SENSORS AND DATA: REPRESENTATION TO SEMANTICS TO DEEP FEATURES

A Dissertation

by

PENG JIANG

Submitted to the Graduate and Professional School of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

May 2023

Major Subject: Mechanical Engineering

ABSTRACT

The field of robotics has rapidly expanded in recent years and has found its way into various sectors, including manufacturing, healthcare, and transportation. This growth is largely attributed to the advancements in deep learning techniques, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), which have made it possible to create intelligent robots capable of performing complex tasks independently. With the ability to interact with the environment and learn from experience, robots can now tackle previously daunting real-world problems like object recognition, natural language processing, and motion planning while also being able to adapt to changing conditions and optimize their performance.

This dissertation places significant emphasis on the role of perception sensors in enabling robots to understand and engage with their environment, particularly in off-road terrain. The research investigates several types of sensors, including cameras, LIDAR, and radar, and how they provide insights into a robot's surroundings. Additionally, the study explores different levels of sensor data representation, ranging from raw data to semantic information and deep features. The dissertation introduces an off-road dataset for semantic segmentation that includes semantic labels for raw data and proposes a benchmark for point cloud and image semantic segmentation. It also delves into various semantic segmentation problems for different types of sensors, including camera images, LIDAR point cloud, and raw RADAR. The research proposes a semantic segmentation framework for off-road image segmentation, a technique to transfer labels from one LIDAR point cloud dataset to another dataset, and a pipeline to transfer LIDAR semantic segmentation labels to radar data. The study also proposes a technique to learn cross-modal deep features using contrastive learning. Finally, the research employs higher-level information, such as semantic information and deep features, to address multi-modal extrinsic calibration for cameras-LIDAR and LIDAR-radar. The expected outcome of this research is to improve autonomous navigation in off-road environments, and the dissertation provides new resources and avenues for further research.

DEDICATION

To my family and friends.

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my advisor Dr. Srikanth Saripalli for his unwavering support and guidance throughout the research process. His encouragement, funding, and expert advice were instrumental in the completion of this dissertation. I am especially grateful for the freedom they gave me to pursue my own ideas and direction for the project.

I would also like to extend my thanks to the members of my dissertation committee, Dr. Dylan Shell, Dr. Dileep Kalathil, and Dr. Swaminathan Gopalswamy. Their feedback, suggestions, and expert knowledge were invaluable in shaping my research and providing constructive criticism that led to a stronger final product.

To my parents and family, thank you for your love, support, and encouragement. Your unwavering belief in my abilities and your willingness to provide me with the emotional and financial support I needed to pursue my academic goals have been a constant source of motivation for me.

To my friends, thank you for your unwavering support and for being a constant source of encouragement throughout my academic journey. Your words of encouragement, positive attitude, and willingness to lend a helping hand have been invaluable.

Finally, I would like to thank my labmates for their help and support. Their input, feedback, and willingness to lend a helping hand have been essential in completing my degree.

CONTRIBUTORS AND FUNDING SOURCES

Contributors

This work was supported by a dissertation committee consisting of Professor Srikanth Saripalli and Professor Swaminathan Gopalswamy of the J. Mike Walker ’66 Department of Mechanical Engineering , Professor Dylan Shell of the Department of Computer Science and Engineering and Professor Dileep Kalathilof the Department of Electrical and Computer Engineering.

Data annotation for Chapter 2 was provided by Philip Osteen and Maggie Wigness from DEVCOM Army Research Laboratory. In addition, Section 3.1 was done in cooperation with Kasi Viswanath, Kartikeya Singh, and Professor P.B. Sujit from the Department of Electrical Engineering and Computer Science at IISER Bhopal.

All other work conducted for the dissertation was completed by the student independently.

Funding Sources

Graduate study was supported by other outside source of funding.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

1. INTRODUCTION

Sensors are essential for enabling robots to perceive and comprehend the world around them, especially in off-road terrains. Different types of sensors are utilized in robotics, including proximity sensors, inertial measurement units (IMUs), and GPS sensors. However, perception sensors, such as cameras, lidar, radar, and sonar, provide critical information to the robot about its environment. This information includes the position and motion of objects, surface features, and other pertinent data, even in challenging off-road environments. Different types of perception sensors provide unique types of information and have complementary properties. Camera and Light Detection and Ranging (LiDAR) sensors are critical components of autonomous vehicles due to their ability to offer high-resolution color information and accurate spatial information, respectively. Cameras provide high-resolution color information, while LiDAR can provide precise spatial information at long ranges. Meanwhile, RADAR sensors have a longer wavelength and are less susceptible to small particles such as dust, fog, rain, or snow that can impair the performance of camera and LiDAR sensors. Additionally, RADAR has a longer detection range and the ability to penetrate materials, enabling the detection of objects beyond the line of sight of LiDAR sensors.

Based on different algorithm and process, the data can be represented in different levels. The raw data obtained from these perception sensors consists of numerical values or signals that represent the sensor's measurements or observations. This raw representation provides a foundation for algorithms and processing to extract more meaningful information from the data. Through, some basic processing, we can get information about the surrounding environemtn. Further processing and analysis are then applied to the raw data to obtain semantic representation, such as the position and motion of objects or surface features, that are more meaningful and interpretable for the robot. Alternatively, deep feature representation can be learned from the raw data using machine learning algorithms, which provide a more abstract and complex representation of the environment.

Perception sensors and the different types of representations that can be derived from their data play a critical role in a broad range of robotics applications, including off-road exploration and search and rescue missions, manufacturing, logistics, healthcare, and defense. This dissertation explores multi-modal sensor data at different levels: from raw data to semantic information to deep features.

1.1 Problem Statement and Related Work[*]

1.1.1 Summary of Problems and Contribution

As previously discussed, multi-modal sensors are essential for enabling robots to understand and interact with their environment. However, there are still many unanswered questions at each level of information that can be extracted from the raw sensor data. This dissertation aims to investigate various challenges in robotics as followed:

- **Multi-modal Off-road Semantic Segmentation Dataset**: Autonomous navigation systems using only LiDAR and INS perform poorly in off-road environments. Visual perception systems can provide complementary information for better navigation. There are limited datasets available for off-road navigation research, making it challenging to develop and test new models. We introduce RELLIS-3D, a new multimodal dataset with pixel-wise and point-wise annotations for off-road environments. The dataset challenges current state-of-the-art deep learning models, highlighting the need for further research in these unstructured environments.

- **Off-road Image Semantic Segmentation**: Off-road environments are challenging for autonomous driving due to their complex and varied textures and undefined boundaries. Existing methods for off-road semantic segmentation often struggle due to the unique features of off-road environments, such as illumination changes. Our contribution is a framework called OFFSEG that groups classes in off-road datasets into four categories to resolve class imbalance issues and achieve good results for fine semantic segmentation. Our approach can be extended to include other sub-classes and tested under different climatic and weather conditions.

[*]Part of the content are from our published paper [1][2][3][4][5]

- **LIDAR Semantic Segmentation Domain Adaptation**: Point clouds pose challenges for applying deep learning, leading to the development of several methods to address these issues. Most learning-based methods for semantic segmentation are supervised, but slight differences between training and test data can affect their performance. We propose a boundary-aware domain adaptation approach to improve the semantic segmentation of the lidar point cloud. We design a model that can extract domain shared and domain private features and utilize the learned boundary to refine segmentation results.

- **Off-road Radar Semantic Segmentation**: Radar data is challenging to use for semantic segmentation due to its sparsity and limited information. Previous methods used hand labeling or low-level features for classification but lacked rich information on object types. Some methods used weak supervision, but most research has focused on urban scenes. We propose a pipeline to transfer LiDAR semantic segmentation labels to radar and train an image-based model for off-road environments.

- **Cross-modal Deep Features Learning**: Due to their complementary properties, combining camera and LIDAR sensors are essential components of autonomous vehicles. While many methods have been explored in the literature to generate features for a single modality, finding unified cross-modal dense features is still challenging. The existing patch-based methods require dense point clouds to learn, and no unified neural network structures are versatile to process all modalities. We propose using a U-Net variant of the PointNet++ architecture for the point cloud and a U-Net CNN architecture for images to overcome these difficulties. Additionally, we propose the Tuple-Circle loss function for cross-modal deep contrastive learning.

- **Camera-LIDAR Extrinsic Calibration**: The existing methods for Camera-LIDAR calibration suffer from various limitations, such as the need for controlled environments or specifically designed targets, time-consuming manual feature engineering, and lack of generalization to new scenarios. We propose a novel online, targetless, extrinsic calibration algorithm that leverages semantic information for cross-modal data association. The proposed algorithm uses a fully differentiable mutual information neural estimate to estimate mutual information between dense semantic labels derived from the data.

- **Radar-LIDAR Extrinsic Calibration**:The extrinsic calibration between LIDAR and RADAR sensors is challenging due to the limited resolution and information loss in the third dimension of RADAR measurements. we propose a method to calibrate LIDAR and RADAR using a small Multi-Layer Perceptron (MLP) to model the target as a collection of RADAR return energy. The model is trained to learn the correlation between the RADAR sensor pose relative to the target and the return energy, allowing it to be used for calibration.

1.1.2 Multi-modal Off-road Semantic Segmentation Dataset

Autonomous navigation systems that rely solely on LIDAR and an inertial navigation system (INS) have been shown to perform poorly in off-road environments [6, 7, 8]. These systems leverage geometric information but need a higher-level semantic understanding of the environment that could make path planning and navigation more efficient. For example, bushes are identified as obstacles by LIDAR. However, these parts of the environment may be traversable for larger platforms. The importance of semantic awareness has led to the emergence of visual perception systems that directly feed information to navigation systems to complement depth sensors [6, 9, 10, 11, 12]. LIDAR provides highly accurate 3D information, is not affected by varying illumination, and adds reflectance information to characterize object surfaces uniquely. In contrast, cameras provide dense color and texture information to obtain fine-grained semantic information.

In the past decade, numerous datasets have been published for scene understanding in autonomous vehicles and robots. The majority of these datasets focus on urban environments and on-road navigation and typically provide only 2D annotations for RGB camera images, such as CamVid [13], Cityscapes [14], Mapillary Vistas [15], D2-City [16], and BDD100k [17]. However, some datasets have incorporated multiple sensor modalities, such as KITTI [18], nuScenes[19], and A2D2 [20]. To further enhance the quality of data available for research, the SemanticKITTI dataset [21] has added extensive semantic annotations to the LIDAR subset of the KITTI dataset. With the availability of these datasets, research on autonomous driving and robots has made great strides in recent years.

Compared to the plethora of available urban environment datasets, relatively few datasets are available for off-road navigation. Off-road environments are characterized by unstructured class boundaries, uneven terrain, and irregular features, making it challenging to transfer models from other types of environments directly. Additionally, there are significant differences in class distributions across distinct off-road environments. Table 1.1 provides an overview of existing off-road datasets and their data and annotation characteristics. For example, the RUGD dataset [22] offers RGB images for semantic segmentation in off-road environments with detailed annotations but lacks multiple sensor modalities. The Freiburg Forest dataset [23] provides multi-spectral images but does not include additional modalities such as LIDAR and has limited annotated data. The YCOR dataset [6] offers both images and point clouds, but only the images are annotated. The NREC Agricultural Person-Detection Dataset provides annotations for person detection in off-road environments, but only for a single class. Dabbiru et al. [24, 25] have presented a framework for generating simulated and annotated point cloud data for LIDAR semantic segmentation, which could potentially be used for transfer learning with real datasets like RELLIS-3D. Despite these existing datasets, there is still a gap in the data available for off-road navigation research. The lack of available data, especially with multiple sensor modalities, presents a significant challenge for researchers in the field and highlights the importance of efforts to gather and curate new datasets.

Table 1.1: Overview of off-road datasets. Ours is by far the largest dataset with multiple modalities.

Name	Sensors	# Annotations[1]	# Classs[2]	Annotation Type	Modality
RUGD [22]	camera	7546	24	pixel-wise	RGB
DeepScene [23]	camera	366	6	pixel-wise	RGB, Depth, NIR, NRG, NDVI, EVI
Pezzementi et al [26]	camera	95000	1	bounding box	RGB
YCOR [6]	camera	1076	8	pixel-wise	RGB
Dabbiru et al [24]	simLiDAR	2743	6	point-wise	Point Cloud
Ours	camera, LIDAR	6235/13556	20	pixel/point-wise	RGB, Point Cloud

1.1.3 Semantic Segmentation

Robots need to understand the objects and structures present in the environment to navigate and perform tasks effectively in off-road environments. This involves developing algorithms for semantic segmentation, which assigns a label to each pixel or point in the data obtained from images, lidars, and radars. The resulting information can be used for tasks such as object recognition and scene understanding, making it useful for various applications, such as inspection, exploration, rescue, and reconnaissance missions.

1.1.3.1 Image semantic segmentation

Off-road environments are typically characterized by rich and varied textures, with undefined boundaries and less detailed than urban areas. The non-uniform terrain description in these environments challenges robust autonomous driving systems. On the other hand, on-road driving has received more attention in terms of datasets for segmentation and semantic scene understanding, with several state-of-the-art benchmarks available for urban environments. Unlike urban environments, off-road environments have unstructured class boundaries, uneven terrain, strong textures, and irregular features that hinder the direct transfer of models between environments. Additionally, there are significant variations in class distributions across different off-road environments.

Early research on off-road semantic segmentation mainly focused on coarse-grained methods, which are typically formulated as binary classification problems. One such method is the road detection algorithm for front-view monocular cameras using the Road Probabilistic Distribution Model (RPDM) and online learning, as described in [27]. Another approach is vanishing point detection for off-road road detection, which involves estimating the vanishing point associated with the main part of the road, followed by segmenting the corresponding road area based on the detected vanishing point, as proposed in [28]. In [29], an approach to road recognition based on Kalman filtering and EM theory is presented, which results in a fast recursive filter that can adaptively track unpaved roads. However, these early methods have limitations in handling fine-grained semantic segmentation due to their inability to generate strong feature representations.

In recent years, advances in deep learning have led to the development of methods for fine-grained semantic segmentation. For instance, Rothrock et al. [30] implemented Fully Convolutional Networks (FCN) for terrain classification of the Martian surface, while Sgibnev et al. [31] focused on the practical application of modern lightweight architectures for mobile robotic systems. However, directly applying existing segmentation networks for off-road semantic segmentation often results in performance degradation due to intrinsic problems in such environments, such as illumination changes. To address this challenge, Jin et al. [32] proposed a built-in memory module for semantic segmentation. Additionally, Guan et al. [33] introduced GANav, a novel group-wise attention mechanism that identifies safe and navigable regions in off-road terrains and unstructured environments from RGB images. Despite the recent advances in off-road semantic segmentation, these methods often overlook a crucial aspect of off-road driving. Unlike on-road scenes, where detailed classes like signboards, traffic lights, etc., are necessary, off-road environments require fewer features. Therefore, it is possible to group classes in off-road datasets into four categories: traversable, non-traversable, obstacles, and sky, based on their semantic contributions to the environment. The sky class includes the region present in the sky, while the traversable class comprises all possible traversable regions in the dataset. The non-traversable class covers all surface regions that are not traversable and do not act as obstacles during off-road navigation. Lastly, the obstacle class contains all possible obstacles in the dataset. Grouping the classes into these four categories adequately resolve the class imbalance issue. Additional details, such as dirt, mud, gravel, etc., can be included in the traversable class to aid in determining the drivable path during autonomous off-road driving.

1.1.3.2 LIDAR semantic segmentation

Understanding the scene is crucial for the development of autonomous robotics and vehicles. The semantic information of the scene can be utilized for navigation, decision-making, and semantic mapping. Cameras and LIDAR are commonly used in autonomous vehicles for obtaining visual data. Although cameras provide high-resolution color and texture information, they are sensitive to variations in illumination, making semantic segmentation for LIDAR scans critical. The recent advancement of deep learning technology has led to significant progress in 3D semantic segmentation using LIDAR data [34]. Furthermore, the emergence of several LIDAR datasets from autonomous driving companies, such as [20, 16, 21, 35], has further motivated research in this field.

However, the irregularity and lack of structure in point clouds pose challenges for applying deep learning directly to them. The simplest form of point cloud representation is to use dense voxel grids and apply 3D convolution to predict the results [36, 37, 38]. However, this approach has redundancy issues, and 3D convolution is computationally inefficient. To address these issues, researchers have worked to reduce the computation cost and improve performance [39, 40, 41, 42]. Point cloud-based methods [43, 44, 45, 46] attempt to operate directly on the point set to reduce computation cost and achieve good performance. Point clouds can also be represented as graphs and meshes to explore the relationships between points further. Additionally, a LIDAR scan can be projected onto 2D space and represented as a dense range map without significant loss of information. This projection enables 2D convolution to be directly applied to the LIDAR scan. Projection-based methods [47, 48, 49] directly apply 2D convolution to the LIDAR scan and achieve high accuracy.

Most current learning-based methods for semantic segmentation are supervised, requiring a vast amount of annotated data for training. However, trained models' performance can suffer from slight differences between the test and training data, such as variations in sensor setups, manufacturing differences, and scene contents. The problem of adapting a model trained on labeled data to work on new data without annotating new data is an unsupervised domain adaptation problem. This problem exists in both 2D image data and 3D point cloud data, but the characteristics of sparsity, irregularity and unstructured distribution make this problem more difficult for the point cloud. Unsupervised Domain adaptation (UDA) is a subfield of transfer learning that aims to learn a discriminative model in the presence of domain shift between domains. With the development of deep learning, many UDA methods have been proposed. Hoffman et al. [50] introduced the CycleGAN mechanism into the domain adaptation field and proposed a discriminatively-trained cycle-consistent adversarial domain adaptation model (CyCADA). In the specific case of domain adaptation for point cloud, several works have been designed to address this problem. Wu et al. [51] utilized geodesic correlation alignment to perform adaptation between real and synthetic LIDAR data. Rist et al. [52] designed a voxel-based architecture to extract features of the LIDAR point cloud and used a supervised training methodology to learn traversable features. Salah et al. [53] converted LIDAR into bird-eye view images and used a CycleGAN method to adapt the synthetic LIDAR domain to real domains. Wang et al. [54] also used the bird-eye view as a representation and a two-scale model to perform cross-range adaptation for LIDAR 3D object detection. Qin et al. [55] proposed a multi-scale 3D adaptation network to jointly align global and local features at multi-levels. Yi et al. [56] represented the point cloud as voxels and addressed the LIDAR sampling domain gap for cars and pedestrians by converting the problem into a surface completion task. Zhao et al. [57] used ePointDA to bridge the domain shift at the pixel level by explicitly rendering dropout noise for synthetic LIDAR and at the feature level by spatially aligning the features between different domains. Langer et al. [58] fused sequential labeled LIDAR scans into a dense mesh and created semi-synthetic data to perform training. Jaritz et al. [59] explored how to learn from multi-modality and proposed cross-modal UDA (xMUDA) to adapt 2D images' semantic information to 3D point clouds. However, it is important to note that some of the methods mentioned above are designed to work with a limited number of object classes, such as cars and pedestrians. Additionally, the adapted models resulting from these methods may only be effective on the target dataset and may not generalize well to the source datasets. These limitations can hinder the practical application of the models.

1.1.3.3 RADAR semantic segmentation

Compared to other sensing modalities, such as cameras and LIDAR, radar data is more sparse and provides only two-dimensional information, which presents a significant challenge for semantic segmentation. Most state-of-the-art semantic segmentation methods require manual labeling of ground truth data, which is nearly impossible to achieve for raw radar data. As a result, radar data has received less attention in semantic segmentation research. Early efforts in radar semantic segmentation used occupancy grid methods to discriminate between free and occupied space but failed to provide rich information about object types [60]. To overcome this, Lombacher et al. [61] used a Fully Convolutional Neural Network (FCNN) to segment a radar scan based on the probability of occupancy, and hand labels were used to classify objects. Scheiner et al. [62] used low-level radar cube data and extracted 50 different features from the measurement data to perform multi-class classification using random forest and long short-term memory (LSTM) classifiers. Building on this work, Scheiner et al. [63] increased the number of features to 98 and used recurrent neural networks (RNNs) and classifier ensembles to achieve higher accuracy in classification results. Instead of using hand-crafted features, Schumann et al. [64] treated radar point clouds as a regular point clouds and modified PointNet++ to perform semantic segmentation directly. To avoid the time-consuming nature of hand labeling, Kaul et al. [65] used weak supervision to label radar data by semantically segmenting camera streams and combining the results with LIDAR range measurements to provide labeled images. However, most of the aforementioned work has focused on urban scenes. Compared to urban scenes, radar data captured in off-road environments is more unstructured and less interpretable, making manual labeling impossible, let alone using supervised learning methods to train models.

1.1.4 Cross-modal Deep Features

A camera is capable of providing high-resolution color information but is sensitive to changes in illumination and lacks direct spatial measurement. In contrast, a LIDAR can provide accurate spatial information at long ranges and is robust to illumination changes, but its resolution is much lower than that of the camera, and it does not measure color. However, combining information from two sensors is always challenging due to differences in representation (pixels vs. points) and information (visual vs. geometric), among other factors. Common methods for integrating information from multiple sensors involve manually defining common features across the two modalities, such as edges, gradients, and semantics. This is widely used in traditional calibration methods [5], which is usually the first step when using multiple sensors. Other applications include mapping, localization, and SLAM [66, 67, 68]. Therefore, it is important to define common features across sensor modalities in all applications.

With the development of deep learning, many methods have been explored in the literature to generate features for a single modality, such as image [69] or point cloud [70]. Although these features can be used to find inter-modal correspondences, they cannot be directly applied to cross-modal situations. A few works have focused on cross-modal descriptors, and most of them are patch-based methods that require dense point clouds to learn [66, 67, 68]. One of the earliest methods for cross-modal feature learning is presented in [71], where a CNN is used to map an image to a point in the embedding space, which is created by using a 3D shape similarity measure. The embedding allows for cross-view image retrieval, image-based shape retrieval, as well as shape-based image retrieval. In [72], the authors use the triple loss and combine the teacher/student method to create a shared 2D-3D embedding space for image-based global localization in LIDAR-maps. [73] utilizes a Generative Adversarial Network (GAN) to extract cross-domain symmetric place descriptors for localizing a single camera with respect to a point cloud map for indoor and outdoor scenes. In [74], the authors propose a method for learning robust local feature representation by leveraging both textures from images and geometric information from point clouds. In [68], an end-to-end deep network architecture is presented to jointly learn the descriptors for 2D and 3D keypoints from images and point clouds, respectively. As a result, the approach is able to directly match and establish 2D-3D correspondences from the query image and 3D point cloud reference map for visual pose estimation. [66] proposes LCD, which uses a dual auto-encoder neural network and triplet loss to learn a shared latent space representing the 2D and 3D data. However, the method requires a point cloud with RGB information. Finding unified cross-modal dense features is more challenging than finding unified inter-modal features. One reason is that there are no unified neural network structures that are versatile enough to process all modalities. Images are always processed by 2D convolutional networks. Meanwhile, various architectures can process point clouds according to different representations, making it difficult to train two models simultaneously. Secondly, compared with other 3D scanners, LIDAR creates sparser point clouds, making this problem much more challenging. Moreover, [67] and our preliminary tests show that using the existing loss functions is challenging for point clouds and image cross-feature learning.

In summary, while there have been some successful attempts at cross-modal feature learning, it is still a challenging problem. Many existing methods are patch-based and require dense point clouds, making them less effective when dealing with sparse point clouds. Additionally, the lack of unified neural network structures and the difficulty of training two models simultaneously remain challenging issues. Further research is needed to address these challenges and develop more effective cross-modal feature learning methods.

1.1.5 Mulit-modal Sensor extrinsic Calibration

1.1.5.1 CAMERA and LIDAR Calibration

Calibrating multi-sensor systems is a critical task in robotics. LIDAR and camera calibration are among the most extensively studied areas in the field of calibration research. Traditional 3D-LIDAR and camera calibration methods can be classified into offline, non-learning, and target-based methods. Target-based methods, inspired by target-based 2D-LIDAR camera extrinsic calibration methods [75, 76], utilize known objects within the sensors' common Field of View (FoV) with features being geometric features of the objects, such as edges [77, 78], planes [79, 80, 81]. Although target-based offline methods can provide accurate results and are easier to evaluate due to experimental controls, unpredictable environmental conditions (e.g., heat, vibration, impact, etc.) can degrade calibration accuracy over time, and therefore sensors should also be calibrated online during operation [82].

Targetless methods can further be divided into non-learning and learning-based methods. Learning-based methods rely on neural network models to learn relevant features without manual definition [83, 84, 85, 86]. However, these methods require a dataset with ground truth calibration parameters for training the model, which may not always be available. Moreover, neural network models can struggle with transfer learning from one dataset to another [78, 87]. Targetless non-learning methods make use of pre-defined features, such as edges [82, 77], gradients [88], surface reflectivity, and semantic information. For instance, Nagy et al. [89] employed a structure from motion (SfM) pipeline to generate points from images and then registered them with LIDAR points to obtain a basic calibration, which was further refined using semantic information. Zhu et al. [90] used semantic masks from images to construct a height map, encouraging laser points to fall on pixels labeled as obstacles. Wang et al. [91] proposed a new metric to calibrate LIDAR and camera by reducing the distance between misaligned points and pixels, leveraging semantic information from both image and point cloud.

1.1.5.2 RADAR and LIDAR Calibration

Calibrating RADAR with other sensors is more complicated than the camera and LIDAR calibration. The main challenges in calibrating RADAR with other sensors are recovering the missing third-dimension information and making the reflectors visible to both sensors. To address these challenges, El Natour et al. [92] proposed a 3D reconstruction method based on sensor geometry and a calibration facility using a Luneburg lens and multiple corner reflectors with different colors for visual detectability. Persic et al. [93] designed a compact target with a triangular trihedral corner reflector and a flat styrofoam triangle board that could be detected by both RADAR and LIDAR and proposed two-step optimizations (Reprojection Error Optimization and FOV Optimization) for 6-DOF calibration between RADAR and LIDAR. In a subsequent paper, Persic et al. [94] added chequerboard patterns to the triangle board to enable calibration between RADAR, LIDAR, and CAMERA, using RCS Optimization instead of FOV Optimization for a more robust result. Domhof et al. [95] designed a styrofoam board with four circular holes and placed a corner reflector behind it, allowing sparse LIDAR beams to detect the holes. The open-source tool *radar_to_lidar_calib* [96] uses correlative scan matching via the Fourier Mellin transform [97] to estimate the translation and rotation between the LIDAR and the Navtech RADAR, while OpenCalib [98] can calibrate LIDAR, CAMERA, and RADAR without targets. However, these models have limitations as they only consider a subset of data and do not account for various factors that impact the RADAR return signal, such as shape, material, direction, and the RADAR's power and frequency.

1.2 Dissertation overview

This dissertation explores multi-modal sensor data at different levels: from raw data to semantic information to deep features. We start by creating an off-road dataset with semantic labels for raw data. For semantic representation, we focus on off-road image semantic segmentation and study semantic transfer learning for images. We also examine semantic domain transfer learning for LiDAR point clouds and explore the transfer of semantic labels from radar to LiDAR. In terms of deep features, we investigate two methods for learning cross-modal deep features through contrastive learning between images and LiDAR point clouds. Finally, we address multi-modal calibration between CAMERA-LIDAR and LIDAR-RADAR using higher-level information.

- **Chapter 2** introduce RELLIS-3D, a multimodal dataset collected in an off-road environment. The dataset contains annotations for 13,556 LiDAR scans and 6,235 images and presents challenges to existing algorithms related to class imbalance and environmental topography. We evaluate the current state-of-the-art deep learning semantic segmentation models on this dataset. The experimental results show that RELLIS-3D presents challenges for algorithms designed for segmentation in urban environments. This novel dataset provides the resources needed by researchers to continue to develop more advanced algorithms and investigate new research directions to enhance autonomous navigation in off-road environments.

- **Chapter 3** explores different semantic segmentation problems for different types of sensors. We propose a framework for off-road semantic segmentation (OFFSEG) that involves a pooled class semantic segmentation with four classes (sky, traversable region, non-traversable region, and obstacle) using state-of-the-art deep learning architectures. We also introduce a color segmentation methodology to segment specific sub-classes (grass, puddle, dirt, gravel, etc.) from the traversable region to understand a scene better. Additionally, we present a boundary-aware domain adaptation model for LiDAR scan full-scene semantic segmentation (LiDARNet). The model can extract both the domain private features and the domain shared features with a two-branch structure, and we embedded Gated-SCNN into the segmentor component of LiDARNet to learn boundary information while learning to predict full-scene semantic segmentation labels. Moreover, we describe how we transfer LiDAR semantic labels to radar data without manually labeling the data by using a pre-trained model to create semantic labels for the LiDAR point cloud and then creating a semantic point cloud map that projects the semantic labels from the point cloud map onto radar data. Finally, we use the generated model to train a semantic segmentation model on radar data.

- **Chapter 4** treats learning cross-modal features as a dense contrastive learning problem. We propose a Tuple-Circle loss function for cross-modality feature learning. We develop a variant of widely used PointNet++ architecture for point cloud and U-Net CNN architecture for images to learn good features and maintain generality. Moreover, we conduct experiments on a real-world dataset to show the effectiveness of our loss function and network structure. We demonstrate that our models learn information from both images and LiDAR by visualizing the features.

- **Chapter 5** proposes SemCal, a method that uses semantic information for a LiDAR and camera system. We leverage a neural information estimator to estimate the mutual information (MI) of semantic information extracted from each sensor measurement, facilitating semantic-level data association and introducing a semantic-based initial calibration method using 2D MI-based image registration and Perspective-n-Point (PnP) solver. This chapter also presents a novel solution for 3D RADAR-LIDAR calibration in autonomous systems. The method employs simple targets to generate data, including correspondence registration and a one-step optimization algorithm. The optimization aims to minimize the reprojection error while utilizing a small multi-layer perception (MLP) to perform regression on the return energy of the sensor around the targets. The proposed approach uses a deep learning framework such as PyTorch and can be optimized through gradient descent.

- **Chapter 6** concludes the thesis and discusses potential avenues for future work.

2. MULTI-MODAL OFF-ROAD DATASET*

In the first part of this dissertation, we aim to emphasis the significance of raw data in machine learning algorithms, particularly those used in off-road robotics. To this end, we will focus on creating high-quality datasets that capture inputs from multiple multi-modal sensors, such as LIDAR, and camera. With the increasing use of multiple sensors, there is a growing need for large-scale, diverse datasets to support the development of robust scene representation models. Moreover, we will enhance our datasets by annotating both LIDAR point clouds and camera images with semantic labels, allowing us to evaluate various semantic segmentation algorithms and establish a benchmark for future reference.

2.1 Introduction to RELLIS-3D

To support the development of reliable autonomous navigation systems in off-road environments, we introduce RELLIS-3D, a new dataset that provides multi-modal information. The dataset was captured using a Clearpath Robotics Warthog platform, and all data was recorded in the Ground Research facility at Texas A&M University's Rellis Campus. This off-road environment offers a diverse range of features, including runways, aprons, forests, bushes, pastures, lakes, and varied terrain (as shown in Fig. 2.1).

The RELLIS-3D dataset is comprised of a large set of raw sensor data synchronized with Precision Time Protocol (PTP), including color camera images, laser scans, high-precision global positioning measurements, inertial measurement from a combined Global Positioning and Inertial Navigation System (GPS/INS), and depth images from a 3D stereo camera. This dataset is a valuable resource for the research community, as it is the first multi-modal off-road navigation dataset that provides synchronized raw sensor data and a large number of ground truth annotations. By including the full set of raw autonomy data, we facilitate additional algorithms, such as those that fuse visual and inertial/depth data, to be developed and tested without any new data collection.

*The major content of this chapter was published in [1]

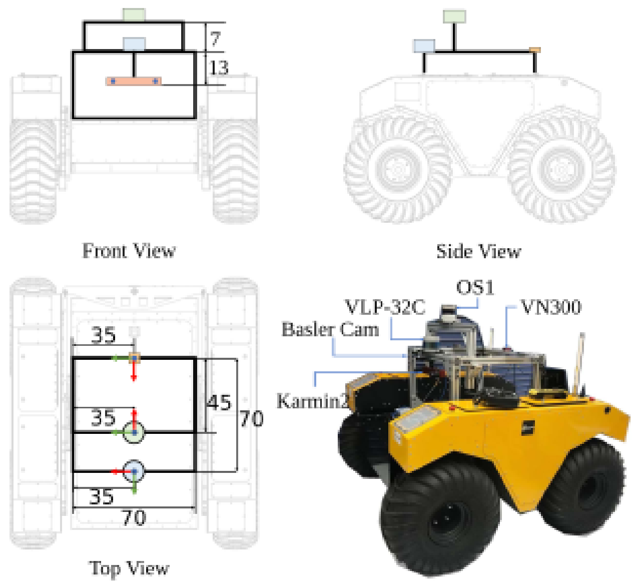


Figure 2.1: Warthog Platform Configuration. Illustration of the dimensions and mounting positions of the sensors with respect to the robot body. (Units: cm)

The major contributions of RELLIS-3D are:

- Five sequences of synchronized sensor data, including RGB camera images, LIDAR point clouds, stereo images, high-precision GPS measurements, and IMU data, are released in the Robot Operating System (ROS) bag format.
- The dataset provides 6,235 pixel-wise image annotations and semantic labels for 13,556 full LIDAR point cloud scans.
- RELLIS-3D includes training, validation, and testing splits, and an initial analysis using state-of-the-art image and point cloud semantic segmentation algorithms. These results highlight the challenges of semantic segmentation for off-road data and help identify open areas for further research that can be advanced by the RELLIS-3D dataset.

2.1.1 Data Description

The dataset includes five different traversal sequences, each recorded on a non-paved trail of the Ground Research facility. Three of the sequences were recorded on the first trail, which is

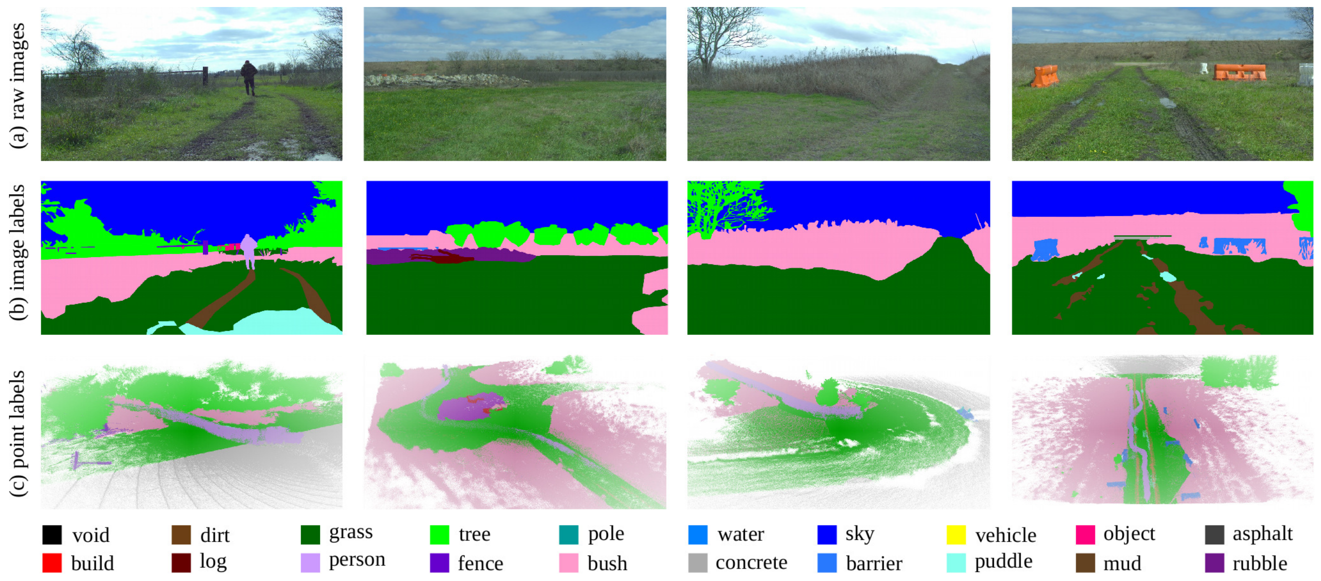


Figure 2.2: Ground truth annotations examples provided in the RELLIS-3D dataset. Images are densely annotated with pixel-wise labels from 20 different visual classes. LiDAR scans are point-wise labeled with the same ontogloy.

covered with bushes and sparse trees and differ in the direction the robot was moving and the day the data was collected. Another sequence captures the environment of the second trail that passes through a pasture and traverses a forested area, while the last sequence was recorded on a hill surrounded by a lake and a highway. The sequences were collected by teleoperating the robot to follow the trails, and each sequence includes approximately five minutes of data.

With the goal of providing multi-modal data to enhance autonomous off-road navigation, the authors defined an ontology of object and terrain classes. The ontology largely derives from the RUGD dataset, but also includes unique terrain and object classes not present in RUGD. These unique classes include mud, man-made barriers, and rubble piles, as well as a finer-grained class structure for water sources, i.e., puddle and deep water. These classes present different traversability scenarios for most robotic platforms and provide a more complete picture of the challenges faced by autonomous robots in off-road environments. Overall, the dataset includes 20 classes, including a void class, as seen in Fig. 2.2.

2.1.2 Annotations

The pixel-wise image annotations were provided by Appen[*], a company specializing in crowdsourcing for training data annotation. To ensure consistency in annotations, work was assigned to trained annotators and a single annotator was assigned a sequence of frames from a single video sequence. Moreover, the annotations from the crowdsourcing platform underwent several rounds of in-house verification to correct any missing, incorrect, or inconsistent labels.

In order to make the ground truth labeling process more manageable, the camera stream was downsampled to 5Hz. Furthermore, annotations were not duplicated for frames where the robot was stationary, leading to a total of $6,235$ images with pixel-wise annotations.

The point-wise annotations for 3D point clouds were initialized by projecting the more than $6,000$ image annotations onto point clouds using the camera-LIDAR calibration. The annotations were then refined using the 3D point cloud annotation application provided by SemanticKITTI[21]. Given the importance of LIDAR scan alignment for quality annotations, we first registered and closed loops in the sequences using a SLAM system [99]. This process outputted each scan's position based on the SLAM results. Through this process, $13,556$ scans received full point-wise annotations, with each scan including up to $13,056$ points.

2.1.3 Dataset Statistics

Figures 2.3 and 2.4 show the class distribution breakdown for image and point cloud annotations, respectively. The class distribution among both modalities is highly imbalanced. For image annotations, sky, grass, tree, and bushes make up 94% of the total labeled pixels. Among the LIDAR data, grass, tree, and bushes make up 80% of the total point labels. Differences in resolution, viewing angles, and sensor mechanism leads to the divergence in label distributions between the image and point cloud data. For example, because of the sensor mechanism, LIDAR is unable to detect sky, but because the LIDAR has a 360° viewing angle it picks up more person labels than the imagery since human operators usually followed the robot during data collection.

[*]https://appen.com/

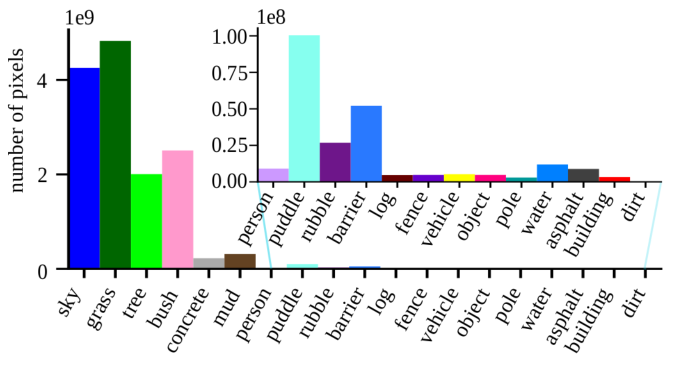


Figure 2.3: Image Label distribution. The sky, grass, tree and bush constitute the major classes.

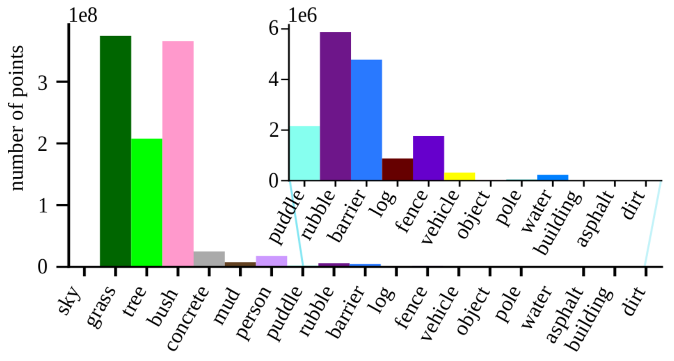


Figure 2.4: Point Cloud Label distribution. The grass, tree, and bush also dominate the population.

The non-uniform class distribution present in RELLIS-3D is common among datasets used for semantic segmentation [100, 15, 21]. Moreover, class imbalance is a problem that perception algorithms will encounter upon deployment. Although imbalanced class distributions exist in almost all current available urban datasets, the overall class distributions across distinct urban datasets are quite similar [21, 15]. However, the class imbalances in off-road environments are highly dependent on the particular environment, and the class imbalance within a dataset is more severe, making the semantic segmentation of rare classes more challenging than for urban environments.

2.2 Semantic Segmentation Benchmark

2.2.1 Evaluation Metrics

For semantic segmentation, we evaluate algorithm performance with the widely used mean intersection-over-union (mIoU) metric [101], given by

$$mIoU = \frac{1}{C}\sum_{c=1}^{C}\frac{TP_c}{TP_c + FP_c + FN_c} \tag{2.1}$$

where TP_c, FP_c and FN_c represent the number of true positive, false positive and false negative predictions for class c, and C is the number of classes.

2.2.2 Image Semantic Segmentation

We provide an evaluation of 2D image semantic segmentation on our dataset using two state-of-the-art architectures: HRNETV2+OCR [102, 103] and Gated-SCNN [104].

HRNETv2+OCR uses the High-Resolution Network (HRNet)[102] as its backbone and the Object-Contextual Representations (OCR) model [103] to explore the object-contextual representation of each pixel. HRNet maintains high-resolution representations throughout the whole model, unlike most other backbones that downsample input first and then upsample the features. OCR improves the pixel representation by aggregating the pixel features lying in the object region.

Gated-SCNN[104] is a two-stream CNN architecture for semantic segmentation that explicitly processes shape information in a separate branch, yielding a classical segmentation stream and a parallel shape stream. The architecture uses the higher-level activation in the classical stream to predict semantic segmentation and the low-level activation in the shape stream to abstract the shape information.

For this experiment, we split RELLIS-3D into a training set with 3,302 images, a validation set with 983 images, and a testing set with 1,672 images. When creating these splits, we tried to keep the training set large, while creating a testing set that was diverse including both similar and dissimilar scenarios seen in the training set. The image experiment uses 19 classes[†] (including void) for training and testing.

Table 2.1 and Fig. 2.5 shows the results of the image segmentation evaluation, which fall short of expectation. The Gated-SCNN model achieved only 52.92% mIoU, while it reached 74.7% mIoU [104] on Cityscapes. HRNet+OCR achieved 51.55% mIoU, but reached 81.1% mIoU on Cityscapes. We believe the performance degradation is mainly caused by our dataset's serious class imbalance. This is supported by comparing Fig. 2.3 and Fig. 2.5. Notice that the classes log, pole, water, and building have the lowest IoUs, and also represent the classes with fewest labels. These incorrect detections are of importance as they affect navigation decisions; for example, water is not traversable, unlike puddle, and human-made poles in an off-road area might provide warning information. Beyond the imbalance, the off-road environment's unclear boundaries also cause problems for both algorithms and human labelers. For humans, the indefinite boundaries make annotation difficult as compared with an urban roadway. The GSCNN algorithm utilizes boundary information to help perform segmentation, but this might lead to performance degradation for classes such as bush, grass, and trees with unclear boundaries (see Table 2.1). For these classes, perfect boundary segmentation might not be necessary, but this problem inspires us to design new algorithms that can focus on the boundaries of specific classes, such as human-made objects and water.

2.2.3 LIDAR Point Cloud Semantic Segmentation

There are two types of deep learning methods for LIDAR Point Cloud: point-based methods and the projective method [34]. We provide an evaluation of point cloud semantic segmentation on our dataset using two state-of-the-art architectures: SalsaNext [48] and KPConv [45].

SalsaNext is an uncertainty-aware semantic segmentation model for full 3D LIDAR point cloud in real-time. SalsaNext has an encoder-decoder architecture and works on projected LIDAR point cloud data. The encoder introduces a new residual dilated convolution stack with gradually increasing receptive fields, while the decoder uses the pixel-shuffle layer to recover the resolution

[†]We omit the dirt class in this evaluation because it is extremely sparse in the annotations.

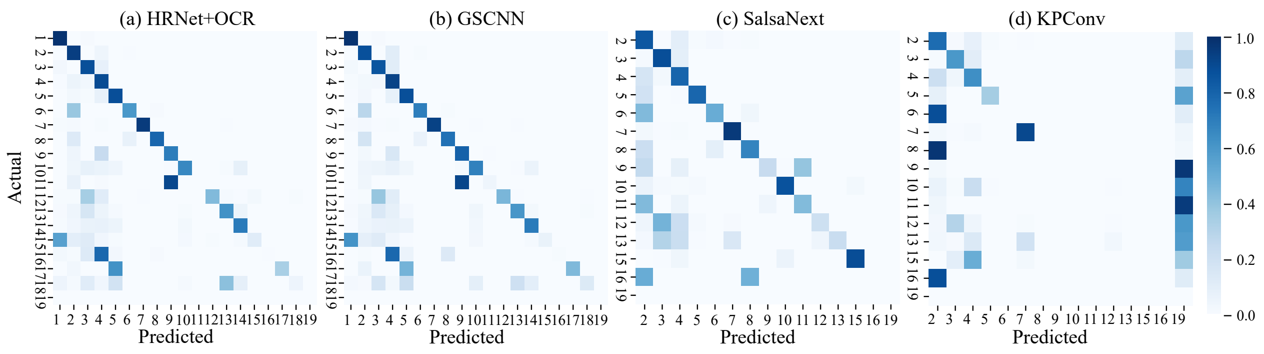


Figure 2.5: Confusion matrix. The y and x axis numbers represent classes ids (1: sky; 2: grass; 3: tree; 4: bush; 5: concrete; 6: mud; 7: person; 8: puddle; 9: rubble; 10: barrier; 11: log; 12: fence; 13: vehicle; 14: object; 15: pole; 16: water; 17: asphalt;18: building; 19: void.) Note that the sky label is omitted for the point cloud algorithm confusion matrices.

instead of deconvolution or upsampling layers.

Kernel Point Convolution (KPConv) is a specified, designed point cloud convolution operation that is more flexible than fixed grid convolution. KPConv uses a series of local, 3D convolution kernels to apply to the input points close to them, using a k-d tree to find nearby points. The regular subsampling strategy in the paper makes the KPconv operation more efficient and robust to varying densities. In the experiment, we use the KP-FCNN architecture [45] for semantic segmentation.

For the point cloud experiment, we follow the same data splits as for the image data. The training set has 7,800 scans, the validation set has 2,413 scans, and the testing set has 3,343 scans. Because the LIDAR scans are unable to establish points for classes such as sky, or objects far away (e.g., buildings in RELLIS-3D), the point cloud experiments use only the 15 classes with annotations (including void) for training and testing.

Table 2.1 and Fig. 2.5 show the results of the point cloud semantic segmentation. SalsaNext achieved 43.07% mIoU and KPConv achieved 19.07% mIoU, which is far under their performance on SemanticKITTI dataset, which was 59.5% mIoU and 58.8%, respectively. The imbalance phenomena in the point cloud dataset challenges these algorithms as well. Compared with SalsaNext, the degradation of KPConv is more obvious. We believe that the extremely imbalanced and unstructured features of our dataset mainly cause the degradation. While training, the KPConv model

Table 2.1: RELLIS-3D Semantic Segmentation Benchmark: Single image (20 classes) / scan (16 classes) for all baselines on test set.

models	sky	grass	tree	bush	concrete	mud	person	puddle	rubble	barrier	log	fence	vehicle	object	pole	water	asphalt	building	mean
hrnet+OCR	96.94	90.20	80.53	76.76	84.22	43.29	89.48	73.94	62.03	54.86	0.00	39.52	41.54	46.44	9.51	0.72	33.25	4.60	51.55
gscnn	97.02	84.95	78.52	70.33	83.82	45.52	90.31	71.49	66.03	55.12	2.92	41.86	46.51	54.64	6.90	0.94	44.18	11.47	52.92
salsanext	-	64.74	79.04	72.90	75.27	9.58	83.17	23.20	5.01	75.89	18.76	16.13	23.12	-	56.26	0.00	-	-	43.07
kpconv	-	56.41	49.25	58.45	33.91	0.00	81.20	0.00	0.00	0.00	0.00	0.40	0.00	-	0.00	0.00	-	-	19.97

does not learn on the whole LIDAR scan but instead on sampled neighborhoods of selected points. During training, there are two sampling strategies: random sampling and sampling based on the label distribution. The second strategy tried to mitigate the imbalanced distribution problem, but this attempt only increases results by **0.6%** mIoU.

2.3 Conclusion

In this chapter, we present RELLIS-3D, a large-scale multimodal dataset with annotations for off-road semantic segmentation. The unique challenges posed by off-road environments, such as imbalanced class distributions and unstructured features, are addressed by the dataset.

We found that current state-of-the-art deep learning models perform poorly on RELLIS-3D, highlighting the need for further research in off-road semantic segmentation. To address this, we plan to extend RELLIS-3D in the future by incorporating higher-order semantic labels and increasing environmental diversity.

Additionally, the high-accuracy GPS data and stereo images in our dataset provide opportunities to investigate odometry benchmarks and the use of semantic information to improve odometry estimation results. Overall, RELLIS-3D serves as a valuable resource for researchers and practitioners working in the field of off-road semantic segmentation.

3. MULTI-MODAL SEMANTIC SEGMENTATION

Semantic segmentation is a critical task in computer vision and autonomous driving, allowing machines to recognize and classify objects within a scene. It involves dividing an image or point cloud into multiple semantic regions and assigning each region a label that corresponds to a specific object or background class. This information is crucial for various applications, including object detection, path planning, and obstacle avoidance. In recent years, semantic segmentation has garnered significant attention in the field of autonomous driving, as it is a fundamental component of many perception-based tasks. The emergence of multimodal sensors, such as cameras, LIDARs, and radars, has sparked a growing interest in exploring their complementary strengths for semantic segmentation. In this chapter, we will explore various semantic segmentation challenges for different types of sensor data, including cameras, LIDARs, and radars.

3.1 Image Semantic Segmentation[*]

3.1.1 Method

To overcome these issues, we propose a framework for off-road semantic segmentation (OFFSEG) consists of two stages: Group segmentation for four classes and color segmentation of traversable region. The input is a raw RGB image and the output is a pixel-wise annotated RGB image. An overview of OFFSEG architecture is represented in Figure 3.1.

3.1.1.1 Group segmentation

Most CNN-based architectures consist of an encoder-decoder structure which downsamples the input to extract features and then upsamples with a pooling layer. This results in loss of spatial details. Architectures like HRNET[102] adopts a high resolution multiple branches to recover the spatial information. As shown in Figure 3.1 .a, the first stage of our framework is to perform group segmentation on 4 classes using CNN-based segmentation method. Specifically, for the two dataset RELLIS-3D(24 classes) and RUGD (20 classes) datasets in our experiment, we re-

[*]The major content of this section was published in [2]

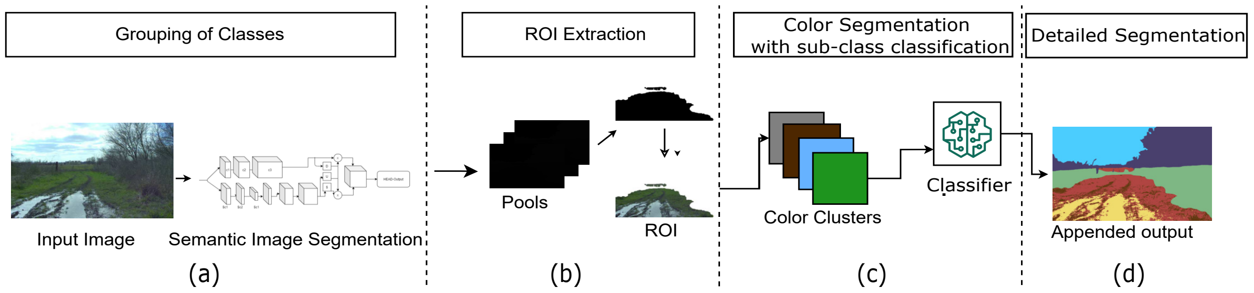


Figure 3.1: OFFSEG consists of two stages. (a) Pooling of different classes into four and performing semantic image segmentation (b) The RoI (region of interest) is obtained from the segmentation (c) Color segmentation algorithm is used to segment and classify sub classes like grass, mud, puddle, etc. (d) The append output.

categorized classes into four classes, 1) sky, 2) traversable, 3) non-traversable 4) obstacle.

From RELLIS-3D dataset 6 classes were pooled to traversable, 3 were pooled to non-traversable and 10 were pooled to obstacles. As the RELLIS-3D dataset had 94% of the pixels distributed between sky, grass, tree and bushes, pooling them into four different classes solved the problem of class imbalance issue as shown in Figure 3.2. The pixel-wise annotation of the classes from RELLIS-3D and RUGD were then converted into these four classes for training on the semantic segmentation network.

The traversable class includes sub-classes like puddle, mud, dirt, gravel, etc. as given in Table 3.1. These sub-classes play an important role in determining path during robotic navigation on the off-road environments. Another reason to consider only the traversable class as our region of interest (RoI) is to ignore all other unusable sub-classes present in the environment which are not necessary for determining traversable paths in autonomous driving like pole, bush, etc. These instances do not require fine segmentation to achieve.

3.1.1.2 Color segmentation and sub-class classification

K-Means algorithm has been used to extract the color pools from the output obtained in the previous group segmentation stage. Color pools are used to distinguish between several components

Class Distribution for RELLIS-3D and RUGD			
Sky	**Traversable**	**Non Traversable**	**Obstacles**
Sky[RE,RU]	Grass[RE,RU]	Bush[RE,RU]	Vehicle[RE,RU]
	Dirt[RE,RU]	Void[RE]	Barrier[RE]
	Asphalt[RE,RU]	Water[RE,RU]	Log[RE,RU]
	Concrete[RE,RU]	Deep Water[RE]	Pole[RE]
	Puddle[RE]		Object[RE]
	mud[RE]		Building[RE,RU]
	Sand[RU]		Person[RE,RU]
	Gravel[RU]		Fence[RE,RU]
	Mulch[RU]		Tree[RE,RU]
	Bridge[RU]		Rubble[RE]
	Rockbed[RU]		Pole[RU]
			Container[RU]
			Bicycle[RU]
			Sign[RU]
			Rock[RU]
			Table[RU]

Table 3.1: Polled classes from RELLIS-3D[RE] and RUGD[RU]

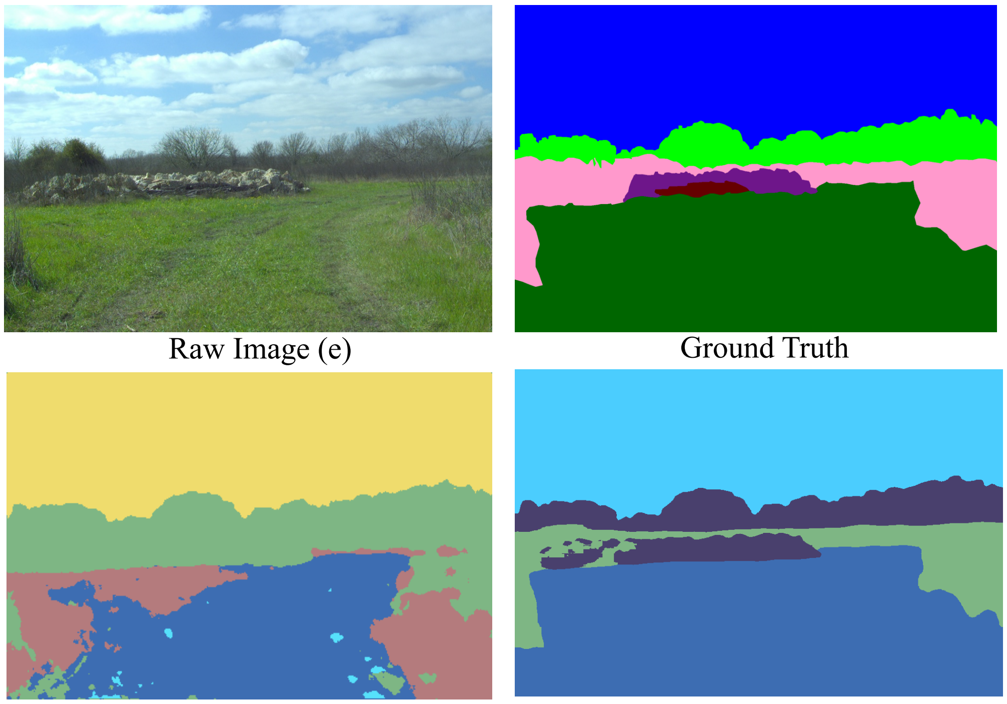


Figure 3.2: Segmentation results from OFFSEG for four class model have been compared with the HRNET 20 class model

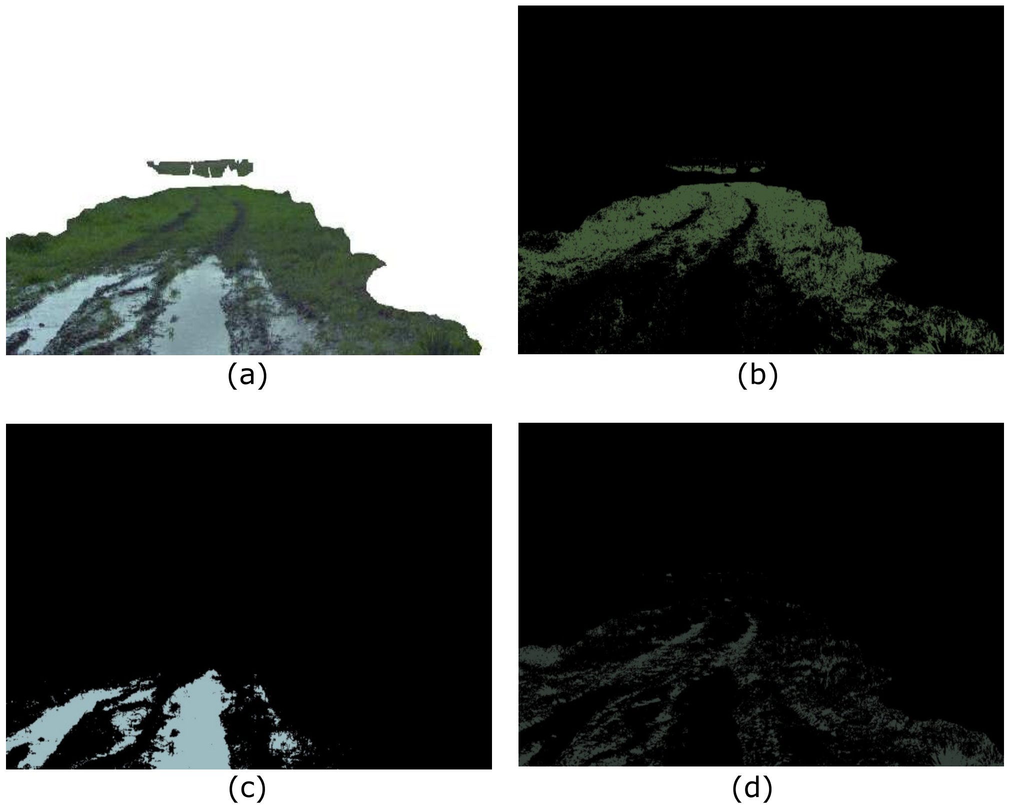


Figure 3.3: Color segmentation results on RELLIS-3D (a) Traversable class obtained as RoI from segmentation (b) grass, (c) puddle, (d) mud obtained from color segmentation from RoI.

present in an off-road environment. Each cluster obtained from the centroid has been transferred into the classification model which gives us the mapping of the required sub-classes in our region of interest as shown in Figure 3.1.b. The color segmentation algorithm extracts the color masks from the image and inputs these masks into the classification model. The classifier classifies the sub-classes in terms of color clusters and determines the sub-classes like mud, puddle, grass, water, etc. as shown in Figure 3.1.c. Next, these obtained masks are appended on the segmentation output which was obtained from semantic segmentation resulting in final segmentation as shown in Figure 3.5.

3.1.1.3 Data pre-processing and data generation for classification

In order to classify the sub-classes of the traversable region, we need to create an image-oriented dataset for the training of a classification model. This dataset comprises of the detailed sub-classes present in the traversable region. The training samples in RELLIS-3D has 6 classes in the traversable region – grass, mud, puddle, dirt, asphalt and concrete, while RUGD has 8 classes in the traversable region – dirt, sand, grass, water, asphalt, gravel, mulch and concrete.

3.1.1.4 Training of classification model

The output from color segmentation needs to be further classified into different sub-classes in the traversable region. The classifier differentiates the masks extracted from K-Means clustering and assigns the respective classes to them. Table 3.1 shows the different sub-classes present in the traversable region of both RELLIS-3D and RUGD datasets.

3.1.2 Experiment Results

The OFFSEG approach was evaluated on two state-of-the-art off-road datasets RELLIS-3D and RUGD using BiSeNetV2 and HRNETV2+OCR architectures. We have also tested OFFSEG on IISERB campus data.

3.1.2.1 Group Segmentation

The evaluation of image semantic segmentation of the converted classes of RELLIS-3D and RUGD were done using two state-of-the-art architectures: BiSeNetV2[105] and HRNETV2+OCR[106, 102]. BiSeNetV2 consists of two branches: detail branch and semantic branch. The detail branch extracts spatial details consisting of low-level information and uses shallow layers with wide channels. Meanwhile, semantic branch extracts high-level semantics employing low channel capacity. Then an aggregation layer merges extracted features from the two branches and upsample the output from aggregation layer.

HRNETV2+OCR consists of a High-Resolution Network which acts as a backbone and Object-Contextual Representations (OCR) to enhanced pixel representation of objects. Unlike other segmentation models HRNET maintains high resolution throughout the model avoiding the down-sample and upsample process. OCR aggregated the features extracted from HRNET to improve pixel representation. We used 3,302 images for training set, 983 images for validation set and the testing with 1672 images for the RELLIS-3D. For RUGD, we used 4732 images for training set,

Models \ Classes	Sky	Traversable	Non-Traversable	Obstacles	**mIoU**
BiSeNet-V2 [RELLIS-3D]	97.09%	92.30%	77.12%	79.93%	86.61%
HRNETV2 [RELLIS-3D]	96.85%	86.04%	66.22%	74.18%	80.82%
BiSeNet-V2 [RUGD]	90.85%	91.83%	47.81%	90.20%	80.17%
HRNETV2 [RUGD]	92.27%	94.18%	59.92%	91.60%	84.49%

Table 3.2: OffSeg mIoU of experiment results

932 images for the validation set and 1827 images for the testing set.

3.1.2.2 Quantitative analysis of the architectures used for segmentation

From Table3.2, the mean IoU obtained for RELLIS-3D on BiSeNetV2 and HRNETV2+OCR were 86.61% and 80.82% respectively. The mean IoU obtained for RUGD on BiSeNetV2 and HRNETV2+OCR were 80.17% and 84.49% respectively. The results obtained in Figure 3.2 shows the prediction of BiSeNetv2 on a RELLIS-3D frame which contains the class log. The obstacle class in the prediction covers most of the log ground truth labels inferring higher predictions than the prediction of HRNETV2+OCR trained on 20 classes for the log which had 0.0% IoU.

3.1.2.3 Subclass segmentation

We obtain color clusters using K-Means algorithm. The set of random k-points has been assigned with the closest centroid from the image which further combines these centroids into separate clusters. By adopting an iterative approach, we obtain a set of all possible color clusters present in the RGB layers. The number of required clusters depends on the incorporation of subclass properties in a frame with respect to the sub-classes present in the traversable region of the dataset.

3.1.2.4 Color segmentation and sub-class classification

The color masks obtained after applying color segmentation on the RoI is shown in Figure 3.3 and Figure 3.4. The accuracy of OFFSEG is determined by how precisely the sub-classes are

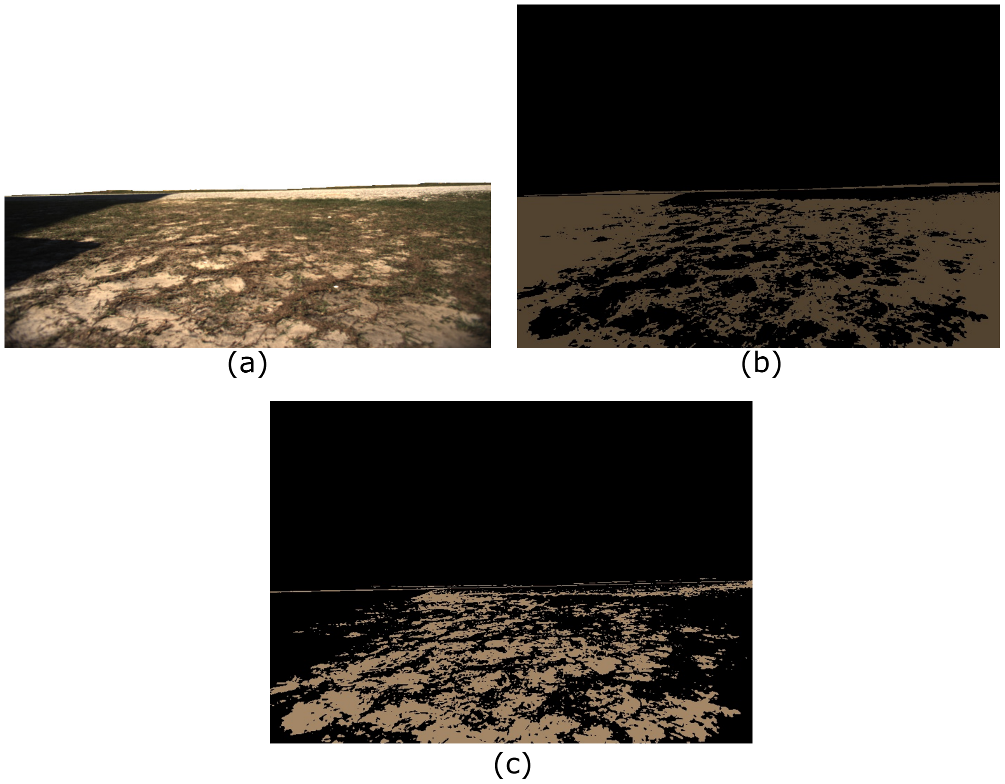


Figure 3.4: Color segmentation results on RUGD (a) Traversable class obtained as RoI from segmentation (b) mulch, (c) gravel obtained from color segmentation from RoI.

classified.

We trained a classification model using transfer learning with MoblieNetV2[107] as the classifier model pre-trained on ImageNet dataset[108]. The training inputs are classes listed in the traversable region of Table 3.1 from both RELLIS-3D and RUGD dataset.The classifier was trained on 23,967 images for 9 classes which achieved a mean accuracy of 97.3% and the outputs obtained from the model are used into the color segmentation algorithm which appends only classified subclasses into the final result. Note that, quantitative analysis for the color segmentation would lead to inaccurate results as the ground truth for classes in the traversable region is very vague whereas the outputs in our approach are more feature-rich with distinct boundaries. The detailed outputs obtained using OFFSEG expands the application space of the model.

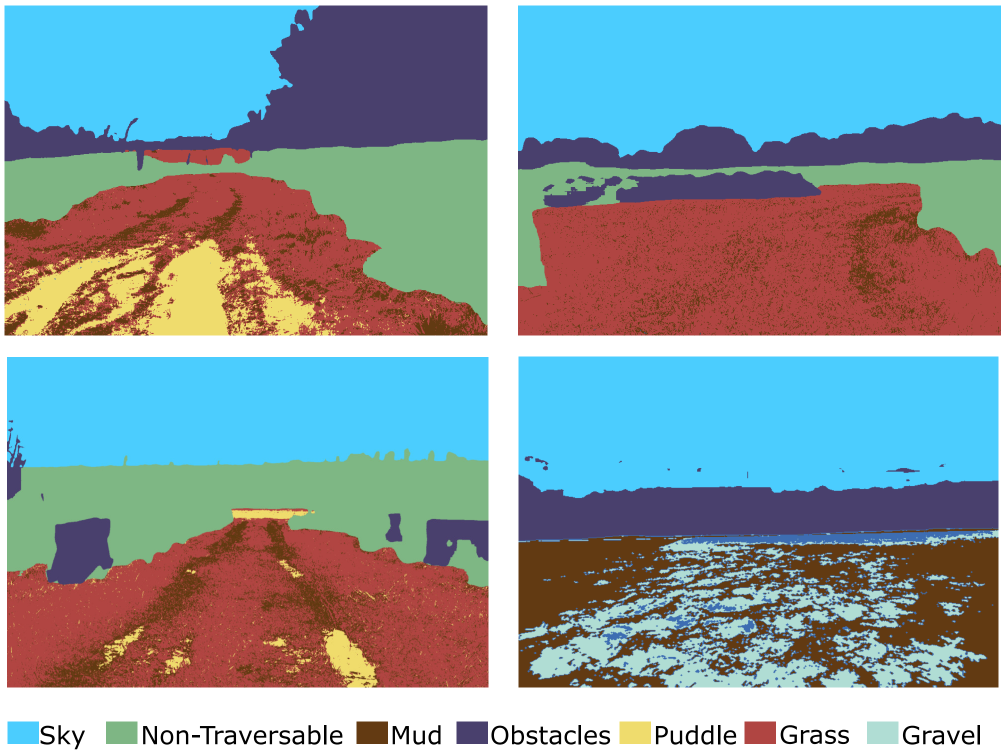


Figure 3.5: Final segmentation of classes (sky, traversable, non-traversable, obstacles) and subclasses (mud, puddle, grass, gravel) in a,e,b,d frames respectively.

3.2 LIDAR Semantic Segmenation*

3.2.1 Method

3.2.1.1 Input Representation

This paper focuses on the domain adaptation for full-scene semantic segmentation from one real-world LIDAR scan dataset to another real-world LIDAR scan dataset. Our model used a projection-based model as the segmentor backbone. The input of a projection-based model is the projected LIDAR scan refer to Eq.3.1, where r is the range, (x, y, z) are the coordinates, (w, h) are width and height of the image, f is the angle of the field of view of LIDAR, and f_{up} is the up angle of the field of view. After the projection, we can get a range image and a point index image. Furthermore we also get a 3D coordinate map of the point cloud. Many projection-based models

*The content of this section was published in [3]

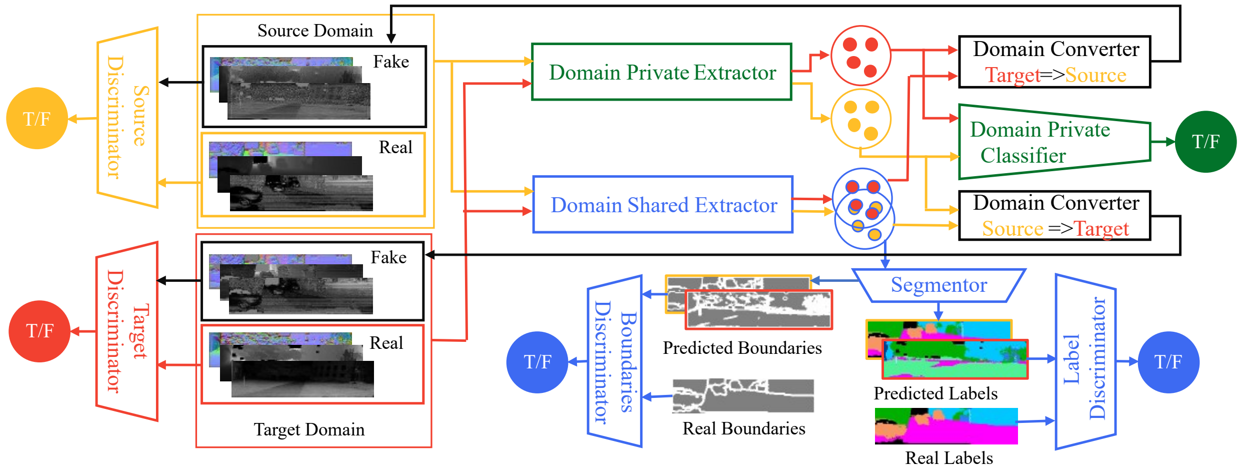


Figure 3.6: Information Flow Graph: The data (source and target) are fed into two branches: one branch is composed of a domain private extractor f_P and a domain private classifier f_D which can differentiate the input from the two domains, another branch is a domain shared extractor f_C which extracts the common feature between the two domains including semantic information Y and boundries information B. A segmentor f_{Seg} predicts labels $\hat{Y}$ and boundries information $\hat{B}$ based on the features from domain shared extractor. The predicted boundaries are sent to a boundaries discriminator D_B, and the predicted labels are sent to a labels discriminator D_Y. Next, the domain private features and domain shared features are fed into domain converters ($f_{S\to T}$ converts source data into target domain, $f_{T\to S}$ converts target data into source domain). The coversion are learning through an adversarial learning procedure. Therefore, the coverted data are seperately fed into domain discriminators (D_T and D_S). Along with this, the converted data are also fed back to the model to repeat the above procedures.

[51, 47, 48] use the range image, reflectivity map, and coordinate map as input. But the point cloud from the different platforms has different coordinate systems, which is not good for domain adaptation. Meanwhile, the normal map have fixed range for all LiDARs. Besides, according to [109], the normal map can help the model perform semantic segmentation for depth image. Therefore, we use a normal map accompanying with the range image, reflectivity map, as input.

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \frac{1}{2}[1 - arctan(y,x)\pi^{-1}]w \\ [1 - (arcsin(zr^{-1}) + f_{up})f^{-1}]h \end{pmatrix} \tag{3.1}$$

In our experiment, we notice that a lot of stripe pattern appears on projected images and labels (see Fig.3.7(a)), which affects the domain adaptation process. To reduce the negative effect, we

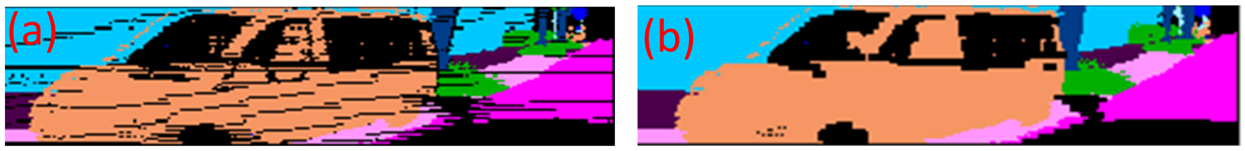


Figure 3.7: (a) the original label; b) the inpainted label;

pre-process the data (see Fig.3.7(b)). We utilize Closing morphological and subtract operations to locate stripe pattern on mask image. Then we inpaint the reflectivity and range image using the Navier-Stokes inpainting algorithm [110], and fill the label using the holes' nearest neighbors. And the normal map is computed from the inpainted range image.

3.2.1.2 Network Structure

Generally, we expect a model that can complete a task for data from similar domains. However, feature difference between two similar domains causes a model, which learns from one domain (called source domain S), can not perform well on another domain (called target domain T). Therefore, we expect a method that can adapt a model from one domain to another domain. If the target domain does not provide ground truth, the problem is called unsupervised domain adaptation. In this paper, the task is full-scene semantic segmentation for LIDAR scan. In this problem, we use X_S denotes source data, Y_S denotes source labels, and X_T denotes target data, but target labels are not accessible.

Based on the intuition that two similar domains should contain shared information across the two domains and private information to each domain. And the adaptable information should be contained in shared information of two domains. Therefore, we designed an end-to end trainable model that splits input data into domain shared and private features. The model then utilizes the extracted shared features to perform semantic segmentation. The model contains two extractors: a shared feature extractor f_P and a private feature extractor f_D see Fig.3.6. To induce the two extractors to produce such split information, we add a loss function that encourages the independence of these parts, and connect the private feature extractor to a classifier f_C to differentiate the data from two domains. Besides, we feed the output of the domain shared extractor to a segmentor to complete the same task (predict semantic labels $\hat{Y}$).

To ensure that the private features are still useful and to further reduce the domain gap, we introduce the CycleGAN mechanism [111] to induce the models to learn two mappings between two domains. The domain private and shared features are fed into domain converters to convert the data from one domain to another domain: ($f_{S \to T}$ converts source data into target domain, $f_{T \to S}$ converts target data into source domain). The conversion is learning through an adversarial learning procedure. Therefore, the converted data are separately fed into domain discriminators (D_T and D_S). Meanwhile, we add Gated-SCNN [104] on the side of the segmentor to extract boundary maps B while learning to predict semantic segmentation. The boundary maps are the predicted boundary of semantic labels. We utilize the output boundaries to penalize the label predictions from the target domain. To further penalize the label output, we add a boundaries discriminator D_B, and a labels discriminator D_Y to penalize the output label and boundary.

3.2.1.3 Multi-task Learning

The domain adaptation procedure is essentially a multi-task learning procedure. The tasks include domain private feature classification, boundaries extraction, semantic segmentation, domain mutual conversion, similarity measurement of domain shared features and divergence measurement of the domain private features. The complete loss function can be written as follows:

$$L = \lambda_P L_P + \lambda_B L_B + \lambda_{Seg} L_{Seg} + \lambda_M L_M + \lambda_C LC + \lambda_D L_D \tag{3.2}$$

where L_P, L_B, L_{Seg}, L_M, L_C and L_D correspond to the loss of domain private feature classification, boundaries extraction, semantic segmentation, domain mutual conversion, domain similarity and domain difference and λ_P, λ_B, λ_{Seg}, λ_M λ_C and $\lambda_D$are hyperparameters that control the weighting between losses.

The domain private feature classification task is a binary classification problem, which uses standard binary cross-entropy loss Eq. (3.3).

$$L_{BCE}(Y,\hat{Y}) = E_{y\sim Y}[y\log(\hat{y}) + (1-y)log(1-\hat{y})] \tag{3.3}$$

Therefore, the classification loss is 3.4, where $\delta(X) = \{1 : x \in X_S; 0 : x \in X_T\}$.

$$L_P = L_{BCE}(\delta(X), f_D(f_P(X))) \tag{3.4}$$

For the boundaries extraction task, we have access to labels of source data, which allows us to get the boundaries of source data B_S. We then use standard binary cross-entropy (BCE) loss on predicted boundary maps $\hat{B}_s$ of source data. From experiments, the network inclines to generate blank results if there was no penalty on target data. Therefore, we add a GAN loss to encourage the network to predict boundaries for target data too. We express GAN loss as Eq.(3.5)

$$L_{GAN}(G, D_Y, X, Y) = E_{y\sim Y}[\log D_Y(y)] + E_{x\sim X}[\log(1 - D_Y(G(x)))] \tag{3.5}$$

Then, the complete loss function of boundary extraction task can be written as Eq.(3.6), where $G_B(x)$ equals $f_C(x)$,and $\lambda_{B_{GAN}}$, $\lambda_{B_{BCE}}$ are hyper-parameters for balancing the effect between the GAN loss and BCE loss.

$$L_{bd} = \lambda_{B_{BCE}} L_{BCE}(B_S, \hat{B}_S) + \lambda_{B_{GAN}} L_{GAN}(G_B, D_B, X_T, B_T) \tag{3.6}$$

For the semantic segmentation task, L_{Seg} consist of two parts: L^S_{Seg} of source data and L^T_{Seg} of target data. We employ standard cross-entropy (CE) loss with dual boundary regularizer [104] and Lovász-Softmax loss on predicted labels of source data.

$$L^S_{Seg} = \lambda_{SS1} L_{CE}(Y_S, \hat{Y}_S) + \lambda_{SS2} L_{dual}(Y_S, \hat{Y}_S, \hat{B}_S) + \lambda_{SS3} L_{Loasz}(Y_S, \hat{Y}_S, \hat{B}_S) \tag{3.7}$$

Where $G_{Seg}(X) = f_{seg}(f_C(X)))$.

For target data, we employ GAN loss to learn segmentation [112]. Besides, because learning the boundary map is easier than learning semantic segmentation. Therefore, we used the boundary prediction $\hat{B}_T$ to penalize the label predictions $\hat{Y}_T$ of the target data. We add a Laplacian layer to extract the boundary of the predicted labels and use the L1 loss to measure the difference between the boundray prediction and the predicted label boundary. In the end, the segmentation loss of source data is Eq.(3.8)

$$\begin{aligned} L_{Seg}^{T} =& \lambda_{ST1} L_{GAN}(G_{Seg}, D_{Seg}, X_T, Y_S) \\ &+ \lambda_{ST2} E_{x_t \sim X_T}^{b_t \sim \hat{B}_T} [\| \ Laplacian(G_{Seg}(x_t)) - \hat{b}_t \ \|_1] \end{aligned} \tag{3.8}$$

In order to further eliminate the effect of domain difference, we introduce the CycleGAN mechanism into our model, which leads to the fourth task: domain mutual conversion task. We expect that through learning the domain mutual conversion, the model can find the interior relationship between two domains. The mutual conversion task requires two mapping functions: $G_{S\to T}$ maps data from the source domain to target domain, $G_{T\to S}$ maps data from target domain to source domain. The two mappings function can be expressed as 3.9.

$$\begin{aligned} G_{T\to S}(X) &= f_{T\to S}(f_P(f_H(x_t)), f_C(X)) \\ G_{S\to T}(X) &= f_{S\to T}(f_P(f_H(x_s)), f_C(X)) \end{aligned} \tag{3.9}$$

Based on the two mapping functions, we can define the domain mutual conversion loss as follows:

$$L = \lambda_{M_{inv}} L_{inv} + \lambda_{M_{cyc}} L_{cyc} \tag{3.10}$$

Where L_{inv} and L_{cyc} represent domain invariance loss and cycle consistency loss.

The domain invariance means that the data domain will not be changed if it passes through its domain convertor. For example, we will get data in source domain $X_{S(S)}$ after data from source domain X_S pass the mapping function $G_{T\to S}$. This invariance character of the mapping function can be learned through the following function:

$$\begin{aligned}L_{inv}(G_{S\to T}, G_{T\to S}, X_S, X_T) = &E_{x_s\sim X_S}[\parallel \hat{x}_{s(s)} - x_s \parallel_1] \\ &+ E_{x_t\sim X_T}[\parallel \hat{x}_{t(t)} - x_t \parallel_1] \\ &+ L_{GAN}(G_{S\to T}, D_T, \hat{X}_{S(S)}, X_T) \\ &+ L_{GAN}(G_{T\to S}, D_S, \hat{X}_{T(T)}, X_T)\end{aligned} \tag{3.11}$$

On the other end, cycle consistency means that after data passes two different mapping functions, its domain should be in its original domain. For example, source domain data X_S first passes the mapping function $G_{S\to T}$. The converted results $X_{T(S)}$ passes the mapping function $G_{T\to S}$. We will finally get data $X_{S(T(S))}$, which should be in the source domain. The cycle consistency loss can be defined as:

$$\begin{aligned}L_{cyc}(G_{S\to T}, G_{T\to S}, X_S, X_T) =&E_{x_s\sim X_S}[\parallel \hat{x}_{s(t(s))} - x_s \parallel_1] \\ &+ E_{x_t\sim X_T}[\parallel \hat{x}_{t(s(t))} - x_t \parallel_1] \\ &+ E_{x_s\sim X_S}[\parallel G_{Seg}(\hat{x}_{s(t(s))}) - G_{Seg}(x_s) \parallel_1] \\ &+ E_{x_t\sim X_T}[\parallel G_{Seg}(\hat{x}_{t(s(t))}) - G_{Seg}(x_t) \parallel_1]\end{aligned} \tag{3.12}$$

We meausure the similarity of the shared features between the original data and converted data using $L1$ loss:

$$\begin{aligned}L_C(f_C, G_{S\to T}, G_{T\to S}, X_S, X_T) =&E_{x_s\sim X_S}[\parallel f_C(\hat{x}_{t(s)}) - f_C(x_s) \parallel_1] \\ &+ E_{x_t\sim X_T}[\parallel f_C(\hat{x}_{s(t)}) - f_C(x_t) \parallel_1]\end{aligned} \tag{3.13}$$

To measure the divergence of the shared features and private features, we define the loss via as soft subspace orthogonality constraint between the private and shared features [113]:

$$L_D = \parallel \mathbf{H_c^s}^\top \mathbf{H_p^s} \parallel_F + \parallel \mathbf{H_c^t}^\top \mathbf{H_p^t} \parallel_F \tag{3.14}$$

Where $\| \bullet \|_F$ is the squared Frobenius norm. $\mathbf{H}_\mathbf{p}^\mathbf{s}$ and are $\mathbf{H}_\mathbf{p}^\mathbf{t}$ are the matrices whose row are the private features from the source domain and target domain respectively. And $\mathbf{H}_\mathbf{c}^\mathbf{s}$ and are $\mathbf{H}_\mathbf{c}^\mathbf{t}$ are the matrices whose row shared features from the two different domain.

3.2.2 Experiment

We evaluated our algorithm on three datasets: SemanticKITTI dataset [21], SemanticPOSS [35], and our dataset(SemanticUSL). The information on the other two datasets is as followed:

- SemanticKITTI is labeled from the KITTI dataset collected around Karlsruhe's mid-size city, rural areas, and highways. The data was collected using a Volkswagen Passat B6 with a Velodyne HDL-64E. The Semantic KITTI dataset has 23201 labeled scans and 28 classes.

- SemanticPOSS was collected at the Peking University campus and contained many dynamic and complex scenes, which are different from SemanticKITTI. The platform is a JEEP with a Pandora 48 channel LIDAR. The dataset has 2988 frames and 14 classes.

In summary, the three datasets were collected on different platforms and different sites. These differences cause the divergence of point cloud distributions, noise, and reflectivity, etc. In this section, we study our approach's domain adaptation ability among the SemanticKITTI, SemanticPOSS, and SemanticUSL datasets by treating one as the source domain and another as the target domain. We trained original SalsaNext on source data to provide a reference. Besides, we also compared our method with the pixel-level adapted CyCADA method [114], because both approaches are using the CycleGAN mechanism. And we also are interested in how well the CyCADA can adapt projected 3D data. We train a SalsaNext model on the adapted data by CyCADA. We reported the results in Table 3.3.

The pixel-level CyCADA methods do not work well on the projected 3D points cloud. See Table.3.3(third column with "cyckitti/usl/poss"), most of the model can keep the same performance trained with the adapted data but can't generalize to the target domain. In the SemanticPOSS $\rightarrow$ SemanticUSL case, the model can't keep the same performance if trained with the adapted data. This adaptation method tries to visually bring close two domains in global features like reflectivity and norm distribution. However, there's no guidance for local features adaptation. Another reason could be SemanticPOSS doesn't have enough data to complete CycleGAN conversion. The performance degrades a lot on the transformation with SemanticPOSS.

On the other hand, see Table.3.3(third column with "ours" and "ourskitti/usl/poss"), the results shows that our method has better adaptation results. Firstly, after adaptation, the performance of our model on the source domain decreased slightly. Meanwhile, the model recovered over 60% performance on the target domain. For example, in the case of adapting POSS to USL, the model got 52.18%mIoU on the SemanticPOSS dataset and got 31.72% on SemanticUSL. The results indicate the amount of label data in the source domain affects the results. In KITTI $\rightarrow$ USL case, the adapted model's performance on the SemanticUSL dataset is even higher than the model trained with USL data only. We also provide ablation studies about the effect of the CycleGAN mechanism and the boundary penalty on the adaptation results. Without the CycleGAN mechanism, our model got 36.55% mIoU by adapting from SemanticKITTI to SemanticUSL. Without the CycleGAN, the IoU of Car decreased to 28.76% because car detection relies on reflectivity [49]. The CycleGAN mechanism can adapt the reflectivity features between two domains. By disabling the model's boundary-aware part, the model got 37.20% mIoU by performing domain adaptation from SemanticKITTI to SemanticUSL. Compared with the model's output after disabling the boundary-aware function, the label with boundary-aware parts has better shapes.

3.3 RADAR Semantic Segmentation

3.3.1 Method

The growing demand for autonomous vehicle technology in adverse weather conditions, such as rain and snow, calls for effective solutions. While CAMERA and LIDAR sensors have been successful in normal conditions, inclement weather can negatively impact their performance. RADAR sensors, such as those produced by Navtech [115], offer a promising alternative. RADAR sensors have a longer wavelength, making them less affected by small particles like fog, rain, or snow that can impair the performance of CAMERA and LIDAR sensors. Additionally, RADAR has a more

Table 3.3: Domain Adaptation Experiment results on SemanticKITTI, SemanticPOSS and SemanticUSL

Source	Target	Method	person	rider	car	trunk	vegetation	traffic-sign	pole	object	building	fence	bike	ground	mIoU
KITTI	KITTI	kitti	62.09	74.21	93.59	61.15	91.11	37.99	57.94	50.36	84.82	54.64	15.48	94.13	64.79
		cycposs	64.22	76.44	92.36	60.64	90.57	37.75	57.09	46.80	84.08	51.35	15.35	93.80	64.20
		cycusl	58.42	69.05	92.31	56.33	90.53	37.23	56.09	44.70	82.04	47.51	13.63	93.66	61.79
		oursposs	47.46	68.52	94.06	73.90	47.62	37.33	59.15	58.24	88.45	27.75	29.41	56.11	57.33
		oursusl	46.04	68.86	94.95	69.91	81.49	38.60	63.65	50.05	88.07	22.20	37.74	91.19	62.73
	POSS	source	22.77	1.78	35.91	16.86	39.84	7.08	9.73	0.18	57.03	1.64	18.17	41.99	21.08
		cycada	0.00	0.00	0.00	1.45	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.12
		ours	31.39	23.98	70.78	21.43	60.68	9.59	17.48	4.97	79.53	12.57	0.78	82.41	34.63
	USL	source	33.90	0.00	27.45	10.68	36.89	16.20	12.72	5.68	41.61	3.55	31.60	75.95	24.69
		cycada	0.38	0.00	28.70	13.83	57.11	20.70	23.83	3.78	53.14	22.30	9.24	72.36	25.45
		ours	33.17	0.00	67.75	38.95	85.60	49.93	43.44	8.94	72.86	44.06	23.07	93.18	46.75
POSS	POSS	source	64.47	48.25	85.77	29.71	62.71	27.29	38.19	8.07	84.90	48.50	65.56	72.56	53.00
		cyckitti	64.80	48.05	84.52	29.16	61.81	26.42	33.99	8.44	84.15	47.45	65.52	71.87	52.18
		cycusl	32.82	17.73	71.88	16.76	53.41	16.89	16.96	1.03	71.43	24.20	49.45	67.25	36.65
		ourskitti	61.13	45.33	82.88	30.93	60.82	32.60	32.46	7.05	82.82	37.42	60.05	72.64	50.51
		oursusl	60.70	44.93	85.19	31.33	61.60	35.12	35.33	9.83	84.30	41.21	65.10	71.62	52.19
	KITTI	source	5.20	0.50	22.57	0.54	44.00	1.90	12.83	0.08	43.09	0.70	0.40	5.62	11.45
		cycada	0.28	0.68	4.67	0.32	23.75	0.75	5.01	0.49	12.29	0.83	0.06	6.94	4.67
		ours	23.64	24.86	73.31	23.67	72.38	4.17	31.28	2.48	59.41	0.36	0.53	68.68	32.06
	USL	source	2.45	0.00	16.15	1.21	27.94	1.34	4.52	0.62	44.37	0.12	1.16	8.05	8.99
		cycada	0.00	0.00	0.00	0.05	9.40	0.19	1.12	0.15	5.06	0.28	0.00	28.01	3.69
		ours	30.38	0.00	45.73	28.69	63.08	22.29	33.92	4.12	63.70	1.89	9.42	77.49	31.73
USL	USL	source	51.96	0.00	12.57	26.29	72.89	11.18	47.22	15.11	59.78	39.61	0.00	85.61	35.19
	KITTI	source	1.11	0.01	28.77	5.61	38.75	3.93	15.43	1.80	29.77	1.94	1.24	46.45	14.57
		cycada	0.17	0.00	10.43	5.06	31.86	0.53	10.26	0.96	36.31	5.42	0.14	47.47	12.38
		ours	14.91	0.00	66.72	28.28	67.61	12.95	30.67	1.00	57.15	18.82	3.94	75.60	31.47
	POSS	source	4.67	0.00	21.66	3.04	27.96	1.61	6.03	0.09	41.67	2.55	5.93	63.08	14.86
		cycposs	5.28	0.00	9.93	5.30	25.66	1.95	5.93	0.01	52.01	0.75	0.36	50.58	13.15
		ours	11.50	0.00	44.41	19.33	41.39	5.97	13.27	0.00	71.37	0.99	1.53	66.08	22.99

extended detection range and the ability to penetrate materials, which allows for detecting objects beyond the line of sight of LIDAR sensors. These features make RADAR ideal for use in adverse weather conditions. While RADAR sensors offer robust performance in inclement weather but RADAR has lower spatial resolution and higher noise than LIDAR sensors, making radar data more challenging to process.

Given the difficulties in assigning semantic labels to raw radar data due to its sparsity and loss of third-dimension information, we propose a pipeline to transfer the semantic labels from LIDAR to RADAR. The pipeline consists of the following steps:

- Calibration of the extrinsic transformation between LIDAR and RADAR;
- Collection of data, including LIDAR, RADAR, and IMU, through the use of robots in an off-road environment;

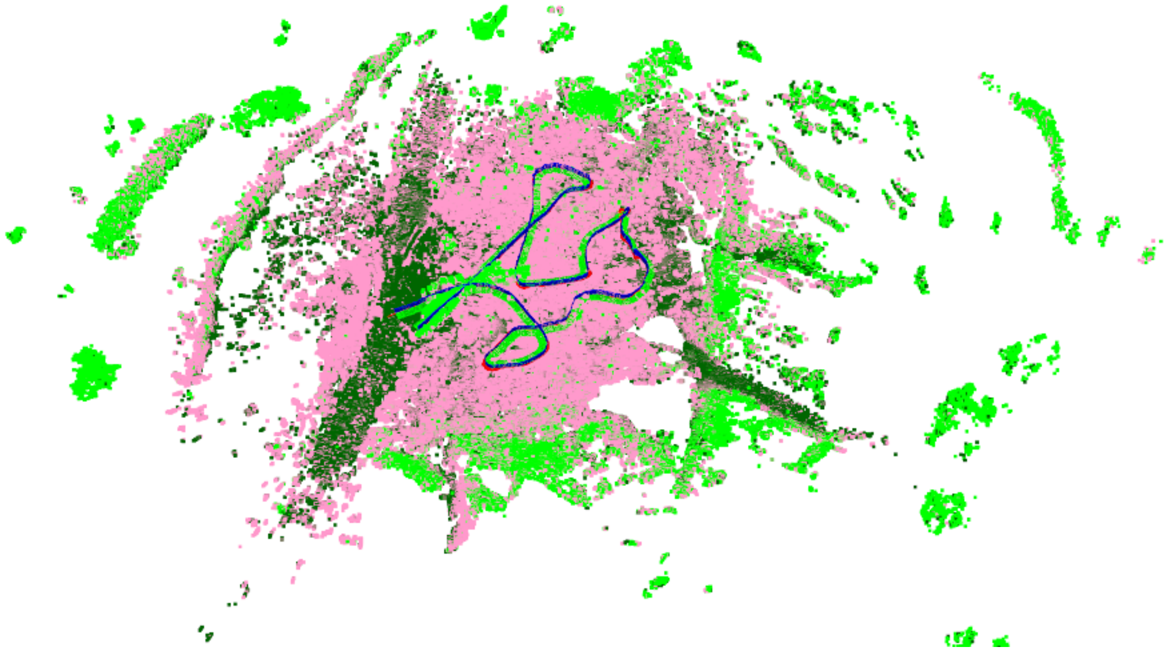


Figure 3.8: Accumulated Point Cloud with semantic labels

- Application of semantic models to LIDAR data;
- Accumulation of LIDAR scans using a LIDAR odometry algorithm;
- Fusion of LIDAR scan labels based on the LIDAR odometry;
- Semantic information from LIDAR to RADAR is projected using the extrinsic transformation between the two sensors.
- Train a model on the generated LIDAR labels

For the calibration of LIDAR and RADAR, we propose a method that involves using a small Multi-Layer Perceptron (MLP) to model the target as a collection of RADAR return energy. The model is trained to learn the relationship between the RADAR sensor pose relative to the target and the return energy, which enables it to be used for calibration. This process is treated as a regression problem and involves minimizing the regression loss, reprojection loss, and ray pass loss. Further details on this method will be discussed in Section 5.2.

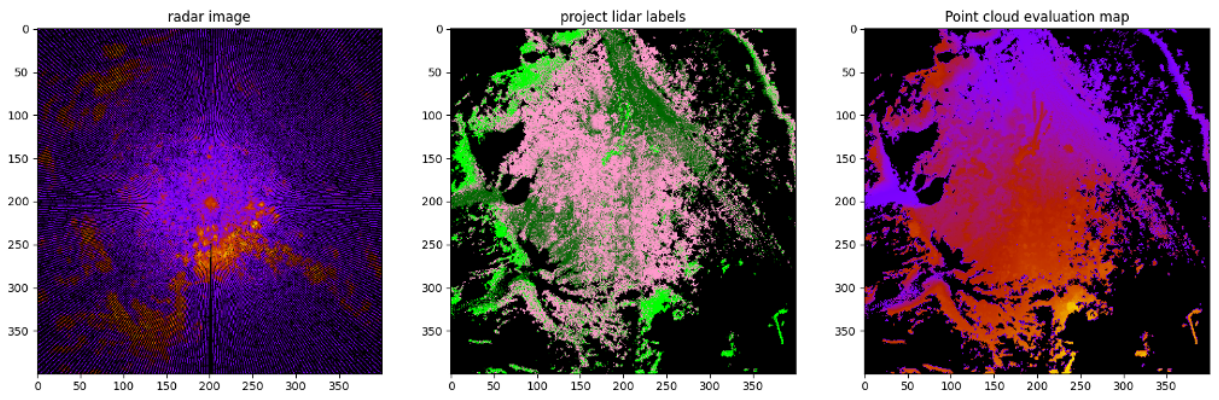


Figure 3.9: Radar and Point Map Projection on XY plane

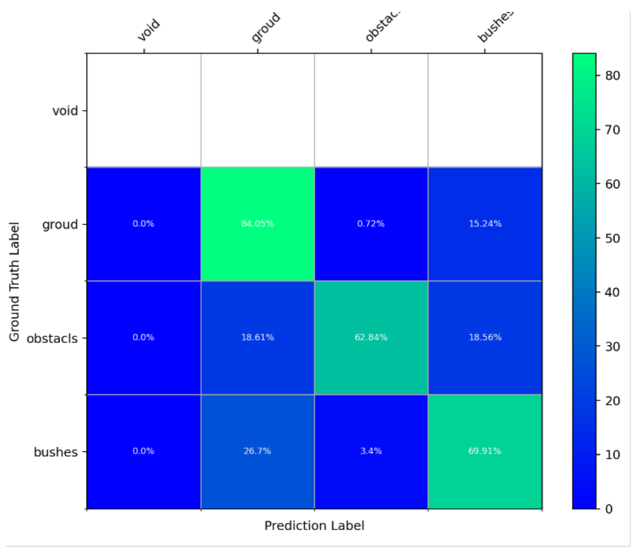
For semantic segmentation of LIDAR data, we use the Cylinder3D model[116]. We first train Cylinder3D on our labeled LIDAR data and then perform inference on the collected LIDAR data. We chose cylinder3D as it considers the 3D properties of LIDAR and is more resilient to changes in sensors.

For LIDAR odometry estimation, we use the method described in [117, 118] to accumulate LIDAR scans. The accumulated LIDAR scans result in a dense 3D point cloud. We then voxelized the 3D point cloud and assigned semantic labels to each voxel based on the frequency of labels of points within the voxel. The final point cloud map can be visualized in Fig.3.8

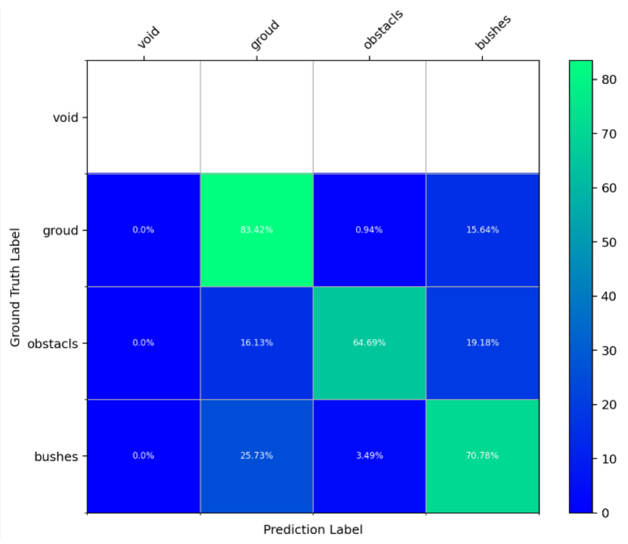
Finally, the LIDAR labels are projected onto the RADAR data using the extrinsic transformation between LIDAR and RADAR.

Once the semantic labels of the radar data have been obtained, a semantic segmentation model can be trained using the generated labels as the ground truth. However, it is important to note that radar data differ from LIDAR or image data in terms of sparsity and information loss. This makes it challenging to distinguish between many classes. To overcome this, the number of classes is reduced to four - void, bushes, obstacles, and ground. This simplification makes it easier for the model to learn and predict accurate results.

The representation of RADAR data used to train a model can impact its performance. In semantic segmentation, spatial information is crucial. To maintain the spatial relationship, the



(a) One Channel Input



(b) Three Channel Input

Figure 3.10: RADAR Semantic Confusion Confusion Matrix

RADAR data is projected onto the x-y plane of the RADAR coordinate, and the RADAR return energy is used as input.

3.3.2 Experiment

To evaluate our method, we tested it on a real-world dataset. For LIDAR semantic segmentation training, we used our RELLIS-3D dataset to train the model and reduced the classes as follows:

- **Ground**: Grass, Dirt, Asphalt, Concrete, Puddle, Mud;
- **Bushes**: Bushes
- **Obstacles**: Vehicle, Barrier, Log, Pole, Object, Building, Person, Fence, Tree, Rubble

We trained a semantic segmentation model using the DeepLabv3 model [119]. We tested two types of inputs: one where we used only one frame of radar as input and another where we accumulated sequential three radar frames as input. The confusion matrices for the different inputs are shown in Fig. 3.10. The two different inputs achieved 72.27% and 72.96% mean intersection over union (mIoU), respectively. There was only a slight improvement by using three channels as input.

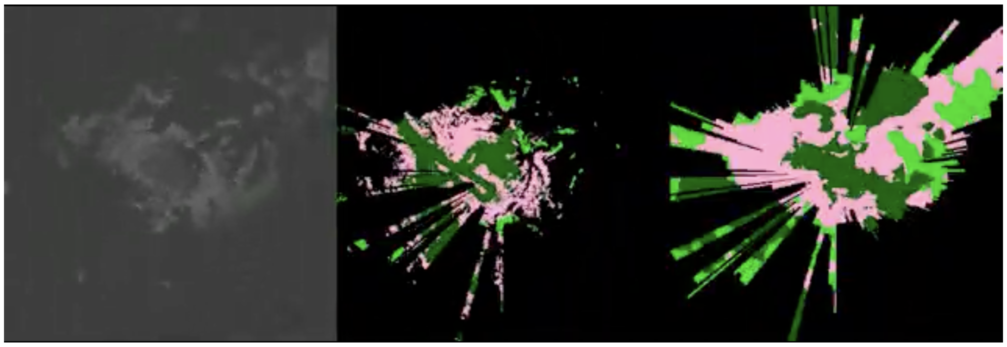


Figure 3.11: Revised image caption: "Radar Semantic Segmentation Results: From left to right are the input radar image, ground truth annotations, and predicted semantic segmentation results.

Among the three classes, the ground class achieved the highest IoU, while obstacles had the lowest IoU.

3.4 Conclusion

In this chapter, we explore semantic segmentation problems for different sensor data. For image semantic segmentation, we focus on the semantic segmentation framework for images from off-road environments. We present an off-road semantic segmentation (OFFSEG) framework for fine semantic segmentation on two off-road datasets. The OFFSEG framework shows affirmative results in achieving a good mIoU in off-road environments. The sub-class segmentation within the traversable region from OFFSEG can be used for robust scene understanding and optimized path planning for navigation through off-road environments. Our framework also performs well in solving the class imbalance issue, which is a significant challenge in the benchmarks of the RELLIS-3D dataset.

For LIDAR point clouds, we investigate how to transfer labels from one LIDAR point cloud dataset to another LIDAR point cloud dataset. We propose a boundary-aware domain adaptation approach for semantic segmentation of LIDAR point clouds. Our approach is based on a model that can extract domain shared features and domain private features. We use the Gated-SCNN to enable the domain shared feature extractor to keep boundary information in the domain shared features and utilize the learned boundary to refine the segmentation results. We conduct experiments on the

SemanticKITTI, SemanticPOSS, and SemanticUSL datasets, and the results show that our model can keep almost the same performance on the source domain after adaptation and achieve an 8%-22% mIoU performance increase in the target domain.

For radar data, we focus on training a radar semantic segmentation model for off-road environments. We propose an automatic pipeline to transfer semantic segmentation labels from LIDAR to radar. We train an image-based semantic segmentation model for radar data based on the generated labels and achieve a 72.96% mIoU.

4. CROSS-MODAL DEEP FEATURES[*]

Images and point clouds provide distinct information to robots, and finding correspondences between data from different sensors is crucial for various tasks, such as localization, mapping, and navigation. While learning-based descriptors have been developed for single sensors, there has been limited research on cross-modal features. In this chapter, we will explore the development of cross-modal features through dense contrastive learning. To achieve this, we will propose our own loss function and models for both 2D images and 3D point clouds. Our approach will enable the learning of robust representations that can capture shared information across different modalities.

4.1 Method

4.1.1 Tuple-Circle Loss

Contrastive learning, an approach to self-supervised learning, allows the model to learn rich representative features. Contrastive representation learning aims to learn such a feature space where similar sample pairs stay close while dissimilar ones (called negative samples x^-) are far apart (see Fig.4.1). In a single modality setting, each sample in a dataset is treated as a different instant (called anchor sample x). Similar counterparts (called positive samples x^+) for training are created by random augmentations, e.g., rotating an image, gaussian blur. We consider an image-lidar pair acquired from the same scene and their augmented counterparts as a dataset in our setting and the pixel and point corresponding to the same physical locations as the positive pair.

In cross-modal learning, it is ideal to have a loss function that can handle multiple positive and negative samples from different modalities. Many loss functions have been proposed for contrastive learning. Sun et al. introduced Circle Loss, a unified loss function for deep feature learning based on pair similarity optimization [120]. The Circle Loss is derived from a pair similarity optimization viewpoint and incorporates self-paced weights and margins (see Eq.(4.1)).

[*]The content of this chapter was published in [4]

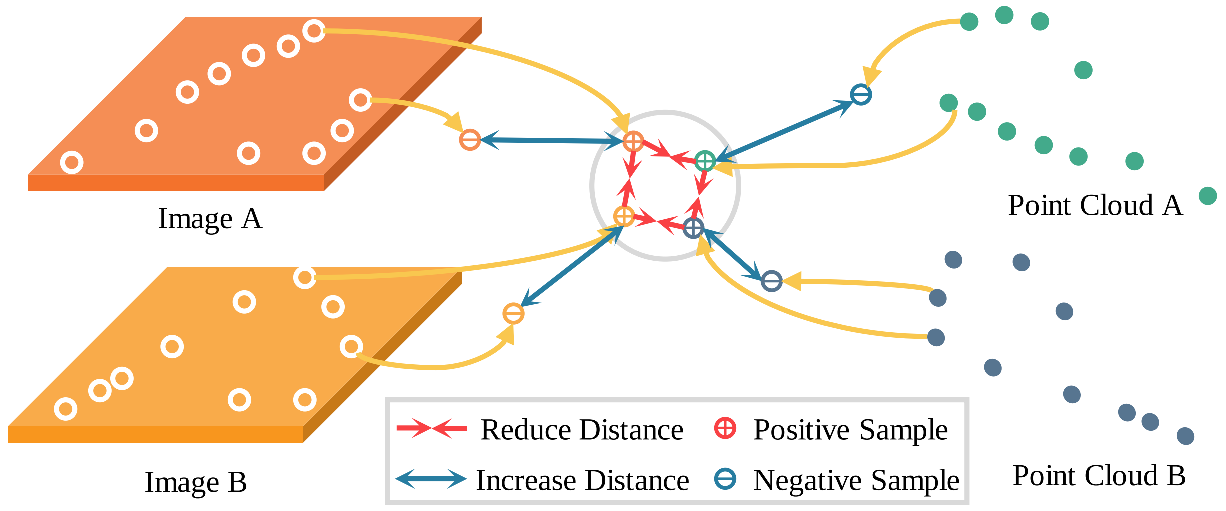


Figure 4.1: **Cross-modal Contrastive Learning**: Image A/B and Point Cloud A/B are acquired from the same scene. The contrastive learning process is to try to reduce the distance between the positive pairs and increase the distance between negative pairs.

$$\begin{aligned} \mathcal{L}_{circle} &= \log\left[1+\mathcal{S}^{-}\mathcal{S}^{+}\right] \\ \mathcal{S}^{-} &= \sum_{j=1}^{L}\exp\left(\alpha_j^{-}\left(s_j^{-}-\triangle^{-}\right)\right) \\ \mathcal{S}^{+} &= \sum_{i=1}^{K}\exp\left(\alpha_i^{+}\left(s_i^{+}-\triangle^{+}\right)\right) \end{aligned} \tag{4.1}$$

In Circle Loss, the self-paced weights are defined as $\alpha_j^{-} = \gamma \max(s_j^{-}+m, 0)$ and $\alpha_i^{+} = \gamma \max(1+m-s_i^{+}, 0)$. L and K represent the number of negative and positive samples, respectively. The negative pair margin is set as $\triangle^{-} = m$ and the positive pair margin is $\triangle^{+} = 1-m$. The Circle Loss function has only three hyper-parameters: a scale factor γ, a margin m that controls the radius of the decision boundary, and similarity measures s^{+} and s^{-} between the features of positive and negative samples, respectively, defined as $s^{+} = f\left(x, x^{+}\right)$ and $s^{-} = f\left(x, x^{-}\right)$.

Our preliminary testing indicates that the self-paced weights play a crucial role in the convergence of cross-modality learning. The weights of negative pairs, α_j^{-}, decrease as the distance between the negative pairs increases, ensuring that features from the two different modalities are

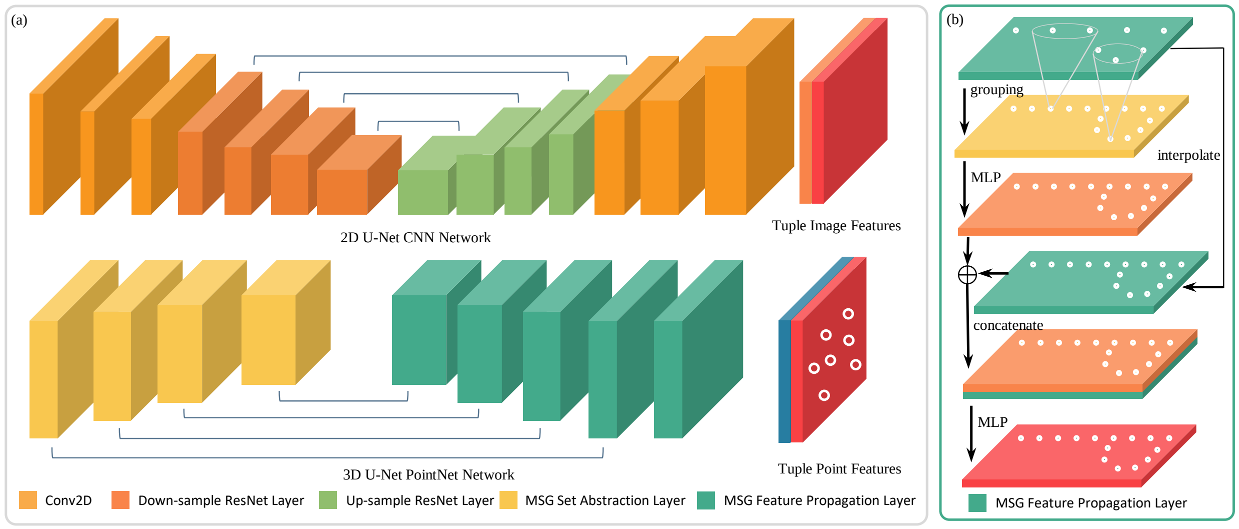


Figure 4.2: **Architecture of 2D and 3D Model** (a) The 2D model includes 3 Convolutional networks and four ResNet layers to downsample the input and 4 upsample the ResNet layer to upsample the features into the original size. The 3D model includes four multi-scale grouping set abstraction layers to downsample the input points and four multi-scale grouping feature propagation layers to upsample the features. (b) The modified feature propagation layers include multi-scale grouping operators.

not pushed too far apart before convergence to the same feature space. Similarly, the self-paced weights of positive samples, α_i^+, also have a similar effect. As the positive samples become close in feature space, the weights decrease, allowing optimization to focus on other pairs.

However, like other approaches[121, 122, 123], Circle Loss also lacks consideration of the unique properties of cross-modal learning, and it employs a unified representation to represent the features, which are all of the same sizes. To address this limitation, we propose Tuple-Circle Loss generalized Circle Loss for cross-modal feature learning.

In cross-modal learning, samples from different modalities can share common information but also have modality-specific information [124, 3, 125]. To better capture this information, we represent features from each modality as a k-tuple, separating shared information and private information. The features are represented as $x_f = [x_{t1}, x_{t2}, \dots x_{tk}]$, where $k-1$ vectors represent shared information and one vector represents private information. For inter-modality similarity, we measure $s^+inter = f\left(x, x^+\right)$, and for cross-modality similarity, we measure $s^+crossk = f\left(x_{tk}, x_{tk}^+\right)$.

Our final loss function is expressed as:

$$\mathcal{L}_{tuple} = \log\left[1 + (\mathcal{S}^{-}_{inert} + \mathcal{S}^{-}_{cross})(\mathcal{S}^{+}_{inert} + \mathcal{S}^{+}_{cross})\right] \tag{4.2}$$

where $S^{+}_{inter} and S^{-}_{inter}$ are sum of similarity measurement within modality as described in Eq. (4.3) and $S^{+}_{cross} and S^{-}_{cross}$ sum of similarity measurement for cross modality as shown in Eq. (4.4).

In an Image-Point Cloud cross modality setting, we represent the features as $x_f = [x_{sh}, x_{pr}]$ where x_{sh} denotes shared features between image and point cloud and x_{pr} represents modality-specific private features in point cloud or image. During training, we use one set of a positive pair(x, x^{+}) and $N-1$ negative pairs $\left[(x, x^{-}_{1}), \ldots, (x, x^{-}_{N-1})\right]$ from one modality. For inter-modal learning, we have:

$$\begin{aligned} \mathcal{S}^{-}_{inter} &= \sum_{j=1}^{N-1} \exp\left(\alpha^{-}_{img_j}\left(s^{-}_{img_j} - \triangle^{-}\right)\right) + \sum_{j=1}^{N-1} \exp\left(\alpha^{-}_{pcd_j}\left(s^{-}_{pcd_j} - \triangle^{+}\right)\right) \\ \mathcal{S}^{+}_{inter} &= \exp\left(\alpha^{+}_{img}\left(s^{+}_{img} - \triangle^{+}\right)\right) + \exp\left(\alpha^{+}_{pcd}\left(s^{+}_{pcd} - \triangle^{+}\right)\right) \end{aligned} \tag{4.3}$$

For cross modality, we can have 4 cross-modality positive pairs and $2N-2$ negative pairs by combination:

$$\begin{aligned} \mathcal{S}^{-}_{cross} &= \sum_{j=1}^{2N-1} \exp\left(\alpha^{-}_{sh_j}\left(s^{-}_{sh_j} - \triangle^{-}\right)\right) \\ \mathcal{S}^{+}_{cross} &= \sum_{i=1}^{4} \exp\left(\alpha^{+}_{sh_i}\left(s^{+}_{sh_i} - \triangle^{+}\right)\right) \end{aligned} \tag{4.4}$$

In this paper, we use cosine similarity to measure the similarity between two features. For inter modality, we use the whole vector to compute cosine similarity $s^{+} = \frac{x_f x^{+}_f}{|x_f||x^{+}_f|}$ and for cross modalities, we only use the shared part of the feature vector to compute the similarity $s^{+}_{sh} = \frac{x_{sh} x^{+}_{sh}}{|x_{sh}||x^{+}_{sh}|}$.

4.1.2 Models

We develop a dual U-Net structure framework(see Fig.4.2) for cross-modality dense feature learning and describe the network structure next.

There are several approaches to learning from LiDAR data. One way is to project LiDAR data onto a spherical plane and process further using the 2D CNN network. By using this point cloud representation, we are able to unify both models under 2D CNN architecture[126]. However, the projection process leads to the loss of 3D information of the point cloud. For generality of our model and allow our model to be applied to point clouds from other 3D sensors, we chose PointNet/PointNet++ as the model for point cloud feature learning. PointNet/PointNet++ was designed to process raw point clouds directly, and several architectures for 3D data[127] are designed on a similar structure and applied to our method.

We first explore the vanilla multi-scale group (MSG) version of PointNet++, which was designed for the Point Cloud semantic segmentation task [43]. The model has an encoder consisting of a farest-sampling layer and sets abstraction layers, and the decoder is comprised of feature propagation layers. However, the model has difficulty producing good point-level features. A potential reason for this is that the feature propagation layers in the decoder create point features by interpolating the neighborhood features, and there is no learnable weight for the interpolation. This limits the description ability of the feature.

We make a simple modification by adding a set-abstraction layer before each feature propagation layer to overcome this limitation. As shown in Fig.4.2(b), we create new features for up-sampled points by grouping their neighborhood in the old point clouds and passing it to a multilayer perceptron (MLP). After that, we concatenate the new features with the interpolated features and feed them to the MLP like standard feature propagation layers described in [43]. This modification allows the set-abstraction layers to provide more neighbor information and learn-able weights for interpolation.

We propose a 2D CNN network with a U-Net structure for image feature learning. The encoder is a ResNet having four ResNet layers[128], while the decoder is an inverse of the encoder that replaces the first convolution layer with the deconvolution layer to up-sample the feature. Convolutional neural networks (CNNs) are inherently unable to handle non-trivial geometric transformations due to the fixed geometric structures in their building modules[129]. The PointNet++, on the other hand, is more flexible for modeling geometric transformations and has a broader receptive field view. To make the 2D model more flexible, we replace the second convolutional layers of the

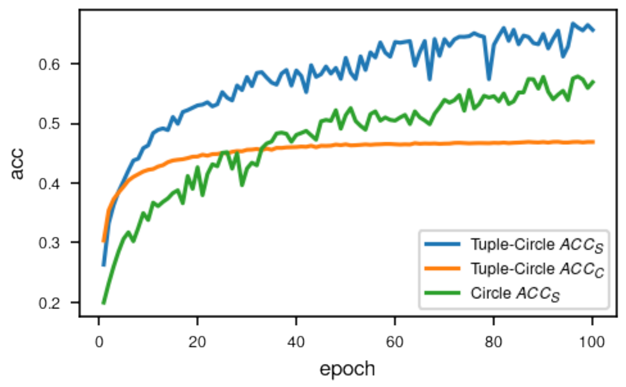


Figure 4.3: Validation accuracy of Tuple-Circle Loss (blue) vs. Circle Loss (green) during the training Process

ResNet block in the encoder with deformable convolutional layers[130]. The same modification is made in the decoder model.

4.2 Experiment Results

4.2.1 Experiment Setting

4.2.1.1 Datasets

In order to verify our approach, the evaluation dataset is required to have the image and point cloud pairs with known pixel-point level correspondence (known sensor calibration). Datasets meeting this requirement exist in papers, such as [131, 1]. Other works like [67, 66] construct datasets in order to have dense correspondence between point and pixel. However, not to constraint to our model to dense point cloud and show the generality of our method, we use camera image and LiDAR scan pairs sequences from the KITTI360 dataset[131]. We trained our models on sequences 0, 2, 4, 5, 6, 7, and 9 of KITTI360 and performed validation on sequence 3. During training, the image sequences were randomly cropped to 256×512, and the LiDAR point cloud was down-sampled to 10000 points.

4.2.1.2 Implementation

A lambda workstation with two NVIDIA Titan RTX was used for training, and, Pytorch library [132] was used to implement the models. The learning rate was initialized at 0.01 and decayed every epoch by 0.985 for 100 epochs. Finally, we used AdamW optimizer to optimize the models[133], and the total size of the feature vector was 256 for all the models.

4.2.2 Evaluation

Since our goal is to find the unified cross-modal features between two modalities and not to detect key points for matching, we evaluate our features by randomly sampling 500 points from two modalities and calculating the percentage of correct matches. For inter-modal matching, we use full features $[x_{sh}, x_{pr}]$ to compute cosine similarity. For cross-modal matching, we only use the shared part features x_{sh} to perform matching. In the following part of the paper, ACC_I and ACC_P denote the inter-modal matching accuracy of image and point cloud, respectively. ACC_C denote the cross-modality matching accuracy using full feature vectors $[x_{sh}, x_{pr}]$, and ACC_S denotes the cross-modal matching accuracy using shared feature vectors x_{sh}.

4.2.3 Tuple-Circle Loss vs. Circle Loss

In our preliminary test, other widely used contrastive loss functions [121, 134, 123] can easily allow the two models for different modalities to learn good inter-modal features while training simultaneously. However, two models cannot or only slowly converge to learn cross-modal features. Thus, we compare the proposed Tuple-Circle loss function with the Circle loss function [120, 67]. Fig.4.3 shows the accuracy changes during training. There are three lines in Fig.4.3 which denote the ACC_S of Tuple-Circle loss (blue) and Circle loss (green), and ACC_C of Tuple-Circle loss. We can see that the Tuple-Circle loss function converges faster than the Circle loss function and converges to better results. Another interesting observation of Tuple-Circle loss is that the full features can also be used to distinguish across modalities and show better performance at the early stage. However, after some epochs, the full features cannot get better results and are outperformed by the shared features. The Table 4.1 shows the comparison of different model settings. Rows 1 and

2 show that using Tuple-Circle loss results in better feature learning for inter and cross-modality matching. However, cross-modal matching accuracy is much lower than inter-matching accuracy. Fig.4.4 shows that the distance between more than 40% of mismatched cross-modal pairs is less than 1.5 pixels after projecting the LiDAR point cloud on the image.

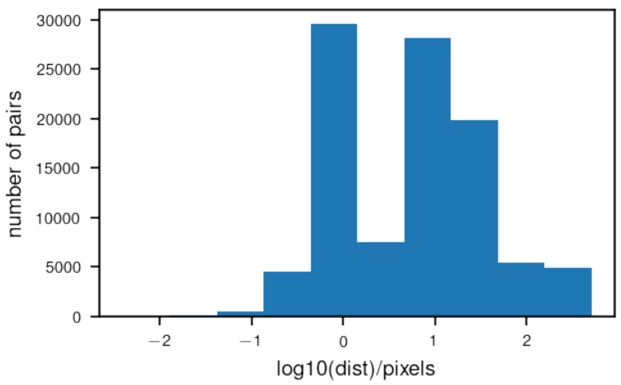


Figure 4.4: Histogram of distance between the mismatched cross-modal pairs in log_{10} scale. x-axis denotes the distance between the mismatched cross-modal pairs in log_{10} scale; y-axis denotes the number of pairs.

4.2.3.1 Models Comparison

We also test the performance of different model setups described in setion4.1.2 on feature learning. Table 4.1 shows the comparison of different model settings; DCN denotes the use of Deformable convolution layers in the 2D model, and ASFP denotes using the set abstraction forward propagation. The results show that ASFP layers help to learn better point cloud features and cross-modality features (highlighted in bold in Table.4.1). According to [67], a broader reception field help to learn inter and cross features, and a Deformable convolution layer can provide a larger and more flexible reception field [130]. However, the DCN layer did not help us learn cross features from our experiment

Table 4.1: Comparison of Different Model Settings

Tuple	DCN	ASFP	ACC_I	ACC_P	ACC_C	ACC_S
✗	✓	✓	96.1%	91.5%	-	57.9%
✓	✓	✓	**99.9%**	**97.3%**	**46.9%**	**66.7%**
✓	✗	✗	99.9%	78.6%	45.3%	63.1%
✓	✗	✓	99.9%	97.4%	47.0%	67.4%

Table 4.2: Comparison of Different Shared Feature Size

Shared Size	ACC_I	ACC_P	ACC_C	ACC_S
64	99.9%	97.5%	45.0%	67.9%
128	99.9%	97.4%	47.0%	67.4%
192	99.9%	97.3%	46.3%	68.0%

4.2.3.2 Shared Features Size Comparison

We analyzed the impact of various shared feature sizes on feature learning performance. Table.4.2 presents the results. It was noted that the intra-matching performance improved with the implementation of Tuple-Circle loss. However, the cross-modality matching results remained largely unchanged with varying shared feature sizes. This leads us to believe that the cross-modality shared information learned does not have a high level of entropy.

4.2.4 Visualization

What do the models learn across different modalities? To answer this question and visualize the learned features, we employ the Cosine-KMean Clustering method [135]. We use the full features $x_f = [x_{sh}, x_{pr}]$ as input to the method and attempt to classify them into 200 clusters. The results are shown in Fig. 4.6(b). We repeat the same clustering on the full features of the corresponding point cloud and display the results in Fig. 4.6(e). The results in Fig. 4.6(e) appear to be noisy due to the low resolution of the point cloud, while Fig. 4.6(b) shows a clear pattern, indicating more texture properties than geometric properties, which has been demonstrated in [136, 137].

For the shared features x_{sh}, we concatenate features from both modalities and classify all 200 clusters using the Cosine-KMean Clustering method. The clustering results for pixels and points are shown in Figs. 4.6(b) and (d), respectively. The results demonstrate that both image and point cloud clusters overlap, and the clustering results reveal position-related properties. In Fig. 4.6(d), we show the clustering results of image pixel positions. We use the positions $(row, column)$ of all pixels in an image as a dataset and perform KMean clustering on the dataset. This results in a figure similar to a Voronoi diagram, as shown in Fig. 4.6(d). Comparing Fig. 4.6(d) and (c), similarities can be observed. Fig. 4.6(c) shows that the shared features may encode 2D position information, as revealed in Fig. 4.6(d), and the 3D depth information of the corresponding point cloud, as shown in Fig. 4.6(f).

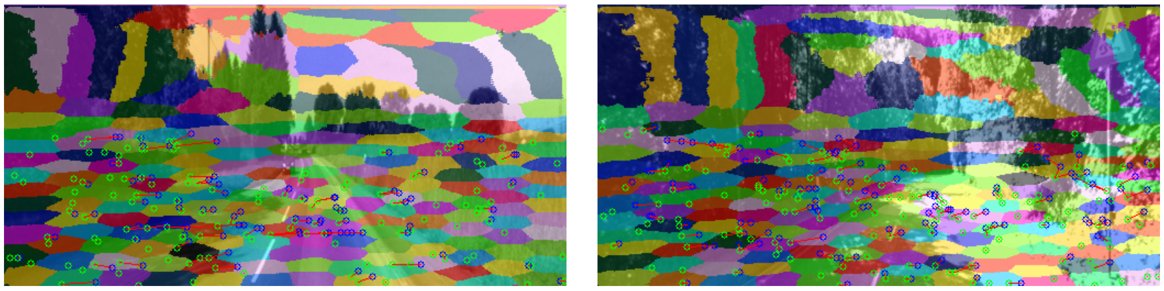

(a) (b)

Figure 4.5: **Cross-modality Matching Visualization**: Green and Blue circles represent points from the point cloud and image individually; Red lines denote the mismatching; (Note single green due to the small distance between two mismatched points.)

We also analyze the error matching across different modalities based on the clustering results. Figs. 4.5(a) and (b) show some mismatches. The visualization reveals that most mismatches occur within the same clusters or on the border of two clusters.

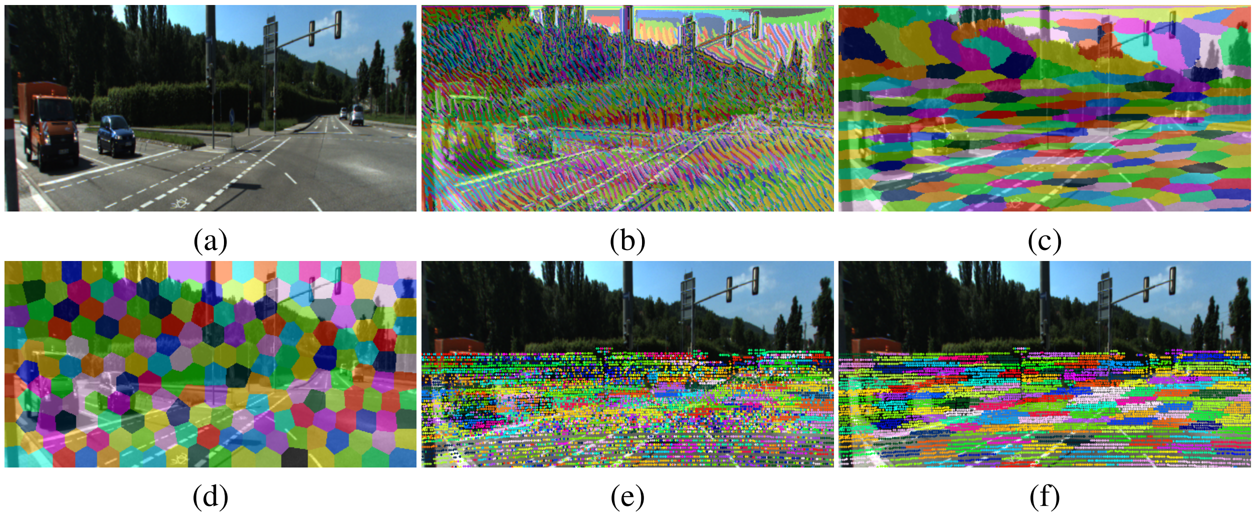


Figure 4.6: **Feature KMean Clustering Visualization**: (a) Original image (b) Full feature clustering results of image; (c) Shared feature clustering results of image; (d) Pixel coordinate clustering results; (e)Full feature clustering results of point cloud; (f) Shared feature clustering results of point cloud;

4.3 Conclusion

This chapter presents a novel Tuple-Circle loss function for cross-modality feature learning to extract common features from different sensor information modalities. Our experiments demonstrate that the Tuple-Circle loss leads to faster and more efficient convergence of the model. Additionally, we introduce a variant of PointNet++ architecture for point cloud analysis, which enhances inter-modal and cross-modal matching. A U-Net CNN architecture with deformable convolutional layers is utilized for image feature learning, and its effectiveness is evaluated in our experiments. Using the Cosine-Kmean clustering method, the learned features are visualized, and it is shown that our approach enables the two models of distinct modalities to learn geometrically meaningful shared features.

5. MULTIMODAL SENSOR CALIBRATION

Multi-modal sensor calibration is an important aspect in the field of autonomous vehicles and robotics. It involves synchronizing and aligning multiple types of sensors, such as cameras, LiDAR, and radar, to ensure that the data collected from each of them is consistent and accurately reflects the physical environment. This process is crucial in creating a unified representation of the environment for perception and decision making tasks. In this chapter, we focus on two specific types of multi-modal sensor calibration: camera-LiDAR calibration and radar-LiDAR calibration. These are important calibrations as they involve sensors that have different operating principles and modalities, yet play a crucial role in the perception and decision making pipeline. In the following sections, we will propose our methods involved in calibrating these specific sensor pairs.

5.1 Semanitc CAMERA-LIDAR Calibration[*]

5.1.1 Method

The following formula describes the transformation relationship of a point p_i^L from the LiDAR local coordinate system to the camera projection plane:

$$\begin{bmatrix} p_i^{uv} \\ 1 \end{bmatrix} = \mathrm{K} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathrm{R} & \mathrm{t} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p_i^L \\ 1 \end{bmatrix} \tag{5.1}$$

- $p_i^L = \begin{bmatrix} x_i^L & y_i^L & z_i^L \end{bmatrix}^T$ represents the coordinate of point p_i in Lidar local frame;
- t represents the 3×1 translation vector;
- R represents the 3×3 rotation matrix;
- K represents the 3×3 intrinsic matrix of camera;

[*]The content of this section was published in [5]

- $p_i^{uv} = \begin{bmatrix} u_i & v_i \end{bmatrix}^T$ represents the coordinate of point p_i in Camera projection plane;

We assume that the intrinsic calibration of the camera and LiDAR have been successfully performed. The extrinsic calibration parameters, which include the rotation R and translation t, are given by Eq.5.1. In addition to the point cloud and image coordinates, we also utilize the dense semantic labels of the point cloud, L^L, and the image, L^C, to improve the accuracy of the extrinsic calibration. These semantic labels can be obtained through human annotation or a learned model.

We treat the semantic label values of each point cloud and its corresponding image pixel as two random variables, X and Y, and the goal of the calibration is to maximize the mutual information between these two variables. To achieve this, the following three operations must be performed:

1. **Transformation Matrix Computation**: $P^{uv} = \text{Proj}(P^L, R, t)$ projects 3D points from LiDAR coordinate to camera coordinate;

2. **Image Samping**: $\widetilde{L}^C = \text{Sample}(L^C, P^{uv})$ samples semantic label values from image labels based on projected LiDAR coordinates.

3. **Mutual Information Estimation**: $I(X, Y) = \text{MI}(\widetilde{L}^C, \widetilde{L}^L)$ estimates mutual information based on the samples from the semantic labels of LiDAR points and corresponding image pixels.

Therefore, our full optimization objective function can be written as Eq.5.2

$$R, t = \arg\max_{R,t} \text{MI}(\text{Sample}(L^C, \text{Proj}(P^L, R, t))) \tag{5.2}$$

5.1.2 Optimization

Optimizing the cost function in Eq.5.2 aims to find the optimal values of the rigid-body transformation parameters between LiDAR and the camera. Any optimization technique that can converge to the global optimum can be employed, with gradient-based optimization methods being popular due to their fast convergence properties. we make the cost function fully differentiable to allow for optimization using gradient-based methods. We describe the steps to make the transformation matrix computation, image sampling, and mutual information estimation differentiable, resulting in the whole algorithm presented in Algorithm.1. With this approach, we can efficiently and accurately optimize the cost function and obtain the correct values for the extrinsic calibration parameters between the LiDAR and the camera.

5.1.2.1 Transformation Matrix Computation

The computation of $P^{uv} = \text{Proj}(P^L, R, t)$ involves a rigid 3D transformation T, which can be represented as a matrix exponential ($T = \exp(H)$). The matrix H can be parameterized as a sum of weighted basis matrices of the Lie algebra $se(3)$ ($H = \sum_{i=0}^{5} v_i B_i$) as described in [138]. This allows for the representation of T with six parameters (as seen in Eq.5.3), which can be optimized using standard partial derivative computations to compute the gradient during optimization.

$$T = \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} = \exp\left(\sum_{i=0}^{5} v_i B_i\right) = \sum_{j=0}^{\infty} \frac{(\sum_{i=0}^{5} v_i B_i)^n}{n!} \tag{5.3}$$

5.1.2.2 Image Sampling

Differentiable image sampling is widely used in deep learning for computer vision [139]. The sampling operation can be written as

$$\widetilde{l_i^C} = \sum_{h}^{H} \sum_{w}^{W} l_{hw}^{C} k\left(u_i - h; \Phi_x\right) k\left(v_i - w; \Phi_y\right) \tag{5.4}$$

The parameters Φ_x and Φ_y represent the sampling kernel $k()$, which determines the method of image interpolation (e.g., bilinear). l_{hw}^C refers to the semantic label of a pixel located at (h, w) on the image, and $\widetilde{l_i^C}$ is the corresponding semantic label of the point p_i after it has been projected onto the image plane.

```
Algorithm 1: 3D Calibration
  input  : Lidar Point Cloud $P$, Point Cloud Labels $L^P$, Image Labels $L^C$, Camera Intrinsic
           Matrix $K$, Initial Transformation Matrix $T_{init}$
  output: Transformation Matrix $T$
  Use $T_{init}$ to initialize $v_i$;
  Use random initialization for MINEnet parameters $\theta$;
  Initialize learning rate $\alpha$, $\beta$, optimizer and learning rates scheduler;
  while not converge do
      $T = \sum_{i=0}^{5} v_i B_i$;
      Sample $b$ minibatch points $P_b$ and labels $L_b^P$ from $P$ and $L^P$;
      $P_b^{u,v}, = \text{Proj}(P_b, T, K)$;
      $\widetilde{L}_b^C = \text{Sample}(L^C, P_b^{u,v})$;
      $MI = \text{MINE}(L_b^P, \widetilde{L}_b^C)$;
      Update MIME parameter: $\theta + = \alpha \bigtriangledown_\theta$ MI;
      Update matrix exponential parameters: $v + = \beta \bigtriangledown_v$ MI;
  end
  Return $T = \sum_{i=0}^{5} v_i B_i$;
```

5.1.2.3 Mutual Information Estimation

Mutual information is a fundamental quantity for measuring the relationship between random variables. Traditional approaches are non-parametric (e.g., binning, likelihood-ratio estimators based on support vector machines, non-parametric kernel-density estimators), which are not differentiable. Several mutual information neural estimators have been proposed in recent works [140, 141, 142]. In our implementation, we use MINE[140] to estimate mutual information. This method uses the Donsker-Varadhan (DV) duality to represent MI as

$$\widehat{I(X;Y)}_n = \sup_{\theta \in \Theta} \mathbb{E}_{P_{(X,Y)}} [F_\theta] - \log \left(\mathbb{E}_{P(X)P(Y)} \left[e^{F_\theta} \right] \right) \tag{5.5}$$

, where $P(X, Y)$ is the joint density for random variables X and Y and $P(X)$ and $P(Z)$ are marginal densities for X and Y. F_θ is a function parameterized by a neural network, where θ are the parameters of the neural network.

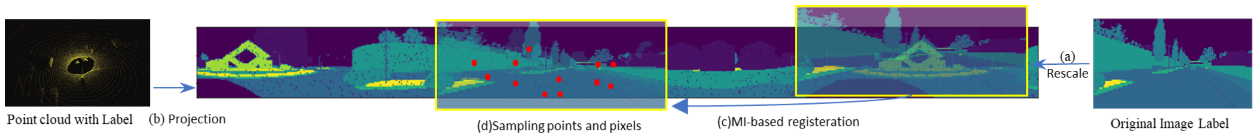


Figure 5.1: Initial Calibration Procedure: (a) Zoom the camera image label; (b) Project LiDAR label into 2D cylinder plane; (c) Register 2D semantic Images; (d) Sample points and pixels from the overlapped area of the two semantic labels.

Algorithm 2: Initial Calibration

```
input : Lidar Point Cloud $P$, Point Cloud Labels $L^P$, Lidar Field of View $FoV^L_H, FoV^L_V$,
        Lidar Channel Number $H^L$ and Ring Point Number $W^L$ Image Labels $L^I$,
        Camera Field of View $FoV^C_H, FoV^C_V$, Image Height $H^I$ and Width $W^I$
output: Initial Transformation Matrix $T_{init}$
$L^P_{cy} = \text{SphericalProj}(P, L^P, H^L, W^L)$;
$W^I_z, H^I_z = \frac{W^L}{Fov^L_V} FoV^C_V, \frac{H^L}{FoV^L_H} Fov^C_H$;
$L^I_z = \text{Zoom}(L^I, W^I_z, H^I_z)$;
Register $L^I_z$ and $L^P_{cy}$ using 2D MI-based method;
Sample pixels $I^P_{cy}$ and $I^I_z$ from the overlapping between $L^P_{cy}$ and $L^I_z$;
Recover image pixels $I^I_s = \text{DeZoom}(I^I_z, W^I_z, H^I_z, W^I, H^I)$;
Recover points $P_s = \text{DeSphericalProj}(I^P_{cy}, P)$;
$T_{init} = \text{PnPsolver}(P_s, I^I_s)$;
```

5.1.3 Initial Calibration

A good initialization is crucial to ensure fast and accurate convergence in the targetless calibration approach. To use semantic information, we formulate the initial calibration process as a combination of 2D image registration, and Perspective-n-Point (PnP) problem [143]. First, we project LiDAR points into a 2D spherical plane [3] and obtain a 2D semantic range image. The LiDAR is a low-resolution 360-degree camera with a partial overlap in the field of view with the ordinary camera, as shown in Fig. 5.1. Then, we zoom the semantic label of the camera image to the same resolution as LiDAR using nearest neighbor interpolation. Afterward, we perform 2D MI-based image registration on the two semantic images. The registration results in raw correspondences between LiDAR points and camera pixels. We sample a few pairs of points and pixels and solve the PnP problem to obtain an initial calibration between the LiDAR and the camera. The complete algorithm is described in Algorithm.2.

The success of the final calibration largely depends on the quality of initialization. We propose using multiple scans to initialize the calibration process to ensure reliable results. We take the mean of the initial parameters obtained from each scan and use the modified Z-score proposed by Iglewicz and Hoaglin [144] to detect and remove outliers. This approach increases the chances of obtaining a good initialization, leading to a more successful final calibration.

$$M_i = \frac{0.6745\,(x_i - \tilde{x})}{\mathrm{MAD}} \tag{5.6}$$

M_i denote the Z-score. x_i denote the data. MAD denotes the median absolute deviation and $\tilde{x}$ denotes the median. We considered the initialization parameters whose absolute value of Z-scores are greater than 3.5 as outliers[144]. If the outliers accounted for more than 60%, we decide the initialization has failed.

5.1.4 Experiment and Results

This section describes the experiments to evaluate the accuracy and robustness of the proposed automatic calibration technique. We first evaluate our methods on a synthetic dataset, then on the real-world KITTI360 [145] and RELLIS-3D [1] datasets.

5.1.4.1 Synthetic dataset

To evaluate the performance and robustness of our method, we created a dataset of paired LiDAR and camera data with semantic labels using the Carla simulator [146]. The simulator can generate 21 classes; however, we utilized a 20-class ontology in our experiments. The camera images have a resolution of 1280×720, and the LiDAR has a horizontal resolution of 800 and a vertical resolution of 64, each channel containing approximately 800 points per frame. Although the point cloud and image are wholly overlapped in our synthetic dataset (as shown in Fig. 5.3(a)), it still provides accurate transformations and semantic labels. However, it should be noted that some real-world sensor and environmental characteristics are not perfectly modeled due to the

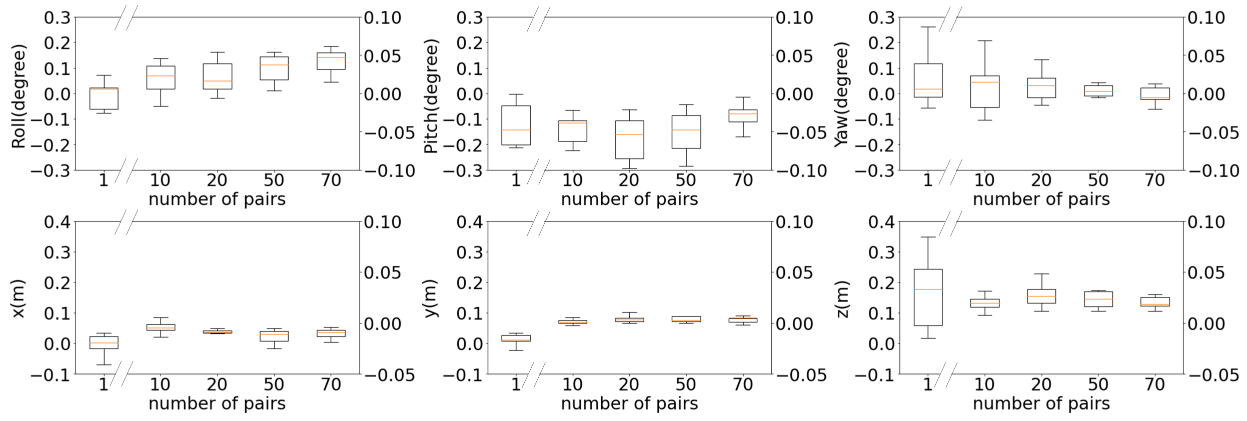


Figure 5.2: Calibration error for varying number of pairs of frames.

limitations of the simulation.

We evaluated the performance and robustness of our methods by testing the effect of the number of data frames on overall performance. Using the ground truth labels, we first estimated the initial transformation as described in Section 5.1.3. We then tested the procedure with 1, 10, 20, 50, and 70 pairs of frames. As shown in Fig. 5.2, the error variance decreases as the number of pairs increases. This is expected as the estimator is able to achieve better mutual information estimation with more data, thereby increasing the scene's diversity and reducing the likelihood of local minima in the optimization.

Next, we evaluated the impact of noisy labels on calibration accuracy. We added random noise to both the image and point cloud labels (as shown in Fig.5.3). The results are displayed in Fig.5.4. As anticipated, the accuracy decreased, and the variance increased with increasing label noise. Although the error from deep learning predictions is not Gaussian in reality, it is easier to control the error ratio by adding noise to ground truth labels. Thus, this experiment demonstrates the effect of varying levels of noise, which will be further explored using model predictions in the next section.

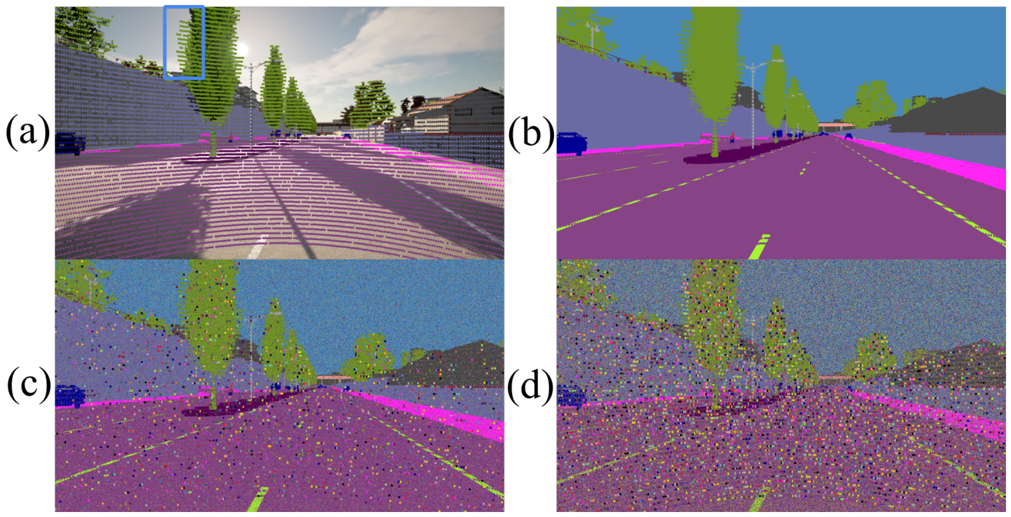


Figure 5.3: (a) Simulated LiDAR point cloud projected on image; (b) Ground truth image label with point cloud; (c) image label and point cloud label with 20% error (d) image label and point cloud label with 50% error.

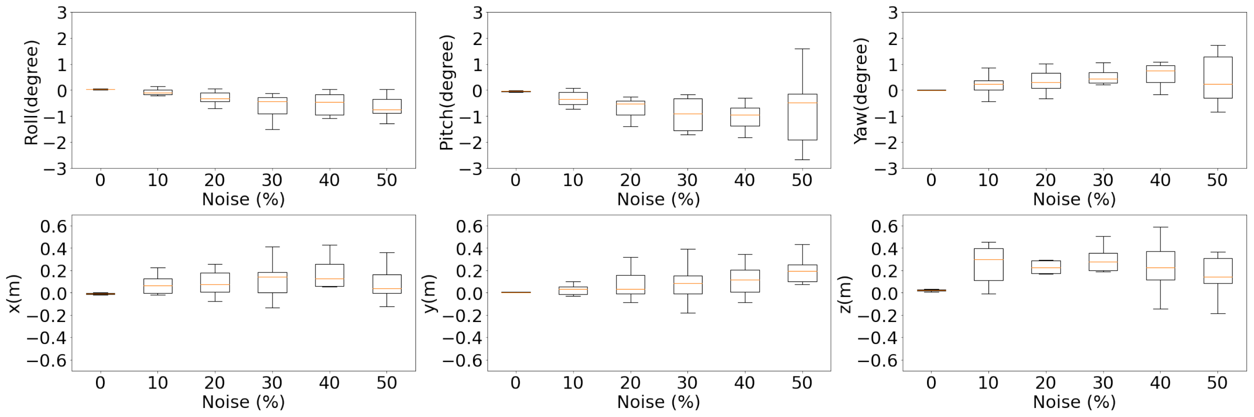


Figure 5.4: Calibration error for different levels of label noise

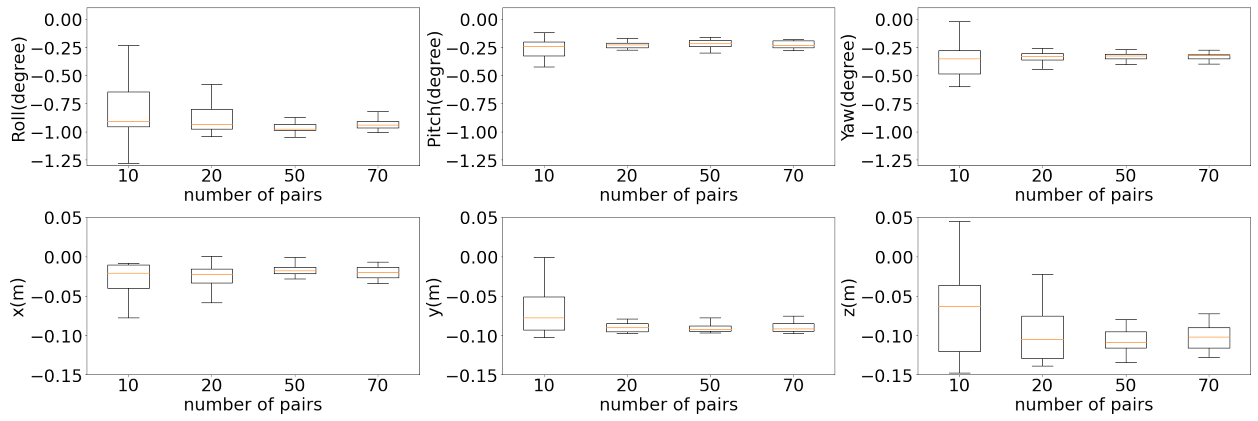


Figure 5.5: Calibration error using different number of pairs on RELLIS-3D Ground Truth Labels

5.1.4.2 Real-world datasets

Despite the benefits of synthetic data in providing accurate ground truth transformations and semantic labels, real-world sensor and environmental characteristics are not perfectly modeled in these simulations. To evaluate the performance of our methods in real-world scenarios, we conducted experiments on two annotated LiDAR and camera datasets, KITTI360 and RELLIS-3D. The KITTI360 dataset was collected in an urban environment, while the RELLIS-3D dataset was collected in an off-road environment. Both datasets provide ground truth calibration parameters between the LiDAR and camera, which we used to assess the robustness of our methods.

We evaluate our method's performance using human-annotated labels and label predictions from deep learning models. We use the HRNet+OCR model [104] to generate predictions for image semantic segmentation. For LiDAR point cloud labeling, we use the SalsaNext model [48]. On the RELLIS-3D dataset, the HRNet+OCR model achieved a mIoU of 48.83%, and the SalsaNext model achieved 40.20%. These models were trained on 20 and 16 classes, respectively, considering the differences in the detection capabilities of each sensor. We merge the classes into 14 common categories for calibration and exclude the `sky` class. Fig. 5.5 shows the results of using ground truth labels, and Fig. 5.6 shows the results of using the prediction from deep learning models. We can see that angle errors using ground truth labels is within 1.5 degree using ground truth labels, and the translation error is within 0.15 meters. Unsurprisingly, the results of label prediction have

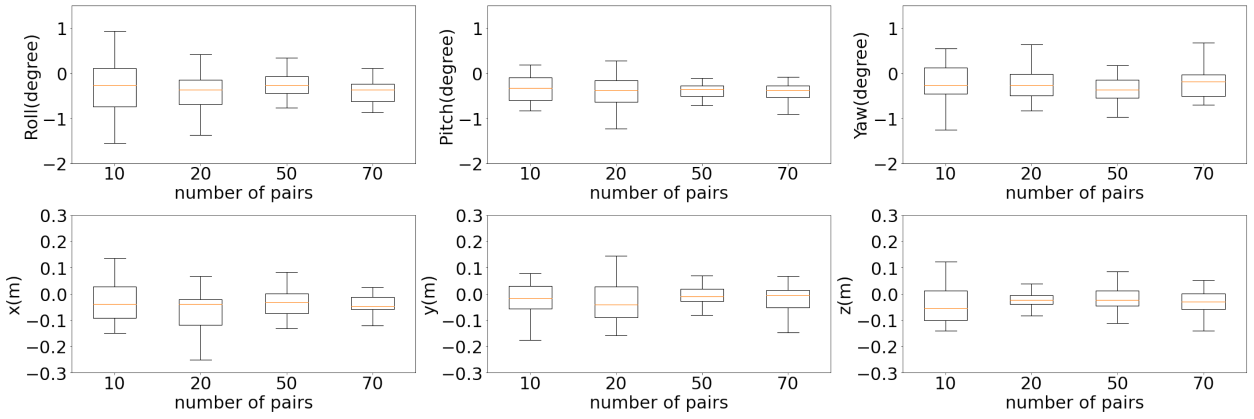


Figure 5.6: Calibration error using different number of pairs on RELLIS-3D Label Predictions

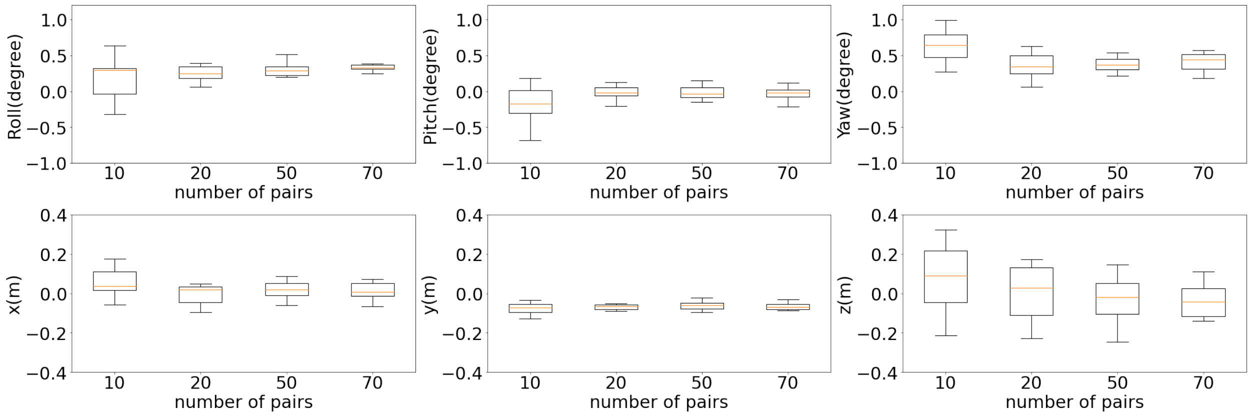


Figure 5.7: Calibration error using different number of pairs on KITTI360 Ground Truth Labels

more significant errors. Moreover, the error of the ground truth label results has an obvious bias. The annotation process might cause this. In RELLIS-3D, each image was annotated by humans individually. At the same time, the LiDAR scans were registered as an integrated scene and annotated together. Therefore, compared with image labels, LiDAR scan labels have better sequential consistency. Meanwhile, the labels generated by the deep learning model have better sequential consistency. We used the same two models to train on the KITTI360 dataset. The HRNet+OCR can achieve 31.02% mIoU on the KITTI360 dataset and the SalsaNext can achieve 20.08% on the KITTI360 dataset. This datasets has 45 classes which we merge into 34 for training. We use our proposed method to initialize the calibration and use 10, 20, 50, and 70 pairs of LiDAR-Camera

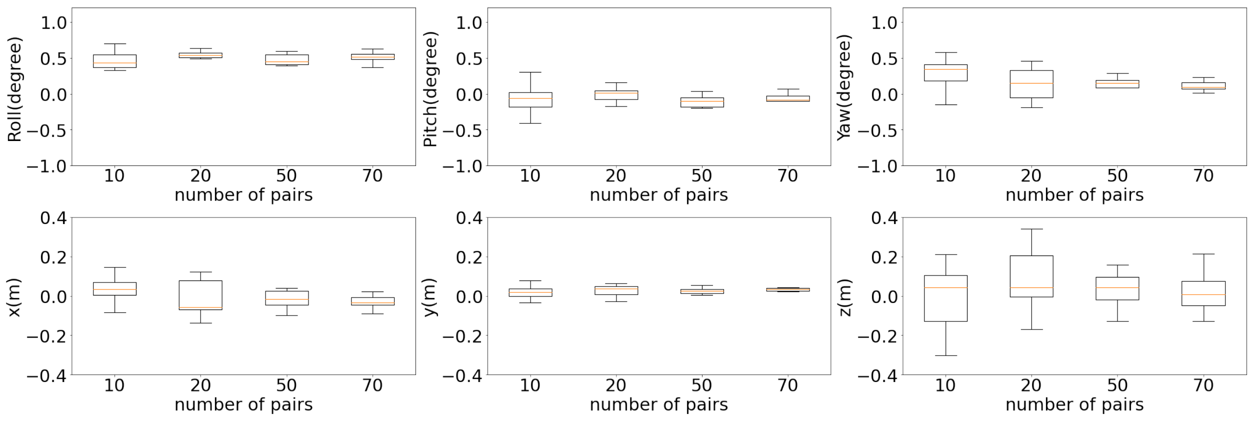


Figure 5.8: Calibration error using different number of pairs on KITTI360 Label Predictions

Table 5.1: Calibration results of KITTI360

Methods	Roll(°)	Pitch(°)	Yaw(°)	X(m)	Y(m)	Z(m)
KITTI360	73.72	-71.37	64.56	0.26	-0.11	-0.83
SOIC	74.08	**-71.21**	64.75	0.11	**-0.12**	-1.29
PMI	71.19	–64.38	58.35	2.28	-0.01	-0.70
Ours	**73.86**	-71.54	**64.69**	0.23	-0.17	**-0.89**

data to perform calibration. Fig. 5.7 shows the results of using ground truth labels and Fig. 5.8 shows the results of using the prediction from deep learning models. We can see that angle error is within 1 degree predictions, and the translation error is within 0.4 meters using both of ground truth labels and label.

To evaluate against baseline calibration approaches, we compared our method with SOIC[91] on the two datasets. We also implemented Pandey's [147] method to use the mutual information between the sensor-measured surface intensities to calibrate the sensors, however we use MINE[140] to estimate the MI instead of the kernel method described in [147]. Our implementation of Pandey's method is noted as **PMI** in Table5.1 and 5.2. All methods were initialized by our proposed method. As shown in Table. 5.1-5.2, our method provide the closest results with the provided parameters of each dataset. Meanwhile, Fig. 5.9 and 5.10 shows that our calibration produces better cross-sensor data fusion than the other two methods.   In the results of PMI, we

Table 5.2: Calibration results of RELLIS-3D

Methods	Roll(°)	Pitch(°)	Yaw(°)	X(m)	Y(m)	Z(m)
RELLIS3D	70.80	67.78	-68.06	-0.04	-0.17	-0.13
SOIC	70.08	68.62	-66.68	0.00	-0.15	4.41
PMI	75.65	69.87	-64.24	-0.08	-0.33	**-0.10**
Ours	**70.24**	**67.63**	**-67.32**	**-0.05**	**-0.19**	-0.06

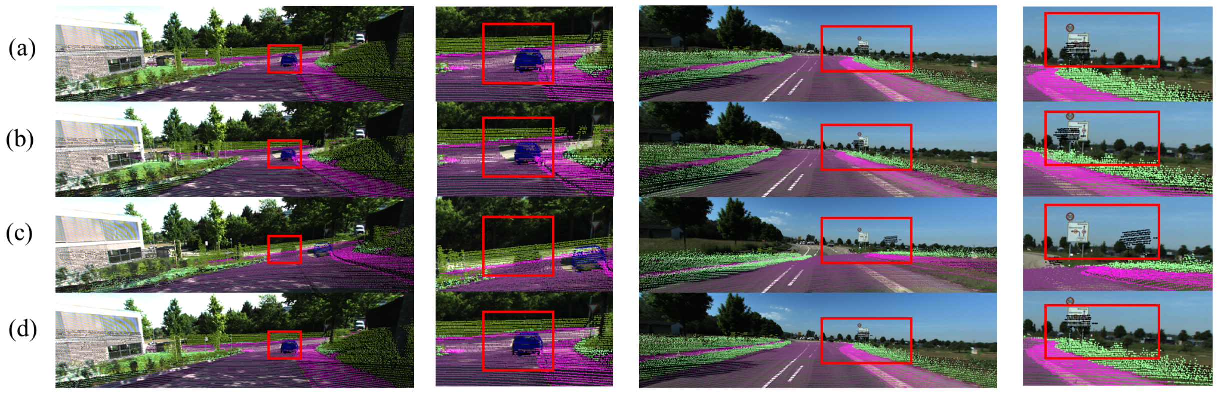


Figure 5.9: (a) KITT360 calibration; (b) SOIC Results; (c) PMI Results; (d) SemCal Results(Ours).

can see there is a significant misalignment on the z-axis. This is because the grayscale image pixel value of the sky is similar to the reflectivity of the grass in LiDAR, so the method tried to align the grass ground from the LiDAR data with the sky in the camera data. Finally, we analyzed the run time of our algorithm. Our experiment used ten labels to run the initialization algorithms and iterated the mutual information estimator, which always converged, for 4000 iterations. We ran the experiment 10 times and computed the mean run time. The initialization procedure, which only needs to be run once, took 30 seconds, and the calibration itself takes 156 seconds. Although this algorithm cannot be run in real-time, it is still fast enough to run online as a background process.

5.1.4.3 Implementation

In the experiments, we implement our algorithm using Pytorch [148]. The neural mutual information estimation is a three- layers neural network. The first layer is a linear layer followed by a Relu activation function [149]. The input size equals two times the number of the data channels,

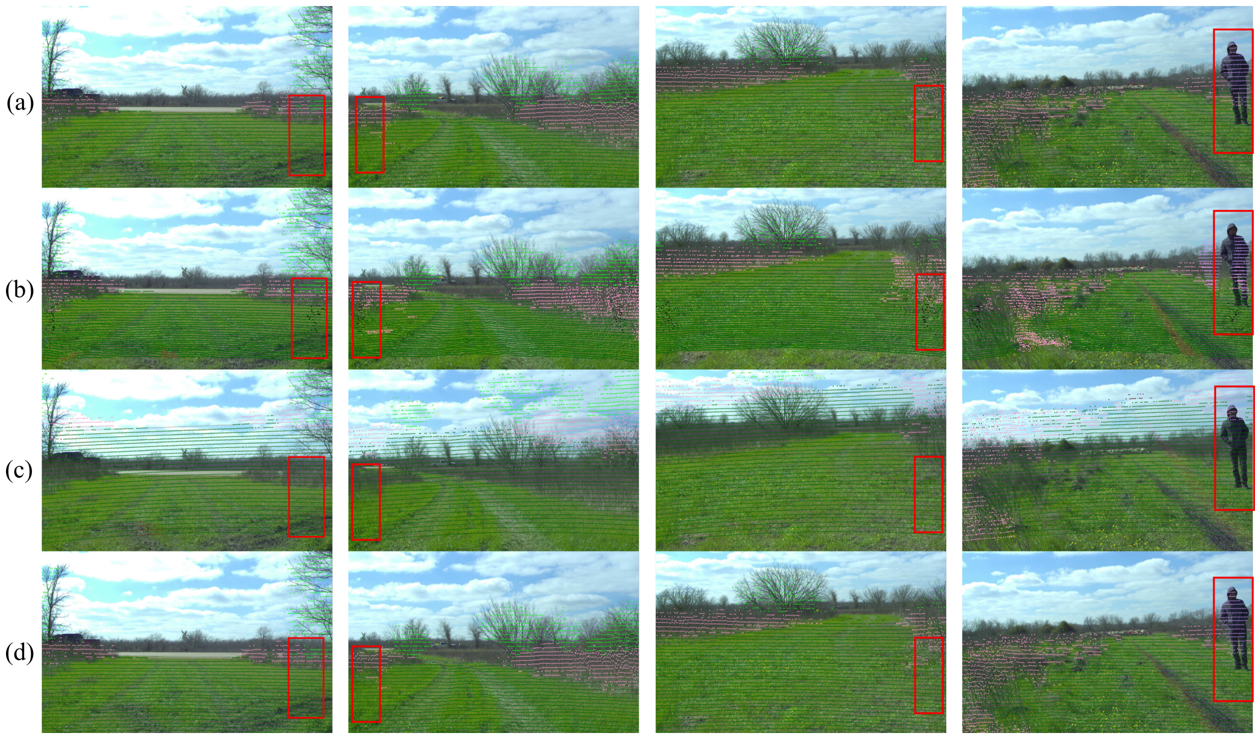


Figure 5.10: (a) RELLIS-3D calibration; (b) SOIC Results; (c) PMI Results; (d) SemCal Results(Ours).

and the output is 100. The second layer is also a linear layer followed by a Relu activation function, and the output size is 100. The last layer is again a liner layer, and the output size is 1. We used Adam optimizer [150] to maximize the objective function [5.2]. For image sampling, we use a bilinear kernel to sample the label pixels. We set $n = 10$ in Eq. 5.3 to estimate the transformation matrix.

5.2 RADAR LIDAR CALIBRATION

5.2.1 Method

5.2.1.1 Calibration Target Design

Calibrating both LIDAR and RADAR requires a target that is visible to both sensors. For RADAR, it is desirable to have a target with a high RADAR Cross-Section (RCS) to improve the detection rates. Marine aluminum RADAR reflectors (see Fig. 5.11 (a)), which are cheap and readily available, can be used for this purpose, but they can affect the LIDAR measurements.

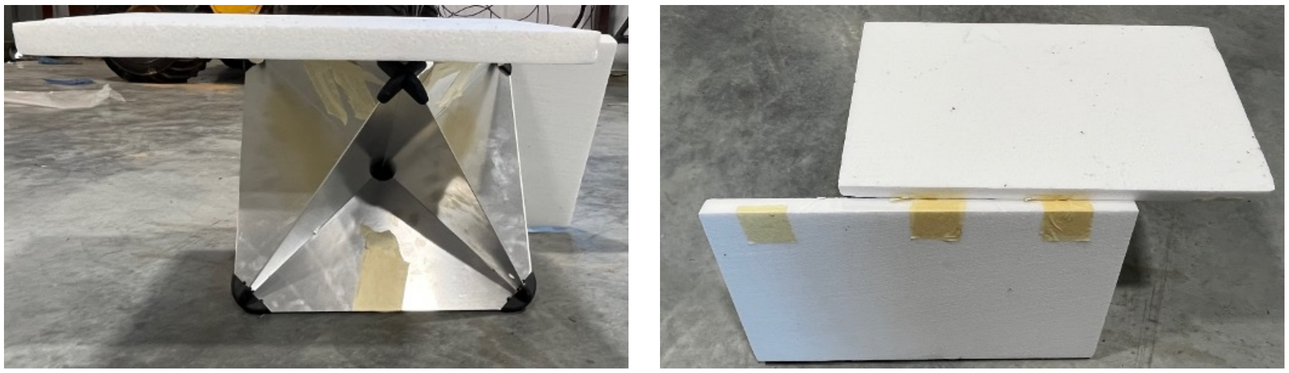

(a) Side view (b) Front view

Figure 5.11: Aluminum cctahedral RADAR reflector front with two side covered by two polystyrene foams

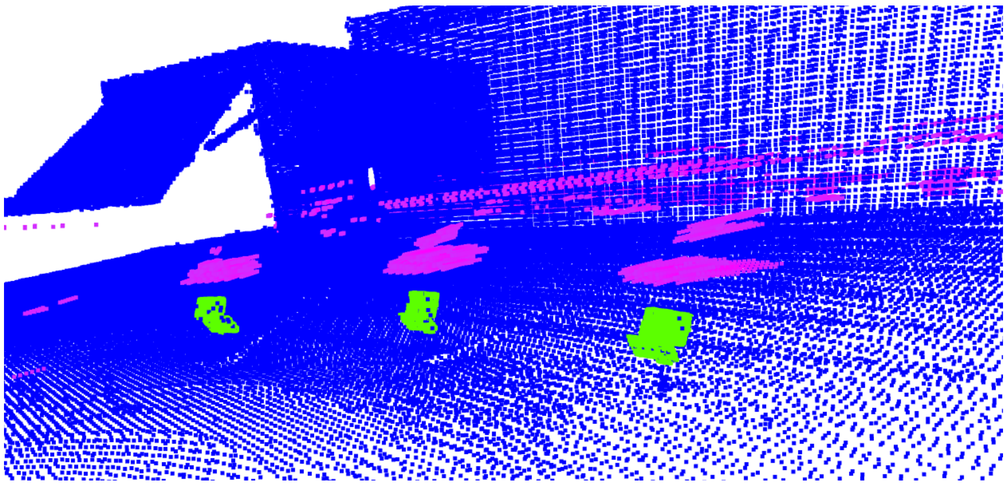


Figure 5.12: Registered LIDAR Point Cloud (blue), Filtered RADAR measurements(Purple), Extracted Target Point Cloud(green)

Furthermore, their small size makes them difficult to detect by LIDAR. To overcome this, we use two polystyrene foams to cover two sides of the reflector and construct a non-symmetrical shape as see Fig. 5.11 (b), which has a larger visible area for LIDAR, making it easier to register and determine the reflector's position in the scene, which will be used for calibration.

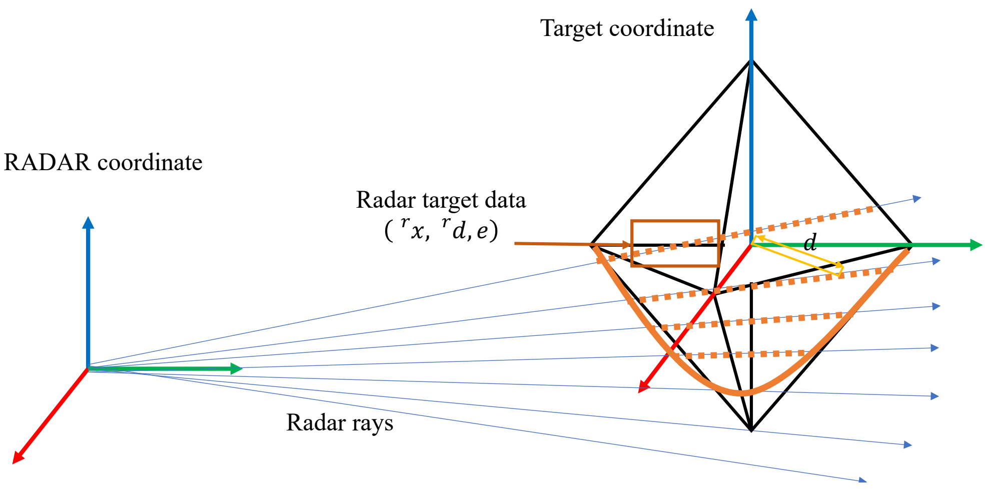


Figure 5.13: Sample the RADAR measurement data $({}^{r}\boldsymbol{x}_i, {}^{r}\boldsymbol{d}_i, \boldsymbol{e}_i)$ around the target centers; Compute the data distance d between the LIDAR target center and the RADAR target center

5.2.1.2 Data Collection

To ensure accurate and undistorted data from both sensors, we have employed the IMU data from the LIDAR sensor to detect the static LIDAR and RADAR frames. We used the FPFH feature [151] and RANSAC global registration to register all the static LIDAR frames together, resulting in an initial pose of all the LIDAR data in the scene. We then refined the pose of each LIDAR frame using the Iterative Closest Point (ICP) algorithm to generate a dense LIDAR point cloud, which was then utilized for target detection to prevent any incomplete frames. By collecting data from multiple robot movements, we obtained each LIDAR frame's position within the registered point cloud, thus eliminating the need for target extraction for each frame. For the RADAR target extraction, we first filtered out most of the raw RADAR by setting a threshold. Then we used raw LIDAR and RADAR transformation to get the raw data target measurement, which was obtained by manually measuring. We defined the target center as the first measurement in the RADAR data. The data collection and pre-processing results is shown in Fig.5.12.

5.2.2 Optimization

We develop our calibration method using a gradient descent optimization framework implemented with PyTorch. Our objectives for the calibration are three-fold: 1) minimizing the reprojection error, 2) reducing the regression error of an MLP, and 3) minimizing the ray pass loss, which considers whether a ray passes through the target or not. The transformation matrix $\boldsymbol{T}$ between the LIDAR and RADAR sensors can be represented in many ways. While optimizing, the rotation matrix $R(\boldsymbol{w})$ was represented as axis-angle representation $\boldsymbol{w}$ ($\mathfrak{so}(3) \rightarrow \mathrm{SO}(3)$). While evaluation, we represent the rotation using Euler angles $\boldsymbol{\theta} = [\theta_x, \theta_y, \theta_z]$. The translation is represented by three variables $\boldsymbol{t} = [t_x, t_y, t_z]$.

5.2.2.1 Reprojection Loss

To compute reprojection loss, we first convert the LIDAR target center point ${}^{l}\boldsymbol{x}_{l,i}$ from the LIDAR sensor coordinate to the RADAR sensor coordinate using the following equation:

$$ {}^{r}\boldsymbol{x}_{l,i} = {}^{r}_{l}R \cdot {}^{l}\boldsymbol{x}_{l,i} + {}^{r}_{l}\boldsymbol{t} \tag{5.7} $$

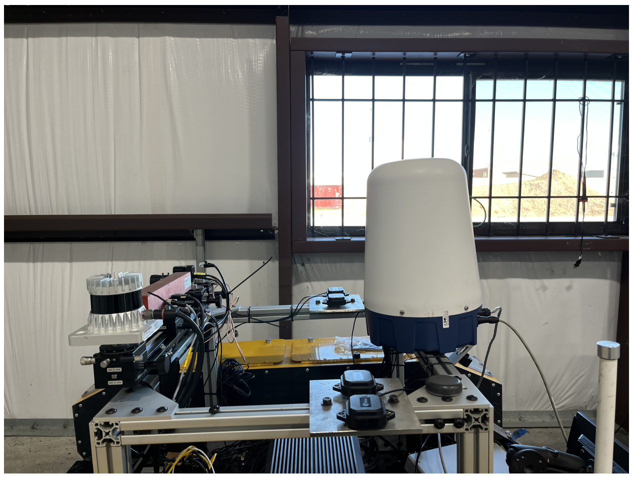
Where ${}^{r}_{l}R$ is the rotation matrix and ${}^{r}_{l}\boldsymbol{t}$ is the translation vector. Next, the point ${}^{r}\boldsymbol{x}_{l,i}$ is converted to spherical coordinate ${}^{r}\boldsymbol{s}_{l,i} = [{}^{r}r_{l,i}, {}^{r}\theta_{l,i}, {}^{r}z_{l,i}]$. The RADAR measurement ${}^{r}\boldsymbol{s}_{r,i} = [{}^{r}r_{r,i}, {}^{r}\theta_{r,i}, -]$ is assumed to be the ground truth. The absolute differences loss (L_1) is applied to the azimuth angle and radial distance between the transformed LIDAR target center data and the RADAR measurement.

$$ l_{rep} = w_r L_1({}^{r}r_r, {}^{r}r_l) + w_\theta L_1({}^{r}\theta_r, {}^{r}\theta_l) \tag{5.8} $$

where w_θ and w_r are two weights that can be set manually to adjust the optimization direction and balance the relative importance of radial distance and azimuth angle, due to their differing value scales.

5.2.2.2 Regression Loss

Getting a 6 DoF calibration between LIDAR and RADAR is challenging due to the missing 3D information in RADAR measurement. However, as shown in [94], the RCS is correlated with the target's 3D position in the RADAR coordinate, leading to the modeling of the relationship between RCS and target position as a curve model in the paper. Inspired by this work and recent


(a) Side view

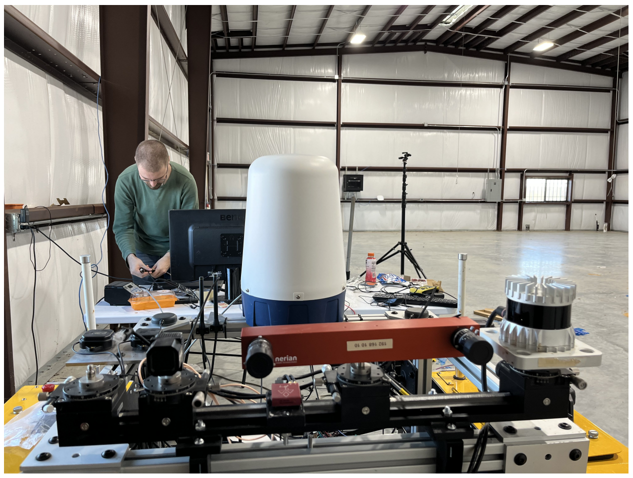

(b) Front view

Figure 5.14: Clearpath Warthog with one Ouster OS1 128 Channel LIDAR and one Navtech CIR-DEV RADAR

advancements in NeRF, we use a small Multi-Layer Perceptron (MLP) to model the relationship between the RADAR sensor pose relative to the target and the return energy. Instead of just using the target center to model the relationship, we use the neighborhood of the target centers too. See. Fig.5.13, we sample the RADAR measurement data around the target centers and represent the data as $({}^{r}\boldsymbol{x}_{r,i}, {}^{r}\boldsymbol{d}_{r,i}, e_i)$, where ${}^{r}\boldsymbol{x}_{r,i}$ denote the position of the RADAR data in RADAR coordinate, ${}^{r}\boldsymbol{d}_{r,i}$ represent the ray direction from the RADAR sensor origin to the measurement, and e_i represent the return energy, which is raw data value of the Navtech RADAR.

In the optimization process, we transform the RADAR measurement $({}^{r}\boldsymbol{x}_{r,i}, {}^{r}\boldsymbol{d}_{r,i}, e_i)$ from the RADAR coordinate to the local target coordinate based on the current estimated RADAR-LIDAR extrinsic calibration ${}^{w}_{r}T$ and the target pose ${}^{w}_{t}T$ in the scene which can be obtained during the data collection steps described in 5.2.1.2. After the transformation ,we got data $({}^{t}\boldsymbol{x}_{r,i}, {}^{t}\boldsymbol{d}_{r,i}, e_i)$ in the target local coordinate.

Based on [152] and experiment, we use Eq.5.9 to apply position encoding to each of the three coordinate values in the direction ${}^{t}\boldsymbol{d}_{r,i}$ and position ${}^{t}\boldsymbol{x}_{r,i}$ separately. Eq.5.9 is a mapping from $\mathbb{R}$ into a higher dimensional space $\mathbb{R}^{2L}$. Positional encoding facilitates the network to optimize parameters by easily mapping input to higher-dimensional space. [152] showed that using a high-frequency function for mapping original input enables better fitting of data that contains high-frequency variation.

$$\begin{aligned} P(x, 2i) &= \sin\left(2^i \pi x\right) \\ P(x, 2i+1) &= \cos\left(2^i \pi x\right) \end{aligned} \tag{5.9}$$

We use the encoded direction $P({}^t\boldsymbol{d}_{r,i})$ and position $P({}^t\boldsymbol{x}_{r,i})$ as the input of the MLP and predict the corresponding return energy $\hat{\boldsymbol{e}}_i$.

$$\hat{e}_i = MLP(P({}^t\boldsymbol{x}_{r,i}), P({}^t\boldsymbol{d}_{r,i})) \tag{5.10}$$

During optimization, we minimize the absolution difference (L_1 loss) between the ground truth value $\boldsymbol{e_i}$ and the predictions $\hat{\boldsymbol{e}}_i$

$$l_{mlp} = L_1(\hat{e}_i, e_i) \tag{5.11}$$

5.2.2.3 Ray-pass Loss

To ensure that our optimization results align with the properties of the sensors, we added a ray pass loss to optimization objectives. The high return energy of RADAR target measurement should be the result of a ray passing through the target from the RADAR sensor origin. To simplify, we modeled the target as a ball rather than an octahedron. The loss is defined as in Equation 5.12, where d is the distance between the ray and the center of the target in the target coordinate system, and r is the radius of the circumscribed sphere. If the distance is greater than the radius, it indicates that the ray does not pass through the target, and the difference between the distance and radius is used as the penalty. If the distance is smaller than the radius, the penalty is set to zero.

$$l_{ray} = \max\left(0, d_i - r\right) \tag{5.12}$$

5.2.2.4 *Total Losses*

The overall optimization objective is:

$$l = w_{rep}l_{rep} + w_{mlp}l_{mlp} + w_{ray}l_{ray} \tag{5.13}$$

where the w_{rep}, w_{mlp} and w_{ray} are weights for each loss.

5.2.3 EXPERIMENT

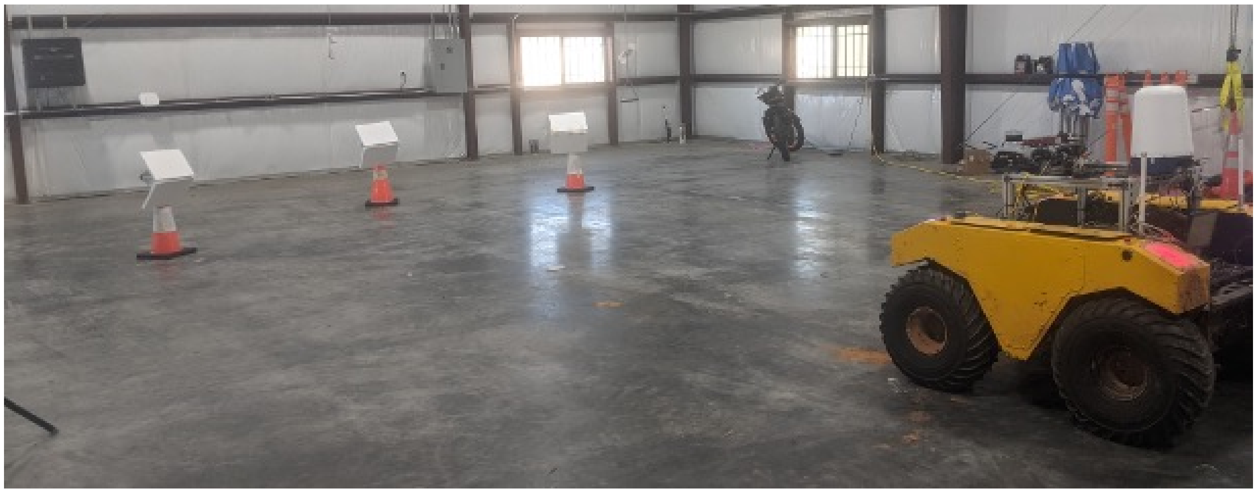
5.2.3.1 *Sensors and Experiment Setting*

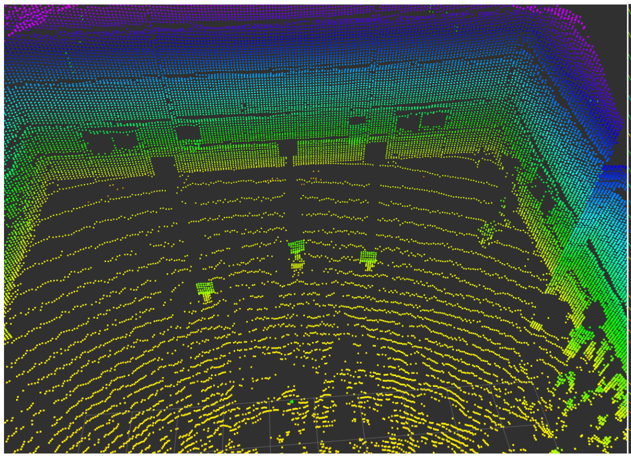
The experiment involved a Clearpath Warthog mobile robot equipped with an Ouster OS1 3D LIDAR and a Navtech CIR-DE RADAR, as depicted in Fig. 5.15. The LIDAR has 128 channels and 2048 points per channel, with an 45 degrees vertical field of view (FoV) and operates at 10 Hz. The RADAR has a range resolution of 0.044m and azimuth resolution of 0.9 degrees, with an 1.8 degrees vertical field of view (FoV) and works a 4 Hz. Both sensors have 360 degrees horizontal field of view (FoV). The sensor setup is shown in Figure 5.14. A 3D-printed wedge was attached to the bottom of the RADAR with an incline of around three degrees.

The experiment used three targets, and the robot was positioned at different locations to collect data, as shown in Fig. 5.15. The experiment was conducted outdoors, and five different target settings were used. After the data processing steps described in 5.2.1.2, 104 paired RADAR-LIDAR data were collected.

5.2.3.2 *Implementation*

Our extrinsic calibration algorithm was implemented using the PyTorch framework and optimized using the Adam optimizer [153]. The MLP consisted of four linear layers, each with a hidden size of 128 and using the ReLU activation function. A learning rate of 0.005 was applied to the network, while the learning rates for the rotation and translation parameters were set to 0.005 and 0.001, respectively. During the experiment, the weights for the reprojection error, regression,


(a) Sensors and Targets Setup for Data Collection


(b) Raw LIDAR data

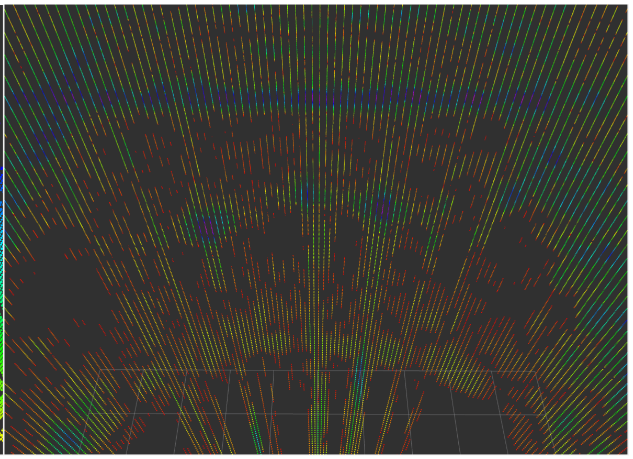

(c) Raw RADAR data

Figure 5.15: Experiment settings: Three targets with fixed positions form a scene, and the robot drives around the targets.

and ray-pass losses were set to 1000, 1000, and 100, respectively. To compute the regression loss, we sampled data around the estimated RADAR target center within a radius of 0.6 meters.

5.2.3.3 Results

To evaluate the quality of the calibration results, we conducted experiments on a real-world dataset with different loss function configurations: including only the reprojection error loss (*rep*), the reprojection error loss and regression loss (*mlp*), the reprojection error loss, regression loss, and ray-pass loss (*mlp+ray*) and the reprojection error loss and ray-pass loss (*rep+ray*). Table 5.3

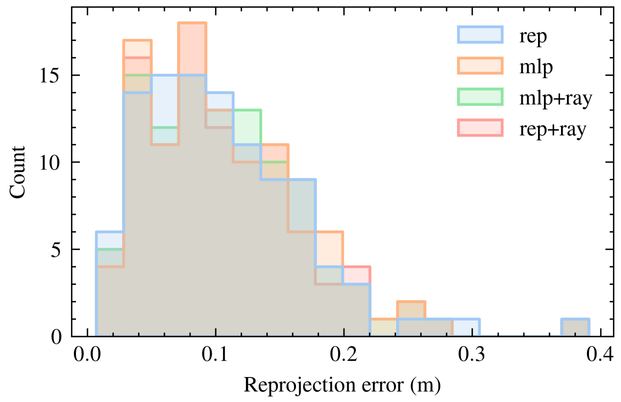


Figure 5.16: Histogram of reprojection errors for RADAR-LIDAR calibration using different loss configuration

presents the initial calibration parameters obtained manually and the results of calibrations using different optimization objectives. The results demonstrate differences, particularly in θ_x and θ_y, and t_z, due to the limitations of the RADAR measurement. However, as previously mentioned, we intentionally introduced a three-degree wedge while installing RADAR sensors. As shown in Table 5.3, all calibration results reflect this. We used the reprojection error in the RADAR coordinate as the evaluation metric, and Fig. 5.16 shows the distribution of the reprojection error. Despite the differences, all four settings have similar reprojection error distributions, which does not necessarily indicate which calibration results are better.

To further evaluate the quality of the calibration results, we examine the relationship between the target and RADAR ray. The RADAR ray from the sensor origin to the RADAR target center

Table 5.3: RADAR-LIDAR Calibration Results

Loss	θ_x	θ_y	θ_z	t_x	t_y	t_z
initial	0.00	0.00	0.00	0.50	-0.25	0.05
rep	0.97	6.97	1.17	0.57	-0.26	0.04
mlp	0.26	2.04	1.01	0.57	-0.25	0.05
mlp+ray	-0.14	2.63	1.05	0.56	-0.26	-0.03
rep+ray	0.08	3.35	1.07	0.57	-0.26	0.01

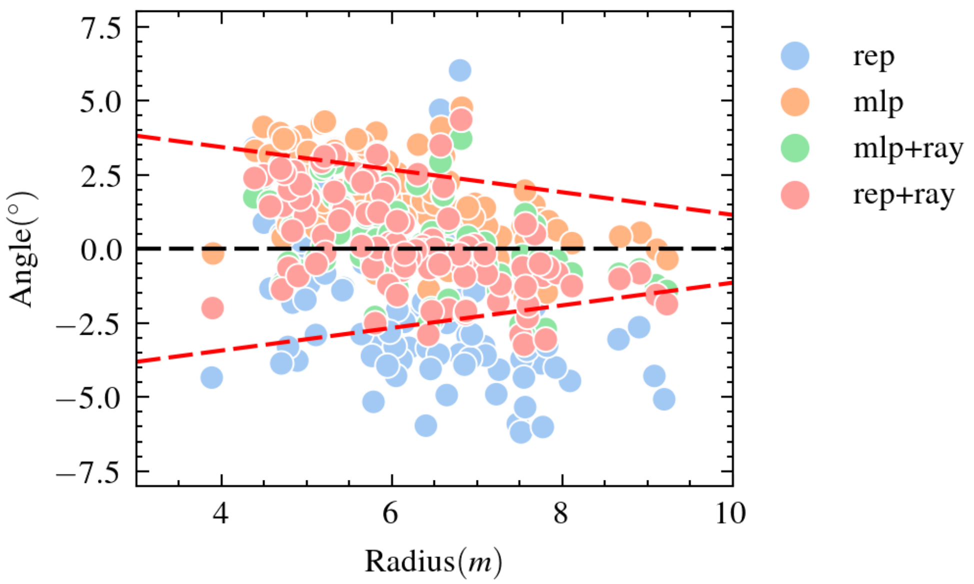


Figure 5.17: LIDAR target centers distribution in the RADAR coordinate. The x-axis represents the radial distance between the target center and the RADAR origin, the y-axis represents the angle between the RADAR x-y plane and the target centers.

must hit the target and return the energy, as indicated by the high return energy. Fig. 5.17 displays the distribution of the LIDAR target centers in the RADAR coordinate after projection using the calibration results. The x-axis represents the radial distance between the LIDAR target center and the RADAR origin. In contrast, the y-axis represents the angle between the RADAR x-y plane and the LIDAR target centers. The two red dashed lines indicate the boundaries where the RADAR ray can hit the target at different distances based on the physical size of the target. From Fig.5.17, we observe that the calibration results that only use the reprojection error loss have many points outside the boundary, which is not physically correct. On the other hand, the calibration results of *mlp+ray* and *rep+ray* have most of the projected points within the boundaries. The *mlp* also has most of the points within or close to the boundary, which implies that the objective function is effective.

In order to evaluate the robustness of our method, we conducted a Monte Carlo analysis by randomly subsampling our dataset to 50% of its original size and ran 100 iterations of optimization on different subsampled datasets. The results are presented as boxplots in Fig. 5.20. As predicted

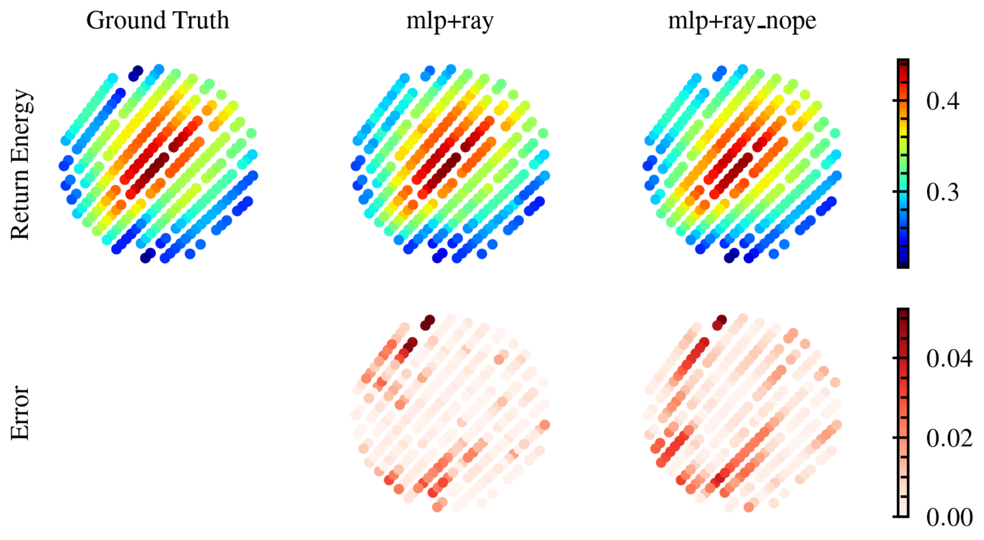


Figure 5.18: Regression results from MLP

and in line with previous studies [93, 94], the parameters t_x, t_y, and θ_z that are well-represented by the RADAR measurements exhibit lower variance compared to the other parameters. In contrast, the parameters t_z, θ_x, and θ_y show larger variance. The results from the *rep+ray* experiment display the highest variance in z, indicating that the ray-pass loss function is highly sensitive to the amount of data. By incorporating the regression loss, which utilizes both reprojection error and the relationship between the return energy and the position of the target center's surroundings, the variance is reduced compared to only using reprojection error and ray-pass, even with fewer data points. Finally, Fig.5.19 demonstrates that the use of positional encoding results in smaller errors in regression. Several examples of the regression are illustrated in Fig.5.18.

5.3 Conclusion

Multimodal sensor calibration is an important issue in computer vision and robotics. A well-calibrated system provides accurate information about the environment and improves the performance of various applications, such as object recognition and tracking. In this chaptere, we discussed for LIDAR-CAMERA calibration and LIDAR-RADAR Calibration.

The first method is the fully differential LiDAR-Camera calibration method that uses semantic

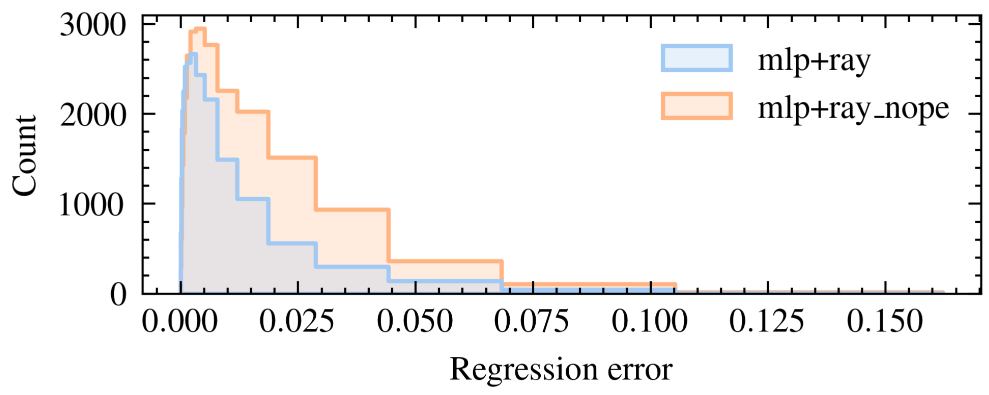


Figure 5.19: Histogram of regression errors of MLP with/without positional encoding

information from both sensor measurements. This method optimizes mutual information between semantic labels to estimate the extrinsic parameters of the sensors. By utilizing semantic information, the method does not require specific targets or hand-measured initial estimates, making it more flexible and user-friendly. The method is also fully differentiable, which allows it to be implemented using popular gradient descent optimization frameworks. Additionally, the use of mutual semantic information opens up the possibility of leveraging sub-semantic intermediate features of deep networks for calibrating LiDAR and camera, and could also be embedded into a deep fusion architecture to estimate rough calibration estimates between sensors.

The second method is the novel extrinsic 6-DoF calibration method for a RADAR-LIDAR system that was presented in a recent study. This method uses a specially designed calibration target that allows both sensors to accurately detect and locate the target within their respective operating parameters. The calibration process involves three optimization objectives: reprojection error, regression error, and ray-pass loss. The proposed method was implemented using a deep-learning framework and optimized via gradient descent. Experiments conducted on real-world data validated the effectiveness of the proposed method and showed significant improvements in the estimation of extrinsic calibration parameters.

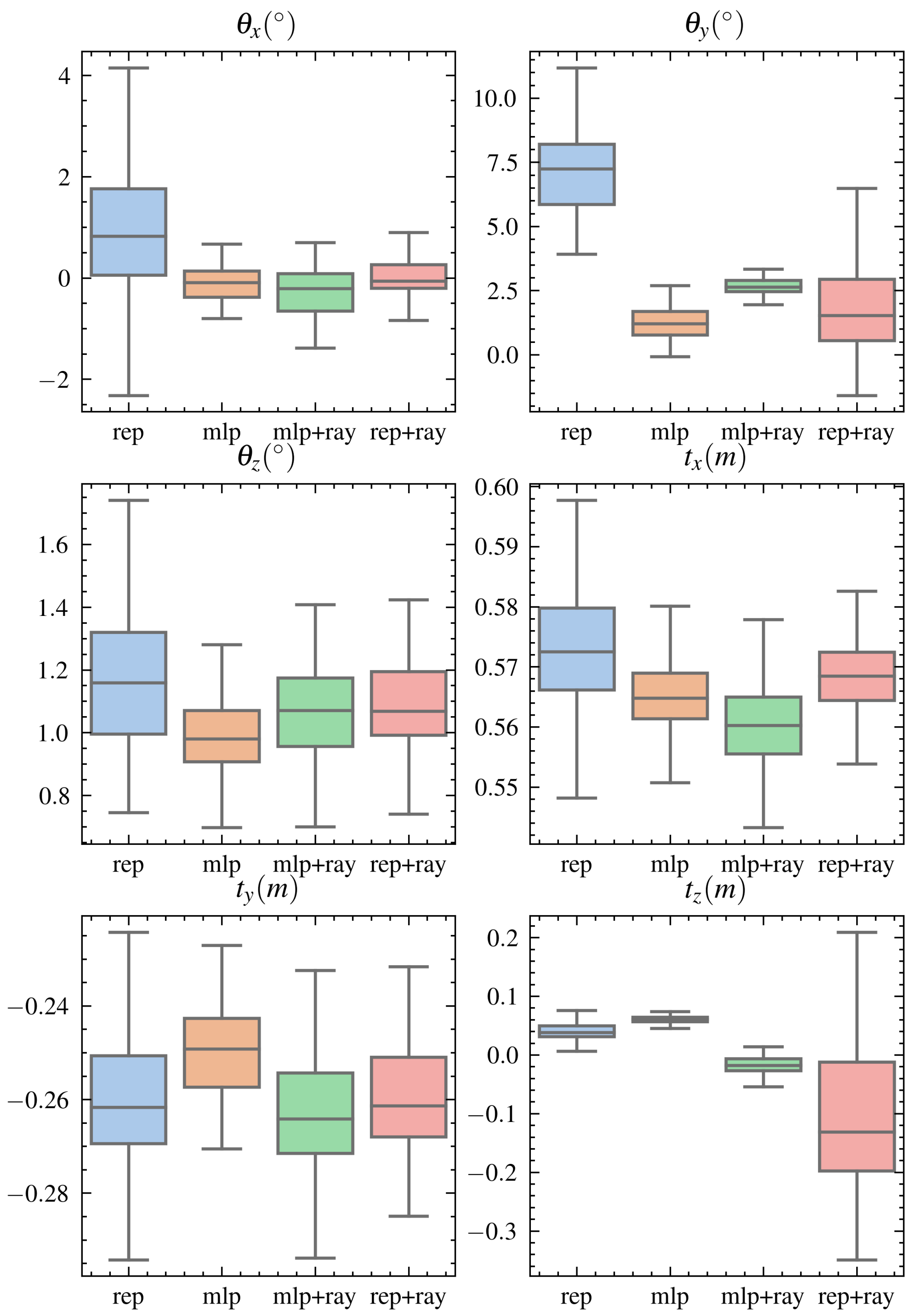


Figure 5.20: Monte Carlo analysis results for RADAR–LIDAR calibration using different loss configurations

6. CONCLUSION AND DISCUSSION

This dissertation emphasizes the critical role of perception sensors in enabling robots to interact effectively with and comprehend their environment, particularly in off-road terrain. The study explores various sensor types and different levels of sensor data representation, from raw data to semantic information and deep features. We have introduced the RELLIS-3D dataset, a new multimodal dataset with pixel-wise and point-wise annotations for off-road environments, which challenges current state-of-the-art deep learning models and underscores the need for further research in these unstructured environments. Since the dataset's publication, several works have followed the research or utilized the dataset for their studies [154, 155, 156, 157, 158, 159, 160, 161, 162]. We are pleased that the dataset has been useful to the community. In addition to the dataset, this dissertation proposes a semantic segmentation framework for off-road image segmentation, a technique for transferring labels from one LIDAR point cloud dataset to another, and a pipeline for transferring LIDAR semantic segmentation labels to radar data. These semantic segmentation-related works have also garnered attention [163, 164, 165, 166, 167, 168, 169]. To overcome the limitation of semantic labels, the study also proposes a technique to learn cross-modal deep features using contrastive learning. After discussing various representations, the study focuses on extrinsic calibration. Using semantic information for cross-modal data association, we address extrinsic calibration for camera-LIDAR calibration. For radar-LIDAR calibration, we use a target-based method and train a neural network to capture the correlation between return energy and lost dimension.

However, when we created the dataset and tried to apply semantic segmentation in off-road environment, we started to think whether the standard semantic segmentation task in computer vision is a good method for robots to understand their environment. There are several problems with Semantic Segmentation or its advanced version Panoptic segmentation[170, 171, 172]. Firstly, semantic segmentation classes are usually pre-defined, such as the 20 classes in the RELLIS-3D dataset and these pre-defined classes are usually object levels. However, the real world has more than 20 classes. And for different tasks, robots may require different levels of detail in their understanding of the environment or properties of the objects . For instance, in a navigation task, robots may only need to determine whether an area is traversable or not. Secondly, manually labeled datasets for semantic segmentation are imperfect, especially in images where borders are often subtle or ill-defined. In off-road environments, the problem becomes even more pronounced as compared to urban or indoor environments due to their unstructured nature.

To replace predefined labels, we initially tried using deep features which can have semantic and geometry information of the scene. However, robots were built, operated, and designed to work with humans. Therefore, interactions with humans are required. And human usually use language to describe and understand the world which might means linguistic description is necessary at some level. With recent developments in large language models (LLM) [173, 174] and multi-modal learning [175], promising directions for open-vocabulary semantic segmentation have been proposed [176, 177, 178]. Open-vocabulary semantic segmentation is a task that aims to segment an image into semantic regions according to text descriptions. For example, given an image of a street scene and a text description "car," the task is to identify all the pixels that belong to cars in the image. There are different methods for performing this task, but one common approach is to use a two-stage framework [179]. The first stage generates class-agnostic mask proposals that cover potential regions of interest in the image. The second stage leverages pre-trained vision-language models, such as CLIP [175], to classify masked regions according to text descriptions. This is very similar to our Off-Seg frame which produce masks and use classifier to provide class. However, because our classifier is fine-tuned on the pre-defined class of the datasets, Off-Seg cannot generalize to the unseen classes. But the Open-vocabulary segmentation still also have problems. The methods relies on supervised mask from current datasets which contain additional information about classes. In addition to the two-stage framework, a more conventional approach is to use the widely-used fully convolution network (FCN). As a dominant method in supervised semantic segmentation, FCN formulates the semantic segmentation as a pixel-wise classification problem. Specifically, given an image, FCN generates a high-resolution feature map, and a set of learned classifiers is applied on each pixel to produce segmentation predictions [179]. However, due to most current pre-trained vision-language models are trained on image level which lead FCN framework performance is lower than the two-stage method. Besides, compared to supervised methods, open-vocabulary methods still can't compete with supervised method. Besides, currently most research still focus on object level segmentation less attention were put the other types of properties.

While mIoU is a commonly used metric to evaluate the performance of semantic segmentation algorithms, it may not necessarily be the best metric for evaluating robotics tasks, especially in off-road environments. The boundaries between different semantic classes are often ill-defined, and imperfect annotation can lead to difficulties in evaluating the performance of these algorithms. To better evaluate the performance of these algorithms in off-road environments, it may be useful to explore and develop new metrics, datasets, or tools that are specifically tailored to robotics tasks. For example, one approach could be to treat semantic segmentation as a component of a larger task in the robotic system, such as navigation. We can evaluate the performance of the overall navigation task under the same settings while using different segmentation methods, and measure metrics such as travel time, collision rate, and minimum distance. This approach would provide a more realistic evaluation of the performance of the algorithms in the context of their intended use.

In this dissertation, we discussed the importance of sensors and data representation in robotics. However, a crucial problem that remains unaddressed is how to effectively integrate high-level semantic information or deep features with other important scene information, such as geometry. This is an essential problem that must be solved for robots to achieve accurate navigation and localization in real-world environments. Traditional methods, such as voxel maps, can store this information but may not be efficient in terms of storage and processing. Recently, new methods have emerged using neural implicit representations, which can directly store and process this information[180, 181, 182, 183, 184, 185, 186]. Neural implicit representations are continuous functions that can be trained on known views of a scene, providing higher resolution and more detail than traditional methods. Furthermore, neural networks can represent a scene as a signed distance function and differential function, enabling their use in robotics tasks such as navigation[187] and localization[188]. Another advantage of neural implicit representations is their ability to synthesize new views, which can be used to create data for other tasks or simulations. Additionally, neural networks can naturally store deep features within the implicit representation[189]. However, there are still some limitations to the current state of neural implicit representations. For example, they can be slow for both training and rendering new views, and they may face difficulties in scaling to large scenes. Moreover, they can only represent static scenes, and they "bake in" lighting, which may limit their accuracy in certain lighting conditions. Additionally, they require accurate camera poses. Despite these limitations, the promising properties of neural implicit representations have received a lot of attention, and much research is being conducted to solve these issues [190]. In conclusion, integrating semantic information and deep features with other necessary scene information is a crucial problem that must be solved for robots to perform effectively in complex environments, and neural implicit representations provide a promising solution to this problem.

Finally, in this dissertation, we proposed a novel approach to multi-modal sensor calibration that differs from traditional methods. Traditionally, common features such as edges, gradients, or optometry trails are used to calibrate different sensor types. Instead, we utilized semantic information for cameras and lidar, since semantic segmentation is a well-established technique for these two sensors. For radar, although we used a target-based method, we trained a neural network to capture the correlation between return energy and target pose. Our method was inspired by recent advances in neural rendering technology [152], which is a type of neural implicit representation that can reconstruct a scene using multiple images and camera poses and generate new images from new viewpoints while capturing the geometry of the scene. This approach may provide a promising direction for calibrating different types of sensors and evaluating the calibration results. One limitation of this approach is that it is sensitive to errors in the sensor pose of each input sensor measurement. If the pose is not accurate enough, the synthesized sensor measurement will not be accurate. However, there are methods to optimize both the reconstruction loss and sensor pose [191]. Therefore, we can leverage this limitation for calibrate and evaluate sensor calibration. Currently, most of the applications are focused on image synthesis, but we can expand it to other types of sensors such as LIDAR and RADAR. Since the original method can capture geometry using only images and camera poses, it is intuitive to combine it with other range sensors. Although radar signals can penetrate some materials, neural representation can capture geometry as well as other features such as semantics and materials. Therefore, we believe that with proper design and sufficient data, it is possible to simulate radar data. However, optimizing both the reconstruction loss and the sensor pose can be challenging. Moreover, even if the network can accurately reconstruct sensor measurements and converge to a certain sensor pose, it is not guaranteed to converge to the correct sensor poses. Therefore, the methods and architecture of the model need to be carefully designed and proper evaluated.

REFERENCES

[1] P. Jiang, P. Osteen, M. Wigness, and S. Saripalli, “Rellis-3d dataset: Data, benchmarks and analysis,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1110–1116, 2021.

[2] K. Viswanath, K. Singh, P. Jiang, P. Sujit, and S. Saripalli, “Offseg: A semantic segmentation framework for off-road driving,” in *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*, pp. 354–359, 2021.

[3] P. Jiang and S. Saripalli, “Lidarnet: A boundary-aware domain adaptation model for point cloud semantic segmentation,” in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2457–2464, 2021.

[4] P. Jiang and S. Saripalli, “Contrastive learning of features between images and lidar,” in *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*, pp. 411–417, 2022.

[5] P. Jiang, P. Osteen, and S. Saripalli, “Semcal: Semantic lidar-camera calibration using neural mutual information estimator,” in *2021 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pp. 1–7, 2021.

[6] D. Maturana, P.-W. Chou, M. Uenoyama, and S. Scherer, “Real-Time Semantic Mapping for Autonomous Off-Road Navigation,” in *F. Serv. Robot.* (M. Hutter and R. Siegwart, eds.), (Cham), pp. 335–350, Springer International Publishing, 2018.

[7] L. D. Jackel, E. Krotkov, M. Perschbacher, J. Pippine, and C. Sullivan, “The DARPA LAGR program: Goals, challenges, methodology, and phase I results,” *Journal of Field Robotics*, vol. 23, pp. 945–973, nov 2006.

[8] S. Thrun, “Winning the darpa grand challenge,” in *Machine Learning: ECML 2006* (J. Fürnkranz, T. Scheffer, and M. Spiliopoulou, eds.), (Berlin, Heidelberg), pp. 4–4,

Springer Berlin Heidelberg, 2006.

[9] A. Hussein, P. Marin-Plaza, D. Martin, A. De La Escalera, and J. M. Armingol, "Autonomous off-road navigation using stereo-vision and laser-rangefinder fusion for outdoor obstacles detection," in *IEEE Intelligent Vehicles Symposium, Proceedings*, vol. 2016-Augus, pp. 104–109, Institute of Electrical and Electronics Engineers Inc., aug 2016.

[10] A. Huertas, L. Matthies, and A. Rankin, "Stereo-based tree traversability analysis for autonomous off-road navigation," in *Proceedings - Seventh IEEE Workshop on Applications of Computer Vision, WACV 2005*, pp. 210–217, IEEE Computer Society, 2005.

[11] R. Manduchi, A. Castano, A. Talukder, and L. Matthies, "Obstacle detection and terrain classification for autonomous off-road navigation," *Autonomous Robots*, vol. 18, pp. 81–102, jan 2005.

[12] Y. Yang, D. Tang, D. Wang, W. Song, J. Wang, and M. Fu, "Multi-camera visual SLAM for off-road navigation," *Robotics and Autonomous Systems*, vol. 128, p. 103505, jun 2020.

[13] G. J. Brostow, J. Fauqueur, and R. Cipolla, "Semantic object classes in video: A high-definition ground truth database," *Pattern Recognit. Lett.*, vol. 30, no. 2, pp. 88–97, 2009.

[14] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The Cityscapes Dataset for Semantic Urban Scene Understanding," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-December, pp. 3213–3223, IEEE Computer Society, dec 2016.

[15] G. Neuhold, T. Ollmann, S. R. Bulo, and P. Kontschieder, "The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes," in *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2017-October, pp. 5000–5009, Institute of Electrical and Electronics Engineers Inc., dec 2017.

[16] Z. Che, G. Li, T. Li, B. Jiang, X. Shi, X. Zhang, Y. Lu, G. Wu, Y. Liu, and J. Ye, "D^2-city: A large-scale dashcam video dataset of diverse traffic scenarios," 2019.

[17] H. Yu, Z. Yang, L. Tan, Y. Wang, W. Sun, M. Sun, and Y. Tang, "Methods and datasets on semantic segmentation: A review," *Neurocomputing*, vol. 304, pp. 82–103, aug 2018.

[18] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research (IJRR)*, 2013.

[19] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[20] J. Geyer, Y. Kassahun, M. Mahmudi, X. Ricou, R. Durgesh, A. S. Chung, L. Hauswald, V. H. Pham, M. Mühlegg, S. Dorn, T. Fernandez, M. Jänicke, S. Mirashi, C. Savani, M. Sturm, O. Vorobiov, M. Oelker, S. Garreis, and P. Schuberth, "A2d2: Audi autonomous driving dataset," 2020.

[21] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, "SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR Sequences," in *Proc. of the IEEE/CVF International Conf. on Computer Vision (ICCV)*, 2019.

[22] M. Wigness, S. Eum, J. G. Rogers, D. Han, and H. Kwon, "A rugd dataset for autonomous navigation and visual perception in unstructured outdoor environments," in *IROS*, 2019.

[23] A. Valada, G. Oliveira, T. Brox, and W. Burgard, "Deep multispectral semantic scene understanding of forested environments using multimodal fusion," in *ISER*, 2016.

[24] L. Dabbiru, C. Goodin, N. Scherrer, and D. Carruth, "Lidar data segmentation in off-road environment using convolutional neural networks (cnn)," *SAE International Journal of Advances and Current Practices in Mobility*, vol. 2, no. 2020-01-0696, pp. 3288–3292, 2020.

[25] C. Goodin, M. Doude, C. Hudson, and D. Carruth, "Enabling Off-Road Autonomous Navigation-Simulation of LIDAR in Dense Vegetation," *Electronics*, vol. 7, p. 154, aug 2018.

[26] Z. Pezzementi, T. Tabor, P. Hu, J. K. Chang, D. Ramanan, C. Wellington, B. P. Wisely Babu, and H. Herman, "Comparing apples and oranges: Off-road pedestrian detection on the na-

tional robotics engineering center agricultural person-detection dataset," *Journal of Field Robotics*, vol. 35, no. 4, pp. 545–563, 2018.

[27] S. Zhou and K. Iagnemma, "Self-supervised learning method for unstructured road detection using fuzzy support vector machines," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1183–1189, 2010.

[28] H. Kong, J.-Y. Audibert, and J. Ponce, "Vanishing point detection for road detection," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 96–103, 2009.

[29] H. Jeong, Y. Oh, J. H. Park, B. S. Koo, and S. W. Lee, "Vision-based adaptive and recursive tracking of unpaved roads," *Pattern Recogn. Lett.*, vol. 23, p. 73–82, jan 2002.

[30] B. Rothrock, R. Kennedy, C. Cunningham, J. Papon, M. Heverly, and M. Ono, "Spoc: Deep learning-based terrain classification for mars rover missions," in *AIAA SPACE 2016*, p. 5539, AIAA, 2016.

[31] I. Sgibnev, A. Sorokin, B. Vishnyakov, and Y. Vizilter, "Deep Semantic Segmentation for the Off-Road Autonomous Driving," *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 43B2, pp. 617–622, 2020.

[32] Y. Jin, D. Han, and H. Ko, "Memory-based Semantic Segmentation for Off-road Unstructured Natural Environments," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 24–31, 2021.

[33] T. Guan, D. Kothandaraman, R. Chandra, A. J. Sathyamoorthy, K. Weerakoon, and D. Manocha, "GANav: Efficient Terrain Segmentation for Robot Navigation in Unstructured Outdoor Environments."

[34] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun, "Deep Learning for 3D Point Clouds: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, jun 2020.

[35] Y. Pan, B. Gao, J. Mei, S. Geng, C. Li, and H. Zhao, “Semanticposs: A point cloud dataset with large quantity of dynamic instances,” in *2020 IEEE Intelligent Vehicles Symposium (IV)*, pp. 687–693, 2020.

[36] Y. Zhou and O. Tuzel, “VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 4490–4499, nov 2018.

[37] C. R. Qi, H. Su, M. Niebner, A. Dai, M. Yan, and L. J. Guibas, “Volumetric and multi-view CNNs for object classification on 3D data,” in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2016-December, pp. 5648–5656, IEEE Computer Society, dec 2016.

[38] Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao, “3D ShapeNets: A deep representation for volumetric shapes,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 07-12-June-2015, pp. 1912–1920, jun 2015.

[39] B. Graham and L. van der Maaten, “Submanifold sparse convolutional networks,” 2017.

[40] B. Graham, M. Engelcke, and L. Van Der Maaten, “3D Semantic Segmentation with Submanifold Sparse Convolutional Networks,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, pp. 9224–9232, nov 2018.

[41] C. Choy, J. Gwak, and S. Savarese, “4D spatio-temporal convnets: Minkowski convolutional neural networks,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2019-June, pp. 3070–3079, apr 2019.

[42] H. Tang, Z. Liu, S. Zhao, Y. Lin, J. Lin, H. Wang, and S. Han, “Searching efficient 3d architectures with sparse point-voxel convolution,” in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVIII*, pp. 685–702, Springer, 2020.

[43] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, “Pointnet++: Deep hierarchical feature learning on point sets in a metric space,” in *Advances in Neural Information Processing Systems*

(I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[44] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 77–85, Institute of Electrical and Electronics Engineers Inc., nov 2017.

[45] H. Thomas, C. R. Qi, J. E. Deschaud, B. Marcotegui, F. Goulette, and L. Guibas, "KPConv: Flexible and deformable convolution for point clouds," in *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2019-October, pp. 6410–6419, Institute of Electrical and Electronics Engineers Inc., oct 2019.

[46] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen, "PointCNN: Convolution On X-Transformed Points," *arXiv*, pp. 1–11, jan 2018.

[47] A. Milioto, I. Vizzo, J. Behley, and C. Stachniss, "Rangenet++: Fast and accurate lidar semantic segmentation," in *2019 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 4213–4220, IEEE, 2019.

[48] T. Cortinhal, G. Tzelepis, and E. E. Aksoy, "Salsanext: Fast, uncertainty-aware semantic segmentation of lidar point clouds for autonomous driving," 2020.

[49] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. Keutzer, "SqueezeSegV2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a LiDAR point cloud," in *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 2019-May, pp. 4376–4382, Institute of Electrical and Electronics Engineers Inc., may 2019.

[50] J. Hoffman, E. Tzeng, T. Park, J. Y. Zhu, P. Isola, K. Saenko, A. A. Efros, and T. Darrell, "CyCADA: Cycle-Consistent Adversarial Domain adaptation," in *35th International Conference on Machine Learning, ICML 2018*, vol. 5, pp. 3162–3174, nov 2018.

[51] B. Wu, X. Zhou, S. Zhao, X. Yue, and K. Keutzer, "SqueezeSegV2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a LiDAR point cloud," in *Proc. - IEEE Int. Conf. Robot. Autom.*, vol. 2019-May, pp. 4376–4382, Institute of Electrical and Electronics Engineers Inc., sep 2019.

[52] C. B. Rist, M. Enzweiler, and D. M. Gavrila, "Cross-sensor deep domain adaptation for LiDAR detection and segmentation," in *IEEE Intell. Veh. Symp. Proc.*, vol. 2019-June, pp. 1535–1542, Institute of Electrical and Electronics Engineers Inc., jun 2019.

[53] K. Saleh, A. Abobakr, M. Attia, J. Iskander, D. Nahavandi, M. Hossny, and S. Nahvandi, "Domain adaptation for vehicle detection from bird's eye view lidar point cloud data," in *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pp. 3235–3242, 2019.

[54] Z. Wang, S. Ding, Y. Li, M. Zhao, S. Roychowdhury, A. Wallin, G. Sapiro, and Q. Qiu, "Range Adaptation for 3D Object Detection in LiDAR," *Proc. - 2019 Int. Conf. Comput. Vis. Work. ICCVW 2019*, pp. 2320–2328, sep 2019.

[55] C. Qin, H. You, L. Wang, C. C. J. Kuo, and Y. Fu, "PointDAN: A Multi-Scale 3D Domain Adaption Network for Point Cloud Representation," *Adv. Neural Inf. Process. Syst.*, vol. 32, nov 2019.

[56] L. Yi, B. Gong, and T. Funkhouser, "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 (CVPR)*, pp. 15363–15373, June 2021.

[57] S. Zhao, Y. Wang, B. Li, B. Wu, Y. Gao, P. Xu, T. Darrell, and K. Keutzer, "ePointDA: An end-to-end simulation-to-real domain adaptation framework for LiDAR point cloud segmentation," sep 2020.

[58] F. Langer, A. Milioto, A. Haag, J. Behley, and C. Stachniss, "Domain Transfer for Semantic Segmentation of LiDAR Data using Deep Neural Networks," in *2020 IEEE/RSJ Interna-*

tional Conference on Intelligent Robots and Systems (IROS), pp. 8263–8270, IEEE, oct 2020.

[59] M. Jaritz, T.-H. Vu, R. de Charette, E. Wirbel, and P. Perez, “xMUDA: Cross-Modal Unsupervised Domain Adaptation for 3D Semantic Segmentation,” in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12602–12611, IEEE, jun 2020.

[60] R. Weston, S. Cen, P. Newman, and I. Posner, “Probably unknown: Deep inverse sensor modelling radar,” in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 5446–5452, 2019. ISSN: 2577-087X.

[61] J. Lombacher, K. Laudt, M. Hahn, J. Dickmann, and C. Wöhler, “Semantic radar grids,” in *2017 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1170–1175, 2017.

[62] N. Scheiner, N. Appenrodt, J. Dickmann, and B. Sick, “Radar-based feature design and multiclass classification for road user recognition,” in *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 779–786, 2018.

[63] N. Scheiner, N. Appenrodt, J. Dickmann, and B. Sick, “Radar-based road user classification and novelty detection with recurrent neural network ensembles,” in *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 722–729, 2019.

[64] O. Schumann, M. Hahn, J. Dickmann, and C. Wöhler, “Semantic segmentation on radar point clouds,” in *2018 21st International Conference on Information Fusion (FUSION)*, pp. 2179–2186, 2018.

[65] P. Kaul, D. de Martini, M. Gadd, and P. Newman, “RSS-net: Weakly-supervised multi-class semantic segmentation with FMCW radar,” in *2020 IEEE Intelligent Vehicles Symposium (IV)*, pp. 431–436, 2020. ISSN: 2642-7214.

[66] Q.-H. Pham, M. A. Uy, B.-S. Hua, D. T. Nguyen, G. Roig, and S.-K. Yeung, “LCD: Learned Cross-Domain Descriptors for 2D-3D Matching,” *arXiv*, 2019.

[67] B. Wang, C. Chen, Z. Cui, J. Qin, C. X. Lu, Z. Yu, P. Zhao, Z. Dong, F. Zhu, N. Trigoni, and A. Markham, "P2-Net: Joint Description and Detection of Local Features for Pixel and Point Matching," *arXiv*, 2021.

[68] M. Feng, S. Hu, M. H. Ang, and G. H. Lee, "2D3D-Matchnet: Learning To Match Keypoints Across 2D Image And 3D Point Cloud," *2019 International Conference on Robotics and Automation (ICRA)*, vol. 00, pp. 4790–4796, 2019.

[69] X. Wang, R. Zhang, C. Shen, T. Kong, and L. Li, "Dense Contrastive Learning for Self-Supervised Visual Pre-Training," *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 00, pp. 3023–3032, 2021. Unlike other unsupervised learning methods, DenseCL implements self-supervised learning by optimizing a pairwise contrastive (dis)similarity loss at the pixel level between two views of input images.

[70] C. Choy, J. Park, and V. Koltun, "Fully convolutional geometric features," in *ICCV*, 2019.

[71] Y. Li, H. Su, C. R. Qi, N. Fish, D. Cohen-Or, and L. J. Guibas, "Joint embeddings of shapes and images via CNN image purification," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 6, pp. 1–12, 2015.

[72] D. Cattaneo, M. Vaghi, S. Fontana, A. L. Ballardini, and D. G. Sorrenti, "Global visual localization in LiDAR-maps through shared 2D-3D embedding space," *2020 IEEE International Conference on Robotics and Automation (ICRA)*, vol. 00, pp. 4365–4371, 2020.

[73] P. Yin, L. Xu, J. Zhang, H. Choset, and S. Scherer, "i3dLoc: Image-to-range Cross-domain Localization Robust to Inconsistent Environmental Conditions," *arXiv*, 2021.

[74] X. Xing, Y. Cai, T. Lu, S. Cai, Y. Yang, and D. Wen, "3DTNet: Learning Local Features using 2D and 3D Cues," *2018 International Conference on 3D Vision (3DV)*, pp. 435–443, 2018.

[75] S. Wasielewski and O. Strauss, "Calibration of a multi-sensor system laser rangefinder/camera," in *Intelligent Vehicles Symposium, Proceedings*, pp. 472–477, IEEE, 1995.

[76] R. Gomez-Ojeda, J. Briales, E. Fernandez-Moral, and J. Gonzalez-Jimenez, “Extrinsic calibration of a 2d laser-rangefinder and a camera based on scene corners,” in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3611–3616, 2015.

[77] J. Kang and N. L. Doh, “Automatic targetless camera–LIDAR calibration by aligning edge with Gaussian mixture model,” *Journal of Field Robotics*, vol. 37, pp. 158–179, jan 2020.

[78] G. Iyer, R. K. Ram, J. K. Murthy, and K. M. Krishna, “CalibNet: Geometrically Supervised Extrinsic Calibration using 3D Spatial Transformer Networks,” in *IEEE International Conference on Intelligent Robots and Systems*, pp. 1110–1117, Institute of Electrical and Electronics Engineers Inc., dec 2018.

[79] S. Mishra, G. Pandey, and S. Saripalli, “Extrinsic Calibration of a 3D-LIDAR and a Camera,” in *IEEE Intelligent Vehicles Symposium, Proceedings*, pp. 1765–1770, Institute of Electrical and Electronics Engineers Inc., 2020.

[80] S. Mishra, P. R. Osteen, G. Pandey, and S. Saripalli, “Experimental evaluation of 3D-LIDAR camera extrinsic calibration,” in *IEEE International Conference on Intelligent Robots and Systems*, pp. 9020–9026, Institute of Electrical and Electronics Engineers Inc., oct 2020.

[81] Z. Pusztai and L. Hajder, “Accurate calibration of lidar-camera systems using ordinary boxes,” in *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2017.

[82] J. Levinson and S. Thrun, “Automatic online calibration of cameras and lasers,” in *Proceedings of Robotics: Science and Systems*, (Berlin, Germany), June 2013.

[83] X. Lv, B. Wang, D. Ye, and S. Wang, “Lccnet: Lidar and camera self-calibration using cost volume network,” 2021.

[84] N. Schneider, F. Piewak, C. Stiller, and U. Franke, “RegNet: Multimodal sensor registration using deep neural networks,” in *IEEE Intelligent Vehicles Symposium, Proceedings*, pp. 1803–1810, IEEE, jun 2017.

[85] G. Zhao, J. Hu, S. You, and C. C. J. Kuo, "Calibdnn: Multimodal sensor calibration for perception using deep neural networks," 2021.

[86] K. Yuan, Z. Guo, and Z. Jane Wang, "RGGNet: Tolerance Aware LiDAR-Camera Online Calibration with Geometric Deep Learning and Generative Model," *IEEE Robotics and Automation Letters*, vol. 5, pp. 6956–6963, oct 2020.

[87] C. Tan, F. Sun, T. Kong, W. Zhang, C. Yang, and C. Liu, "A survey on deep transfer learning," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11141 LNCS, pp. 270–279, Springer Verlag, oct 2018.

[88] Z. Taylor, J. Nieto, and D. Johnson, "Multi-Modal Sensor Calibration Using a Gradient Orientation Measure," *Journal of Field Robotics*, vol. 32, pp. 675–695, aug 2015.

[89] B. Nagy, L. Kovacs, and C. Benedek, "SFM and Semantic Information Based Online Targetless Camera-LIDAR Self-Calibration," in *Proceedings - International Conference on Image Processing, ICIP*, vol. 2019-Septe, pp. 1317–1321, IEEE Computer Society, sep 2019.

[90] Y. Zhu, C. Li, and Y. Zhang, "Online Camera-LiDAR Calibration with Sensor Semantic Information," in *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 4970–4976, Institute of Electrical and Electronics Engineers Inc., may 2020.

[91] W. Wang, S. Nobuhara, R. Nakamura, and K. Sakurada, "Soic: Semantic online initialization and calibration for lidar and camera," mar 2020.

[92] G. El Natour, O. Ait Aider, R. Rouveure, F. Berry, and P. Faure, "Radar and vision sensors calibration for outdoor 3d reconstruction," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2084–2089, 2015. ISSN: 1050-4729.

[93] J. Peršić, I. Marković, and I. Petrović, "Extrinsic 6dof calibration of 3d LiDAR and radar," in *2017 European Conference on Mobile Robots (ECMR)*, pp. 1–6, 2017.

[94] J. Peršić, I. Marković, and I. Petrović, "Extrinsic 6dof calibration of a radar–LiDAR–camera system enhanced by radar cross section estimates evaluation," *Robotics and Autonomous Systems*, vol. 114, pp. 217–230, 2019.

[95] J. Domhof, J. F. Kooij, and D. M. Gavrila, "An extrinsic calibration tool for radar, camera and lidar," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 8107–8113, 2019. ISSN: 2577-087X.

[96] K. Burnett, "radar_to_lidar_calib." original-date: 2020-10-23T17:41:38Z.

[97] B. Reddy and B. Chatterji, "An FFT-based technique for translation, rotation, and scale-invariant image registration," *IEEE Transactions on Image Processing*, vol. 5, no. 8, pp. 1266–1271, 1996. Conference Name: IEEE Transactions on Image Processing.

[98] G. Yan, L. Zhuochun, C. Wang, C. Shi, P. Wei, X. Cai, T. Ma, Z. Liu, Z. Zhong, Y. Liu, M. Zhao, Z. Ma, and Y. Li, "OpenCalib: A multi-sensor calibration toolbox for autonomous driving."

[99] D. Maturana and S. Scherer, "VoxNet: A 3D Convolutional Neural Network for real-time object recognition," in *IEEE Int. Conf. Intell. Robot. Syst.*, vol. 2015-Decem, pp. 922–928, Institute of Electrical and Electronics Engineers Inc., dec 2015.

[100] D. Feng, C. Haase-Schutz, L. Rosenbaum, H. Hertlein, C. Glaser, F. Timm, W. Wiesbeck, and K. Dietmayer, "Deep Multi-Modal Object Detection and Semantic Segmentation for Autonomous Driving: Datasets, Methods, and Challenges," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–20, feb 2020.

[101] M. Everingham, S. M. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes Challenge: A Retrospective," *International Journal of Computer Vision*, vol. 111, pp. 98–136, jun 2014.

[102] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, W. Liu, and B. Xiao, "Deep high-resolution representation learning for visual recognition," *TPAMI*, 2019.

[103] Y. Yuan, X. Chen, X. Chen, and J. Wang, "Segmentation transformer: Object-contextual representations for semantic segmentation," 2021.

[104] T. Takikawa, D. Acuna, V. Jampani, and S. Fidler, "Gated-SCNN: Gated shape CNNs for semantic segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, vol. 2019-October, pp. 5228–5237, Institute of Electrical and Electronics Engineers Inc., oct 2019.

[105] C. Yu, C. Gao, J. Wang, G. Yu, C. Shen, and N. Sang, "Bisenet v2: Bilateral network with guided aggregation for real-time semantic segmentation," 2020.

[106] Y. Yuan, X. Chen, and J. Wang, "Object-contextual representations for semantic segmentation," 2020.

[107] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," 2019.

[108] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR 2009*, 2009.

[109] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik, "Learning rich features from rgb-d images for object detection and segmentation," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VII 13*, pp. 345–360, Springer, 2014.

[110] M. Bertalmío, A. L. Bertozzi, and G. Sapiro, "Navier-Stokes, fluid dynamics, and image and video inpainting," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, 2001.

[111] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," 2020.

[112] P. Luc, C. Couprie, S. Chintala, and J. Verbeek, "Semantic segmentation using adversarial networks," 2016.

[113] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan, "Domain separation networks," in *Advances in Neural Information Processing Systems*, pp. 343–351, 2016.

[114] W. C. Hung, Y. H. Tsai, Y. T. Liou, Y. Y. Lin, and M. H. Yang, "Adversarial learning for semi-supervised semantic segmentation," *British Machine Vision Conference 2018, BMVC 2018*, feb 2019.

[115] D. Barnes, M. Gadd, P. Murcutt, P. Newman, and I. Posner, "The oxford radar robotcar dataset: A radar extension to the oxford robotcar dataset," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6433–6438, 2020.

[116] H. Zhou, X. Zhu, X. Song, Y. Ma, Z. Wang, H. Li, and D. Lin, "Cylinder3D: An Effective 3D Framework for Driving-scene LiDAR Semantic Segmentation."

[117] W. Xu and F. Zhang, "FAST-LIO: A Fast, Robust LiDAR-Inertial Odometry Package by Tightly-Coupled Iterated Kalman Filter," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3317–3324, 2021.

[118] W. Xu, Y. Cai, D. He, J. Lin, and F. Zhang, "FAST-LIO2: Fast Direct LiDAR-Inertial Odometry," *IEEE Transactions on Robotics*, pp. 1–21, 2022.

[119] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking Atrous Convolution for Semantic Image Segmentation."

[120] Y. Sun, C. Cheng, Y. Zhang, C. Zhang, L. Zheng, Z. Wang, and Y. Wei, "Circle Loss: A Unified Perspective of Pair Similarity Optimization," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 00, pp. 6397–6406, 2020. Circle loss provide a unified perspective for optimize pair similarity.

[121] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a Similarity Metric Discriminatively, with Application to Face Verification," *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, pp. 539–546, 2005.

[122] K. Sohn, "Improved deep metric learning with multi-class n-pair loss objective," in *Advances in Neural Information Processing Systems* (D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, eds.), vol. 29, Curran Associates, Inc., 2016.

[123] A. van den Oord, Y. Li, and O. Vinyals, "Representation Learning with Contrastive Predictive Coding," *arXiv e-prints*, p. arXiv:1807.03748, July 2018.

[124] Y. Liu, Q. Fan, S. Zhang, H. Dong, T. Funkhouser, and L. Yi, "Contrastive multimodal fusion with tupleinfonce," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 754–763, October 2021.

[125] K. Bousmalis, G. Trigeorgis, N. Silberman, D. Krishnan, and D. Erhan, "Domain separation networks," in *Advances in Neural Information Processing Systems* (D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, eds.), vol. 29, Curran Associates, Inc., 2016.

[126] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[127] J. Zhang, X. Zhao, Z. Chen, and Z. Lu, "A review of deep learning-based semantic segmentation for point cloud," *IEEE Access*, vol. 7, pp. 179118–179133, 2019.

[128] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[129] M. M. Bronstein, J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst, "Geometric deep learning: Going beyond euclidean data," *IEEE Signal Processing Magazine*, vol. 34, no. 4, pp. 18–42, 2017.

[130] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 764–773, 2017.

[131] Y. Liao, J. Xie, and A. Geiger, "KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d," *arXiv preprint arXiv:2109.13410*, 2021.

[132] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *Advances in neural information processing systems*, vol. 32, 2019.

[133] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, OpenReview.net, 2019.

[134] M. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (Y. W. Teh and M. Titterington, eds.), vol. 9 of *Proceedings of Machine Learning Research*, (Chia Laguna Resort, Sardinia, Italy), pp. 297–304, PMLR, 13–15 May 2010.

[135] A. Banerjee, I. S. Dhillon, J. Ghosh, and S. Sra, "Clustering on the unit hypersphere using von mises-fisher distributions," *Journal of Machine Learning Research*, vol. 6, no. 46, pp. 1345–1382, 2005.

[136] R. Geirhos, P. Rubisch, C. Michaelis, M. Bethge, F. A. Wichmann, and W. Brendel, "Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness," *CoRR*, vol. abs/1811.12231, 2018.

[137] C. Leng, H. Zhang, B. Li, G. Cai, Z. Pei, and L. He, "Local feature descriptor for image matching: A survey," *IEEE Access*, vol. 7, pp. 6424–6434, 2019.

[138] C. Wachinger and N. Navab, "Simultaneous registration of multiple images: Similarity metrics and efficient optimization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 5, pp. 1221–1233, 2013.

[139] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, "Spatial Transformer Networks," *Advances in Neural Information Processing Systems*, vol. 2015-January, pp. 2017–2025, jun 2015.

[140] M. I. Belghazi, A. Baratin, S. Rajeswar, S. Ozair, Y. Bengio, A. Courville, and R. D. Hjelm, “Mutual information neural estimation,” in *35th International Conference on Machine Learning, ICML 2018*, vol. 2, pp. 864–873, jan 2018.

[141] S. Mukherjee, H. Asnani, and S. Kannan, “CCMI : Classifier based conditional mutual information estimation,” 2019.

[142] A. K. Mondal, A. P. Prathosh, A. Bhattacharjee, S. Kannan, S. Mukherjee, and H. Asnani, “C-MI-GAN: Estimation of conditional mutual information using minmax formulation,” in *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence, UAI 2020*, pp. 869–878, 2020.

[143] V. Lepetit, F. Moreno-Noguer, and P. Fua, “EPnP: An accurate O(n) solution to the PnP problem,” *International Journal of Computer Vision*, vol. 81, pp. 155–166, feb 2009.

[144] B. Iglewicz and D. Hoaglin, “The asqc basic references in quality control: statistical techniques,” *How to detect and handle outliers*, vol. 16, pp. 1–87, 1993.

[145] J. Xie, M. Kiefel, M.-T. Sun, and A. Geiger, “Semantic instance annotation of street scenes by 3d to 2d label transfer,” in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[146] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, “CARLA: An open urban driving simulator,” in *Proceedings of the 1st Annual Conference on Robot Learning*, pp. 1–16, 2017.

[147] G. Pandey, J. R. McBride, S. Savarese, and R. M. Eustice, “Automatic Extrinsic Calibration of Vision and Lidar by Maximizing Mutual Information,” *Journal of Field Robotics*, vol. 32, pp. 696–722, aug 2015.

[148] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “PyTorch: An imperative style, high-performance deep learning library,” 2019.

[149] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[150] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, International Conference on Learning Representations, ICLR, dec 2015.

[151] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (FPFH) for 3d registration," in *2009 IEEE International Conference on Robotics and Automation*, pp. 3212–3217, 2009. ISSN: 1050-4729.

[152] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis." _eprint: 2003.08934.

[153] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An imperative style, high-performance deep learning library."

[154] M. E. Atik and Z. Duran, "An Efficient Ensemble Deep Learning Approach for Semantic Point Cloud Segmentation Based on 3D Geometric Features and Range Images," *Sensors*, vol. 22, no. 16, p. 6210, 2022.

[155] X. Cai, M. Everett, J. Fink, and J. P. How, "Risk-aware off-road navigation via a learned speed distribution map," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2931–2937, IEEE, 2022.

[156] T. Guan, Z. He, R. Song, D. Manocha, and L. Zhang, "Tns: Terrain traversability mapping and navigation system for autonomous excavators," 2022.

[157] F. Islam, M. M. Nabi, and J. E. Ball, "Off-road detection analysis for autonomous ground vehicles: A review," *Sensors*, vol. 22, no. 21, p. 8463, 2022.

[158] H. Liu, M. Yao, X. Xiao, and H. Cui, "A hybrid attention semantic segmentation network for unstructured terrain on Mars," *Acta Astronautica*, 2022.

[159] A. J. Sathyamoorthy, K. Weerakoon, T. Guan, J. Liang, and D. Manocha, "TerraPN: Unstructured Terrain Navigation using Online Self-Supervised Learning," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7197–7204, IEEE, 2022.

[160] S. Sharma, L. Dabbiru, T. Hannis, G. Mason, D. W. Carruth, M. Doude, C. Goodin, C. Hudson, S. Ozier, and J. E. Ball, "Cat: Cavs traversability dataset for off-road autonomous driving," *IEEE Access*, vol. 10, pp. 24759–24768, 2022.

[161] H. Wei, E. Xu, J. Zhang, Y. Meng, J. Wei, Z. Dong, and Z. Li, "BushNet: Effective semantic segmentation of bush in large-scale point clouds," *Computers and Electronics in Agriculture*, vol. 193, p. 106653, 2022.

[162] X. Xiao, B. Liu, G. Warnell, and P. Stone, "Motion planning and control for mobile robot navigation using machine learning: A survey," *Autonomous Robots*, vol. 46, no. 5, pp. 569–597, 2022.

[163] B. Bešić, N. Gosala, D. Cattaneo, and A. Valada, "Unsupervised domain adaptation for lidar panoptic segmentation," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3404–3411, 2022.

[164] A. H. Gebrehiwot, P. Vacek, D. Hurych, K. Zimmermann, P. Pérez, and T. Svoboda, "Teachers in concordance for pseudo-labeling of 3D sequential data," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 536–543, 2022.

[165] L. Kong, N. Quader, V. E. Liong, and H. Zhang, "Conda: Unsupervised domain adaptation for lidar segmentation via regularized domain concatenation," 2022.

[166] L. Nunes, R. Marcuzzi, X. Chen, J. Behley, and C. Stachniss, "SegContrast: 3D point cloud feature representation learning through self-supervised segment discrimination," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2116–2123, 2022.

[167] Y. Wang, J. Yin, W. Li, P. Frossard, R. Yang, and J. Shen, "Ssda3d: Semi-supervised domain adaptation for 3d object detection from point cloud," 2022.

[168] Z. Yuan, C. Wen, M. Cheng, Y. Su, W. Liu, S. Yu, and C. Wang, "Category-Level Adversaries for Outdoor LiDAR Point Clouds Cross-Domain Semantic Segmentation," *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[169] Z. Yuan, M. Cheng, W. Zeng, Y. Su, W. Liu, S. Yu, and C. Wang, "Prototype-guided Multi-task Adversarial Network for Cross-domain LiDAR Point Clouds Semantic Segmentation," *IEEE Transactions on Geoscience and Remote Sensing*, 2023.

[170] A. Kirillov, K. He, R. Girshick, C. Rother, and P. Dollár, "Panoptic Segmentation."

[171] B. Cheng, A. G. Schwing, and A. Kirillov, "Per-Pixel Classification is Not All You Need for Semantic Segmentation."

[172] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar, "Masked-attention Mask Transformer for Universal Image Segmentation."

[173] M. Chen, J. Tworek, H. Jun, Q. Yuan, H. P. d. O. Pinto, J. Kaplan, H. Edwards, Y. Burda, N. Joseph, G. Brockman, A. Ray, R. Puri, G. Krueger, M. Petrov, H. Khlaaf, G. Sastry, P. Mishkin, B. Chan, S. Gray, N. Ryder, M. Pavlov, A. Power, L. Kaiser, M. Bavarian, C. Winter, P. Tillet, F. P. Such, D. Cummings, M. Plappert, F. Chantzis, E. Barnes, A. Herbert-Voss, W. H. Guss, A. Nichol, A. Paino, N. Tezak, J. Tang, I. Babuschkin, S. Balaji, S. Jain, W. Saunders, C. Hesse, A. N. Carr, J. Leike, J. Achiam, V. Misra, E. Morikawa, A. Radford, M. Knight, M. Brundage, M. Murati, K. Mayer, P. Welinder, B. McGrew, D. Amodei, S. McCandlish, I. Sutskever, and W. Zaremba, "Evaluating Large Language Models Trained on Code."

[174] "Reflections on Foundation Models."

[175] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning Transferable Visual Models From Natural Language Supervision."

[176] J. Xu, J. Hou, Y. Zhang, R. Feng, Y. Wang, Y. Qiao, and W. Xie, "Learning open-vocabulary semantic segmentation models from natural language supervision," 2023.

[177] F. Liang, B. Wu, X. Dai, K. Li, Y. Zhao, H. Zhang, P. Zhang, P. Vajda, and D. Marculescu, "Open-Vocabulary Semantic Segmentation with Mask-adapted CLIP."

[178] H. Luo, J. Bao, Y. Wu, X. He, and T. Li, "Segclip: Patch aggregation with learnable centers for open-vocabulary semantic segmentation," 2022.

[179] M. Xu, Z. Zhang, F. Wei, Y. Lin, Y. Cao, H. Hu, and X. Bai, "A Simple Baseline for Open-Vocabulary Semantic Segmentation with Pre-trained Vision-Language Model," in *Computer Vision – ECCV 2022* (S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, eds.), Lecture Notes in Computer Science, pp. 736–753, Springer Nature Switzerland, 2022.

[180] D. Hoeller, N. Rudin, C. Choy, A. Anandkumar, and M. Hutter, "Neural Scene Representation for Locomotion on Structured Terrain."

[181] C. Jiang, A. Sud, A. Makadia, J. Huang, M. NieBner, and T. Funkhouser, "Local Implicit Grid Representations for 3D Scenes," in *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6000–6009, IEEE, 2020.

[182] S. Lionar, L. Schmid, C. Cadena, R. Siegwart, and A. Cramariuc, "Neuralblox: Real-time neural representation fusion for robust volumetric mapping," in *2021 International Conference on 3D Vision (3DV)*, pp. 1279–1289, 2021.

[183] H. Oleynikova, A. Millane, Z. Taylor, E. Galceran, J. Nieto, and R. Siegwart, "Signed distance fields: A natural representation for both mapping and planning," in *RSS 2016 Workshop: Geometry and Beyond-Representations, Physics, and Scene Understanding for Robotics*, University of Michigan, 2016.

[184] Y. Pan, Y. Kompis, L. Bartolomei, R. Mascaro, C. Stachniss, and M. Chli, "Voxfield: Non-projective signed distance fields for online planning and 3d reconstruction," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5331–5338, 2022.

[185] E. Sucar, S. Liu, J. Ortiz, and A. J. Davison, “imap: Implicit mapping and positioning in real-time,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6229–6238, 2021.

[186] X. Zhong, Y. Pan, J. Behley, and C. Stachniss, “SHINE-Mapping: Large-Scale 3D Mapping Using Sparse Hierarchical Implicit Neural Representations.”

[187] M. Adamkiewicz, T. Chen, A. Caccavale, R. Gardner, P. Culbertson, J. Bohg, and M. Schwager, “Vision-Only Robot Navigation in a Neural Radiance World.”

[188] J. Sun, Y. Xu, M. Ding, H. Yi, J. Wang, L. Zhang, and M. Schwager, “NeRF-Loc: Transformer-Based Object Localization Within Neural Radiance Fields.”

[189] C. Wang, M. Chai, M. He, D. Chen, and J. Liao, “CLIP-NeRF: Text-and-Image Driven Manipulation of Neural Radiance Fields.”

[190] F. Dellaert and L. Yen-Chen, “Neural volume rendering: Nerf and beyond,” 2021.

[191] C.-H. Lin, W.-C. Ma, A. Torralba, and S. Lucey, “Barf: Bundle-adjusting neural radiance fields,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5741–5751, 2021.