: (1) Scan pattern discrepancy causes densely observed vehicles and pedestrians to be missed, (2) Weather artefacts such as reflections on the ground due to rain, appear different across various scan patterns, causing missed detections or false positives, (3) Sparse lidar is challenging for box height estimation, (4) Poor (or lack of) motion compensation for the ego-vehicle causes artefacts, inaccurate labels, and therefore, prediction “errors”. In the image, the upper predicted box may not meet the  $IoU=0.7$  threshold for a true positive detection. Predictions for image (1-4) were made with PV-RCNN++ (centerhead) and trained on nuScenes (1), Lyft (2) and Waymo (3-4). We elaborate on these in Section 4.3.3.

Another aspect of the domain gap is due to labelling discrepancies. For example, Waymo labelling guidelines specify that side mirrors of vehicles are to be included

**Table 4.14** – Performance reduction when comparing pre-trained detectors on different source domains  $AP_{CD}$  to training from scratch on target domain data  $AP_{oracle}$  where  $AP\% \text{ Reduction} = \frac{AP_{CD} - AP_{oracle}}{AP_{oracle}} \times 100$ .

| Source   | Target   | Detector | % $AP_{3D}$ Reduction |        |
|----------|----------|----------|-----------------------|--------|
|          |          |          | Veh.                  | Ped.   |
| Waymo    | nuScenes | PV-A     | -69.9%                | -80.8% |
|          |          | PV-C     | -66.8%                | -78.0% |
|          |          | VX-A     | -72.4%                | -86.2% |
|          |          | VX-C     | -69.2%                | -82.9% |
|          |          | IA-SSD   | -77.3%                | -86.6% |
| Lyft     | nuScenes | PV-A     | -74.3%                | -83.1% |
|          |          | PV-C     | -76.1%                | -87.5% |
|          |          | VX-A     | -74.5%                | -86.4% |
|          |          | VX-C     | -75.3%                | -88.8% |
|          |          | IA-SSD   | -83.1%                | -88.9% |
| Lyft     | Waymo    | PV-A     | -30.3%                | -49.5% |
|          |          | PV-C     | -35.3%                | -60.6% |
|          |          | VX-A     | -30.3%                | -41.7% |
|          |          | VX-C     | -33.3%                | -45.5% |
|          |          | IA-SSD   | -45.8%                | -67.4% |
| nuScenes | Waymo    | PV-A     | -54.9%                | -91.5% |
|          |          | PV-C     | -54.6%                | -90.8% |
|          |          | VX-A     | -60.6%                | -87.8% |
|          |          | VX-C     | -60.7%                | -84.7% |
|          |          | IA-SSD   | -62.1%                | -93.3% |
| nuScenes | Lyft     | PV-A     | -56.1%                | -63.6% |
|          |          | PV-C     | -55.4%                | -55.9% |
|          |          | VX-A     | -62.8%                | -65.4% |
|          |          | VX-C     | -60.1%                | -60.2% |
|          |          | IA-SSD   | -62.1%                | -63.6% |
| Waymo    | Lyft     | PV-A     | -25.7%                | -1.0%  |
|          |          | PV-C     | -22.5%                | 7.2%   |
|          |          | VX-A     | -24.1%                | -11.4% |
|          |          | VX-C     | -25.3%                | -2.2%  |
|          |          | IA-SSD   | -19.3%                | 43.4%  |

in the label whereas nuScenes, Lyft and KITTI do not include side mirrors. This discrepancy can considerably affect the IoU calculation when comparing predictions with ground-truth boxes for detector evaluation. Additionally, there are inconsistent category definitions. The "Vehicle" class of the Waymo dataset includes cars, trucks,

buses, construction vehicles, and even motorcycles whereas nuScenes and Lyft classify them separately. This is an issue when adapting Waymo detectors to KITTI for example, as KITTI lacks specific categories for motorcycles and construction vehicles, leading to numerous "false positive" detections when using a detector trained on the "Vehicle" category from Waymo. Furthermore, there are labelling inconsistencies within the "Cyclist" category across datasets. Waymo labels "Cyclists" only if there is a rider, whereas Lyft and nuScenes have no "Cyclist" category but rather an object-based, "bicycle" category that includes parked bicycles without a rider.

#### 4.3.4 Conclusion

In this section we demonstrated that the scan pattern discrepancy is a large component of the domain gap between different point cloud datasets. Simply accumulating more frames in training allows a 3D detector to see more variation in object scan pattern representations, resulting in a significant increase in generalization to various target domains. From our analysis, we find that the effectiveness of 3D detection on a given target domain is primarily influenced by the level of correlation between the source and target dataset. Differences in scan patterns can cause 3D detectors trained on low density lidar data to overlook densely observed objects within a high density lidar point cloud. This oversight occurs because objects with such point density were absent from the training dataset. Moreover, accurate evaluation is challenging due to differences in labelling guidelines such as the inclusion of side mirrors or class definitions. To complicate matters further, data is processed differently across datasets. For instance, the Lyft dataset does not do motion compensation, therefore leading to instances where valid positive predictions are mistakenly treated as false positives.

We use the insights from this section as a starting point for improving the MS3D framework. Firstly, we use multi-frame pre-trained 3D detectors for our ensemble instead of single-frame, due to their significant improvement over single-frame pre-training shown in Section 4.3.3. To properly utilize multi-frame pre-trained detectors, we generate multiple bounding box sets by varying  $T_{\Delta\text{MAX}}$  at inference to consider

all possible scan patterns which we use in our ensemble. Additionally, we employ a weighing strategy that assigns higher weight to detectors that were pre-trained on the lidar density most similar to the target domain. We avoid weighing individual detectors above another based on our observation in Table 4.14, where the performance of detectors on a new target domain is primarily influenced by its source dataset rather than its specific architecture.

## 4.4 MS3D++

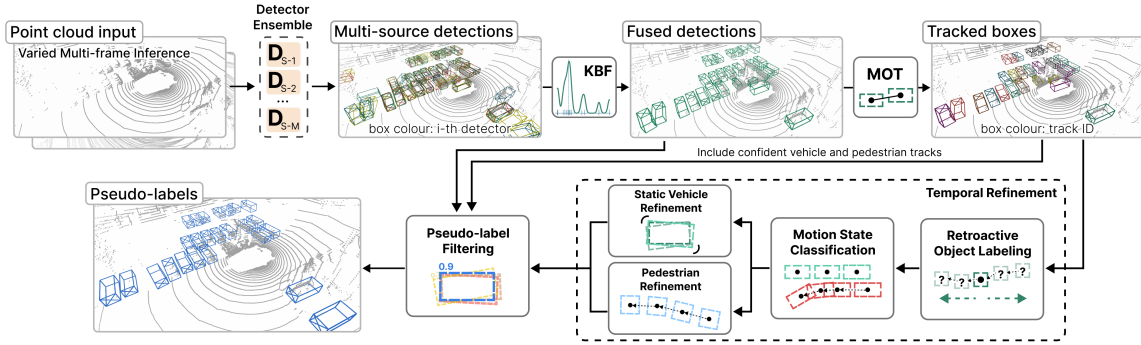
### 4.4.1 Overview

In this section we incorporate insights from Section 4.3 to build upon MS3D. Our improved framework, MS3D++, introduces auto-labelling for the pedestrian class, multi-frame pre-trained detector ensembling and iterative label refinement.

MS3D++ is built upon three fundamental observations. Firstly, similar to MS3D, we recognize that different detector architectures trained on different point cloud datasets possess distinct expertise [Tsai et al., 2023b]. For instance, a Waymo-trained detector with smaller voxel sizes may be better suited for a newly released high density lidar compared to a nuScenes-trained detector. Using an ensemble of experts allows us to obtain a more robust set of initial pseudo-labels on an unfamiliar target domain. Secondly, many works have demonstrated the benefits of short-sequence point cloud accumulation when trained and tested on the same lidar setup [Caesar et al., 2020, Yang et al., 2021]. Our study reveals that multi-frame pre-trained detectors can also bring significant improvements when tested in a cross-domain setting. This allows us to achieve increased detection of sparse and far-range objects, substantially boosting the performance of our domain adaptation baseline. Lastly, to address the issue of unreliable detector confidence scores in a cross-domain context, we use temporal information which has been proven effective in 3D box refinement [Qi et al., 2021, Ma et al., 2023, Fan et al., 2023a] for a supervised setting. Leveraging temporal information allows us to accurately distinguish true positive class predictions and

accurately refine box localization.

MS3D++ is designed to achieve two primary goals in self-training: First, maximise precision through accurately localized and categorized labels to avoid reinforcing false positives. Secondly, we aim to maximise recall to ensure sufficiently accurately labelled samples for improving model performance without compromising precision. MS3D++ can be broken down into three key steps: (1) Increasing box proposal recall through ensembling and tracking; (2) Temporal refinement to improve precision; (3) Multi-round self-training to boost recall while maintaining precision. Fig. 4.10 illustrates our proposed MS3D++ framework which will be introduced in the following sub-sections.



**Figure 4.10** – Our multi-source self-training framework, MS3D++. Given a set of  $M$  pre-trained, multi-frame 3D detectors from multiple source domains  $\mathbf{D}_{S,m}$  where  $m = 1, 2, \dots, M$ , we generate predictions for a varying number of accumulated point cloud frames which are fused with KBF and tracked. Our temporal refinement uses object characteristics to ascertain class labels and refine bounding box localization. The set of pseudo-labels is iteratively improved with each update, utilizing the re-trained detectors from the previous round in its ensemble.

In summary, MS3D++ significantly improves upon MS3D by (1) proposing a robust pseudo-labelling strategy for pedestrians, a non-rigid body class; (2) introducing Varied Multi-Frame Inference to effectively use multi-frame pre-trained detectors for both vehicles and pedestrians in a cross-domain setting; (3) introducing a multi-round self-training framework to increase pseudo-label recall while preserving precision; (4) surpassing the performance of our previous work, therefore establishing a new state-of-the-art for 3D detectors on unseen target domains without requiring any manually labelled data.

### 4.4.2 Method

The goal of MS3D++ is to use multiple pre-trained detectors from various source domains to automatically label an unlabelled target domain. We use multiple pre-trained detectors with their predictions denoted in this section as the set of bounding boxes  $\{B_m^i\}_{m=0}^M$  representing the  $m$ -th prediction set on the  $i$ -th frame of the unlabelled target dataset for a total of  $M$  prediction sets. Each test-time augmentation or Varied Multi-Frame Inference (VMFI) (Section 4.3.3) are also included in the  $m$ -th bounding box set. We consider a sequence of target domain point clouds  $\{P_i \in \mathbf{R}^{n_i \times C}\}$ ,  $i = 1, 2, \dots, N$  with the  $i$ -th frame point cloud  $P_i$  consisting of  $n_i$  points with  $C$  channels for each point, for  $N$  total frames. For the cross-domain setting, we only consider the channels  $XYZT$  where  $T$  is the time delta in seconds from the current frame at time  $t = 0$ . For MS3D++ we assume known sensor poses  $\{E_i = [R_i|t_i] \in \mathbf{R}^{3 \times 4}\}$ ,  $i = 1, 2, \dots, N$  at each frame in world-coordinates for ego-motion compensation.

#### Kernel Density Estimation Box Fusion (KBF)

In MS3D, we introduced KBF,  $\kappa(\cdot)$ , for effective fusion of multiple predicted boxes which uses Kernel Density Estimation (KDE) for fusing the centroid  $(c_x, c_y, c_z)$ , dimensions  $(l, w, h)$ , heading  $\theta$  and a confidence score  $s$  of a set of boxes separately. KDE positions a kernel  $K$  at each data point  $x_i$  and sums them together to get a probability density estimate  $\hat{f}(x)$ , shown in Eq. (4.1). A weight  $w_i$  for each data point and a kernel bandwidth  $h$  can be adjusted for fine-tuning the PDF estimate. KBF remains a core component for all components of MS3D++ with the additional use of  $w_i$  for weighing of bounding box sets in Section 4.4.2 using insights from Section 4.3.

#### Increasing Box Proposal Recall

A primary challenge for self-training in unsupervised domain adaptation is the lack of strong pre-trained detectors for pseudo-label generation. For example, as shown

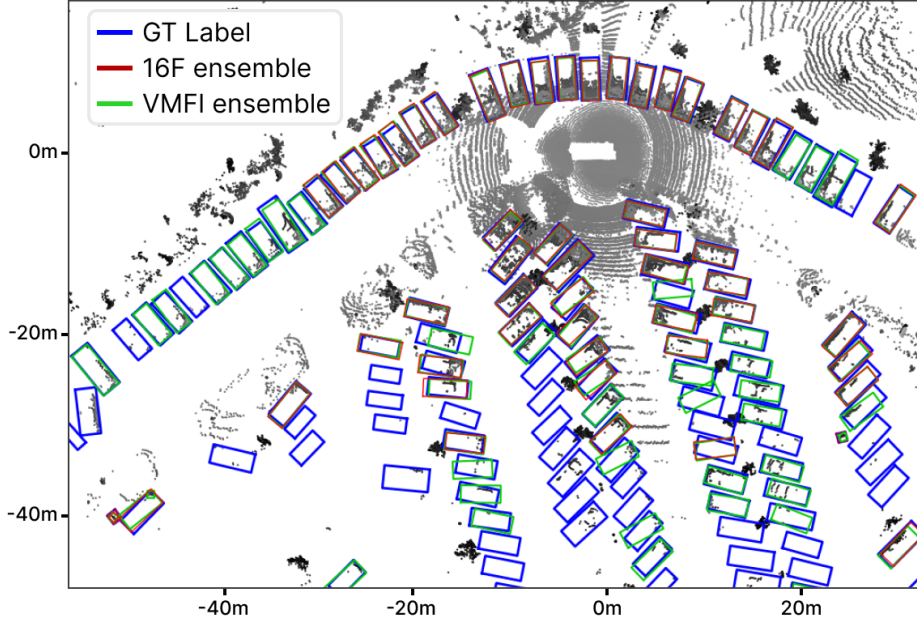
in Section 4.3, training PV-RCNN++ [Shi et al., 2023] and VoxelRCNN [Deng et al., 2020] on Waymo and testing on nuScenes data can result in up to 70% decrease in Vehicle detection rate compared to training the same models from scratch with nuScenes data. This weak baseline can lead to a large number of missed detections and misaligned confidence scores. To achieve effective pseudo-labelling, it becomes crucial to maximise recall by increasing the number of object proposals, thereby increasing our ability to capture all objects within the scene (i.e., reducing false negatives). In 3D detector methods, it is also common to output a high number of predictions before culling down with NMS [Yin et al., 2021] or a second-stage refinement [Chen et al., 2022, Shi et al., 2020]. Following this, MS3D++ aims to first increase box proposal recall with detector ensembling and tracking for inter/extrapolation.

**Varied Multi-Frame Inference.** In Fig. 4.9 from Section 4.3, we illustrated that certain scan pattern differences between datasets can cause pre-trained detectors to incorrectly predict a high number of false negatives (i.e., missed detections). To boost the recall in our initial box proposals, we first use the insights from Table 4.8 and adopt multi-frame pre-trained detectors to raise the baseline performance for unseen target domains.

Thereafter, following the findings of Table 4.10, we employ Varied Multi-Frame Inference (VMFI) which varies the number of accumulated frames at inference. With VMFI, we take the guess-work out of matching scan patterns, and simply include all variations in the ensemble. In contrast to MS3D, we remove the need for 16-frame detection sets as we show in Fig. 4.11 that VMFI can obtain detections for a much farther range of objects. Finally, we also generate detections for each VMFI with Test-Time Augmentation (TTA) for each detector. From these detection sets, we use KBF to obtain a single set of box proposals for each  $i$ -th frame  $B_{\text{KBF}}^i = \kappa(\{B_{m(\text{VMFI})}^i, B_{m(\text{TTA})}^i\})$ .

**Detector Weighing.** From Section 4.3.3, we showed that each detector has varying performance on different target domains. However due to the idiosyncrasies of particular source-target domain pairs, it is challenging to build an intuition of the optimal detector for specific target domains. Therefore to alleviate the strain on having to select optimal detectors, we simply include all the detectors into our ensemble with





**Figure 4.11** – Ensembling with Varied Multi-frame Inference (VMFI) obtains an extended range of detections compared to using 16-frames for inference. This could be attributed to the exacerbated scan pattern domain gap when feeding 16-frames to a pre-trained multi-frame detector of a different source domain.

the exception of IA-SSD due to its low cross-domain performance. We show in Section 4.4.3 that KBF is able to successfully fuse the different detectors with minimal loss of performance. Furthermore, we devise a weighing strategy guided by simple intuition: assigning higher weight to detectors that were pre-trained on the lidar density most similar to the target domain. We avoid weighing individual detectors above another based on our observation in Table 4.14, where the performance of detectors on a new target domain is primarily influenced by its source dataset rather than its specific architecture. This approach also enables visual assessment of the ensembling performance, eliminating the need for a manually labelled validation set to determine the optimal combinations and weights.

**Multi-Object Tracking.** Due to the misaligned confidence scores of pre-trained detectors in a cross-domain setting, there are often highly accurate detections with scores as low as 0.3. For example, with Waymo as the target domain, many accurate pedestrian predictions from nuScenes/Lyft detectors have scores in the range  $[0.3, 0.6]$ . The challenge however, is separating accurate pedestrian boxes from the myriad of

false positive detections that also exist within that range which we further explain in Section 4.4.2. For this stage, to prevent premature elimination of true positives, we run a Kalman Filter-based tracker [Pang et al., 2021] at low score thresholds for pedestrians ( $T_p$ ), and vehicles ( $T_{v,all}$  and  $T_{v,static}$ ) on  $B_{KBF}$ . We generate two separate tracks for vehicles at different thresholds: (1)  $T_{v,all}$  tracks both dynamic and static vehicles, and (2)  $T_{v,static}$  focuses on static vehicle tracking with a high BEV IoU threshold for track association. Following our prior work, we classify all tracks into static or dynamic based on pre-defined thresholds of begin-to-end distance and box centre variance. A high IoU for  $T_{v,static}$  is critical in minimising false positive tracks of parked vehicles that are classified as "dynamic" due to inaccurate box localization. While the tracking step includes both dynamic and static vehicles, extrapolation of static vehicles is simpler and less prone to false positives. For instance, when a tracked dynamic vehicle turns a corner, becomes occluded, or changes lanes at a far range, the extrapolated track can easily deviate significantly from the actual path or suffer from poor box localization. To address this, we use interpolation and extrapolations only for confident dynamic tracks by setting a higher score threshold for  $T_{v,all}$  than  $T_{v,static}$ .

### Temporal Refinement

Due to misaligned confidence scores in a cross-domain setting, it is often challenging to decide if a predicted box is a true or a false positive for filtering of box proposals. Simply setting a confidence score threshold often proves inadequate as high confidence scores may not correlate with an accurate box localization or object class. Furthermore, selecting a threshold involves balancing between data variation and quality. A high threshold may reduce false positives, but limit the data variation and as a result, model improvement. Conversely, a low threshold may increase data variation but lead to more false positives and poorly localized boxes, resulting in the accumulation of errors with each self-training round. In this section, we explain how MS3D++ leverages temporal consistency and object characteristics to provide a more effective solution for refining box proposals to obtain increased pseudo-label precision. For the

final pseudo-label set, we define a positive score threshold  $s_{\text{pos}}$  such that every box with a score greater than  $s_{\text{pos}}$  is considered a pseudo-label.

**Retroactive object labelling.** In human perception, identifying certain objects from a distance or when occluded can be challenging. To address this, our eyes continuously track the object until its class can be confidently determined. Once recognized, we retroactively apply this object class label to all past and future observations of the same object. In the context of MS3D++, we adopt a similar approach, where we track detections with a low threshold to obtain potential class proposals. These tracks are not immediately treated as pseudo-labels. We withhold the use of the tracks as labels until we confidently recognize the object class, through a number of high confidence detections (i.e.,  $> s_{\text{pos}}$ ). Practically, since pseudo-labelling is done in an offline setting, we simply filter every track to ensure they contain at least  $N_{\text{pos}}$  confident detections above score  $s_{\text{pos}}$  to obtain the refined tracks:  $T_{\text{v},\text{all}(\text{refined})}$ ,  $T_{\text{v},\text{static}(\text{refined})}$  and  $T_{\text{p}(\text{refined})}$ . We note that this approach is mostly effective for  $T_{\text{v},\text{static}}$  and  $T_{\text{p}}$  which further elaborate on.

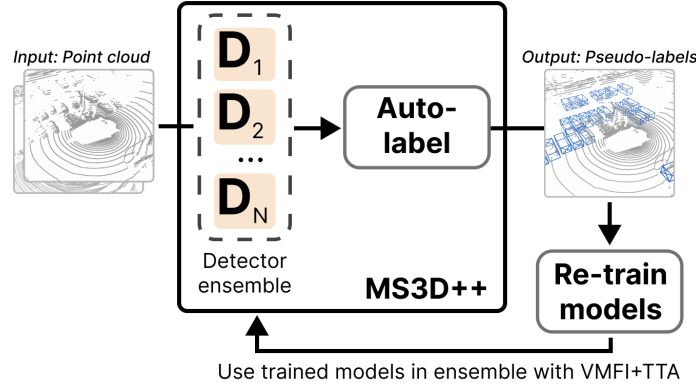
**Static vehicle refinement.** For dynamic vehicle tracking, although interpolated or extrapolated tracks may provide good centre prediction, the tracked box is impractical as a pseudo-label if its orientation is incorrect. Due to the elongated shape of vehicles, heading errors can easily cause many of the vehicle points to fall outside the bounding box, therefore resulting in poorly localized pseudo-labels that affect the self-training process. This particularly affects the extrapolation of sparse vehicle observations at farther ranges. Stationary vehicles on the other hand are much easier to handle. For  $T_{\text{v},\text{static}}$ , we simply require a few strong detections of the object to be able to use it for refining all tracked observations of the stationary vehicle. Therefore, to increase the precision of vehicle labelling, we focus on scenes with static vehicles. Additionally, because vehicles are rigid body objects, dynamic vehicles share the same appearance as static vehicles. This allows detectors trained on accurately labelled static vehicles to effectively handle dynamic vehicles as well. Following our previous work, we adopt KBF for fusing the box predictions of  $H$  historical observation frames,  $B_{\text{v},\text{static}}^{i,j} = \kappa(\{T_{\text{v},\text{static}}\}_{i-H}^i)$ , to obtain a static box  $B_{\text{v},\text{static}}^{i,j}$  for the  $j$ -th vehicle in the

$i$ -th frame due to potential localization errors [Tsai et al., 2023b, Caesar et al., 2020].

**Pedestrian refinement.** Pre-trained detectors commonly misclassify pole-like objects as “Pedestrians”. The confidence scores of such predictions fall within a similar range (around 0.3-0.6) to actual “Pedestrians” in the scene. Since pedestrians are non-rigid body objects that are frequently in motion, the static refinement approach used for “Vehicles” is not directly applicable. However, we leverage this unique characteristic of pedestrians to our advantage. While pole-like objects and pedestrians may appear similar when static, the distinction becomes clearer when the object starts moving. Therefore, we classify the motion of tracks from  $T_{p(\text{refined})}$  and use the dynamic tracks  $T_{p(\text{refined}),\text{dyn}}$  as pseudo-labels. This strategy is more practical than dynamic vehicle tracking due to the geometric property of pedestrians. Unlike vehicles, the approximate symmetry of pedestrians from a BEV perspective allows for relatively high IoU with ground truth even with incorrect heading predictions. As a result, obtaining reasonably localized pedestrian pseudo-labels with retroactive object labelling on  $T_{p(\text{refined}),\text{dyn}}$  is considerably more feasible compared to dynamic vehicles. Nonetheless, pedestrian labelling need not solely be limited to dynamic pedestrians. After a few rounds of self-training, explained in Section 4.4.2, detector confidence scores become better calibrated, making it easy to distinguish true pedestrians by thresholding. At this point, retroactive object labelling can be used for both static and dynamic pedestrians.

### Pseudo-Label Filtering

To obtain our final set of pseudo-labels, we add the following boxes to each  $i$ -th frame and apply NMS for filtering: (1) VMFI Boxes  $B_{\text{KBF}}^i$ , (2) Static vehicle boxes  $B_{\text{v},\text{static}}^i$ , (3) Tracked vehicle boxes  $T_{\text{v},\text{all}(\text{refined})}^i$ , (4) Dynamic pedestrian tracked boxes  $T_{\text{p}(\text{refined}),\text{dyn}}^i$ . Following this, we remove any pseudo-labels with less than 1 point in the box for a single-frame point cloud and only retain pseudo-labels above a score threshold  $s \geq s_{\text{pos}}$ .



**Figure 4.12** – Iterative label refinement of MS3D++: In the first round of self-training, we employ a range of detectors from various source domains. The pseudo-labels are used to re-train the 3D detectors with multi-frame accumulation. We use the re-trained detectors to create a new ensemble for the next round.

### Iterative Label Refinement

Iterative improvement through multiple self-training rounds has been shown to be a promising technique [You et al., 2022, Zhang et al., 2023]. With each self-training round, the re-trained detector has increased detection range and improved alignment between score and class/localization. This increases the confidence scores for true predictions allowing easy distinction from false positives. The improvement in score alignment also extends to higher confidence detections at farther ranges, allowing for more precise tracking and refinement of far-range static vehicles and dynamic pedestrians for the subsequent pseudo-label set. The iterative label refinement for MS3D++ is shown in Fig. 4.12.

For our first round of self-training, we use a large ensemble of pre-trained detectors from various source domains for pseudo-label generation. With this pseudo-label set, we re-train a set of detectors with short-sequence, multi-frame accumulation. At this round, VMFI is not required as the re-trained models are trained on the target domain’s multi-frame scan pattern. To mitigate confirmation bias (i.e., the reinforcement of false positives) during each round of self-training, we set high  $s_{\text{pos}}$  and  $N_{\text{pos}}$  thresholds for all classes. Given the ego-vehicle motion in the scene, driving past tracked objects often result in a few high confidence detections at some point in the driving sequence. This in turn enables our static vehicle and dynamic pedestrian

refinement to extrapolate and interpolate pseudo-labels to all tracked observations thereby increasing recall. This conservative approach for selecting  $s_{\text{pos}}$  and  $N_{\text{pos}}$  enables the gradual increase of pseudo-label quality and recall while maintaining high precision.

### 4.4.3 Experiments

#### Setup

**Datasets and evaluation metrics** For our experiments we use three datasets: Waymo, nuScenes and Lyft, as introduced in Section 4.3 where each large-scale dataset is varied in lidar scan pattern, weather and geography. The datasets are summarised in Table 4.7. Due to label inconsistency in the cyclist category across domains as mentioned in Section 4.3.3, we focus on the “Vehicle” and “Pedestrian” class. For nuScenes and Lyft, our “Vehicle” supercategory includes the car, truck and bus subcategories. Following existing works [Yang et al., 2021, Hu et al., 2023, Wei et al., 2022], we use the KITTI average precision evaluation metric over 40 recall positions in BEV and 3D, denoted as  $\text{AP}_{\text{BEV}}$  and  $\text{AP}_{\text{3D}}$ . For all tables we report the AP on the official validation split of each dataset at IoU thresholds of 0.7 for “Vehicle” and 0.5 for “Pedestrian”.

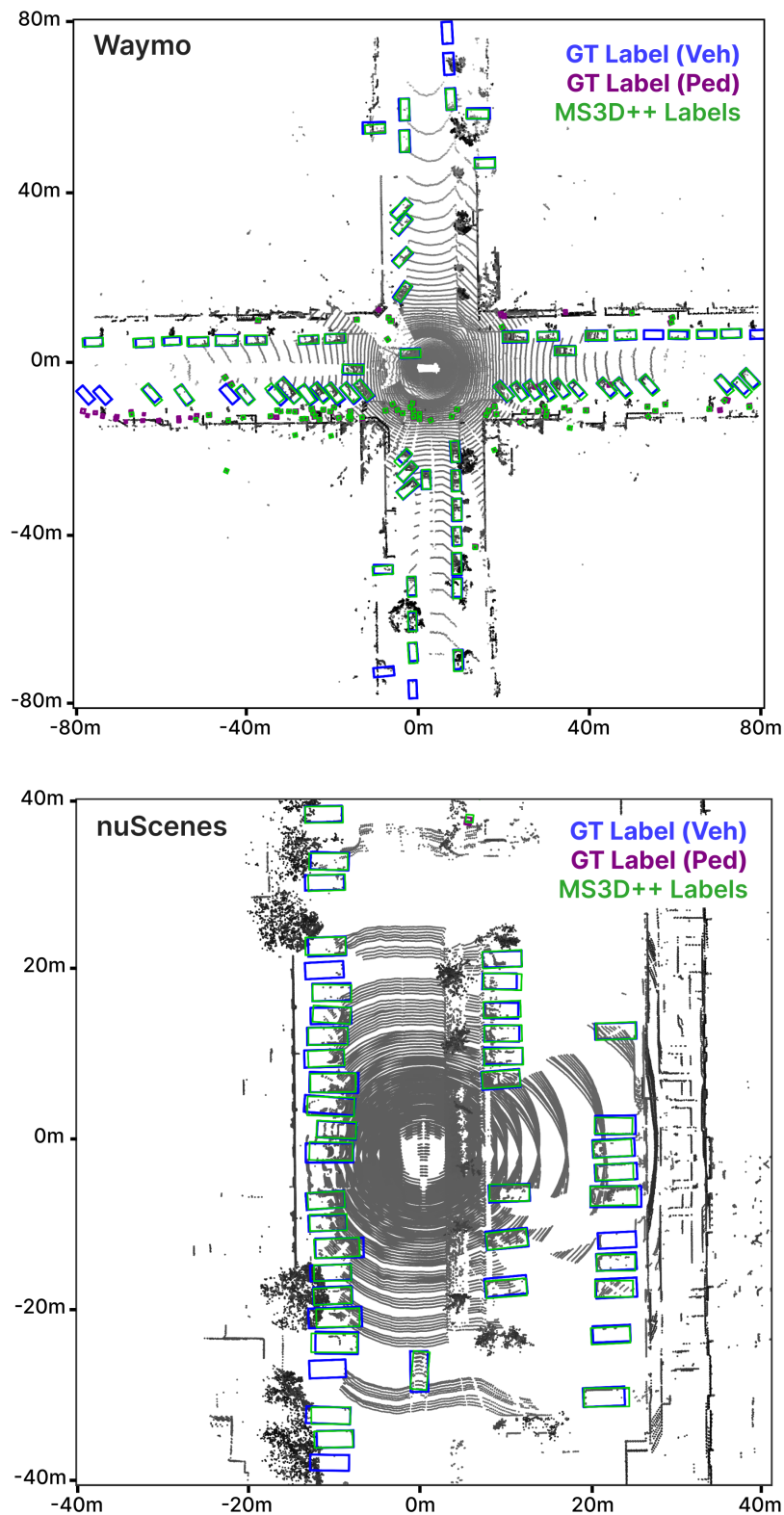
**Comparison methods** We compare MS3D++ with the following methods: (1) Source-only, directly applying the detector on the target domain without any domain adaptation; (2) ST3D [Yang et al., 2021] serves as a single-source, self-training baseline, using a single detector without temporal refinement for pseudo-labelling; (3) MS3D [Tsai et al., 2023b], our previous work which used SECOND and Centerpoint for detector ensembling with single-round self-training; (4) GT-trained, uses manually annotated ground truth labels to train the detector, providing an upper bound comparison if we were to achieve perfect pseudo-labels. For nuScenes as target domain, we further include other methods that report results for the Waymo→nuScenes domain adaptation task: LD (with ST3D) [Wei et al., 2022], DTS [Hu et al., 2023] and REDB [Chen et al., 2023]. Similar to [Tsai et al., 2023b], GT-trained presents an

apples-to-apples comparison of our pseudo-labels to GT labels. Between GT-trained and MS3Ds, the following are kept the same: data augmentation, dataset size, voxel size, anchor box sizes, number of epochs, batch size and learning rate. This comparison contrasts with the typically reported upper bound “Oracle” supervised benchmark with optimized hyperparameters [Yang et al., 2021, Hu et al., 2023]. For all methods, we train with 4-frame accumulation on Waymo ( $T_{\Delta\text{MAX}} = 0.3$ ), 3-frame on Lyft ( $T_{\Delta\text{MAX}} = 0.4$ ), and 10-frame on nuScenes ( $T_{\Delta\text{MAX}} = 0.45$ ) unless specified otherwise in the table.

**Implementation details** We use the pre-trained detectors: PV-A, PV-C, VX-A and VX-C as specified in Section 4.3.2. Similar to [Tsai et al., 2023b], we sort Waymo and nuScenes by number of vehicles and pedestrians per scene, selecting the top 190 and 193 scenes, respectively. This selection allows us to focus on the static vehicle refinement, as scenes with many cars often represent parking lots. For an unlabelled target domain, a similar approach can be adopted by intentionally collecting data from scenes with carparks and densely populated pedestrian areas. The initial round of pseudo-labels are generated with four pre-trained detectors from various source domains and used to train VX-A and VX-C with multi-frame accumulation. For subsequent self-training rounds, we only use VX-A and VX-C in the ensemble with TTA. This choice is due to the distinct box proposal approaches of anchorhead [Yan et al., 2018, He et al., 2017] and centerhead [Yin et al., 2021], which provide more varied object proposals. For pre-trained detector KBF weighing on Waymo as the target domain, we assign higher weights to Lyft detectors over nuScenes detectors due to Lyft having 64-beam point clouds. Conversely, when Lyft is the target domain, we assign higher weights to Waymo detectors compared to nuScenes detectors. Lastly, for nuScenes as the target domain, we weigh Waymo and Lyft detectors equally. Further implementation details can be found in our open-sourced code.

## Results

In Tables 4.15 and 4.16, we demonstrate that training VX-A and VX-C with MS3D++ pseudo-labels yields an  $\text{AP}_{\text{BEV}}$  that outperforms all existing works and achieves re-



**Figure 4.13** – Visualization of MS3D++ pseudo-labels on Waymo and nuScenes point clouds. We show that we can obtain well localized boxes for sparse observations on both datasets. For a 64-beam lidar (Waymo), we show that MS3D++ can generate accurate pseudo-labels for up to 80m range.



sults comparable to training with manually labelled ground-truths for Waymo and nuScenes datasets. Fig. 4.13 shows examples of the labels generated with MS3D++ used in training. For Waymo as the target domain, the  $AP_{BEV}$  difference between training VX-A with MS3D++ labels compared to GT labels is 3.5  $AP_{BEV}$  for “Vehicle” and 5.17  $AP_{BEV}$  for “Pedestrian”. Similarly, for nuScenes, VX-A trained with MS3D++ shows only a 3.28  $AP_{BEV}$  reduction compared to training with GT labels. Additionally, MS3D++ outperforms all existing works in  $AP_{3D}$  for Waymo but is around  $-1AP_{3D}$  lower in nuScenes. One reason for this is due to the lidar sparsity of nuScenes where poor box height localization for a sparse observation can lead to the pseudo-label containing more ground points than object points, therefore affecting label quality.

**Table 4.15** – **Waymo** as the target domain for unsupervised domain adaptation. Detectors for all methods are trained on 4-frame accumulation ( $T_{\Delta MAX} = 0.3$ ).

| Method                    | Label Source  | Detector | Vehicle      |              | Pedestrian   |              |
|---------------------------|---------------|----------|--------------|--------------|--------------|--------------|
|                           |               |          | $AP_{BEV}$   | $AP_{3D}$    | $AP_{BEV}$   | $AP_{3D}$    |
| Source-Only               | Lyft          | VX-C     | 58.34        | 44.96        | 41.46        | 36.30        |
| Source-Only               | Lyft          | VX-A     | 59.06        | 45.99        | 39.56        | 34.04        |
| ST3D [Yang et al., 2021]  | Lyft          | VX-C     | 57.43        | 43.68        | 45.02        | 41.62        |
| MS3D [Tsai et al., 2023b] | Lyft,nuScenes | VX-C     | 64.34        | 47.75        | -            | -            |
| MS3D++                    | Lyft,nuScenes | VX-C     | <b>70.60</b> | <b>52.80</b> | <b>57.04</b> | <b>51.83</b> |
| MS3D++                    | Lyft,nuScenes | VX-A     | <b>70.31</b> | <b>52.31</b> | <b>52.71</b> | <b>48.92</b> |
| GT-Trained                | Waymo         | VX-C     | 75.11        | 61.28        | 67.86        | 62.92        |
| GT-Trained                | Waymo         | VX-A     | 73.81        | 60.52        | 57.89        | 54.71        |

Overall, the  $AP_{3D}$  lags behind  $AP_{BEV}$  as the label quality is heavily impacted by the challenging nature of box height estimation for sparsely observed objects on all datasets as shown in Fig. 4.9. Nonetheless, we highlight that BEV is often sufficient for various autonomous driving tasks such as obstacle avoidance, motion forecasting, and path planning. We believe that further improvement in 3D box localization can be achieved through leveling the pseudo-label boxes with ground plane segmentation [Oh et al., 2022, Lee et al., 2022] of all accumulated sequence frames, or using a 3D map [Zhang and Singh, 2014]. Additionally, we notice that training a single

**Table 4.16** – **nuScenes** as the target domain for unsupervised domain adaptation.

| Method                    | Sweeps | Label Source | Detector | Vehicle           |                  | Pedestrian        |                  |
|---------------------------|--------|--------------|----------|-------------------|------------------|-------------------|------------------|
|                           |        |              |          | AP <sub>BEV</sub> | AP <sub>3D</sub> | AP <sub>BEV</sub> | AP <sub>3D</sub> |
| Source-Only               | 4      | Waymo        | VX-C     | 42.18             | 22.29            | 16.30             | 11.02            |
| Source-Only               | 4      | Waymo        | VX-A     | 40.53             | 21.17            | 13.19             | 9.81             |
| Source-Only               | 1      | Waymo        | SECOND   | 32.10             | 17.77            | 7.32              | 5.01             |
| REDB [Chen et al., 2023]  | 1      | Waymo        | SECOND   | 30.12             | 18.56            | 2.47              | 2.14             |
| ST3D [Yang et al., 2021]  | 1      | Waymo        | SECOND   | 35.90             | 20.20            | 5.75              | 5.11             |
| DTS [Hu et al., 2023]     | 1      | Waymo        | SECOND   | 41.20             | 23.00            | -                 | -                |
| L.D [Wei et al., 2022]    | 1      | Waymo        | SECOND   | 42.04             | 24.50            | -                 | -                |
| MS3D [Tsai et al., 2023b] | 1      | Lyft, Waymo  | SECOND   | 42.23             | <b>24.76</b>     | -                 | -                |
| MS3D++                    | 1      | Lyft, Waymo  | SECOND   | <b>43.88</b>      | 23.10            | -                 | -                |
| ST3D [Yang et al., 2021]  | 10     | Waymo        | VX-C     | 38.80             | 23.37            | 11.59             | 9.04             |
| MS3D [Tsai et al., 2023b] | 10     | Lyft, Waymo  | VX-C     | 49.23             | <b>27.50</b>     | -                 | -                |
| MS3D++                    | 10     | Lyft, Waymo  | VX-C     | <b>50.34</b>      | 27.20            | <b>25.81</b>      | <b>15.91</b>     |
| MS3D++                    | 10     | Lyft, Waymo  | VX-A     | <b>52.06</b>      | <b>26.52</b>     | <b>27.01</b>      | <b>15.36</b>     |
| GT-Trained                | 1      | nuScenes     | SECOND   | 44.39             | 29.46            | -                 | -                |
| GT-Trained                | 10     | nuScenes     | VX-C     | 56.43             | 37.26            | 41.77             | 32.52            |
| GT-Trained                | 10     | nuScenes     | VX-A     | 55.34             | 36.67            | 38.30             | 29.72            |

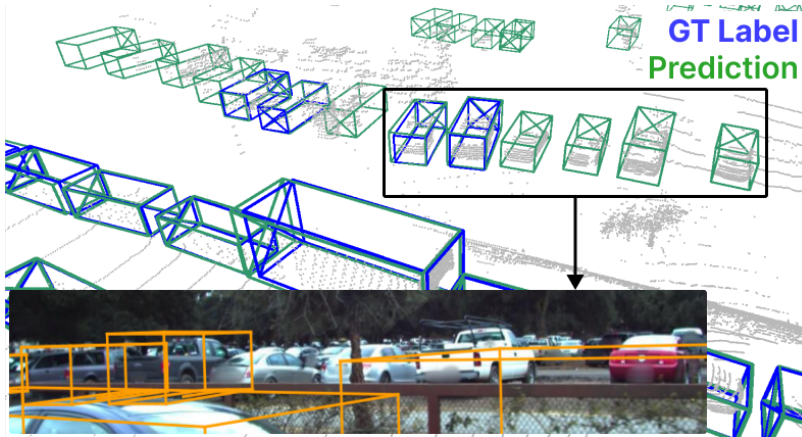
multi-frame detector on its own pseudo-labels (ST3D) can lead to a degradation in performance compared to the Source-only approach for all three target domains. This can be attributed to an amplified domain gap in the multi-frame setting. The gap is likely caused by inadequate localization of dynamic objects, stemming from significant differences in the appearances of motion tails and accumulated point clouds across datasets. Hence, although ST3D may be applicable in a multi-frame adaptation setting, its lack of accurate pseudo-labels ultimately leads to unsatisfactory domain transfer for these datasets.

For Lyft as the target domain in Table 4.17, we obtain a higher AP<sub>3D</sub> than all existing works. However, similar to the observation made by [Yang et al., 2021], we illustrate in Fig. 4.14 that Lyft has many missing ground-truth labels for parked vehicles on the side of the road, as their primary focus is on motion prediction of traffic

participants [Houston et al., 2020]. This makes it challenging to truly evaluate the domain adaptation performance of all methods due to predictions being penalized as false positives. Despite this limitation, we include these results from one round of self-training for the sake of completeness in our evaluation. We postulate that this labelling discrepancy plays a significant role in why Lyft-trained detectors often overlook many densely observed vehicles in Waymo and nuScenes, resulting in a notable decline in performance.

**Table 4.17** – **Lyft** as the target domain for unsupervised domain adaptation. Detectors for all methods are trained on 3-frame accumulation ( $T_{\Delta\text{MAX}} = 0.4$ ).

| Method                    | Label Source   | Detector | Vehicle                  |                         | Pedestrian               |                         |
|---------------------------|----------------|----------|--------------------------|-------------------------|--------------------------|-------------------------|
|                           |                |          | $\text{AP}_{\text{BEV}}$ | $\text{AP}_{3\text{D}}$ | $\text{AP}_{\text{BEV}}$ | $\text{AP}_{3\text{D}}$ |
| Source-Only               | Waymo          | VX-C     | 72.09                    | 57.55                   | 43.74                    | 38.42                   |
| Source-Only               | Waymo          | VX-A     | 74.28                    | 59.69                   | 42.52                    | 37.87                   |
| ST3D [Yang et al., 2021]  | Waymo          | VX-C     | 70.57                    | 57.08                   | <b>47.25</b>             | 41.84                   |
| MS3D [Tsai et al., 2023b] | Waymo,nuScenes | VX-C     | <b>77.30</b>             | 63.43                   | -                        | -                       |
| MS3D++                    | Waymo,nuScenes | VX-C     | 77.00                    | <b>65.96</b>            | 46.89                    | <b>43.26</b>            |
| MS3D++                    | Waymo,nuScenes | VX-A     | <b>77.20</b>             | <b>65.30</b>            | <b>47.15</b>             | <b>43.62</b>            |
| GT-Trained                | Lyft           | VX-C     | 86.85                    | 74.76                   | 60.67                    | 54.18                   |
| GT-Trained                | Lyft           | VX-A     | 85.23                    | 72.70                   | 58.86                    | 50.37                   |



**Figure 4.14** – Missing labels on the Lyft dataset affects accurate evaluation of domain adaptation performance. Additionally, this contributes to the poor performance when testing Lyft-trained detectors on Waymo or nuScenes due to the difference in labelling guidelines.

A major advantage of MS3D++ lies in the flexibility of simply having a set of pseudo-labels for domain adaptation. This means that data augmentation techniques that proves effective in a supervised setting such as multi-frame accumulation can seamlessly integrate with our approach. From our results, we demonstrate the successful application of multi-frame accumulation for multiple classes across various target domains, yielding a significant improvement over Source-only for both rigid and non-rigid (i.e., "Vehicle" and "Pedestrian") classes on multiple target domains. Likewise, we believe that further improvements can be obtained through the use of the popular ground-truth database data augmentation [Yan et al., 2018] to increase the number of labelled samples per point cloud for each round of self-training or just the final round.

**Table 4.18** – VMFI and TTA for ensembling. Precision (P) and Recall (R) range breakdown on Waymo point clouds. Source is given as Training dataset /  $T_{\Delta\text{MAX}}$  / Detector.

| Source          | Class | VMFI TTA |   | 0-30m        |              | 30-50m       |              | 50-80m       |              |
|-----------------|-------|----------|---|--------------|--------------|--------------|--------------|--------------|--------------|
|                 |       |          |   | P            | R            | P            | R            | P            | R            |
| L / 0.4 / VX-C  | Veh.  |          |   | 0.960        | 0.770        | 0.911        | 0.455        | 0.860        | 0.208        |
|                 |       | ✓        |   | 0.959        | 0.805        | 0.919        | 0.492        | 0.868        | 0.233        |
|                 |       | ✓        | ✓ | <b>0.962</b> | <b>0.820</b> | <b>0.926</b> | <b>0.508</b> | <b>0.886</b> | <b>0.242</b> |
|                 | Ped.  |          |   | 0.726        | 0.425        | 0.746        | 0.298        | 0.771        | 0.143        |
|                 |       | ✓        |   | 0.868        | 0.444        | 0.851        | 0.315        | 0.838        | 0.154        |
|                 |       | ✓        | ✓ | <b>0.872</b> | <b>0.489</b> | <b>0.867</b> | <b>0.342</b> | <b>0.863</b> | <b>0.157</b> |
| N / 0.45 / PV-C | Veh.  |          |   | 0.976        | 0.693        | 0.957        | 0.318        | 0.949        | 0.051        |
|                 |       | ✓        |   | <b>0.976</b> | 0.769        | 0.957        | 0.377        | 0.952        | 0.067        |
|                 |       | ✓        | ✓ | 0.975        | <b>0.805</b> | <b>0.958</b> | <b>0.418</b> | <b>0.957</b> | <b>0.078</b> |
|                 | Ped.  |          |   | 0.533        | 0.194        | 0.535        | <b>0.038</b> | 0.625        | <0.01        |
|                 |       | ✓        |   | 0.652        | 0.184        | 0.653        | 0.027        | <b>0.690</b> | <0.01        |
|                 |       | ✓        | ✓ | <b>0.671</b> | <b>0.205</b> | <b>0.680</b> | 0.029        | 0.500        | <0.01        |

## Ablation

**VMFI.** We validate VMFI’s effectiveness in Table 4.18 using two models and source domains, showing its consistent improvement of precision and recall across all classes and ranges. For vehicles, VMFI significantly improves recall at all ranges, mainly

due to increased point density for better detection of static and slow-moving vehicles. Meanwhile, for pedestrians, VMFI substantially increases precision across all ranges, achieving an improvement of up to 20%. This improvement is due to the fact that pedestrians have sparser observations compared to vehicles, primarily due to their smaller size. Multi-frame accumulation is particularly effective on pedestrian detection due to their slow movement speed, resulting in better box localization from increased point density, with minimal side effects from long motion tails. Additionally, incorporating TTA further enhances both precision and recall.

**Table 4.19** – Ensemble combinations for Lyft (L) and nuScenes (N) detectors on Waymo point clouds. We report  $AP_{3D}$  in the ranges 0-30m, 30-50m and 50-80m.

| Source                | PV-A | PV-C | VX-A | VX-C | Vehicle     |             |             | Pedestrian  |             |             |
|-----------------------|------|------|------|------|-------------|-------------|-------------|-------------|-------------|-------------|
|                       |      |      |      |      | 0-30        | 30-50       | 50-80       | 0-30        | 30-50       | 50-80       |
| L                     | ✓    |      |      |      | 74.4        | 44.4        | 20.1        | 38.5        | 25.6        | 12.8        |
| L                     |      | ✓    |      |      | 73.0        | 43.2        | 19.1        | 33.0        | 25.1        | 15.0        |
| L                     |      |      | ✓    |      | 74.6        | 46.0        | 21.4        | 43.6        | 32.0        | 20.5        |
| L                     |      |      |      | ✓    | 73.1        | 45.8        | 21.6        | 46.6        | 37.2        | 27.0        |
| L                     |      |      | ✓    | ✓    | 79.5        | 52.5        | 26.2        | <b>57.3</b> | 46.6        | <b>31.8</b> |
| L                     | ✓    | ✓    |      |      | 78.1        | 49.7        | 24.1        | 48.7        | 35.9        | 19.8        |
| L                     |      | ✓    |      | ✓    | 78.2        | 50.8        | 25.1        | 54.6        | 45.1        | 30.8        |
| L                     | ✓    |      | ✓    |      | 79.8        | 52.4        | 26.2        | 51.7        | 38.1        | 22.0        |
| L                     |      | ✓    | ✓    |      | 79.3        | 51.8        | 25.4        | 52.6        | 40.3        | 24.1        |
| L                     | ✓    |      |      | ✓    | 79.2        | 52.2        | 26.1        | 55.2        | 44.5        | 30.1        |
| L                     | ✓    | ✓    | ✓    | ✓    | 80.6        | 54.6        | <b>28.0</b> | 57.1        | 46.7        | 31.7        |
| N+L                   | ✓    | ✓    | ✓    | ✓    | 82.6        | 52.7        | 25.1        | 46.8        | 38.6        | 28.5        |
| N+L <sub>weight</sub> | ✓    | ✓    | ✓    | ✓    | <b>82.6</b> | <b>55.7</b> | 27.8        | 57.1        | <b>46.7</b> | 31.7        |

**Ensemble contributions and weighing.** In Table 4.19 we present the performance of various detector combinations on the Waymo dataset. We demonstrate that any ensembling any pair of detectors with TTA and VMFI easily surpasses the performance of using a single detector for pseudo-labelling. Although the Lyft-trained VX-A and VX-C are the highest performing pre-trained detectors on Waymo, we show that combining all detectors with KBF outperforms the VX-A and VX-C pair. However, when introducing nuScenes detectors into the ensemble, we observed a decrease in overall pedestrian, and 30-80m vehicle detection. nuScenes-trained detectors tends to predict more false positive pedestrians and vehicles, particularly at farther

ranges. This is potentially due to the fact that nuScenes detectors are trained to make highly confident predictions from sparse observations (e.g., a single lidar ring) for objects beyond 30m range. In a cross-domain context, this leads to noisy predictions due to a different distribution of points between source and target domains, particularly for semi-occluded or far-range structures. Based on these insights, we assign a higher weight to Lyft-trained detectors for vehicle predictions, and we discard the pedestrian predictions from nuScenes-trained models to reduce false positives in our ensemble. We demonstrate that a simple weighing of Lyft detectors for KBF can enhance the overall vehicle  $AP_{3D}$ . Furthermore, we emphasize that this aspect of MS3D++ is not limited to the selected detectors in this paper. Depending on the use-case, incorporating detectors with varying voxel sizes, multi-modal capabilities, or class-specific detectors into the ensemble can further enhance the robustness of the initial pseudo-label set.

**Table 4.20** – Label breakdown of KBF and temporal refinement in MS3D++. Precision (P) and Recall (R) breakdown by range for Waymo point clouds. Delta  $\Delta$  is the difference between Ensemble and Refined.

| Class      | Label    | 0-30m  |              | 30-50m |              | 50-80m |              |
|------------|----------|--------|--------------|--------|--------------|--------|--------------|
|            |          | P      | R            | P      | R            | P      | R            |
| Vehicle    | VX-A     | 0.946  | 0.899        | 0.894  | 0.705        | 0.847  | 0.427        |
|            | VX-C     | 0.950  | 0.896        | 0.904  | 0.695        | 0.864  | 0.403        |
|            | Ensemble | 0.950  | 0.899        | 0.904  | 0.704        | 0.866  | 0.418        |
|            | Refined  | 0.944  | 0.912        | 0.893  | 0.763        | 0.851  | 0.514        |
|            | $\Delta$ | -0.006 | <b>0.013</b> | -0.011 | <b>0.059</b> | -0.015 | <b>0.096</b> |
| Pedestrian | VX-A     | 0.875  | 0.623        | 0.846  | 0.493        | 0.834  | 0.325        |
|            | VX-C     | 0.881  | 0.609        | 0.863  | 0.485        | 0.867  | 0.326        |
|            | Ensemble | 0.872  | 0.646        | 0.853  | 0.518        | 0.849  | 0.351        |
|            | Refined  | 0.846  | 0.689        | 0.818  | 0.574        | 0.802  | 0.431        |
|            | $\Delta$ | -0.026 | <b>0.043</b> | -0.035 | <b>0.056</b> | -0.047 | <b>0.080</b> |

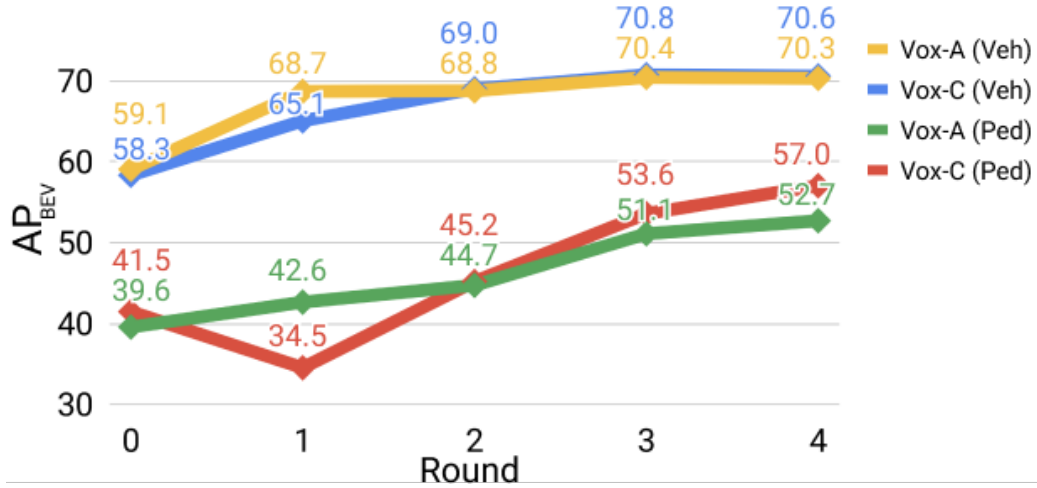
Ensemble (KBF): VX-C and VX-A with VMFI and TTA

Refined: final pseudo-label set of MS3D++

**Label breakdown.** To better understand the effect of KBF ensembling and label refinement, we analyse the precision and recall of our pseudo-labels in Table 4.20. For the vehicle class, KBF ensembling (Ensemble), can boost both precision and recall for

all ranges. However, when it comes to pedestrians, their small size makes them more prone to occlusion, resulting in sparser observations and poorer box localization. As a result, the precision for pedestrians is affected. Nonetheless, we find that ensembling generally improves the recall of pedestrians. Subsequently, we demonstrate that the final set of pseudo-labels after temporal refinement (Refined) greatly increases the recall of our ensemble with a slight trade-off in precision. The improvement in recall is most prominent at the 30-80m range, which effectively extends the detection range of detectors adapted to new target domains.

**Iterative Label Refinement.** Fig. 4.15 highlights the effectiveness of multiple rounds of self-training. The largest improvement in vehicle detection is in the 1st round of self-training using an ensemble of Lyft and nuScenes pre-trained detectors. However, as we proceed with more rounds of self-training, the improvement starts to plateau. On the other hand, for the pedestrian class, the 1st round of self-training leads to a decrease in  $AP_{BEV}$  due to prioritizing a high precision for accurately distinguishing true pedestrians.



**Figure 4.15** – Iterative label refinement on Waymo as the target domain with VoxelRCNN-Anchorhead (VX-A) and VoxelRCNN-Centerhead (VX-C). We show the improvement in detection rate of VX-A and VX-C over multiple rounds of self-training. All detectors are trained on 4-frame point cloud accumulation. The 0th round refers to the detectors without any domain adaptation (i.e., Source-Only).

In subsequent rounds, more aligned confidence scores enables better thresholding of true pedestrians, allowing us to use more pedestrian tracks and increasing overall

pseudo-label recall. We study this by examining the pseudo-label quality for each round in Table 4.21. The analysis clearly demonstrates the high precision and low recall of the first round of pedestrian pseudo-labelling, with a substantial increase with each subsequent round. In general, for both classes, each round exhibits a significant increase in recall, with a minor trade-off in precision. However, we argue that for the autonomous driving application, a high recall is more important for safety considerations. The increase in recall is particularly pronounced at farther ranges (30-80m), where it is approximately 3 to 4 times larger than the drop in precision. For instance, the recall of the pedestrian class in round 4 at 50-80m is  $10\times$  larger than the recall in round 1, with only a 12% decrease in precision. This is subsequently reflected in the significant improvement in pedestrian  $AP_{BEV}$  in later self-training rounds. Similar to Section 4.4.3, the pseudo-labels for each round can be used to train any type of detector for the ensemble of subsequent rounds. This includes training class-specific or multi-modal detectors.

**Table 4.21** – Iterative Label Refinement. Precision (P) and Recall (R) range breakdown of the pseudo-label quality for each self-training round. Delta  $\Delta$  is the difference between round 4 and round 1.

| Class      | Round    | 0-30m  |              | 30-50m |              | 50-80m |              |
|------------|----------|--------|--------------|--------|--------------|--------|--------------|
|            |          | P      | R            | P      | R            | P      | R            |
| Vehicle    | 1        | 0.962  | 0.896        | 0.934  | 0.648        | 0.913  | 0.333        |
|            | 2        | 0.955  | 0.909        | 0.920  | 0.720        | 0.900  | 0.436        |
|            | 3        | 0.944  | 0.912        | 0.898  | 0.759        | 0.865  | 0.505        |
|            | 4        | 0.944  | 0.912        | 0.893  | 0.763        | 0.851  | 0.514        |
|            | $\Delta$ | -0.018 | <b>0.016</b> | -0.041 | <b>0.115</b> | -0.062 | <b>0.181</b> |
| Pedestrian | 1        | 0.912  | 0.324        | 0.922  | 0.131        | 0.906  | 0.043        |
|            | 2        | 0.881  | 0.532        | 0.891  | 0.372        | 0.909  | 0.182        |
|            | 3        | 0.874  | 0.628        | 0.861  | 0.497        | 0.870  | 0.322        |
|            | 4        | 0.846  | 0.689        | 0.818  | 0.574        | 0.802  | 0.431        |
|            | $\Delta$ | -0.066 | <b>0.365</b> | -0.104 | <b>0.443</b> | -0.104 | <b>0.388</b> |

**Pseudo-label generation duration.** The first step of MS3D++ is to generate the predictions from each pre-trained detector with VMFI and TTA for the ensemble. The duration of this depends on the density of the target domain point clouds, detector architecture, and number of detectors used. This can generally range from 3-12



hours. Following this, the remaining steps of MS3D++ can range from 6-20 hours depending on the number of objects in each frame. For instance, on 20k frames from the Waymo dataset, MS3D++ took around 20 hours due to our focus on carparks, and the extensive range of the 64-beam lidar. In contrast, a similar number of frames for nuScenes point clouds took around 6 hours as less objects are observed because of the limited usable range of a 32-beam lidar. A large bottleneck is from the tracker implementation [Pang et al., 2021], which accounts for up to 11 out of the 20 hours for the Waymo dataset. Overall, MS3D++ is still significantly more time efficient than manual labelling. We highlight that MS3D++ can also be used in conjunction with human annotators as a strong set of initial labels, or as a starting point for exploring active learning approaches. MS3D++ only requires a GPU for pre-trained detector box proposals and detector re-training. The pseudo-labelling steps do not require a GPU for computation. Nevertheless, our experiments were run with GPU-optimized IoU computations on an i7-6700K CPU with 60GB RAM and a 2080TI GPU.

## 4.5 Conclusion

In this chapter, we proposed to adapt 3D detectors to new target domains with an auto-labelling approach. Our proposed MS3D frameworks effectively generates high quality pseudo-labels for both low and high density lidar through detector ensembling and temporal refinement.

We presented a comprehensive analysis of the idiosyncrasies of 3D detectors in a cross-domain context to show that simply using a single detector faces many challenges, resulting in the suboptimal performance of single-detector self-training methods like ST3D [Yang et al., 2021]. With such methods there is a need to evaluate the performance of pre-trained 3D detectors for selecting a single detector for self-training. However, this is not feasible in the context of UDA due to the absence of labels in the target domain. Furthermore, in this analysis, we redefine the baseline of 3D detectors in unsupervised domain adaptation, showing that simply using multiple frames for training the pre-trained detector can already provide a significant boost in

performance on multiple target domains.

Through our work, we demonstrate that MSDA is a highly practical approach for unsupervised domain adaptation. We introduced a novel box fusion technique, KBF, for ensembling multiple detectors, resulting in a significant increase in generalization ability across a variety of target domains. With KBF, MS3D bypasses the issue of selecting a single detector for self-training, demonstrating that our ensembling performance can always outperform that of the best pre-trained detector.

Notably, our proposed MS3D framework is versatile, as it is not limited to particular detector architectures, data augmentation or lidar types. This means that regardless of environmental changes, emergence of novel lidar types or detector architectures, our ensemble approach remains effective and relevant. MS3D can use new detector architectures, including multi-modal detectors, to enhance the domain generalization on given target domains. Moreover, the continuous release of new autonomous driving datasets [Alibeigi et al., 2023] with newer lidars adds to our collection pre-trained detectors, allowing us to improve our ensemble on a variety of target domains, including novel lidar types.

Extensive experiments show that models trained with our pseudo-labels obtain state-of-the-art results on all evaluated target domains and are even comparable to training with human-annotated labels. Furthermore, our pseudo-labels are simply a drop-in replacement for ground-truth labels in a supervised training setting. This makes our domain adaptation approach highly versatile, supporting the training of newly developed 3D detectors, or data augmentation techniques. MS3D offers a practical and flexible solution for domain adaptation that maintains detector runtime and eliminates the need for modification of 3D detector architectures or custom learnable modules.

The contributions in this chapter have been presented in the following publications:

- D. Tsai, J. S. Berrio, M. Shan, E. Nebot, and S. Worrall. "MS3D: Leveraging Multiple Detectors for Unsupervised Domain Adaptation in 3D Object Detection" In *2023 IEEE International Intelligent Transportation Systems Conference*

(*ITSC*). IEEE, 2023. (accepted)

- D. Tsai, J. S. Berrio, M. Shan, E. Nebot, and S. Worrall. "MS3D++: Ensemble of Experts for Multi-Source Unsupervised Domain Adaptation in 3D Object Detection" *Submitted to IEEE Transactions on Pattern Analysis and Machine Intelligence (T-PAMI)*

# Chapter 5

## Conclusion

This chapter is organized as follows: Section 5.1 summarizes our proposed methods, Section 5.2 provides an overview of the contributions of this thesis and Section 5.3 presents potential ideas for future work.

### 5.1 Summary

In this thesis, we have presented frameworks for UDA that enable companies and research groups to effectively utilize cutting-edge 3D detector research with a range of lidar technologies. The motivation for our work stems from the observation that 3D Detectors often have a significant performance drop when deployed in an unfamiliar domain with drastic differences from the training data due to domain shift. This results in missing detections for densely observed objects, misaligned confidence scores, and increased high-confidence false positives, rendering the detector highly unreliable. This is a prominent issue as it hinders the widespread adoption of various types of lidar technologies.

In Chapter 3, we present a framework for detector adaptation to multiple target domains through the use of a domain-invariant representation. Our proposed method uses the camera as a complementary sensor with image instance segmentation for

region of interest proposals to act as a bridge between different lidar types. We introduce a camera-lidar calibration pipeline to ensure robust calibration to enable optimal use of the camera for region proposals. Furthermore, we present a surface completion network that gives a consistent unified representation of points across different lidar types. We demonstrate that with a strong lidar-camera calibration and surface completion, we can obtain good 3D detection performance across multiple lidars without requiring re-training on each new target domain.

In Chapter 4, we introduce an auto-labelling framework for re-training a detector on a given target domain. By generating a set of high quality pseudo-labels, our approach enables the re-training of any 3D detector to obtain strong 3D detection performance for a range of lidar types. We adopt a multi-source approach, which enables more robust generation of pseudo-labels compared to single-source approaches. Furthermore, we temporally refine the pseudo-labels to obtain accurate box localization even at farther ranges. We also present a comprehensive study to give insights into the performance of pre-trained detectors in a cross-domain setting. These insights further allowed us to improve our framework substantially, and on more classes. Overall, our experiments demonstrate that 3D detectors trained with our generated pseudo-labels achieve high 3D detection rate on various lidar types, even comparable to training with human-annotated labels.

## 5.2 Contributions

The main contributions of this thesis are the development of two key frameworks for achieving 3D object detection on a variety of domains. These frameworks were comprised of various contributions which we detail below.

### Domain-Invariant Representation

- Proposed a camera-lidar calibration pipeline that automates the calibration parameter estimation process using expert knowledge. Our algorithm selects

optimal calibration poses from recorded variations, therefore simplifying the calibration process.

- Developed an unsupervised multi-target domain adaptation framework that works across different lidar scan patterns, fixed or adjustable, without necessitating model fine-tuning for each pattern. We demonstrate that having a strong calibration between camera and lidar enables the use of camera as a complementary modality for lidar-agnostic 3D detection. By transforming lidar-specific object representations into a unified representation, we mitigate biases towards specific lidar representations.
- Developed a viewer-centred surface completion network that completes the surface of objects observed from the sensor frame. We demonstrate that our network is lidar-agnostic, able to complete the surfaces of objects from multiple types of lidars. Furthermore, we showcase the use of our network in unsupervised domain adaptation within our proposed unsupervised multi-target domain adaptation framework.

### **Auto-labelling**

- Developed a box fusion method, Kernel density estimation Box Fusion (KBF) that effectively combines the predictions of multiple detectors. We demonstrate that KBF consistently outperforms the performance of a single detector. KBF is versatile and allows for detector ensembling within both a supervised and unsupervised setting.
- Introduced an auto-labelling framework for adapting 3D detectors to unfamiliar target domains. We demonstrate the use of KBF for ensembling and label refinement of multiple single-frame pre-trained detectors on a given target domain. Our experiments show that this approach can effectively achieve strong 3D detection on a variety of target domains without human-annotations.
- Comprehensive study of pre-trained 3D detectors' baseline performance across

domains in Chapter 4. This allowed us to re-define the baseline performance of 3D detectors using multi-frame pre-training and we incorporate all insights into the auto-labelling ensembling step.

- Improved our auto-labelling framework for better domain generalization and pedestrian labelling. We incorporate an iterative refinement strategy with multi-stage self-training and demonstrate that our proposed method improves 3D detection rate as well as obtains strong performance for pedestrian detection.

## 5.3 Future Research

This thesis is focused on achieving accurate 3D detection on any given target domain without any human-annotated labels. In practice however, there is often some budget for human-annotation. When considering human labelling in the data pipeline, the problem shifts focus to the task of finding the optimal frames for human-annotation that maximises value for minimal cost. Active learning is a prominent field that aims to find the most valuable frames for human annotators that contribute the most in reducing uncertainty or improving model performance. It aims to maximize the value obtained for each annotation. We believe that unsupervised learning approaches like MS3D can be supplemented by active learning concepts. One application of this in our auto-labelling framework is to model the uncertainty of each frame and replace uncertain frames with human-annotated labels. This allows us to obtain a more robust set of labels for the entire dataset and adapt to more challenging scenarios. Another direction is the use of active learning to identify the most informative data samples for our auto-labelling framework. This would result in less training time and potentially reduce errors in our pseudo-labels.

Lidar-based detectors have reigned as the superior 3D detection approach for many years, however that has begun to shift with more multi-modal 3D detector architectures [Liu et al., 2023, Bai et al., 2022, Li et al., 2022] surpassing the performance of lidar-only 3D detection. In theory, such models should exhibit better domain generalization to unfamiliar target domains due to the use of multiple modalities, however

there has not been any study into this nor any existing UDA approaches that address multi-modal 3D detector adaptation. We believe that multi-modal 3D detectors can further boost our ensembling baseline, and additionally, by considering the camera in our labelling framework, we can better distinguish false positives classes, leading to better pseudo-label quality and iterative label refinement. Expanding on the idea of multiple sources of information, we can additionally incorporate 3D maps [Zhang and Singh, 2014] in our pseudo-label refinement process to better localize the height of sparsely observed objects.

To improve the domain generalization of 3D detectors, a promising approach is to explore the effective use of multi-dataset training [Zhang et al., 2023]. These approaches aim to learn domain-invariant features across datasets such that the model does not generalize to just one type of scan pattern or scene, but rather, learns the underlying principles of the task such as object geometry. By effectively combining multiple datasets, many existing works in 2D perception have shown to bring better domain generalization for a range of target domains. This could additionally be coupled with the integration of large language models to direct the learning of multiple datasets towards particular feature representations of class labels to alleviate the class label discrepancies across datasets. For instance, when training a 3D detector, classes such as car, truck and bus are often considered as different classes where their feature vectors may not be similar. However, with a large language model, the feature vector of a car, truck and bus may be similar as language models can capture the language semantics of class definitions. Such alignment of different classes in a multi-dataset training situation would prove beneficial for enabling a 3D detector to better learn various class representations across lidar datasets and therefore, improve domain generalization.



# List of References

- [Sun et al., 2020] Sun, P., Kretzschmar, H., Dotiwalla, X., Chouard, A., Patnaik, V., Tsui, P., Guo, J., Zhou, Y., Chai, Y., Caine, B., et al. (2020). Scalability in perception for autonomous driving: Waymo open dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2446–2454.
- [Geiger et al., 2012] Geiger, A., Lenz, P., and Urtasun, R. (2012). Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE.
- [Caesar et al., 2020] Caesar, H., Bankiti, V., Lang, A. H., Vora, S., Liong, V. E., Xu, Q., Krishnan, A., Pan, Y., Baldan, G., and Beijbom, O. (2020). nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11621–11631.
- [Shi et al., 2023] Shi, S., Jiang, L., Deng, J., Wang, Z., Guo, C., Shi, J., Wang, X., and Li, H. (2023). PV-RCNN++: Point-voxel feature set abstraction with local vector representation for 3D object detection. *International Journal of Computer Vision*, 131(2):531–551.
- [Kesten et al., 2019] Kesten, R., Usman, M., Houston, J., Pandya, T., Nadhamuni, K., Ferreira, A., Yuan, M., Low, B., Jain, A., Ondruska, P., Omari, S., Shah, S., Kulkarni, A., Kazakova, A., Tao, C., Platinsky, L., Jiang, W., and Shet, V. (2019). Woven Planet Perception Dataset 2020.  
<https://www.woven-planet.global/en/data/perception-dataset>.
- [Deng et al., 2020] Deng, J., Shi, S., Li, P., Zhou, W., Zhang, Y., and Li, H. (2020). Voxel R-CNN: Towards High Performance Voxel-based 3D Object Detection. *arXiv preprint arXiv:2012.15712*.
- [Yuan et al., 2018] Yuan, W., Khot, T., Held, D., Mertz, C., and Hebert, M. (2018). Pcn: Point completion network. In *2018 International Conference on 3D Vision (3DV)*, pages 728–737. IEEE.

- [Yu et al., 2021] Yu, X., Rao, Y., Wang, Z., Liu, Z., Lu, J., and Zhou, J. (2021). PointR: Diverse point cloud completion with geometry-aware transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12498–12507.
- [Wang et al., 2020] Wang, Y., Chen, X., You, Y., Li, L. E., Hariharan, B., Campbell, M., Weinberger, K. Q., and Chao, W.-L. (2020). Train in germany, test in the usa: Making 3d object detectors generalize. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11713–11723.
- [Yang et al., 2021] Yang, J., Shi, S., Wang, Z., Li, H., and Qi, X. (2021). ST3D: Self-training for Unsupervised Domain Adaptation on 3D Object Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10368–10378.
- [Wei et al., 2022] Wei, Y., Wei, Z., Rao, Y., Li, J., Zhou, J., and Lu, J. (2022). LiDAR Distillation: Bridging the Beam-Induced Domain Gap for 3D Object Detection. *arXiv preprint arXiv:2203.14956*.
- [Neubeck and Van Gool, 2006] Neubeck, A. and Van Gool, L. (2006). Efficient non-maximum suppression. In *18th international conference on pattern recognition (ICPR'06)*, volume 3, pages 850–855. IEEE.
- [Solovyev et al., 2021] Solovyev, R., Wang, W., and Gabruseva, T. (2021). Weighted boxes fusion: Ensembling boxes from different object detection models. *Image and Vision Computing*, 107:104117.
- [Alibeigi et al., 2023] Alibeigi, M., Ljungbergh, W., Tonderski, A., Hess, G., Lilja, A., Lindstrom, C., Motorniuk, D., Fu, J., Widahl, J., and Petersson, C. (2023). Zenseact open dataset: A large-scale and diverse multimodal dataset for autonomous driving. *arXiv preprint arXiv:2305.02008*.
- [Tsai et al., 2023] Tsai, D., Berrio, J. S., Shan, M., Nebot, E., and Worrall, S. (2023). MS3D++: Ensemble of Experts for Multi-Source Unsupervised Domain Adaption in 3D Object Detection. *arXiv preprint arXiv:2308.05988*.
- [Wang et al., 2023] Wang, H., Shi, C., Shi, S., Lei, M., Wang, S., He, D., Schiele, B., and Wang, L. (2023). Dsvt: Dynamic sparse voxel transformer with rotated sets. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13520–13529.
- [Chen et al., 2022] Chen, X., Shi, S., Zhu, B., Cheung, K. C., Xu, H., and Li, H. (2022). MPPNet: Multi-Frame Feature Intertwining with Proxy Points for 3D Temporal Object Detection. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part VIII*, pages 680–697. Springer.

- [Liu et al., 2023] Liu, Z., Tang, H., Amini, A., Yang, X., Mao, H., Rus, D. L., and Han, S. (2023). Bevfusion: Multi-task multi-sensor fusion with unified bird’s-eye view representation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2774–2781. IEEE.
- [Pulikkaseril and Langdale-Smith, 2020] Pulikkaseril, C. and Langdale-Smith, N. (2020). Next generation LiDAR for a fully autonomous future. White paper, Baraja.
- [Zhang et al., 2021] Zhang, W., Li, W., and Xu, D. (2021). SRDAN: Scale-Aware and Range-Aware Domain Adaptation Network for Cross-Dataset 3D Object Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6769–6779.
- [Tsai et al., 2021] Tsai, D., Worrall, S., Shan, M., Lohr, A., and Nebot, E. (2021). Optimising the selection of samples for robust lidar camera calibration. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 2631–2638. IEEE.
- [Tsai et al., 2022] Tsai, D., Berrio, J. S., Shan, M., Worrall, S., and Nebot, E. (2022). See Eye to Eye: A Lidar-Agnostic 3D Detection Framework for Unsupervised Multi-Target Domain Adaptation. *IEEE Robotics and Automation Letters*, 7(3):7904–7911.
- [Tsai et al., 2023a] Tsai, D., Berrio, J. S., Shan, M., Nebot, E., and Worrall, S. (2023a). Viewer-centred surface completion for unsupervised domain adaptation in 3D object detection. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9346–9353. IEEE.
- [Tsai et al., 2023b] Tsai, D., Berrio, J. S., Shan, M., Nebot, E., and Worrall, S. (2023b). MS3D: Leveraging Multiple Detectors for Unsupervised Domain Adaptation in 3D Object Detection. *arXiv preprint arXiv:2304.02431*.
- [Yan et al., 2018] Yan, Y., Mao, Y., and Li, B. (2018). Second: Sparsely embedded convolutional detection. *Sensors*, 18(10):3337.
- [Zhou and Tuzel, 2018] Zhou, Y. and 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*, pages 4490–4499.
- [Chen et al., 2023] Chen, Y., Liu, J., Zhang, X., Qi, X., and Jia, J. (2023). Voxelnext: Fully sparse voxelnet for 3d object detection and tracking. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21674–21683.

- [Liu et al., 2019] Liu, Z., Tang, H., Lin, Y., and Han, S. (2019). Point-voxel cnn for efficient 3d deep learning. *Advances in Neural Information Processing Systems*, 32.
- [Zhang et al., 2022] Zhang, Y., Hu, Q., Xu, G., Ma, Y., Wan, J., and Guo, Y. (2022). Not all points are equal: Learning highly efficient point-based detectors for 3d lidar point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18953–18962.
- [Yang et al., 2020] Yang, Z., Sun, Y., Liu, S., and Jia, J. (2020). 3dssd: Point-based 3d single stage object detector. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11040–11048.
- [Shi et al., 2019] Shi, S., Wang, X., and Li, H. (2019). Pointrcnn: 3d object proposal generation and detection from point cloud. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 770–779.
- [Qi et al., 2017a] Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017a). Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660.
- [Qi et al., 2017b] Qi, C. R., Yi, L., Su, H., and Guibas, L. J. (2017b). Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in Neural Information Processing systems*, 30.
- [Shi et al., 2020] Shi, S., Guo, C., Jiang, L., Wang, Z., Shi, J., Wang, X., and Li, H. (2020). Pv-rcnn: Point-voxel feature set abstraction for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10529–10538.
- [Yang et al., 2019] Yang, Z., Sun, Y., Liu, S., Shen, X., and Jia, J. (2019). Std: Sparse-to-dense 3d object detector for point cloud. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1951–1960.
- [Shi et al., 2020] Shi, S., Wang, Z., Shi, J., Wang, X., and Li, H. (2020). From points to parts: 3d object detection from point cloud with part-aware and part-aggregation network. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- [Noh et al., 2021] Noh, J., Lee, S., and Ham, B. (2021). Hvpr: Hybrid voxel-point representation for single-stage 3d object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14605–14614.

- [Chen et al., 2017] Chen, X., Ma, H., Wan, J., Li, B., and Xia, T. (2017). Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1907–1915.
- [Guan et al., 2022] Guan, T., Wang, J., Lan, S., Chandra, R., Wu, Z., Davis, L., and Manocha, D. (2022). M3detr: Multi-representation, multi-scale, mutual-relation 3d object detection with transformers. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 772–782.
- [Zhou et al., 2020] Zhou, Y., Sun, P., Zhang, Y., Anguelov, D., Gao, J., Ouyang, T., Guo, J., Ngiam, J., and Vasudevan, V. (2020). End-to-end multi-view fusion for 3d object detection in lidar point clouds. In *Conference on Robot Learning*, pages 923–932. PMLR.
- [Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. *Advances in neural information processing systems*, 30.
- [Dosovitskiy et al., 2020] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*.
- [Liu et al., 2021] Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. (2021). Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022.
- [Fan et al., 2022] Fan, L., Pang, Z., Zhang, T., Wang, Y.-X., Zhao, H., Wang, F., Wang, N., and Zhang, Z. (2022). Embracing single stride 3d object detector with sparse transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8458–8468.
- [Sheng et al., 2021] Sheng, H., Cai, S., Liu, Y., Deng, B., Huang, J., Hua, X.-S., and Zhao, M.-J. (2021). Improving 3d object detection with channel-wise transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2743–2752.
- [Yang et al., 2021] Yang, Z., Zhou, Y., Chen, Z., and Ngiam, J. (2021). 3d-man: 3d multi-frame attention network for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1863–1872.
- [Qi et al., 2021] Qi, C. R., Zhou, Y., Najibi, M., Sun, P., Vo, K., Deng, B., and Anguelov, D. (2021). Offboard 3d object detection from point cloud sequences. In

- Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6134–6144.
- [Ma et al., 2023] Ma, T., Yang, X., Zhou, H., Li, X., Shi, B., Liu, J., Yang, Y., Liu, Z., He, L., Qiao, Y., et al. (2023). DetZero: Rethinking Offboard 3D Object Detection with Long-term Sequential Point Clouds. *arXiv preprint arXiv:2306.06023*.
- [Fan et al., 2023a] Fan, L., Yang, Y., Mao, Y., Wang, F., Chen, Y., Wang, N., and Zhang, Z. (2023a). Once Detected, Never Lost: Surpassing Human Performance in Offline LiDAR based 3D Object Detection. *arXiv preprint arXiv:2304.12315*.
- [Fan et al., 2023b] Fan, L., Yang, Y., Mao, Y., Wang, F., Chen, Y., Wang, N., and Zhang, Z. (2023b). Once Detected, Never Lost: Surpassing Human Performance in Offline LiDAR based 3D Object Detection. *arXiv preprint arXiv:2304.12315*.
- [Yin et al., 2021] Yin, T., Zhou, X., and Krähenbühl, P. (2021). Multimodal virtual point 3d detection. *Advances in Neural Information Processing Systems*, 34:16494–16507.
- [Vora et al., 2020] Vora, S., Lang, A. H., Helou, B., and Beijbom, O. (2020). Pointpainting: Sequential fusion for 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4604–4612.
- [Li et al., 2022] Li, Y., Yu, A. W., Meng, T., Caine, B., Ngiam, J., Peng, D., Shen, J., Lu, Y., Zhou, D., Le, Q. V., et al. (2022). Deepfusion: Lidar-camera deep fusion for multi-modal 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17182–17191.
- [He et al., 2017] He, K., Gkioxari, G., Dollár, P., and Girshick, R. (2017). Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969.
- [Yin et al., 2021] Yin, T., Zhou, X., and Krahenbuhl, P. (2021). Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11784–11793.
- [Hu et al., 2022] Hu, Y., Ding, Z., Ge, R., Shao, W., Huang, L., Li, K., and Liu, Q. (2022). Afdetv2: Rethinking the necessity of the second stage for object detection from point clouds. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 969–979.
- [Yi et al., 2021] Yi, L., Gong, B., and Funkhouser, T. (2021). Complete & label: A domain adaptation approach to semantic segmentation of LiDAR point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15363–15373.

- [Qin et al., 2019] Qin, C., Wang, L., Zhang, Y., and Fu, Y. (2019). Generatively inferential co-training for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0.
- [Saito et al., 2019] Saito, K., Ushiku, Y., Harada, T., and Saenko, K. (2019). Strong-weak distribution alignment for adaptive object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6956–6965.
- [Yihan et al., 2021] Yihan, Z., Wang, C., Wang, Y., Xu, H., Ye, C., Yang, Z., and Ma, C. (2021). Learning transferable features for point cloud detection via 3d contrastive co-training. *Advances in Neural Information Processing Systems*, 34:21493–21504.
- [Saltori et al., 2020] Saltori, C., Lathuilière, S., Sebe, N., Ricci, E., and Galasso, F. (2020). SF-UDA 3D: Source-Free Unsupervised Domain Adaptation for LiDAR-Based 3D Object Detection. In *2020 International Conference on 3D Vision (3DV)*, pages 771–780. IEEE.
- [Luo et al., 2021] Luo, Z., Cai, Z., Zhou, C., Zhang, G., Zhao, H., Yi, S., Lu, S., Li, H., Zhang, S., and Liu, Z. (2021). Unsupervised domain adaptive 3d detection with multi-level consistency. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8866–8875.
- [Peng et al., 2023] Peng, X., Zhu, X., and Ma, Y. (2023). Cl3d: Unsupervised domain adaptation for cross-lidar 3d detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 2047–2055.
- [Rozantsev et al., 2018] Rozantsev, A., Salzmann, M., and Fua, P. (2018). Beyond sharing weights for deep domain adaptation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(4):801–814.
- [Ganin and Lempitsky, 2015] Ganin, Y. and Lempitsky, V. (2015). Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning*, pages 1180–1189. PMLR.
- [Saito et al., 2018] Saito, K., Watanabe, K., Ushiku, Y., and Harada, T. (2018). Maximum classifier discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3723–3732.
- [Sankaranarayanan et al., 2018] Sankaranarayanan, S., Balaji, Y., Castillo, C. D., and Chellappa, R. (2018). Generate to adapt: Aligning domains using generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8503–8512.

- [Isobe et al., 2021] Isobe, T., Jia, X., Chen, S., He, J., Shi, Y., Liu, J., Lu, H., and Wang, S. (2021). Multi-target domain adaptation with collaborative consistency learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8187–8196.
- [Hong et al., 2018] Hong, W., Wang, Z., Yang, M., and Yuan, J. (2018). Conditional generative adversarial network for structured domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1335–1344.
- [Li et al., 2019] Li, R., Li, X., Fu, C.-W., Cohen-Or, D., and Heng, P.-A. (2019). Pu-gan: a point cloud upsampling adversarial network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7203–7212.
- [Zou et al., 2018] Zou, Y., Yu, Z., Kumar, B., and Wang, J. (2018). Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 289–305.
- [Lian et al., 2019] Lian, Q., Lv, F., Duan, L., and Gong, B. (2019). Constructing self-motivated pyramid curriculums for cross-domain semantic segmentation: A non-adversarial approach. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6758–6767.
- [Li et al., 2020] Li, G., Kang, G., Liu, W., Wei, Y., and Yang, Y. (2020). Content-consistent matching for domain adaptive semantic segmentation. In *European Conference on Computer Vision*, pages 440–456. Springer.
- [Ahmed et al., 2021] Ahmed, S. M., Raychaudhuri, D. S., Paul, S., Oymak, S., and Roy-Chowdhury, A. K. (2021). Unsupervised multi-source domain adaptation without access to source data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10103–10112.
- [He et al., 2021] He, J., Jia, X., Chen, S., and Liu, J. (2021). Multi-source domain adaptation with collaborative learning for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11008–11017.
- [Yao et al., 2021] Yao, X., Zhao, S., Xu, P., and Yang, J. (2021). Multi-source domain adaptation for object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3273–3282.
- [Zhang et al., 2023] Zhang, B., Yuan, J., Shi, B., Chen, T., Li, Y., and Qiao, Y. (2023). Uni3D: A Unified Baseline for Multi-dataset 3D Object Detection. *arXiv preprint arXiv:2303.06880*.



- [Zhang and Pless, 2004] Zhang, Q. and Pless, R. (2004). Extrinsic calibration of a camera and laser range finder (improves camera calibration). In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 3, pages 2301–2306.
- [Zhou and Deng, 2012] Zhou, L. and Deng, Z. (2012). Extrinsic calibration of a camera and a lidar based on decoupling the rotation from the translation. In *IEEE Intelligent Vehicles Symposium, Proceedings*, pages 642–648.
- [Geiger et al., 2012] Geiger, A., Moosmann, F., Car, Ö., and Schuster, B. (2012). Automatic camera and range sensor calibration using a single shot. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 3936–3943. Institute of Electrical and Electronics Engineers Inc.
- [Verma et al., 2019] Verma, S., Berrio, J. S., Worrall, S., and Nebot, E. (2019). Automatic extrinsic calibration between a camera and a 3D Lidar using 3D point and plane correspondences. *2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019*, pages 3906–3912.
- [Park et al., 2014] Park, Y., Yun, S., Won, C. S., Cho, K., Um, K., and Sim, S. (2014). Calibration between color camera and 3D LIDAR instruments with a polygonal planar board. *Sensors (Switzerland)*, 14(3):5333–5353.
- [Dhall et al., 2017] Dhall, A., Chelani, K., Radhakrishnan, V., and Krishna, K. M. (2017). LiDAR-camera calibration using 3D-3D point correspondences. *arXiv preprint arXiv:1705.09785*.
- [Beltrán et al., 2022] Beltrán, J., Guindel, C., de la Escalera, A., and García, F. (2022). Automatic extrinsic calibration method for lidar and camera sensor setups. *IEEE Transactions on Intelligent Transportation Systems*, 23(10):17677–17689.
- [Pusztai and Hajder, 2017] Pusztai, Z. and Hajder, L. (2017). Accurate Calibration of LiDAR-Camera Systems Using Ordinary Boxes. In *Proceedings - 2017 IEEE International Conference on Computer Vision Workshops, ICCVW 2017*, volume 2018-Janua, pages 394–402. Institute of Electrical and Electronics Engineers Inc.
- [Toth et al., 2020] Toth, T., Pusztai, Z., and Hajder, L. (2020). Automatic LiDAR-Camera Calibration of Extrinsic Parameters Using a Spherical Target. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 8580–8586. Institute of Electrical and Electronics Engineers Inc.
- [Zhou et al., 2018] Zhou, L., Li, Z., and Kaess, M. (2018). Automatic Extrinsic Calibration of a Camera and a 3D LiDAR Using Line and Plane Correspondences. In *IEEE International Conference on Intelligent Robots and Systems*, pages 5562–5569. Institute of Electrical and Electronics Engineers Inc.

- [Pandey et al., 2010] Pandey, G., McBride, J., Savarese, S., and Eustice, R. (2010). Extrinsic calibration of a 3D laser scanner and an omnidirectional camera. In *IFAC Proceedings Volumes (IFAC-PapersOnline)*, volume 7, pages 336–341. IFAC Secretariat.
- [Alismail et al., 2012] Alismail, H., Baker, L. D., and Browning, B. (2012). Automatic calibration of a range sensor and camera system. In *Proceedings - 2nd Joint 3DIM/3DPVT Conference: 3D Imaging, Modeling, Processing, Visualization and Transmission, 3DIMPVT 2012*, pages 286–292.
- [Levenberg, 1944] Levenberg, K. (1944). A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, 2(2):164–168.
- [Yang et al., 2018] Yang, Y., Feng, C., Shen, Y., and Tian, D. (2018). Foldingnet: Point cloud auto-encoder via deep grid deformation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 206–215.
- [Groueix et al., 2018] Groueix, T., Fisher, M., Kim, V. G., Russell, B. C., and Aubry, M. (2018). A papier-mâché approach to learning 3d surface generation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 216–224.
- [Sun et al., 2021] Sun, W., Tagliasacchi, A., Deng, B., Sabour, S., Yazdani, S., Hinton, G. E., and Yi, K. M. (2021). Canonical capsules: Self-supervised capsules in canonical pose. *Advances in Neural Information Processing Systems*, 34:24993–25005.
- [Park et al., 2019] Park, J. J., Florence, P., Straub, J., Newcombe, R., and Lovegrove, S. (2019). DeepSDF: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 165–174.
- [Chang et al., 2015] Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al. (2015). Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*.
- [Wu et al., 2015] Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. (2015). 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1912–1920.
- [Stutz and Geiger, 2018] Stutz, D. and Geiger, A. (2018). Learning 3d shape completion from laser scan data with weak supervision. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1955–1964.

- [Duggal et al., 2022] Duggal, S., Wang, Z., Ma, W.-C., Manivasagam, S., Liang, J., Wang, S., and Urtasun, R. (2022). Mending Neural Implicit Modeling for 3D Vehicle Reconstruction in the Wild. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1900–1909.
- [Bernardini et al., 1999] Bernardini, F., Mittleman, J., Rushmeier, H., Silva, C., and Taubin, G. (1999). The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):349–359.
- [Edelsbrunner et al., 1983] Edelsbrunner, H., Kirkpatrick, D., and Seidel, R. (1983). On the shape of a set of points in the plane. *IEEE Transactions on Information Theory*, 29(4):551–559.
- [Kazhdan et al., 2006] Kazhdan, M., Bolitho, M., and Hoppe, H. (2006). Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, volume 7.
- [Gu et al., 2020] Gu, J., Ma, W.-C., Manivasagam, S., Zeng, W., Wang, Z., Xiong, Y., Su, H., and Urtasun, R. (2020). Weakly-supervised 3d shape completion in the wild. In *European Conference on Computer Vision*, pages 283–299. Springer.
- [Kirillov et al., 2023] Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., et al. (2023). Segment anything. *arXiv preprint arXiv:2304.02643*.
- [Huang and Grizzle, 2019] Huang, J.-K. and Grizzle, J. W. (2019). Improvements to Target-Based 3D LiDAR to Camera Calibration. *IEEE Access*, 8:134101–134110.
- [Kim and Park, 2019] Kim, E. S. and Park, S. Y. (2019). Extrinsic calibration of a camera-LIDAR multi sensor system using a planar chessboard. In *International Conference on Ubiquitous and Future Networks, ICUFN*, volume 2019-July, pages 89–91. IEEE Computer Society.
- [Berrio et al., 2021] Berrio, J. S., Shan, M., Worrall, S., and Nebot, E. (2021). Camera-LIDAR integration: Probabilistic sensor fusion for semantic mapping. *IEEE Transactions on Intelligent Transportation Systems*.
- [Gomez-Donoso et al., 2020] Gomez-Donoso, F., Cruz, E., Cazorla, M., Worrall, S., and Nebot, E. (2020). Using a 3d cnn for rejecting false positives on pedestrian detection. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6. IEEE.
- [Ester et al., 1996] Ester, M., Kriegel, H.-P., Sander, J., Xu, X., et al. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pages 226–231.

- [Yuksel, 2015] Yuksel, C. (2015). Sample elimination for generating Poisson disk sample sets. In *Computer Graphics Forum*, volume 34, pages 25–32. Wiley Online Library.
- [Li et al., 2020] Li, E., Wang, S., Li, C., Li, D., Wu, X., and Hao, Q. (2020). SUSTech POINTS: A Portable 3D Point Cloud Interactive Annotation Platform System. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 1108–1115.
- [Chen et al., 2019a] Chen, K., Pang, J., Wang, J., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Shi, J., Ouyang, W., Loy, C. C., and Lin, D. (2019a). Hybrid task cascade for instance segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*.
- [Chen et al., 2019b] Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., Zhang, Z., Cheng, D., Zhu, C., Cheng, T., Zhao, Q., Li, B., Lu, X., Zhu, R., Wu, Y., Dai, J., Wang, J., Shi, J., Ouyang, W., Loy, C. C., and Lin, D. (2019b). MMDetection: Open MMLab Detection Toolbox and Benchmark. *arXiv preprint arXiv:1906.07155*.
- [Lin et al., 2014] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer.
- [Zhou et al., 2018] Zhou, Q.-Y., Park, J., and Koltun, V. (2018). Open3D: A Modern Library for 3D Data Processing. *arXiv:1801.09847*.
- [Team, 2020] Team, O. D. (2020). OpenPCDet: An Open-source Toolbox for 3D Object Detection from Point Clouds.  
<https://github.com/open-mmlab/OpenPCDet>.
- [Qi et al., 2018] Qi, C. R., Liu, W., Wu, C., Su, H., and Guibas, L. J. (2018). Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 918–927.
- [Zhou et al., 2019] Zhou, Y., Barnes, C., Lu, J., Yang, J., and Li, H. (2019). On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5745–5753.
- [Yifan et al., 2019] Yifan, W., Wu, S., Huang, H., Cohen-Or, D., and Sorkine-Hornung, O. (2019). Patch-based progressive 3d point set upsampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5958–5967.
- [You et al., 2022] You, Y., Diaz-Ruiz, C. A., Wang, Y., Chao, W.-L., Hariharan, B., Campbell, M., and Weinbergert, K. Q. (2022). Exploiting Playbacks in

- Unsupervised Domain Adaptation for 3D Object Detection in Self-Driving Cars. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 5070–5077. IEEE.
- [Hu et al., 2023] Hu, Q., Liu, D., and Hu, W. (2023). Density-Insensitive Unsupervised Domain Adaption on 3D Object Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17556–17566.
- [Parzen, 1962] Parzen, E. (1962). On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076.
- [Pang et al., 2021] Pang, Z., Li, Z., and Wang, N. (2021). SimpleTrack: Understanding and Rethinking 3D Multi-object Tracking. *arXiv preprint arXiv:2111.09621*.
- [Weng et al., 2020] Weng, X., Wang, J., Held, D., and Kitani, K. (2020). 3D Multi-Object Tracking: A Baseline and New Evaluation Metrics. *IROS*.
- [Chen et al., 2023] Chen, Z., Luo, Y., Huang, Z., Wang, Z., and Baktashmotlagh, M. (2023). Revisiting Domain-Adaptive 3D Object Detection by Reliable, Diverse and Class-balanced Pseudo-Labeling. *arXiv preprint arXiv:2307.07944*.
- [Yang et al., 2022] Yang, J., Shi, S., Wang, Z., Li, H., and Qi, X. (2022). ST3D++: Denoised Self-training for Unsupervised Domain Adaptation on 3D Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.
- [Berrio et al., 2022] Berrio, J. S., Shan, M., Worrall, S., and Nebot, E. (2022). Camera-LIDAR Integration: Probabilistic Sensor Fusion for Semantic Mapping. *IEEE Transactions on Intelligent Transportation Systems*, 23(7):7637–7652.
- [Jiang et al., 2018] Jiang, B., Luo, R., Mao, J., Xiao, T., and Jiang, Y. (2018). Acquisition of localization confidence for accurate object detection. In *Proceedings of the European conference on computer vision (ECCV)*, pages 784–799.
- [You et al., 2022] You, Y., Luo, K., Phoo, C. P., Chao, W.-L., Sun, W., Hariharan, B., Campbell, M., and Weinberger, K. Q. (2022). Learning to detect mobile objects from lidar scans without labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1130–1140.
- [Zhang et al., 2023] Zhang, L., Yang, A. J., Xiong, Y., Casas, S., Yang, B., Ren, M., and Urtasun, R. (2023). Towards Unsupervised Object Detection From LiDAR Point Clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9317–9328.

- [Oh et al., 2022] Oh, M., Jung, E., Lim, H., Song, W., Hu, S., Lee, E. M., Park, J., Kim, J., Lee, J., and Myung, H. (2022). TRAVEL: Traversable Ground and Above-Ground Object Segmentation Using Graph Representation of 3D LiDAR Scans. *IEEE Robotics and Automation Letters*, 7(3):7255–7262.
- [Lee et al., 2022] Lee, S., Lim, H., and Myung, H. (2022). Patchwork++: Fast and robust ground segmentation solving partial under-segmentation using 3D point cloud. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pages 13276–13283.
- [Zhang and Singh, 2014] Zhang, J. and Singh, S. (2014). LOAM: Lidar odometry and mapping in real-time. In *Robotics: Science and systems*, volume 2, pages 1–9. Berkeley, CA.
- [Houston et al., 2020] Houston, J., Zuidhof, G., Bergamini, L., Ye, Y., Chen, L., Jain, A., Omari, S., Iglovikov, V., and Ondruska, P. (2020). One thousand and one hours: Self-driving motion prediction dataset. *arXiv preprint arXiv:2006.14480*.
- [Bai et al., 2022] Bai, X., Hu, Z., Zhu, X., Huang, Q., Chen, Y., Fu, H., and Tai, C.-L. (2022). Transfusion: Robust lidar-camera fusion for 3d object detection with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1090–1099.