VISUAL-LIDAR SIMULTANEOUS LOCALIZATION AND MAPPING

SEYEDMOSTAFA AHMADI

A THESIS SUBMITTED TO THE FACULTY OF GRADUATE STUDIES IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF MASTER OF SCIENCE

GRADUATE PROGRAM IN EARTH AND SPACE SCIENCE

YORK UNIVERSITY

TORONTO, ONTARIO

October 2023

Abstract

The field of Simultaneous localization and mapping (SLAM) has garnered significant attention in robotics research over the years. While SLAM has demonstrated success, its application in mobile mapping systems presents unique challenges. This study builds upon prior research (mostly [1]), extending their framework to enhance accuracy and perform boundary tests on a specific mobile mapping system. Our contribution introduces a novel SLAM approach termed HDPV-SLAM, addressing critical limitations encountered by existing systems. The first challenge addressed is the sparsity of LiDAR depth data, complicating its correlation with extracted visual features from RGB images. To tackle this, we propose a deep learning-based depth estimation module that iteratively densifies the sparse LiDAR depth information. The second challenge stems from the lack of horizontal overlap between the panoramic camera and the tilted LiDAR sensor, causing difficulties in depth association. To overcome this obstacle, I introduce a hybrid depth association module that optimally combines depth information from independent procedures: feature-based triangulation and depth estimation. This hybrid approach maximizes the utilization of accurate depth data during feature tracking, thereby enhancing overall mapping accuracy. Furthermore, a comprehensive dataset, named YUTO MMS, is presented to the public. This dataset spans 18.95 km and was collected from diverse environments including York University's Keel campus and Teledyne Optech headquarters building.

Acknowledgement

I would like to express my heartfelt gratitude to the following individuals and institutions who have been instrumental in my journey over the past two years in completing this thesis:

First and foremost, I extend my deepest appreciation to my supervisor, Prof. Gunho Sohn, for their unwavering guidance, mentorship, and encouragement throughout this research endeavor. Your expertise and insight have been invaluable in shaping the direction of my work.

I extend my sincere thanks to my esteemed teammates in the SLAM team: Zahra Arjmandi, Dr. Yujia Zhang, and Dr. Jungwon Kang. Your collaboration, discussions, and camaraderie have been pivotal in overcoming challenges and achieving milestones.

To the dearest friends I have made in this lab, Mohammadjavad Ghorbanalivakili and Mohammad Moein Sheikholeslami, your support and companionship have made this academic journey even more rewarding.

I would like to acknowledge the contributions of my labmates, Dr. Amin Alizadeh Naeini, Mohammad Kooshafar, Ali Hassan, Dr. Yeionjeong Jeong, Maryam Jameela, Sunghwan Yoo, Hyungju Lee, Afnan Ahmad, Dr. Connie Ko, Muhammad Kamran, and Zhen Liu. Your diverse perspectives, discussions, and shared experiences have enriched my understanding and enriched the overall lab environment.

My heartfelt gratitude goes out to my family—my father, mother, and sister—for their unending love, support, and understanding. Though separated by distance, your belief in

me has been a constant source of strength.

I am also indebted to my dear friends who have stood by me, offering their encouragement and companionship during this academic journey.

Furthermore, I am thankful for the financial support provided by the Natural Sciences and Engineering Research Council of Canada (NSERC) and York University. Your assistance has enabled me to focus on my research and personal growth without the added burden of financial constraints.

To all those who have touched my life in meaningful ways, whether through guidance, support, or friendship, your presence has made a lasting impact on my academic and personal growth. I am honored to have been surrounded by such a remarkable community.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

ATE	Absolute Trajectory Error
ATV	All-Terrain Vehicle
BRIEF	Binary Robust Independent Elementary Features
DL	Deep Learning
DOF	Degrees of Freedom
EKF	Extended Kalman Filter
FAST	Features from accelerated segment test
FPS	Frames Per Second
GCP	Ground Control Point
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
HDPV-SLAM	Hybrid Depth-augmented Panoramic VS
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
IP Rating	Ingress Protection Rating
LiDAR	Light Detection and Ranging

LMS	LiDAR Mapping Suite
MCL	Monte Carlo Localization
MMS	Mobile Mapping System
ORB	Oriented FAST and rotated BRIEF
PSO	Particle Swarm Optimizer
RMSE	Root mean squared error
RPS	Rotations Per Second
RPV-SLAM	Range-augmented Panoramic VS
RRE	Relative Rotation Error
RTE	Relative Trajectory Error
RTK	Real Time Kinematics
SLAM	Simultaneous localization and mapping
SOTA	state-of-the-art
UKF	Unscented Kalman Filter
UTV	Utility Task Vehicle
VO	Visual Odometry
VS	Visual SLAM
YUTO	York University Teledyne Optech

Chapter 1

Introduction

1.1 Background and Motivation

Simultaneous localization and mapping (SLAM) plays a vital role in robotics and autonomous systems, enabling autonomous navigation and accurate mapping of unknown environments. Its objective is to estimate position and orientation (localization) while constructing a consistent and accurate environment representation (mapping). However, challenges arise from sensor uncertainties, environmental variations, and dynamic real-world scenarios. Robust SLAM algorithms are crucial for reliable and precise estimates, ensuring practical usability and performance in applications such as autonomous vehicles and planetary rovers.

Visual SLAM (VS) has emerged as a prominent research field, offering real-time mapping and localization solutions for a wide range of applications, including robotics, augmented reality, etc. By harnessing visual data from cameras, VS algorithms enable the

creation of precise maps of unknown environments while simultaneously estimating the robot's pose. Nevertheless, ensuring the robustness of VS in dynamic environments and the environments low in feature remains a significant challenge. To tackle this challenge, recent research has been dedicated to the integration of multiple sensors, such as Inertial Measurement Unit (IMU), Global Navigation Satellite System (GNSS), depth sensor, and Light Detection and Ranging (LiDAR), to augment the reliability and accuracy of VS systems. Through the fusion of data from diverse sensor modalities, these advancements aim to surmount the limitations of purely visual-based approaches, empowering VS to achieve more robust localization and mapping performance even in challenging scenarios.

Deep Learning (DL) has made remarkable strides in various research works, showcasing significant advancements. However, when it comes to SLAM, DL-based methods using end-to-end learning strategies have yet to surpass the performance of conventional SLAM methods [2]. This is evident in well-known benchmarks such as KITTI [3], where conventional SLAM methods consistently outperform DL-based approaches by a substantial margin. Despite the capability of DL to automatically learn complex patterns and extract features, traditional SLAM methods, which rely on established algorithms and sensor fusion techniques, exhibit superior performance and robustness in challenging real-world scenarios.

On the other hand, hybrid methods that combine conventional geometric methods with DL techniques have garnered increased attention in recent research [2]. While conventional SLAM systems possess mature geometric theories, they also exhibit limitations

in areas such as poor illumination robustness, poor motion robustness, difficulties in handling dynamic scenes, and insufficient noise tolerance [4]. DL-based methods, on the other hand, have demonstrated capabilities in addressing these challenges. Leveraging the strengths of DL in handling these specific problems, researchers have been exploring the integration of DL modules within the geometric framework of conventional SLAM systems. By replacing certain components with DL modules, the robustness of the SLAM system may be enhanced. This hybrid approach has shown promising results on different datasets, outperforming their parent conventional methods and achieving improved performance. The combination of the geometric expertise of conventional SLAM with the mentioned problem-solving capabilities of DL offers a promising direction for advancing the robustness and performance of SLAM systems in challenging environments.

1.1.1 Research Questions and Knowledge Gaps

Recently, vehicle-mounted Mobile Mapping System (MMS) has emerged as the primary spatial imaging system for capturing high-resolution maps of urban environments. Most commercially-available MMSs combine georeferencing technology with precise, high-speed, long-range laser scanning and high-resolution imaging sensors. In [5], a comprehensive analysis of contemporary MMSs was presented. With the most recent MMS technology, it is now possible to collect enormous quantities of precise, location-based data for various purposes. This includes creating detailed maps suitable for autonomous vehicles and the use of these maps to generate large-scale 3D representations of entire cities [6]. The MMS in our interest is a system equipped with a single multi-beam LiDAR

and a panoramic camera. Our MMS is designed to scan its surrounding environments using the LiDAR tilted towards the ground. The tilted LiDAR offers several advantages in the context of SLAM applications. By virtue of its tilt, the LiDAR sensor can effectively detect and capture features at greater heights surrounding the road, including traffic signs and lights, light poles, highway billboards, and tall buildings. These tall features are predominantly static in nature, which proves highly beneficial for SLAM algorithms as their tracking can provide valuable information for accurate localization and mapping. Conversely, in lower heights, where dynamic objects such as cars, buses, pedestrians, and bicycles are prevalent, the SLAM performance may be compromised. Nevertheless, the ability of the tilted LiDAR to detect and leverage the static tall features contributes significantly to the overall efficacy of SLAM systems, enhancing their robustness and accuracy in urban environments.

The georeferencing capability of MMS significantly relies on advanced GNSS/IMU technology, since the primary purposes of spatial imaging sensors such as cameras and LiDAR are to generate photorealistic textures and metrically-scaled depths. However, this approach introduces several significant challenges:

- Firstly, the reliance on high-end GPS/IMU components proves financially burdensome due to their exorbitant cost.

- Secondly, GPS signals become inaccessible in GPS-denied environments, which is at odds with the overarching objective of MMS to map diverse locations, including tunnels, mines, and urban canyons.

- Thirdly, the integration of GPS/IMU typically necessitates post-processing for fine-grained positioning, a labor-intensive task that can span multiple days under human supervision.

In addition, high demand has been observed for compact and inexpensive MMS designs to reduce system complexity, including using GNSS/IMU less frequently or not at all [5].

These challenges have spurred our pursuit of an alternative solution—leveraging existing camera and LiDAR sensors to execute SLAM. This approach offers several advantages, including cost-efficiency, the ability to operate in GPS-denied contexts, and the capacity to generate near real-time results. While immediate real-time processing is not our primary focus, since our algorithm runs post-data collection, achieving a processing speed superior to that of a human operator suffices for our objectives.

However, the mentioned tilted configuration of the LiDAR sensor introduces unique challenges that affect visual-LiDAR SLAM methodologies. Coupled visual-LiDAR SLAM systems face obstacles as the overlapping region between the camera and the tilted LiDAR sensor does not conform to a rectangular shape primarily covering the road ahead, resulting in a restricted and challenging overlapping region. Furthermore, the topmost LiDAR beams often do not hit objects in the scene due to the absence of tall objects or buildings, causing sparse point distributions in the topmost portion of the LiDAR data.

At the same time, the ability to achieve metrically-scaled results holds significant importance for MMSs. Traditional panoramic VS techniques utilizing monocular vision have demonstrated the capacity to produce up-to-scale outcomes. Existing approaches incorporate Global Positioning System (GPS) data [7] and Ground Control Point (GCP) [8]

to achieve metric scaling. However, our focus lies outside the realm of GPS data utilization, and employing GCPs proves impractical in real-world applications due to their cumbersome nature. Alternatively, recent studies [1, 9, 10] have explored the utilization of LiDAR points in conjunction with a typical pinhole camera to achieve metric scaling. However, attaining the optimal result in this context appears to be distant from the current state-of-the-art (SOTA).

1.2 Research Objectives

This investigation encompasses three key objectives. The first objective is to integrate a mapping-capable MMS with a SLAM system. The second objective is to develop a metrically-scaled SLAM system capable of operating in GPS-denied environments without relying on costly IMU sensors. The goal is to achieve accurate localization and mapping for the challenging scenarios where GPS signals may be unavailable or unreliable. The third objective involves enhancing an existing in-house SLAM system by incorporating deep learning techniques while leveraging its conventional structure. By combining deep learning capabilities with established SLAM methodologies, improved performance and robustness can be achieved. To address these objectives, comprehensive tests were conducted to evaluate the precision and performance of affordable visual sensors in a controlled boundary test scenario.

1.2.1 Contributions

As previously stated, this research aims to address the knowledge gap in the literature by presenting a hybrid SLAM system. In summary, the key contributions of this study are as follows:

- We propose a novel DL-based depth estimation module designed to iteratively densify sparse LiDAR depth using panoramic images and LiDAR point clouds. The proposed module aims to enhance the quality of depth estimation specifically on visual features. Notably, our approach utilizes a unique model that has been trained on perspective cameras but is applied to panoramic images.

- I present a novel hybrid depth association module that addresses the challenge posed by the limited spatial overlap between a panoramic camera and a tilted LiDAR sensor. Our proposed module combines depth information derived from triangulating visual features with LiDAR depth data. By leveraging both sources of information, I aim to overcome the limitations arising from the mismatched spatial coverage of the sensors.

- I propose a hybrid SLAM system, Hybrid Depth-augmented Panoramic VS (HDPV-SLAM) [11], that combines the aforementioned DL-based depth estimation module and the hybrid depth association module. By integrating these modules, our hybrid SLAM system surpasses its parent conventional system in terms of performance and capabilities.

- We introduce the York University Teledyne Optech (YUTO) MMS dataset that

provides a comprehensive collection of data for SLAM and odometry applications, encompassing various sensor modalities such as omnidirectional cameras and tilted LiDAR sensors. The dataset features a unique configuration for a VS system with a tilted LiDAR sensor, enabling the scan of tall features such as traffic signs, lights, and buildings, which are valuable for SLAM and tracking purposes. The dataset includes sequences captured in different environments, including urban areas, residential neighborhoods, and open spaces, allowing for diverse testing and evaluation scenarios. The dataset offers precise ground truth trajectories generated using Real Time Kinematics (RTK) on a Inertial Navigation System (INS) unit, providing accurate pose information for evaluation and benchmarking.

1.3 Thesis Structure

This thesis comprises six chapters that provide a comprehensive exploration of the research topic. Chapter 1 serves as an introduction, presenting the background and motivation behind our study. It also outlines the specific research objectives that this thesis aims to address. Chapter 2 presents a thorough review of the relevant research works in the field of SLAM, with a particular focus on studies related to our research. Additionally, this chapter explores similar datasets to the YUTO MMS dataset, providing a comprehensive overview of the existing resources. Chapter 3 delves into the details of the YUTO MMS dataset, offering a comprehensive description of its composition, sensor modalities, and the environments in which it was captured. This chapter provides readers

with a clear understanding of the dataset and its relevance to our research. Chapter 4 provides an in-depth explanation of HDPV-SLAM system, along with its forming modules. This chapter elucidates the technical aspects of the system, detailing how the modules are integrated to achieve the desired objectives. Chapter 5 focuses on presenting the experimental results obtained from the implementation of the HDPV-SLAM system. It also includes a comprehensive discussion and analysis of the results, allowing for a deeper understanding of the system's performance and effectiveness. Finally, in Chapter 6, the thesis concludes by summarizing the key findings and contributions of the study. It also outlines potential directions for future research, highlighting areas that require further investigation and improvement.

Chapter 2

Related Work

In this chapter, we provide an extensive review of relevant research to align with the scope of our study. We commence with an introduction to well-established categorizations of SLAM based on comprehensive literature review papers, which is presented in the SLAM Categories (2.1) section. Subsequently, our main emphasis lies within the realm of ”Vision-centric Systems” (Section 2.2), as it closely aligns with the core objectives of our research. Within this section, we delve into both visual SLAM and VO methods. Additionally, we explore LiDAR odometry and SLAM approaches in the ”LiDAR-centric Systems” section (Section 2.3), while acknowledging that some papers in the two previous sections incorporate sensor fusion techniques. To further enrich our analysis, we dedicate a section to discuss relevant datasets (Section 2.4), focusing on the collection of similar datasets and the specific sensor configurations employed. Lastly, in the ”Depth Estimation Methods” section (Section 2.5), we critically examine foundational works in this domain, thus establishing the rationale behind the integration of

depth estimation within our SLAM methodology.

2.1 SLAM Categories

The literature on SLAM encompasses a diverse range of categories, which will be discussed in this section based on a comprehensive review of existing literature papers. Subsequently, the reviewed papers will be examined and analyzed within the context of these identified categories. It is worth noting that odometry holds significant importance in SLAM systems. According to [12], their core distinction lies in their objectives. Odometry prioritizes local consistency, incrementally estimating camera/robot poses while possibly conducting local optimization. In contrast, SLAM targets a globally consistent camera/robot trajectory and map, leveraging loop closure to minimize drift by detecting re-visited areas. Usually, odometry is considered to be a part of SLAM. Therefore, our literature review will encompass both odometry and SLAM methodologies.

2.1.1 Optimization-based SLAM and Filter-based SLAM

Numerous review papers exist that classify SLAM systems into Optimization-based SLAM and Filter-based SLAM, such as [13, 14]. We will discuss them in this section and mention their relation to Keyframe-based SLAM.

Optimization-based SLAM

Optimization-based methods in SLAM, such as bundle adjustment, employ a least squares approach to solve the problem of SLAM. This approach is a standard methodology for computing a solution in an overdetermined system by minimizing the sum of the squared errors in the equations, which has been widely used across various domains.

In the context of SLAM, the Optimization-based approach utilizes a graph representation to model the problem. Each node in the graph corresponds to a specific pose of the robot during mapping, typically denoting camera or robot positions at different time steps. The edges connecting the nodes encode spatial constraints, capturing the relative motion between consecutive poses or the observations of common features between different poses.

The introduction of constraints between the poses acknowledges the inherent uncertainty associated with the measurements. Due to sensor noise and other sources of error, the constraints contain uncertainties that need to be addressed. The primary objective of Optimization-based optimization is to construct the graph and find an optimal configuration of the nodes that minimizes the error induced by these constraints. This optimization process refines the estimates of the robot's poses and the underlying map of the environment.

Optimization-based SLAM involves an interplay between the front-end and back-end components. The front-end module is responsible for feature extraction, feature matching, and the identification of new constraints. The consistency of the map, obtained

through the back-end optimization, aids the front-end by narrowing down the search space for new constraints, thereby enhancing the efficiency and accuracy of the overall process. The back-end optimization performs the actual adjustment of the poses and map, utilizing algorithms such as bundle adjustment to determine the optimal configuration that minimizes the aggregate error stemming from the constraints.

Filter-based SLAM

Filter-based SLAM algorithms treat the simultaneous localization and mapping problem as a state estimation task, where the state represents the current position and map information. These algorithms iteratively update the state using filtering techniques, leveraging actions and measurements to refine the estimate over time.

One prominent filter-based SLAM algorithm is the Kalman Filter, which operates in two steps: prediction and update. In the prediction step, the system dynamics are used to estimate the state, while the update step assimilates measurements to improve the state estimate. Weighting the prediction and measurement information is performed based on their respective uncertainties or covariances. The Kalman Filter is particularly effective in linear and Gaussian error scenarios, as it is the optimal estimator for minimizing mean square error. However, limitations arise when the assumptions of linearity and Gaussian errors do not hold.

To address the limitations of the Kalman Filter, the Extended Kalman Filter (EKF) is commonly employed. The EKF extends the Kalman Filter by linearizing the system using a first-order Taylor expansion around the estimate. This extension enables state

estimation in nonlinear systems. Despite its popularity, the EKF can introduce errors and lead to suboptimal performance due to linearization approximations.

Another extension of the Kalman Filter is the Unscented Kalman Filter (UKF), which approximates the probability distribution by sampling sigma points. By employing an unscented transformation, the UKF captures statistics such as the mean and variance for a random variable undergoing a nonlinear transformation. The UKF provides improved estimates in nonlinear systems compared to the Kalman Filter but can be computationally more demanding.

The Particle Filter, also known as the Monte Carlo Localization (MCL) algorithm, represents the state estimate using a set of particles, with each particle representing a potential state hypothesis. The particle weights determine the likelihood of each particle reflecting the true state. As new measurements become available, the particles are re-weighted based on their agreement with the observations. The Particle Filter offers flexibility and can handle nonlinear systems and non-Gaussian errors, making it suitable for a wide range of scenarios. However, challenges such as dimensionality, particle divergence, and computational complexity exist, motivating the development of variant algorithms to address these limitations.

Keyframe-based SLAM

According to [15], all Optimization-based monocular vision SLAM solutions are keyframe-based approaches, and the current SOTA methods follow this framework. Filter-based methods, such as those employing a Kalman filter, combine localization and mapping by

tightly coupling the camera pose with the entire state of all 3D map points in the map. Consequently, in filter-based systems, the camera pose and map are jointly updated at every processed frame. Conversely, keyframe-based methods separate the Localization and Mapping steps. Camera localization is performed on regular frames using a subset of the map, while a Optimization-based optimization is conducted specifically on keyframes, which are selected key moments in the camera trajectory. This separation allows for more efficient processing and optimization of the map using keyframes.

2.1.2 Feature-based SLAM and Direct SLAM

In the field of VS, another categorization exists based on the distinction between feature-based and direct approaches, as discussed in previous studies [16, 17].

Feature-based SLAM

Feature-based SLAM algorithms adopt an approach where images are analyzed to identify and extract specific 2D features or keypoints, such as corners or distinctive image structures. These extracted features serve as the primary basis for estimating the robot's location and mapping its surroundings. By focusing on these informative features, other positional information in the images may be disregarded, simplifying the overall SLAM process.

Feature-based SLAM algorithms employ techniques such as feature detection, feature matching, and feature tracking. Initially, features are detected within the images, typically by identifying regions with significant intensity variations or local geometric prop-

erties. Feature descriptors are then computed to characterize these detected 2D features, providing distinctive representations for matching across different frames. Matching features across consecutive frames helps establish correspondences and track the detected features over time.

The estimation of the robot's location and mapping of the environment is achieved by utilizing the observed feature correspondences. Using geometric techniques, such as triangulation or bundle adjustment, the relative pose between camera frames can be inferred from the known correspondences. This information is then integrated into the map, allowing for the reconstruction of the environment.

Feature-based SLAM algorithms offer advantages such as computational efficiency and robustness to noise and environmental changes. By focusing on key features, these algorithms can reduce the amount of data processing required and simplify the estimation process. Additionally, features often provide distinctive and discriminative information, allowing for accurate localization and mapping.

Feature-based SLAM algorithms may face challenges in situations with limited or ambiguous feature availability, such as environments low in texture or with high repetitive patterns. Nonetheless, feature-based SLAM remains a widely used and effective approach in various real-world applications.

Direct SLAM

Direct SLAM algorithms differ from feature-based methods as they do not rely on explicitly detecting and extracting keypoints or features from images. Instead, direct SLAM al-

gorithms utilize the raw intensities of the images to estimate the robot's location and map its surroundings. By directly utilizing the image intensities, these algorithms can leverage more information from the images, leading to potentially more robust estimations and the creation of more detailed maps of the environment.

In direct VS, the estimation process involves jointly optimizing the camera poses and the underlying map by directly minimizing the photometric error between the observed image intensities and the reprojected intensities from the map. This direct optimization of pixel intensities allows for better utilization of information and can handle challenging scenarios, such as feature-scarce environments, where feature-based approaches may struggle.

However, the use of dense measurements in direct SLAM algorithms comes at the cost of increased computational complexity. For the case of VS, since every pixel in the image contributes to the optimization process, direct methods typically require more computational resources compared to feature-based approaches. The large-scale optimization of image intensities can pose challenges in terms of memory usage and processing time, particularly in real-time or resource-constrained applications.

Despite the higher computational costs, direct SLAM algorithms have demonstrated advantages in terms of robustness and the creation of detailed maps. By leveraging the full image information, these algorithms can handle diverse environmental conditions and produce dense and accurate reconstructions. Direct methods have been successfully applied in various applications, such as augmented reality and autonomous navigation.

2.1.3 Deep Learning in SLAM

Deep learning-based visual odometry methods have emerged as a promising approach, offering advantages in various categories such as supervised learning, unsupervised learning, and hybrid methods that combine deep learning with conventional SLAM techniques [2].

Supervised Learning-based Visual Odometry

Supervised learning-based visual odometry methods leverage ground truth supervision, such as precise camera locations, to train models in an end-to-end manner. These methods directly map input image sequences to camera poses, enabling direct estimation of the camera motion using deep neural networks. Supervised learning-based approaches benefit from the availability of ground truth data but are limited by the requirement for labeled training data and may struggle with generalization to unseen scenarios.

Unsupervised Learning-based Visual Odometry

Unsupervised learning-based visual odometry methods focus on the idea of view synthesis. They attempt to synthesize the view of one image based on estimated depth and the relative pose between two images. By comparing the generated image with the target image, typically using pixel-wise differences or other similarity metrics, the model can learn to estimate accurate camera poses. In other words, they use view synthesis and depth consistency as the supervisory signal, bypassing the need for ground truth annotations. These methods offer more flexibility but are subject to challenges such as handling

occlusions and dealing with scene dynamics.

Hybrid SLAM

Hybrid approaches combine deep learning models with conventional, non-learning-based SLAM methods. These methods exploit the maturity and effectiveness of traditional SLAM techniques while harnessing the robustness and adaptability of deep learning models. Deep learning modules are incorporated into or replace specific components within the conventional SLAM framework. This integration allows for improved performance in challenging scenarios and environments. Hybrid methods bridge the gap between established SLAM methodologies and the advantages offered by deep learning-based approaches.

2.2 Vision-centric Systems

Within this section, we present an in-depth analysis of the reviewed papers pertaining to visual SLAM and odometry, focusing on the aforementioned terminologies. Notably, several of these methods offer the possibility of integrating inertial sensors alongside cameras, thereby augmenting their capabilities. Each approach will be thoroughly discussed individually to provide a comprehensive understanding of their respective methodologies and performance.

2.2.1 Literature

LSD-SLAM (Large-Scale Direct SLAM) [18] is a monocular SLAM algorithm that creates a map of the environment using key-frames comprising a camera image, an inverse depth map, and variance. The algorithm tackles the problem through three components: tracking, depth map estimation, and map optimization. The tracking module estimates camera pose by minimizing photometric error, while the depth map estimation module determines key-frame selection and updates the depth map using stereo comparisons and outlier removal. The map optimization aligns key-frames with different scales by minimizing an error function based on photometric and depth residuals. Additionally, loop-closure detection is performed by comparing transformations between key-frames, considering uncertainties. LSD-SLAM provides efficient and accurate real-time tracking, semi-dense mapping, and loop closure in large-scale environments using a monocular camera.

SVO (Semi-Direct Visual Odometry) [19] combines elements of both direct and indirect SLAM methods while employing sparser maps. Unlike traditional direct methods, SVO extracts feature points from keyframes but uses a direct method for frame-to-frame motion estimation on these tracked features. Bundle adjustment is performed to optimize both the structure and pose. Extracted 2D features are assigned depth using a probabilistic depth-filter, and once their certainty exceeds a given threshold, they become 3D features that are added to the map. SVO offers near-constant runtime and high frame rates, maintaining good positional accuracy even during fast and variable motion. However,

it should be noted that SVO lacks loop closure and global map optimization, focusing primarily on the tracking component of SLAM.

DSO (Direct Sparse Odometry) [20] is a direct method for visual odometry that utilizes a sparse map representation. Unlike SVO, DSO does not rely on feature-point extraction but instead employs the direct photometric method for tracking. However, DSO takes a different approach by dividing the image into regions and sampling pixels from regions with intensity gradients to ensure distribution across the image. This strategy contributes to an elegant solution for direct visual odometry. Although the initial implementation of DSO did not include global map optimization or loop closure, resulting maps exhibited relatively low drift. As a result, DSO offers a robust and accurate odometry solution, despite its sparse map representation and the absence of certain global optimization components typically found in complete SLAM systems.

ORB-SLAM2 [21] is a widely recognized visual SLAM approach that exploits visual features for the purpose of simultaneous tracking and mapping. It employs the ORB descriptor to extract distinctive visual features within the scene. The keyframe-based methodology of ORB-SLAM2 involves the initial detection of visual features in keyframes, followed by their tracking across subsequent frames. Loop closing, a crucial component of this SLAM system, benefits from the utilization of bags of binary words. Notably, ORB-SLAM2 exhibits real-time performance, with all three modules, tracking, mapping, and loop closing, operating in parallel. Moreover, this SLAM framework is not limited to monocular cameras alone, as it is compatible with both stereo and RGB-D cameras.

ROVIO (RObust Visual Inertial Odometry) [22] is a monocular visual-inertial odometry algorithm that is suitable for use in dynamic hand-held experiments and unmanned aerial vehicles (UAVs). ROVIO employs an EKF to track multilevel patch features and uses pixel intensity errors as innovation terms during the update step. The landmark representation in ROVIO involves a bearing vector and inverse-distance parametrization, which are refined as part of the iterated EKF states, eliminating the need for an initialization procedure. Maplab [23], an open and research-oriented visual-inertial mapping framework, includes ROVIOLI (ROVIO with localization integration), which serves as a mapping and localization frontend based on the ROVIO algorithm.

VINS-Mono [24] is a versatile and robust monocular visual-inertial state estimator that leverages a tightly-coupled, nonlinear optimization-based method to achieve precise visual-inertial odometry. By integrating pre-integrated IMU measurements and feature observations, VINS-Mono achieves high-precision estimation. The system incorporates loop detection and loop closure modules, which facilitate efficient re-localization with minimal computational overhead. Moreover, VINS-Mono introduces an online temporal calibration technique [25] to enhance sensor fusion performance. This approach jointly optimizes time offset, camera and IMU states, as well as feature locations in a SLAM system.

DeepVO [26] is a deep learning-based VO method that combines a recurrent neural network (RNN) and a convolutional neural network (ConvNet) architecture. The ConvNet encoder, inspired by the FlowNet structure, extracts visual features suitable for optical flow and self-motion estimation from pairs of input images. These features are

then passed through the RNNs to model the temporal correlation of the extracted features, enabling the incorporation of both past experience and current sensor observations in the pose estimation process. DeepVO is trained on large-scale datasets with ground truthed camera poses as labels. Experimental results demonstrate that DeepVO achieves reasonable pose estimation performance, even in previously unseen scenarios. When evaluated on the KITTI odometry dataset, this data-driven solution outperforms conventional VO methods, such as ORB-SLAM without loop closure.

SFM-Learner [27], introduced by David Lowe's team at Google, is a pioneering work in the field of unsupervised visual odometry. It presents an innovative framework for jointly learning monocular depth estimation and camera motion (6-DoF transformation) without relying on explicit supervision. The approach leverages view synthesis and depth consistency as the guiding signal for training. SFM Learner comprises two networks: a single view depth estimation network and a pose and explainability network. The former network focuses on estimating depth from individual images, while the latter network predicts camera poses and identifies unexplainable pixels caused by factors such as motion, occlusion, or visibility. This integrated approach provides a robust foundation for unsupervised learning of depth and motion estimation, advancing the capabilities of visual odometry systems.

DF-VO (Depth and optical Flow Visual Odometry) [28], exemplifying a hybrid approach, selects features to solve the essential matrix and recover camera motion by utilizing candidates generated through deep learning. The candidate generation process relies on an optical flow prediction network, enabling any pixel within the dense optical flow to

serve as a tracking candidate. In contrast to traditional feature-based methods that solely utilize salient feature points for matching and tracking, DF-VO considers all pixels within the dense optical flow. Notably, DF-VO demonstrates improved performance compared to end-to-end methods, including both supervised and unsupervised approaches.

RPV-SLAM (Range-augmented Panoramic VS) [29] is an innovative RGB-D SLAM that extends the capabilities of ORB-SLAM2 [21] and OpenVSLAM [30] framework with the use of a 360° panoramic camera and sparse LiDAR ranges. This approach enables the system to create a dense depth map for a specific region in the image plane using bi-interpolation, which is then merged with the RGB image to create an RGB-D image. By doing so, some map points are directly generated using depth information, while others are created using triangulation between close keyframes. The resulting RGB-D mode is then processed by ORB-SLAM2's tracking, mapping, and loop closing modules.

2.3 LiDAR-centric Systems

This section presents a comprehensive discussion of the reviewed papers on LiDAR-based SLAM and odometry, building upon the previously mentioned terminologies. It is important to note that certain methods within this domain provide the opportunity to integrate inertial sensors in conjunction with LiDAR technology, further enhancing their capabilities. While the primary focus of this thesis lies in vision-centric systems, we will nonetheless delve into a selection of significant works individually, and examine some of them within our dataset.

2.3.1 Literature

LOAM [31] proposes an innovative methodology that integrates scan matching techniques with a robust motion estimation framework, enabling real-time odometry estimation and map generation. Leveraging the inherent geometric properties of LiDAR point clouds, specifically exploiting point-to-plane correspondence, the LOAM approach achieves optimized odometry estimation and accurate mapping outcomes. Comprehensive experimental evaluations conducted in diverse environments, encompassing both indoor and outdoor scenarios with varying motion dynamics, demonstrate the superior accuracy and real-time performance of LOAM. This advancement holds tremendous potential for autonomous navigation, robot mapping, and augmented reality applications. With its practical implementation and significant contributions to real-time LiDAR-based SLAM, the paper serves as a guiding resource for the advancement of robotic perception and autonomous systems, driving further research and innovation in the field.

Google's Cartographer [32] is a highly regarded real-time solution for indoor mapping, offering globally consistent maps by combining scan-to-submap matching, loop closure detection, and graph optimization. Cartographer is a SLAM system using the backpack mapping platform, which achieves real-time mapping and loop closure at a 5 cm resolution. It uses grid map-based representation with flexible resolution and sensor choices. The Cartographer technique is divided into two processes. The first process is called local SLAM which uses a Ceres scan matcher on a small map, known as a submap, to estimate the pose and orientation of the vehicle. The second process is known as global

SLAM which optimizes the pose by utilizing a larger scan matcher on global maps (a global map generated from assembling all submaps).

The Lidar SLAM in [33] utilizes transformations derived from ICP to construct a pose graph structure. This structure is subsequently employed in loop closure detection to formulate an optimization problem that facilitates updates for keyframes positioned along the trajectory. These updates effectively rectify the constructed environment map and mitigate errors accumulated during ICP odometry. Nonetheless, the system is vulnerable to local minima scenarios where ICP convergence can be compromised.

SuMa [34] and its extension SuMa++ [35] represent advancements in Lidar-based SLAM methodologies through the incorporation of semantic maps. SuMa++ builds upon SuMa's foundation and introduces semantic segmentation of point clouds using laser technology. This integration of semantic information enhances pose estimation accuracy in complex and uncertain environments. Leveraging the semantic coherence between scans and the map, SuMa++ effectively eliminates dynamic objects and utilizes higher-level constraints during the ICP procedure. By integrating both semantic and geometric information from three-dimensional laser range scans, SuMa++ achieves significantly improved pose estimation accuracy, showcasing the potential of semantic-driven approaches in Lidar SLAM.

2.4 Datasets

The availability of diverse and comprehensive datasets plays a crucial role in advancing research and development in the SLAM community. In this section, we present a literature review that focuses on existing related datasets utilized for outdoor SLAM. By conducting a systematic comparison of these datasets, we aim to identify their strengths, limitations, and unique characteristics, with the ultimate goal of highlighting the significance of our dataset, the YUTO MMS Dataset.

2.4.1 Literature

MADMAX [36] focuses on the development of a data set specifically designed for planetary rovers. The data was collected from various Mars analog sites in the Moroccan desert, featuring diverse environmental conditions. The data set comprises 36 navigation experiments and incorporates multiple types of sensors, including monochrome stereo cameras, a color camera, stereo omnidirectional cameras, and an IMU. Notably, the data set does not include a 3D LiDAR sensor. The authors evaluate the performance of ORB-SLAM2 and VINS-mono algorithms on the dataset, providing a valuable baseline for future research. To establish ground truth pose information, they utilize RTK-GPS with two antennas, enabling precise measurements of both position and orientation.

CUL [37] presents a dataset collected in outdoor environments using a vehicle in Gangnam, Pangyo, and Daejeon in South Korea. The data collection utilized a range of sensors including 3D LiDAR, 2 LiDAR, GPS, VRS GPS, 3-axis FOG, IMU, and a

wheel encoder altimeter. Notably, the dataset includes two 3D LiDAR sensors tilted at 45 degrees, positioned on the right and left sides of the vehicle, enabling comprehensive coverage of high-rise buildings. The dataset comprises six sequences with a total distance of approximately 44.81 kilometers. Although ground truth data is not available, the authors utilized the most accurate sensors in conjunction with pose-graph SLAM, semi-automatic ICP (Iterative Closest Point) registration, and manual loop-closure selection to generate a reliable baseline for benchmarking and evaluation purposes. The dataset serves as a valuable resource for researchers in the field of urban LiDAR data analysis and SLAM algorithm development.

NCLT [38] is a valuable resource for researchers working on various tasks such as autonomous driving, robotics, and 3D scene reconstruction. This dataset comprises a large collection of both indoor and outdoor data, including LiDAR scans and high-resolution panoramic images captured by a diverse set of sensors. The sensor suite includes an upright 3D LiDAR, a panoramic camera, and 2D LiDARs, as well as IMU, GPS, and FOG. The dataset provides a diverse set of scenes and environmental conditions, with over 15 months of data collected during 27 collection sessions, each varying in length, time, sky, foliage, and snow. The repeated exploration of the same environment and the use of a Segway robot to collect the data ensures a comprehensive and diverse dataset. For their ground truth, they have used GPS signals and SLAM wherever the GPS signal is not available.

MPO [39] presents two sets of datasets for place categorization, Dense MPO and Sparse MPO. Dense MPO comprises a set of static images, which is not suitable for

SLAM purposes. Sparse MPO, on the other hand, consists of real-time panoramic scans of sparse 3D reflectance point clouds recorded while driving a car, making it suitable for SLAM. The sparse data were acquired while driving a vehicle at speeds between 30 and 50 kph through different areas of Fukuoka city in Japan. The sensors used to acquire the Sparse MPO dataset include an upright 3D LiDAR, an omnidirectional camera, an Inertial Measurement Unit, and a GPS. The dataset was collected under different weather conditions and at various times of the day, providing a diverse range of scenes and environmental conditions, but the problem is that sequences are too short for SLAM [40].

In TUM-Omni [41], in addition to their proposed methods, the authors curated a dataset for evaluation purposes. The dataset was captured using a handheld global-shutter USB3 camera equipped with a 185° field of view fisheye lens, in indoor environments. The authors publicly provide five indoor sequences on their website for research purposes. Ground truth data was acquired using a motion capture system that covers an area of approximately 7×12 m. However, errors were only computed on the part of the trajectories where ground truth data was available, meaning that some sequences fell out of the motion capture-covered region. The authors also generated two additional sequences using the Gazebo simulator, which were not released publicly.

In [42], data collection was conducted in an urban environment encompassing both the Ford research campus area and the downtown area in Dearborn. The dataset was acquired using a modified Ford F-250 vehicle equipped with a variety of sensors, including an upright 3D LiDAR, a panoramic camera, a 2D LiDAR, GPS, and IMU. The vehicle traversed the test environment multiple times, resulting in two distinct trajectories that

provide extensive coverage of the surroundings. The authors have not provided the total sequence length, but I estimated the sequences using Google Maps [1] distance calculations, at around 4 KM in total. Also, in their dataset directory, the images folder within the dataset comprises five subfolders, each dedicated to an undistorted camera view captured by the panoramic camera, without any additional processing.

In the San Francisco dataset [43], the authors employed a range of sensors, including an upright 3D LiDAR, Ladybug3 panoramic camera, IMU, GPS, high-definition cameras (Prosilica), and a Distance Measurement Instrument (DMI), mounted on a mobile mapping vehicle. The dataset was collected in San Francisco and comprises approximately 150,000 panoramic images captured at 4-meter intervals, which were subsequently converted into approximately 1.7 million perspective images. It is important to note that the dataset's sampling rate, though beneficial for landmark identification, may not be suitable for SLAM due to the relatively sparse capture intervals. The authors aligned the LiDAR data and the panoramas with the building outlines, facilitating more accurate landmark identification and localization. This dataset provides a valuable resource for research in mobile mapping, landmark recognition, and related areas.

Bovisa [44] presents a dataset collected using a variety of sensors mounted on a robot. The sensors used include a two-camera setup for binocular vision, a single camera for monocular vision, an omnidirectional camera, two pairs of 2D LiDARs, an IMU, and GPS. The data was collected in three sequences from the same location at the Bovisa campus of Politecnico di Milano, focusing on outdoor environments. The ground truth

[1] `https://www.google.com/maps`

for the dataset is obtained from RTK-GPS. While the authors have not provided the exact sequence length, an estimated length of approximately 4 km was determined using Google Maps.

2.5 AI Depth Estimation Methods

Within this section, we comprehensively review fundamental works in the field of depth estimation, drawing upon their significance to substantiate the justification for our own Depth Estimation module. By examining these foundational contributions, we aim to establish the context and rationale behind our approach in this domain.

2.5.1 Literature

[45] addresses the challenge of training unsupervised depth estimation models. During training, the network receives a color image from the left camera and a target color image from the right camera and learns to generate per-pixel disparities that align the left input image to the right target image. However, to overcome the misalignment issue between the disparity map and the desired left input image, the authors propose sampling from the right image during training to generate the left image. At test time, only one image is required. Although this approach produces disparities with texture copy artifacts, the method achieves higher-quality results, surpassing state-of-the-art supervised methods on the KITTI dataset at the time it was published. The model is trained to predict depth maps with the introduction of a left-right consistency loss to ensure mutual consistency

between the depth maps. Importantly, predictions are made independently for each frame without temporal regularization.

[46] tackles the challenge of estimating depth distributions in a scene, considering the significant variability observed in different images. Existing methods struggle due to the wide range of depth values present, with some images having depth values concentrated in a small sub-interval while others exhibit a wide variety of depth values. To address this complexity, the authors propose an adaptive binning strategy, where the depth interval is divided into variable-sized bins that can change for each input image. The architecture incorporates a patch-based transformer module, called mViT, to extract global information and estimate adaptive bins and range attention maps. The model then estimates the probability distribution over these bins for each pixel, yielding the final depth map through expected value computation. Notably, this framework, which employs transformers for pixel-wise dense prediction, offers flexibility in choosing encoder-decoder architectures. This approach represents a novel design in utilizing transformers for depth estimation and demonstrates the effectiveness of estimating adaptive bins for achieving accurate depth estimation.

Monodepth2 [47] is a prominent method for self-supervised depth estimation utilizing deep convolutional neural networks. It employs a fully convolutional U-Net architecture for accurate depth prediction, coupled with a pose network to ensure temporal consistency in video frames during training (excluding inference). Monodepth2 incorporates two loss functions to optimize network weights: a photometric loss, also known as view synthesis loss, which tracks the relationship between the current and previous images,

and a smoothness loss that promotes depth values to exhibit smooth transitions, minimizing sudden depth changes between neighboring pixels in the output image. With its comprehensive approach and effective loss functions, Monodepth2 has garnered significant attention as one of the leading methods in the field of self-supervised monocular depth estimation.

ManyDepth [48] builds upon the previous work of the Monodepth2 team, although it has not gained the same level of attention. The key innovation of ManyDepth lies in the utilization of a cost volume. By leveraging two consecutive frames, they create a cost volume that explores the likelihood of different depths within a set of ordered planes perpendicular to the optical axis. To estimate the distance between frames, a pose network is employed. Each frame is encoded into a deep feature map and then warped to match the viewpoint of a target frame. However, regions with moving objects or untextured surfaces may result in holes in the cost volume. To address this issue, ManyDepth detects these problematic regions using a motion mask and subsequently employs Monodepth2 to obtain depth estimates specifically for those areas.

DARS [49] is a purpose-built framework that leverages a pre-trained depth estimation network in conjunction with available sparse LiDAR points during the inference process, enabling adaptation of the network to each new input. While DARS excels in providing more accurate and denser depth estimations, it is specifically tailored for perspective images. The framework incorporates a double-stage adaptive refinement scheme to enhance the depth output produced by a baseline network. In the initial stage, a Delaunay-based correction module significantly improves the depth estimation quality. Subsequently, in

the second stage, a PSO [50] fine-tunes the f-BRS [51] parameters (scale and bias) to further enhance the estimation accuracy.

2.6 Summary

In this chapter, we provided an overview of research in the field. First, we categorized SLAM methods, then explored vision-centric and LiDAR-centric systems. We continued with discussion in related datasets and finally introduced some key works in depth estimation techniques.

Chapter 3

YUTO MMS Dataset

In this chapter, we explain the dataset we used in this work. The YUTO MMS Dataset comprises four sequences, totaling over 18.9 kilometers in length, collected on August 12, 2020, and June 21, 2019. These sequences were obtained using a vehicle equipped with a panoramic camera, a tilted LiDAR, a GPS, and an IMU. The data are a result of fieldwork gathered by driving the vehicle at two locations: the York University Keele Campus in Toronto and the Teledyne Optech headquarters in Vaughan. It is pertinent to mention that the dataset paper is currently undergoing the submission process. Once the paper is officially published, the dataset will be made accessible to the public. This chapter presents an overview of the YUTO MMS Dataset, including collection procedure, sensor-related configuration and synchronization, data structure, and format.

The creation and publication of a dataset entail significant effort, and the existence of this dataset owes its realization to the exceptional collaborative efforts within AUSMLab. To delineate my individual contribution, I played a role in quality assessment, testing

against baseline SLAMs, developing user-friendly performance evaluation codes, and primarily, drafting its accompanying research paper.

3.1 Distinctive Dataset Challenges

The YUTO MMS Dataset poses notable challenges for visual SLAM systems, given its inclusion of both natural landscapes and urban environments. Furthermore, the dataset was collected under regular daily conditions, depicting a bustling road with abundant cars and buses as dynamic objects, along with the presence of pedestrians and cyclists. It also encompasses multiple loops within the sequences. Additionally, it has numerous turning points, traffic lights, and stop signs, preventing constant speed and causing acceleration which introduces additional complexities. Dataset offers a valuable opportunity to evaluate the effectiveness and resilience of visual SLAM algorithms in dynamic urban scenarios.

While recent SLAM datasets such as [52–55] have made significant contributions to the challenges of both visual and LiDAR SLAM by incorporating factors like quick camera motions, dynamic objects, environments low in texture, and illumination changes (this is only for vision), none of them have explicitly explored the scenarios in which the 3D LiDAR sensor introduces a new challenge itself. In contrast, our dataset not only presents challenging scenarios for the camera and LiDAR sensors but also introduces an additional challenge for LiDAR usage which has received less attention in the literature but at the same time, holds significant utility. Specifically, we employ a tilted LiDAR

configuration, where the spinning axis of the LiDAR is not upright.

The tilted LiDAR offers several advantages in the context of SLAM applications. By virtue of its tilt, the LiDAR sensor can effectively detect and capture features at greater heights surrounding the road, including traffic signs and lights, light poles, highway billboards, and tall buildings. These tall features are predominantly static in nature, which proves highly beneficial for SLAM algorithms as their tracking can provide valuable information for accurate localization and mapping. Conversely, in lower heights, where dynamic objects such as cars, buses, pedestrians, and bicycles are prevalent, the SLAM performance may be compromised. Nevertheless, the ability of the tilted LiDAR to detect and leverage the static tall features contributes significantly to the overall efficacy of SLAM systems, enhancing their robustness and accuracy in urban environments.

The tilted configuration of the LiDAR sensor introduces unique challenges that affect both LiDAR-centric and coupled visual-LiDAR SLAM methodologies. In LiDAR-centric SLAM, object tracking becomes particularly challenging as objects remain in the narrow LiDAR-covered regions for only brief periods, given the limited coverage width. Moreover, the tilted configuration poses difficulties in point matching and accurate geometry modeling for LiDAR SLAM, as it lacks full rectangular coverage of the road ahead. This limitation complicates feature tracking, since a significant portion of new features emerges directly in front of the vehicle during forward motion. Additionally, the tilted configuration results in sparse coverage, especially at lower heights where only a few objects may fall within LiDAR's range. Coupled visual-LiDAR SLAM systems face obstacles, as well, as the overlapping region between the camera and the tilted LiDAR

sensor does not conform to a rectangular shape primarily covering the road ahead, resulting in a restricted and challenging overlapping region. Furthermore, the topmost LiDAR beams often do not hit objects in the scene due to the absence of tall objects or buildings, causing sparse point distributions in the topmost portion of the lidar data.

Considering the unique challenges posed by the dataset, we have opted to utilize the Maverick MMS, developed by Teledyne Optech. This system is specifically designed for mapping applications, particularly in environments characterized by tall buildings. The Maverick system incorporates the Ladybug 5 panoramic camera and the Velodyne HDL-32E LiDAR, featuring a spinning axis positioned at a 45-degree angle relative to the optical axis of the camera. By employing this configuration, the system can effectively capture comprehensive visual and LiDAR data, ensuring a holistic understanding of the environment. To establish accurate ground truth, the Maverick system utilizes RTK positioning based on the signals from GPS receivers.

3.2 Similar Datasets

In the literature review chapter, we conducted a comprehensive analysis of the most relevant papers in the field of outdoor SLAM datasets, including [36–39, 41–44] summarised in Table 3.1 and Table 3.2, to provide an in-depth understanding of the subject matter. While each reviewed work offers valuable insights and contributes to the advancement of SLAM research, it often lacks certain aspects that are of particular interest in our study. Some datasets utilize non-vehicle carriers, such as handheld devices or robots, which may

not fully capture the complexities of vehicular motion which is crucial for realistic SLAM analysis. Additionally, certain datasets have limited coverage in terms of total distance, which restricts the evaluation of SLAM performance in expansive environments. Furthermore, some datasets focus on indoor scenarios rather than the challenging conditions encountered in outdoor environments. In terms of ground truth accuracy, some datasets do not provide precise calculations, hindering the potential for accurate benchmarking. Moreover, the absence of a tilted LiDAR configuration limits the ability to effectively capture and model high-rise buildings and other tall structures. Lastly, the absence of a panoramic or an omnidirectional camera restricts comprehensive visual perception. In contrast, our dataset, the "YUTO MMS Dataset," stands out for its unique combination of a vehicle-based carrier, vehicle speed while collecting the data, extensive total distance coverage, outdoor scenarios, precise ground truth, tilted LiDAR configuration, and inclusion of a panoramic camera. These distinctive characteristics make the "YUTO MMS Dataset" a valuable resource for advancing research and development in outdoor SLAM applications.

3.3 Applications

While the primary focus of this dataset is on SLAM and odometry applications, it holds potential for various other uses beyond these domains. One such application is depth estimation, where the LiDAR points can serve as a form of weak supervision, aiding in the generation of accurate depth maps. Additionally, the dataset contains a diverse

Table 3.1: Similar Datasets General Information

Name	Year	Carrier	Environment	Total Distance
YUTO MMS	**2023**	**Vehicle**	**Outdoor**	**18.9 km**
MADMAX Mars [36]	2021	Robot	Desert	8.2 km
CUL [37]	2018	Vehicle	Outdoor	44.81 km
NCLT [38]	2016	Robot	Indoor/Outdoor	147 km
Sparse MPO [39]	2016	Vehicle	Outdoor	Short
TUM-Omni [41]	2015	Hand	Indoor	Short
Ford Campus [42]	2011	Vehicle	Outdoor	4 km
San Francisco [43]	2011	Vehicle	Outdoor	?
Bovisa [44] (outdoor)	2009	Robot	Outdoor	4 km

Table 3.2: Similar Datasets Experimental Settings

Name	Omnidirectional Vision	3D LiDAR Configuration	Ground Truth
YUTO MMS	**Yes**	**Tilted**	**RTK-GPS & IMU**
MADMAX Mars [36]	Yes	N/A	RTK-GPS
CUL [37]	No	Tilted	-
NCLT [38]	Yes	Upright	RTK-GPS & SLAM
Sparse MPO [39]	Yes	Upright	GPS
TUM-Omni [41]	Yes	N/A	Motion Capture
Ford Campus [42]	Yes	Upright	GPS & IMU
San Francisco [43]	Yes	Upright	GPS & IMU
Bovisa [44] (outdoor)	Yes	N/A	RTK-GPS

range of objects, encompassing both dynamic and static entities, making it suitable for tasks such as object detection and tracking. Furthermore, the dataset can facilitate place recognition, enabling the development of robust algorithms for identifying and localizing specific locations within the environment. Moreover, the dataset presents opportunities for various computer vision tasks, including learning visual feature descriptors. By offering a comprehensive and diverse set of data, this dataset opens avenues for research and development in a wide range of applications beyond SLAM and odometry.

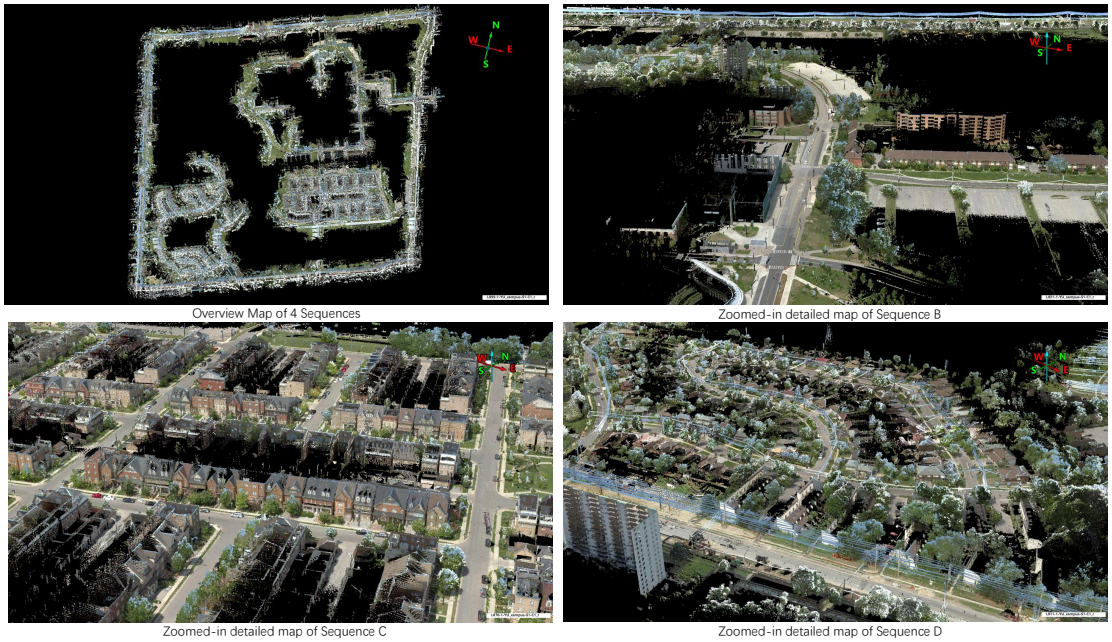


Figure 3.1: Color textured point-cloud of YUTO MMS dataset in York University campus.

3.4 Maverick Mobile Mapping System

3.4.1 System Overview

The development of Maverick MMS by Teledyne Optech was driven by the demand for a versatile and high-performance mobile mapping unit that can be easily transported. The Maverick MMS has proven to be a valuable tool in a wide range of industries and applications, owing to its robust and versatile data collection capabilities. Its datasets have been utilized in projects related to transportation safety, construction, asset management, rail, utilities, and 3D modeling.

3.4.2 System Characteristics

As shown in Table 3.3, the Maverick MMS boasts great portability, weighing in at less than 9 kilograms (20 lb). Its compact dimensions make it highly maneuverable and suitable for use in different settings. This lightweight and compact design allows for easy mounting on various platforms, including backpacks, vehicles, trains, and ATV/UTVs, enabling the system to be utilized in diverse environments and scenarios. Maverick has an Ingress Protection Rating (IP Rating) of IP65, meaning that the system provides a adequate level of protection against dust and water ingress, enabling its use in challenging outdoor conditions. Additionally, with a reasonable operating temperature range, Maverick demonstrates its capability to endure various climatic conditions.

Table 3.3: Maverick MMS Characteristics

Characteristic	Value
Operating temperature	0^o to 43^o (32^o to 110^oF)
Dimensions	34.4 cm x 21.6 cm x 36.3 cm (13.60" x 8.50" x 14.28")
Mount	mount bolts to existing roof racks
Power supply	12 V - 36 V DC
Weight	8.85 kg
IP Rating	IP65

3.4.3 Configuration

The Maverick MMS is mounted at a 45-degree angle between the optical axis of the camera and the spinning axis of the LiDAR. When mounted on a car, this results in the camera's optical axis being parallel to the ground, while the LiDAR's spinning axis is tilted 45 degrees with respect to the ground. In the experiments, images were captured at an average of 7.5 FPS, which is comparatively low in comparison with other datasets, while LiDAR points were acquired at a spinning rate of 15 RPS.

3.4.4 Sensors

The Maverick MMS comprises a Ladybug5 panoramic camera [1] as the imaging component, a Velodyne HDL-32E LiDAR [2] as the LiDAR component, and a high-precision GPS+IMU. It also includes a post-processing software called LiDAR Mapping Suite

[1] https://www.flir.com/support/products/ladybug5-usb3

[2] https://velodynelidar.com/wp-content/uploads/2019/12/97-0038-Rev-N-97-0038-DATASHEETWEBHDL32E_Web.pdf

(LMS) that generates trajectories with centimeter-level precision by performing offline bundle adjustment with the aid of high-precision NovAtel GPS+IMU data and LiDAR points.

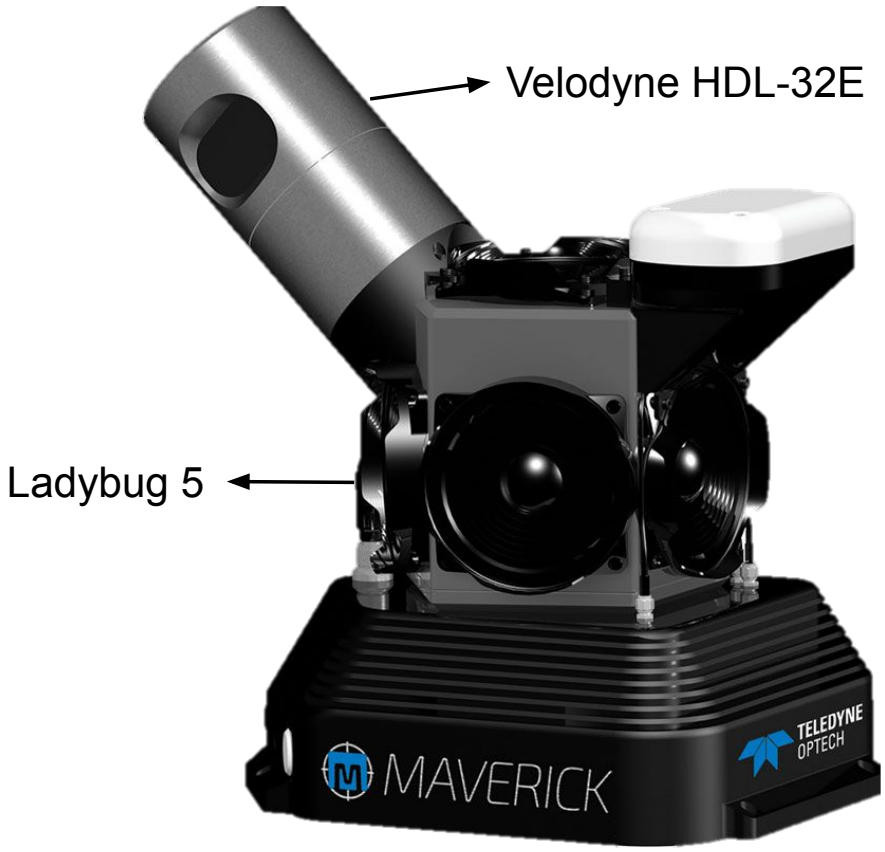


Figure 3.2: Maverick MMS.

LiDAR

As illustrated in Figure 3.2, the LiDAR component used in the Maverick MMS is the Velodyne HDL-32E, which has 32 detector pairs and a vertical field of view ranging from +10 to -30 degrees. It provides a 360-degree horizontal field of view, and generates a point cloud of up to 695,000 points per second with a range of up to 100 meters. The

absolute accuracy of this LiDAR sensor is typically within ±2 cm, making it a reliable and high-precision tool for capturing 3D spatial data from a moving vehicle. Figure 3.3 shows an illustration of the scanning mechanism of the LiDAR sensor. An array of 32 detectors performs a fan-like scanning pattern.

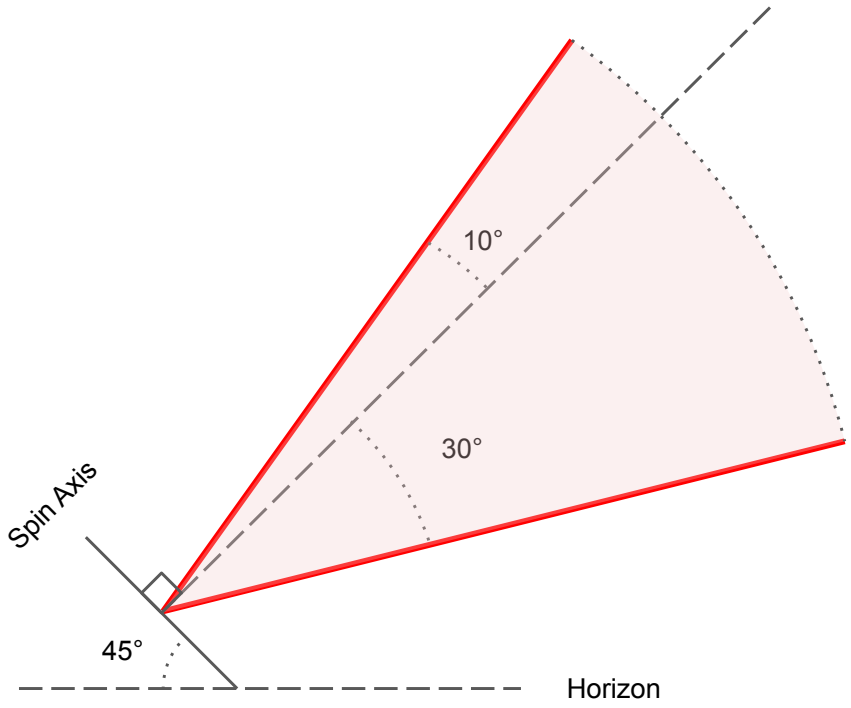


Figure 3.3: Scan mechanism of the LiDAR sensor in the tilted configuration.

Camera

The imaging component of the Maverick MMS is the Ladybug 5 camera, which uses six Sony ICX655 CCD sensors, each with a 2/3" format (see Figure 3.2). The Ladybug 5 provides a 90% field of view of the full sphere, making it capable of capturing panoramic images from all directions. With a focal distance of approximately 200 cm, it can capture high-quality images with a resolution of up to 30 megapixels (5 MP * 6 sensors). The camera's optics consist of six 4.4-mm focal length lenses, and it has been calibrated for spherical distance from 2 meters to infinity. These specifications make the Ladybug 5 camera a powerful tool for capturing high-resolution panoramic imagery in a mobile mapping context.

GPS+IMU

Maverick has a NovAtel SPAN-IGM-S1 which serves as a GPS unit. It represents a powerful solution for navigation and positioning applications, seamlessly merging raw inertial measurements with comprehensive GPS information through the SPAN system. Equipped with three accelerometers, three gyroscopes (gyros), and a NovAtel OEM615 receiver, it guarantees robust performance. The dual frequency GPS antenna ensures reliable satellite signal reception (and is essential for advanced applications such as RTK to achieve centimeter-level accuracy), while the integration of the STIM300 IMU further enhances accuracy of the system by adding the support for IMU-based RTK.

3.5 Data Acquisition and Processing

Table 3.4: Dataset Characteristics

	Sequence A	Sequence B	Sequence C	Sequence D
Region	Parking lot	Campus area	Residential area	Residential area
Camera frames	717	8382	10778	4500
Camera FPS	7.632	7.353	8.165	7.500
Raw image size	4096 x 2048	8000 x 4000	8000 x 4000	8000 x 4000
LiDAR frames	1432	17395	22992	9615
LiDAR RPS	15.266	15.259	17.418	16.025
Travelled Distance	324 m	7035 m	7965 m	3634 m
Run time	94 seconds	19 minutes	22 minutes	10 minutes
Average Driving speed	3.447 m/s	6.171 m/s	6.034 m/s	6.057 m/s
Loop	One small	One large + a few small	Many mid-size	A few
Dynamic objects	No	Yes	Yes	Yes

3.5.1 Operation

The data collection operations for our dataset involved capturing sequences from different locations and environments. Sequence A was collected at Teledyne Optech headquarters in Vaughan on June 21, 2019, between 16:43:29 pm and 16:48:19 pm under sunny weather conditions. During this sequence, the vehicle traversed around the building and

explored the surrounding parking lot area with an approximate drive speed of 3 m/s. Sequences B, C, and D were collected at York University Keele Campus in Toronto on August 12, 2020, from 16:32:16 pm to 18:49:14 pm, also under sunny weather conditions. In Sequence B, the vehicle navigated around the main campus of York University, encompassing a primary loop and several smaller loops. Sequence C focused primarily on the residential area in York Village, located adjacent to York University, and featured multiple loops within this neighborhood. Lastly, in Sequence D, the vehicle predominantly traversed through open spaces, with only a small section passing through a residential area, and it followed a distinct path that included a significant loop within a residential area, along with a few smaller loops (see 3.5). During data collection sequences B,C, and D, the vehicle operates at a considerably low speed as well, akin to a mapping scenario when equipped with the MMS along its trajectory. This deliberate choice of slow movement is driven by two critical considerations:

- Sensor Frame Rate Limitations: The sensors onboard the vehicle, including cameras and LiDAR, are constrained by their frame rate or spinning rate. Moving at a higher speed could potentially lead to a loss of crucial environmental data. This is because the sensors may not be able to capture all features adequately within the limited time frame available for data acquisition.

- Impact on 3D LiDAR Performance: Specifically, the 3D LiDAR sensor's operation is influenced by the vehicle's speed. A faster speed can introduce inaccuracies and errors in the LiDAR data. This is due to the LiDAR's spinning mechanism, which

relies on precise synchronization with the vehicle's movement. Faster speeds may disrupt this synchronization, compromising the quality and accuracy of the acquired LiDAR data.

3.5.2 Calibration

The calibration process plays a pivotal role in the operation of the Maverick MMS system, guaranteeing the seamless integration of sensors with high precision. The process has been explained in [56]. The global coordinate system for GPS/IMU, labeled as w, is established upon the WGS84 geodetic reference frame, providing a universally recognized global reference point (We will mention later that the ground truth is transformed into an X-Y-Z coordinate system so that the SLAM results can be compared with it). Simultaneously, the vehicle's coordinate system, referred to as the body frame b, is employed for local (relative to the origin) positioning and orientation. Within the Maverick MMS, the frames referred to as i, l, and c correspond to the IMU, LiDAR, and camera, respectively. It is noteworthy that a rigorous calibration procedure is conducted for all sensors, utilizing external parameters. These calibration routines ensure proper alignment of the X-Y-Z axes. This alignment is fundamental in enabling precise sensor fusion and facilitating accurate mapping capabilities.

The representation of the sensor's pose utilizes a 4x4 matrix, denoted as $p = [R,t]$. This matrix consists of a 3x3 rotation matrix R and a translation vector t. At a specific time instant k, we use the term p_k to denote the transformation of the local coordinate system, indicating its position and orientation relative to the origin. The motion between

consecutive time instances, referred to as m_k, characterizes the relative pose change from time $k-1$ to k, providing essential information for tracking and mapping applications. For a comprehensive description of a 6-Degrees of Freedom pose, we employ the pose representation T_{ab}, which succinctly conveys the pose of frame b concerning frame a, allowing precise characterization of the relative transformation between the two frames. In Table 3.5, we provide a detailed account of the extrinsic parameters for all sensors. These parameters encompass T_{bi}, representing the fixed relative transformation between the body frame and the Inertial Measurement Unit frame; T_{bl}, denoting the relative transformation between the body frame and the Light Detection and Ranging frame; and T_{bc}, signifying the relative transformation between the body frame and the camera frame. Since Sequence A has been collected at a different time than the other sequences, the parameters related to that sequence vary. These parameters play a pivotal role in the workflow of achieving accurate data from the input sensor data. The workflow is depicted in Figure 3.4.

IMU Calibration

In the tightly-coupled NovAtel SPAN system on the OEM6 platform, an initial calibration procedure is conducted for the IMU. This calibration serves to determine two critical parameters: the lever arm offset between the IMU and the GPS antenna, and the orientation transformation from the body frame to the IMU frame. The calibration leverages the base station's trajectory coordinates for reference. During trajectory processing, the data output is configured to adhere to the NAD83 (CSRS) processing datum within the se-

lected processing frame, ensuring precise alignment and compatibility between the IMU and GPS data streams. The calibration process within the integrated GPS+IMU system commences with the utilization of initial values given by the "SETIMUTOANTOFFSET" command [3]. Subsequently, the "LEVERARMCALIBRATE" command is invoked [4] to manage and oversee the calibration of the lever arm connecting the IMU and the antenna. This calibration procedure persists for a duration of 600 seconds or until the standard deviation of the estimated lever arm values converges below the threshold of 0.05 meters.

LiDAR Calibration

To precisely ascertain the LiDAR's position concerning the body frame, a set of control points on both vertical and horizontal surfaces was designated. Leveraging the mission data acquired by the Maverick system, a self-calibration technique was applied. This methodology encompassed temporal adjustments, LiDAR channel optimizations, boresight refinements, and positional refinements. The iterative self-calibration procedure persisted until the boresight values were rectified to fall below the stringent threshold of 0.008 degrees. For a more detailed illustration, taking boresight as an example, the boresight is treated as unknowns, while the observed positions of control points serve as observations. Subsequently, we iteratively refine the boresight parameters through an adjustment process.

[3] Alternatively known as the NVM command, which sets the initial value of the lever arm.
[4] It is a built-in routine in NovAtel SPAN that automatically calibrates the lever arm.

Panoramic Camera Calibration

The calibration of the six cameras employs Zhang's method [57], which entails capturing checkerboard patterns from various orientations using all camera units. The calibration process comprises an initial closed-form solution, succeeded by a nonlinear refinement guided by the maximum likelihood criterion. Subsequently, the LynxView software is utilized to perform boresight alignment between the cameras and the body frame. This method entails the selection of Light Detection and Ranging Surveyed Lines and Camera Lines, with the constraint formulated as a non-linear optimization problem to ascertain the optimal rigid transformation.

Table 3.5: Calibration Data

Sequence	Transform	Translation in meters	Rotation in degrees
A,B,C,D	T_{bi}	[-0.004, 0.068, -0.226]	[0.500, -1.560, -179.000]]
A	T_{bl}	[0.111, 0.010, -0.181]	[178.869, -44.844, 1.933]
	T_{bc}	[-0.031, 0.000, -0.111]	[0.150, -0.050, -179.650]
B,C,D	T_{bl}	[0.111, 0.010, -0.181]	[179.579, -44.646, 0.601]
	T_{bc}	[-0.031, 0.000, -0.111]	[0.000, 0.000, -179.900]

3.5.3 Synchronization

Trajectory Refinement

The ground truth trajectory is obtained by performing post-processing techniques to refine the GPS trajectory. Prior to trajectory refinement, the trajectory file and its corresponding precision file are readily available. Furthermore, the LiDAR and camera systems are calibrated to ensure accurate data acquisition. During the refinement process,

painted control points, which can be precisely identified and measured in the LiDAR intensity data, are carefully selected. The LiDAR surveyed lines are then chosen, while the camera lines are automatically synchronized with the LiDAR lines (conducted by the software explained in Sectin 3.6.2). Next, the images captured by the camera system are processed, with an image mask applied to exclude certain areas, such as those depicting the survey vehicle, from influencing the colorization of the point cloud. The centers of the control points in the LiDAR point cloud are measured, ensuring that they are unobstructed and contain sufficient laser points for accurate measurements. Once all the control points have been measured, the desired survey lines are selected for further processing. Given that the instrument used for the sample data is already calibrated, the channel corrections and boresight corrections remain fixed in the correction settings. The position offsets are set as ”free unknown,” while the drift and accuracy corrections are set as ”constraint unknown.” The resulting optimized trajectory serves as the ground truth, representing the refined and accurate trajectory of the system. For a visual depiction of this process, please refer to Figure 3.4.

Time Synchronization

In the Maverick MMS, the GPS+IMU system operates at a high frequency of 125 Hz, providing precise timing information. To ensure synchronization, GPS timestamps are used as a reference throughout the trajectory refinement process. By refining the trajectory, we generate ground truth data for all image frames, effectively synchronizing the GPS+IMU system with the panoramic camera. Given that the LiDAR and the panoramic camera

operate on different time scales, with images captured at an average rate of 7.5 FPS and LiDAR scans acquired at a spinning rate of 15 RPS, careful synchronization is required. We achieve this synchronization by aligning the LiDAR data with the GPS+IMU system as part of the trajectory refinement process. The timestamps are recorded using GPS time for accurate temporal alignment. To precisely synchronize the laser scans and panoramic images, we employ an interpolation method to determine the closest timestamp of each laser scan relative to the corresponding image. This interpolation technique ensures that the LiDAR data and panoramic images are temporally aligned, facilitating accurate sensor fusion and subsequent data analysis.

But there is a challenge here, which lies in the trade-off between temporal alignment and spatial accuracy when fusing data from sensors like LiDAR and cameras. While interpolation effectively synchronizes sensor data in time, it may assume that objects captured by these sensors at slightly different moments are perfectly aligned. When Maverick is moving, objects move between sensor readings, causing spatial misalignment. We did not tackle to solve this challenge here.

3.5.4 Translation

Considering the interdependence between certain SLAM methods discussed in Experimental Results and the projection of LiDAR point clouds onto corresponding panoramic images, this section elucidates the essential steps involved in this process. In the next section, we elaborate on the mappings between the pixel coordinate space and image coordinate space (see Figure 3.6).

Mappings Between 3D Points to Pixels in the Panoramic Image

In order to find correspondences from a pixel in the panoramic image to the camera 3D coordinate system, we use the unit sphere projection and the polar coordinate system as intermediates between them.

3D - Unit Sphere Going from the 3D camera coordinate system to the unit sphere for a 3D point, we calculate range r, the distance from the point to the camera, as follows:

$$r = \sqrt{X_c^2 + Y_c^2 + Z_c^2} \tag{3.1}$$

Where (X_c, Y_c, Z_c) points to the mentioned 3D point in the 3D camera coordinate system.

Then, to obtain the unit vector of the 3D point by projecting it onto the unit sphere:

$$(X_u, Y_u, Z_u) = (\frac{1}{r}X_c, \frac{1}{r}Y_c, \frac{1}{r}Z_c) \tag{3.2}$$

Where (X_u, Y_u, Z_u) is the projection of the point on the unit sphere.

Reversely, from the unit sphere to the camera 3D coordinate system, getting the range r for the 3D point from other sources is needed. There are two ways to acquire:

- Depth from LiDAR point-cloud
- SLAM-based methods (e.g. Triangulation)

Assuming either of the mentioned options gives r. We can write:

$$(X_c, Y_c, Z_c) = (rX_u, rY_u, rZ_u) \tag{3.3}$$

Unit Sphere - Polar Mapping from the unit sphere to the polar coordinate system is done by the following equation:

$$\begin{aligned} \alpha &= \arctan 2(X_u, Z_u) \\ \beta &= -\arcsin(Y_u) \end{aligned} \tag{3.4}$$

Where (α, β) is the presentation of the point in the polar coordinate system. α and β are longitude and latitude, respectively. For α and β we have: $\alpha \in [-180^o, 180^o]$ and $\beta \in [-90^o, 90^o]$.

The inverse of Equation 3.4 is calculated by:

$$\begin{aligned} X_u &= \cos\beta \cdot \sin\alpha \\ Y_u &= -\sin\beta \\ Z_u &= \cos\beta \cdot \cos\alpha \end{aligned} \tag{3.5}$$

Polar - Image Mapping from the polar coordinate system to a pixel in the image is shown in the following equation:

$$u = w \cdot (0.5 + \alpha/2\pi)$$
$$v = h \cdot (0.5 - \beta/\pi) \tag{3.6}$$

Where (u, v) is the presentation of the corresponding pixel in the image. And ϕ is the pixel coordinate space where:

$$\phi = \{(u, v) | u, v \in \mathbb{W}, 0 \leq u < w, 0 \leq v < h\} \tag{3.7}$$

Where h and w are the height and width of the image. And the mapping from the pixel in the image to polar coordinate system:

$$\alpha = 2\pi \cdot (u/w - 0.5)$$
$$\beta = -\pi \cdot (v/h - 0.5) \tag{3.8}$$

We use the extrinsic parameters matrix between the panoramic camera and the LiDAR sensor in the MMS. Then, using Equations 3.1, 3.2, 3.4, and 3.6, we can project the 3D points on the corresponding panoramic image.

3.6 Dataset

3.6.1 Benchmark data

Panoramic images

The dataset includes panoramic images captured by the panoramic camera, with detailed information provided in Table 3.6. Although the raw captured files may differ (Table 3.4), the resulting images are stored in the widely used JPG format, all uniformly with dimensions of 2000 pixels in width and 1000 pixels in height. It is worth noting that Sequence A differs slightly from the other sequences in terms of size, as it was captured at a different time. However, the provided information regarding all other aspects remains consistent across all sequences. In terms of image data volume, the dataset encompasses approximately 13.9 GB of data.

Table 3.6: Panoramic images general information

Sequence	A	B	C	D
Images file format	JPG	JPG	JPG	JPG
Images w*h	2000*1000	2000*1000	2000*1000	2000*1000
Number of image files	717	8382	10776	4500
Min file size	1368 KB	456 KB	468 KB	452 KB
Max file size	1592 KB	828 KB	636 KB	1012 KB
Average file size	1484 KB	541 KB	542 KB	561 KB
Total directory volume	1.06 GB	4.5 GB	5.8 GB	2.5 GB

LiDAR

The dataset includes LiDAR point cloud data, with comprehensive details provided in Table 3.7. The data is provided in the widely adopted ".bin" file format, which is consistent with the format used in the publicly available KITTI dataset [3]. This ensures compatibility and ease of integration with existing LiDAR processing pipelines and algorithms. Similar to the panoramic images, Sequence A may differ slightly in size compared to the other sequences due to its capture at a different time. However, the essential information pertaining to point cloud data, such as point density and coordinate representations, remains consistent across all sequences. The total volume of the LiDAR data is approximately 23.3 GB.

Table 3.7: LiDAR data general information

Sequence	A	B	C	D
LiDAR file format	.bin	.bin	.bin	.bin
Number of point cloud files	1432	17395	22992	9615
Min file size	124 KB	16 KB	24 KB	380 KB
Max file size	648 KB	644 KB	708 KB	668 KB
Average file size	422 KB	464 KB	470 KB	485 KB
Total directory volume	604.8 MB	7.7 GB	10.3 GB	4.6 GB

3.6.2 Reference data

LiDAR Mapping Suit

Teledyne Optech's LMS Pro serves as a robust data processing solution specifically designed for the Maverick MMS. The software enables comprehensive data processing for the entire project, encompassing trajectory, camera, and LiDAR data. The ground truth trajectories were obtained using it which is capable of producing trajectories with cm-level accuracy. The LMS utilizes high-precision GPS+IMU data and LiDAR points to perform offline bundle adjustment, ensuring precise alignment and synchronization between the ground truth trajectories and the collected data. By leveraging the capabilities of the LMS, we are able to provide ground truth trajectories that serve as a reliable reference for evaluating and validating SLAM and odometry algorithms in our dataset.

Ground Truth

The ground truth data structure for our dataset is saved in .txt format, providing centimeter-level accurate information (see Figure 3.4 for the workflow) regarding the vehicle's pose in a 6-degree-of-freedom (6- DOF) representation. Each line of the file corresponds to a specific image frame captured during the data collection process. The line contains the x, y, z coordinates representing the position of the vehicle, while the omega, phi, and kappa values denote the orientation of the vehicle at that particular point.

3.6.3 Benchmark data structure

Raw data

Figure 3.8 shows the details of the directory structure of our dataset. Each sequence has its related data, and also we have included auxiliary files for data transformation, sensor calibration, and performance evaluation of the future SLAM systems running on our dataset.

Synchronization data

The synchronization data of the camera and LiDAR is saved in a .txt format, where each line of the file corresponds to a specific image frame and lists the corresponding LiDAR frame.

Calibration data

The IMU calibration parameters of the Maverick MMS remain fixed due to the tight coupling of the GPS and IMU sensors, as illustrated in Table 3.5. However, for accurate localization, it is necessary to perform recalibration of the panoramic camera and LiDAR prior to each data collection. The camera calibration parameters are saved in a .ccp format, while the LiDAR calibration parameters are stored in a .lcp format.

Transformation data

In addition, we have included the depth map corresponding to each panoramic image. The depth values in the map are derived through the process explained in Section 3.5

where raw ranges obtained from LiDAR points projected onto the panoramic image space. The depth maps are saved in JPG format and share the same dimensions as the panoramic images. As for data size, the depth map data amounts to approximately 2.12 GB.

Auxiliary codes

We have developed MATLAB scripts that facilitate the loading and visualization of panoramic images and LiDAR scans within the MATLAB environment. These scripts enable the re-projection of LiDAR ranges onto the corresponding panoramic images using the calibration parameters. In order to evaluate the performance of Odometry or SLAM systems, we employed the rpg trajectory evaluation toolbox [58].

3.7 Summary

In this chapter, we presented a comprehensive dataset captured using the Maverick MMS, designed specifically for mapping applications in challenging urban environments. The dataset encompasses synchronized data from the Ladybug 5 panoramic camera, the tilted Velodyne HDL-32E LiDAR sensor, and the GPS+IMU unit, providing a rich source of information for SLAM and odometry applications. The unique configuration of the Maverick system, with its tilted LiDAR sensor, allows for the detection of tall features and a more comprehensive understanding of the environment. This dataset offers the researchers in the community an opportunity to explore novel approaches for SLAM and

tracking tasks in urban scenarios. Beyond SLAM and odometry, the dataset holds potential for various other applications. The precise ground truth trajectories, generated using RTK positioning, facilitate accurate evaluation and benchmarking. The dataset's diversity, captured in different environments, enables testing and evaluation in a wide range of scenarios. Furthermore, the dataset can be utilized for depth estimation, object detection and tracking, place recognition, and other computer vision tasks. Its comprehensive nature and the availability of synchronized sensor data make it a valuable resource for researchers in these areas.

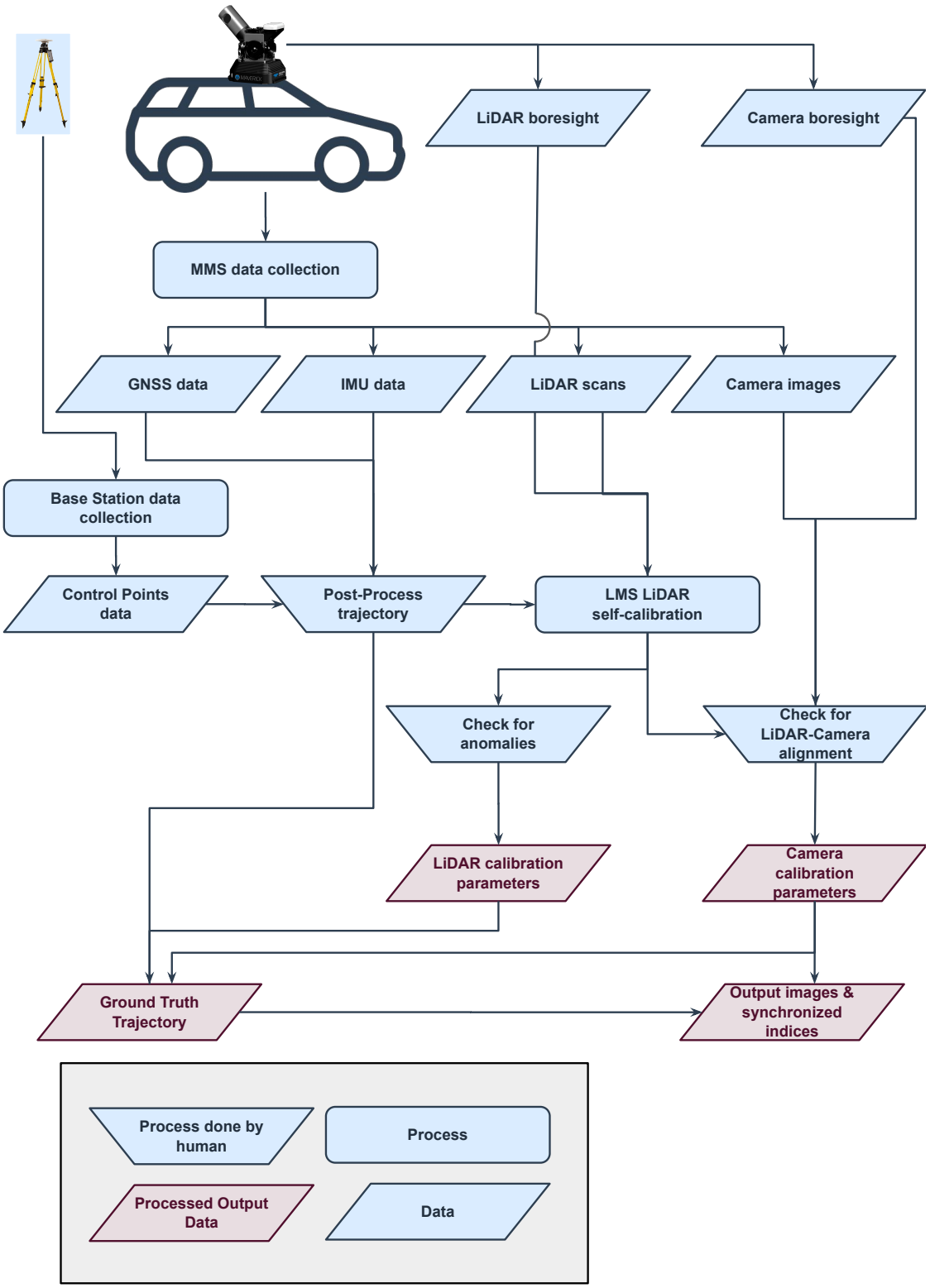


Figure 3.4: Workflow of data generation.

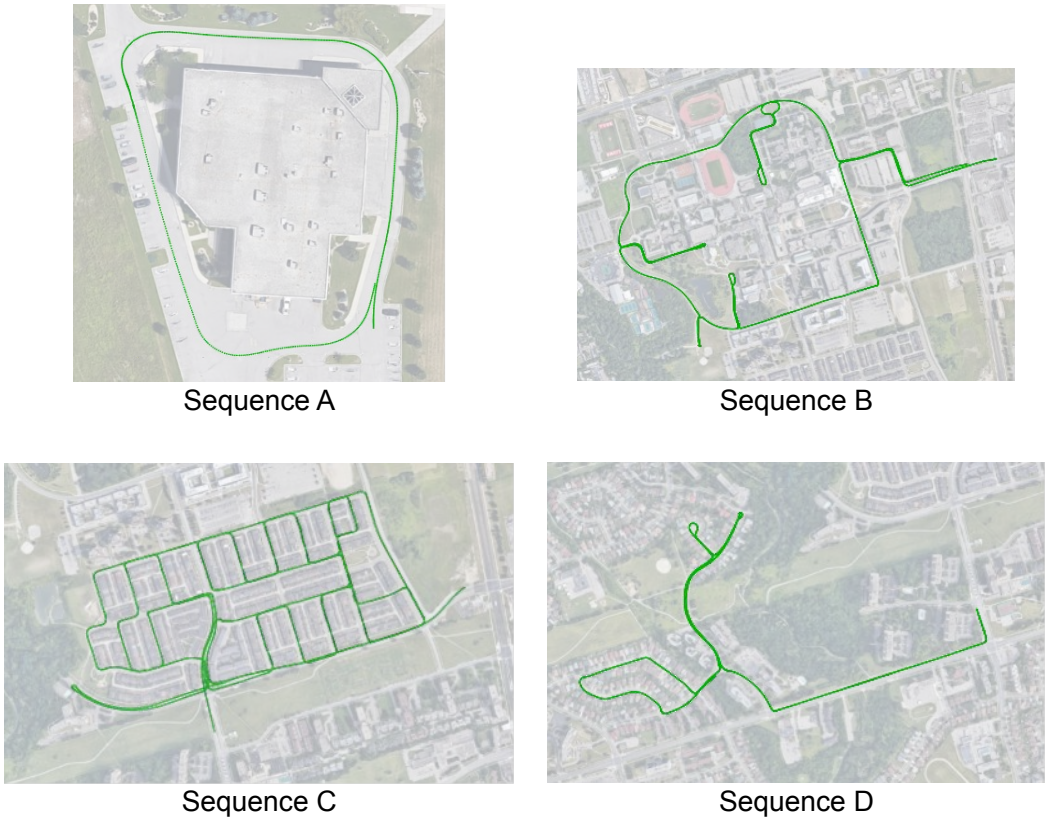


Figure 3.5: YUTO MMS Dataset routes.

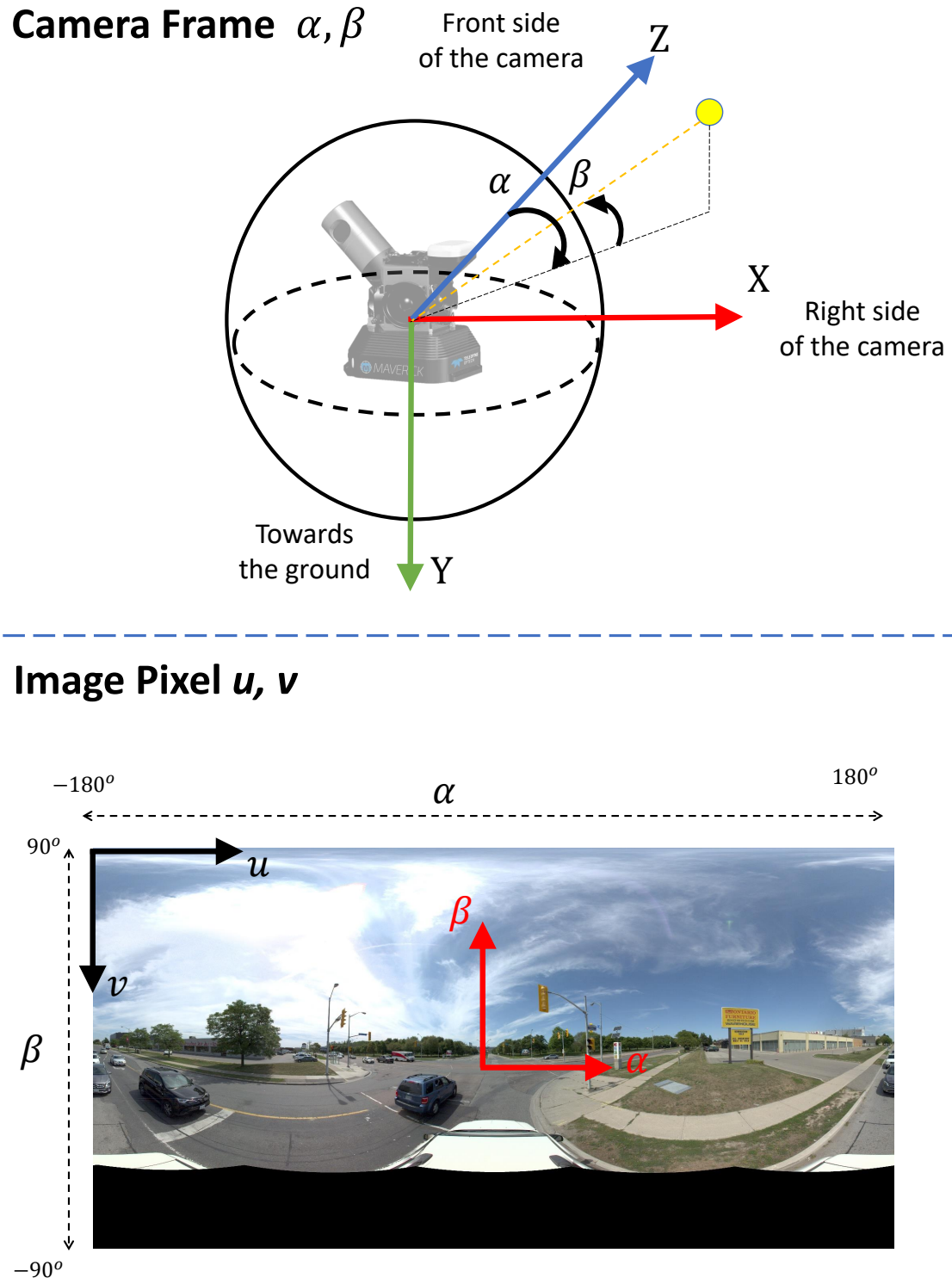


Figure 3.6: Various coordinate systems representation.

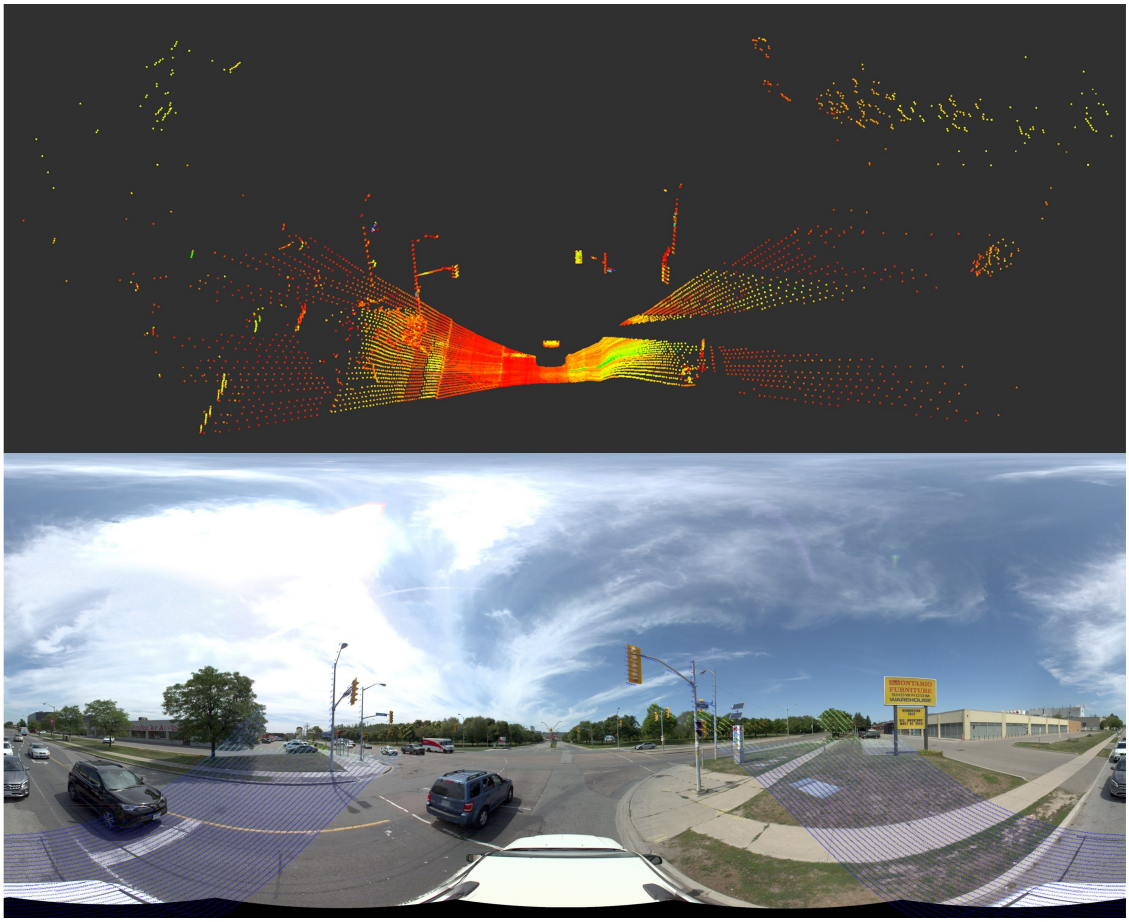


Figure 3.7: The 3D LiDAR points (up) and the points projected on the corresponding panoramic image (down).

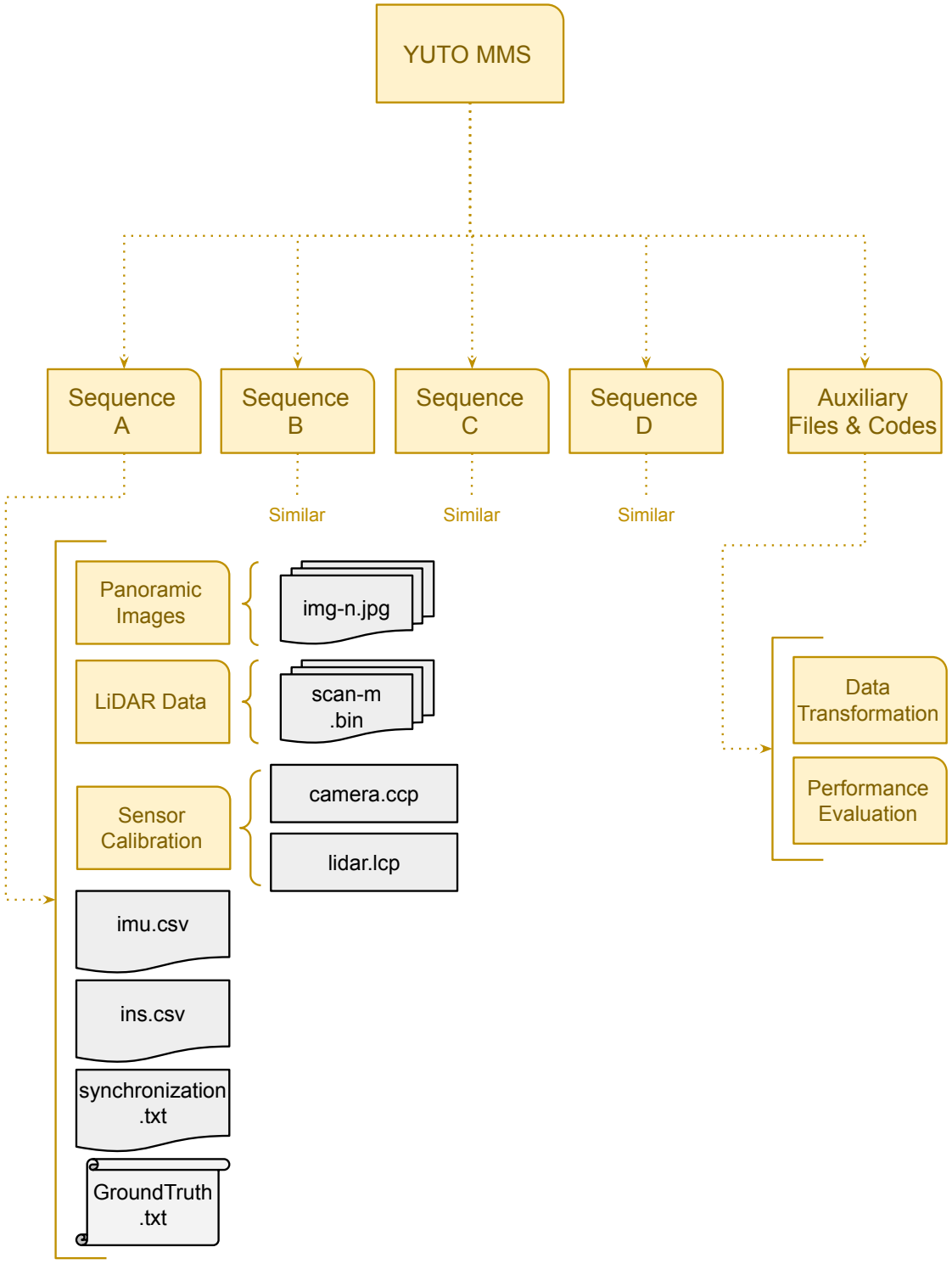


Figure 3.8: YUTO MMS dataset directory structure.

Chapter 4

HDPV-SLAM

In this chapter, we provide a detailed account of the creation of the HDPV-SLAM system, focusing on its two major contributions: Depth Estimation and Depth Association. Before delving into these specific modules, we first provide an overview of the base work upon which the HDPV-SLAM system is built. This serves as a foundation for understanding the subsequent advancements and enhancements. Following the introduction of the base work, we proceed to present each module of the HDPV-SLAM system in greater detail. These modules include SLAM input (Section 4.3), depth estimation (Section 4.4), tracking (Section 4.5), mapping (Section 4.6), depth association (Section 4.7), and loop closing (Section 4.8). By discussing these modules individually, we aim to provide a comprehensive understanding of the HDPV-SLAM system's architecture and functionality. It is worth noting that in line with the goal presented in Chapter 1, our SLAM system primarily focuses on localization, as we have ground truth data available for result evaluation in our dataset. However, our dataset lack ground truth data for evaluating

the mapping results.

The development of a novel SLAM system and its contribution to the academic landscape is inherently a collaborative effort, and my primary focus in this thesis was the creation of the HDPV-SLAM system and the conceptualization of two supplementary modules. Within this scope, Depth Association was conceived solely by me, encompassing its design, coding, optimization, and implementation. In the case of Depth Estimation, it was a collaborative endeavor with fellow AUSMLab colleagues, where I initiated the problem definition, conducted quality assessments of estimations, assisted in parameter optimization, and participated in code debugging.

4.1 Base Work

Range-augmented Panoramic VS (RPV-SLAM) [1] serves as the foundation for our proposed HDPV-SLAM framework. This in-house system has been developed by Dr. Kang and his team at AUSMLab, York University. In the field of panoramic VS, previous systems lacked metrically-scaled trajectories. However, RPV-SLAM addressed this limitation by introducing metrically scaled trajectories. The main contributions of RPV-SLAM can be summarized as follows:

- Firstly, it involved the creation of range maps by projecting LiDAR data onto the panoramic image and subsequently densifying them using bi-interpolation. Notably, the resulting depth information is visually represented as a distinctive rainbow region (overlapping region) within the final image. We will discuss this in

Section 4.4.1.

- Secondly, RPV-SLAM effectively utilized visual features both with and without augmented range information.

- Lastly, RPV-SLAM achieved highly accurate and metrically-scaled results.

To realize these contributions, RPV-SLAM extended the capabilities of ORB-SLAM2 [21], an RGB-D SLAM system, in a straightforward manner. The approach involved densifying the corresponding LiDAR point cloud for each panoramic image, thereby creating a D channel for the RGB image. By leveraging this RGB-D image, RPV-SLAM seamlessly integrated it with the ORB-SLAM2 [21] framework. ORB-SLAM2 is a keyframe-based SLAM system that selectively creates 3D map points in specific frames (known as keyframes) and utilizes these map points for tracking during non-keyframe frames.

To assess the effectiveness of RPV-SLAM, a comparative evaluation was conducted against Google Cartographer [32], a LiDAR-inertial SLAM system that employed the same LiDAR data but with the additional inclusion of an IMU sensor. This comparison served as a benchmark for highlighting the advantages and superior performance of RPV-SLAM in the context of panoramic VS systems.

4.2 System Overview

The HDPV-SLAM system follows a well-defined process to achieve simultaneous localization and mapping. See Figure 4.1 to view the schematic workflow. First, The

raw inputs from the SLAM system, including sensor data from LiDAR and panoramic images, undergo preprocessing and transformations. This step ensures that all necessary data transformations and calibration are applied to the inputs. Using the processed inputs, the system creates pairs of RGB images and sparse depth maps obtained from the LiDAR data. These pairs serve as the primary input for subsequent modules. The panoramic images are processed to extract ORB features. ORB features are widely used for efficient feature matching and tracking in SLAM systems. In the next step, The system utilizes the depth estimation module to generate dense depth maps based on the sparse depth maps and panoramic images. This module leverages innovative DL techniques to densify the sparse depth, improving the accuracy of depth estimation on visual features.

The tracking module is responsible for tracking the ORB features in the existing map points. By continually matching and tracking the features, the system can estimate the camera pose and track its position in the environment. Meanwhile, the mapping module is responsible for creating new map points as needed. It ensures that the system dynamically updates the map to incorporate new visual information and maintain an accurate representation of the environment. The depth association module plays a crucial role in refining the map point positions based on better depth observations. It optimally combines depth information derived from both the dense depth estimation and triangulation to improve the accuracy of the map points. And finally, the loop closing module aims to detect loop closures, where the system revisits a previously visited location. By identifying loops, the system can correct accumulated trajectory errors and enhance the overall accuracy of the SLAM solution.

4.3 SLAM Input

In the first stage, an RGB-D image is created by combining a panoramic RGB image and its corresponding LiDAR point cloud. The image channel D is created by projecting LiDAR data onto the image plane. Using calibration parameters between the camera and the LiDAR sensor, this task is completed in Section 3.5.4.

Then, for each RGB-D image in the capturing order, frame F_t is created, where $t \in \{1,2,3,...,n\}$ and n is the total number of RGB-D images. $\mathbf{F}$ is the array of all frames. Each $F_t \in \mathbf{F}$ contains RGB image $\mathbf{I}_t \in \mathbb{R}^{w \times h \times 3}$ of the width w and the height h and a corresponding projected sparse LiDAR $\mathbf{D}_t^s \in \mathbb{R}^{w \times h}$. In Section 4.4, we describe methods to densify $\mathbf{D}_t^s$ and get $\hat{\mathbf{D}}_t \in \mathbb{R}^{w \times h}$. Also, a frame F_t contains O_t which is the array of the ORB features extracted from $\mathbf{I}_t$ using the ORB feature extractor. For each $o_t^i \in O_t$ we have $o_t^i \in \{(u,v)|u,v \in \mathbb{W}, 0 \leq u < w, 0 \leq v < h\}$where $i \in \{1,2,3,...,m_t\}$ and m_t is the total number of ORB features of I_t.

The output of the SLAM system is the array of all the camera positions J and the set of all the map points ρ. Each $j_t \in \mathbb{R}^3 \times SO(3)$ in J is the position and the orientation of the camera at time t in world coordinate, and each $p \in \mathbb{R}^3$ in ρ shows the position of a map point in the world coordinate. We define subsets of ρ, for each t as we create map points using F_t. The relation between ρ_t for different t is $\{\} = \rho_0 \subset \rho_1 \subseteq \rho_2 \subseteq ...\rho_{n-1} \subseteq \rho_n = \rho$. Each $\rho_t \subset \rho$ is updated with the new map points created from F_t each time a new frame F_t comes in. And when $t = n$, we have $\rho_t = \rho$.

4.4 Depth Estimation Module

First, we present a detailed description of the method employed in our base work, RPV-SLAM [29], which serves as the foundation for our research. Subsequently, we proceed to discuss our novel work, introducing PanoDARS.

4.4.1 Bi-Interpolation

In light of the replacement of the traditional module with our deep learning-based module in the foundational structure, this section expounds on the fundamental procedures undertaken to densify LiDAR points and attain a more comprehensive depth map using bi-interpolation. Later, we aim to compare the performance of our proposed deep learning module with bi-interpolation approach in Chapter 5, shedding light on the key steps that contribute to the enhancement of depth estimation in our system.

We use the extrinsic parameters matrix between the panoramic camera and the LiDAR sensor in the MMS. Then, using Equations 3.1, 3.2, 3.4, and 3.6 from Chapter 3, we can project the 3D points on the corresponding panoramic image. For the set of all the LiDAR points projected at frame t, we use ${}^{l}\varphi_t \subset \phi$. [1] For each ${}^{l}\varphi_t^i \in {}^{l}\varphi_t$ at time t we define a function ${}^{l}\Gamma_t$ returning its range value.

$$ {}^{l}\Gamma_t({}^{l}\varphi_t^i) = r_t^i \tag{4.1} $$

Where i points to the i-th LiDAR point in 3D from the panoramic image.

[1] Please refer to Equation 3.7 for the definition of ϕ.

Dense Depth Map Creation

The projected LiDAR points ${}^{l}\varphi_t$ are sparse and limited to some specific region of the panoramic images, and the probability that they can fall on an actual visual feature for the use of an RGB-D SLAM is low. But we know that the pixels close to ${}^{l}\varphi_t$ in pixel coordinate space probably have similar depth values to their neighboring pixels. One of the most simple ways to estimate the depth for pixels close to pixels in ${}^{l}\varphi_t$, can be averaging range values of these points.

We divide all the pixels in the pixel coordinate space into three partitions. We have defined one of them before; the set of all the pixels that map to a specific LiDAR point in 3D, ${}^{l}\varphi_t$. The second one is ${}^{c}\varphi_t$ which is the set of all the pixels that LiDAR does not cover but are close to pixels in ${}^{l}\varphi_t$. And ${}^{f}\varphi_t$ represents the set of all the pixels that LiDAR does not cover and are far from ${}^{l}\varphi_t$.

$$ {}^{l}\varphi_t \ \cup \ {}^{c}\varphi_t \ \cup \ {}^{f}\varphi_t \ = \ \phi \tag{4.2} $$

We assume pixels in ${}^{c}\varphi_t$ probably have similar a depth to their neighboring pixels from ${}^{l}\varphi_t$. For using this assumption, we perform a triangulation on ${}^{l}\varphi_t$ (Creating triangles using all the pixels that the depth value is available from LiDAR). Given the multitude of available triangulation methods for a given point set, it is imperative to select an optimal approach that minimizes the formation of poorly shaped triangles. Among these methods, the Delaunay triangulation has consistently demonstrated its robustness and widespread

utility. This technique effectively connects neighboring points within a Voronoi diagram, thereby generating a well-suited triangulation. For the depth value of each ${}^{c}\varphi_t^j \in {}^{c}\varphi_t$ that falls into a triangle with vertices $({}^{l}\varphi_t^{i_1}, {}^{l}\varphi_t^{i_2}, {}^{l}\varphi_t^{i_3})$, we can use the range value of the vertices of the triangle $(r_t^{i_1}, r_t^{i_2}, r_t^{i_3})$ to estimate the depth of it.

$$ {}^{c}\Gamma_t({}^{c}\varphi_t^j) = w_1 \cdot r_t^{i_1} + w_2 \cdot r_t^{i_2} + w_3 \cdot r_t^{i_3} \tag{4.3} $$

Where w_1, w_2, and w_3 are weights for the linear barycentric interpolation.

With this description, for each $\varphi_t^i \in \phi$, we define the unified depth function:

$$ \Gamma_t(\varphi_t^i) = \begin{cases} {}^{l}\Gamma_t(\varphi_t^i) & \varphi_t^i \in {}^{l}\varphi_t \\ {}^{c}\Gamma_t(\varphi_t^i) & \varphi_t^i \in {}^{c}\varphi_t \\ Null & \varphi_t^i \in {}^{f}\varphi_t \end{cases} \tag{4.4} $$

As this function is defined on the whole pixel coordinate space (ϕ), we add it as an additional channel to the RGB panoramic image to result in an RGB-D panoramic image.

4.4.2 PanoDARS

As explained earlier, the depth estimation module aims to produce a dense depth map $\hat{\mathbf{D}}_t$ by taking an input RGB image $\mathbf{I}_t$ at the time t and a corresponding sparse LiDAR

$\mathbf{D}_t^s$. This module enables increasing the number of ORB visual features augmented with depth in the overlapping region (OR_t) between the panoramic imagery and the LiDAR point clouds. To do this, a new version of DARS [49] for panoramic depth estimation, called PanoDARS, is proposed (see Figure 4.2). DARS [49] was originally designed for adaptive depth refinement on perspective images, while PanoDARS has to adapt the models pre-trained on perspective images to panoramic ones. However, depth estimation on panoramic images presents some obstacles. First, because there is no publicly accessible standard dataset for panoramic depth estimation, direct supervised models cannot be used. Second, self-supervised methods must be improved by non-metrically scaled estimations and are notably difficult for panoramic geometry. Thus, the suggested solution is to adapt a pre-trained depth estimation baseline over perspective images to our panoramic dataset. The pre-trained depth estimation baseline is Monodepth2 [47] pre-trained on KITTI, and it is used for monocular perspective images at inference time.

PanoDARS consists of two stages, correction and optimization. In [49], the depth maps are divided into three horizontal slices, with the correction values in each slice being independently calculated. Since LiDAR sparsity patterns differ between our dataset and the datasets used in DARS, and also the overlapping region shape is different, the proposed method omits slicing during the correction phase. Moreover, PanoDARS only estimates depth in I_t regions where sparse depth is available nearby (OR_t, where $OR_t = {}^{l}\varphi_t \cup {}^{c}\varphi_t$, as shown in the rainbow-colored image on the left of Figure 4.3).

In the first stage, a correction value $\delta d_t^s \in \Delta\mathbf{D}_t^s$ between each valid pixel in the sparse depth map $d_t^s \in \mathbf{D}_t^s$ and its correspondence $d_t \in \mathbf{D}_t$ is calculated using $\delta d_t^s = d_t - d_t^s$, where

$\mathbf{D}_t$ is the predicted depth from the modified baseline network explained in the following paragraphs.

Then, an interpolation function $Q : \mathbb{R}^2 \mapsto \mathbb{R}$ based on Delaunay triangulation [59] is leveraged to obtain a dense correction map $\Delta\mathbf{D}_t = Q(\Delta\mathbf{D}_t^s)$. Finally, the corrected depth map $\hat{\mathbf{D}}_t = \mathbf{D}_t + \Delta\mathbf{D}_t$ is calculated. $\hat{\mathbf{D}}_t$ is a sufficiently accurate initial value for the second stage.

In the second stage, while the pre-trained weights are fixed, some learnable auxiliary parameters are applied to intermediate features in the baseline. By optimizing those parameters, the predicted depth is refined. Therefore, the overall performance of PanoDARS is as follows.

Given an input RGB image $\mathbf{I}_t \in \mathbb{R}^{w\times h\times 3}$, PanoDARS splits the pre-trained baseline $M : \mathbb{R}^{w\times h\times 3} \mapsto \mathbb{R}^{w\times h}$ into a body $G : \mathbb{R}^{w\times h\times 3} \mapsto \mathbb{R}^{a\times b\times c}$ and a head $H : \mathbb{R}^{a\times b\times c} \mapsto \mathbb{R}^{w\times h}$, where a, b, and c are respectively width, height, and number of channels of the intermediate feature set $G(\mathbf{I}_t) \in \mathbb{R}^{a\times b\times c}$. The auxiliary parameters, scales $\mathbf{S} \in \mathbb{R}^c$ and biases $\mathbf{B} \in \mathbb{R}^c$ are applied on $G(I_t)$, and the depth $\mathbf{D}_t = H(\mathbf{S} \bigotimes G(\mathbf{I}_t) \bigoplus \mathbf{B})$ is predicted, where $\bigotimes$ and $\bigoplus$ represent channel-wise multiplication and addition. Note that if we put S as the identity tensor and B is 0 tensor, $\mathbf{D}_t$ will be exactly the prediction of baseline network on the input of $\mathbf{I}_t$.

Afterwards, the correction module $C : \mathbb{R}^{w\times h} \mapsto \mathbb{R}^{w\times h}$ carries out the first refinement stage on $\mathbf{D}_t$ and returns $\hat{\mathbf{D}}_t = C(\mathbf{D}_t, \mathbf{D}_t^s)$. The auxiliary parameters $\mathbf{X} \in \mathbb{R}^{2c}$, i.e., concatenated channel-wise scales ($\mathbf{S}$) and biases ($\mathbf{B}$), are learnable. Therefore, the following optimization problem can be formulated as $L(\hat{\mathbf{D}}_t(\mathbf{I}_t, \mathbf{X} + \Delta\mathbf{X}), \mathbf{D}_t^s) \rightarrow \min_{\Delta\mathbf{X}}$, where $\Delta\mathbf{X}$ is

the corrections applied on the parameters and L is the cost function given to Particle Swarm Optimizer (PSO) [60]:

$$L = \sqrt{\frac{1}{|{}^{l}\varphi_t|} \sum^{{}^{l}\varphi_t} \left\| log(\hat{d}_t) - log(d_t^s) \right\|} \tag{4.5}$$

4.5 Tracking Module

The tracking module establishes relationships between extracted visual features and existing map points. The tracking module accepts an input of F_t for each t in sequential order at the start of a SLAM run. The matching process within the tracking module will locate matches between ρ_{t-1} and generate an array of matched map points P_t and O_t. P_t has the same size as O_t (which is m_t) and can also contain $Null$ values at certain indices. For each matched pair of o_t^i and a map point from ρ_{t-1}, P_t contains a p_t^i pointing to the map point. $p_t^i = Null$ if o_t^i does not match any map point from ρ_{t-1}. The tracking module can estimate j_t based on these local matches. The system then designates a frame F_t as a keyframe when it determines that the tracking will perform poorly in the future with estimates of j_t based on a limited number of map points.

4.6 Mapping Module

When the tracking module decides F_t to be a keyframe, the mapping module creates new map points using F_t and puts them together with ρ_{t-1} to save all of them in ρ_t. The

module attempts to create a map point for each o_t^i that its corresponding p_t^i is $Null$. If $o_t^i \in OR_t$, it creates a map point using o_t^i, j_t, and $\hat{\mathbf{D}}_t(o_t^i)$. If $o_t^i \notin OR_t$, it will triangulate F_t with the nearest neighboring keyframe F_s to estimate the depth for o_t^i and create the map point with the same set of information. If triangulation is not successful, the module will not create a map point for o_t^i. The mapping module is also responsible for local bundle adjustment, which is accomplished by following [1].

4.7 Depth Association Module

I propose a novel hybrid depth association module for associating optimal depth information with visual features in OR_t. After the tracking module has tracked all visible map points in the current frame F_t and estimated j_t, this depth association module is triggered. The proposed module is executed even if F_t is not a keyframe for augmenting O_t with $\hat{\mathbf{D}}_t$, unlike conventional methods. A key motivation for using non-keyframes in the depth association is to cross-validate the accuracy of tracked map points and update the map points by conducting the depth association optimally. As previously described, a map point can be generated using either the direct assignment of LiDAR-driven depth to a visual feature or the visual feature-based triangulation; it depends on the position of the corresponding feature in the image plane (i.e., being inside OR_t or out of it). Consequently, the depth value associated with a tracked map point may vary. The new method resolves this inconsistency issue by executing an optimal depth association task following established policies.

Algorithm 1 shows the suggested hybrid depth association process. It inputs F_t and the tracked array of map points P_t. For each i, if the matching module has not assigned any map points to o_t^i or if o_t^i is out of OR_t, it skips the i and goes for the next member of O_t. If o_t^i passes these conditions, then a ${}^{new}p_t^i \in \mathbb{R}^3$ will be created using o_t^i, its depth value ($\hat{\mathbf{D}}_t(o_t^i)$), and its depth value and the position and orientation of the camera at time t (j_t). But, sometimes the tracking module can make mistakes in assigning ORB visual features to the corresponding map points. So I have defined a rejection threshold θ (in meters) to prevent big mistakes and their effect on the whole trajectory. If the distance between the current position of p_t^i and the newly calculated ${}^{new}p_t^i$ is bigger than θ, we will not change anything. If not, then we have defined two policies here:

- If p_t^i has been created using triangulation and not modified using any depth map later, the algorithm will change the location of p_t^i to ${}^{new}p_t^i$.

- Or, if the depth value of the visual feature that previously has modified (or created) the position of p_t^i ($\hat{\mathbf{D}}_p(o_p^i)$) is larger than the current depth ($\hat{\mathbf{D}}_t(o_t^i)$), again the algorithm will change the location of p_t^i to ${}^{new}p_t^i$.

Figure 4.3 shows an example of this algorithm for the first policy with a simple scenario where we only have one ORB feature in each frame. And the second policy helps with the precise estimation of the position of the map points. Each time a map point is observed with a smaller depth value, the algorithm recalculates its position based on it.

Algorithm 1 Depth Association Module Algorithm

Input: Current frame F_t and the tracked array of map points P_t

$\theta \leftarrow$The rejection threshold
for Each p_t^i in P_t **do**
  if $p_t^i = Null$ or $o_t^i \notin OR_t$ **then**
    continue
  end if
  $F_p \leftarrow$Last keyframe who created or modified p_t^i
  ${}^{new}p_t^i \leftarrow$Estimated 3D position using o_t^i, $\hat{\mathbf{D}}_t(o_t^i)$, and j_t
  if distance(p_t^i, ${}^{new}p_t^i$) $< \theta$ **then**
    if (p_t^i created using triangulation and not modified after that) or ($\hat{\mathbf{D}}_t(o_t^i) < \hat{\mathbf{D}}_p(o_p^i)$) **then**
      p_t^i's position $\leftarrow {}^{new}p_t^i$
      Update p_t^i in ρ_t and all the previous subsets
    end if
  end if
end for

4.8 Loop Closing Module

Finally, after the loop closing module detects a loop, it performs the global bundle adjustment on the trajectory inside the loop. The global bundle adjustment performs a pose-graph optimization on j_t of the camera in each $F_t \in \{$the subset of keyframes in the loop$\}$. And also it optimizes all of the map points created in those keyframes. For the pose-graph optimization, we have used a similarity transformation.

4.9 Summary

In this chapter, we introduce the innovative HDPV-SLAM architecture and provide a detailed explanation of each module that forms the foundation of the SLAM system. Drawing inspiration from hybrid SLAM systems, we propose a novel approach that combines the conventional SLAM architecture with a DL module specifically designed for depth estimation. Furthermore, we introduce the depth association module, which represents a

novel approach to utilize the available depth information provided by the valuable LiDAR data. By effectively incorporating and associating depth information, our system aims to enhance the overall accuracy and reliability of the SLAM process. Through a comprehensive exploration of each SLAM module, we provide insights into their functionalities and how they contribute to the overall performance of the HDPV-SLAM architecture.

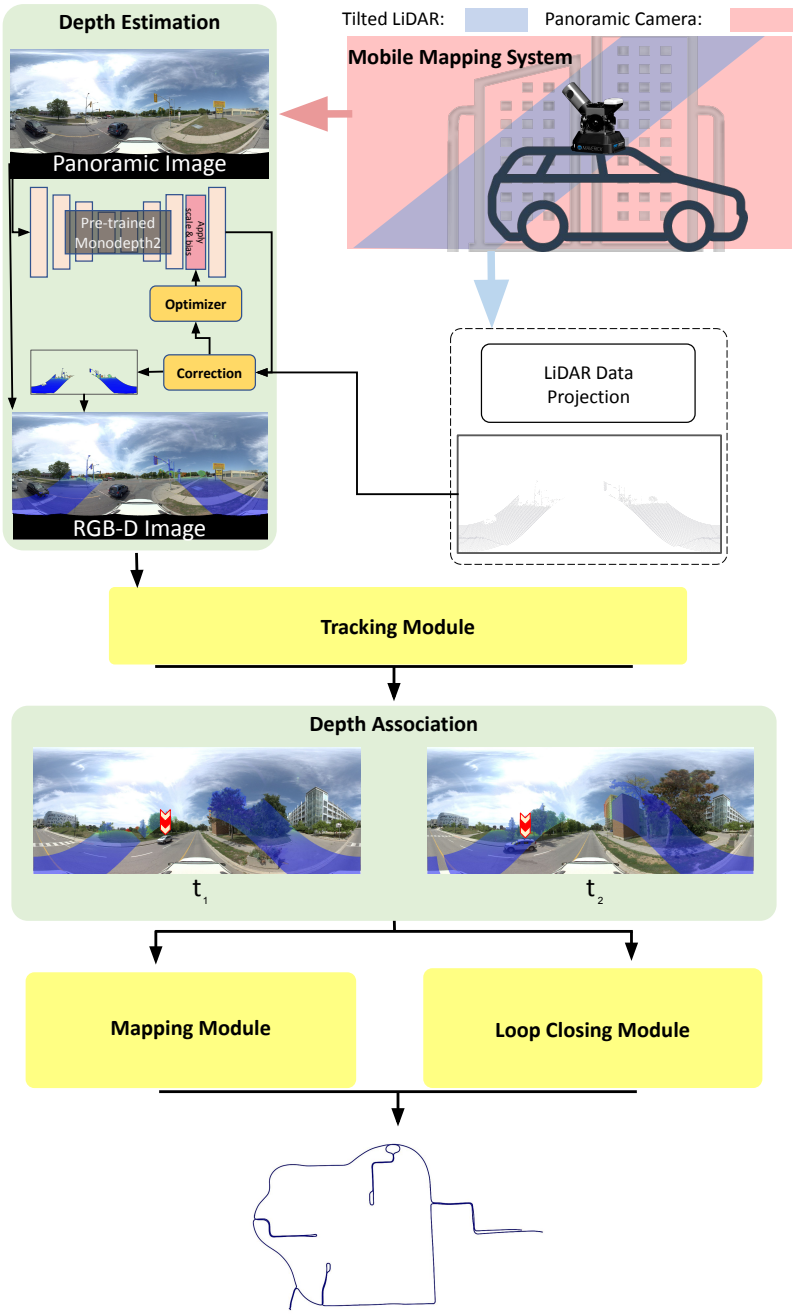


Figure 4.1: A schematic workflow designed for HDPV-SLAM using MMS's panoramic camera and tilted LiDAR scanner. HDPV-SLAM proposes new modules including depth estimation and depth association (shown in green blocks).

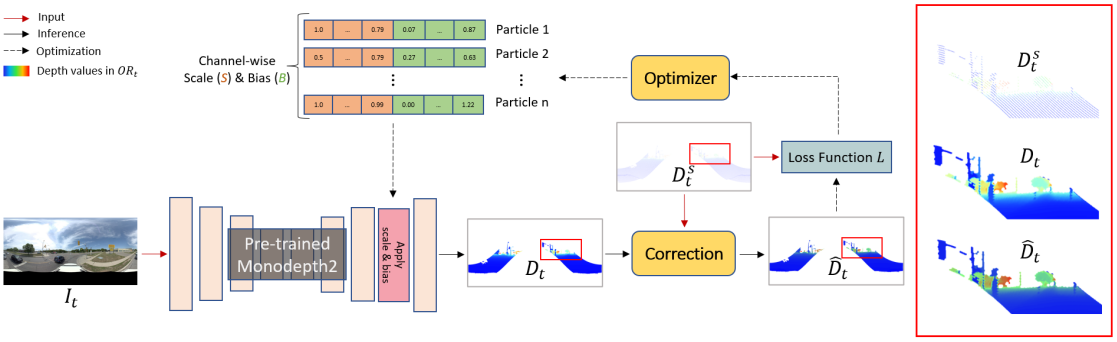


Figure 4.2: A schematic architecture designed for the depth estimation module, called PanoDARS. PanoDARS uses panoramic imagery as a primary source for predicting the depths to generate RGB-D images, while LiDAR data is used as supplementary information to correct the predicted depth. A pre-trained network predicts a depth image $\mathbf{D}_t$ using panoramic imagery as primary source. Given a sparse depth $\mathbf{D}_t^s$, the predicted $\mathbf{D}_t$ is adjusted to the acquired LiDAR point clouds for generating a corrected depth $\hat{\mathbf{D}}_t$. Finally, a refined dense depth is generated by optimizing embedded feature space to minimize the loss between $\mathbf{D}_t^s$ and $\hat{\mathbf{D}}_t$ in an iterative manner during the inference phase.

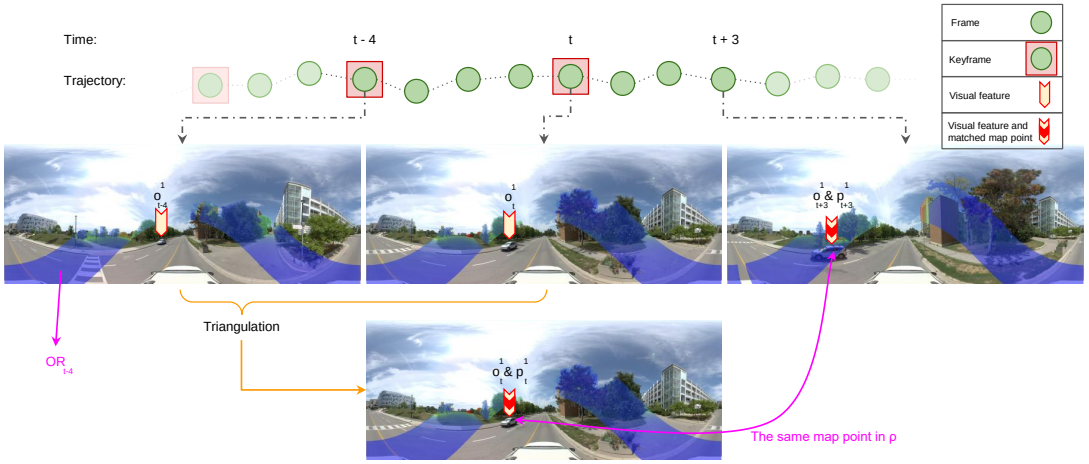


Figure 4.3: A map point p_t^1 is created at time t by triangulating visual features of times $t-4$ and t. This triangulation is because these two visual features are not in OR_{t-4} and OR_t. In F_{t+3}, o_{t+3}^1 points to the middle of the parked car's rooftop and matches with the previously created p_t^1 at time t which is now called p_{t+3}^1. At time $t+3$, o_{t+3}^1 is inside OR_{t+3}, which means the depth data is available. This enables the system to estimate the map points' 3D position more precisely through the depth coming from the depth estimation method.

Chapter 5

Experimental Results and Discussion

In this chapter, we present the experimental evaluation of our proposed HDPV-SLAM system, conducted on the YUTO MMS dataset. The dataset provides a comprehensive real-world scenario for assessing the performance of our system. The preliminary experiments focus on two major components: the Depth Estimation module and the Depth Association module, both of which contribute to the overall robustness and accuracy of the system. We begin by analyzing the effectiveness of the Depth Estimation module, considering the challenges posed by the absence of ground truth for depth estimation evaluation. We then delve into the necessity of the Depth Association module, investigating the rate of keyframe conversion and the creation of map points using both triangulation and depth information. Finally, we conclude this section by presenting the results of the final trajectory, which will provide a comprehensive assessment of the overall performance and efficacy of our HDPV-SLAM system. It is worth to mention that in order to conduct preliminary experiments and thorough ablation studies, we selected sequence B from the

YUTO MMS dataset. Sequence B encompasses diverse environments, including urban areas, residential neighborhoods, and open spaces. By selecting this sequence, we ensure that our experimental analysis captures a broad range of challenges (see Table 3.4) and accurately reflects the performance of our HDPV-SLAM system in various settings.

5.1 Depth Estimation

The evaluation of the Depth Estimation module poses certain challenges due to the absence of ground truth information for assessing its correctness. To provide some quantitative measurements, we calculated the Root mean squared error (RMSE) of the depth estimates obtained from the module, considering points where LiDAR data is available (see Table 5.1). RMSE provides a single value that summarizes how well a model or estimator fits the observed data. Lower RMSE values indicate a closer fit between predicted and actual values, while higher RMSE values suggest greater discrepancies. While I acknowledge that this method has limitations, it serves as a valuable means of comparison within the available constraints.

Figure 5.1 offers a qualitative comparison of a random scene between the Depth Estimation module (PanoDARS) and the bi-interpolation method employed in our base work, RPV-SLAM. In our comparison between the depth maps generated by PanoDARS module and the previous bi-interpolation method, a notable difference is evident, particularly when examining a magnified portions of the two depth maps. Notably, the magnified section of our depth map, shown in Figure 5.1, exhibits sharper details and a better alignment

with the corresponding real RGB image, showcasing the enhanced accuracy and visual quality achieved by our method. On the other hand, the depth map generated by the bi-interpolation method appears blurry and lacks the sharpness observed in our results.

Moreover, the bi-interpolation method introduces noticeable noise in the depth map, as highlighted by the presence of red circles in the comparison (Figure 5.1). In contrast, our proposed module exhibits a higher level of noise reduction and produces more reliable and smoother depth estimates. The absence of noise artifacts in our results further supports the superiority of PanoDARS in generating clean and accurate depth maps.

Table 5.1: RMSE (m) of the two depth estimation methods on LiDAR points

Sequence	Bi-Interpolation	PanoDARS
A	1.01	0.01
B	0.97	0.00
C	0.76	0.01
D	1.20	0.01

5.2 Depth Association

The Depth Association module is a novel addition to our proposed HDPV-SLAM system, and its necessity needs to be examined. Two key aspects will be evaluated in this subsection to assess its impact. Firstly, we will investigate the rate of keyframes per frame, which determines the average number of frames converted into keyframes. Secondly, we will analyze the number of map points generated through triangulation and

compare them with those created through depth estimation. This experiment will enable us to understand the effect of the theta parameter, which governs the depth association process.

To evaluate the necessity of the Depth Association module, we present the results in Table 5.2, which illustrates the number of frames and keyframes in each sequence of the dataset. Our findings reveal that, on average, 23.9% to 27.51% of the frames are converted into keyframes. This implies that in a keyframe-based SLAM system lacking the depth association module (e.g., RPV-SLAM), approximately 72.49% to 76.1% of the depth information produced by the depth estimation module remains unused. Such a substantial amount of unutilized data represents a significant waste of resources and adversely affects the overall accuracy of the system.

Table 5.3 presents the results of the second experiment on Sequence B, comparing the creation of map points with and without the depth provided by the depth estimation module. When we set θ to zero, it is equal to our base work (RPV-SLAM), where approximately 29.62% of the map points rely on depth information (produced by the bi-interpolation method). However, by incorporating the Depth Association module, we maximize the utilization of the available depth information. Increasing the theta parameter in the module leads to a higher percentage of map points being created or revised using the depth information. For instance, at $\theta = 5$ meters, we observe that 44.77% of the map points make use of the depth information obtained from the depth estimation module. This demonstrates the effectiveness of our proposed module in augmenting depth information in map point creation. We selected this range for θ values based on

preliminary test experiments. We observed that more favorable results were consistently achieved with smaller values of θ, motivating our choice of this range.

Table 5.2: Keyframe Conversion Rate and Frame Statistics

	Number of Frames	Average number of Keyframes	Percentage
Sequence A	717	192.2	26.80%
Sequence B	8382	2306.4	27.51%
Sequence C	10778	2576.2	23.90%
Sequence D	4500	1117.8	24.84%

Table 5.3: Comparison of Map Point Creation with and without Depth Information

	θ (meters)	0	1	2	3	4	5
Sequence B	with depth	40,459.60	55979.4	56996.2	59590	61035.2	63858
	without depth	96,123.60	87542.4	83672.8	78763	78470	78771.6
	Percent	29.62%	39.00%	40.52%	43.07%	43.75%	44.77%

5.3 SLAM Trajectory

5.3.1 Evaluation Metrics

Absolute Trajectory Error

Absolute Trajectory Error (ATE) [61] is an evaluation metric in SLAM that quantifies the positional discrepancy between the estimated trajectory and ground truth. It measures the

accuracy of the estimated trajectory by computing the Euclidean distance or RMSE between corresponding pose points. ATE provides a global assessment of localization performance, enabling researchers to compare SLAM algorithms and drive advancements in accurate trajectory estimation. This error generally focuses on the shape of the trajectory and compares it with the ground truth.

Relative Trajectory Error

Relative Trajectory Error (RTE) [62] in SLAM quantifies the positional deviation between estimated and ground truth trajectories at intervals of 100, 200, ..., and 800 meters. It measures the Euclidean distance or RMSE between corresponding pose points to evaluate the accuracy of relative motion estimation. RTE provides valuable insights into the performance of SLAM algorithms in maintaining accurate inter-frame motion estimation at different distances, aiding in the assessment and improvement of localization and mapping systems. This error focuses on local errors in smaller pieces of trajectories.

Relative Rotation Error

Relative Rotation Error (RRE) [62] in SLAM measures the angular deviation between estimated and ground truth relative rotations, often expressed in degrees per meter. It quantifies the rotational accuracy of the system, enabling the evaluation of rotation estimation performance at varying scales. By considering the rotational error relative to the translation distance, RRE provides valuable information on the consistency and precision of rotation estimates in SLAM algorithms. Just like RTE, it focuses on local errors

in smaller pieces of trajectories.

5.3.2 Results

First, an ablation study was conducted on Sequence B of the dataset (see Table 5.4). Then, the proposed method was compared with the baseline methods, i.e., LOAM [31], Google Cartographer [32], ORB-SLAM2 [21], VINS-mono [24], and RPV-SLAM [1] (see Tables 5.5 and 5.6). For all of the comparisons, we have used the evaluation metrics mentioned earlier.

The parameter optimization part of our ablation study aims to find the optimum value for θ in Algorithm 1. The ablation study itself also aims to evaluate the effectiveness of depth association and depth estimation modules. Bi-interpolation described in [1] has been used as a rival to the depth estimation module.

According to Table 5.4, in the absence of the depth association module, i.e., the first three rows, RPV-SLAM and the proposed method show similar performance in terms of both ATE and RTE, yet significantly better than Cartographer. Further, it proves that the bi-interpolation and the depth estimation module have no superiority over each other when there is no depth association module. On the other hand, considering different values of θ, bi-interpolation and depth estimation module illustrate a similar behavior, where increasing θ leads to worse ATE and RTE. Moreover, the best accuracy for both densification methods was obtained using $\theta = 2$. One plausible explanation for this phenomenon could be attributed to the inherent characteristics of the objects present within the scene, including vehicles, trees, and waste containers. These objects exhibit a consis-

tent average diameter of approximately 2 meters, as observed within our dataset.

As Table 5.4 suggests, when the depth association module was utilized (with θ), more accurate results were obtained. In addition, the depth estimation module outperforms bi-interpolation in both ATE and RTE, given identical values for θ. Regardless of selected depth densification methods, $\theta = 2$ shows the best performance in both ATE and RTE. In conclusion, we can attain the best performance when the depth estimation module and $\theta = 2$ are used. It means the depth estimation and depth association module have contributed to the improvement of the SLAM performance in both ATE and RTE.

Tables 5.5 and 5.6 show the results of our best setting (using depth estimation module and depth association module with $\theta = 2$) in comparison with LOAM [31], Google Cartographer [32], ORB-SLAM2 [21], VINS-mono [24], and RPV-SLAM [1]. Please note that RPV-SLAM is the only method that uses the same sensors as our method; all other methods differ in their sensor settings from our SLAM. For example, we specifically tested ORB-SLAM2 using monocular images to highlight the advantages of using panoramic images in visual SLAM. VINS-Mono, a visual-inertial SLAM system, was also tested using monocular images and IMU data from our Maverick MMS. Additionally, we examined LOAM, a well-known LiDAR-based odometry system, and Google Cartographer, a LiDAR-inertial SLAM system that utilizes IMU data for z-direction estimation.

We need to add that ORB-SLAM2 solely utilizes the frontal image of the panoramic camera, resulting in up-to-scale outcomes. To facilitate comparison with the ground truth trajectory, which is metrically scaled, we perform a rigid-body transformation followed

by scaling. All other methods generate metrically scaled results, so we only perform a rigid-body transform and then calculate the errors. Also, VINS-Mono relies exclusively on the frontal image of the panoramic camera in combination with Inertial Measurement Unit data.

The results in tables 5.5 and 5.6 demonstrate that, among the camera-centric SLAM systems, both RPV-SLAM and HDPV-SLAM outperform the state-of-the-art ORB-SLAM2 and VINS-Mono methods. In contrast, the LOAM algorithm failed to generate accurate trajectories due to the usage of a tilted LiDAR, highlighting the importance of incorporating IMU data to indicate the heading direction. Additionally, Google Cartographer exhibited occasional tracking loss and only produced partial trajectories.

Overall, relatively poor results are obtained in residential areas due to unfavorable illumination conditions such as shadows. As expected, the parking lots sequence produced the best ATE performance because of its shorter length and lower scene complexity. Furthermore, the largest improvement in ATE was achieved in Sequence C due to the relatively shorter LiDAR ranges in residential areas.

To conclude, HDPV-SLAM produced the best results in comparison with the other tested methods over all the sequences of YUTO MMS. Although the performance of HDPV-SLAM varies depending on the sequence, the other SLAM systems follow a similar pattern in their performances as well.

Table 5.4: SLAM trajectory results for ablation study on Sequence B.

SLAM Method	**Densification Method**	θ **(m)**	**ATE (m)**	**RTE (%)**	**RRE (°/m)**
RPV-SLAM	Bi-Interpolation	N/A	12.91	1.51	0.0009
Proposed Method	Depth Estimation (PanoDARS)	N/A	12.95	1.64	0.0010
Proposed Method	Bi-Interpolation	1	11.99	1.62	**0.0009**
Proposed Method	Bi-Interpolation	2	9.93	1.43	0.0010
Proposed Method	Bi-Interpolation	3	10.19	1.56	**0.0009**
Proposed Method	Bi-Interpolation	4	13.01	1.63	0.0010
Proposed Method	Bi-Interpolation	5	13.7	1.74	0.0010
Proposed Method	Depth Estimation (PanoDARS)	1	11.28	1.54	**0.0009**
Proposed Method	Depth Estimation (PanoDARS)	2	**9.58**	**1.23**	0.0011
Proposed Method	Depth Estimation (PanoDARS)	3	11.96	1.65	**0.0009**
Proposed Method	Depth Estimation (PanoDARS)	4	12.24	1.52	0.0010
Proposed Method	Depth Estimation (PanoDARS)	5	14.16	1.95	0.0010

Table 5.5: Comparison of positioning accuracy w.r.t. ATE [m]

Sequence	LOAM	Cartographer	ORB-SLAM2	VINS	RPV-SLAM	HDPV-SLAM
A	Fail	4.02	5.89	4.00	1.62	**1.40**
B	Fail	152.23	100.87	86.90	12.91	**9.58**
C	Fail	183.62	155.91	160.77	30.66	**11.93**
D	Fail	58.58	10.67	12.88	5.67	**4.69**

Table 5.6: Comparison of positioning and orientation accuracy w.r.t. RTE [%] and RRE [deg/m]

Sequence		LOAM	Cartographer	ORB-SLAM2	VINS	RPV-SLAM	HDPV-SLAM
A	RTE	Fail	6.79	7.77	4.69	3.93	**3.59**
	RRE	Fail	0.0507	0.0677	0.0410	**0.0040**	0.0041
B	RTE	Fail	15.05	13.77	10.78	3.10	**1.23**
	RRE	Fail	0.0090	0.0099	0.0109	**0.0009**	0.0011
C	RTE	Fail	5.76	4.88	3.99	3.75	**2.77**
	RRE	Fail	0.0133	0.0289	0.0301	0.0057	**0.0034**
D	RTE	Fail	4.65	2.88	3.09	1.35	**0.90**
	RRE	Fail	0.0137	0.0148	0.0178	0.0017	**0.001**

5.4 Discussion

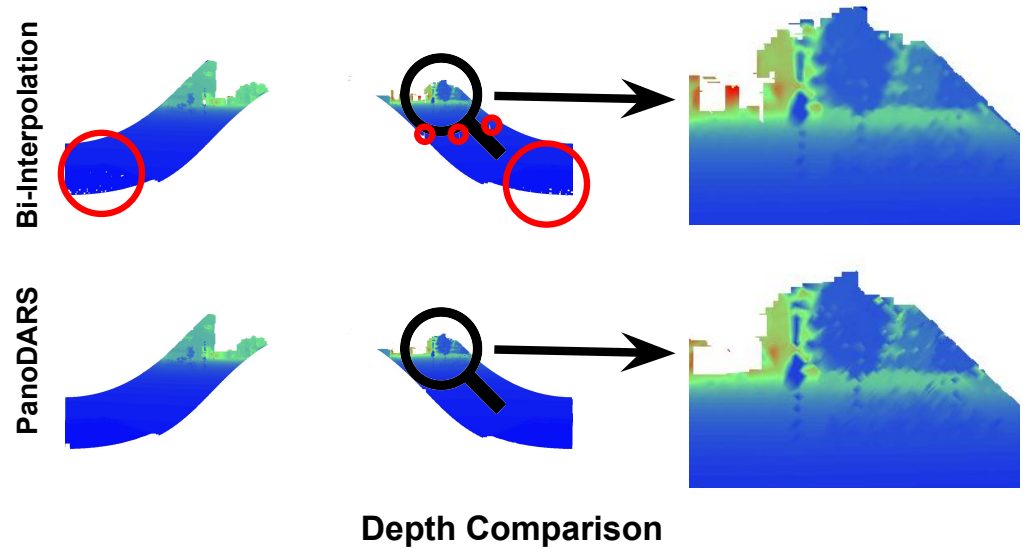
As seen in Table 5.4, the proposed technique outperforms the alternatives in terms of ATE. ATE is a metric that compares the entire trajectory to the ground truth and handles the form matching between them, indicating that the proposed method preserves the shape more effectively than the other methods. Furthermore, in three-quarters of the dataset, the RTE and RRE of the proposed technique are superior to those of the baseline methods, and in one-quarter, they are second best by a slight margin. All of these comparisons demonstrate that the proposed strategy is superior.

While our method has demonstrated impressive performance, surpassing baseline methods, we acknowledge that there remains potential for further enhancement, as outlined in Chapter 6. This conjecture is informed by the notable achievements of SOTA SLAM systems in the KITTI dataset, which approach an RTE accuracy of approximately 0.5%. While it is plausible to strive for similar excellence in our dataset, it is essential

to recognize that distinctions between our dataset and the KITTI dataset may influence achievable outcomes.

5.5 Summary

We presented the novel HDPV-SLAM system that uses a depth estimation module and a depth association module. The depth estimation module generates depth information for panoramic images adapting a perspective model as its baseline. Furthermore, the depth association module leverages all the possible DL-based depth information in keyframes and non-keyframes. However, the proposed method has some limitations as well. We noticed YUTO MMS dataset has dynamic scene objects like cars, buses, people, etc. The SLAM system suffers from these dynamic objects that degrade its performance. As a future work, the visual features coming from these dynamic objects can be filtered out to prevent errors. Another limitation is that the depth association module uses a rule-based algorithm for updating map points. For future investigations, semantic information can be utilized for more precise validation of the matched visual feature and map point pairs.



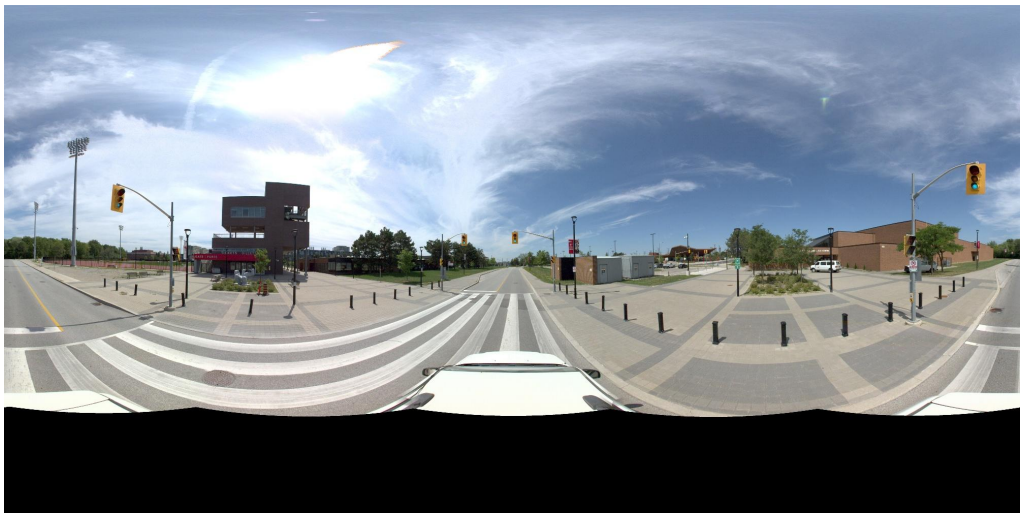


Figure 5.1: Depth estimation module qualitative evaluation. The red circles show the noise artifacts.

Chapter 6

Conclusions

In conclusion, this thesis presented two significant contributions: the YUTO MMS dataset and the HDPV-SLAM system. The YUTO MMS dataset served as a comprehensive resource for researchers in the field of SLAM and odometry, specifically targeting challenging urban environments. Captured using the Maverick Mobile Mapping system, the dataset provided synchronized data from the Ladybug 5 panoramic camera, the tilted Velodyne HDL-32E LiDAR sensor, and the GPS+IMU unit, offering a rich source of information for SLAM and odometry applications. The unique configuration of the Maverick system, with its tilted LiDAR sensor, enabled the detection of tall features and a more comprehensive understanding of the environment.

With its precise ground truth trajectories generated using RTK positioning, the YUTO MMS dataset facilitated accurate evaluation and benchmarking of SLAM and tracking algorithms. Moreover, the dataset's diversity, captured in various urban environments, allowed for testing and evaluation in a wide range of scenarios. Beyond SLAM and

odometry, the dataset held potential for applications such as depth estimation, object detection and tracking, place recognition, and other computer vision tasks. Its comprehensive nature and the availability of synchronized sensor data made it a valuable resource for researchers in these areas.

However, it is important to acknowledge the limitations of the YUTO MMS dataset. The data collection was constrained to outdoor environments with favorable weather conditions, which may not fully represent real-world scenarios with challenging conditions, such as poor lighting, adverse weather conditions, and feature-scarce regions. To address these limitations and enhance the dataset's applicability, future work should focus on expanding the benchmark to include challenging scenarios, incorporating data from indoor environments, and considering environments that combine both indoor and outdoor settings.

In addition to the dataset, this thesis introduced the HDPV-SLAM system, which utilized a depth estimation module and a depth association module to improve the accuracy and robustness of SLAM in panoramic images. The depth estimation module adapted a perspective model as its baseline to generate depth information, while the depth association module leveraged all possible DL-based depth information in both keyframes and non-keyframes. This combination of modules enhanced the system's ability to handle panoramic images and achieve more precise depth estimation.

Despite its promising performance, the HDPV-SLAM system also encountered certain limitations. Dynamic scene objects, such as cars, buses, and people, present challenges for the SLAM system and degrade its performance. Future work could focus on

filtering out visual features originating from dynamic objects to prevent errors and improve overall accuracy. Additionally, incorporating semantic information into the depth association module could enable more precise validation of matched visual feature and map point pairs, further enhancing the system's performance and reliability.

In summary, this research has made significant contributions to the advancement of SLAM and odometry methodologies. The YUTO MMS dataset provided researchers with a comprehensive resource for experimentation, evaluation, and benchmarking in challenging urban environments. The HDPV-SLAM system introduced innovative techniques for depth estimation and association in panoramic images, showcasing the potential for more accurate and robust mapping and localization systems. By identifying limitations and suggesting future research directions, this work opens avenues for further investigation and development in the field of SLAM, paving the way for improved mapping and localization solutions in diverse real-world scenarios.

Bibliography

[1] J. Kang, Y. Zhang, Z. Liu, A. Sit, and G. Sohn, "Rpv-slam: Range-augmented panoramic visual slam for mobile mapping system with panoramic camera and tilted lidar," in *2021 20th International Conference on Advanced Robotics (ICAR)*. IEEE, 2021, pp. 1066–1072.

[2] C. Chen, B. Wang, C. X. Lu, N. Trigoni, and A. Markham, "A survey on deep learning for localization and mapping: Towards the age of spatial machine intelligence," *arXiv preprint arXiv:2006.12567*, 2020.

[3] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 3354–3361.

[4] S. Li, D. Zhang, Y. Xian, B. Li, T. Zhang, and C. Zhong, "Overview of deep learning application on visual slam," *Displays*, p. 102298, 2022.

[5] M. Elhashash, H. Albanwan, and R. Qin, "A review of mobile mapping systems: From sensors to applications," *Sensors*, vol. 22, no. 11, p. 4262, 2022.

[6] O. Al-Bayari, "Road rehabilitation using mobile mapping system and building information model," *Jordan Journal of Earth and Environmental Sciences, 10 (2)*, pp. 113–117, 2019.

[7] Y. Shi, S. Ji, Z. Shi, Y. Duan, and R. Shibasaki, "Gps-supported visual slam with a rigorous sensor model for a panoramic camera in outdoor environments," *Sensors*, vol. 13, no. 1, pp. 119–136, 2012.

[8] S. Ji, Z. Qin, J. Shan, and M. Lu, "Panoramic slam from a multiple fisheye camera rig," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 159, pp. 169–183, 2020.

[9] J. Graeter, A. Wilczynski, and M. Lauer, "Limo: Lidar-monocular visual odometry," in *2018 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2018, pp. 7872–7879.

[10] Y. Zhu, C. Zheng, C. Yuan, X. Huang, and X. Hong, "Camvox: A low-cost and accurate lidar-assisted visual slam system," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 5049–5055.

[11] M. Ahmadi, A. A. Naeini, Z. Arjmandi, Y. Zhang, M. M. Sheikholeslami, and G. Sohn, "Hdpv-slam: Hybrid depth-augmented panoramic visual slam for mobile mapping system with tilted lidar and panoramic visual camera," *arXiv preprint arXiv:2301.11823*, 2023.

[12] K. Yousif, A. Bab-Hadiashar, and R. Hoseinnezhad, "An overview to visual odometry and visual slam: Applications to mobile robotics," *Intelligent Industrial Systems*, vol. 1, no. 4, pp. 289–311, 2015.

[13] C. Chen, H. Zhu, M. Li, and S. You, "A review of visual-inertial simultaneous localization and mapping from filtering-based and optimization-based perspectives," *Robotics*, vol. 7, no. 3, 2018. [Online]. Available: https://www.mdpi.com/2218-6581/7/3/45

[14] Z. Kong and Q. Lu, "A brief review of simultaneous localization and mapping," in *IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*, 2017, pp. 5517–5522.

[15] G. Younes, D. Asmar, E. Shammas, and J. Zelek, "Keyframe-based monocular slam: design, survey, and future directions," *Robotics and Autonomous Systems*, vol. 98, pp. 67–88, 2017.

[16] Y. Chen, Y. Zhou, Q. Lv, and K. K. Deveerasetty, "A review of v-slam," in *2018 IEEE International Conference on Information and Automation (ICIA)*, 2018, pp. 603–608.

[17] N. Yang, R. Wang, and D. Cremers, "Feature-based or direct: An evaluation of monocular visual odometry," *arXiv preprint arXiv:1705.04300*, pp. 1–12, 2017.

[18] J. Engel, T. Schöps, and D. Cremers, "Lsd-slam: Large-scale direct monocular slam," in *European conference on computer vision*. Springer, 2014, pp. 834–849.

[19] C. Forster, M. Pizzoli, and D. Scaramuzza, "Svo: Fast semi-direct monocular visual odometry," in *2014 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2014, pp. 15–22.

[20] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 611–625, 2017.

[21] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE transactions on robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[22] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct EKF-based approach," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Hamburg, Germany: IEEE, Sep. 2015, pp. 298–304. [Online]. Available: http://ieeexplore.ieee.org/document/7353389/

[23] T. Schneider, M. Dymczyk, M. Fehr, K. Egger, S. Lynen, I. Gilitschenski, and R. Siegwart, "Maplab: An Open Framework for Research in Visual-Inertial Mapping and Localization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1418–1425, Jul. 2018. [Online]. Available: http://ieeexplore.ieee.org/document/8276568/

[24] T. Qin, P. Li, and S. Shen, "Vins-mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[25] J. Huai, Y. Zhang, and A. Yilmaz, "Real-time large scale 3d reconstruction by fusing kinect and imu data." *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences*, vol. 2, 2015.

[26] S. Wang, R. Clark, H. Wen, and N. Trigoni, "Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 2043–2050.

[27] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1851–1858.

[28] H. Zhan, C. S. Weerasekera, J.-W. Bian, R. Garg, and I. Reid, "Df-vo: What should be learnt for visual odometry?" *arXiv preprint arXiv:2103.00933*, 2021.

[29] J. Kang, Y. Zhang, Z. Liu, A. Sit, and G. Sohn, "Rpv-slam: Range-augmented panoramic visual slam for mobile mapping system with panoramic camera and tilted lidar," in *2021 20th International Conference on Advanced Robotics (ICAR)*. IEEE, 2021, pp. 1066–1072.

[30] S. Sumikura, M. Shibuya, and K. Sakurada, "Openvslam: A versatile visual slam framework," in *Proceedings of the 27th ACM International Conference on Multimedia*, 2019, pp. 2292–2295.

[31] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time," in *Robotics: Science and Systems*, 2014.

[32] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2D LIDAR SLAM," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. Stockholm, Sweden: IEEE, May 2016, pp. 1271–1278.

[33] E. Mendes, P. Koch, and S. Lacroix, "Icp-based pose-graph slam," in *2016 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)*. IEEE, 2016, pp. 195–200.

[34] J. Behley and C. Stachniss, "Efficient surfel-based slam using 3d laser range data in urban environments." in *Robotics: Science and Systems*, vol. 2018, 2018, p. 59.

[35] X. Chen, A. Milioto, E. Palazzolo, P. Giguere, J. Behley, and C. Stachniss, "Suma++: Efficient lidar-based semantic slam," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 4530–4537.

[36] L. Meyer, M. Smíšek, A. Fontan Villacampa, L. Oliva Maza, D. Medina, M. J. Schuster, F. Steidle, M. Vayugundla, M. G. Müller, B. Rebele *et al.*, "The madmax data set for visual-inertial rover navigation on mars," *Journal of Field Robotics*, vol. 38, no. 6, pp. 833–853, 2021.

[37] J. Jeong, Y. Cho, Y.-S. Shin, H. Roh, and A. Kim, "Complex urban lidar data set," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 6344–6351.

[38] N. Carlevaris-Bianco, A. K. Ushani, and R. M. Eustice, "University of michigan north campus long-term vision and lidar dataset," *The International Journal of Robotics Research*, vol. 35, no. 9, pp. 1023–1035, 2016.

[39] H. Jung, Y. Oto, O. M. Mozos, Y. Iwashita, and R. Kurazume, "Multi-modal panoramic 3d outdoor datasets for place categorization," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2016, pp. 4545–4550.

[40] O. Martínez Mozos, K. Nakashima, H. Jung, Y. Iwashita, and R. Kurazume, "Fukuoka datasets for place categorization," *The International Journal of Robotics Research*, vol. 38, no. 5, pp. 507–517, 2019.

[41] D. Caruso, J. Engel, and D. Cremers, "Large-scale direct slam for omnidirectional cameras," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 141–148.

[42] G. Pandey, J. R. McBride, and R. M. Eustice, "Ford campus vision and lidar data set," *The International Journal of Robotics Research*, vol. 30, no. 13, pp. 1543–1552, 2011.

[43] D. M. Chen, G. Baatz, K. Köser, S. S. Tsai, R. Vedantham, T. Pylvänäinen, K. Roimela, X. Chen, J. Bach, M. Pollefeys *et al.*, "City-scale landmark identification on mobile devices," in *CVPR 2011*. IEEE, 2011, pp. 737–744.

[44] A. Bonarini, W. Burgard, G. Fontana, M. Matteucci, D. G. Sorrenti, J. D. Tardos *et al.*, "Rawseeds: Robotics advancement through web-publishing of sensorial and elaborated extensive data sets," in *In proceedings of IROS*, vol. 6, 2006, p. 93.

[45] C. Godard, O. Mac Aodha, and G. J. Brostow, "Unsupervised monocular depth estimation with left-right consistency," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 270–279.

[46] S. F. Bhat, I. Alhashim, and P. Wonka, "Adabins: Depth estimation using adaptive bins," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 4009–4018.

[47] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow, "Digging into self-supervised monocular depth estimation," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 3828–3838.

[48] J. Watson, O. Mac Aodha, V. Prisacariu, G. Brostow, and M. Firman, "The temporal opportunist: Self-supervised multi-frame monocular depth," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1164–1174.

[49] A. Alizadeh Naeini, M. M. Sheikholeslami, and G. Sohn, "An Adaptive Refinement Scheme for Depth Estimation Networks," *Sensors (Basel, Switzerland)*, vol. 22, no. 24, p. 9755, Dec. 2022.

[50] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-international conference on neural networks*, vol. 4. IEEE, 1995, pp. 1942–1948.

[51] K. Sofiiuk, I. Petrov, O. Barinova, and A. Konushin, "f-brs: Rethinking backpropagating refinement for interactive segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 8623–8632.

[52] S. Cortés, A. Solin, E. Rahtu, and J. Kannala, "Advio: An authentic dataset for visual-inertial odometry," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 419–434.

[53] M. Helmberger, K. Morin, B. Berner, N. Kumar, G. Cioffi, and D. Scaramuzza, "The hilti slam challenge dataset," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7518–7525, 2022.

[54] L. Zhang, M. Helmberger, L. F. T. Fu, D. Wisth, M. Camurri, D. Scaramuzza, and M. Fallon, "Hilti-oxford dataset: A millimeter-accurate benchmark for simultaneous localization and mapping," *IEEE Robotics and Automation Letters*, vol. 8, no. 1, pp. 408–415, 2022.

[55] L. Gao, Y. Liang, J. Yang, S. Wu, C. Wang, J. Chen, and L. Kneip, "Vector: A versatile event-centric benchmark for multi-sensor slam," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 8217–8224, 2022.

[56] Y. Zhang, J. Kang, and G. Sohn, "Pvl-cartographer: Panoramic vision-aided lidar cartographer-based slam for maverick mobile mapping system," *Remote Sensing*, vol. 15, no. 13, 2023. [Online]. Available: https://www.mdpi.com/2072-4292/15/13/3383

[57] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[58] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry," in *IEEE/RSJ Int. Conf. Intell. Robot. Syst. (IROS)*, 2018.

[59] I. Amidror, "Scattered data interpolation methods for electronic imaging systems: a survey," *Journal of Electronic Imaging*, vol. 11, no. 2, p. 157, Apr. 2002. [Online]. Available: http://electronicimaging.spiedigitallibrary.org/article.aspx?doi=10.1117/1.1455013

[60] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4. Perth, WA, Australia: IEEE, 1995, pp. 1942–1948. [Online]. Available: http://ieeexplore.ieee.org/document/488968/

[61] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2012, pp. 573–580, iSSN: 2153-0866.

[62] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *2012 IEEE Conference on Computer Vision and Pattern Recognition*, Jun. 2012, pp. 3354–3361, iSSN: 1063-6919.