

MAPPING AND LOCALIZATION USING LIDAR FIDUCIAL MARKERS

YIBO LIU

A DISSERTATION SUBMITTED TO THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

GRADUATE PROGRAM IN
EARTH AND SPACE SCIENCE
YORK UNIVERSITY
TORONTO, ONTARIO
JANUARY 2025

© YIBO LIU, 2025

Abstract

Light Detection and Ranging (LiDAR) sensors are crucial for autonomous systems, yet the advancements and applications of LiDAR fiducial markers (LFMs) remain limited compared to the widespread use of visual fiducial markers (VFMs) in camera-based applications. Bridging this technological gap is of great significance for robotics and computer vision applications, but challenging due to the unstructured, sparse nature of 3D LiDAR point clouds and the 2D-specific design of previous fiducial marker algorithms. In this dissertation, a novel framework for mapping and localization using LFMs is proposed to benefit a variety of real-world applications, including the collection of 3D assets and training data for point cloud registration, 3D map merging, Augmented Reality (AR), and many more. In particular, this dissertation investigates the development of LFM and the utilization of LFM in mapping and localization from the following three aspects.

First, in response to the absence of an LFM system as convenient and reliable as VFMs, an Intensity Image-based LiDAR Fiducial Marker (IFM) system is developed. The markers are thin, letter-sized sheets of paper or board with patterns compatible with popular VFM systems, without affecting the 3D geometry of the environment. A marker detection method is introduced that locates the 3D fiducials in the point cloud through the intensity image. Then, an algorithm is presented that uses the detected 3D fiducials to estimate the LiDAR 6-degree-of-freedom (6-DOF) pose.

Second, to tackle the limitations of the vanilla IFM method, an improved algorithm for detecting LFMs is proposed. This enhancement extends marker detection from the single-view point cloud to the 3D map and increases the detectable distance of markers, thereby making downstream tasks such as 3D map merging feasible. In particular, a new pipeline is designed to analyze a 3D point cloud from both intensity and geometry perspectives. This approach differs from conventional 3D object detection methods, which rely solely on 3D geometric features and can only detect spatially distinguishable objects.

Third, a novel approach for mapping and localization using LFMs is developed. It processes unordered, low-overlap point clouds by registering them into a single point cloud for mapping and determining their relative poses for localization. Initially, an improved adaptive threshold marker detection method is developed to provide robust detection results when viewpoints among point clouds change dramatically. Subsequently, a maximum a-posteriori (MAP) problem is formulated for registering the multiview point clouds, with a two-level graph framework developed to address it. The first-level graph, constructed as a weighted graph, is designed to efficiently and optimally infer initial values of point cloud poses from the unordered set. The second-level graph is constructed as a factor graph for the global optimization of point cloud poses, marker poses, and marker corner positions. In addition, a new dataset named Livox-3DMatch is collected using the proposed approach and incorporated into the training of the state-of-the-art learning-based multiview point cloud registration methods, resulting in evident improvements across various benchmarks.

Extensive experiments with various LiDAR models in diverse indoor and outdoor scenes demonstrate the effectiveness and superiority of the proposed framework. The framework serves as an efficient, low-cost, and reliable tool for robotics and computer vision applications, including AR, 3D asset and training data collection, degraded scene reconstruction, Global Positioning System (GPS)-denied localization, and 3D map merging.

Dedication

To my wife, son, and parents

Acknowledgments

First, I would like to express my sincere gratitude to my supervisor, Prof. Jinjun Shan. Although the research focus of our group is mainly on control theory, he has generously allowed me the flexibility to explore topics in robotics and computer vision. He has kindly provided me with such extensive support, including resources for study and research and personal life. I also want to thank my committee members, Prof. Armenakis and Prof. Jianguo Wang, as well as the oral exam committee members, Prof. Zhu, Prof. Ping Wang, Prof. Hu, and Prof. Waslander, for their insightful feedback, which strengthened my dissertation. Thanks to my colleagues at SDCNLab—Ti, Marc, Hassan, Samira, Mingfeng, Penghai, Hao, Junjie, Xiaoyu, and Yida—for their kindness and support. I am grateful to my co-authors—Hunter, Amal, Kelly, Andrew, and Robert—for their valuable contributions. I would like to acknowledge the senior researchers—Shiyuan, Adeel, Brian, Guile, Shuo, Han, Yuan, Yang, Binbin, Tongtong, and Bingbing—whose guidance shaped my research perspective. I extend my appreciation to Haiping Wang, Hongpei Yin, Jiahe Cui, Yicong Fu, Jie Li, and Professor Hao Fan for their assistance. Lastly, I am deeply thankful for my parents’ unwavering support and to my wife, Siyu Wu, for her contributions to our family that allowed me to focus on my research. My son, Orion Wu Liu, inspires me every day. This journey was impossible without all of you.

Table of Contents

Abstract	ii
Dedication	iv
Acknowledgments	v
Table of Contents	vi
List of Tables	x
List of Figures	xii
Nomenclature	xvii
List of Acronyms	xxvi
1 Introduction	1
1.1 Motivation	1
1.2 Related Work and Challenges	5
1.2.1 Visual Fiducial Markers	5
1.2.2 LiDAR Fiducial Objects	6
1.2.2.1 LiDAR Fiducial Object Patterns	6

1.2.2.2	LiDAR Fiducial Object Placement	7
1.2.3	LiDAR-based Mapping and Localization	9
1.2.3.1	LiDAR-based SLAM	9
1.2.3.2	Multiview Point Cloud Registration	9
1.2.4	Training Data Collection for Point Cloud Registration	10
1.3	Contributions	11
1.4	Dissertation Organization	13
2	Intensity Image-based LiDAR Fiducial Marker System	15
2.1	Overview	15
2.2	Preliminaries	18
2.2.1	Three-dimensional Transformation	18
2.2.2	Lie Group and Lie Algebra	19
2.2.3	LiDAR Technology	20
2.3	Marker Detection	21
2.3.1	Generation of the Intensity Image	21
2.3.2	Selections of the Angular Resolutions	23
2.3.3	3D Fiducials Estimation	28
2.4	LiDAR Pose Estimation	30
2.5	Experimental Validation	35
2.5.1	Experimental Setup	35
2.5.2	Qualitative Evaluation	36
2.5.3	Quantitative Evaluation	38
2.5.4	Computational Time Analysis	43
2.5.5	Limitations Analysis	43

3	Improvements to Vanilla IFM Localization	46
3.1	Overview	46
3.2	Joint Analysis of Point Clouds from Intensity and Geometry Perspectives .	48
3.2.1	Downsampling Based on 3D Intensity Gradients	48
3.2.2	Spatial Distribution Analysis of Downsampling Result	52
3.2.3	Filtering Out Unwanted Clusters	53
3.3	Marker Localization via Intermediate Plane	56
3.3.1	Motivation for Adopting the Intermediate Plane	57
3.3.2	Utilization of the Intermediate Plane	60
3.4	3D LiDAR Map Construction	62
3.4.1	Livox Mapping	62
3.4.2	Traj LO	63
3.5	Experimental Validation	64
3.5.1	Extending LFM Localization to 3D Maps:	
	Tackling the First Limitation	64
3.5.2	Extending the Valid LFM Localization Range:	
	Addressing the Second Limitation	68
3.5.3	Marker Localization Accuracy Improvement	69
3.5.4	Application and Discussion	70
4	Exploiting LFMs for Mapping and Localization	72
4.1	Overview	72
4.2	Methodology	74
4.2.1	Adaptive Threshold LFM Detection	74
4.2.2	Problem Formulation	76
4.2.3	First-level Graph	77

4.2.4	Second-level Graph	80
4.3	Livox-3DMatch Dataset	83
4.4	Experimental Validation	86
4.4.1	Experimental Setup	86
4.4.2	Evaluation of the Adaptive Marker Detection Method	86
4.4.3	Evaluation of Point Cloud Registration Accuracy	88
4.4.4	Application 1: 3D Asset Collection from Sparse Scans	93
4.4.5	Application 2: Training Data Collection and Enhancement of Existing Learning-Based Methods	97
4.4.6	Application 3: Reconstructing a Degraded Scene	99
4.4.7	Application 4: Localization in a GPS-denied Environment	102
4.4.8	Application 5: 3D Map Merging	105
4.4.9	Ablation Studies	107
4.4.10	Limitations	109
5	Conclusion and Future Work	110
5.1	Conclusion	110
5.2	Future Work	112
	References	115
	List of Publications	126

List of Tables

2.1	Pose estimation accuracy of the IFM system with different Euler angles. . .	40
2.2	Comparison of the IFM system and LiDARTag.	42
3.1	Quantitative evaluation of marker localization on 3D maps.	65
3.2	Quantitative evaluation of marker localization at various distances.	69
3.3	Comparison of the proposed approach, the vanilla IFM, and AprilTag3 [1] with respect to pose estimation accuracy.	70
4.1	Demonstration of the necessity of the proposed adaptive marker detection algorithm	88
4.2	Quantitative comparison with MDGD [2], SGHR [3], SE3ET [4], GeoTrans [5], and Teaser++ [6] on our dataset.	92
4.3	Comparison with LOAM methods regarding reconstruction	97
4.4	Quantitative evaluation of the enhancement of SGHR [3] and MDGD [2] due to the addition of Livox-3DMatch to the training.	99
4.5	Comparison with SfM-M [7] regarding localization accuracy in a degraded scene.	102
4.6	Comparison with KISS-ICP [8] regarding localization.	104

4.7	Comparison of 3D map merging results between MDGD [2], SGHR [3], and our method.	107
4.8	Ablation studies on the first and second graphs.	108

List of Figures

1.1	Comparison of (a) a typical calibration board, (b)(c) VFM, (d) LiDARTag, and (e) prisms in terms of object patterns and placement.	8
2.1	An illustration of the flexibility and generalizability of the proposed IFM. . .	16
2.2	Comparison of LiDARTag [9] (top) and the proposed IFM (bottom).	18
2.3	The schematic diagram of the working principle of LiDAR.	21
2.4	An illustration of the coordinate systems and notations.	22
2.5	The intensity images generated under different angular resolution settings. . .	25
2.6	Sampling patterns of the mechanical LiDAR and solid-state LiDAR, with sampling points represented by red scatter plots.	26
2.7	An illustration of image preprocessing in the system.	27
2.8	An illustration of the algorithm to estimate the 3D coordinates of a detected but unscanned 2D feature point.	29
2.9	The side view of α . ϕ_d , ϕ_k , and ϕ_u are the inclinations of $\mathbf{p}_d$, $\mathbf{p}_k$, and $\mathbf{p}_u$, respectively.	30
2.10	An illustration of the experimental setup.	36
2.11	Marker detection results on the preprocessed intensity images.	37
2.12	Pose estimation accuracy of the IFM system and the AprilTag 3 system at different distances.	39

2.13	An illustration of the limitation of spherical projection for 3D maps. This map is constructed using Traj LO [10].	44
3.1	An overview of the improvements to the vanilla IFM.	47
3.2	The example used to explain the design purpose and result of each step. . .	49
3.3	The effect of applying downsampling from the intensity perspective. . . .	51
3.4	A diagram to illustrate the design of a typical square fiducial marker [1]. .	52
3.5	The effect of clustering on the downsampling result.	53
3.6	A diagram of the possible OBB size for a given marker.	54
3.7	The effect of filtering out the unwanted clusters.	56
3.8	The result of extracting points falling into the preserved OBBs from the raw point cloud.	58
3.9	An illustration of how the intermediate plane helps solve the occlusion issue.	59
3.10	An illustration of how the intermediate plane helps solve the occlusion issue.	59
3.11	The fiducial marker localization result after applying the intermediate plane method. The vertices of the localized fiducial marker are rendered red. . . .	61
3.12	An illustration of the potential rotational ambiguity issue of $\mathbf{T}_{OBB}$	62
3.13	The LiDAR scans the indoor parking lot from right to left, moving rapidly when scanning marker ID 0.	66
3.14	Traj LO [10] is utilized to create this 3D map.	67
3.15	Comparison of the intensity images generated by the vanilla IFM and the proposed method at different distances.	68
4.1	An overview of the proposed framework for mapping and localization using LFMs.	73
4.2	The raw intensity image binarized with different threshold values.	74

4.3	An illustration of the first-level graph. After applying the proposed adaptive marker detection to all scans, an exhaustive weighted graph is constructed, with the scans and markers as nodes and the point-to-point errors as edge weights. The aim is to derive the relative pose between each non-anchor scan and the anchor scan with optimal accuracy. However, for a given non-anchor scan, such as f_2 in this simple case, there may be multiple possible paths in the exhaustive graph leading to the anchor scan (f_1). Thus, Dijkstra's algorithm [11] is employed to find the optimal path with the minimum accumulation of point-to-point errors (weights).	79
4.4	The procedures for formulating the second-level graph. The variable nodes are represented by circles and the factor nodes are represented by squares. In Stage One, when a marker is detected in a scan, six types of nodes are added to the graph, including (1) scan pose in $\{G\}$, (2) marker pose in $\{G\}$, (3) corner positions in $\{G\}$, (4) marker pose from $\{M\}$ to $\{G\}$, (5) corner positions in $\{F\}$, and (6) corner positions in $\{M\}$, along with their corresponding edges. In Stage Two, all the marker detection results are traversed, and the operation from Stage One is repeated for each detected marker. In Stage Three, a prior factor connecting the anchor scan is added, along with factors representing the relative poses between the anchor scan and non-anchor scans.	82

4.5	Comparison of 3DMatch and Livox-3DMatch. (a): A random sample from 3DMatch. The point cloud sampled by an RGB-D camera has a regular pattern and less noise. (b): A random sample from Livox-3DMatch. The Livox LiDAR point cloud has an irregular pattern and more noise. (c): An example of an outdoor scene in Livox-3DMatch. (d): A selectively sampled challenging case in which the overlapping regions are mainly planes.	85
4.6	Setup for testing the adaptive threshold marker detection algorithm: (a) between groups of buildings, (b) in an open outdoor area, and (c) in a large indoor parking lot.	87
4.7	A comparison with SOTA methods, including MDGD [2], SGHR [3], SE3ET [4], GeoTrans [5], and Teaser++ [6], on ten scenes. The scenes include the office (1-3), the meeting room (4), the lounge (5,6), the kitchen (7), the office building (8,9), and the thicket (10). Each scene consists of three scans, except scenes 2 and 9, which are composed of two scans.	91
4.8	An illustration of the experimental setup and a visual comparison of the proposed method against the SOTA methods (MDGD [2], SGHR [3], SE3ET [4], GeoTrans [5], and Teaser++ [6]) regarding instance reconstruction from sparse scans.	96
4.9	Visual comparison of the instance reconstruction. From top to bottom: ground truth, Ours, Traj LO [10], Livox Mapping [12], and LOAM Livox [13].	97
4.10	An illustration of the scenes in Livox-3DMatch.	98
4.11	An illustration of the experimental setup and a visual comparison of the proposed method against the SOTA methods (MDGD [2], SGHR [3], SE3ET [4], GeoTrans [5], Teaser++ [6], and SfM-M [7]) in a degraded scene.	101

4.12	A comparison of the sensor trajectories obtained from the proposed method and SfM-M [7]. G.T. refers to the ground truth.	102
4.13	(a): An illustration of the experimental setup. (b): Comparison of the trajectories given by different methods.	104
4.14	A comparison of the 3D map merging results of MDGD [2], SGHR [3], and our method.	106
4.15	A comparison of the point cloud registration results without the first graph, without the second graph, and with both graphs for two cases.	108

Nomenclature

Topology

α	Plane determined by $\mathbf{p_u}$, $\mathbf{p_k}$, and $\mathbf{p_d}$
β	Plane where the marker is located
$\mathbf{P}$	Intermediate plane
$\mathcal{C}_{loam}$	Set of corner features extracted by LOAM
$\mathcal{E}_{kine}$	Kinematics energy
$\mathcal{E}_{marg}$	Marginalization energy
$\mathcal{E}_{reg}$	Registration energy
$\mathcal{I}$	Raw intensity image
$\mathcal{P}'_j$	Set of points on the intermediate plane
$\mathcal{P}_j^G$	Set of points belonging to $\mathcal{P}_l$ transmitted to the origin of $\{W\}$
$\mathcal{P}_{loam}$	Set of sampled data used by LOAM

$\mathcal{P}_{Traj}$	Set of sampled data used by Traj LO
$\mathcal{S}_{loam}$	Set of plane features extracted by LOAM
$\mathcal{Z}$	Set of measurements
$\mathcal{P}_{\mathcal{I}}$	Point set composed of the neighbouring points around $\mathbf{p}_0$
$\mathcal{P}_{\mathcal{L}}$	Set of detected 3D fiducials expressed in $\{L\}$
$\mathcal{P}_{\mathcal{W}}$	Set of detected 3D fiducials expressed in $\{W\}$
$\mathcal{P}_j$	Set of points enclosed within the j -th OBB
$\mathfrak{so}(3)$	Lie algebra associated with $SO(3)$
l	Intersection of α and β
Θ	Set of variables
Θ^*	Set of optimal variables
f_i	LiDAR point cloud acquired at the i -th viewpoint
Q	Queue for saving detected markers
Q_{temp}	Temporary queue for saving detected markers
$SE(3)$	Lie Group
$SO(3)$	Special Orthogonal Group
z_k	Measurement
$\mathbb{R}^{n \times m}$	Set of real numbers of a dimension $n \times m$

Frames of Reference

$\{a\}, \{b\}$	Two random coordinate systems
$\{F\}_i$	Local coordinate system of f_i . $\{F\}_i$ is aligned with $\{L\}$ when the frame is acquired
$\{G\}$	Global/World coordinate system
$\{I\}$	Image coordinate system
$\{L\}$	LiDAR coordinate system
$\{M\}_j$	Marker coordinate system of j -th marker

Functions

$\Gamma()$	Function of marker detection
$\hat{\mathbf{I}}(\mathbf{p})$	Approximated function of intensity with respect to $\mathbf{p}$
$\kappa(\mathbf{p}_i)$	Function to calculate the curvature
$\lceil \cdot \rceil$	Rounding a value to the nearest integer
$\log(\cdot)$	Matrix logarithm
$\mathbf{N}(\mathbf{p}_i)$	Function to calculate the normal
$\mathbf{T}(t)$	Function of time representing the LiDAR trajectory
$\text{diag}()$	Operation for creating a diagonal matrix
$\text{Rot}(\mathbf{T})$	Extracting the rotation matrix from $\mathbf{T}$
$\text{Trans}(\mathbf{T})$	Extracting the translation vector from $\mathbf{T}$
$\ominus$	Straight subtraction for elements

$\Psi()$	Function of binarization
$\tau(t_i)$	Function of time representing the LiDAR trajectory at the i -th time segment
$\vee$	Map operator that finds the unique vector $\xi \in \mathbb{R}^{3 \times 1}$ corresponding to a given skew-symmetric matrix $\log(\mathbf{R}) \in \mathbb{R}^{3 \times 3}$
$\det()$	Determinant of a matrix
$h_k()$	Measurement function
$I(\mathbf{p})$	Function of intensity with respect to $\mathbf{p}$
$Trace()$	Trace of a square matrix
$(\cdot)$	Operation for transmitting 3D points between frames of reference

Scalars

$\bar{I}$	Mean of the intensity values of points in $\mathcal{P}_{\mathcal{I}}$
δ	Step size
λ, λ^*	Threshold for binarization and Optimal threshold for binarization
μ	Ratio
ω	Thickness of the object to which the LiDATag is attached
Θ_a	Angular resolution in the azimuth direction
Θ_h	Angular resolution in the azimuth direction given by the user manual
Θ_i	Angular resolution in the inclination direction
Θ_v	Angular resolution in the inclination direction given by the user manual

a	Side length of the marker
b	Coefficient of $\hat{\mathbf{I}}(\mathbf{p})$
b	Intercept
i	Intensity value
I_h	Height of the intensity image
I_w	Width of the intensity image
l, w, h	Length, width, and height of the OBB
L_{OBB}	Cuboid diagonal length of the OBB
P_h	Maximum angular height of the point cloud
P_w	Maximum angular width of the point cloud
Q_l	Length of Q
$Q_{temp,l}$	Length of Q_{temp}
S	Search scope
S_{OBB}	Area of the 2D OBB
t_b	Amplification factor
t_{cor}, t_{sur}	Thresholds for judging corner and plane features
t_L	Marker size of a LiDARTag
t_M	Thickness of the marker
Thr	Threshold for calculating recall

u_k, v_k	2D image coordinates of $\mathbf{u}_k$
u_o	Offset in the azimuth direction
v_o	Offset in the inclination direction
x_d, y_d, z_d	3D coordinates of $\mathbf{p}_d$
x_k, y_k, z_k	3D coordinates of $\mathbf{p}_k$
x_u, y_u, z_u	3D coordinates of $\mathbf{p}_u$
e_{pp}	Point-to-point error
θ, ϕ, r	Sphercial coordinates expressed in the LiDAR coordinate system
θ_d, ϕ_d, r_d	Sphercial coordinates of $\mathbf{p}_d$
θ_k, ϕ_k, r_k	Sphercial coordinates of $\mathbf{p}_k$
θ_u, ϕ_u, r_u	Sphercial coordinates of $\mathbf{p}_u$
u, v	2D image coordinates
x, y, z	3D coordinates
x_L, y_L, z_L	Cartesian coordinates expressed in the LiDAR coordinate system

Vectors

$\mathbf{0}^{1 \times 3}$	Zero vector with the dimension of 1×3
$\mathbf{C}$	Coefficient vector
$\mathbf{E}$	Regression coefficient vector
$\mathbf{f}$	3D coordinates of a fiducial

$\mathbf{I}_{in}$	Response variable vector
$\mathbf{p}^{j,s}$	3D coordinates of the s -th corner of the j -th marker expressed in $\{W\}$
$\mathbf{p}_a, \mathbf{p}_b$	3D coordinates of a point in the coordinate systems $\{a\}$ and $\{b\}$
$\mathbf{p}_L$	An observed point in the 3D point cloud
$\mathbf{q}_j, \mathbf{q}'_j$	Centroid-aligned coordinates
$\mathbf{t}$	Translation vector
$\mathbf{u}$	Projection of $\mathbf{p}_L$ on the image plane
$\mathbf{f}_1, \dots, \mathbf{f}_n$	Detected 3D fiducials expressed in $\{L\}$
$\mathbf{f}'_1, \dots, \mathbf{f}'_n$	Predefined 3D fiducials expressed in $\{W\}$
$\mathbf{O}_L$	Origin of the LiDAR coordinate system
$\mathbf{p}_0$	A given 3D point
$\mathbf{p}_1, \dots, \mathbf{p}_n$	Neighbouring points around $\mathbf{p}_0$
$\mathbf{p}_G$	Point transmitted to the origin of $\{W\}$
$\mathbf{p}_k$	3D point corresponding to $\mathbf{u}_k$
$\mathbf{p}_u, \mathbf{p}_d$	A pair of 3D points corresponding to the observed pixels that are symmetric about $\mathbf{u}_k$
$\mathbf{u}^r$	2D pixel with range information
$\mathbf{u}_k$	2D fiducial that is detected but corresponds to an unobserved region in the raw intensity image

ξ	Lie algebra coordinates
$^j\mathbf{p}^{j,s}$	3D coordinates of the s -th coroner of the j -th marker expressed in $\{M\}_j$
${}_i\mathbf{p}^{j,s}$	3D coordinates of the s -th coroner of the j -th marker expressed in $\{F\}_i$

Matrices

$\mathbf{A}$	Coefficient matrix of $\hat{\mathbf{I}}(\mathbf{p})$
$\mathbf{A}\mathbf{A}^T$	Random positive definite matrix
$\mathbf{B}$	Random orthonormal matrix
$\mathbf{D}$	Design matrix
$\mathbf{H}\mathbf{A}^T$	Covariance matrix of $\mathbf{q}$ and $\mathbf{q}'$
$\mathbf{I}$	Identity matrix
$\mathbf{R}$	Rotation matrix
$\mathbf{T}$	Extrinsic matrix
$\mathbf{T}^{j,k}$	Pose that transforms 3D points from $\{M\}_j$ to $\{M\}_k$
$\mathbf{T}^j$	Pose that transforms 3D points from $\{M\}_j$ to $\{W\}$
$\mathbf{T}_i^j$	Pose that transforms 3D points from $\{M\}_j$ to $\{F\}_i$
$\mathbf{T}_{i,m}$	pose that transforms 3D points from $\{F\}_i$ to $\{F\}_m$
$\mathbf{T}_{in}$	Transmission from $\{W\}$ to the intermediate plane
$\mathbf{T}_i$	Pose that transforms 3D points from $\{F\}_i$ to $\{W\}$
$\mathbf{T}_{OBB}$	Pose that transforms 3D points from $\{W\}$ to the OBB frame

$\mathbf{U}\mathbf{\Lambda}\mathbf{V}^T$	SVD of $\mathbf{H}$
$\mathbf{X}$	Orthonormal matrix which is the product of $\mathbf{V}$ and $\mathbf{U}^T$
$\mathbf{M}_1, \mathbf{M}_2$	Weight matrices of $\mathbf{p}_d$ and $\mathbf{p}_u$
Σ	Covariance matrix

List of Acronyms

LiDAR	Light Detection and Ranging
LFMs	LiDAR fiducial markers
VFM	Visual fiducial markers
IFM	Intensity Image-based LiDAR Fiducial Marker
6-DOF	6-degree-of-freedom
MAP	Maximum a-posteriori
GPS	Global Positioning System
SfM	Structure-from-Motion
SLAM	Simultaneous Localization and Mapping
RGB-D	Red Green Blue-Depth
IMUs	Inertial Measurement Units
AR	Augmented Reality
LOAM	LiDAR Odometry and Mapping
LO	LiDAR Odometry
ICP	Iterative Closest Point
ROS	Robot Operating System
FoV	Field of View
SVD	Singular Value Decomposition

MoCap	OptiTrack Motion Capture
CPU	Central Processing Unit
OBB	Oriented Bounding Box
PCA	Principal Component Analysis
PC	Primary Component
LM	Levenberg-Marquardt
ID	Identification
RTK	Real-Time Kinematic
RMSEs	Root-mean-square errors
$RMSE_T$	Root-mean-square error of translation
$RMSE_R$	Root-mean-square error of translation rotation
GPU	Graphics Processing Unit
CD	Chamfer Distance
RR	Registration Recall

1 Introduction

1.1 Motivation

Light Detection and Ranging (LiDAR) sensors are crucial ranging sensors for autonomous systems [10, 14, 15]. In contrast to another vital sensor, the camera, the development and application of LiDAR fiducial markers (LFMs) are rare. Specifically, research on Visual Fiducial Marker (VFM) systems [1, 16–19], fiducial marker systems developed for cameras, has a long history, with extensive experience and significant research achievements accumulated. VFMs provide environments with controllable artificial features, making feature extraction and matching simpler and more reliable. VFM systems have been applied in various camera-based applications, including Augmented Reality (AR) [20], human-robot interaction [16, 17], navigation [21–23], multi-sensor calibration [24, 25], Structure-from-Motion (SfM) [7, 26], and visual Simultaneous Localization and Mapping (SLAM) [27–29].

LiDAR-based mapping and localization [2, 3, 10, 12–14, 30] are fundamental in robotics and computer vision. Similar to visual SLAM [28, 29, 31] and SfM [7, 26] in camera-based mapping and localization, there are two main categories of solutions. The first category is LiDAR-based SLAM approaches [10, 12–14], which process consecutive LiDAR scans in real time to map the scene and localize the LiDAR sensor simultaneously. The second

category is multiview point cloud registration methods [2, 3, 30]. Compared with SLAM approaches that process LiDAR scans sequentially, these methods handle a set of unaligned point clouds all at once in an offline manner. The mapping is achieved by registering the multiview point clouds into one complete point cloud, and localization is accomplished by finding the relative pose between point clouds. To clarify the motivation, this dissertation will answer three questions in this field:

- Given the success of VFM in camera-based applications, why is it still desirable to exploit the LiDAR sensor and LFM?
- Considering the existence of previous LiDAR fiducial objects, why is it favorable to develop a new type of LiDAR fiducial marker system?
- Given that previous LiDAR-based mapping and localization methods do not rely on fiducial markers, why is it advantageous to exploit LFM in this dissertation?

Response to the First Question. There are two reasons why LiDAR sensors and LFM are irreplaceable by cameras and VFM. First, as a ranging sensor, the role of LiDAR in autonomous systems [32–34] cannot be replaced by the camera. While Red Green Blue-Depth (RGB-D) cameras can also measure depth, their effective range is usually less than 10 meters [35–37], whereas LiDAR can scan objects several hundred meters away [10, 14, 15]. Additionally, due to the unique modality of LiDAR point clouds, data captured by LiDAR cannot be replaced by data acquired from cameras. Extensive learning-based models [15, 34, 38] rely on LiDAR data for training. Second, unlike VFM detection, which is sensitive to ambient light [1, 18, 19], LiDAR-based object detection is robust to unideal illumination conditions [9, 39], such as purely dark or overexposed environments. Third, VFM detection suffers from rotational ambiguity [7, 27, 40] and requires discarding ambiguous measurements, whereas LiDAR-based planar object pose estimation is free from

this issue [9].

Response to the Second Question. There have been some fiducial objects developed for LiDAR sensors [9, 41–43] for calibration purposes. There are three reasons why it is desired to develop a new type of LFM system. First, most existing LiDAR fiducial objects [9, 41–43] are designed for specific LiDAR models and are not generalizable to a wide range of solid-state and mechanical LiDARs. Second, unlike popular VFMs such as AprilTag [1] and ArUco [18], which are thin-sheet objects attached to surfaces without affecting the 3D geometry of the environment, most existing LiDAR fiducial objects [9, 41, 42] are thick boards placed on tripods, acting as additional 3D objects that alter the environment. These extra 3D objects are undesirable in applications such as scene reconstruction and data collection. Third, a typical calibration board [41–43] does not have complex patterns generated from elaborately designed encoding-decoding algorithms like VFMs [1, 16–19], which are robust against false positives and negatives.

Thus, the first major objective of this dissertation is to develop a unified LFM system that is applicable to a variety of LiDAR sensors, while being as convenient to use as VFMs: printed on sheets of paper/board and attached to surfaces without affecting the 3D environment. The unified system is desirable because we do not need to redesign the algorithm/marker every time the sensor is changed, and it is beneficial for sensor fusion of different models. Additionally, the system should incorporate a reliable encoding-decoding algorithm, like VFMs [1, 16–19], for robust detection.

Response to the Third Question. The existing point cloud registration methods, including geometry-based methods [6, 8, 13, 44] and learning-based methods [2–5, 45], rely on shared geometric features between point clouds to align them. Hence, these methods struggle in scenes with scarce shared features, such as extremely low overlap cases and degraded scenes [46]. Consequently, the existing methods [2–6] are unsuitable for tasks

such as capturing the complete 3D shape of a novel object from a sparse set of scans taken from dramatically different viewpoints, gathering data from unseen scenarios for training, reconstructing degraded scenes, or merging large-scale 3D maps with low overlap. In contrast, the introduction of LFM makes feature extraction and matching simpler and more reliable. Thanks to this, the proposed method is robust to any unseen scenarios with extremely low overlap, making it a convenient, efficient, and low-cost tool for diverse applications that pose significant challenges to existing methods. To help readers understand this point, a case with similar motivation in a camera-based application is presented: Although marker-free SfM methods [47, 48] exist, when constructing the Omniobject3D [49] dataset—a collection of posed multiview images of real-world objects—VFM-based SfM is employed to efficiently and robustly estimate the relative pose between images using a chessboard composed of AprilTags [1]. One may consider adding external sensors, such as Inertial Measurement Units (IMUs), to the LiDAR sensor to tackle challenging textureless scenes or cases with dramatic viewpoint changes. However, this involves multi-sensor calibration and synchronization. In comparison, LFMs made from sheets of paper or board are low-cost, easy to use, and reliable. This motivation is the same as that of VFM-based SfM [7, 26] and visual SLAM [27, 28] studies.

In summary, the main objective of this dissertation is to develop a framework for mapping and localization using LFMs. This framework will serve as a fundamental tool for robotics and computer vision, enabling a variety of real-world applications, including AR, 3D asset collection from sparse scans, training data collection in unseen scenes, reconstruction of degraded scenes, localization in GPS-denied environments, and merging large-scale low overlap maps. In particular, considering that training data collection for point cloud registration is time-consuming and expensive, and existing datasets [35–37] are limited to indoor scenes captured using RGB-D cameras, it is desirable to construct a new training

dataset in a straightforward and low-cost manner to benefit learning-based point cloud registration methods [2, 3].

1.2 Related Work and Challenges

1.2.1 Visual Fiducial Markers

VFM [1, 16–19, 50, 51] are particular objects, commonly planar, that provide the environment with controllable artificial features. They were originally developed for augmented reality [20], but have been widely applied in various robotics and computer vision applications, such as human-robot interaction [16, 17], navigation [21–23], multi-sensor calibration [24, 25], SfM [7, 26], and visual SLAM [27–29]. The previous research on the VFM system mainly focuses on the following four aspects: (1) A higher detection rate. Line segmentation methods have been continuously upgraded alongside VFM systems [16, 17]. In addition to line segmentation, non-square shape detection methods, such as ellipse detection algorithms [19, 50, 51], have also emerged and made valuable contributions. These advancements aim to improve the detection of line segments, candidate quads [1, 18], or circles [19, 50, 51], thereby enhancing the marker detection rate under varying ambient light and/or motion blur. (2) A lower false positive rate. The encoding-decoding algorithms of the latest VFM systems [1, 18] have been upgraded compared to earlier generations [16, 17], making the identification of candidate markers more reliable and reducing false positives during decoding. (3) A lower computational time. Again, through the amelioration of methods for graphic segmentation and algorithms adopted in encoding-decoding algorithms, the VFM systems are accelerated. For instance, the speed of the third generation of Apirltag [1] is almost 5 times faster than that of the second generation [16]. (4) Resolving the rotation ambiguity problem. The rotational ambiguity means

a planar marker could project onto the same pixels from two different poses when the perspective effect is weak [52]. Much research [53, 54] has been conducted to solve this problem by adding external sensors or changing the marker from 2D to 3D. Unfortunately, rotational ambiguity is still an unresolved problem for 2D VFM detection without external information.

However, the aforementioned VFM algorithms were developed for 2D images, not 3D LiDAR point clouds. Given the differences between 2D image and 3D point cloud modalities, it is challenging—though beneficial and desirable—to transfer the advancements made in VFM research to LiDAR-based applications [9].

1.2.2 LiDAR Fiducial Objects

1.2.2.1 LiDAR Fiducial Object Patterns

Most existing LiDAR fiducial objects [41–43] are designed for calibration purposes. Fig. 1.1(a) shows a typical calibration board, a thick board with holes and/or regions covered by high-intensity materials. While a calibration board can provide fiducials (holes, corners, and high-intensity regions) and it is feasible to assign specific indexes to the fiducials, a calibration board is fundamentally different from fiducial marker systems [1, 18]. This is because the design of calibration board patterns does not incorporate the core element of a fiducial marker system—the encoding-decoding algorithm. Consequently, calibration boards [41, 42] do not support rich patterns with unique identifications (IDs) and the robust detection provided by encoding-decoding algorithms [1, 16–19, 50, 51] as shown in Figs. 1.1(b)(c). The development of VFMs and LiDAR fiducial objects had been disjointed until the proposal of LiDARTag [9], as illustrated in Fig. 1.1(d). LiDARTag, as the first fiducial marker system for LiDAR sensors, integrates the encoding-decoding algorithm of AprilTag 3 [1] into LiDAR fiducial object detection. Consequently, it inherits the rich

patterns and robust detection features of AprilTag 3. However, LiDARTag only studies square markers with patterns belonging to AprilTag 3 and does not support square markers with different pattern sources, for instance, ArUco [18], or non-square markers, such as CCTag [19]. Developing a general algorithm that fully incorporates the achievements of VFM research while not requiring a specific marker pattern and shape is challenging. As shown in Fig. 1.1(e), the other widely used LiDAR fiducial objects are prisms [55], which are designed to reflect laser beams and produce very high intensity, making them easy to detect. However, the cost of prisms is much higher than that of VFMs, and they also do not contain the encoding-decoding algorithm to support rich patterns that are robust to false positives/negatives.

1.2.2.2 LiDAR Fiducial Object Placement

3D LiDAR point clouds are sparse and unstructured compared to 2D images, making it challenging to develop an LFM system [9]. Existing LiDAR fiducial objects [42, 56], including LiDARTag [9], use conventional geometry-based clustering methods [15, 57, 58], requiring spatial distinction. Thus, as seen in Fig. 1.1(a) and (d), they are often placed on tripods, introducing extra 3D objects that impact the environment. For instance, LiDARTag adopts the single-linkage clustering algorithm [58] and requires that the marker must have at least $t_L\sqrt{2}/4$ clearance around it, where t_L represents the marker’s size. If the marker is attached to a wall or a box, it is necessary to ensure that $\omega > t_L\sqrt{2}/4$, where ω is the thickness of the marker’s 3D object. Now, consider the case where ten LiDARTags are placed in an indoor environment. This situation would result in a room crowded with tripods and large markers. Although the sizes of prisms [55] (see Fig. 1.1(e)) are smaller than those of calibration boards or LiDARTags, they still act as additional objects. Moreover, in applications such as training data collection for learning-based point

cloud registration methods [2,3], these additional 3D objects are undesirable as they wreck the 3D geometry of the scene. However, considering the different modalities of 3D LiDAR point clouds and 2D images, it is challenging to develop an LFM system as convenient as VFMs (see Fig. 1.1(c)).

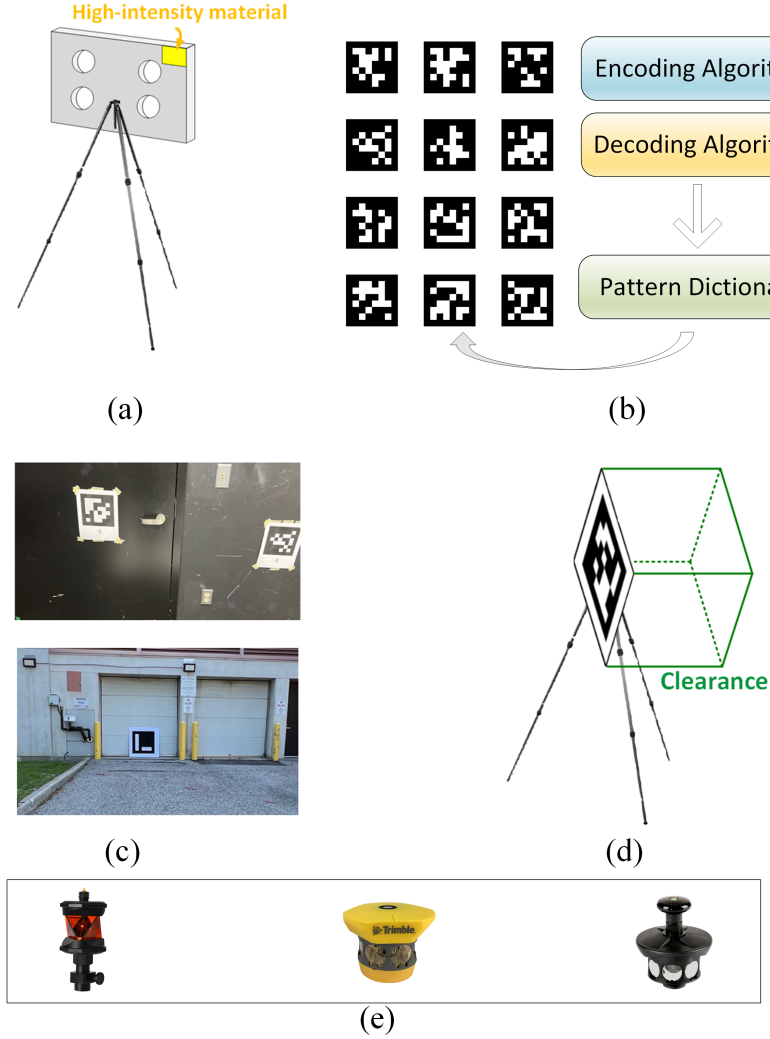


Figure 1.1: Comparison of (a) a typical calibration board, (b)(c) VFM, (d) LiDARTag, and (e) prisms in terms of object patterns and placement.

1.2.3 LiDAR-based Mapping and Localization

1.2.3.1 LiDAR-based SLAM

LiDAR Odometry and Mapping (LOAM) [14] is a milestone work in LiDAR-based SLAM, inspiring numerous follow-up studies [8, 10, 12, 13, 59, 60]. It splits the SLAM problem into two parallel tasks: LiDAR Odometry (LO) and mapping. LO is performed on the front end at high frequency to extract feature points (*e.g.* corners, surfaces, *etc.*) and use them to estimate the LiDAR pose in real-time. Mapping operates on the back end at a lower frequency to refine the odometry results and construct a more accurate map. LiDAR-based SLAM methods [8, 10, 12–14, 59, 60] focus on sequentially processing consecutive LiDAR scans and are designed for use cases that particularly require real-time response. Applications with the following three features are not use cases for LiDAR-based SLAM: (1) require processing a set of point clouds all at once, (2) involve an unordered set and/or point clouds with low overlap (dramatic viewpoint changes), and (3) allow for offline operation. In contrast, multiview point cloud registration methods are designed for these cases.

1.2.3.2 Multiview Point Cloud Registration

Multiview point cloud registration [2, 3, 30, 61] is also a popular solution for LiDAR-based mapping and localization. It directly processes a set of point clouds, which can be unordered and have low overlap, in an offline manner. In particular, mapping is achieved by registering the multiview point clouds into a complete point cloud, while localization is done by determining the relative pose between them. The mapping and localization framework introduced in this dissertation belongs to this category.

Geometry-based point cloud registration methods, such as variants of Iterative Closest Point (ICP) methods [8, 13] and Teaser++ [6], mainly focus on designing efficient and

robust algorithms for describing geometry and extracting geometric features (*e.g.*, points, lines, and surfaces/normals) to find correspondence between point clouds and align them. In learning-based point cloud registration methods like Predator [45] and SGHR [3], neural networks are designed to learn features representing the geometry [4, 5, 35, 61, 62], and then learn to utilize these features for registering the point clouds. However, most existing methods [8, 13, 35, 62] apply only to high overlap scenarios, making them less robust in practical applications [45]. Some recent learning-based research proposes multiview registration methods [3, 30, 61] and studies low overlap cases [3, 45]. Despite their promising performance on benchmarks [35–37], the generalization of learning-based methods to unseen scenarios (*i.e.*, out-of-distribution cases of training data) remains problematic. Moreover, the existing methods [2–6] face challenges in handling degraded scenes, such as repetitive structures and textureless environments. In camera-based applications [7, 26, 27], VFMs have been utilized to tackle challenging degraded scenes and low-overlap cases, while exploiting LFMs for multiview point cloud registration remains an open problem.

1.2.4 Training Data Collection for Point Cloud Registration

Training data is crucial for the development of learning-based point cloud registration methods [2, 3], but collecting it is time-consuming and expensive. For instance, in previous dataset collections [35–37], external sensors such as cameras, IMUs, and GPS are employed to obtain ground truth poses among point clouds. This requires careful multi-sensor calibration and synchronization, which can be time-consuming and labor-intensive. A common approach [3, 45] to evaluating a learning-based point cloud registration model is to train it on 3DMatch [35] and test it on various benchmarks, including 3DMatch [35], ETH [36], and ScanNet [37]. However, the 3DMatch benchmark is mainly constructed from RGB-D camera captures of indoor scenes [35]. Collecting a new training dataset with

outdoor scenes and point clouds captured by a LiDAR sensor for point cloud registration is beneficial for improving the performance of learning-based methods. Unfortunately, this is challenging due to the lack of an efficient, reliable, and easy-to-use tool for the task.

1.3 Contributions

This dissertation makes several contributions to addressing the technological gap in utilizing LFMs for mapping and localization introduced in Section 1.2. Specifically, the main contributions of this dissertation are as follows:

- **Intensity Image-Based LiDAR Fiducial Marker (IFM) System.** A new IFM system is introduced that supports various LiDAR models and integrates seamlessly with existing VFM patterns ¹. This system offers a practical and cost-effective solution by allowing users to generate markers with standard printing methods, making the process as convenient as current VFM systems.
- **Advanced LFM Detection Method.** A novel approach is developed to detect 3D fiducial markers through intensity images. This enables the integration of VFM systems into the IFM system, enhancing its flexibility compared to existing systems like LiDARTag, which are limited to specific marker patterns.
- **Improved Pose Estimation.** A new pose estimation technique for LiDAR is proposed, achieving higher accuracy than traditional VFM-based camera systems. The method also eliminates rotation ambiguity and provides robustness against lighting variations.

¹Please note that the focus of our research is enabling LiDAR sensors to detect thin-sheet markers with patterns from VFM research, rather than proposing new patterns.

- **Enhanced Localization of Thin-Sheet LFMs.** Limitations of the vanilla IFM are addressed. A method is introduced to extend LFM localization to 3D LiDAR maps and increase the detection distance of LFMs.
- **Joint Analysis of Intensity and Geometry.** A novel pipeline is proposed for detecting planar fiducial markers by jointly analyzing 3D point clouds from both intensity and geometry perspectives. This differs from conventional methods, which rely solely on geometric features and are ineffective for detecting planar objects indistinguishable from their background.
- **Framework for Mapping and Localization.** A comprehensive framework for mapping and localization using LFMs is developed. This framework registers unordered multiview point clouds with low overlap, providing a reliable tool for various applications, such as 3D asset collection from sparse scans, training data collection in unseen scenes, degraded scene reconstruction, GPS-denied environment localization, and 3D map merging.
- **Livox-3DMatch Dataset.** A new training dataset, Livox-3DMatch, is created, augmenting the original 3DMatch data. This expanded dataset improves the performance of state-of-the-art learning-based methods across multiple benchmarks.
- **Adaptive Threshold Detection Algorithm.** A viewpoint-robust LFM detection algorithm is developed, ensuring reliable detection in varying real-world conditions.

These contributions collectively advance the use of LFMs in robust and efficient LiDAR-based mapping and localization systems, addressing critical challenges and unlocking new possibilities in robotics and computer vision.

1.4 Dissertation Organization

This dissertation is organized into five chapters, in addition to the Introduction. Chapter 2 is about the development of a general fiducial marker system for the LiDAR sensor, named IFM. Section 2.1 presents an overview of the IFM system, including its overall pipeline and advantages compared to closely related prior work LiDARTag [9]. Chapter 2.2 mainly introduces the preliminary knowledge of three-dimensional transformation, Lie group and Lie algebra, and LiDAR technology. Section 2.3 introduces a method for detecting 3D fiducials in a point cloud through the intensity image. Section 2.4 presents the estimation of the 6-degree-of-freedom (6-DOF) LiDAR pose using the IFM. Section 2.5 reports the qualitative and quantitative experimental results and analysis regarding the efficiency and limitations of the vanilla IFM. The developments listed in this chapter, and related materials have been published in the journal article [39]. The open-source implementation, based on C++ and Robot Operating System (ROS), as well as the datasets, are available at <https://github.com/York-SDCNLab/IILFM>.

Chapter 3 discusses an algorithm that addresses the two limitations of the vanilla IFM: its inapplicability to 3D LiDAR maps and its limited detectable range. Section 3.1 presents an overview of the proposed algorithm, including a comparison to the vanilla IFM in terms of the pipeline for processing point clouds and applicable scenes. Section 3.2 discusses a new method that jointly analyzes point clouds from both intensity and geometry perspectives to find point clusters that cloud contain fiducial markers. Section 3.3 presents a method for localizing fiducials within the point clusters via a designed intermediate plane. Section 3.4 introduces how the 3D maps are constructed. Section 3.5 reports the experimental results demonstrating the improvements brought by the proposed method over the vanilla IFM. The developments listed in this chapter, and related materials have been published

in the journal article [63]. The open-source implementation based on C++ is available at <https://github.com/York-SDCNLab/Marker-Detection-General>.

Chapter 4 is about the development of a framework exploiting LFMs for mapping and localization. Section 4.1 provides an overview of the proposed framework, including its overall pipeline and the rich downstream robotic and computer vision applications. Section 4.2 introduces the methodology. Section 4.3 introduces the construction of a new training dataset for point cloud registration, named Livox-3DMatch. Section 4.4 reports the experimental results that validate the proposed framework and demonstrate its rich applications in robotic and computer vision. The developments listed in this chapter and related materials are accepted in the journal article [64]. The open-source implementation, based on C++ and Python, as well as the datasets, are available at <https://github.com/yorklyb/L-PR>.

Finally, the conclusions and future work are presented in Chapter 5.

2 Intensity Image-based LiDAR Fiducial Marker System

2.1 Overview

This chapter introduces the vanilla IFM system. As illustrated in Fig. 2.1, the proposed IFM is a general system with ample flexibility. As seen, unlike previous LiDAR fiducial objects, the markers are thin-sheet objects attached to other surfaces without affecting the 3D geometry of the environment. In particular, Figs. 2.1(a,b,c) show that different VFM systems, such as (a) AprilTag [1], (b) CCTag [19], and (c) ArUco [18], can be easily embedded. Moreover, the system is applicable to both solid-state LiDAR Figs. 2.1(a, b) and mechanical LiDAR Fig. 2.1(c). An AR demo using the proposed system is shown in Fig. 2.1(d): the teapot point cloud is transmitted to the location of the marker in the LiDAR's point cloud based on the pose provided by the IFM system.

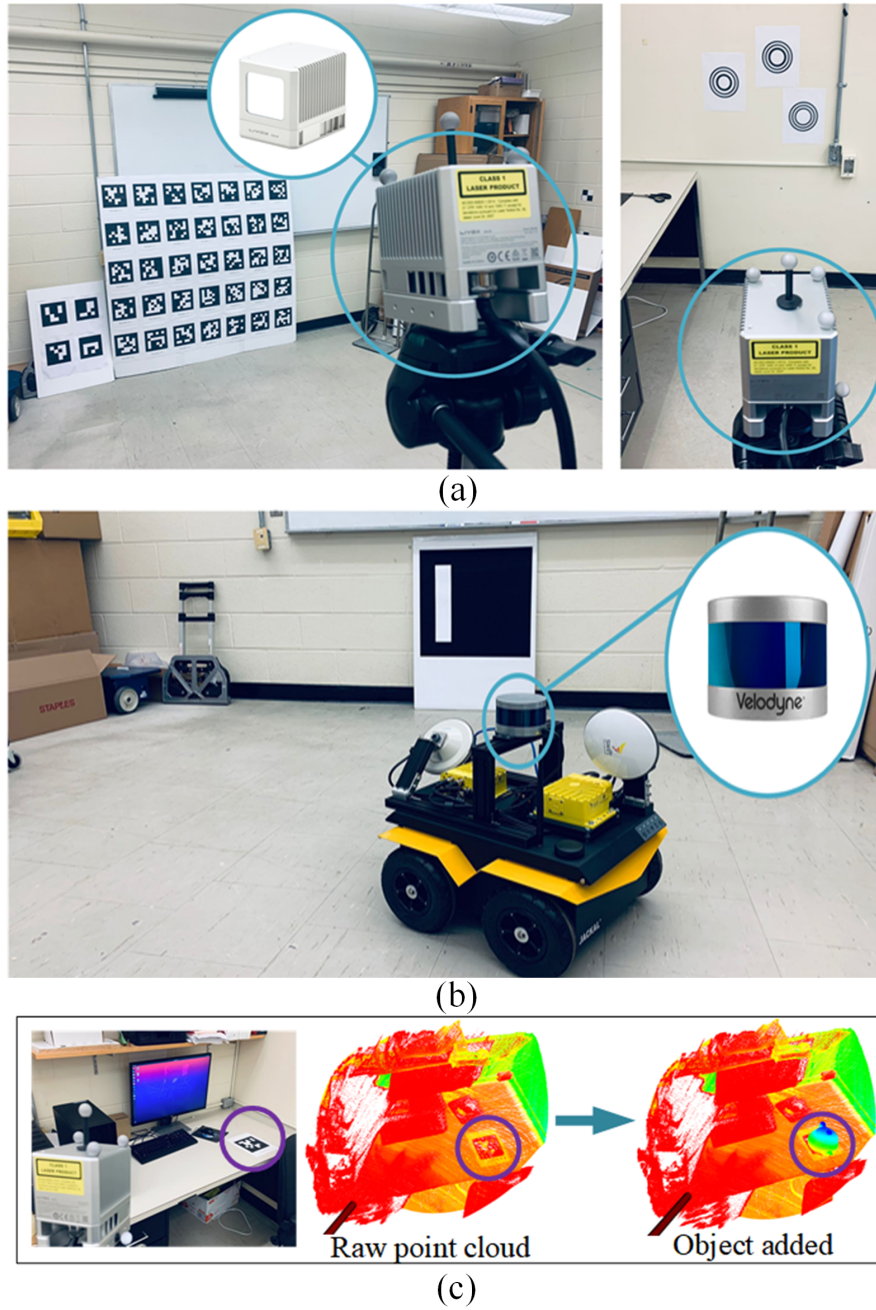


Figure 2.1: An illustration of the flexibility and generalizability of the proposed IFM.

LiDARTag [9] is a highly relevant prior work. Fig. 2.2 shows a comprehensive comparison between LiDARTag and the proposed system. The comparison regards three aspects.

- (i) Convenience of usage. The LiDARTag [9] system adopts the single-linkage clustering in the tag detection stage that brings restrictions on marker placement. Namely, there must be adequate clearance around the marker’s object, which makes the marker an extra object added to the spatial environment. Furthermore, according to the implementation of LiDARTag, the method to find the boundaries of the markers in the tag decoding stage requires that the marker be perpendicular to the ground. In contrast, owing to the fact that the IFM system does not adopt the clustering method based on spatial features, it has no restrictions on marker placement. That is, the user can place the marker as they do with the conventional VFM.
- (ii) Extensibility. The current version of LiDARTag only supports markers with patterns belonging to the AprilTag 3 family [1]. In the IFM system, however, different VFM systems (square and non-square) function directly since the marker detection is executed on the 2D intensity image.
- (iii) User-friendly design. The LiDARTag system [9], though definitely well-programmed, adopts many settings that are different from the implementations of VFM systems. For example, users have to utilize the marker’s local frame as the inertial coordinate system by default to define the sensor (LiDAR) pose. However, the development of the proposed system follows the successful VFM systems, such as AprilTag [1] and ArUco [18]. For instance, we leave the authority of defining the inertia coordinate system to the users as [1,18] do. As a result, users who are familiar with the popular VFM systems can commence with our system comfortably.

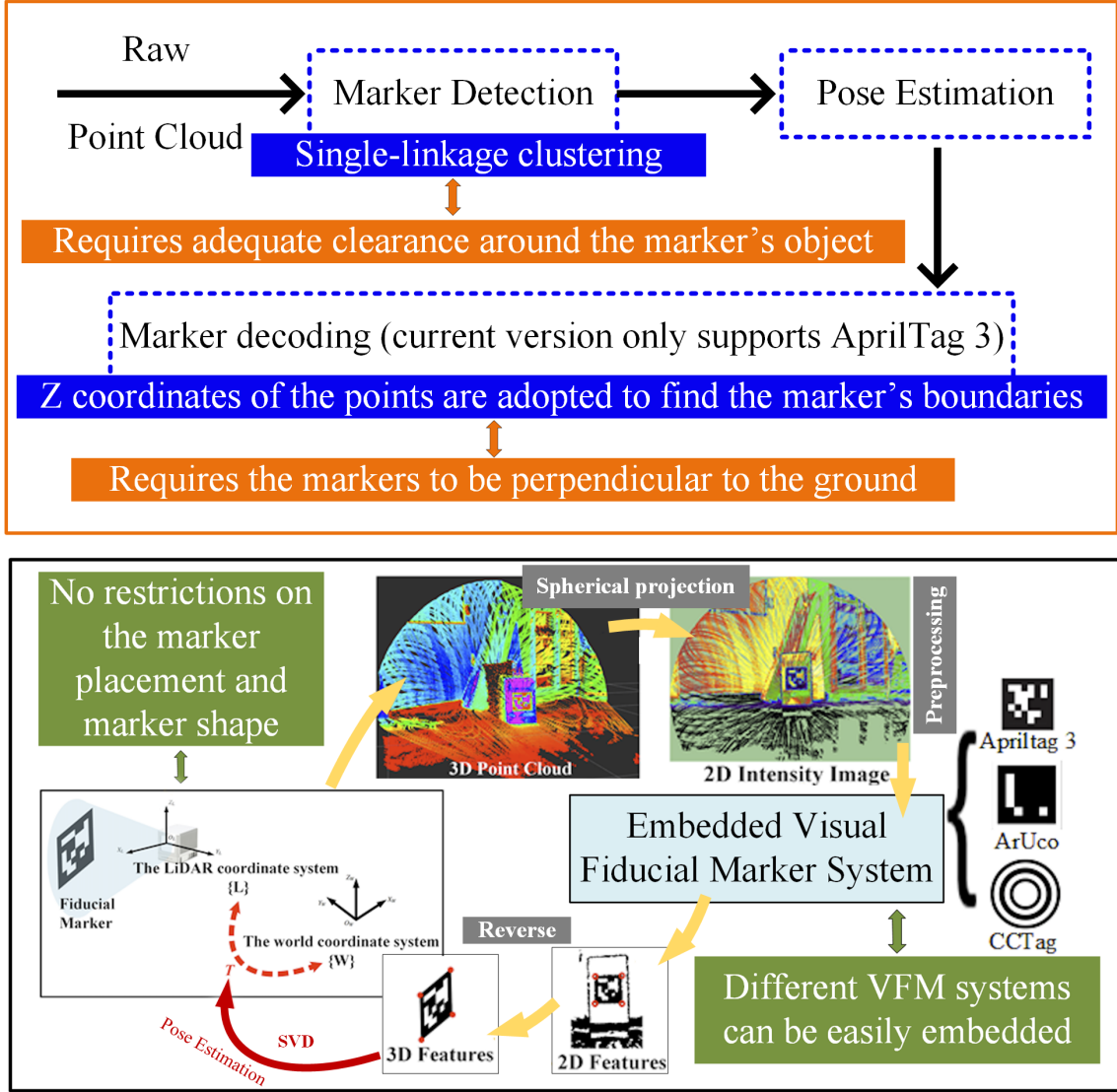


Figure 2.2: Comparison of LiDARTag [9] (top) and the proposed IFM (bottom).

2.2 Preliminaries

2.2.1 Three-dimensional Transformation

Suppose that $\mathbf{p}_a = [x, y, z]^T$ is a 3D point expressed in the coordinate system $\{a\}$. To express the point in another coordinate system $\{b\}$ as $\mathbf{p}_b = [x', y', z']^T$, translation and

rotation transformations need be applied to $\mathbf{p}_a$. In particular, the extrinsic matrix, $\mathbf{T} \in \mathbb{R}^{4 \times 4}$, is employed to describe the 6-DOF pose:

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^{1 \times 3} & 1 \end{bmatrix}, \quad (2.1)$$

where $\mathbf{t} \in \mathbb{R}^{3 \times 1}$ denotes the translation vector and $\mathbf{R} \in \mathbb{R}^{3 \times 3}$ denotes the rotation matrix.

Afterwards, $\mathbf{p}_b$ is obtained as follows:

$$\begin{bmatrix} \mathbf{p}_b \\ 1 \end{bmatrix} = \mathbf{T} \begin{bmatrix} \mathbf{p}_a \\ 1 \end{bmatrix}. \quad (2.2)$$

To simplify the notation, the operator $(\cdot)$ is introduced to denote:

$$\mathbf{p}_b = \mathbf{T} \cdot \mathbf{p}_a. \quad (2.3)$$

Please note that $(\cdot)$ is not a dot product as $\mathbf{T}$ is 4×4 and $\mathbf{p}$ is 3×1 . So it is different from a regular dot product. Eq. (2.2) shows what it means.

2.2.2 Lie Group and Lie Algebra

The rotation matrix $\mathbf{R} \in SO(3)$, which is the special orthogonal group representing rotations [65]:

$$SO(3) = \{\mathbf{R} \in \mathbb{R}^{3 \times 3} \mid \mathbf{R}\mathbf{R}^T = \mathbf{1}, \det(\mathbf{R}) = 1\}. \quad (2.4)$$

The pose matrix $\mathbf{T} \in SE(3)$, which is the special Euclidean group representing poses [65]:

$$SE(3) = \{\mathbf{T} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ 0 & 1 \end{bmatrix} \in \mathbb{R}^{4 \times 4} \mid \mathbf{R} \in SO(3), \mathbf{t} \in \mathbb{R}^{3 \times 1}\}. \quad (2.5)$$

$SO(3)$ and $SE(3)$ are two specific Lie groups. Every matrix Lie group is associated with a Lie algebra [65]. The Lie algebra $\mathfrak{so}(3)$ that is associated with $SO(3)$ is defined as

follows:

$$\begin{aligned}
SO(3) &\rightarrow \mathfrak{so}(3) : \\
\log(\mathbf{R}) &= \frac{\theta}{2\sin(\theta)}(\mathbf{R} - \mathbf{R}^T), \\
\xi &= \log(\mathbf{R})_{\vee},
\end{aligned} \tag{2.6}$$

where $\theta = \arccos \frac{1}{2}(\text{Trace}(\mathbf{R}) - 1)$. $\log(\cdot)$ is the matrix logarithm and $\xi \in \mathbb{R}^{3 \times 1}$ is the Lie algebra coordinates. $\vee$ is the *vee* map operator that finds the unique vector $\xi \in \mathbb{R}^{3 \times 1}$ corresponding to a given skew-symmetric matrix $\log(\mathbf{R}) \in \mathbb{R}^{3 \times 3}$ [65, 66].

2.2.3 LiDAR Technology

LiDAR is a technology that uses laser pulses to measure the distance between the sensor and objects in the environment. It creates precise three-dimensional representations of the scanned scene. The LiDAR sensor emits laser beams, and by measuring the time it takes for the reflected light to return to the sensor, the positions of 3D points in space are calculated. In addition to the geometric data, a LiDAR sensor can also capture intensity values, which reflect the strength of the returned signal. These intensity values are influenced by factors such as surface material, texture, and angle of incidence. The inclusion of intensity information enhances the ability to distinguish between different types of surfaces, improving tasks like object detection, scene understanding, and feature recognition. Fig. 2.3 illustrates the schematic diagram of the working principle of LiDAR.

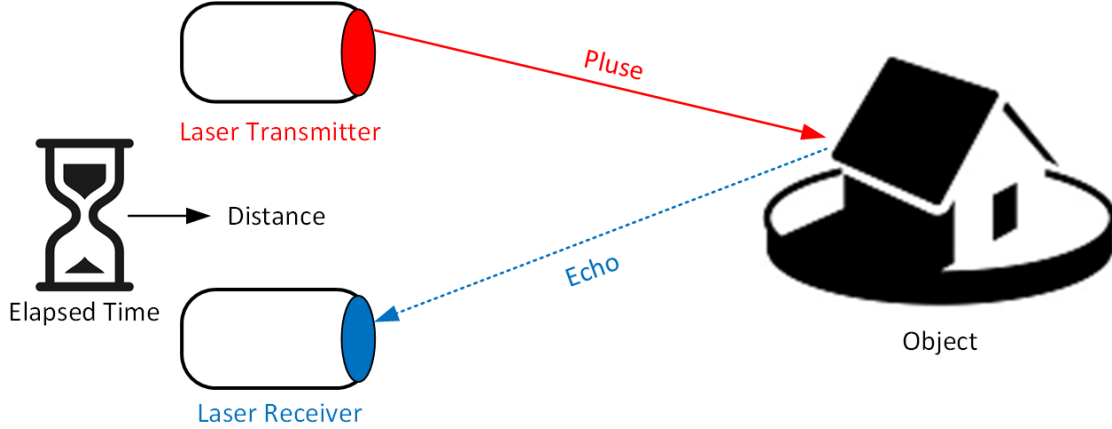


Figure 2.3: The schematic diagram of the working principle of LiDAR.

2.3 Marker Detection

2.3.1 Generation of the Intensity Image

As introduced in LiDARTag [9], the reason LiDAR fiducial objects act as additional 3D objects, abandoning the free-placement advantage of VFMs, is due to the gap between structured images and unstructured point clouds. Nevertheless, this gap is not insurmountable. In particular, there is a notable hot trend in LiDAR-based 3D object detection/segmentation [15, 67]: Neural Networks that are originally developed for 2D object detection/segmentation can be utilized to detect/segment the objects in the range/intensity image(s) of the 3D LiDAR point cloud. This indicates that the range/intensity image(s) generated from the LiDAR point cloud can be a pathway to transfer the research accomplishments on the 2D image to the 3D point cloud. Following this inspiration, and considering that the black-and-white marker is explicitly visible in the point cloud rendered by intensity, the intensity image is utilized for LFM development.

The generation of an intensity image from a given unstructured point cloud can be

summarized as transferring all the 3D points in the point cloud onto a 2D image plane by spherical projection and rendering the corresponding pixels with intensity values. Fig. 2.4 shows the coordinate systems and notations. $\mathbf{p}_L = [x_L, y_L, z_L, i]^T$ is an observed point in the 3D point cloud, with $[x_L, y_L, z_L]^T$ denoting its Cartesian coordinates w.r.t. the LiDAR coordinate system $\{L\}$ and i being the intensity. $\mathbf{u}$ is the projection of $\mathbf{p}_L$ onto the image plane, which has coordinates $[u, v]^T$ w.r.t. the image coordinate system, $\{I\}$.

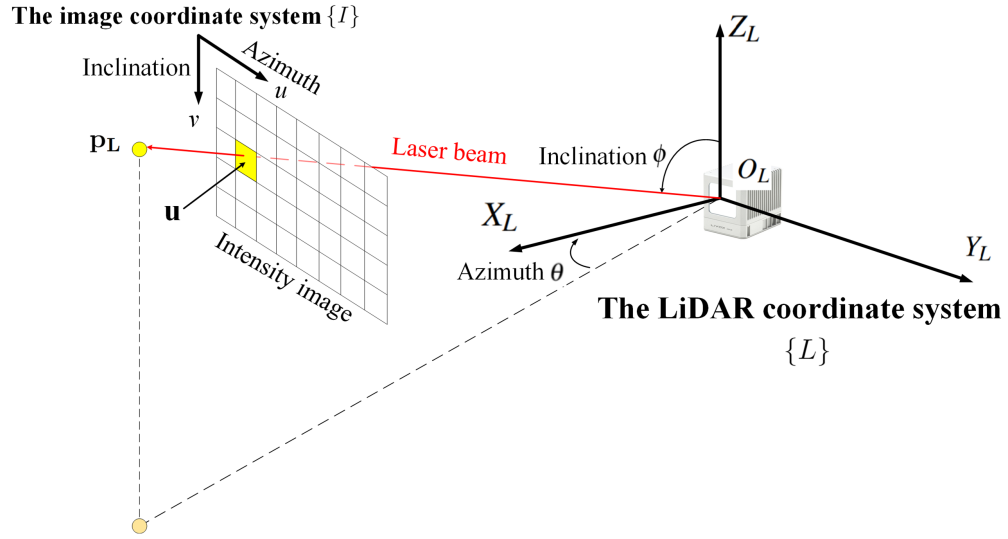


Figure 2.4: An illustration of the coordinate systems and notations.

Following [65], the Cartesian coordinates of $\mathbf{p}_L$ are first transformed to spherical coordinates $[\theta, \phi, r]^T$:

$$\begin{aligned}\theta &= \arctan\left(\frac{y_L}{x_L}\right), \\ \phi &= \arctan\left(\frac{z_L}{\sqrt{x_L^2 + y_L^2}}\right), \\ r &= \sqrt{x_L^2 + y_L^2 + z_L^2},\end{aligned}\tag{2.7}$$

where θ and ϕ denote the azimuth and inclination, respectively. r is the range from $\mathbf{p}_L$ to

the origin of $\{L\}$. Then, the image coordinates $[u, v]^T$ of $\mathbf{u}$ are given by:

$$u = \lceil \frac{\theta}{\Theta_a} \rceil + u_o, \quad v = \lceil \frac{\phi}{\Theta_i} \rceil + v_o \quad (2.8)$$

where $\lceil \cdot \rceil$ represents rounding a value to the nearest integer. Θ_a and Θ_i are the angular resolutions in u (azimuth) and v (inclination) directions, respectively. u_o and v_o are the offsets:

$$\theta = \Theta_a(u - u_o), \quad \phi = \Theta_i(v - v_o). \quad (2.9)$$

Assume it is desired that the point with zero-azimuth and zero-inclination to be projected to the center of the image, then the offsets will be: $u_o = I_w/2$ and $v_o = I_h/2$, where I_w and I_h being the image width and height which are determined by the maximum angular width P_w and height P_h of the point cloud $I_w = \lceil \frac{P_w}{\Theta_a} \rceil$, $I_h = \lceil \frac{P_h}{\Theta_i} \rceil$. The pixel $[u, v]^T$ is then rendered by a specific color corresponding to the intensity value i . Refer to [68, 69] to see how the correspondence between color and intensity value is generated. For each pixel, the range information r is also saved for the sake of the following pose estimation. Thereafter, we step through the point cloud and repeat the above process. The pixels that are not mapped to any points will remain unobserved and are rendered by a unique predefined value. Note that if these pixels are visited later on in the marker detection process, they will not return any 3D points because they represent unobserved regions.

2.3.2 Selections of the Angular Resolutions

The selections of Θ_a and Θ_i are crucial as they affect the quality of the intensity image directly. However, the solution to this problem is straightforward. Suppose that the horizontal angular resolution and vertical angular resolution given by the user manual of the employed LiDAR are Θ_h and Θ_v . We should set $\Theta_a = \Theta_h$ and $\Theta_i = \Theta_v$. Fig. 2.5

illustrates the effect of choosing different values of Θ_a and Θ_i on the intensity image, with the unobserved regions rendered in light green.

Figs. 2.5(a,b,c) are intensity images generated from the same point cloud given by one LiDAR scan of a Velodyne ULTRA Puck (mechanical LiDAR, $\Theta_h = 0.4^\circ$ and $\Theta_v = 0.33^\circ$). The LiDAR scan is extracted from the dataset provided by [9]. Moreover, Figs. 2.5(a,b,c) are cropped to display. Figs. 2.5(d,e,f) are intensity images generated from the same point cloud which is the integration of multiple LiDAR scans of a Livox Mid-40 (solid-state LiDAR, $\Theta_h = \Theta_v = 0.05^\circ$). The detailed settings for each subimage are as follows. (a): $\Theta_a = 0.05^\circ$ and $\Theta_i = 0.05^\circ$; raw image size = 7200×800 . (b): $\Theta_a = 0.4^\circ$ and $\Theta_i = 0.33^\circ$; raw image size = 900×121 . (c): $\Theta_a = 0.6^\circ$ and $\Theta_i = 1.0^\circ$; raw image size = 600×40 . (d): $\Theta_a = \Theta_i = 0.025^\circ$; raw image size = 1538×1178 . (e): $\Theta_a = \Theta_i = 0.05^\circ$; raw image size = 771×591 . (f): $\Theta_a = \Theta_i = 0.5^\circ$; raw image size = 81×62 .

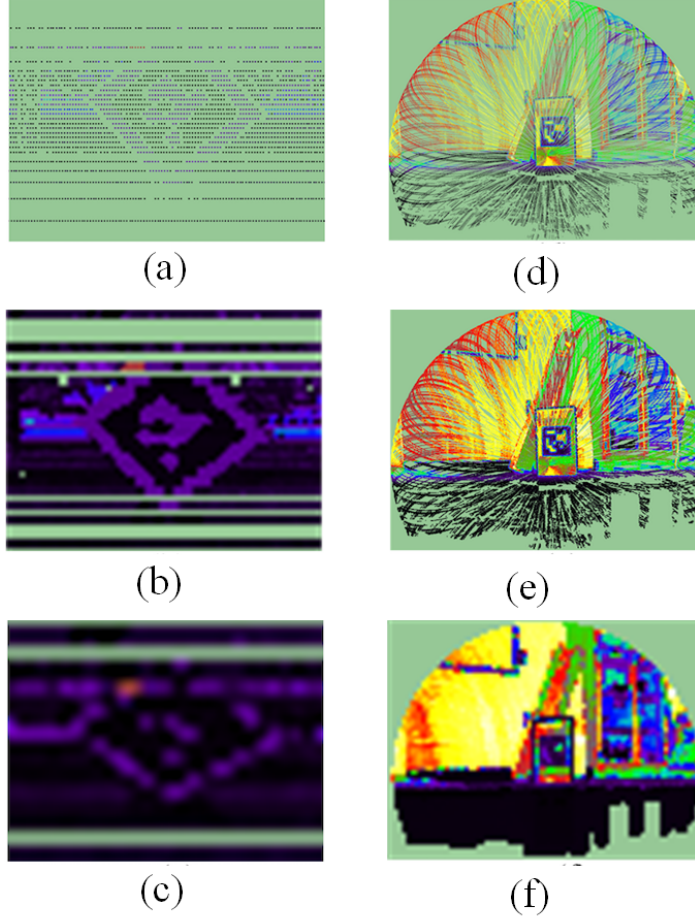


Figure 2.5: The intensity images generated under different angular resolution settings.

In summary, when $\Theta_a < \Theta_h$ and $\Theta_i < \Theta_v$, too many unobserved regions appear around or inside the marker's area (See Fig. 2.5(a) and (d)). When $\Theta_a > \Theta_h$ and $\Theta_i > \Theta_v$, too many points overlap for the same pixels based on Eq. (2.8), such that the marker's pattern could disappear (See Fig. 2.5(c) and (f)).

Nevertheless, as shown in Fig. 2.6, the LiDAR cannot sample perfectly evenly in the inclination/azimuth space. Specifically, Fig. 2.6(a) shows the general sampling pattern of a mechanical LiDAR expressed in the azimuth/inclination coordinate system. This is a general schematic that does not correspond to any LiDAR model. Fig. 2.6(b) presents

the sampling pattern of the Livox Mid-40 LiDAR after one-second integration. Note that the sampling patterns vary when it comes to different solid-state LiDAR models. But generally, the unscanned regions within the valid Field of View (FoV) appear as spots. Thus, even if the user manual of the employed LiDAR is followed to set $\Theta_a = \Theta_h$ and $\Theta_i = \Theta_v$, the unwanted unobserved regions around or inside the marker's area and the points overlapping issue are still inevitable on account the fact that Θ_h and Θ_v are approximate values themselves. Based on the experiments, the points overlapping issue under $\Theta_a = \Theta_h$ and $\Theta_i = \Theta_v$ could have an acceptable influence on pose estimation while the unobserved regions impede marker detection.

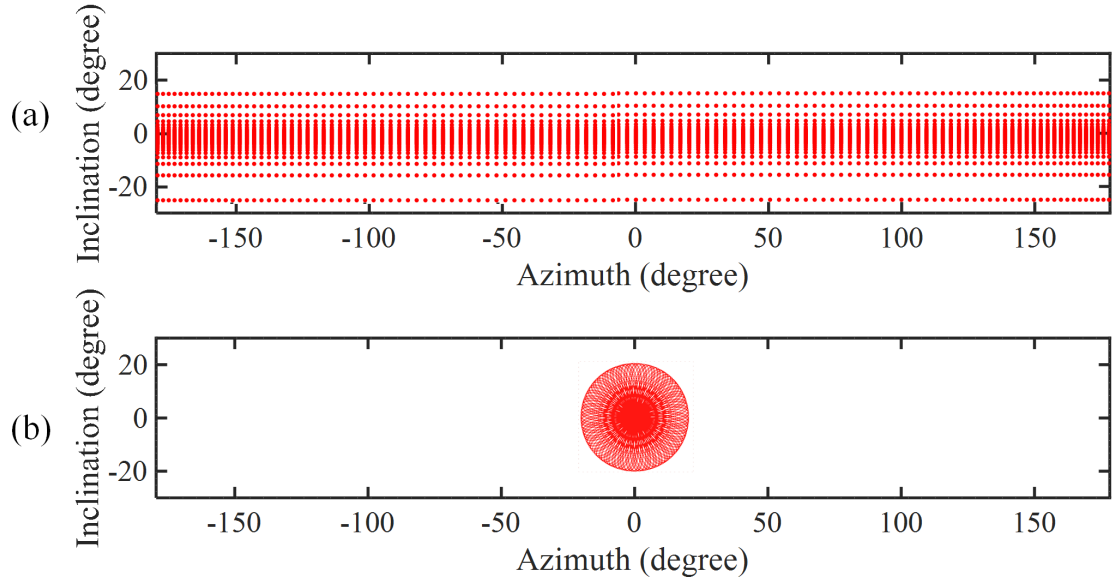


Figure 2.6: Sampling patterns of the mechanical LiDAR and solid-state LiDAR, with sampling points represented by red scatter plots.

To remove the image noise caused by unobserved regions, image preprocessing is carried out as shown in Fig. 2.7 before inputting the raw intensity image (Fig. 2.7(a)) into the embedded VFM system. The preprocessing includes grayscale conversion (Fig. 2.7(b)),

naive thresholding (Fig. 2.7(c)), and optional Gaussian blurring [70] (Fig. 2.7(d)). In particular, the raw intensity image (Fig. 2.7(a)) is first converted to a grayscale image (Fig. 2.7(b)). Then it becomes the binary image (Fig. 2.7(c)) using naive thresholding. After the naive thresholding, the image noise caused by varying low-intensity values is removed. Gaussian blurring [70] is recommended when the embedded VFM system, such as CCTag [19], does not contain it. Fig. 2.7(d) shows the intensity image of a CCTag after the naive thresholding while the detector cannot detect the marker in it. Fig. 2.7(e) presents the intensity image after applying Gaussian Blur [70] on Fig. 2.7(d). Now the marker is detectable. The methods utilized in the preprocessing are simple but not trivial, which implies that without them the marker detection will fail.

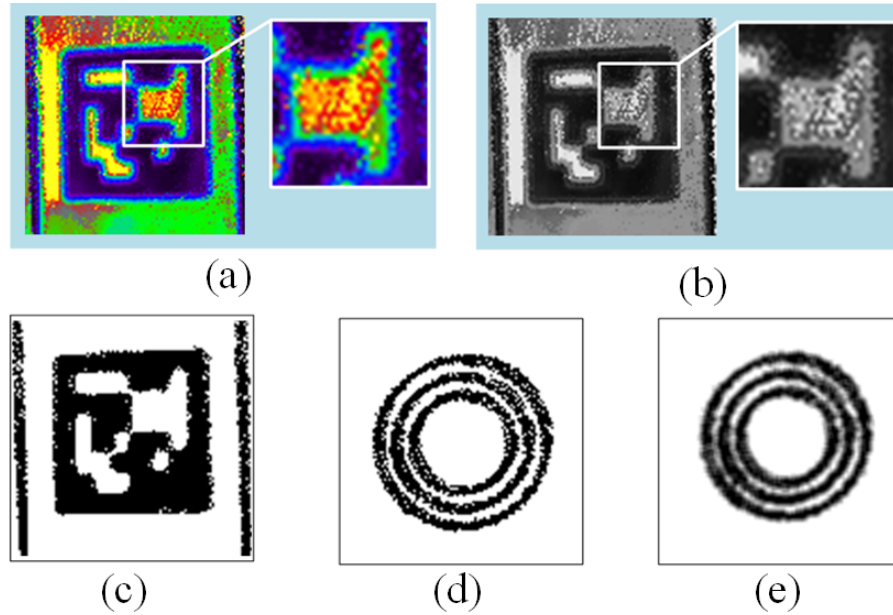


Figure 2.7: An illustration of image preprocessing in the system.

2.3.3 3D Fiducials Estimation

The preprocessed image introduced in the previous section is then inputted into the embedded VFM system. Thereafter, the VFM system provides the detection information. In this section, we introduce how to project the 2D fiducials given by the detection information back to the 3D space, such that they become 3D fiducials expressed in the LiDAR coordinate system. As for the square markers, the fiducials refer to the four vertices of the quad. While some of the VFM systems adopt the non-square design, the proposed method has no restriction on the marker shape.

As mentioned in Section 2.3.1, the range information is stored for each pixel in the intensity image as well. Thus, a 2D pixel with range information ($\mathbf{u}^r = [u, v, r]^T$) can be projected back to the 3D Cartesian coordinate system by solving the inverse of Eqs. (2.7-2.8) to find the corresponding $[x_L, y_L, z_L]^T$. However, in the real world, it cannot be guaranteed that the fiducials of the marker are exactly scanned by the LiDAR. Namely, the detected 2D features in Fig. 2.7(c) could correspond to the unobserved regions in the raw intensity image. As a matter of fact, this occurs frequently in our experiments, as well as in the dataset provided by [9]. Hence, when these detected but unscanned features are checked, there will be no range value r returned and it is not feasible to solve the inverse of Eqs. (2.7-2.8).

To resolve this problem, we propose an algorithm as illustrated in Fig. 2.8. The algorithm is based on the fact that the markers are planar. Suppose that there is a 2D feature point $\mathbf{u}_k = [u_k, v_k]^T$ that is detected but corresponds to an unobserved region in the raw intensity image. The azimuth θ_k of $\mathbf{u}_k$ is determined by Eq. (2.8) through $\theta_k = \Theta_a(u_k - u_0)$. Define the unknown 3D point corresponding to $\mathbf{u}_k$ as $\mathbf{p}_k = [x_k, y_k, z_k]^T$ (the yellow point in Fig. 2.8). Hereafter, suppose that in the same column as $\mathbf{u}_k$, there is a pair of observed pixels that are symmetric about $\mathbf{u}_k$, and define their corresponding points

as $\mathbf{p}_u = [x_u, y_u, z_u]^T$ and $\mathbf{p}_d = [x_d, y_d, z_d]^T$ (the black point in Fig. 2.8). As illustrated in Eq. (2.8), pixels in the same column approximately share the same azimuth. Thus, $\mathbf{p}_u$, $\mathbf{p}_k$, and $\mathbf{p}_d$ are on the same plane, α , which is specified by fixing the azimuth ($\theta = \theta_k$). After that, define the plane where the marker is located as β . Obviously, the intersection of α and β is a straight line $\mathbf{l}$. Under the assumption that $\mathbf{p}_u$ and $\mathbf{p}_d$ are on the marker plane β , it can be shown that $\mathbf{p}_u$, $\mathbf{p}_k$, and $\mathbf{p}_d$ are collinear and they all fall on $\mathbf{l}$. To introduce the algorithm more clearly, Fig. 2.9 shows the side view of plane α .

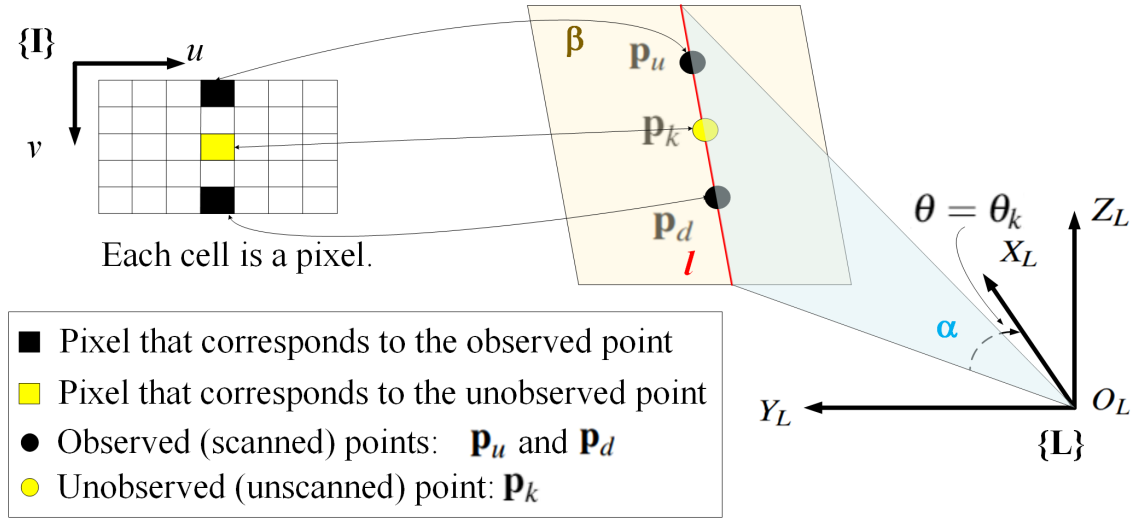


Figure 2.8: An illustration of the algorithm to estimate the 3D coordinates of a detected but unscanned 2D feature point.

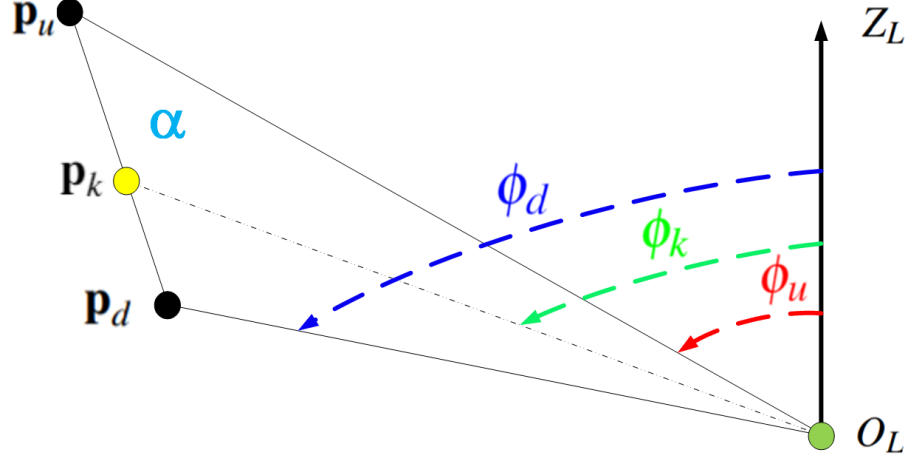


Figure 2.9: The side view of α . ϕ_d , ϕ_k , and ϕ_u are the inclinations of $\mathbf{p}_d$, $\mathbf{p}_k$, and $\mathbf{p}_u$, respectively.

Undoubtedly, $\mathbf{O}_L \mathbf{p}_k$ is the angle bisector of $\angle \mathbf{p}_u \mathbf{O}_L \mathbf{p}_d$ owing to $\phi_d - \phi_k = \phi_k - \phi_u$. Hence, in the light of the angle bisector properties, we have $\mathbf{p}_k \mathbf{p}_d / \mathbf{p}_u \mathbf{p}_k = \mathbf{O}_L \mathbf{p}_d / \mathbf{O}_L \mathbf{p}_u$. Note that $\mathbf{O}_L \mathbf{p}_d$ and $\mathbf{O}_L \mathbf{p}_u$ are the ranges of $\mathbf{p}_d$ and $\mathbf{p}_u$ which can be obtained if they are scanned. Thus, the unknown 3D coordinates of $\mathbf{p}_k$ are estimated by $\mathbf{p}_k = \mathbf{M}_1 \mathbf{p}_d + \mathbf{M}_2 \mathbf{p}_u$, where $\mathbf{M}_1 = \text{diag}(\frac{\mu}{1+\mu}, \frac{\mu}{1+\mu}, \frac{\mu}{1+\mu})$ and $\mathbf{M}_2 = \text{diag}(\frac{1}{1+\mu}, \frac{1}{1+\mu}, \frac{1}{1+\mu})$ with μ being the ratio $\mathbf{O}_L \mathbf{p}_d / \mathbf{O}_L \mathbf{p}_u$. So far, the 2D fiducials in $\{I\}$, observed and unobserved by the LiDAR, are projected back to $\{L\}$.

2.4 LiDAR Pose Estimation

The aim of LiDAR pose estimation is to seek the Euclidean transformation, $\mathbf{T} = [\mathbf{R} | \mathbf{t}]$, from the world (inertia) coordinate system $\{G\}$ to the LiDAR coordinate system $\{L\}$. $\mathbf{R}$ is a 3×3 orthogonal matrix that represents the rotation. $\mathbf{t} \in \mathbb{R}^3$ is the translation vector. Suppose that $\mathbf{f} \in \mathbb{R}^3$ are the 3D coordinates of a feature point, the operation of $\mathbf{T}$ on $\mathbf{f} \in \mathbb{R}^3$ is $\mathbf{T} \cdot \mathbf{f} = \mathbf{R} \mathbf{f} + \mathbf{t}$ (Refer to Section 2.2.1 if needed). LiDAR pose estimation can be

resolved through optimally aligning two point sets while in real-world applications, such as SLAM and perception, the point correspondences between the two point sets are unknown, such that the correspondences are also needed to be optimally and iteratively searched [71]. However, as seen in the following, with the help of the fiducial marker system, the search for correspondences can be skipped in LiDAR pose estimation, which is a vital benefit brought by using the fiducial marker system.

Thus far two sets of 3D points are obtained. 1) n fiducials w.r.t. $\{L\}$, denoted by $\mathcal{P}_{\mathcal{L}} = \{\mathbf{f}_1, \dots, \mathbf{f}_n\}$, which are given by the 3D fiducials estimation introduced in Section 2.3.3; 2) n fiducials w.r.t. $\{G\}$, denoted by $\mathcal{P}_{\mathcal{W}} = \{\mathbf{f}_1', \dots, \mathbf{f}_n'\}$, which are predefined. Furthermore, the points in $\mathcal{P}_{\mathcal{L}}$ and $\mathcal{P}_{\mathcal{W}}$ are matched based on the ID number and vertex index given by the marker detection. Hence, the LiDAR pose estimation can be transformed into finding $[\mathbf{R}|\mathbf{t}]$ that optimally align $\mathcal{P}_{\mathcal{L}}$ and $\mathcal{P}_{\mathcal{W}}$. This is inherently a least square problem:

$$\mathbf{R}^*, \mathbf{t}^* = \arg \min_{\mathbf{R}, \mathbf{t}} \sum_{j=1}^n \left\| \mathbf{f}_j - (\mathbf{R}\mathbf{f}_j' + \mathbf{t}) \right\|^2. \quad (2.10)$$

Considering that the point-correspondence between $\mathcal{P}_{\mathcal{L}}$ and $\mathcal{P}_{\mathcal{W}}$ is quite reliable thanks to the embedded VFM system, $\mathbf{R}^*$ and $\mathbf{t}^*$ can be calculated in closed form by the Singular Value Decomposition (SVD) method [65, 72]. The centroids of $\mathcal{P}_{\mathcal{L}} = \{\mathbf{f}_1, \dots, \mathbf{f}_n\}$ and $\mathcal{P}_{\mathcal{W}} = \{\mathbf{f}_1', \dots, \mathbf{f}_n'\}$ are defined as follows:

$$\begin{aligned} \mathbf{f} &= \frac{1}{n} \sum_{j=1}^n (\mathbf{f}_j), \\ \mathbf{f}' &= \frac{1}{n} \sum_{j=1}^n (\mathbf{f}_j'). \end{aligned} \quad (2.11)$$

Then, Eq. (2.10) can be reorganized as:

$$\begin{aligned}
& \sum_{j=1}^n \left\| \mathbf{f}_j - (\mathbf{R}\mathbf{f}'_j + \mathbf{t}) \right\|^2 \\
&= \sum_{j=1}^n \left\| \mathbf{f}_j - \mathbf{R}\mathbf{f}'_j - \mathbf{t} - \mathbf{f} + \mathbf{R}\mathbf{f}' + \mathbf{f} - \mathbf{R}\mathbf{f}' \right\|^2 \\
&= \sum_{j=1}^n \left\| \left(\mathbf{f}_j - \mathbf{f} - \mathbf{R}(\mathbf{f}'_j - \mathbf{f}') \right) + (\mathbf{f} - \mathbf{R}\mathbf{f}' - \mathbf{t}) \right\|^2 \\
&= \sum_{j=1}^n \left(\left\| \mathbf{f}_j - \mathbf{f} - \mathbf{R}(\mathbf{f}'_j - \mathbf{f}') \right\|^2 + \left\| \mathbf{f} - \mathbf{R}\mathbf{f}' - \mathbf{t} \right\|^2 + 2 \left(\mathbf{f}_j - \mathbf{f} - \mathbf{R}(\mathbf{f}'_j - \mathbf{f}') \right) (\mathbf{f} - \mathbf{R}\mathbf{f}' - \mathbf{t}) \right). \tag{2.12}
\end{aligned}$$

Following the definitions of $\mathbf{f}$ and $\mathbf{f}'$ shown in Eq. (2.11), we have:

$$\sum_{j=1}^n \left(\mathbf{f}_j - \mathbf{f} - \mathbf{R}(\mathbf{f}'_j - \mathbf{f}') \right) (\mathbf{f} - \mathbf{R}\mathbf{f}' - \mathbf{t}) = 0. \tag{2.13}$$

Namely, the last term of Eq. (2.12) is zero. Thus, based on the derivation given in Eq. (2.12), the problem introduced in Eq. (2.10) is transformed into the following form:

$$\mathbf{R}^*, \mathbf{t}^* = \arg \min_{\mathbf{R}, \mathbf{t}} \sum_{j=1}^n \left(\left\| \mathbf{f}_j - \mathbf{f} - \mathbf{R}(\mathbf{f}'_j - \mathbf{f}') \right\|^2 + \left\| \mathbf{f} - \mathbf{R}\mathbf{f}' - \mathbf{t} \right\|^2 \right) \tag{2.14}$$

As seen in Eq. (2.14), the rotation ($\mathbf{R}$) and translation ($\mathbf{t}$) have been decoupled. In particular, the first term only involves $\mathbf{R}$, so $\mathbf{R}$ can be computed first and then substituted into the second term to obtain $\mathbf{t}$. Define the centroid-aligned coordinates as:

$$\begin{aligned}
\mathbf{q}_j &= \mathbf{f}_j - \mathbf{f} \\
\mathbf{q}'_j &= \mathbf{f}'_j - \mathbf{f}'
\end{aligned} \tag{2.15}$$

By substituting the centroid-aligned coordinates into the first term of Eq. (2.14), the computation of rotation is described as follows:

$$\mathbf{R}^* = \arg \min_{\mathbf{R}} \sum_{j=1}^n \left\| \mathbf{q}_j - \mathbf{R}\mathbf{q}'_j \right\|^2. \tag{2.16}$$

Expanding Eq. (2.16) yields:

$$\begin{aligned} & \sum_{j=1}^n \left\| \mathbf{q}_j - \mathbf{R}\mathbf{q}'_j \right\|^2 \\ &= \sum_{j=1}^n \left(\mathbf{q}_j^T \mathbf{q}_j + \mathbf{q}'_j{}^T \mathbf{R}^T \mathbf{R} \mathbf{q}_j - 2\mathbf{q}_j^T \mathbf{R} \mathbf{q}'_j \right). \end{aligned} \quad (2.17)$$

Note that, since $\mathbf{R} \in SO(3)$, $\mathbf{R}^T \mathbf{R} = \mathbf{I}$. Hence, only the third term in Eq. (2.17) involves $\mathbf{R}$. As a result, the problem described in Eq. (2.17) is transformed into:

$$\begin{aligned} & \sum_{j=1}^n -\mathbf{q}_j^T \mathbf{R} \mathbf{q}'_j \\ &= \sum_{j=1}^n -\text{Trace} \left(\mathbf{R} \mathbf{q}'_j \mathbf{q}_j^T \right) \\ &= -\text{Trace} \left(\mathbf{R} \sum_{j=1}^n \mathbf{q}'_j \mathbf{q}_j^T \right). \end{aligned} \quad (2.18)$$

Let

$$\mathbf{H} = \sum_{j=1}^n \mathbf{q}'_j \mathbf{q}_j^T. \quad (2.19)$$

The computation of $\mathbf{R}$ is transformed from Eq. (2.16) to:

$$\min_{\mathbf{R}} -\text{Trace}(\mathbf{R}\mathbf{H}) \Leftrightarrow \max_{\mathbf{R}} \text{Trace}(\mathbf{R}\mathbf{H}) \quad (2.20)$$

Lemma For any positive definite matrix $\mathbf{A}\mathbf{A}^T$, and any orthonormal matrix $\mathbf{B}$

$$\text{Trace}(\mathbf{A}\mathbf{A}^T) \geq \text{Trace}(\mathbf{B}\mathbf{A}\mathbf{A}^T). \quad (2.21)$$

The detailed proof of this Lemma is provided in [72]. Find the SVD of $\mathbf{H}$:

$$\mathbf{H} = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^T, \quad (2.22)$$

where $\mathbf{U}$ and $\mathbf{V}$ are 3×3 orthonormal matrices, and $\mathbf{\Lambda}$ is a 3×3 diagonal matrix with non-negative elements. Define an orthonormal matrix as follows:

$$\mathbf{X} = \mathbf{V}\mathbf{U}^T. \quad (2.23)$$

Then, the following equation is obtained:

$$\begin{aligned}\mathbf{XH} &= \mathbf{V}\mathbf{U}^T\mathbf{U}\mathbf{\Lambda}\mathbf{V}^T \\ &= \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T,\end{aligned}\tag{2.24}$$

which is symmetrical and positive definite. Consequently, based on the Lemma, for any 3×3 orthonormal matrix $\mathbf{B}$,

$$Trace(\mathbf{XH}) \geq Trace(\mathbf{BXH}).\tag{2.25}$$

Namely, among all 3×3 orthonormal matrices, $\mathbf{X}$ maximizes $Trace(\mathbf{RH})$. Therefore,

$$\mathbf{R}^* = \mathbf{X} = \mathbf{V}\mathbf{U}^T\tag{2.26}$$

is the solution to the problem described in Eq. (2.20). Finally, $\mathbf{t}^*$ can be computed by substituting $\mathbf{R}^*$ into the second term of Eq. (2.14). In this research, the points in $\mathcal{P}_{\mathcal{W}}$ are coplanar but not collinear. According to [72], the solution of Eq. (2.10) is unique. This is the reason why the proposed system is free from the multi-solution problem, which is unlike the VFM systems troubled by the rotation ambiguity problem [27, 52, 53]. This is a superiority of the proposed system over the planar VFM systems [1, 18].

Note that the LiDAR pose is specified by the definitions of $\{G\}$ and $\{L\}$. It is a common method to define $\{G\}$ by predefining the vertices [1]. However, the predefined vertices are optional in both the AprilTag system [1] and the proposed system. Without the predefined vertices, the definition of $\{G\}$ is missing. Consequently, the AprilTag system [1] will only output the image coordinates of the vertices in the image plane and the proposed system only outputs the 3D features w.r.t. $\{L\}$. In the implementation, the predefinition of vertices is customizable since we want to leave the authority of defining $\{G\}$ to the users as the AprilTag system [1] does.

2.5 Experimental Validation

2.5.1 Experimental Setup

To qualitatively evaluate IFM, two LiDAR models—Livox Mid-40 (solid-state LiDAR) and VLP-16 (mechanical LiDAR)—are employed, and patterns from three popular VFM systems, AprilTag [1], CCTag [19], and ArUco [18], are tested. To further verify the pose estimation accuracy of the IFM system, we compare the pose estimation result given by our system with the ground truth provided by the OptiTrack Motion Capture (MoCap) system (See Fig. 2.10). The MoCap system is composed of 16 OptiTrack cameras and provides the 6-DOF pose information of the predefined rigid body at 100 Hz. The low-cost solid-state LiDAR, Livox Mid-40, is employed to scan a marker pasted on the wall. The marker (ID = 0), with the size of 17.2 cm×17.2 cm, belongs to the *tag36h11* family of AprilTag 3 [1] and is printed on letter-sized paper. The location of the marker’s center is at (3.620, 0.00, 0.485) m w.r.t. $\{G\}$. Moreover, the rosbag provided by LiDARTag [9] is utilized to quantitatively evaluate IFM using the Velodyne ULTRA Puck LiDAR (mechanical LiDAR) with the AprilTag marker.

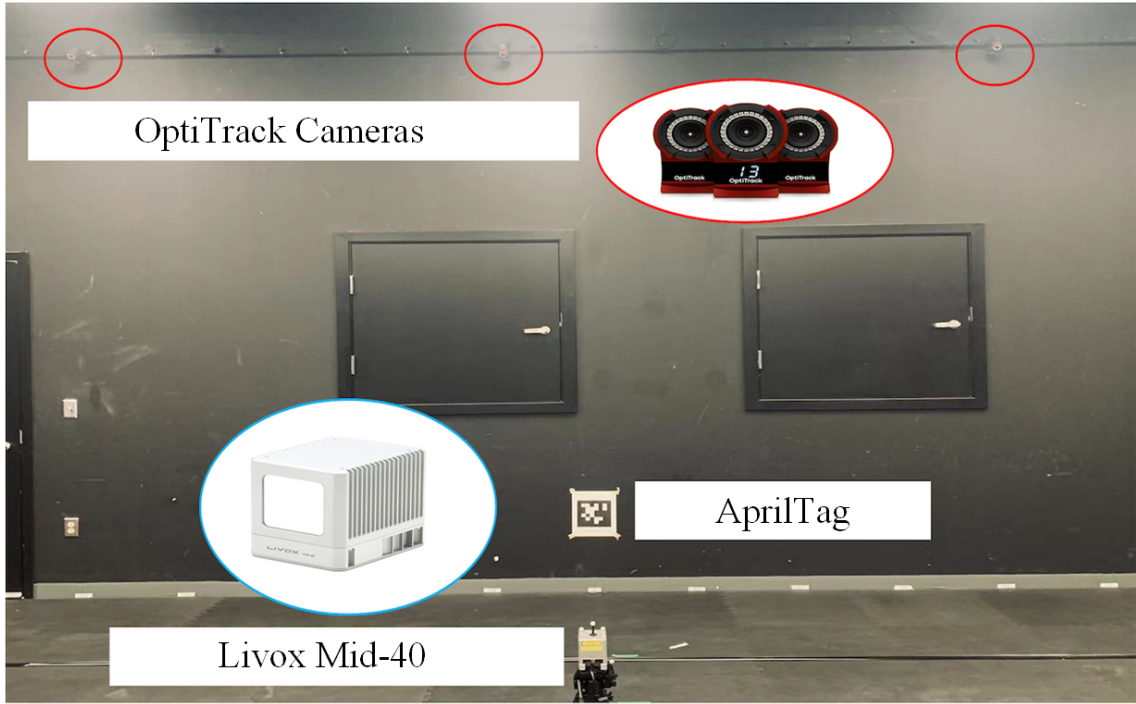


Figure 2.10: An illustration of the experimental setup.

2.5.2 Qualitative Evaluation

To qualitatively demonstrate the flexibility of the proposed IFM system, Fig. 2.11 is presented. Specifically, Fig. 2.11(a) corresponds to the scenario shown in Fig. 2.1(a): a Livox Mid-40 is scanning an AprilTag grid and an ArUco grid. All the markers, 35 AprilTags (family: tag36h11, marker size: 17.2 cm $\times$ 17.2 cm), and 4 ArUcos (family: 4 $\times$ 4, marker size: 16 cm $\times$ 16 cm), are detected. Fig. 2.11(b) corresponds to Fig. 2.1(b): a Livox Mid-40 is scanning three CCTag (family: 3 rings, marker radius: 7.2 cm) attached to the wall. All the markers are detected. Fig. 2.11(c) corresponds to Fig. 2.1(c): a VLP-16 is scanning an ArUco (family: original, maker size: 40 cm $\times$ 40 cm). The marker is detected.

As seen in Fig. 2.11, the usage of the IFM system is as convenient as the VFM systems. In particular, the user can place the letter-size markers [1,18] densely to compose a marker

grid, as shown in Fig. 2.11(a), as well as attach some non-square markers [19] to the wall freely, as shown in Fig. 2.11(b). In summary, there is no spatial restriction on marker placement. It should be noted that if the angular resolution of the LiDAR is relatively large, large marker size and simpler marker pattern are recommended for the sake of the intensity image quality. For instance, the vertical angular resolution of the VLP-16 is 1.33° , thus a 40 cm $\times$ 40 cm original ArUco [18] is adopted, as shown in Fig. 2.11(c).

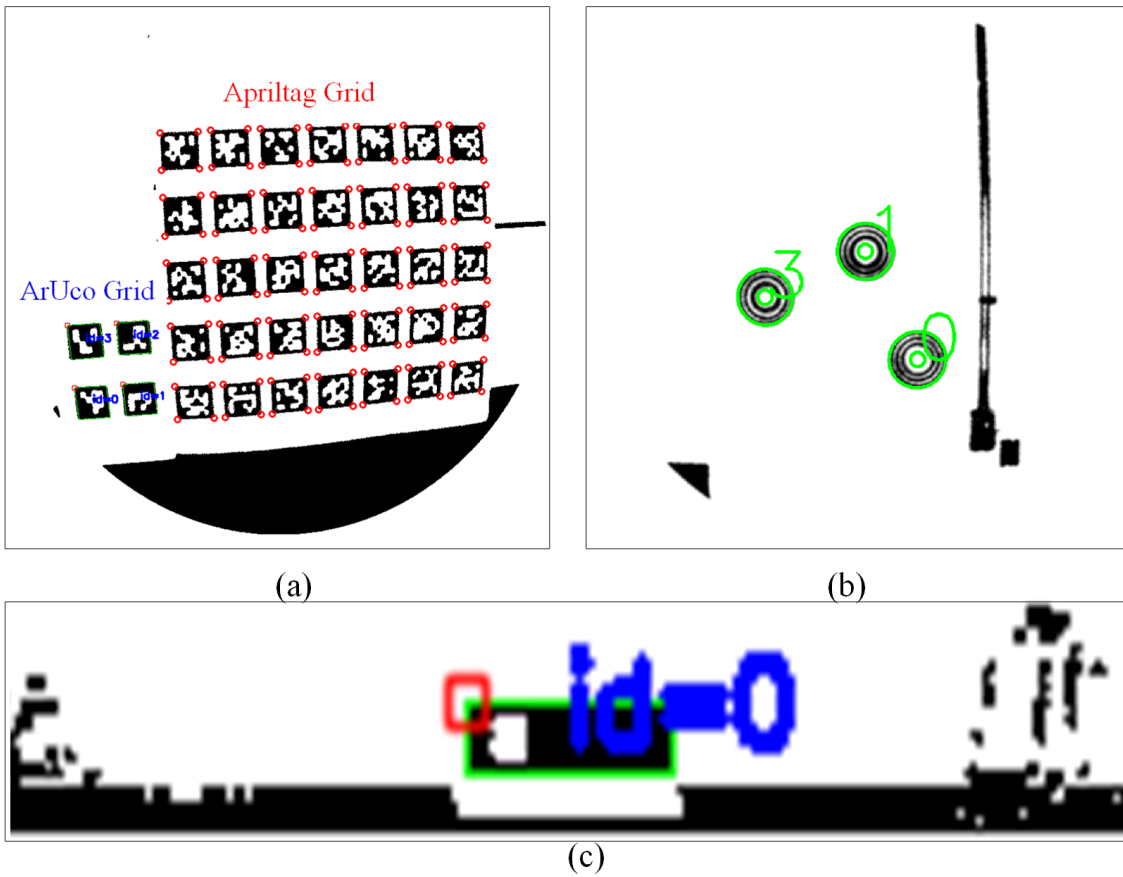


Figure 2.11: Marker detection results on the preprocessed intensity images.

2.5.3 Quantitative Evaluation

As a reminder, the experimental setup is shown in Fig. 2.10. Firstly, the orientation of the LiDAR (Livox Mid-40) is fixed, with the roll, pitch, and yaw angles approximately being zeros. Only the distance from the marker’s plane to the LiDAR’s $O_L Y_L - O_L Z_L$ plane is changed. Then, to compare the conventional VFM system with the proposed system, we test AprilTag 3 [1] with a camera (Omnivision OV7251) under the same experimental setup. To intuitively demonstrate the changing trend of accuracy w.r.t. the distance, we present the histogram of errors in Fig. 2.12. In particular, the error refers to the absolute difference between the measurement and the ground truth. A detailed table containing all the measurements and ground truth from Fig. 2.12 is available in Table 3.3, where the vanilla IFM refers to the approach introduced in this section.

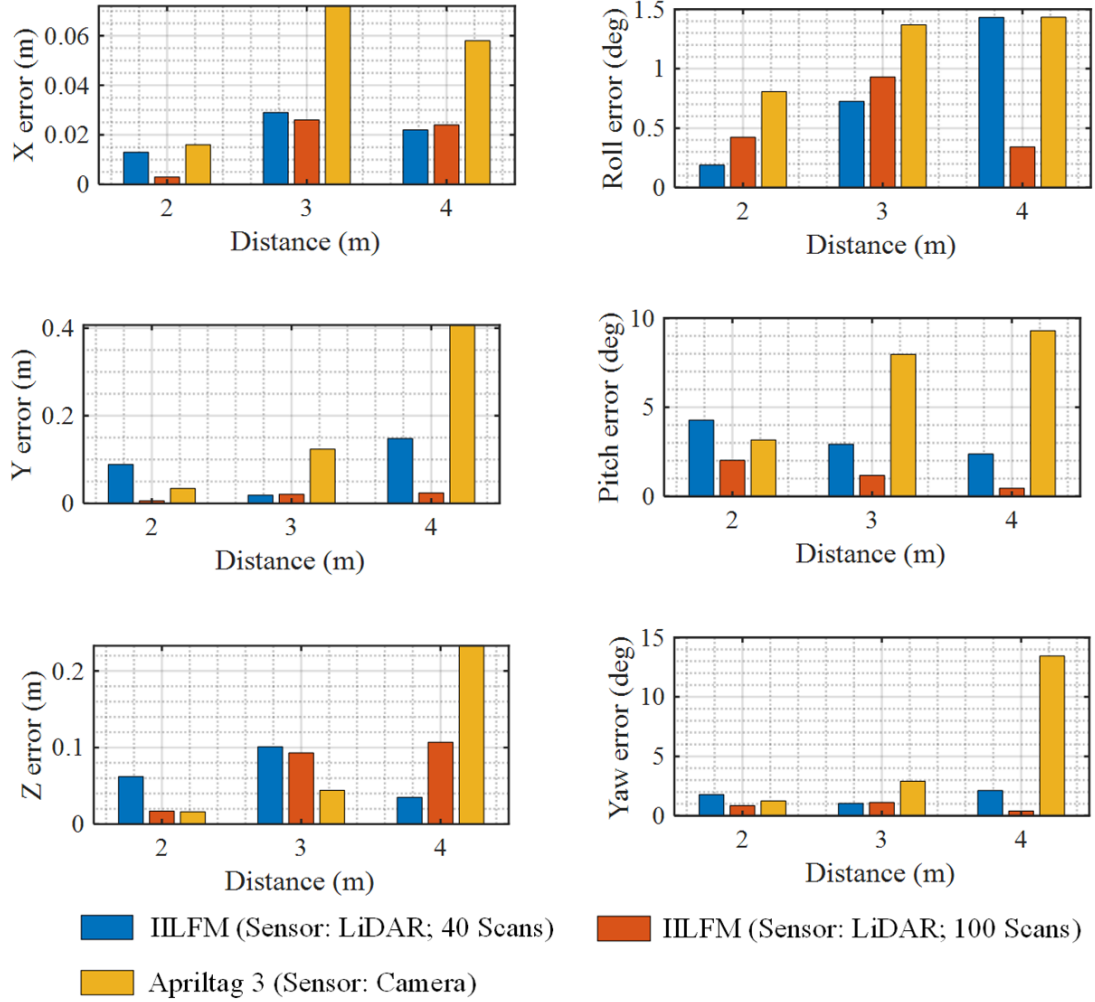


Figure 2.12: Pose estimation accuracy of the IFM system and the AprilTag 3 system at different distances.

Three issues are illustrated in Fig. 2.12: (1) When more LiDAR scans are utilized, the pose estimation accuracy is slightly boosted. The reason behind it is that more LiDAR scans indicate a higher coverage percentage in the FoV [14], which implies better intensity image quality. (2) The IFM system shows comparable accuracy as the VFM system. (3) Unlike the VFM system [1,16,17], the pose estimation accuracy does not degrade evidently

as the distance increases.

Thereafter, the distance from the marker to the LiDAR is set as 2 meters and only the rotation of the LiDAR is adjusted (See Table 2.1 where the number of scans = 40).

Table 2.1: Pose estimation accuracy of the IFM system with different Euler angles.

Setup	Term	Ground Truth	IFM	Error
pitch $\approx -15^\circ$	x (m)	1.629	1.661	-0.032
	y (m)	-0.066	-0.011	-0.055
	z (m)	0.618	0.699	-0.081
	roll (deg)	-2.673	-4.936	2.263
	pitch (deg)	-15.060	-21.145	6.085
	yaw (deg)	-0.171	0.546	-0.717
pitch $\approx 15^\circ$	x (m)	1.684	1.622	0.062
	y (m)	-0.063	-0.088	0.026
	z (m)	0.590	0.660	-0.070
	roll (deg)	3.657	0.900	2.757
	pitch (deg)	14.849	10.961	3.888
	yaw (deg)	1.136	-1.921	3.057
yaw $\approx -15^\circ$	x (m)	1.638	1.612	0.027
	y (m)	-0.618	-0.237	-0.381
	z (m)	0.610	0.585	0.025
	roll (deg)	-0.009	-0.792	0.783
	pitch (deg)	-0.662	-2.088	1.426
	yaw (deg)	-15.387	-17.937	2.550
yaw $\approx 15^\circ$	x (m)	1.627	1.632	-0.005
	y (m)	-0.194	-0.146	-0.048
	z (m)	0.609	0.553	0.057
	roll (deg)	0.490	-1.994	2.484
	pitch (deg)	-0.388	-0.359	-0.028
	yaw (deg)	14.924	10.678	4.246

By comparing the errors in Fig. 2.12 and Table 2.1, it is seen that when the plane of the LiDAR is angled towards the marker plane, the pose estimation performance of the IFM system is as good as that when the LiDAR is perpendicular to the marker.

Although the results in Fig. 2.12 and Table 2.1 might be inferior to other high-accuracy solutions, such as total stations and prisms [55], the merit of the proposed framework lies in its low cost, flexibility, and convenience.

To validate the pose estimation accuracy of our system on the mechanical LiDAR as well as to compare the proposed system with the state-of-the-art LFM system, LiDARTag [9], which is also the only existing fiducial marker system for the LiDAR as far as we know, Table 2.2 is presented. Specifically, we use the rosbag (ccw_10m.bag) provided by [9] as the benchmark on which we conduct the comparison. ccw_10m.bag records the raw data of a 32-Beam Velodyne ULTRA Puck LiDAR scanning a 1.22 m×1.22 m AprilTag (tag16h6), from a distance of 10 meters while the relative angle between the LiDAR plane and the marker is around 45°. Only the vertices estimation is compared on account that [9] solely provides the ground truth of the vertices and the vertices estimation is a follow-up process after the pose estimation in the LiDARTag system [9, 25].

Table 2.2 illustrates that the IFM system is slightly inferior to LiDARTag in terms of accuracy. This is mainly because the LiDARTag system estimates the pose by finding the transmission that projects points inside the marker cluster into a predefined template (bounding box) at the origin of LiDAR [9, 25], while our system adopts the vertices to compute the pose directly just as the AprilTag [1] and ArUco [18] systems do.

Table 2.2: Comparison of the IFM system and LiDARTag.

System	Vertex	x (m)	y (m)	z (m)	Error (m)
Ground Truth [9]	1	9.739	0.758	-0.161	–
	2	9.940	0.072	-0.732	–
	3	10.272	-0.414	-0.032	–
	4	10.072	0.271	0.539	–
LiDARTag [9]	1	9.736	0.762	-0.174	0.015
	2	9.944	0.066	-0.730	0.009
	3	10.271	-0.405	-0.016	0.019
	4	10.063	0.291	0.539	0.022
IFM	1	9.728	0.766	-0.184	0.013
	2	9.963	0.052	-0.705	0.033
	3	10.307	-0.432	-0.016	0.044
	4	10.045	0.263	0.564	0.041

Our system utilizes fewer correspondences compared to [9, 25], and thus the ranging noise on a single point could have more effects on the pose estimation of our system. It should be noted that the intention of proposing the LFM system is to improve the real-world applications, such as SLAM [27], multi-sensor calibration [25], and AR [20]. As shown in Fig. 2.12, the proposed system outperforms the AprilTag system [1], which is already widely used in the above-mentioned applications. Moreover, the VFM systems [1, 18] use only four vertices of a marker (or a limited number of points if the marker is non-square). Note that it is definitely feasible to acquire more feature points from the inner coding area of the marker, however, this will increase the complexity of the VFM systems. Hence, following a simple but effective design concept, we also use only four vertices of a marker in our proposed system. If higher accuracy is needed, it is feasible to use the proposed marker detection information to find the point clustering of the marker in the point cloud and then input the point clustering into the pose estimation block of LiDARTag. However, as mentioned previously, the superiority of the IFM system is the flexibility and

extensibility. For example, marker detection is infeasible for the LiDARTag system [9] in the scenarios shown in Fig. 2.1 and Fig. 2.10 since the marker placement does not satisfy the requirement of LiDARTag, and in addition, the current version of LiDARTag does not support any non-square marker, such as CCTag [19].

2.5.4 Computational Time Analysis

A computational time analysis is conducted on a desktop with Intel Xeon W-1290P Central Processing Unit (CPU). The LiDARTag [9] runs at around 100 Hz on it. The time consumption of the marker detection of our system, which includes intensity image generation, preprocessing, 2D features detection, and computation of 3D features, mainly depends on the size of the intensity image and the embedded VFM system. Suppose that the embedded VFM system is AprilTag 3 [1], the marker detection takes around 7 ms (approx 143 Hz) in the case shown in Fig. 2.11(c) where $\Theta_a = 0.3^\circ$, $\Theta_i = 1.33^\circ$, and intensity image size $= 1201 \times 27$. For the case shown in Fig. 2.11(a), the marker detection takes around 25 ms (approx 40 Hz), where intensity $\Theta_a = \Theta_i = 0.05^\circ$ and image size $= 771 \times 591$. The time consumption of the following pose estimation process is around $1.8 \mu s$ as the closed-form solution can be obtained directly through SVD [72] (Refer to Section 2.4).

2.5.5 Limitations Analysis

There are some limitations to the application of LFM systems. First, the distance from the object (marker) to the LiDAR must exceed the minimum detectable range. This limitation is caused by the hardware attributes and it affects all the LiDAR applications not only the LFM systems. Secondly, to utilize the solid-state LiDAR, it is required to wait for the growth of the scanned area inside the FoV to obtain a relatively dense point cloud. Again this is a limitation caused by the hardware attributes. In contrast, the mechanical

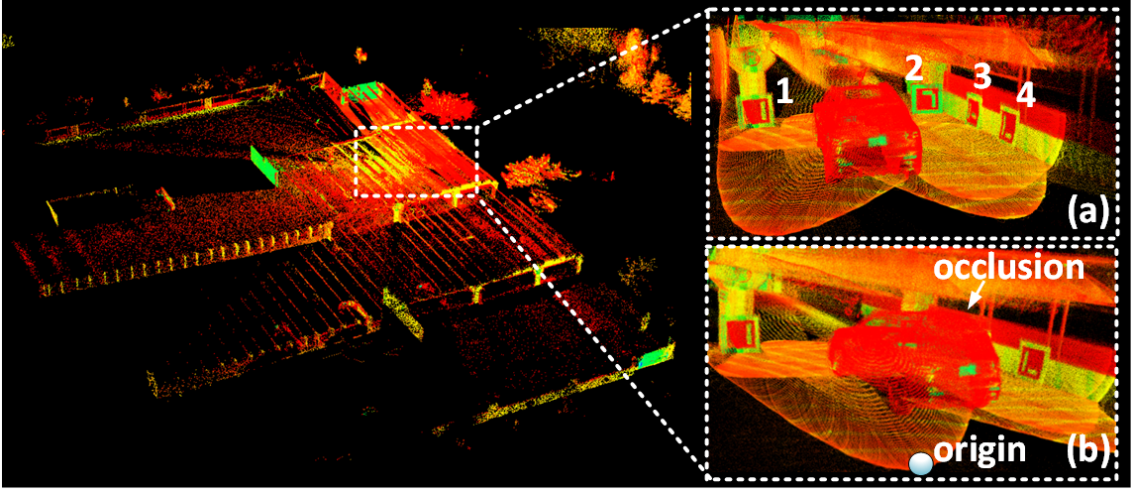


Figure 2.13: An illustration of the limitation of spherical projection for 3D maps. This map is constructed using Traj LO [10].

LiDAR only requires one LiDAR scan to work, however, a larger and simpler marker is recommended if the angular resolution is large. Finally, the false positives are occasionally found in the experiments while the issue of wrong ID detection (not exactly the same as the false positive but similar) is also reported in LiDARTag [9]. For the proposed system, the false positive rate is mainly determined by the embedded VFM system, thus novel VFM systems with lower false positive rates, such as [1, 18], are preferred. Moreover, due to the adoption of 3D-to-2D spherical projection, the vanilla IFM exhibits two limitations:

(1) IFM can only detect fiducials in a single-view point cloud and is not applicable to a 3D LiDAR map, as spherical projection only applies to a single-view point cloud [15]. Take the 3D map shown in Fig. 2.13 as an example. Unless we manually adjust the perspective to view the map as shown in Fig. 2.13(a), the four tags are not simultaneously visible due to occlusion, as seen in Fig. 2.13(b).

(2) As the distance between the tag and the LiDAR increases, the tag’s projection size decreases until it becomes too small to be detected.

The next section addresses these two limitations caused by the 3D-to-2D spherical projection. Moreover, the vanilla IFM employs a constant threshold to process intensity images (See Fig. 2.7(c)), which is sufficient for static scenes with trivial or no viewpoint changes. However, the marker detection of the vanilla IFM is not robust when the viewpoints change in a wide range, which hinders its application to in-the-wild multiview point cloud registration. An adaptive threshold marker detection method is developed to address this problem in Section 4.2.1.

3 Improvements to Vanilla IFM

Localization

3.1 Overview

This chapter introduces an algorithm that addresses the two limitations of the vanilla IFM, caused by the adoption of 3D-to-2D projection, as introduced in Section 2.5.5. The overview of the improvements to the vanilla IFM is shown in Fig. 3.1. First, as seen in Fig. 3.1(a), the proposed algorithm extends thin-sheet LFM localization from single-view point clouds to 3D LiDAR maps. One might consider such a solution to localize 3D fiducials in the map: incorporating marker detection into the front end of SLAM and detecting fiducials during SLAM. For example, [73] applies this solution by integrating AprilTag [1] detection into a visual SLAM method (Vins-mono [74]). The feasibility of this solution is not denied. However, the concern is that it involves low-level modifications to the front end of the SLAM method. Therefore, it is not generalizable, as different LiDAR-based SLAM frameworks [10,12] have various front-end pipelines. Consequently, every time the baseline SLAM method changes, the modification must be made again. In contrast, as shown in Fig. 3.1(a), the proposed algorithm is a generalizable method that can be applied to maps built using different methods.

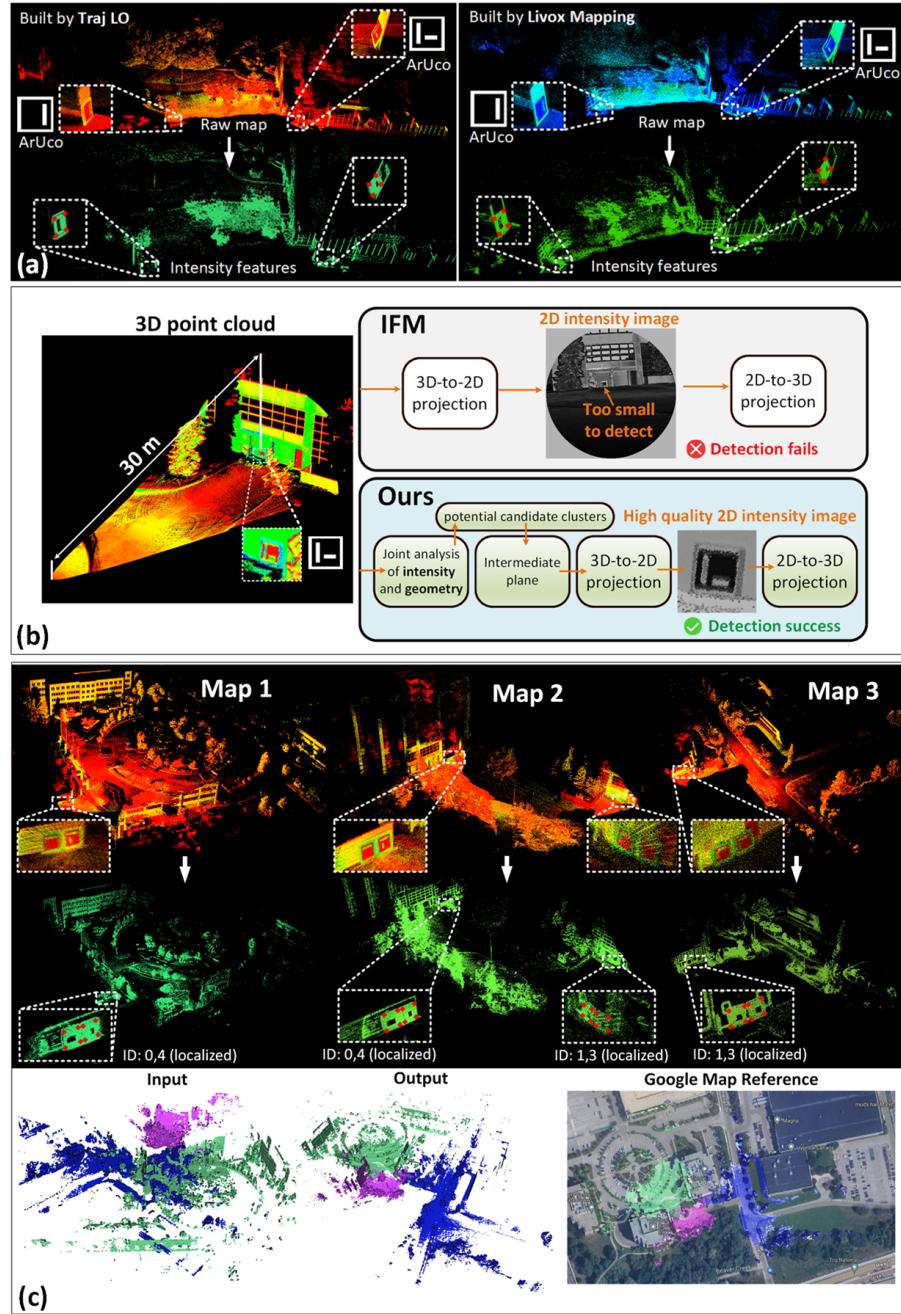


Figure 3.1: An overview of the improvements to the vanilla IFM.

Second, compared to the vanilla IFM, which directly projects the 3D point cloud onto an image plane, as shown in Fig. 3.1(b), the proposed method jointly analyzes the point cloud

from both intensity and geometry perspectives to extract potential candidate clusters. By introducing an intermediate plane to project these clusters, higher-quality intensity images are obtained. As a result, the proposed method improves the range over which markers can be detected. In this example, the distance is 30 meters, and the marker size is $69.2 \text{ cm} \times 69.2 \text{ cm}$, whereas the proposed method can localize the marker, but the vanilla IFM cannot. Third, as seen in Fig. 3.1(c), the extension of LFM localization to 3D maps enables downstream tasks such as 3D map merging. These low-overlap 3D maps are merged using marker localization results from the proposed method and the algorithm in Section 4.

3.2 Joint Analysis of Point Clouds from Intensity and Geometry Perspectives

3.2.1 Downsampling Based on 3D Intensity Gradients

A synthesis point cloud (see Fig. 3.2) with a simple scene is utilized to explicitly illustrate the design purpose and results of each operation in our pipeline. The two presenters are holding two different AprilTags [1]. As illustrated in the top view, observing along the X-axis of the global coordinate system, the back subpoint cloud is totally blocked by the front one. Thus, although this is not a 3D LiDAR map, it possesses the most important feature of a 3D LiDAR map in this research: the spherical projection (Eq. (2.7)) is not applicable. Note that Fig. 3.2 is only used to explicitly present the results and explain design purposes due to its simplicity. The ultimate objective is to localize the fiducial markers on a 3D map, which represents a larger and more complex scene.

The LFMs are sheets of thin paper that are not spatially distinguishable from the attached planes. Thus, it is infeasible to adopt previous geometric features-based 3D object detection methods [15, 58] to find the LFMs. Therefore, a new feature extraction

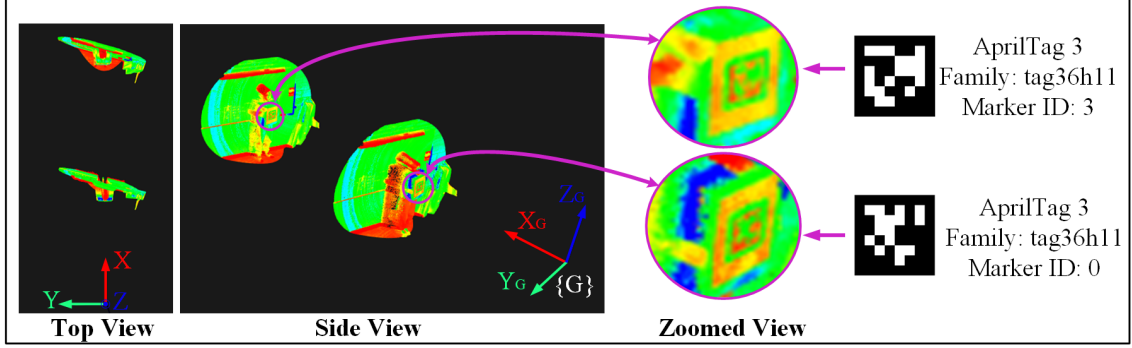


Figure 3.2: The example used to explain the design purpose and result of each step.

solution is developed to address this problem. The analysis is given as follows. A fiducial marker is composed of a black-and-white pattern and, as a result, presents as a high-intensity contrast object in the view of a LiDAR (see the zoomed views of Fig. 3.1, Fig. 2.13, and Fig. 3.2). This indicates that the point cloud can first be analyzed from the intensity perspective.

In particular, downsampling is conducted on the raw point cloud based on the 3D intensity gradients. The intensity is taken as a function $I(\mathbf{p})$ of the 3D coordinates $\mathbf{p} = [x, y, z]^T$. Suppose that the given 3D point/location is $\mathbf{p}_0 = [x_0, y_0, z_0]^T$, the point set composed of the neighbouring n points around $\mathbf{p}_0$ is defined as $\mathcal{P}_I = \{\mathbf{p}_1, \dots, \mathbf{p}_n\}$. In practice, the following equation is used to approximate $I(\mathbf{p})$:

$$\hat{\mathbf{I}}(\mathbf{p}) = \mathbf{C}^T \mathbf{p} + b, \quad (3.1)$$

where $\mathbf{C} \in \mathbb{R}^{3 \times 1}$ is the coefficient vector and $b \in \mathbb{R}$ is the intercept. Since the intensity values of points in $\mathcal{P}_I$ are known, $\mathbf{C}$ can be estimated by solving the following model fit (least square) problem:

$$\arg \min_{\mathbf{C}^*, b^*} \sum_{i=1}^n \left\| \hat{\mathbf{I}}(\mathbf{p}_i) - I(\mathbf{p}_i) \right\|^2. \quad (3.2)$$

Coefficient regression [65] is performed to solve this problem. Let the design matrix be:

$$\mathbf{D} = \begin{bmatrix} 1 & \Delta x_1 & \Delta y_1 & \Delta z_1 \\ 1 & \Delta x_2 & \Delta y_2 & \Delta z_2 \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ 1 & \Delta x_n & \Delta y_n & \Delta z_n \end{bmatrix}, \quad (3.3)$$

where $\mathbf{D} \in \mathbb{R}^{n \times 4}$. $\Delta x_i = (x_i - x_0)$, $\Delta y_i = (y_i - y_0)$, $\Delta z_i = (z_i - z_0)$. The first column is the constant term, representing the intercept. Centering is applied to the design matrix, as the focus is on the local gradients of intensity value at a given 3D position within $\mathcal{P}_I$. Since the goal is to fit the intensity value as a function of 3D positions, the response variable vector would be:

$$\mathbf{I}_{in} = [\Delta I_1, \Delta I_2, \dots, \Delta I_n]^T, \quad (3.4)$$

where $\Delta I_i = (I_i - \bar{I})$ with $\bar{I} \in \mathbb{R}$ being the mean of the intensity values of the points in $\mathcal{P}_I$. Again, centering is performed as the focus is on the local gradients. Suppose that the regression coefficient vector $\mathbf{E} \in \mathbb{R}^{4 \times 1}$ is defined as:

$$\mathbf{E} = \begin{bmatrix} \mathbf{C}^* \\ b^* \end{bmatrix}. \quad (3.5)$$

Following [65], the regression coefficient vector can be calculated by:

$$\mathbf{E} = (\mathbf{D}^T \mathbf{D})^{-1} \mathbf{D}^T \mathbf{I}_{in}. \quad (3.6)$$

$\hat{\mathbf{I}}(\mathbf{x})$ is obtained by substituting the elements of $\mathbf{E}$ into Eq. (3.1). Denote the 3D intensity gradients at the given 3D position as $\nabla I \in \mathbb{R}^{1 \times 3}$:

$$\begin{aligned} \nabla I &= \frac{\partial \hat{\mathbf{I}}(\mathbf{p})}{\partial \mathbf{p}} \\ &= \frac{\partial \mathbf{C}^{*T} \mathbf{p}}{\partial \mathbf{p}} + \frac{\partial b^*}{\partial \mathbf{p}} \\ &= \mathbf{C}^{*T} \end{aligned} \quad (3.7)$$

The direction of ∇I of a given point indicates the direction where the intensity has the fastest decline and the norm, $|\nabla I|$, implies the rate of descent. Inspired by LOAM [14], which selects a point as a feature if its geometric curvature is larger than a threshold, a 3D point is preserved if its $|\nabla I|$ is larger than a threshold in the downsampling procedure. The result following downsampling execution is depicted in Fig. 3.3. As seen, the majority of unnecessary points are filtered out.

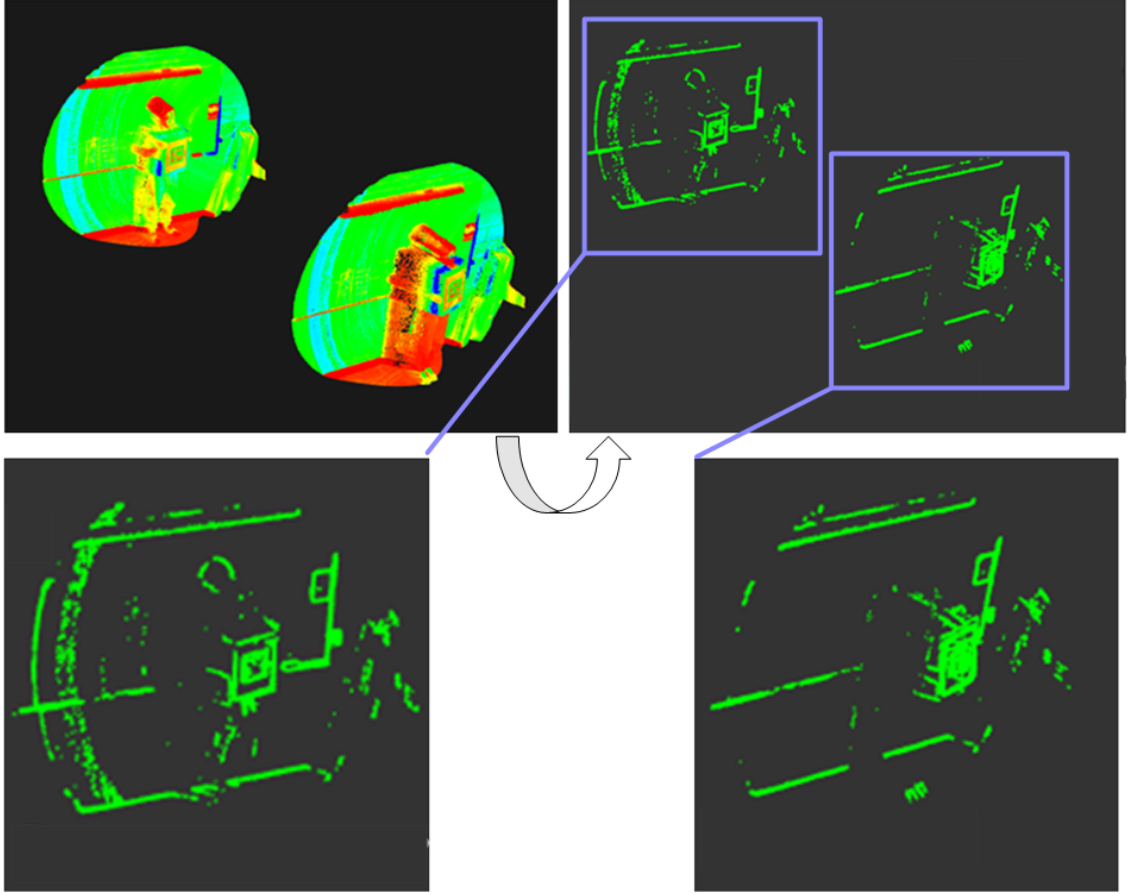


Figure 3.3: The effect of applying downsampling from the intensity perspective.

3.2.2 Spatial Distribution Analysis of Downsampling Result

The downsampling preserves the points belonging to the outlines of all objects with high intensity-contrast. In this section, the point cloud is analyzed from a geometric perspective. The foundation of doing so is the fact that the points belonging to the fiducial markers will be isolated from those of the other objects after the downsampling (See the zoomed view of Fig. 3.3). This is due to the design of the marker’s pattern as shown in Fig. 3.4.

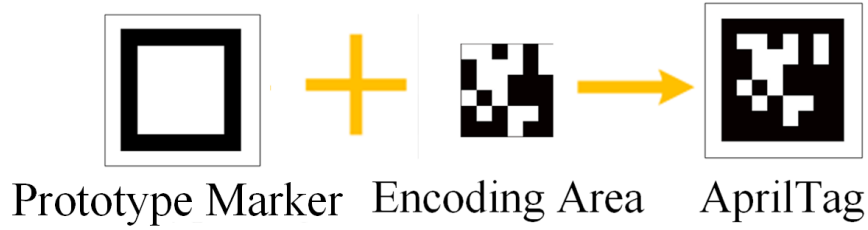


Figure 3.4: A diagram to illustrate the design of a typical square fiducial marker [1].

In particular, a square fiducial marker is a combination of the prototype marker (a black frame inside a white frame) and the encoding area. The white regions of the prototype marker naturally have higher intensity values than the black regions, and thus, the prototype marker after the downsampling is rendered as a square double-ring that isolates the points inside the coding area from the environment. The isolation makes the points belonging to the markers spatially distinguishable from those of the other objects. Thus, we employ the method introduced in [75] to further segment the downsampling result into clusters. Each cluster is represented by an Oriented Bounding Box (OBB) in Fig. 3.5.

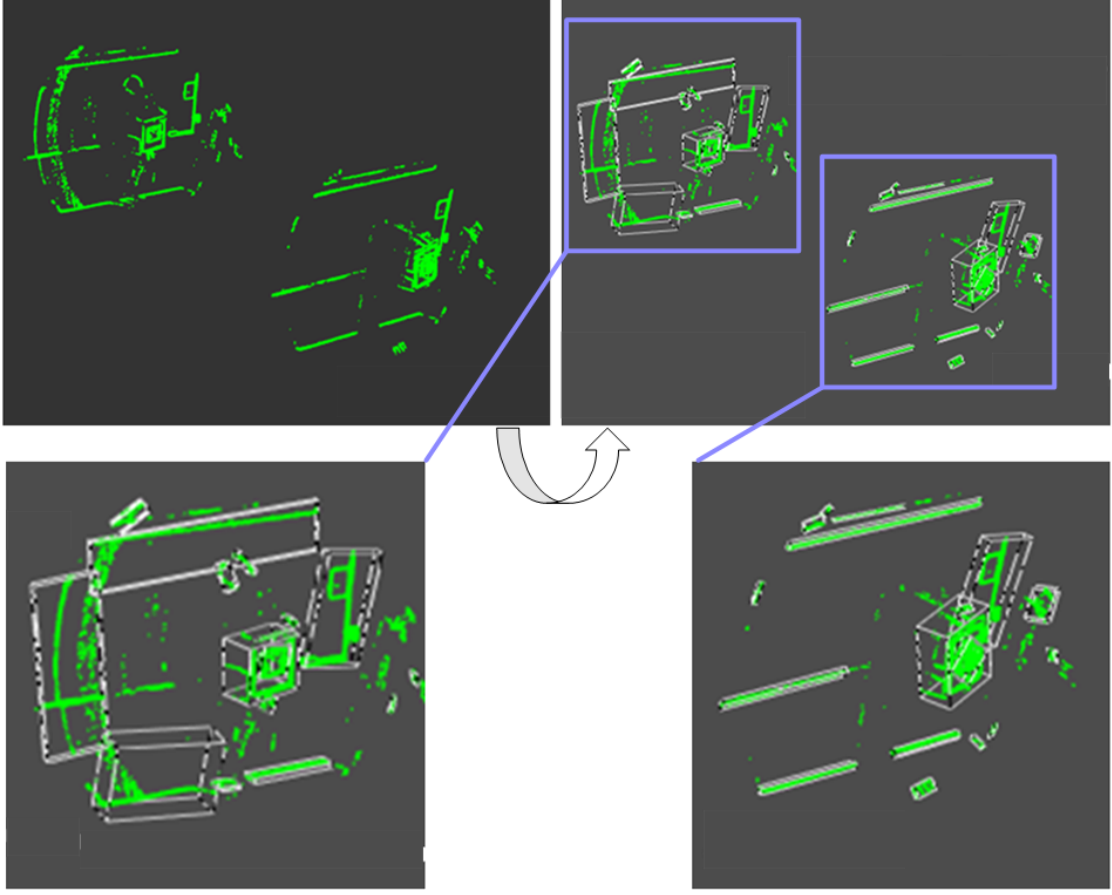


Figure 3.5: The effect of clustering on the downsampling result.

3.2.3 Filtering Out Unwanted Clusters

As depicted in Fig. 3.5, there are many OBBs after clustering. In this section, the geometric characteristics of each OBB are analyzed to verify if it has the potential to be a fiducial marker. Specifically, the bounding box of a cluster needs to satisfy two criteria to be recognized as a valid candidate.

Criterion 1. The first criterion is subject to the marker size:

$$\sqrt{2a^2 + t_M^2} \leq L_{OBB} \leq \sqrt{4a^2 + t_M^2}, \quad (3.8)$$

where $L_{OBB} = \sqrt{l^2 + w^2 + h^2}$ is the cuboid diagonal of the OBB with l , w , and h ($h \leq t_M$)

being the length, width, and height, respectively. a denotes the side length of the marker. t_M is the trifling thickness of the marker.

Explanation of Criterion 1. The trivial height is neglected for now, and the possible size range of the OBB is considered in 2D space. Fig. 3.6 shows the 2D geometric relations.

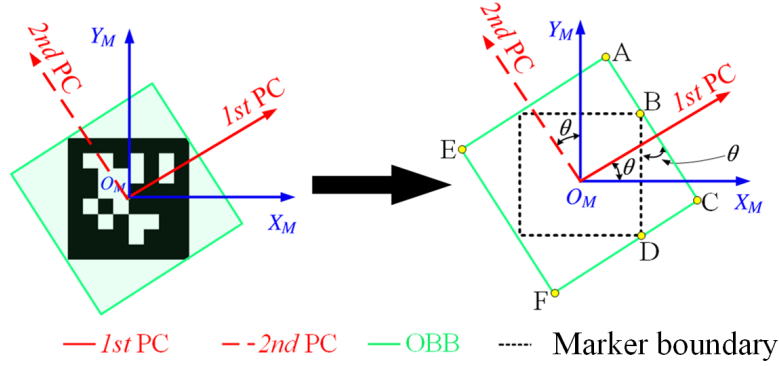


Figure 3.6: A diagram of the possible OBB size for a given marker.

Principal Component Analysis (PCA) is performed on the distribution of points belonging to the marker. Suppose that $O_M - X_M - Y_M$ is the marker coordinate system and θ is the angle between the first Primary Component (PC) and the x-axis. On account of the fact that the first PC and second PC are perpendicular to each other [76], the angle between the second PC and the y-axis is also θ . In addition, $\angle DBC = \theta$. Thus, the side length of the OBB, expressed as $a(\cos\theta + \sin\theta)$, falls within the range $[a, \sqrt{2}a]$, given that $\theta \in [0, 2\pi]$. Furthermore, the area of the 2D OBB, S_{OBB} , is in the following range:

$$a^2 \leq S_{OBB} \leq 2a^2. \quad (3.9)$$

Returning to 3D space, the marker's thickness, δ , is also taken into account. The first criterion becomes Eq. (3.8).

Criterion 2. The second criterion is shown in Eq. (3.10):

$$1/1.5 \leq l/w \leq 1.5. \quad (3.10)$$

Explanation of Criterion 2. This criterion is based on the fact that the shape of the marker is square and the OBB projection on the plane of length and width is also square. Ideally, $l \approx w$. Whereas in the real world, the marker cannot be perfectly scanned by LiDAR, and LiDAR also has ranging noise. As a result, the shape of the OBB on the length and width plane could be distorted. Hence, the requirement on $l \approx w$ is relaxed as shown in Eq. (3.10). It is presented in Fig. 3.7 that the unwanted OBBs are filtered out after the operation introduced in this section.

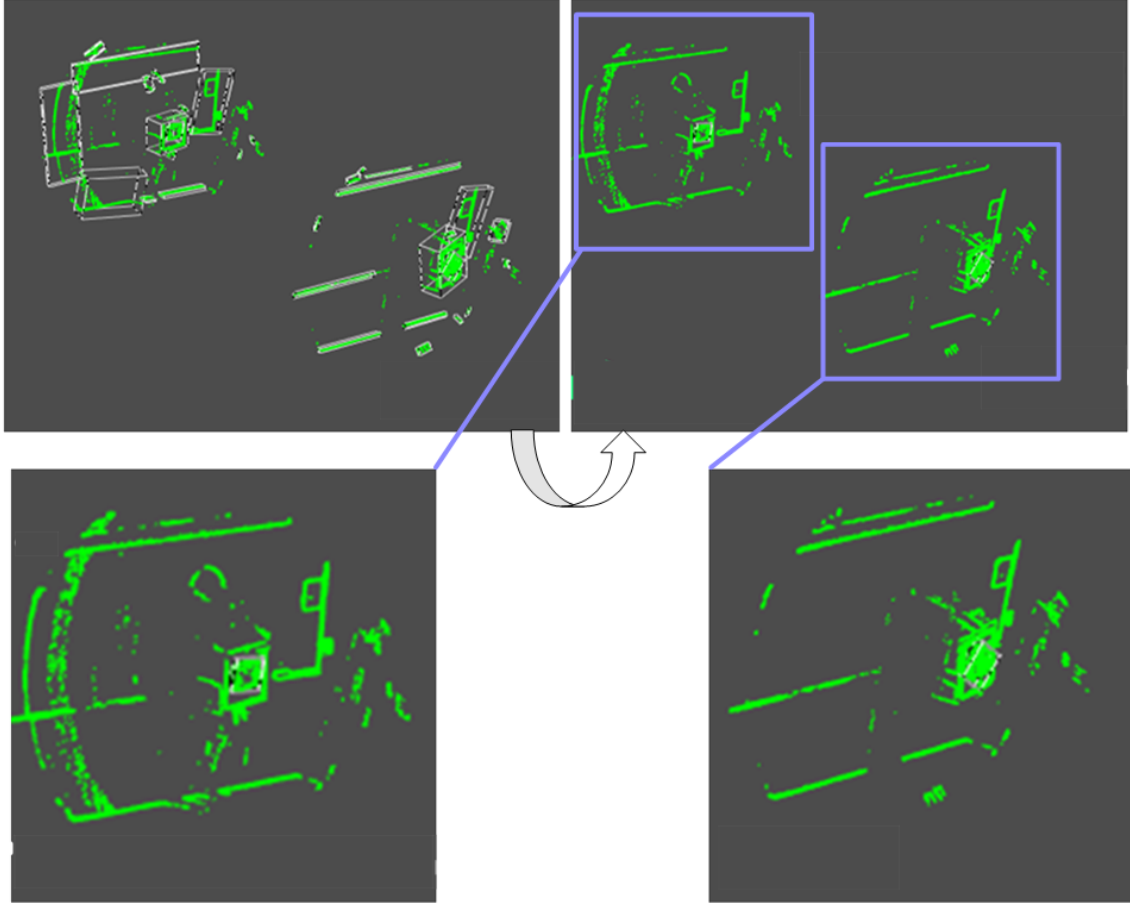


Figure 3.7: The effect of filtering out the unwanted clusters.

3.3 Marker Localization via Intermediate Plane

After analyzing the point cloud from the 3D intensity gradients and 3D geometric perspectives, locations of objects with high intensity-contrast and shapes/sizes similar to the fiducial marker are obtained. These locations (OBBs) are then inspected in the raw point cloud. When extracting points falling into the OBBs, a buffer is adopted to extend the OBBs, preserving more regions around an OBB in case it does not completely cover the fiducial marker. This indicates that, for each OBB, the pose (position and orientation),

$\mathbf{T}_{OBB}^1$, is kept while the size is enlarged by multiplying the length, width, and height with an amplification factor, t_b . The recommended value of t_b is twice the marker’s side length based on our experiment.

3.3.1 Motivation for Adopting the Intermediate Plane

An intermediate plane-based method is proposed to determine if an OBB contains a fiducial marker. There are two reasons for adopting the intermediate plane. First, as seen in Fig. 3.8, although the points belonging to the candidate markers are extracted from the raw point cloud, unfortunately, the spherical projection (Eq. (2.7)) cannot be applied in this case, as occlusion still exists when observing the point cloud from the origin. (Review Fig. 3.2 if needed).

¹ $\mathbf{T}_{OBB}$ denotes the transmission from the world coordinate system $\{G\}$ to the OBB frame. The OBB frame refers to a coordinate system whose X, Y, and Z axes are parallel to the length, width, and height of the OBB.

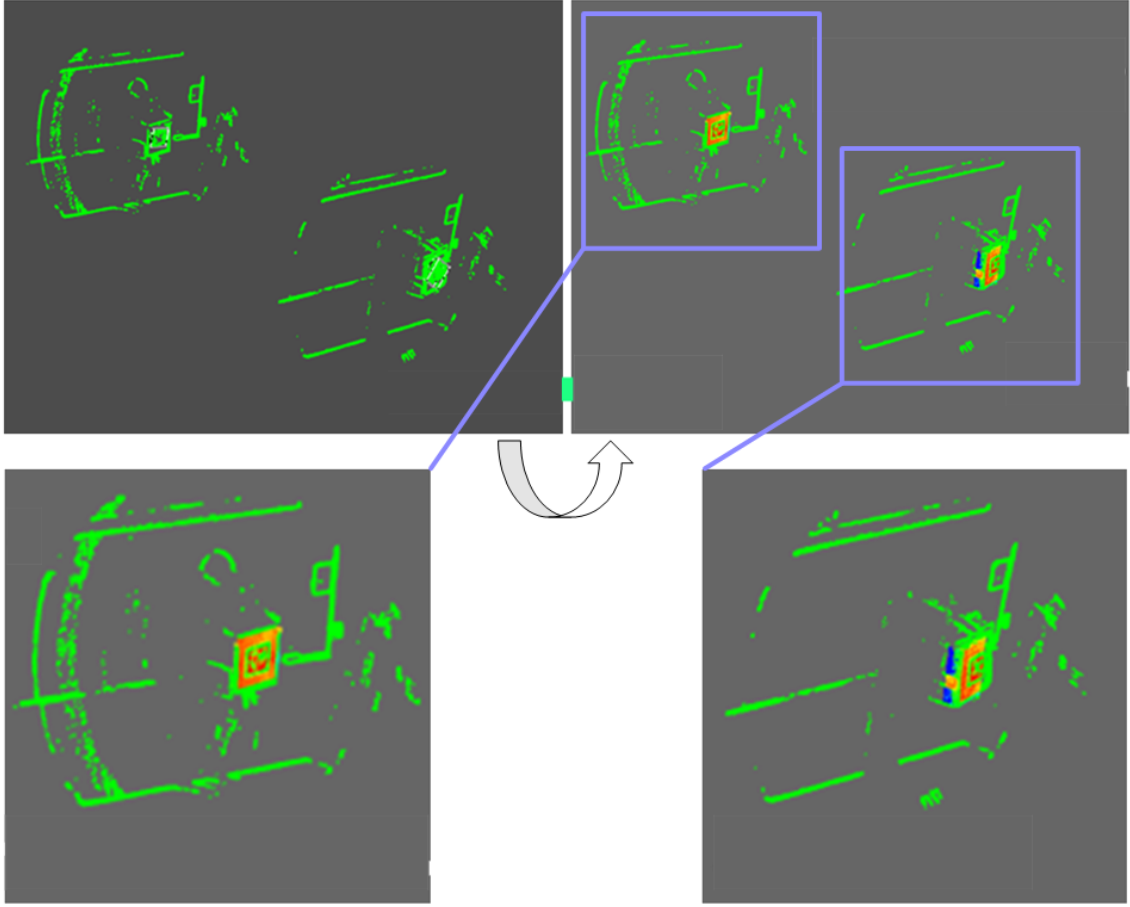


Figure 3.8: The result of extracting points falling into the preserved OBBs from the raw point cloud.

In contrast, as depicted in Fig. 3.9, the adoption of the intermediate plane resolves this problem by transferring the clusters onto the intermediate plane one by one, thereby addressing the occlusion issue.

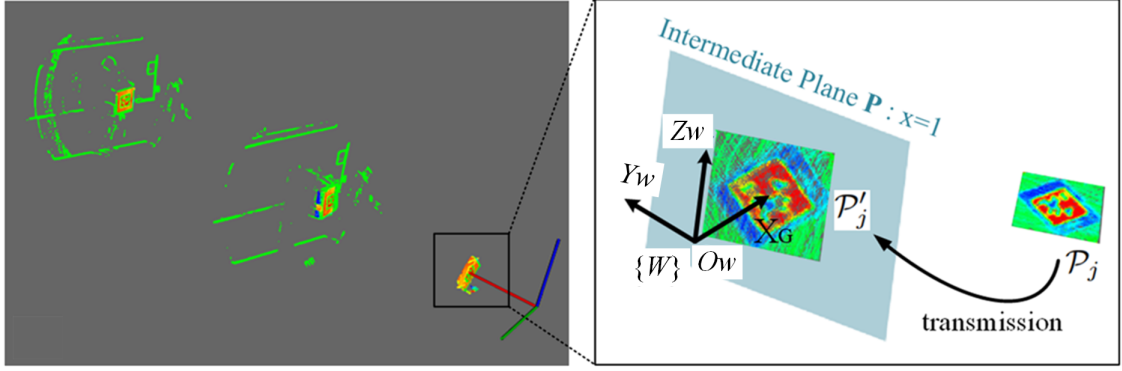


Figure 3.9: An illustration of how the intermediate plane helps solve the occlusion issue.

Second, as shown in Fig. 3.1(b), when the marker is far from the LiDAR, directly applying the spherical projection will result in a projection that is too small to be detected. However, as presented in Fig. 3.10, the introduction of the intermediate plane addresses this challenge. Specifically, transferring the point cloud of a fiducial marker onto the intermediate plane physically reduces the distance between the point cluster and the LiDAR, thereby resulting in a projection of better quality.

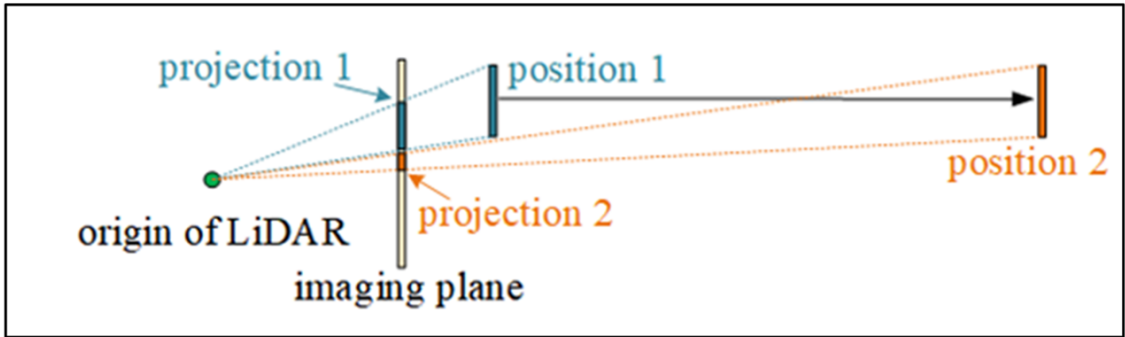


Figure 3.10: An illustration of how the intermediate plane helps solve the occlusion issue.

3.3.2 Utilization of the Intermediate Plane

Given that the pose of each OBB ($\mathbf{T}_{OBB}$) is known, the perspective for observing these candidate clusters can be adjusted using $\mathbf{T}_{OBB}$. In particular, the viewpoint is fixed at the origin, and the points extracted from each OBB are transferred one by one to the intermediate plane (denoted by $\mathbf{P}$), as shown in Fig. 3.9. The detailed transmission process is as follows. Define the point set of the j -th OBB as $\mathcal{P}_j$ and a point of $\mathcal{P}_j$ as $\mathbf{p}_j \in \mathbb{R}^3$. $\mathbf{p}_j$ is first transmitted to the origin of the world coordinate system $\{G\}$ through the inverse of $\mathbf{T}_{OBB}$:

$$\mathbf{p}_G = \mathbf{T}_{OBB}^{-1} \cdot \mathbf{p}_j = \mathbf{R}^{-1} \cdot \mathbf{p}_j - \mathbf{R}^{-1}\mathbf{t}, \quad (3.11)$$

where $\mathbf{p}_G$ is the point transmitted to the origin of $\{G\}$. $\mathbf{T}_{OBB} = [\mathbf{R}|\mathbf{t}]$ where $\mathbf{R}$ is the 3×3 orthogonal rotation matrix and $\mathbf{t}$ is the 3×1 translation vector. After transmitting all the points in $\mathcal{P}_j$ using Eq. (3.11), a new point set $\mathcal{P}_j^G$ is obtained. Then, all points belonging to $\mathcal{P}_j^G$ are transferred to the intermediate plane $\mathbf{P}$:

$$\mathbf{p}'_j = \mathbf{T}_{in} \cdot \mathbf{p}_G = \mathbf{R}_{in} \cdot \mathbf{p}_G + \mathbf{t}_{in}, \quad (3.12)$$

where $\mathbf{p}'_j$ is the point transmitted to the intermediate plane. $\mathbf{T}_{in} = [\mathbf{R}_{in}|\mathbf{t}_{in}]$ where $\mathbf{R}_{in} = \mathbf{I}_{3 \times 3}$ is an identity matrix and $\mathbf{t}_{in} = [1\text{m}, 0, 0]^T$ since the plane equation is $x = 1\text{m}$. Define the point set on the intermediate plane as $\mathcal{P}'_j$. Given that $\mathcal{P}'_j$ is a point cloud with no occlusions, marker localization in it is straightforward using the vanilla IFM. IFM returns the marker ID and the locations of the vertices labeled by index if the marker is found within the candidate OBB. Then, the locations of vertices can be transferred back to the original positions in the 3D map using the reverse processes of Eqs. (3.11-3.12), and the poses of markers (from $\{G\}$ to the marker coordinate system) are obtained by solving SVD. The final marker localization result is presented in Fig. 3.11.

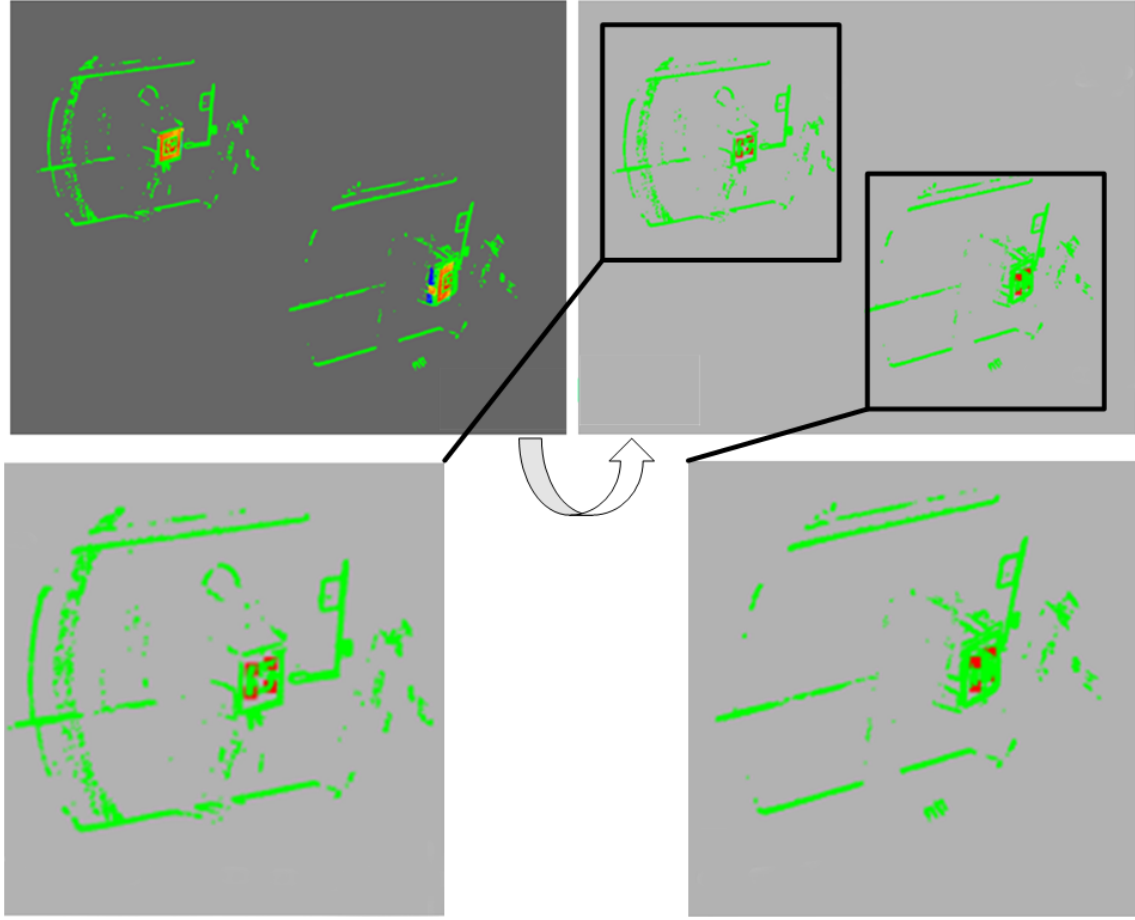


Figure 3.11: The fiducial marker localization result after applying the intermediate plane method. The vertices of the localized fiducial marker are rendered red.

Note that as depicted in Fig. 3.12, the rotation of $\mathbf{T}_{OBB}$ may be ambiguous, potentially causing a flipped intensity image due to the symmetry of the 3D OBB. In practice, both the raw intensity image and the flipped intensity image are checked, as a pattern in the coding library will not have its flipped version present [1, 18].

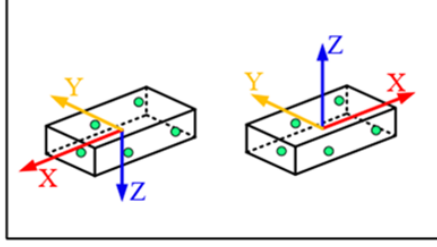


Figure 3.12: An illustration of the potential rotational ambiguity issue of $\mathbf{T}_{OBB}$.

3.4 3D LiDAR Map Construction

This section briefly introduces how the 3D LiDAR maps tested in Section 3.5 are constructed. In particular, two methods—Livox Mapping [12] and Traj LO [10]—are utilized in this dissertation for constructing large-scale 3D maps.

3.4.1 Livox Mapping

Livox Mapping [12] is the official implementation of LiDAR-based SLAM released by Livox, which is built on LOAM [14]. Suppose that the set of sampled data of the current scan is:

$$\mathcal{P}_{loam} = \{\{x_1, y_1, z_1, I_1\}, \{x_2, y_1, z_2, I_2\}, \dots, \{x_n, y_n, z_n, I_n\}\}. \quad (3.13)$$

To eliminate motion blur effects, the raw data is preprocessed by applying motion compensation. This involves either piecewise processing, where a frame is divided into subframes and processed in parallel to reduce the time interval, or linear interpolation, which modifies 3D point positions using the constant velocity assumption. Then, corner and plane feature points are extracted based on the curvature and normal of each point:

$$\begin{aligned} \mathcal{C}_{loam} &= \{\mathbf{p}_i \in \mathcal{P}_{loam} \mid \kappa(\mathbf{p}_i) > t_{cor}\}, \\ \mathcal{S}_{loam} &= \{\mathbf{p}_i \in \mathcal{P}_{loam} \mid \mathbf{N}(\mathbf{p}_i) < t_{sur}\}, \end{aligned} \quad (3.14)$$

where $\mathcal{C}_{loam}$ and $\mathcal{S}_{loam}$ are the sets of corner features and surface features, respectively. $\kappa(\mathbf{p}_i)$ and $\mathbf{N}(\mathbf{p}_i)$ are the functions to calculate the curvature and normal of $\mathbf{p}_i$, respectively. t_{cor} and t_{sur} are the predefined thresholds. Then, voxel grid filtering is applied to down-sample the feature points. The downsampled features are matched and registered with the features in the previous scan to estimate the relative pose between the current scan and the previous scan. Moreover, to eliminate the effect of dynamic objects, the features with too large residuals are removed. Finally, the preserved points are accumulated using the estimated poses to construct the 3D map.

3.4.2 Traj LO

Traj LO [10] is the state-of-the-art pure LiDAR-based SLAM method. The set of sampled data of the current scan is:

$$\mathcal{P}_{Traj} = \{\{x_1, y_1, z_1, I_1, t_1\}, \{x_2, y_1, z_2, I_2, t_2\}, \dots, \{x_n, y_n, z_n, I_n, t_n\}\}, \quad (3.15)$$

where t_i is the sampling time of $\mathbf{p}_i$. As seen, compared to Eq. (3.13), Traj LO also utilizes the temporal information. First, both piecewise processing and linear interpolation are performed for motion compensation. In particular, the current scan (time window) is divided into K equidistant segments of small resolution. Thus, the time interval becomes tiny by choosing a large value of K , enhancing the constant velocity assumption and allowing the method to handle rapid motion effectively. Traj LO proposes that the LiDAR trajectory (poses) is a function of time. Specifically, the LiDAR trajectory at the current time window, denoted by $\mathbf{T}(t)$, consists of K piecewise functions:

$$\mathbf{T}(t) = \{\tau(t_1), \tau(t_2), \dots, \tau(t_K)\}, \quad (3.16)$$

where $\tau(t_i)$ is the function representing the LiDAR trajectory at the i -th time segment. Traj LO applies point-to-plane ICP for registration without selecting feature. $\mathbf{T}(t)$ is

estimated by jointly minimizing the following energy function:

$$\arg \min_{\mathbf{T}(t)^*} (\mathcal{E}_{reg} + \mathcal{E}_{kine} + \mathcal{E}_{marg}), \quad (3.17)$$

where $\mathcal{E}_{reg}$, $\mathcal{E}_{kine}$, and $\mathcal{E}_{marg}$ represent the registration energy, kinematics energy, and marginalization energy, respectively. Since each 3D point is associated with a sampling timestamp, its location in the world coordinate system can be determined using $\mathbf{T}(t)$. The 3D map is constructed by transforming the observed 3D points into the world coordinate system.

3.5 Experimental Validation

3.5.1 Extending LFM Localization to 3D Maps:

Tackling the First Limitation

As a reminder, the first limitation of the vanilla IFM is its restriction to single-view point clouds, rather than 3D maps. This section demonstrates how the proposed method addresses this limitation. In particular, as presented in Table 3.1, the proposed method is tested on eight 3D maps built from rosbags collected using a Livox MID 40 LiDAR [12]. There are 2 to 4 ArUco markers, measuring $69.2 \text{ cm} \times 69.2 \text{ cm}$, in each map. The number of localized markers out of the total in each map is also included in Table 3.1. The error in Table 3.1 is the average Euclidean distance between the estimated fiducial locations and the ground truth, which is obtained manually using CloudCompare [77], a 3D annotation tool.

Table 3.1: Quantitative evaluation of marker localization on 3D maps.

Scene	Localized/Total	Error (m)
Fig. 3.1(a)–Traj LO [10]	2/2	0.016
Fig. 3.1(a)–Livox Mapping [12]	2/2	0.013
Fig. 3.13-Traj LO [10]	3/3	0.015
Fig. 3.13-Livox Mapping [12]	2/3	0.020
Fig. 3.1(c)–Map 1	2/2	0.026
Fig. 3.1(c)–Map 2	4/4	0.021
Fig. 3.1(c)–Map 3	2/2	0.017
Fig. 3.14	4/4	0.013

The results in Fig. 3.1(a) (outdoor office park, two maps) and Fig. 3.13 (indoor small parking lot, two maps) demonstrate the generalizability of the proposed algorithm to maps constructed using various methods [10, 12].

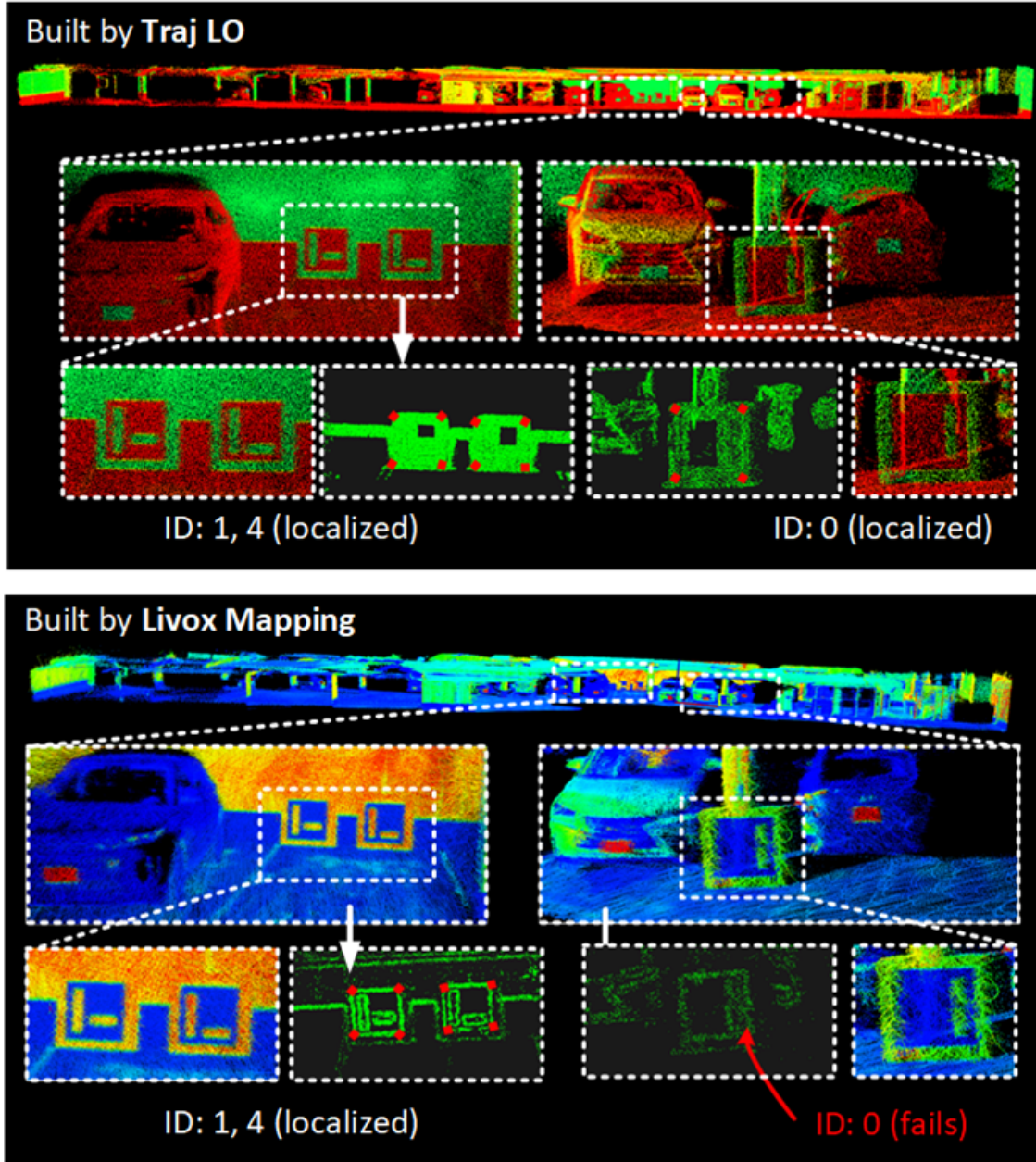


Figure 3.13: The LiDAR scans the indoor parking lot from right to left, moving rapidly when scanning marker ID 0.

Each scene uses the same rosbag as input for different SLAM methods, including Traj LO [10] and Livox Mapping [12], to build 3D maps. As shown, the proposed method

successfully localizes all markers in these maps except for one marker in the map built by Livox Mapping in Fig. 3.13. This is because Traj LO, as the state-of-the-art pure LiDAR-based SLAM method, is more robust to rapid motion [10] than Livox Mapping, and therefore produces maps with less motion blur. For this reason, all the remaining 3D maps in this section are built using Traj LO [10]. Thereafter, four additional scenes are presented, as shown in Fig. 3.1(c) (maps 1-3) and Fig. 3.14. As seen, all the fiducial markers are successfully localized.

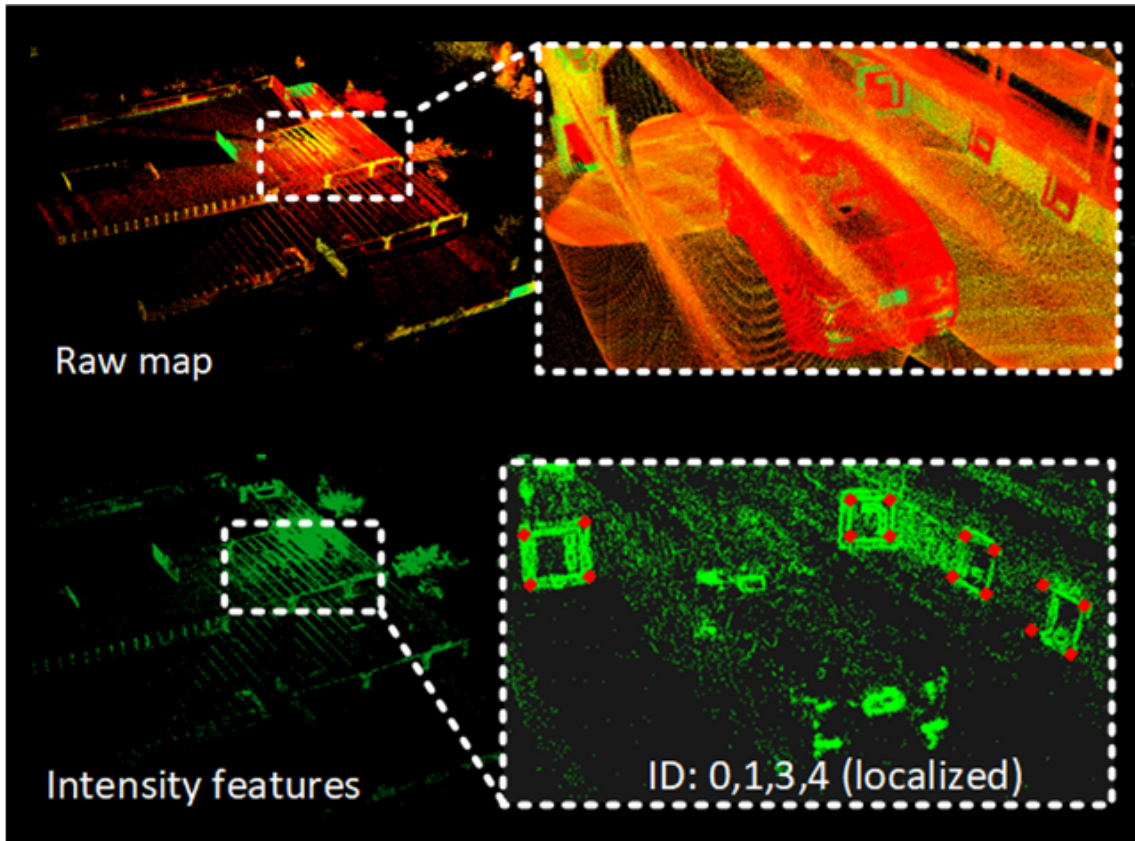


Figure 3.14: Traj LO [10] is utilized to create this 3D map.

3.5.2 Extending the Valid LFM Localization Range:

Addressing the Second Limitation

The second limitation of the vanilla IFM is that the projection of the marker is too small to be detected when the distance between the marker and LiDAR is far. This section reports experimental results that demonstrate the improvements achieved by the proposed method in this regard. Specifically, as depicted in Fig. 3.15, we fix the position of a $69.2\text{ cm} \times 69.2\text{ cm}$ ArUco marker and move the LiDAR to increase the relative distance. The intensity images obtained by the vanilla IFM and the proposed method at different distances are also presented in Fig. 3.15.

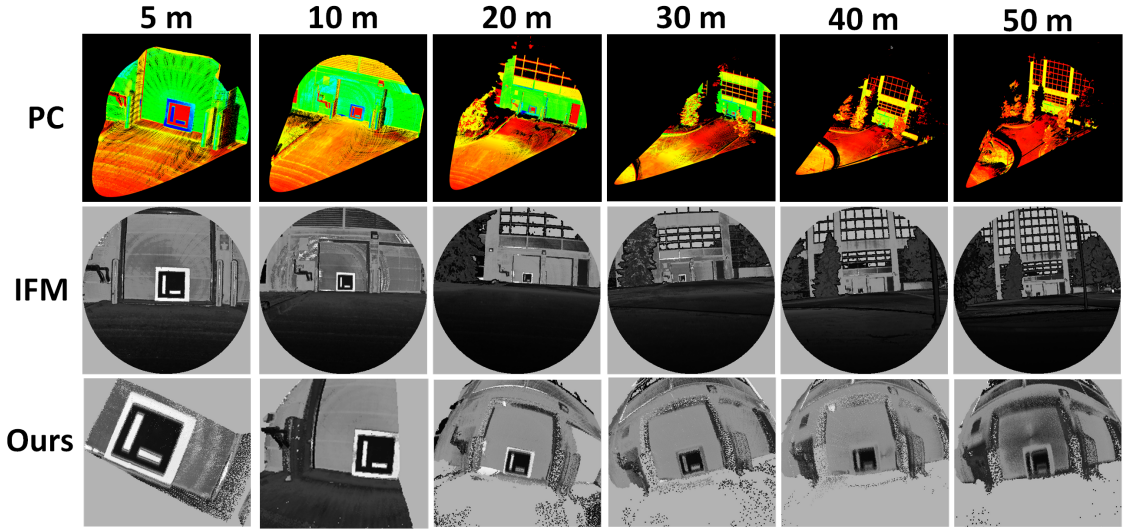


Figure 3.15: Comparison of the intensity images generated by the vanilla IFM and the proposed method at different distances.

The quantitative evaluation results are presented in Table 3.2. As seen, the vanilla IFM fails to localize the marker when the distance reaches 20 meters, whereas the proposed method remains valid even when the distance extends to 50 meters. The intensity images shown in Fig. 3.15 illustrate the reason: as the distance increases, the projection of

the marker becomes smaller and smaller using the vanilla IFM. In contrast, the proposed method extracts the point cluster belonging to the marker and transforms it onto the intermediate plane, thereby maintaining a high-quality projection where the marker remains detectable. However, it was observed that at a distance of 50 meters, although the marker is localized, the method returns a false ID number. Upon observing the intensity image at 50 meters, we believe this occurs because the captured points become sparser and noisier as the distance increases, leading to poorer quality of the point cloud for the marker.

Table 3.2: Quantitative evaluation of marker localization at various distances.

Method/Distance (m)	5	10	20	30	40	50
Vanilla IFM Error (m)	0.018	0.033	fails	fails	fails	fails
Ours Error (m)	0.009	0.016	0.019	0.023	0.031	0.037 (false ID)

3.5.3 Marker Localization Accuracy Improvement

In this section, the same experimental setup shown in Fig. 2.10 is used to demonstrate that the proposed method boosts pose estimation accuracy compared to the vanilla IFM. In particular, a $16.4 \text{ cm} \times 16.4 \text{ cm}$ AprilTag [1] is placed in front of the LiDAR, and the ground truth pose is provided by a motion capture system. In addition to the vanilla IFM, a comparison is also made with the widely-used AprilTag 3 [1]. When testing AprilTag 3, the adopted sensor is a camera. As shown in Table 3.3, both the accuracy of the vanilla IFM and AprilTag 3 degrades as the distance from the sensor to the marker increases, but the degradation in accuracy is less severe for IFM compared to AprilTag 3. The accuracy of the proposed approach is slightly better than the vanilla IFM at a distance of 2m. However, unlike the vanilla IFM, the proposed approach maintains decent accuracy as the distance increases. The reason is illustrated in Fig. 3.15: the proposed method

generates higher-quality intensity images thanks to cluster extraction and the adoption of the intermediate plane.

Table 3.3: Comparison of the proposed approach, the vanilla IFM, and AprilTag3 [1] with respect to pose estimation accuracy.

Distance	Method	X error (m)	Y error (m)	Z error (m)
2m	AprilTag3 [1]	0.016	0.034	0.016
	Vanilla IFM	0.003	0.006	0.017
	Ours	0.002	0.005	0.011
3m	AprilTag3 [1]	0.058	0.124	0.044
	Vanilla IFM	0.026	0.021	0.093
	Ours	0.006	0.009	0.015
4m	AprilTag3 [1]	0.072	0.407	0.233
	Vanilla IFM	0.024	0.024	0.107
	Ours	0.008	0.014	0.016
Distance	Method	Roll error (deg)	Pitch error (deg)	Yaw error (deg)
2m	AprilTag3 [1]	0.807	3.166	1.244
	Vanilla IFM	0.423	0.457	0.399
	Ours	0.315	0.305	0.391
3m	AprilTag3 [1]	1.369	7.963	2.904
	Vanilla IFM	0.930	1.182	0.859
	Ours	0.343	0.322	0.455
4m	AprilTag3 [1]	1.433	9.292	13.343
	Vanilla IFM	0.342	2.020	1.111
	Ours	0.302	0.389	0.478

3.5.4 Application and Discussion

The localized fiducials on the 3D maps are beneficial for downstream tasks, such as map merging. Specifically, as presented in Fig. 3.1(c), multiple large-scale 3D maps built by LiDAR-based SLAM can be merged using the method introduced in Section 4, utilizing the fiducials provided by the proposed method on 3D maps.

As illustrated by the experiments conducted in Sections 3.5.1 and 3.5.2, the inten-

tion behind improving the vanilla IFM is to handle large-scale outdoor scenes and cases where the distance between the LiDAR and the marker is significant. Nevertheless, the improvements come at the cost of increased computational time. Thus, unless faced with challenging large-scale scenarios, the vanilla IFM is capable of handling regular small-scale scenes and is preferred for its simplicity and efficiency.

4 Exploiting LFMs for Mapping and Localization

4.1 Overview

This chapter introduces a framework for mapping and localization using LFMs. Specifically, mapping is performed by registering multiview point clouds, while localization is achieved by estimating the relative poses between them. Fig. 4.1 provides an overview of the proposed framework. As shown, the LFMs are thin sheets of board or paper attached to surfaces, with no impact on the environment’s geometry, making them suitable for tasks such as training dataset collection for point cloud registration. Given an unordered set of low-overlap point clouds, the proposed method efficiently registers them in a global frame. Due to the use of a constant threshold for processing intensity images (See Fig. 2.7(c)), the vanilla IFM becomes unstable when the viewpoint varies significantly. To address this issue, an adaptive threshold marker detection method, robust to viewpoint changes, is proposed. The multiview point cloud registration problem is formulated as a MAP problem, and tackled using two-level graphs. The first-level graph, constructed as a weighted graph, efficiently processes the unordered point clouds and estimates the initial poses between them. The weights represent the pose estimation error of each marker observation, and the optimal initial poses are obtained by finding the shortest path from an anchor scan to

each non-anchor scan.



Figure 4.1: An overview of the proposed framework for mapping and localization using LFM.

Using the initial values, the second-level graph, a factor graph, resolves the MAP problem by globally optimizing the poses of point clouds, markers, and marker corner positions. We conduct both qualitative and quantitative experiments to demonstrate the superior-

ity of the proposed method over competitors [2–8]. Especially, the proposed method is robust to any unseen scenarios with extremely low overlap, making it a convenient, efficient, and low-cost tool for diverse applications that pose significant challenges to existing methods. As shown in Fig. 4.1, the downstream applications include 3D asset collection from sparse scans, training dataset collection for point cloud registration in unseen scenes, reconstruction of degraded scenes, navigation in GPS-denied environments, and merging of large-scale low overlap 3D maps.

4.2 Methodology

4.2.1 Adaptive Threshold LFM Detection

In the vanilla IFM method, binarization is applied to the raw intensity image due to the imaging noise (See the zoomed views of Fig. 4.2). As shown in Fig. 4.2, the effect is determined by the threshold λ .

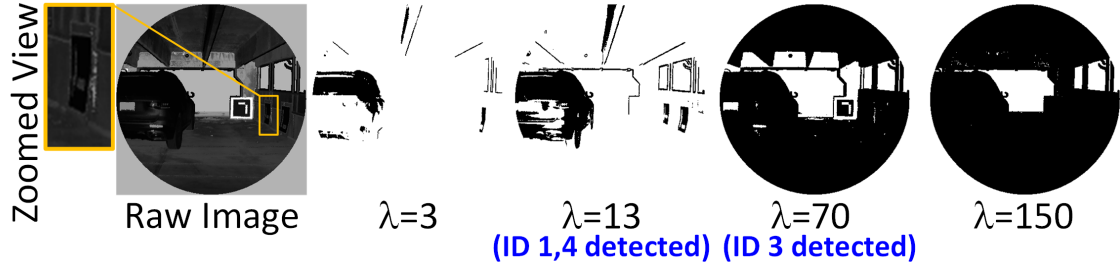


Figure 4.2: The raw intensity image binarized with different threshold values.

In Section 2, it is found that λ can be selected as a constant if the viewpoint does not change drastically. However, with significant changes in the scene, the value of λ needs adjustment. As an example, consider the case shown in Fig. 4.2, where three markers are placed. Markers identification (ID) 1 and 4 are detectable when $\lambda=13$, while marker ID

3 is detectable when $\lambda=70$. Thus, $\lambda=13$ and $\lambda=70$ are the optimal thresholds compared to other values for markers ID 1 and 4 and marker ID 3, respectively, denoted by λ^* . Therefore, unless an appropriate λ is carefully selected for each point cloud (viewpoint) and for each marker, the LFM detection will fail. Unlike the vanilla IFM, which determines λ^* based on experience or experimentation, an algorithm to automatically seek λ^* is developed in this section. Given that the focus is on marker detection rather than image denoising,

Algorithm 1: Search for the optimal threshold, λ^* .

Input: Raw intensity image, $\mathcal{I}$

Output: The optimal threshold, λ^* .

Initialize parameters: Search scope, S . step size, δ . queue for saving detected markers, $Q = []$. length of Q , $Q_l = 0$. temporary queue, $Q_{temp} = []$. length of Q_{temp} , $Q_{temp,l} = 0$. the optimal threshold, $\lambda^* = 0$. search step, $i = 0$.

Define the binarization operation as $\Psi(\mathcal{I}, \lambda)$.

Define the marker detector operation as $\Gamma(\mathcal{I})$.

while $i < S$ **do**

$\lambda = \delta \times i$

$\mathcal{I} = \Psi(\mathcal{I}, \lambda)$

$\Gamma(\mathcal{I}) \rightarrow Q_{temp}$

if $Q_{temp,l} \geq Q_l$ **then**

for *marker* **in** Q_{temp} **do**

if *marker* **not in** Q **then**

 append *marker* to Q

$\lambda^* = \lambda$

$i = i + 1$

Return the image binarized with λ^* and Q .

Algorithm 1 is designed to utilize the detection result as feedback for the automatic search of λ^* . In particular, the core of the algorithm is to maximize the length of a memory queue

for saving detected markers (denoted by Q) by gradually increasing λ . Namely, the aim is to detect as many markers as possible by finding the optimal threshold for each marker in the scene. After applying **Algorithm 1**, the 2D fiducials located on the image binarized with λ^* are projected into 3D fiducials using the subsequent steps of IFM. The proposed adaptive threshold LFM detection algorithm addresses the problem of a constant threshold being inapplicable to all point clouds and markers, especially when LiDAR moves in the wild and scenes undergo significant changes.

4.2.2 Problem Formulation

In this section, the problem formulation is introduced. Suppose that the marker size is a . In the marker coordinate system $\{M\}_j$, the 3D coordinates of the four corners of the j -th marker, ${}^j\mathbf{p}^{j,1}$, ${}^j\mathbf{p}^{j,2}$, ${}^j\mathbf{p}^{j,3}$, and ${}^j\mathbf{p}^{j,4}$, are $[-a/2, -a/2, 0]^T$, $[a/2, -a/2, 0]^T$, $[a/2, a/2, 0]^T$, and $[-a/2, a/2, 0]^T$, respectively. LFM detection returns the 3D coordinates of the corners expressed in $\{F\}_i$ (the local coordinate system of the i -th scan f_i). Thus, the 6-DOF transmission from $\{M\}_j$ to $\{F\}_i$, denoted as $\mathbf{T}_i^j$, can be found by solving the following least square problem:

$$\mathbf{T}_i^{j,*} = \arg \min_{\mathbf{T}_i^j} \sum_{s=1}^4 \left\| \mathbf{T}_i^j \cdot {}^j\mathbf{p}^{j,s} - {}_i\mathbf{p}^{j,s} \right\|^2. \quad (4.1)$$

The SVD method [65] is employed to compute $\mathbf{T}_i^{j,*}$. The set of measurements, including

- (i) the corner positions w.r.t. $\{M\}_j$,
- (ii) the corner positions w.r.t. $\{F\}_i$,
- (iii) the marker poses w.r.t. $\{F\}_i$,

are denoted as $\mathcal{Z}$. To register the multiview point clouds, it is necessary to find a globally consistent pose for each scan so that the point clouds can be transformed into a complete

point cloud in the world coordinate system $\{G\}$. To this end, the following variables are considered:

- (i) the poses of point clouds w.r.t. $\{G\}$,
- (ii) the poses of markers w.r.t. $\{G\}$,
- (iii) the marker corner positions w.r.t. $\{G\}$.

The set of variables is represented by Θ . Finally, a MAP inference problem is formulated: given the measurements, $\mathcal{Z}$, the goal is to find the optimal variable set Θ^* that maximizes the posterior probability $P(\Theta \mid \mathcal{Z})$:

$$\Theta^* = \arg \max_{\Theta} P(\Theta \mid \mathcal{Z}). \quad (4.2)$$

In the following, a framework consisting of two-level graphs is designed to solve this MAP problem. Inspired by the coarse-to-fine pipeline of SGHR [3], a first-level graph is developed to efficiently and exhaustively determine the relative poses among point clouds, while a second-level graph is used to globally optimize the variables.

4.2.3 First-level Graph

As aforementioned, the variables in Θ to be optimized cannot be directly obtained through the measurements in $\mathcal{Z}$. Moreover, deriving the initial values of the variables requires both efficiency and accuracy, given that the input is an unordered set of low overlap point clouds. The first-level graph is designed to address this challenge. First, the computation of the relative pose between two point clouds is considered. Suppose the j -th marker appears in the scenes of two scans f_i and f_m , $\mathbf{T}_i^j$ and $\mathbf{T}_m^j$ can be calculated using Eq. (4.1). Consequently, the relative pose between f_i and f_m , denoted as $\mathbf{T}_{i,m}$, is available from

$$\mathbf{T}_{i,m} = (\mathbf{T}_i^j)^{-1} \mathbf{T}_m^j, \quad (4.3)$$

where $(\mathbf{T}_i^j)^{-1}$ indicates the inverse of $\mathbf{T}_i^j$. $(\mathbf{T}_i^j)^{-1}$ indicates the pose that transforms 3D points from $\{F\}_i$ to $\{M\}_j$. Although the method introduced in Eq. (4.3) is straightforward, it cannot be directly applied because the input in this work is an unordered set of point clouds that do not follow a temporal or spatial sequence. Specifically, for scans (point clouds) with no shared marker observations, their relative pose has to be calculated through pose propagation among other scans.

However, multiple alternative paths could exist. Even for scans that share marker observations, there could be more than one overlapped marker.

Hence, to accurately and efficiently estimate relative poses among scans, it is necessary to design an algorithm to determine which scans and markers to apply Eq. (4.3). Specifically, the objective is to infer the relative poses with the highest possible accuracy using only the necessary low-dimensional information through a simple process. Thus, as shown in Fig. 4.3, the first-level graph is constructed in the form of a weighted graph.

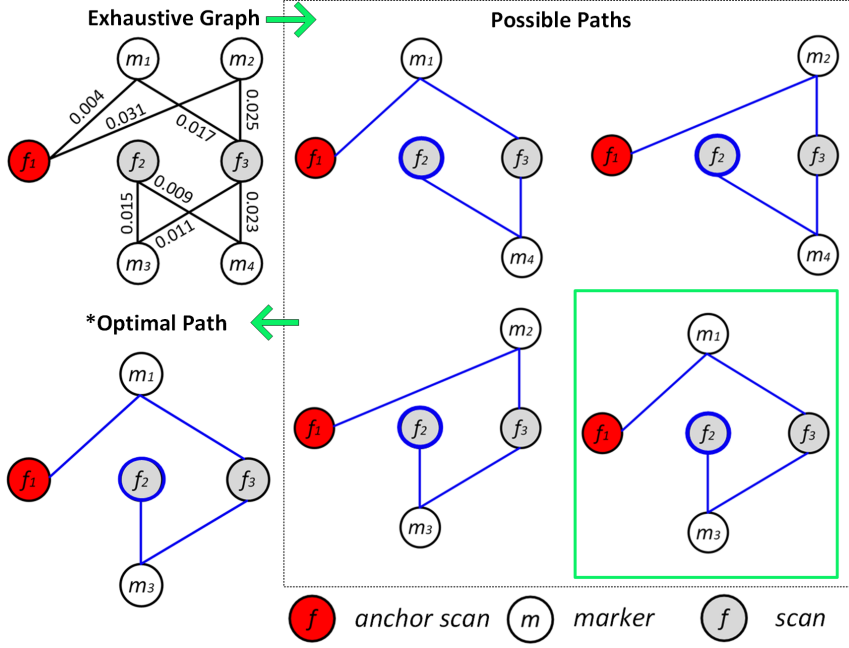


Figure 4.3: An illustration of the first-level graph. After applying the proposed adaptive marker detection to all scans, an exhaustive weighted graph is constructed, with the scans and markers as nodes and the point-to-point errors as edge weights. The aim is to derive the relative pose between each non-anchor scan and the anchor scan with optimal accuracy. However, for a given non-anchor scan, such as f_2 in this simple case, there may be multiple possible paths in the exhaustive graph leading to the anchor scan (f_1). Thus, Dijkstra’s algorithm [11] is employed to find the optimal path with the minimum accumulation of point-to-point errors (weights).

In particular, when processing the point clouds with the proposed adaptive threshold method one by one, if a marker is detected in a point cloud, the corresponding scan node and marker node are added to the graph along with a weighted edge. The edge weight is the pose estimation point-to-point error e_{pp} :

$$e_{pp} = \sum_{s=1}^4 \left\| \mathbf{T}_i^{j,*} \cdot {}^j \mathbf{p}^{j,s} - {}_i \mathbf{p}^{j,s} \right\|^2. \quad (4.4)$$

Eq. (4.4) indicates the substitution of the result given by SVD back into the right side of Eq. (4.1). $e_{pp} \in \mathbb{R}^+$ is employed as the metric to evaluate the quality of pose estimation of the marker in the corresponding point cloud. The first scan is defined as the anchor scan. The local coordinate system of the anchor scan is set as the world coordinate system. Namely, $\{F\}_1 = \{G\}$. Only the relative poses between the anchor scan and each non-anchor scan are considered. Although there could be multiple paths from the given scan to the anchor scan (see Fig. 3.8), the information on pose estimation quality has been saved by constructing the first-level graph as a weighted graph. Therefore, Dijkstra’s algorithm [11] is employed to obtain the shortest path. Then, the relative pose is computed along the shortest path iteratively using Eq. (4.3). Given that the relative pose computation is achieved with the lowest accumulation of e_{pp} , the estimation result achieves the highest possible accuracy. Moreover, the search for the optimal path to propagate poses is based merely on the one-dimensional e_{pp} , without incorporating any 6-DOF poses or 3D locations.

In this way, the point cloud poses w.r.t. $\{G\}$ are obtained. Since the marker detection provides the marker poses and the 3D positions of marker corners w.r.t. the local coordinate system of corresponding scan, the initial values of the marker poses w.r.t. $\{G\}$ and the corner positions w.r.t. $\{G\}$ can be derived through the point cloud poses w.r.t. $\{G\}$.

4.2.4 Second-level Graph

To address the MAP problem introduced in Eq. (4.2), we construct the second-level graph as a factor graph to globally optimize the variables using the initial values obtained from the first-level graph. Specifically, we create the second-level graph in three stages.

Stage One. Suppose that the j -th marker is detected in the i -th scan, the following six types of nodes are added to the second graph. The variable nodes include:

- (1) Node $\mathbf{T}^j$, which refers to the 6-DOF pose of the j -th marker *w.r.t.* $\{G\}$;
- (2) Node $\mathbf{T}_i$, which refers to the 6-DOF pose of the i -th scan *w.r.t.* $\{G\}$;
- (3) Nodes $\{\mathbf{p}^{j,1}, \mathbf{p}^{j,2}, \mathbf{p}^{j,3}, \mathbf{p}^{j,4}\}$, which refer to 3D coordinates of the corners of the j -th marker *w.r.t.* $\{G\}$.

The factor nodes include:

- (4) Node $\mathbf{T}_i^j$, which refers to the measurement of the 6-DOF pose of the j -th marker *w.r.t.* $\{F\}_i$;
- (5) Nodes $\{^j\mathbf{p}^{j,1}, ^j\mathbf{p}^{j,2}, ^j\mathbf{p}^{j,3}, ^j\mathbf{p}^{j,4}\}$, which refer to the measurement of the 3D coordinates of the corners of the j -th marker *w.r.t.* $\{M\}_j$;
- (6) Nodes $\{_i\mathbf{p}^{j,1}, _i\mathbf{p}^{j,2}, _i\mathbf{p}^{j,3}, _i\mathbf{p}^{j,4}\}$, which refer to 3D coordinates of the corners of the j -th marker *w.r.t.* $\{F\}_i$.

By adding variables representing the 3D coordinates of the corners, we can also optimize the fiducial localization results. The first stage in Fig. 4.4 shows the added nodes and edges when a marker is detected in a scan.

Stage Two. Thereafter, we traverse all the marker detection results and conduct the operation from Stage One for each detected marker. After processing all the detected markers, the second-level graph becomes the one shown in Stage Two of Fig. 4.4. Since the operation in this stage essentially consists of repetitions of Stage One, the node types are no different from those in the previous stage.

Stage Three. Considering that the local coordinate system of the first scan, $\{F\}_1$, is set as the global coordinate system, $\{G\}$, we add a prior factor that connects to the first (anchor) scan node, $\mathbf{T}^1$. Finally, we add factor nodes representing the relative poses between the

anchor scan and each non-anchor scan. Up to this point, the factor graph is completed, as shown in Stage Three of Fig. 4.4.

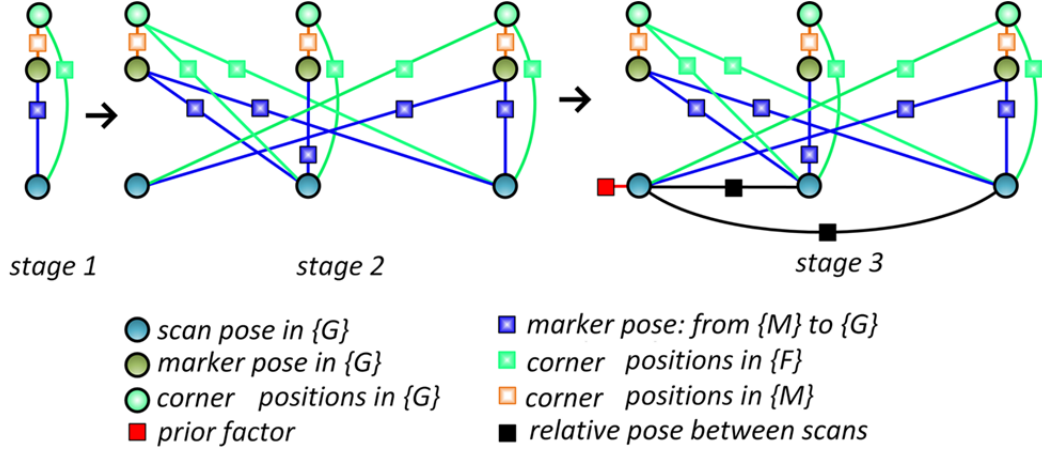


Figure 4.4: The procedures for formulating the second-level graph. The variable nodes are represented by circles and the factor nodes are represented by squares. In Stage One, when a marker is detected in a scan, six types of nodes are added to the graph, including (1) scan pose in $\{G\}$, (2) marker pose in $\{G\}$, (3) corner positions in $\{G\}$, (4) marker pose from $\{M\}$ to $\{G\}$, (5) corner positions in $\{F\}$, and (6) corner positions in $\{M\}$, along with their corresponding edges. In Stage Two, all the marker detection results are traversed, and the operation from Stage One is repeated for each detected marker. In Stage Three, a prior factor connecting the anchor scan is added, along with factors representing the relative poses between the anchor scan and non-anchor scans.

Following [78, 79], since the factor graph is determined, the posterior probability $P(\Theta \mid \mathcal{Z})$ is factorized as:

$$P(\Theta \mid \mathcal{Z}) = \prod_k P^{(k)}(\Theta), \quad (4.5)$$

where $P^{(k)}$ are the factors in the second graph. We follow the standard pipeline [78] to

model them as Gaussians:

$$P^{(k)}(\Theta) \propto \exp \left(-\frac{1}{2} \|h_k(\Theta) \ominus z_k\|_{\Sigma_k}^2 \right), \quad (4.6)$$

where $h_k(\Theta)$ is the measurement function and z_k is a measurement. $\|e\|_{\Sigma}^2 \triangleq e^T \Sigma^{-1} e$ denotes the squared Mahalanobis distance with Σ being the covariance matrix. Following [66], if $z_k \in \mathbb{R}^{3 \times 1}$ is a 3D position, $\ominus$ refers to straight subtraction for elements. If $z_k \in SE(3)$ is a 6-DOF pose, $\ominus$ generates 6-dimensional Lie algebra (refer to Section 2.2.2) coordinates:

$$\mathbf{T}_A \ominus \mathbf{T}_B = [[\log(\text{Rot}(\mathbf{T}_B^{-1} \mathbf{T}_A))]_{\vee}^T, \text{Trans}(\mathbf{T}_B^{-1} \mathbf{T}_A)^T]^T, \quad (4.7)$$

where for a $\mathbf{T} \in SE(3)$, $\text{Rot}(\mathbf{T})$ denotes the rotation matrix $\mathbf{R} \in SO(3)$ and $\text{Trans}(\mathbf{T})$ denotes translation vector $\mathbf{t} \in \mathbb{R}^{3 \times 1}$. $\log(\cdot)$ represents the matrix logarithm. $\vee$ is the *vec* map operator. The detailed introduction of $\vee$ is provided in Section 2.2.2. With the Gaussian modeling, the objective function in Eq. (4.5) is transformed into a least square problem by applying the negative logarithm:

$$\arg \min_{\Theta} (-\log \prod_k P^{(k)}(\Theta)) = \arg \min_{\Theta} \frac{1}{2} \sum_k \|h_k(\Theta) \ominus z_k\|_{\Sigma_k}^2. \quad (4.8)$$

We utilize the Levenberg-Marquardt algorithm [79] to solve this problem. The acquisition of initial values is introduced in Section 4.2.3. The noise covariance matrices are determined by the quantitative experiments conducted in [39].

4.3 Livox-3DMatch Dataset

A common approach [3, 45] to evaluating a learning-based point cloud registration model is to train it on 3DMatch [35] and test it on various benchmarks, including 3DMatch [35], ETH [36], and ScanNet [37]. However, the 3DMatch benchmark is mainly constructed from RGB-D camera captures of indoor scenes [35]. In this dissertation, a new dataset,

named Livox-3DMatch, is collected to enrich the training data for learning-based methods. The enrichment contains two key components.

First, the sensor we adopt is a Livox MID-40 LiDAR, which has different sampling patterns compared to the RGB-D camera (See Fig. 4.5(a) and (b)). Thus, it adds point clouds with new features to the training data.

Second, scenes that are absent or rare in 3DMatch are selectively sampled, thereby enriching the scenes in the training data. For example, the valid depth range of an RGB-D camera is usually less than ten meters, making it unsuitable for sampling outdoor scenes. In contrast, some outdoor scenes (See Fig. 4.5(c)) are collected for Livox-3DMatch, considering that a LiDAR can sample objects a few hundred meters away. Moreover, some challenging cases (See Fig. 4.5(d)) are gathered where the overlapping regions are mainly planes, which are rare in 3DMatch [35]. A more detailed introduction to the scenes in Livox-3DMatch is given in Sections 4.4.3 and 4.4.4. In Section 4.4.5, we demonstrate that the proposed Livox-3DMatch can boost the performance of the state-of-the-art learning-based methods [2, 3] on various benchmarks [35–37].

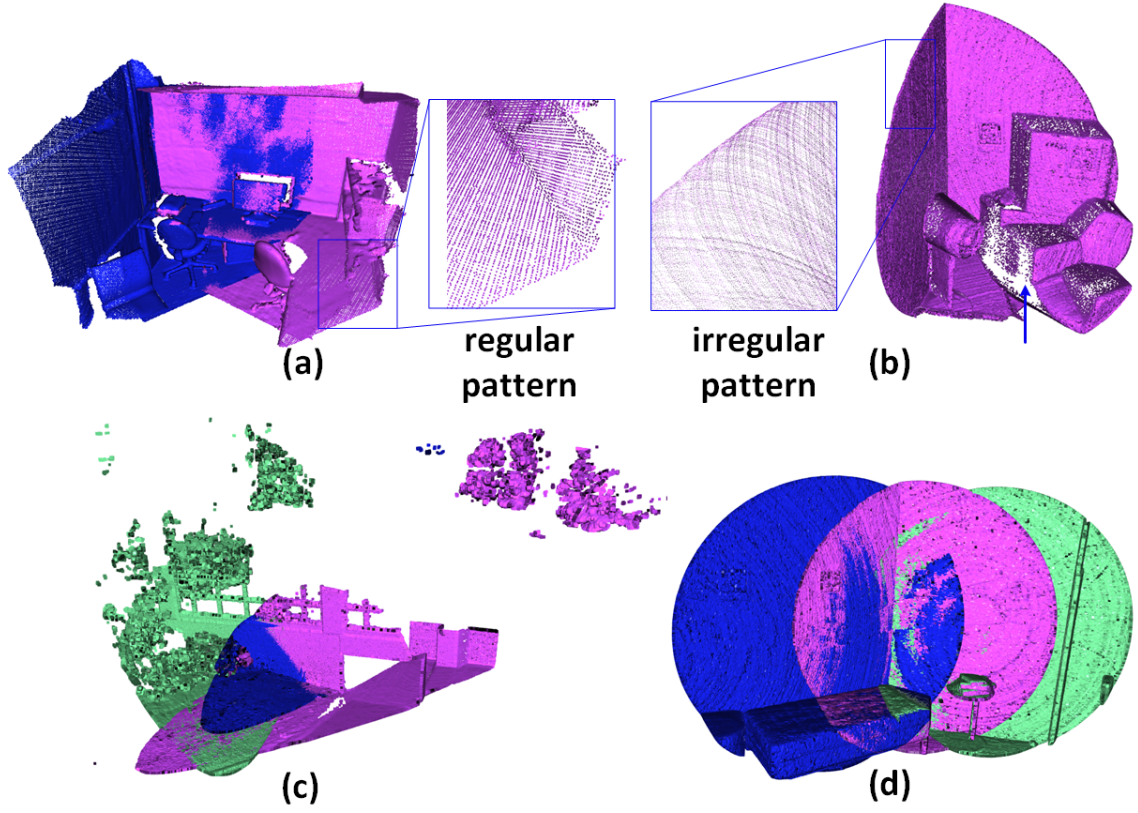


Figure 4.5: Comparison of 3DMatch and Livox-3DMatch. (a): A random sample from 3DMatch. The point cloud sampled by an RGB-D camera has a regular pattern and less noise. (b): A random sample from Livox-3DMatch. The Livox LiDAR point cloud has an irregular pattern and more noise. (c): An example of an outdoor scene in Livox-3DMatch. (d): A selectively sampled challenging case in which the overlapping regions are mainly planes.

4.4 Experimental Validation

4.4.1 Experimental Setup

In this section, both the solid-state LiDAR (Livox Mid-40) and mechanical LiDAR (RS-Ruby-128) are employed to validate the proposed framework. In particular, the point clouds are sampled from various indoor and outdoor scenes, including offices, a meeting room, lounges, a kitchen, office buildings, and thickets. In indoor scenes, letter-sized AprilTags are utilized as LFMs. In outdoor scenes, poster-sized ArUcos are employed to provide LiDAR fiducials. For experiments concerning registration accuracy, the ground truth poses are obtained by manually registering through CloudCompare [77], a popular 3D annotation tool. The ground truth for evaluating the quality of 3D asset collection is a high-fidelity 3D model acquired from a 3D assets website. For experiments involving localization accuracy, the ground truth trajectory is provided by the MoCap system for the indoor scene and by Real-Time Kinematic (RTK) for the outdoor scene.

4.4.2 Evaluation of the Adaptive Marker Detection Method

Data. The Livox MID-40 LiDAR and $69.2 \text{ cm} \times 69.2 \text{ cm}$ ArUco marker(s) are placed in three different scenes, as shown in Fig. 4.6. The three scenes are: between groups of buildings, in open outdoor areas, and in large indoor parking lots. In each scene, we collect point clouds and test the proposed adaptive threshold method at different relative positions.

Results and Analysis. Table 4.1 presents the results, where x and y indicate the relative position of the LiDAR in the marker (ID 4) coordinate system. Specifically, in scene Fig. 4.6(c), there are two markers (ID 4 and ID 1), so the column for λ^* contains two values. As seen in the table, λ^* varies significantly as the relative position changes, demonstrating

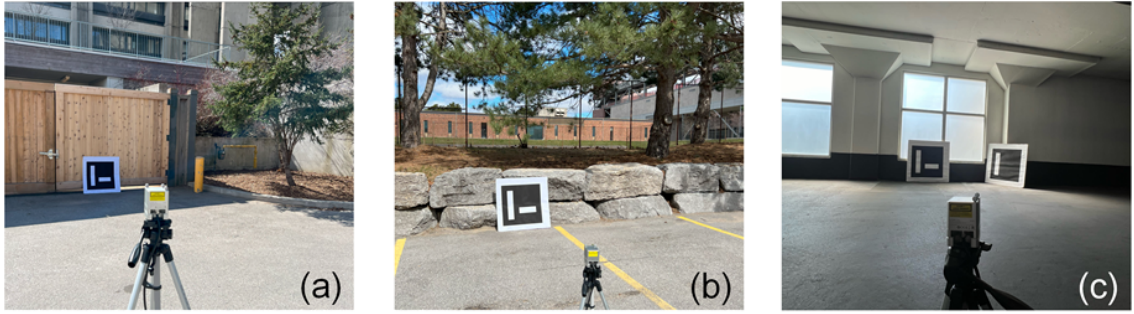


Figure 4.6: Setup for testing the adaptive threshold marker detection algorithm: (a) between groups of buildings, (b) in an open outdoor area, and (c) in a large indoor parking lot.

the necessity of the proposed adaptive marker detection method. In addition, the results shown in Fig. 4.6(c) also illustrate the necessity of the memory queue design, considering that a single λ^* might not be applicable to all markers in the same scene. Please note that the intention of reporting the values of the optimal threshold in Table 4.1 is to demonstrate that a constant threshold, as adopted in [39], is not applicable to this task. However, the major concern of the adaptive threshold marker detection algorithm is to detect all markers in a scene. Namely, the marker detection results saved in the memory queue are the most important output rather than the thresholds.

Table 4.1: Demonstration of the necessity of the proposed adaptive marker detection algorithm

Scene	x (m)	y (m)	λ^*	Scene	x (m)	y (m)	λ^*	Scene	x (m)	y (m)	λ^*
Fig. 4.6(a) Between groups of buildings	0	4	36	Fig. 4.6(b) In an open outdoor area	0	4	7	Fig. 4.6(c) In a large indoor parking lot	4	4	20/6
	4	4	9		4	4	8		4	4	8/8
	-4	4	6		-4	4	6		-4	4	6/8
	0	5	20		0	5	9		0	5	15/10
	5	5	8		5	5	8		5	5	8/12
	-5	5	7		-5	5	9		-5	5	7/13
	0	6	19		0	6	9		0	6	8/13
	6	6	10		6	6	6		6	6	11/11
	-6	6	11		-6	6	8		-6	6	11/10
	0	7	17		0	7	9		0	7	19/20
	7	7	8		7	7	8		7	7	15/14
	-7	7	9		-7	7	9		-7	7	28/16
	0	8	15		0	8	13		0	8	10/16
	8	8	9		8	8	11		8	8	11/17
	-8	8	10		-8	8	15		-8	8	12/15
	0	9	17		0	9	10		0	9	13/11
	9	9	8		9	9	12		9	9	20/26
	-9	9	9		-9	9	12		-9	9	16/12
	0	10	14		0	10	13		0	10	24/28
	10	10	11		10	10	10		10	10	10/19
	-10	10	12		-10	10	15		-10	10	15/19

4.4.3 Evaluation of Point Cloud Registration Accuracy

Data. Given that the existing point cloud registration benchmarks [35–37] lack fiducial markers in the scenes, a new dataset is constructed with the Livox MID-40, as shown in Fig. 4.7. As illustrated in the caption of Fig. 4.7, the newly collected dataset covers various indoor and outdoor scenes. Indoors, multiple $16.4 \text{ cm} \times 16.4 \text{ cm}$ AprilTags [1] are positioned in the environment. Outdoors, multiple $69.2 \text{ cm} \times 69.2 \text{ cm}$ ArUcos [18] are placed in the environment. Note that since the LFMs in this work are thin sheets of objects attached to other surfaces, they are almost invisible in the point clouds. This is infeasible if LiDARTags [9] or calibration boards [41–43] are adopted to provide fiducials.

These scenes are challenging due to low overlap, and the overlap rate [45] of each scene is also presented in Table 4.2. The ground truth poses between scans are obtained manually using CloudCompare [77]. Note that manually aligning point clouds is a labor-intensive and time-consuming process, highlighting the benefits of developing an automatic tool like the proposed framework. The inference time of the methods is compared using an AMD Ryzen 7 5800X CPU.

Competitors and Metrics. Competitors include the latest state-of-the-art (SOTA) multiview point cloud registration methods, MDGD [2] (RA-L’24), SGHR [3] (CVPR’23), and the SOTA pairwise methods, SE3ET [4] (RA-L’24), GeoTrans [5] (TPAMI’23), and Teaser++ [6] (T-RO’20). All the competitors are learning-based methods, except for Teaser++, which is a geometry-based method. For all pairwise methods, we manually select pairs of point clouds that have overlap as the inputs. The root-mean-square errors (RMSEs) [28] are employed as the metric:

$$\begin{aligned} \text{RMSE}_T &= \sqrt{\sum_{n=0}^{N_s} e_{T,n}^2 / (N_s + 1)}, \\ \text{RMSE}_R &= \sqrt{\sum_{n=0}^{N_s} e_{R,n}^2 / (N_s + 1)}, \end{aligned} \tag{4.9}$$

where $e_{T,n}^2$ and $e_{R,n}^2$ represent the squared Euclidean distances between the ground truth and the estimates of translation and rotation, respectively. N_s denotes the number of samples. We also compare the inference time of the methods with an AMD Ryzen 7 5800X CPU.

Results and Analysis. The qualitative and quantitative results are presented in Fig. 4.7 and Table 4.2, respectively. The pairwise methods [4–6], although provided manually selected pairs of point clouds with overlap, struggle in all scenes. This is because they are tailored for extracting geometric features either in a learning-based manner [4, 5] or through conventional geometry [6]. Thus, when the overlap ratio between point clouds is

low and the overlapped regions lack sufficient geometric features, these methods struggle to find features and utilize them to align point clouds. In contrast, the multiview point cloud registration methods [2,3] show some successful cases. In particular, SGHR [3] successfully registers the point clouds in scenes 1 to 3 and one point cloud pair in scene 4. MDGD [2], built upon the framework of SGHR [3] but developing a new matching distance-based overlap estimation module, aligns scenes 1 and 2 with decent accuracy and also successfully registers one pair of point clouds in scenes 4, 5, 7, and 10. The reason is that, in addition to learning pairwise registration, the multiview point cloud registration methods [2,3] also learn to further optimize or refine the pose graph constructed using pairwise registration results. Despite this, they fail in other scenes. Reviewing their success cases, it is found that these indoor scenes are similar to those in their training dataset [35]. While failure cases, such as scenes 4 to 10, are rare or lacking in the training dataset. This indicates that their generalization to unseen scenarios is limited due to the out-of-distribution problem. The proposed method achieves the best performance. Specifically, our method does not require the overlapping regions to have explicit geometric features, thanks to the thin-sheet format of our LiDAR fiducial markers. Moreover, by virtue of the proposed adaptive marker detection algorithm, our method can robustly align any novel indoor or outdoor scenes with thin-sheet markers. The proposed method also yields the best efficiency as it focuses on registering point clouds through LiDAR fiducial markers rather than analyzing the entire point clouds. The process of our method takes several dozen seconds, while other methods need several minutes on the AMD Ryzen 7 5800X CPU.

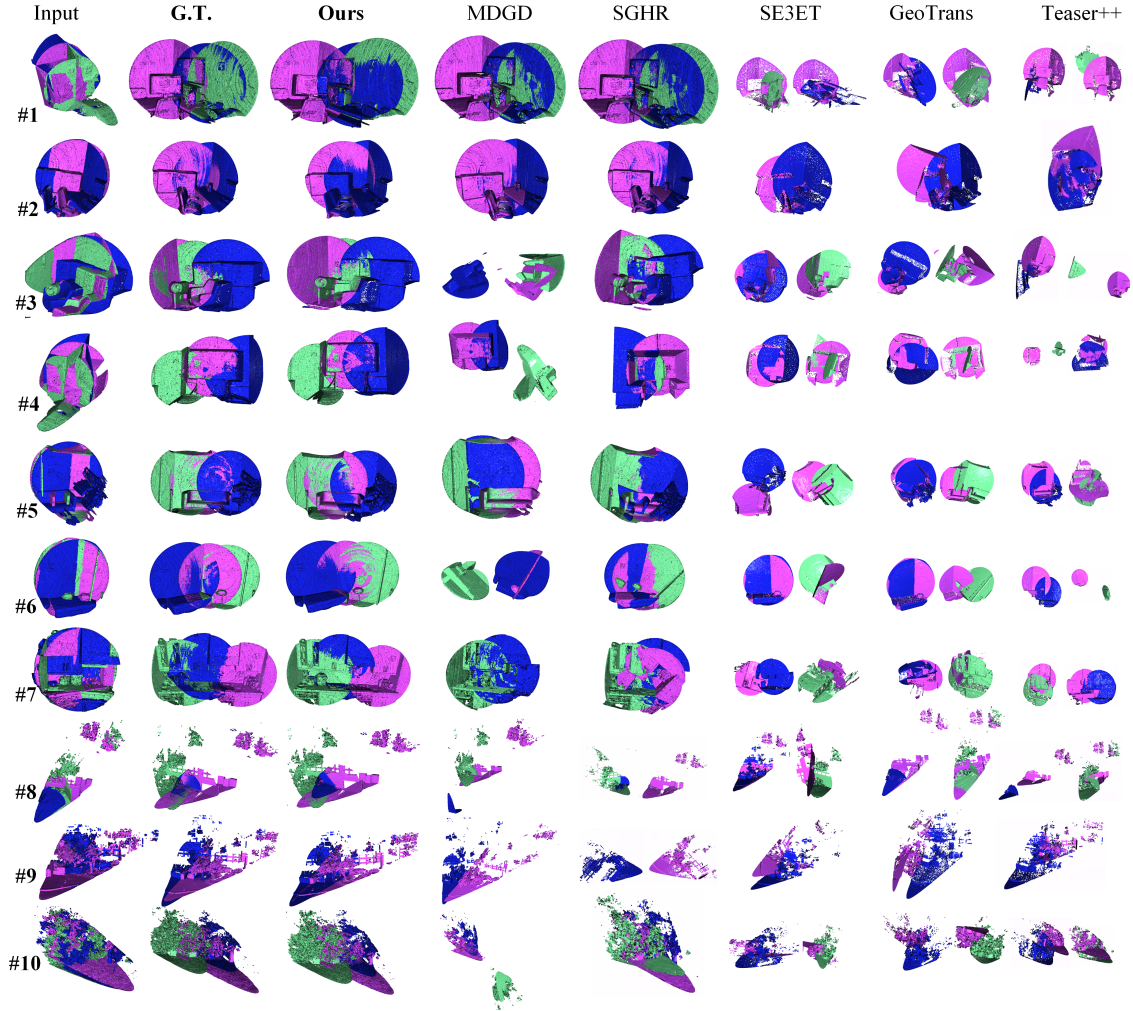


Figure 4.7: A comparison with SOTA methods, including MDGD [2], SGHR [3], SE3ET [4], GeoTrans [5], and Teaser++ [6], on ten scenes. The scenes include the office (1-3), the meeting room (4), the lounge (5,6), the kitchen (7), the office building (8,9), and the thicket (10). Each scene consists of three scans, except scenes 2 and 9, which are composed of two scans.

Table 4.2: Quantitative comparison with MDGD [2], SGHR [3], SE3ET [4], GeoTrans [5], and Teaser++ [6] on our dataset.

Scene #		1 (Office)	2 (Office)	3 (Office)	4 (Meeting Room)	5 (Lounge)
Avg/Min Overlap Rate (%)		27.17/14.29	46.88/46.88	44.66/22.26	25.80/1.40	43.82/15.48
Teaser++ [6] (T-RO'20)	RMSE _R (rad)	1.504	1.795	2.394	2.277	2.493
	RMSE _T (m)	1.724	1.890	2.075	2.252	1.911
	Time (s)	313.1	298.3	336.4	322.3	388.9
GeoTrans [5] (TPAMI'23)	RMSE _R (rad)	1.331	1.451	1.935	1.632	1.532
	RMSE _T (m)	1.502	1.671	1.912	1.552	1.325
	Time (s)	91.8	87.3	95.1	93.9	105.3
SE3ET [4] (RA-L'24)	RMSE _R (rad)	0.925	0.834	1.776	1.358	1.381
	RMSE _T (m)	0.996	1.113	1.736	1.273	1.207
	Time (s)	116.8	94.6	99.5	95.7	115.7
SGHR [3] (CVPR'23)	RMSE _R (rad)	0.152	0.060	1.198	1.825	1.311
	RMSE _T (m)	0.020	0.010	0.094	0.156	0.231
	Time (s)	846.1	785.0	886.5	825.3	851.7
MDGD [2] (RA-L'24)	RMSE _R (rad)	0.032	0.062	1.457	0.933	0.679
	RMSE _T (m)	0.015	0.009	0.352	1.035	0.113
	Time (s)	626.1	573.3	645.5	619.2	612.0
Ours	RMSE _R (rad)	0.036	0.068	0.089	0.065	0.088
	RMSE _T (m)	0.017	0.011	0.028	0.031	0.032
	Time (s)	31.1	21.2	35.7	32.6	36.2
Scene #		6 (Lounge)	7 (Kitchen)	8 (Building)	9 (Building)	10 (Thicket)
Avg/Min Overlap Rate (%)		50.66/27.24	31.31/4.19	19.06/12.20	44.65/44.65	12.16/5.42
Teaser++ [6] (T-RO'20)	RMSE _R (rad)	1.946	1.632	1.818	1.779	1.876
	RMSE _T (m)	1.733	2.089	1.861	1.848	1.891
	Time (s)	419.8	441.5	397.1	327.2	401.2
GeoTrans [5] (TPAMI'23)	RMSE _R (rad)	1.471	1.198	1.355	1.377	1.526
	RMSE _T (m)	1.235	1.132	1.730	1.554	1.531
	Time (s)	122.3	127.3	110.5	91.1	121.4
SE3ET [4] (RA-L'24)	RMSE _R (rad)	1.352	1.077	1.156	1.232	1.376
	RMSE _T (m)	1.130	1.095	1.469	1.413	1.392
	Time (s)	136.0	139.3	123.5	97.3	132.0
SGHR [3] (CVPR'23)	RMSE _R (rad)	1.696	3.42	2.792	3.034	2.810
	RMSE _T (m)	0.274	0.949	2.171	0.686	1.787
	Time (s)	979.3	983.6	909.9	794.6	950.4
MDGD [2] (RA-L'24)	RMSE _R (rad)	1.715	0.856	0.722	0.651	0.779
	RMSE _T (m)	0.422	0.337	1.553	0.686	0.939
	Time (s)	726.4	730.5	689.6	590.6	717.4
Ours	RMSE _R (rad)	0.078	0.067	0.069	0.087	0.101
	RMSE _T (m)	0.043	0.019	0.082	0.069	0.077
	Time (s)	43.5	53.8	39.9	24.9	42.2

4.4.4 Application 1: 3D Asset Collection from Sparse Scans

Collecting the complete shape of a novel object [38] from sparse observations is advantageous, as sparse scans require less labor and offer better efficiency. However, it is also a challenging task due to the low overlap between scans. In this test, we evaluate the instance reconstruction quality of the proposed method.

Data. The experimental setup is depicted in Fig. 4.8. Four $69.2\text{ cm} \times 69.2\text{ cm}$ ArUco [18] markers are placed in the environment. As depicted by the yellow trajectory, the Livox MID-40 LiDAR follows a looping path to scan the vehicle, pausing at five viewpoints for a few seconds to collect relatively dense point clouds due to its sparse scanning pattern [13]. The rostopic of the point cloud published by the LiDAR sensor is recorded as a rosbag throughout the sampling process. Then, the same rosbag is provided to all competitors. However, point cloud registration methods [2–6], including ours, utilize only a portion of the rosbag, *i.e.*, five point clouds with significant viewpoint changes, as shown in Fig. 4.8. This is also a challenging low-overlap case, with the average and minimum overlap rates being 13.39% and 1.02%, respectively. To evaluate the instance reconstruction quality, *Supervisely* [80], a popular 3D annotation tool, is used to extract the vehicle’s point cloud from the reconstruction result. Since the manufacture and model (Mercedes-Benz GLB 250) of the vehicle to be reconstructed are known, a high-fidelity 3D model acquired from a 3D assets website acts as the ground truth shape.

Competitors and Metrics. Competitors include the state-of-the-art point cloud registration methods (MDGD [2], SGHR [3], SE3ET [4], GeoTrans [5], and Teaser++ [6]) and LOAM methods (Traj LO [10], Livox Mapping [12], and LOAM Livox [13]). Pairs of point clouds with overlapping regions are manually provided to the pairwise methods [4–6]. MDGD [2], SGHR [3], and our method directly process the set of point clouds. The LOAM methods take the entire rosbag as input. Following [34], the Chamfer Distance (CD) and

Recall are computed between the ground truth shape and the reconstruction result. In particular, CD is defined as

$$CD(\mathbf{X}, \mathbf{Y}) = \sum_{x \in \mathbf{X}} \min_{y \in \mathbf{Y}} \|x - y\|_2^2 + \sum_{y \in \mathbf{Y}} \min_{x \in \mathbf{X}} \|x - y\|_2^2. \quad (4.10)$$

where $\mathbf{X}$ and $\mathbf{Y}$ are two point sets with x and y being the 3D points belonging to them, respectively. Recall is defined as follows:

$$\text{Recall}(\mathbf{X}, \mathbf{Y}) = \frac{1}{|\mathbf{X}|} \sum_{x \in \mathbf{X}} \left[\min_{y \in \mathbf{Y}} \|x - y\|_2^2 \leq thr \right], \quad (4.11)$$

where thr is a predefined threshold [34].

Results and Analysis. The qualitative results for point cloud registration methods and LOAM methods are shown in Fig. 4.8 and Fig. 4.9, respectively. The quantitative results are presented in Table 4.3. As shown in Fig. 4.8, MDGD [2] and SGHR [3] successfully align the third and fifth scans. In particular, the third and fifth scans have the highest overlap ratio of 59.33% among all pairs as they are captured from similar perspectives. However, when dealing with point clouds that have lower overlap ratios, the competitors [2–6] struggle to register them. This comparison illustrates that, although these existing methods can handle some unseen scenarios with high overlap, they are not generalizable to novel low overlap cases. Moreover, the failure of these existing methods in the instance reconstruction task implies that they are not suitable for efficient 3D asset collection. By contrast, the proposed method successfully registers the unordered multiview point clouds.

In terms of instance reconstruction quality, as depicted in Fig. 4.9, our reconstructed shape preserves intricate details well compared to the ground truth shape. The Traj LO [10] result shows almost no drift, though the surface is noisy. The Livox Mapping [12] result exhibits noticeable drift and surface noise, while LOAM Livox [13] demonstrates severe drift. Moreover, these LOAM methods require the entire rosbag instead of just five sparse

scans. As shown in Table 4.3, the proposed method achieves the highest reconstruction quality. Specifically, the CD and Recall of the reconstructed result are **0.003** and **96.22%**, respectively. The decent performance in terms of reconstruction quality metrics [34,38] indicates that our method can serve as a convenient, efficient, and low-cost tool for collecting high-fidelity 3D assets using the LiDAR sensor. In particular, we consider the proposed method efficient, as it only requires four or five scans from dramatically changed viewpoints to reconstruct the complete shape.

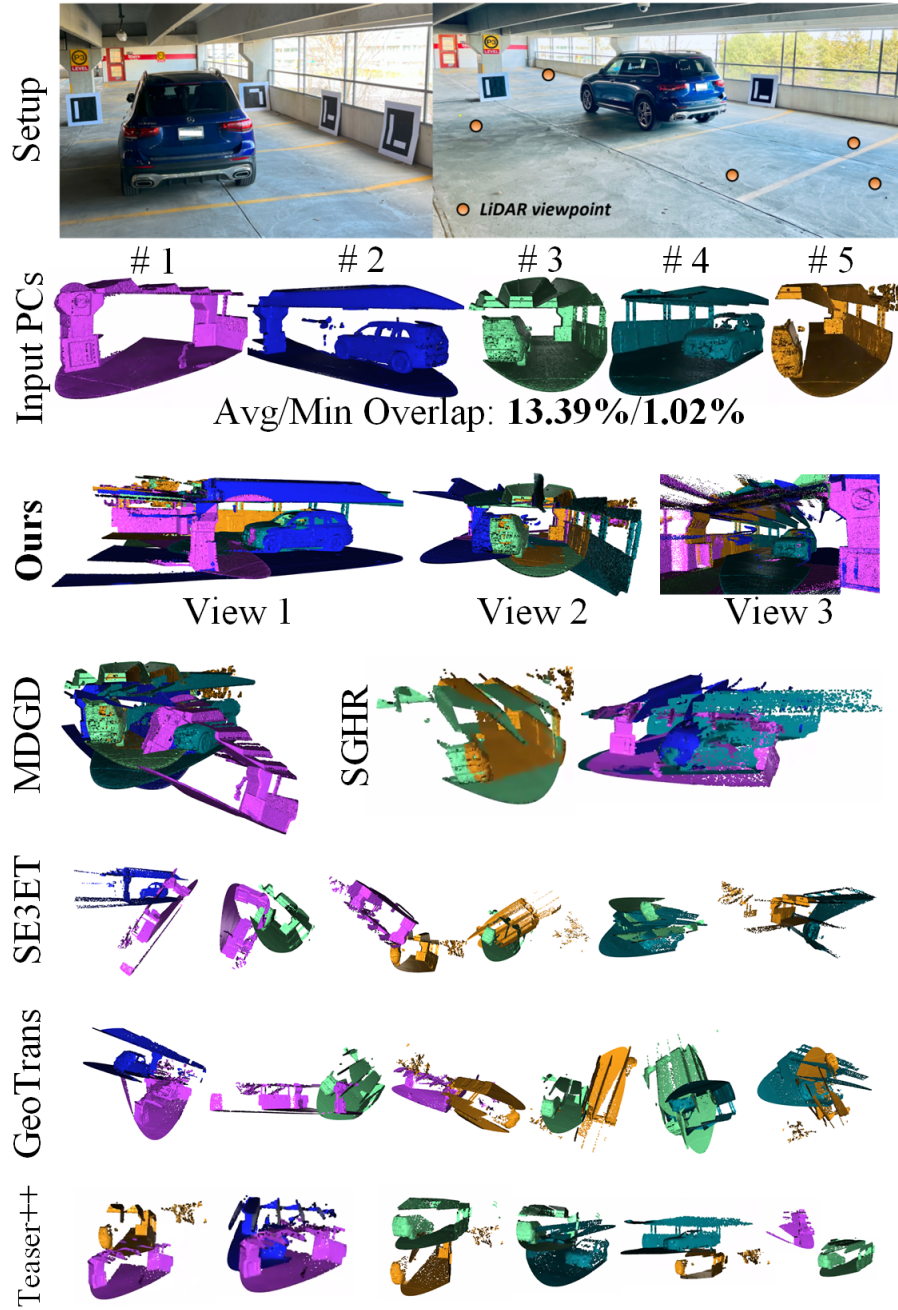


Figure 4.8: An illustration of the experimental setup and a visual comparison of the proposed method against the SOTA methods (MDGD [2], SGHR [3], SE3ET [4], GeoTrans [5], and Teaser++ [6]) regarding instance reconstruction from sparse scans.

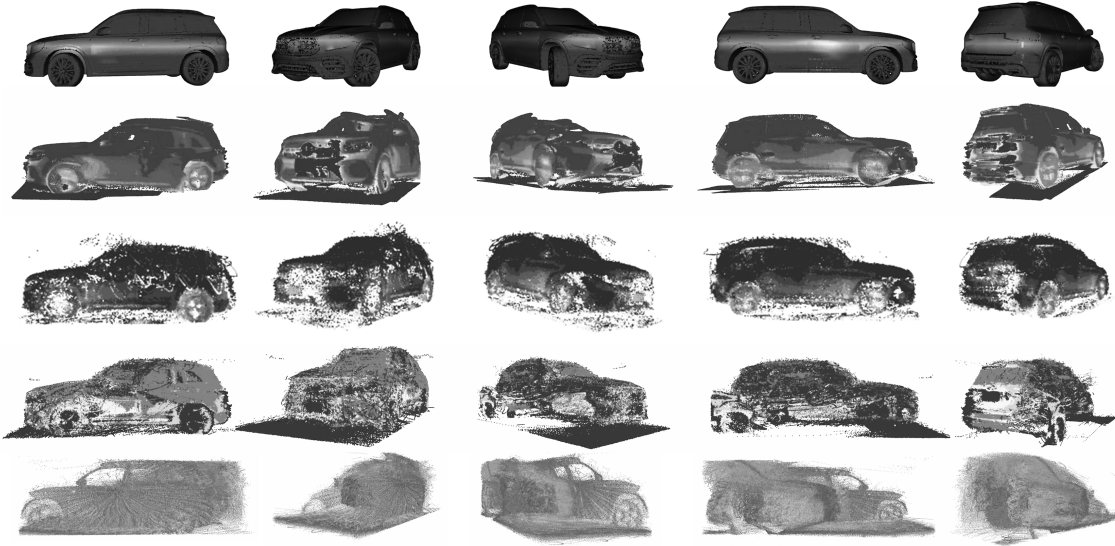


Figure 4.9: Visual comparison of the instance reconstruction. From top to bottom: ground truth, Ours, Traj LO [10], Livox Mapping [12], and LOAM Livox [13].

Table 4.3: Comparison with LOAM methods regarding reconstruction

Method \ Metric	CD ↓	Recall (%) ↑
Livox Mapping [12]	0.0106	75.27
LOAM Livox [13]	0.0335	78.82
Traj LO [10]	0.0107	82.88
Ours	0.0030	96.22

4.4.5 Application 2: Training Data Collection and Enhancement of Existing Learning-Based Methods

Training data is crucial for learning-based methods. Most existing methods [2,3] are trained on the 3DMatch dataset [35]. Augmenting the training data is beneficial for improving the generalization ability of learning-based methods. However, the existing methods cannot be utilized for collecting training data in unseen scenes, as unseen scenes imply out-of-

distribution cases and the limited effectiveness of the existing methods. The proposed method can serve as a valuable tool for collecting training data. In this test, we demonstrate that training data collected using our method can enhance the performance of the state-of-the-art methods across various benchmarks.

Data. Using the proposed method, all the point clouds shown in Figs. 4.7 and 4.8, comprising 11 scenes with 33 scans, are aligned and processed into the format required for training by MDGD [2] and SGHR [3]. The newly collected data is named Livox-3DMatch. We train the models [2,3] from scratch using only 3DMatch and a combination of 3DMatch and Livox-3DMatch.

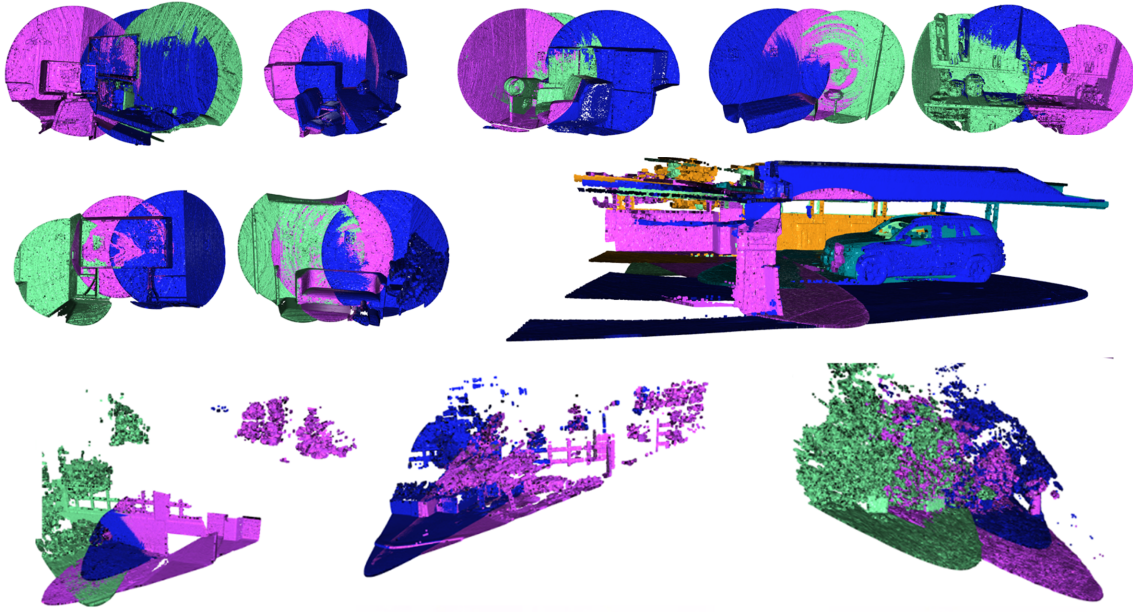


Figure 4.10: An illustration of the scenes in Livox-3DMatch.

Benchmarks and Metrics. We compare the performance of the models [2, 3] trained with and without our data on three popular benchmarks: 3DMatch [35], ETH [36], and ScanNet [37]. Following [3], Registration Recall (RR) is used to evaluate performance on 3DMatch [35] and ETH [36], while RMSEs are used to report performance on ScanNet [37].

RR is defined as follows:

$$\text{RR} = \frac{N_{\text{success}}}{N_{\text{total}}}. \quad (4.12)$$

where N_{success} refers to the number of successful registrations, and N_{total} be the total number of registration attempts. For the registration to be considered successful, the RMSEs (See Eq. (4.9)) must be below a predefined threshold.

Results and Analysis. First, the number of pairs in the training data increases from 14,400 to **17,700**, a boost of **22.91%**, thanks to the introduction of our data. The quantitative results are reported in Table 4.4. As seen, the RR of SGHR [3] is increased by **2.90%** on 3DMatch [35] and **4.29%** on ETH [36]. The translation error and rotation error on ScanNet [37] are decreased by **22.72%** and **11.19%**, respectively. The RR of MDGD [2] is improved by **1.71%** on 3DMatch [35] and **2.89%** on ETH [36]. The translation error and rotation error on ScanNet [37] are decreased by **22.45%** and **7.80%**, respectively. The reason behind these improvements is mentioned in Section 4.3: the introduction of our data enriches the learnable features during training, which benefits the generalizability of the models.

Table 4.4: Quantitative evaluation of the enhancement of SGHR [3] and MDGD [2] due to the addition of Livox-3DMatch to the training.

Method	Dataset	RR $\uparrow$ (%)	Dataset	RR $\uparrow$ (%)	Dataset	RMSE _T (m) $\downarrow$ Mean/Med	RMSE _R (deg) $\downarrow$ Mean/Med
SGHR [3]	3DMatch [35]	92.68	ETH [36]	93.74	ScanNet [37]	0.66/0.51	23.50/22.08
SGHR [3] + Livox-3DMatch (our data)		95.58		98.03		0.51/0.45	20.87/17.21
MDGD [2]		94.26		96.06		0.49/0.44	20.51/19.82
MDGD [2] + Livox-3DMatch (our data)		95.97		98.95		0.38/0.31	18.91/17.36

4.4.6 Application 3: Reconstructing a Degraded Scene

An important application of fiducial markers in the real world is enhancing the robustness of reconstruction and localization in degraded scenes when additional sensors are unavailable.

Therefore, in this test, we evaluate our method in a textureless, degraded scene.

Data. The setup is shown in the reference of Fig. 4.11. This scenario has repetitive structures and weak geometric features. We attach thirteen $16.4\text{ cm} \times 16.4\text{ cm}$ AprilTags to the wall. The LiDAR scans the scene from 11 viewpoints. We also captured 72 images with an iPhone 13 to use as input for SfM-M [7]. The ground truth trajectories are given by an OptiTrack Motion Capture system.

Competitors and Metrics. The competitors are the SOTA point cloud registration methods [2–6]. However, considering these competitors struggle in the degraded scene, we add SfM-M [7], the SOTA marker-based SfM method, as a competitor. This addition enables a comprehensive and meaningful comparison. We employ the RMSEs as the metric.

Results and Analysis. The qualitative results are presented in Fig. 4.11. As seen, none of the point cloud registration methods [2–6] can align a single pair of point clouds. This is because these cases only have planar overlap regions and thus are even more challenging than those in Fig. 4.7 and Fig. 4.8. In comparison, the proposed method and SfM-M [7] successfully reconstruct this degraded scene as they utilize the fiducial markers. The comparison of these two methods in terms of sensor trajectories is shown in Fig. 4.12. The quantitative results shown in Table 4.5 demonstrate that the proposed approach achieves better localization accuracy, which is expected given that LiDAR is a ranging sensor.

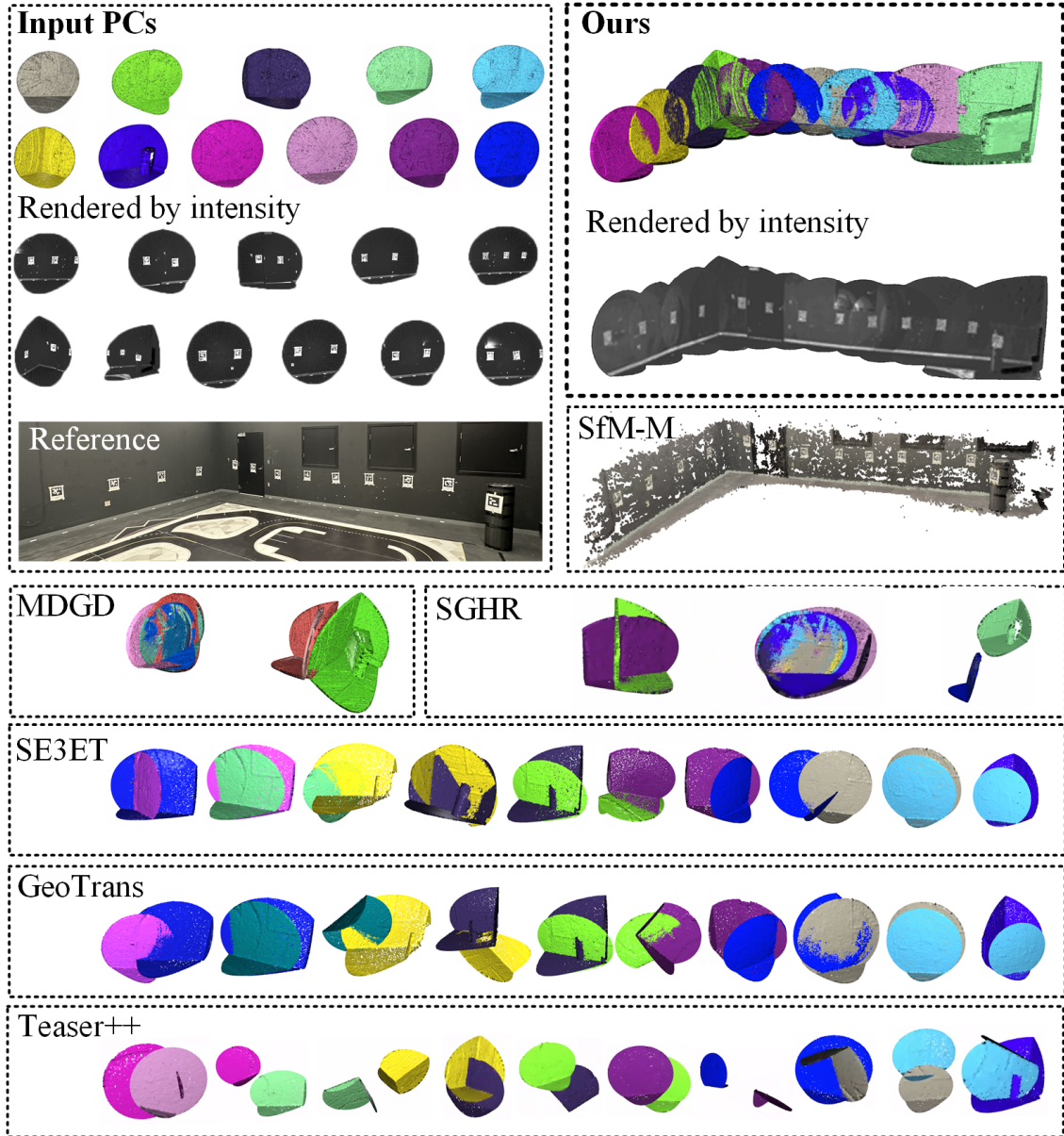


Figure 4.11: An illustration of the experimental setup and a visual comparison of the proposed method against the SOTA methods (MDGD [2], SGHR [3], SE3ET [4], GeoTrans [5], Teaser++ [6], and SfM-M [7]) in a degraded scene.

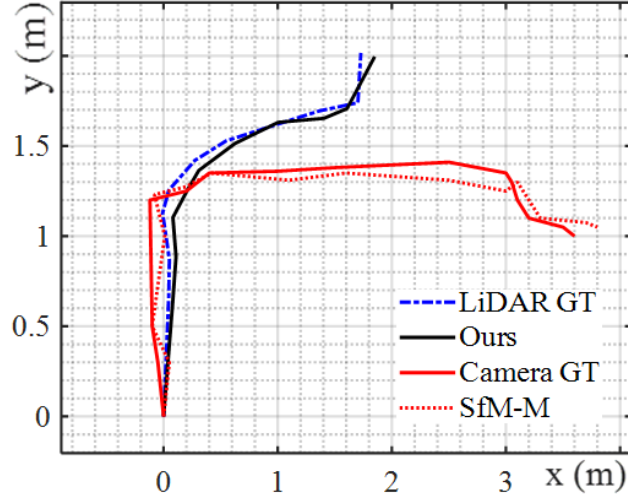


Figure 4.12: A comparison of the sensor trajectories obtained from the proposed method and SfM-M [7]. G.T. refers to the ground truth.

Table 4.5: Comparison with SfM-M [7] regarding localization accuracy in a degraded scene.

Method \ Metric	RMSE _T (m) ↓	RMSE _R (rad) ↓
SfM-M [7]	0.0551	0.0491
Ours	0.0490	0.0384

4.4.7 Application 4: Localization in a GPS-denied Environment

Localization is also a crucial application of point cloud registration methods. Fiducial markers are a popular tool for providing localization information in GPS-denied environments, such as indoor parking lots. In this test, we evaluate the proposed method in this context.

Data. The experimental setup is shown in Fig. 4.13(a): four $69.2 \text{ cm} \times 69.2 \text{ cm}$ ArUco [18] markers are deployed in the environment. The vehicle, equipped with an RS-Ruby 128-beam mechanical LiDAR, follows an 8-shaped trajectory without pausing and samples 364

LiDAR scans. We conduct the experiment on the roof of a large parking lot to acquire the ground truth trajectory from the Real-Time Kinematic.

Competitor and Metric. Note that most LO methods are limited to specific LiDAR models, and modifying the method to accommodate the features of a particular LiDAR model is not trivial [13]. Considering this, the choice is made to compare with KISS-ICP [8], the state-of-the-art general pure LO method, which can be directly applied to the RS-Ruby 128 LiDAR without requiring modifications. Again, RMSEs are employed as the metric.

Results and Analysis. The visual comparison and quantitative comparison of localization results are presented in Fig. 4.13(b) and Table 4.6, respectively. Our method exhibits less drift in the middle of the trajectory and demonstrates better overall localization accuracy.

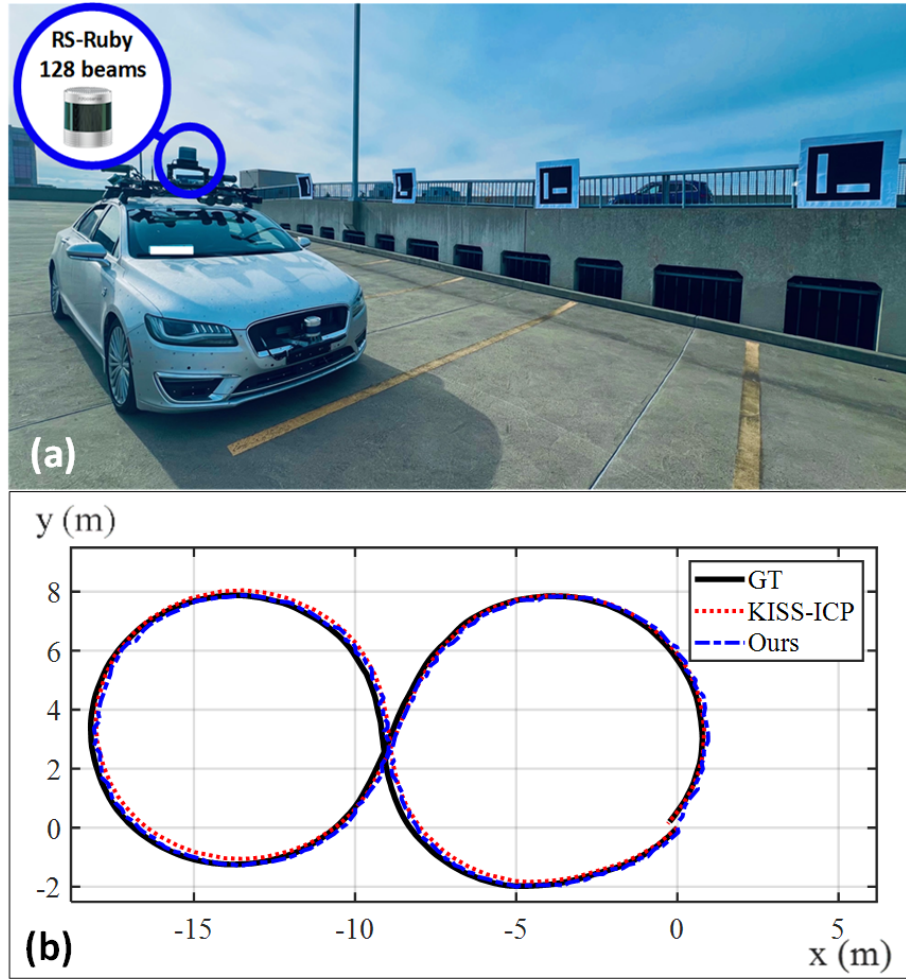


Figure 4.13: (a): An illustration of the experimental setup. (b): Comparison of the trajectories given by different methods.

Table 4.6: Comparison with KISS-ICP [8] regarding localization.

Method \ Metric	$RMSE_T$ (m) $\downarrow$	$RMSE_R$ (rad) $\downarrow$
KISS-ICP [8]	0.1976	0.1617
Ours	0.1715	0.1394

4.4.8 Application 5: 3D Map Merging

To validate the performance of the proposed method in large-scale scenarios, we apply the proposed method to the 3D map merging task in this test, which involves merging multiple large-scale, low overlap 3D maps into a single frame.

Data. We collected three large-scale LiDAR maps. They are constructed using the SOTA LiDAR-based SLAM method, Traj-LO [10], by scanning the York University campus with a Livox MID-40 LiDAR. The ground truth poses between the maps are manually obtained using *CloudCompare* [77].

Competitors and Metrics. The competitors are SOTA multiview point cloud registration methods, including MDGD [2] and SGHR [3]. Moreover, unlike previous tests, the LiDAR fiducial markers on the 3D maps are localized using the algorithm from our previous work [81], and the marker detection results serve as input for the our pipeline. We employ the RMSEs as the metric.

Results and Analysis. The visual and quantitative comparisons are shown in Fig. 4.14 and Table 4.7, respectively. As shown in Fig. 4.14, neither MDGD nor SGHR can address this challenging task. This is due to the fact that the overlap in large-scale maps is too scarce. Specifically, the overlap rates [45] are 4.87% between map 1 and map 2, 3.96% between map 1 and map 3, and 2.59% between map 2 and map 3. On the other hand, both MDGD and SGHR start by analyzing the features of each individual point cloud. As the scale of the point cloud becomes larger, the portion of the features belonging to the overlap regions decreases, making them unsuitable for large-scale, low overlap map merging. In addition, as the scale of point clouds becomes larger, the absolute error values of MDGD and SGHR also increase. In comparison, even though the scales of the point clouds in this test are much larger than those in the previous test, the proposed method successfully merges these large-scale, low overlap 3D maps. This stems from the observation that the proposed

method focuses on utilizing the thin-sheet LiDAR fiducial markers and is not sensitive to scale changes. As introduced in [81], the accuracy of marker localization degrades as the scale increases. Thus, the absolute error values of our method also increase compared to previous tests.

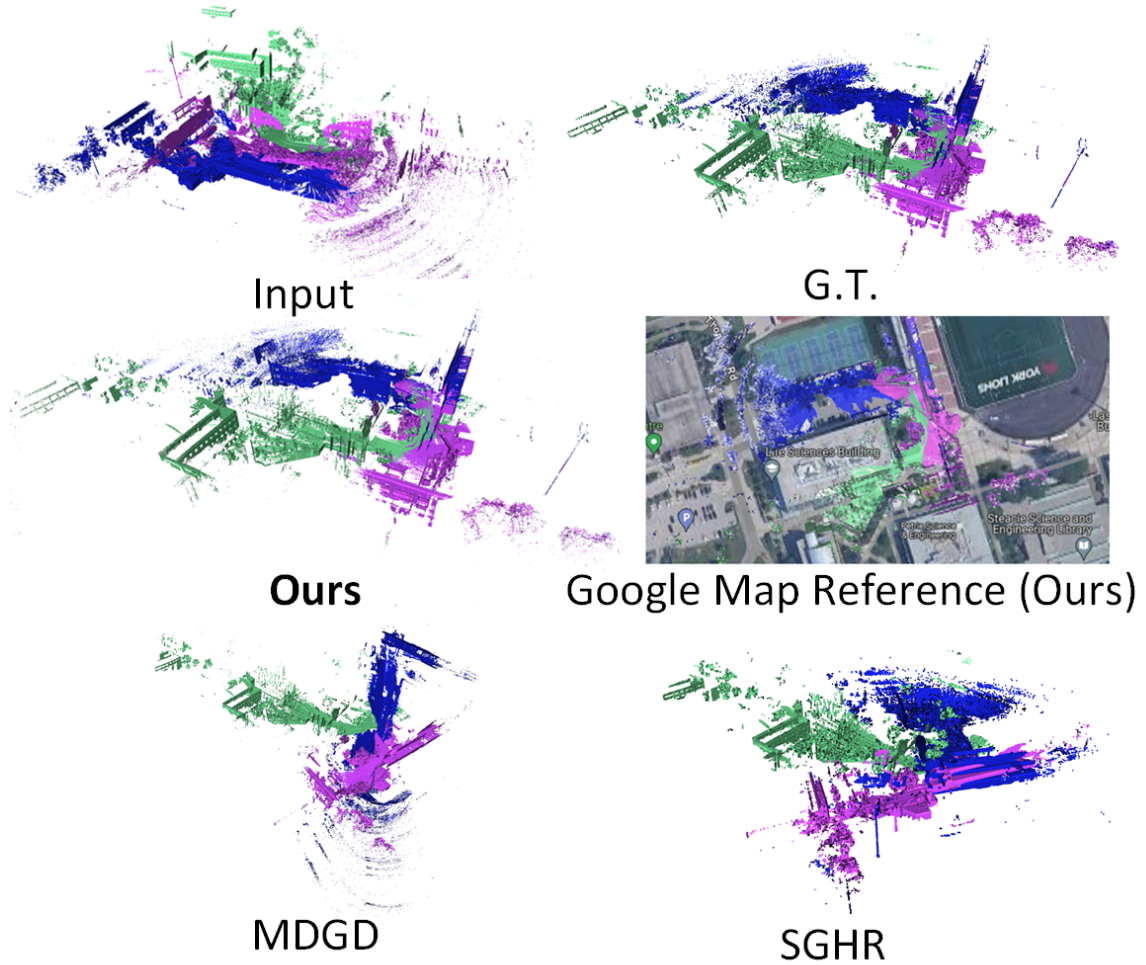


Figure 4.14: A comparison of the 3D map merging results of MDGD [2], SGHR [3], and our method.

Table 4.7: Comparison of 3D map merging results between MDGD [2], SGHR [3], and our method.

Method \ Metric	RMSE _T (m) ↓	RMSE _R (rad) ↓
MDGD [2]	2.7883	1.9210
SGHR [3]	4.0933	2.3925
Ours	0.2305	0.1771

4.4.9 Ablation Studies

The proposed framework is composed of two levels of graphs. To demonstrate the necessity of this overall architecture, we conduct ablation studies in this section. In particular, we study the effects of removing the first and second graphs in two cases: Fig. 4.8 and the kitchen scene in Fig. 4.7. The visual comparison and quantitative results are shown in Fig. 4.15 and Table 4.8, respectively. When the first graph is removed, the factors representing the relative poses between frames in Fig. 3.10 have to be set to identity due to the lack of initial values provided by the first graph. However, the role of the second graph is to further optimize the variables based on their initial values rather than finding the optimal solution from scratch. Consequently, as shown in Fig. 4.15 and Table 4.8, severe misalignment or degradation in registration accuracy occurs. When the second graph is removed, the multiview point clouds are directly registered using the initial values obtained from the first graph, without any further refinement. As shown in Fig. 4.15 and Table 4.8, removing the second graph causes a slight degradation in registration accuracy. Moreover, the degradation in the kitchen case of Fig. 4.7 is slighter than that in Fig. 4.8. This is because the effect of the second graph is case-by-case, determined by the quality of the initial values provided by the first graph. Namely, the better the initial values are, the less important the second graph becomes. However, in the real world, pose estimation of markers cannot be perfect. In addition, as seen in Fig. 4.4, we also add the marker corners

to the second graph so that the pose estimation of each individual marker can be further optimized along with other variables in the graph. Therefore, it is beneficial to apply the second graph in practice. In summary, the proposed framework adopts a coarse-to-fine pipeline, where the first graph corresponds to the coarse stage, while the second graph is the fine stage. Removing any of them will cause performance degradation.

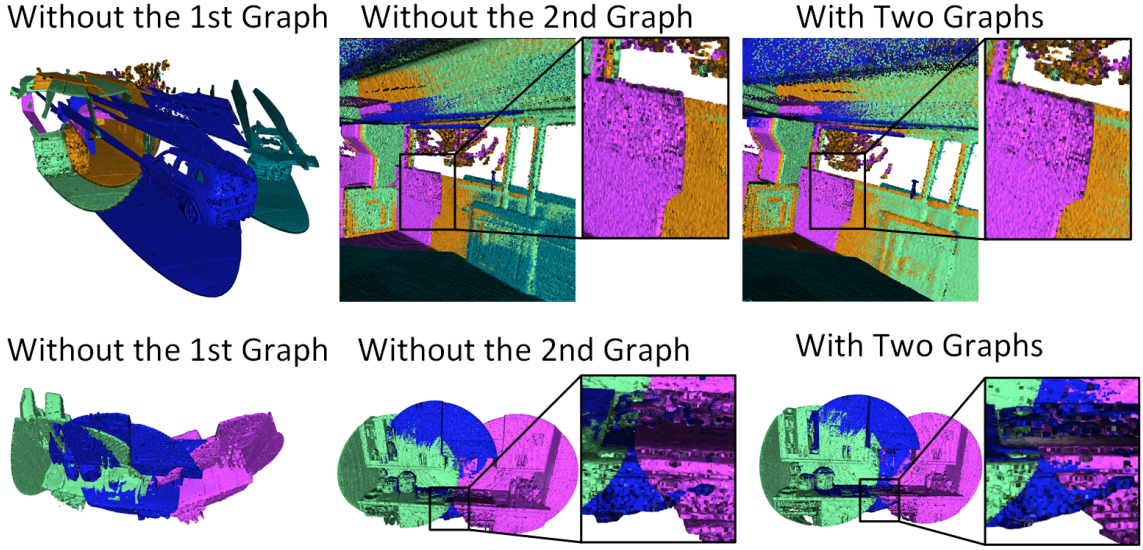


Figure 4.15: A comparison of the point cloud registration results without the first graph, without the second graph, and with both graphs for two cases.

Table 4.8: Ablation studies on the first and second graphs.

Scene	Framework	RMSE_T (m) $\downarrow$	RMSE_R (rad) $\downarrow$
Fig. 4.8	without first graph	0.431	0.459
	without second graph	0.078	0.084
	with both graphs	0.066	0.072
Fig. 4.7 (Kitchen)	without first graph	0.088	0.112
	without second graph	0.021	0.073
	with both graphs	0.019	0.067

4.4.10 Limitations

Despite the promising results of the proposed algorithm, there are potential limitations. Firstly, although LiDARs are robust to unideal illumination conditions, adverse weather phenomena such as rain, snow, and fog can affect LiDAR measurements, thereby reducing the effectiveness of the proposed method. Secondly, the adopted low-cost LFMs, made of thin-sheet paper or boards, may deform after long-term use in the wild. However, this is not a concern for one-time applications such as data collection. Finally, deploying the LFMs requires some labor, but their value is demonstrated in this dissertation.

5 Conclusion and Future Work

5.1 Conclusion

To facilitate robotics and computer vision applications, a LiDAR-based mapping and localization method is needed that can robustly handle unordered, low overlap, multiview point clouds simultaneously, similar to how Structure-from-Motion (SfM) for cameras processes unordered images all at once. However, unlike in the field of SfM, where the use of visual fiducial markers (VFMs) has been widely studied, the development of the LiDAR fiducial marker (LFM) and its utilization in such a framework remains a technological gap. This dissertation addresses these pressing issues from the following perspectives.

To address the lack of a general LFM system that can be applied to different LiDAR models and is compatible with popular VFM patterns, a novel intensity image-based LiDAR fiducial marker (IFM) system is proposed.

- Unlike LiDARTag [9] which requires extra 3D objects to be added to the environment, the usage of the IFM is as convenient and easy as the VFM systems [1, 18]. Namely, the users can produce the marker by printing the VFM on regular letter-size paper with a regular printer and then place the marker anywhere they like.
- A novel marker detection method is proposed to detect 3D fiducials through the intensity image. Thanks to this, the VFM systems proposed in the past, present,

and even future can be easily embedded into the IFM system. This is a superiority of the proposed system over LiDARTag [9] which only supports square markers with patterns from AprilTag 3 [1].

- A pose estimation approach for the LiDAR via the proposed IFM is introduced, which has better accuracy than the VFM-based pose estimation for the camera. In addition, unlike the VFM system, the proposed pose estimation is free from the rotation ambiguity problem [52, 53] and is robust to changes in ambient light.

Due to the adoption of 3D-to-2D spherical projection, the vanilla IFM exhibits two limitations: (i) IFM can only detect fiducials in a single-view point cloud and does not apply to a 3D LiDAR map, and (ii) as the distance between the tag and the LiDAR increases, the projection size of the tag decreases until it is too small to be detected. In response to these limitations, a novel method has been developed to improve the localization of thin-sheet LFMs.

- The LFM localization is extended from the single-view point cloud to the 3D LiDAR map and the detectable distance of LFMs is enhanced.
- A new pipeline for jointly analyzing a 3D point cloud from both intensity and geometry perspectives is proposed, as fiducial tags are planar objects with high-intensity contrast and are indistinguishable from the plane to which they are attached. This differs from conventional 3D object detection methods that rely merely on 3D geometric features and can only detect spatially distinguishable objects.

Finally, building upon the developed algorithms for LFM detection, this dissertation discusses how to exploit LFMs for mapping and localization.

- A novel framework for mapping and localization using LFMs is designed, where mapping and localization are achieved by registering unordered multiview point clouds,

even with low overlap. The proposed framework serves as an efficient and reliable tool for diverse robotics and computer vision applications, including 3D asset collection, training data collection, reconstruction of degraded scenes, localization in GPS-denied environments, and 3D map merging.

- A new training dataset called Livox-3DMatch using the proposed framework is collected, which augments the original 3DMatch training data [35] from 14,400 pairs to 17,700 pairs (a 22.91% increase). Training on this augmented dataset improves the performance of the SOTA learning-based methods on various benchmarks. In particular, the RR of SGHR [3] is increased by 2.90% on 3DMatch [35] and 4.29% on ETH [36]. The translation error and rotation error on ScanNet [37] are decreased by 22.72% and 11.19%, respectively. The RR of MDGD [2] increases by 1.71% on 3DMatch [35] and 2.89% on ETH [36]. The translation error and rotation error on ScanNet [37] decrease by 22.45% and 7.80%, respectively.
- An adaptive threshold LFM detection algorithm that is robust to viewpoint changes in the wild is developed.

5.2 Future Work

In this dissertation, a framework for mapping and localization using LFMs is proposed to benefit various robotic and computer vision applications. While the results are promising, several issues remain that require further exploration. The following are potential future research directions to enhance this work.

The first issue is the utilization of learning-based methods in LFM detection. Since the proposed IFM is open-source, much inspiring user feedback has been received, which also motivated the reconsideration of LFM detection. For example, both LiDARTag [9]

and IFM [39] propose introducing the encoding-decoding algorithms of VFM systems in LFM detection. However, since the 'code' of a marker's pattern typically exceeds 16 bits—meaning more than 16 black-and-white blocks—the LiDAR must have a high resolution to fully capture the code/bit information. Based on feedback from users in the autonomous driving industry who use LFM for calibration, marker detection sometimes fails as the bit information is lost when the distance increases. Unfortunately, this is caused by the sparse features of the LiDAR data, which have not been adequately addressed so far. Despite this, a notable technology called LiDAR intensity densification [82], which learns to generate densified LiDAR intensity data from sparse and occluded LiDAR data, has the potential to mitigate this problem. Another solution that has a similar effect is Neural LiDAR Fields [83], which learns LiDAR-based novel view synthesis and supports the upsampling of LiDAR data. Moreover, considering that many 3D object detection and segmentation models are designed to learn directly from sparse and occluded data, it is possible to tackle this problem using a learning-based method. Nevertheless, it should be noted that the lack of corresponding training data presents a significant challenge. In particular, the training data should include a substantial number of annotated thin-sheet LFMs, and the model should be designed to learn detection by utilizing intensity information.

The second issue is the need to collect more data using the proposed framework. Although a new dataset named Livox-3DMatch has been collected for point cloud registration research in this dissertation, it would be beneficial to use the proposed framework to collect additional data and a greater variety of datasets. For example, 3D assets of irregular vehicles, such as construction trucks equipped with various machines, are rare but crucial for autonomous driving simulation. Thus, collecting more 3D assets of irregular vehicles using the proposed framework is advantageous. LiDAR-based novel view synthesis is an

emerging topic. In camera-based research, large-scale datasets like OmniObject3D [49] provide extensive object-level posed images. Developing a similar dataset for LiDAR that offers posed point clouds would enhance LiDAR-based novel view synthesis [83,84], which is a significant downstream and future application of the proposed framework.

The third issue is to explore other types of materials and formats to construct markers for better utilization of the intensity information. In this research, we only study black-and-white markers made of paper, as we aim to develop a low-cost solution. However, for scenes where the distance between the fiducial and the sensor is hundreds of meters, it is beneficial to introduce fiducial objects such as prisms. Moreover, by leveraging other materials and formats, such as prisms, to construct LFMs, it is feasible to eliminate the reliance on 2D space, while keeping the system as unified as prisms work for different LiDAR models.

The fourth issue is to fuse LFM detection with VFM detection for a robust autonomous system. Although utilizing the algorithms proposed in this research, it is feasible to detect thin-sheet markers with LiDARs alone, redundant design is considered necessary in autonomous systems nowadays. As introduced in the dissertation, LFM detection can help address rotational ambiguity and unideal illumination conditions, which are challenging for VFM detection. On the other hand, there are many mature computer vision algorithms to derain, defog, and desnow, which are challenging for LFM detection. In addition, fusing LFM and VFM detection and constructing the pose estimation of an autonomous system as a global optimization problem has the potential to provide better accuracy.

References

- [1] M. Krogius, A. Haggemiller, and E. Olson, “Flexible layouts for fiducial tags,” in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2019, pp. 1898–1903.
- [2] S. Li, J. Zhu, Y. Xie, N. Hu, and D. Wang, “Matching distance and geometric distribution aided learning multiview point cloud registration,” *IEEE Robotics and Automation Letters*, vol. 9, no. 11, pp. 9319–9326, 2024.
- [3] H. Wang, Y. Liu, Z. Dong, Y. Guo, Y.-S. Liu, W. Wang, and B. Yang, “Robust multiview point cloud registration with reliable pose graph initialization and history reweighting,” in *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 9506–9515.
- [4] C. E. Lin, M. Zhu, and M. Ghaffari, “Se3et: Se(3)-equivariant transformer for low-overlap point cloud registration,” *IEEE Robotics and Automation Letters*, vol. 9, no. 11, pp. 9526–9533, 2024.
- [5] Z. Qin, H. Yu, C. Wang, Y. Guo, Y. Peng, S. Ilic, D. Hu, and K. Xu, “Geotransformer: Fast and robust point cloud registration with geometric transformer,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 8, pp. 9806–9821, 2023.

- [6] H. Yang, J. Shi, and L. Carlone, “Teaser: Fast and certifiable point cloud registration,” *IEEE Transactions on Robotics*, vol. 37, no. 2, pp. 314–333, 2020.
- [7] Z. Jia, Y. Rao, H. Fan, and J. Dong, “An efficient visual sfm framework using planar markers,” *IEEE Transactions on Instrumentation and Measurement*, vol. 72, pp. 1–12, 2023.
- [8] I. Vizzo, T. Guadagnino, B. Mersch, L. Wiesmann, J. Behley, and C. Stachniss, “KISS-ICP: In Defense of Point-to-Point ICP – Simple, Accurate, and Robust Registration If Done the Right Way,” *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 1029–1036, 2023.
- [9] J.-K. Huang, S. Wang, M. Ghaffari, and J. W. Grizzle, “Lidartag: A real-time fiducial tag system for point clouds,” *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4875–4882, 2021.
- [10] X. Zheng and J. Zhu, “Traj-lo: In defense of lidar-only odometry using an effective continuous-time trajectory,” *IEEE Robotics and Automation Letters*, vol. 9, no. 2, pp. 1961–1968, 2024.
- [11] E. W. Dijkstra, “A note on two problems in connexion with graphs,” in *Edsger Wybe Dijkstra: His Life, Work, and Legacy*, 2022, pp. 287–290.
- [12] Livox-SDK. Livox Mapping. (2020). [Online]. Available: https://github.com/Livox-SDK/livox_mapping
- [13] J. Lin and F. Zhang, “Loam livox: A fast, robust, high-precision lidar odometry and mapping package for lidars of small fov,” in *Proc. IEEE International Conference on Robotics and Automation*, 2020, pp. 3126–3131.

- [14] J. Zhang and S. Singh, “Loam: Lidar odometry and mapping in real-time.” in *Robotics: Science and systems*, vol. 2, no. 9. Berkeley, CA, 2014, pp. 1–9.
- [15] L. Fan, X. Xiong, F. Wang, N. Wang, and Z. Zhang, “Rangedet: In defense of range view for lidar-based 3d object detection,” in *Proc. IEEE/CVF International Conference on Computer Vision*, 2021.
- [16] J. Wang and E. Olson, “Apriltag 2: Efficient and robust fiducial detection,” in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2016, pp. 4193–4198.
- [17] E. Olson, “Apriltag: A robust and flexible visual fiducial system,” in *Proc. IEEE International Conference on Robotics and Automation*, 2011, pp. 3400–3407.
- [18] F. J. Romero-Ramirez, R. Muñoz-Salinas, and R. Medina-Carnicer, “Speeded up detection of squared fiducial markers,” *Image and vision Computing*, vol. 76, pp. 38–47, 2018.
- [19] L. Calvet, P. Gurdjos, C. Griwodz, and S. Gasparini, “Detection and Accurate Localization of Circular Fiducials under Highly Challenging Conditions,” in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 562 – 570.
- [20] D. Avola, L. Cinque, G. L. Foresti, C. Mercuri, and D. Pannone, “A practical framework for the development of augmented reality applications by using aruco markers,” in *Proc. International Conference on Pattern Recognition Applications and Methods*, vol. 2, 2016, pp. 645–654.
- [21] Y. Liu, H. Schofield, and J. Shan, “Navigation of a self-driving vehicle using one fiducial marker,” in *Proc. IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, 2021, pp. 1–6.

- [22] Y. Liu, A. Haridevan, H. Schofield, and J. Shan, “Application of ghost-deblurgan to fiducial marker detection,” in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2022, pp. 6827–6832.
- [23] T. Liao, A. Haridevan, Y. Liu, and J. Shan, “Autonomous vision-based uav landing with collision avoidance using deep learning,” in *Science and Information Conference*. Springer, 2022, pp. 79–87.
- [24] J. Rehder, J. Nikolic, T. Schneider, T. Hinzmann, and R. Siegwart, “Extending kalibr: Calibrating the extrinsics of multiple imus and of individual axes,” in *Proc. IEEE International Conference on Robotics and Automation*, 2016, pp. 4304–4311.
- [25] J.-K. Huang and J. W. Grizzle, “Improvements to target-based 3d lidar to camera calibration,” *IEEE Access*, vol. 8, pp. 134 101–134 110, 2020.
- [26] R. Muñoz-Salinas, M. J. Marín-Jimenez, E. Yeguas-Bolivar, and R. Medina-Carnicer, “Mapping and localization from planar markers,” *Pattern Recognition*, vol. 73, pp. 158–171, 2018.
- [27] R. Muñoz-Salinas, M. J. Marin-Jimenez, and R. Medina-Carnicer, “Spm-slam: Simultaneous localization and mapping with squared planar markers,” *Pattern Recognition*, vol. 86, pp. 156–171, 2019.
- [28] S. Zhang, J. Shan, and Y. Liu, “Variational bayesian estimator for mobile robot localization with unknown noise covariance,” *IEEE/ASME Transactions on Mechatronics*, vol. 27, no. 4, pp. 2185–2193, 2022.
- [29] —, “Particle filtering on lie group for mobile robot localization with range-bearing measurements,” *IEEE Control Systems Letters*, vol. 7, pp. 3753–3758, 2023.

- [30] S. Jin, I. Armeni, M. Pollefeys, and D. Barath, “Multiway point cloud mosaicking with diffusion and global optimization,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 20 838–20 849.
- [31] S. Zhang, J. Shan, and Y. Liu, “Approximate inference particle filtering for mobile robot slam,” *IEEE Transactions on Automation Science and Engineering*, pp. 1–12, 2024.
- [32] Z. Yang, S. Manivasagam, Y. Chen, J. Wang, R. Hu, and R. Urtasun, “Reconstructing objects in-the-wild for realistic sensor simulation,” in *Proc. of IEEE International Conference on Robotics and Automation*, 2023, pp. 11 661–11 668.
- [33] P. Sun, H. Kretzschmar, X. Dotiwalla, A. Chouard, V. Patnaik, P. Tsui, J. Guo, Y. Zhou, Y. Chai, B. Caine *et al.*, “Scalability in perception for autonomous driving: Waymo open dataset,” in *Proc. of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 2446–2454.
- [34] Y. Liu, K. Zhu, G. Wu, Y. Ren, B. Liu, Y. Liu, and J. Shan, “Mv-deepsdf: Implicit modeling with multi-sweep point clouds for 3d vehicle reconstruction in autonomous driving,” in *Proc. of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 8306–8316.
- [35] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser, “3dmatch: Learning local geometric descriptors from rgb-d reconstructions,” in *Proc. of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1802–1811.
- [36] F. Pomerleau, M. Liu, F. Colas, and R. Siegwart, “Challenging data sets for point cloud registration algorithms,” *The International Journal of Robotics Research*, vol. 31, no. 14, pp. 1705–1711, 2012.

- [37] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, “Scannet: Richly-annotated 3d reconstructions of indoor scenes,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5828–5839.
- [38] S. Duggal, Z. Wang, W.-C. Ma, S. Manivasagam, J. Liang, S. Wang, and R. Urtasun, “Mending neural implicit modeling for 3d vehicle reconstruction in the wild,” in *Proc. of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 1900–1909.
- [39] Y. Liu, H. Schofield, and J. Shan, “Intensity image-based lidar fiducial marker system,” *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6542–6549, 2022.
- [40] R. Muñoz-Salinas, M. J. Marín-Jimenez, E. Yeguas-Bolivar, and R. Medina-Carnicer, “Mapping and localization from planar markers,” *Pattern Recognition*, vol. 73, pp. 158–171, 2018.
- [41] J. Beltrán, C. Guindel, A. de la Escalera, and F. García, “Automatic extrinsic calibration method for lidar and camera sensor setups,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 10, pp. 17 677–17 689, 2022.
- [42] L. Tao, L. Pei, T. Li, D. Zou, Q. Wu, and S. Xia, “Cpi: Lidar-camera extrinsic calibration based on feature points with reflection intensity,” in *Spatial Data and Intelligence: First International Conference, SpatialDI 2020, Virtual Event, May 8–9, 2020, Proceedings 1*. Springer, 2021, pp. 281–290.
- [43] Y. Xie, L. Deng, T. Sun, Y. Fu, J. Li, X. Cui, H. Yin, S. Deng, J. Xiao, and B. Chen, “A4lidartag: Depth-based fiducial marker for extrinsic calibration of solid-state lidar and camera,” *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 6487–6494, 2022.

- [44] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.
- [45] S. Huang, Z. Gojcic, M. Usvyatsov, A. Wieser, and K. Schindler, “Predator: Registration of 3d point clouds with low overlap,” in *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2021, pp. 4267–4276.
- [46] C. Yang, Z. Chai, X. Yang, H. Zhuang, and M. Yang, “Recognition of degradation scenarios for lidar slam applications,” in *Proc. IEEE International Conference on Robotics and Biomimetics*. IEEE, 2022, pp. 1726–1731.
- [47] J. L. Schonberger and J.-M. Frahm, “Structure-from-motion revisited,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4104–4113.
- [48] S. Wang, V. Leroy, Y. Cabon, B. Chidlovskii, and J. Revaud, “Dust3r: Geometric 3d vision made easy,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 20 697–20 709.
- [49] T. Wu, J. Zhang, X. Fu, Y. Wang, J. Ren, L. Pan, W. Wu, L. Yang, J. Wang, C. Qian *et al.*, “Omniobject3d: Large-vocabulary 3d object dataset for realistic perception, reconstruction and generation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 803–814.
- [50] C. Lu, S. Xia, M. Shao, and Y. Fu, “Arc-support line segments revisited: An efficient high-quality ellipse detection,” *IEEE Transactions on Image Processing*, vol. 29, pp. 768–781, 2019.

- [51] C. Lu, S. Xia, W. Huang, M. Shao, and Y. Fu, “Circle detection by arc-support line segments,” in *Proc. IEEE International Conference on Image Processing*, 2017, pp. 76–80.
- [52] T. Collins and A. Bartoli, “Infinitesimal plane-based pose estimation,” *International journal of computer vision*, vol. 109, no. 3, pp. 252–286, 2014.
- [53] Y. Liu, H. Schofield, and J. Shan, “Navigation of a self-driving vehicle using one fiducial marker,” in *Proc. IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, 2021, pp. 1–6.
- [54] S.-F. Ch’ng, N. Sogi, P. Purkait, T.-J. Chin, and K. Fukui, “Resolving marker pose ambiguity by robust rotation averaging with clique constraints,” in *Proc. IEEE International Conference on Robotics and Automation*, 2020, pp. 9680–9686.
- [55] M. Vaidis, P. Giguère, F. Pomerleau, and V. Kubelka, “Accurate outdoor ground truth based on total stations,” in *2021 18th Conference on Robots and Vision (CRV)*, 2021, pp. 1–8.
- [56] P. Furgale, J. Rehder, and R. Siegwart, “Unified temporal and spatial calibration for multi-sensor systems,” in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 1280–1286.
- [57] A. D. Sappa and M. Devy, “Fast range image segmentation by an edge detection strategy,” in *Proc. International Conference on 3-D Digital Imaging and Modeling*, 2001, pp. 292–299.
- [58] S. C. Johnson, “Hierarchical clustering schemes,” *Psychometrika*, vol. 32, no. 3, pp. 241–254, 1967.

- [59] H. Wang, C. Wang, C.-L. Chen, and L. Xie, “F-loam: Fast lidar odometry and mapping,” in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 4390–4396.
- [60] Q. Tong and C. Shaozu. A-LOAM: Advanced implementation of loam. (2019). [Online]. Available: <https://github.com/HKUST-Aerial-Robotics/A-LOAM>
- [61] Z. Gojcic, C. Zhou, J. D. Wegner, L. J. Guibas, and T. Birdal, “Learning multiview 3d point cloud registration,” in *Proc. of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 1759–1769.
- [62] M. Khoury, Q.-Y. Zhou, and V. Koltun, “Learning compact geometric features,” in *Proc. of the IEEE international conference on computer vision*, 2017, pp. 153–161.
- [63] Y. Liu, J. Shan, and H. Schofield, “Improvements to thin-sheet 3d lidar fiducial tag localization,” *IEEE Access*, vol. 12, pp. 124 907–124 914, 2024.
- [64] Y. Liu, J. Shan, A. Haridevan, and S. Zhang, “L-pr: Exploiting lidar fiducial marker for unordered low overlap multiview point cloud registration,” *IEEE Transactions on Instrumentation and Measurement*, 2025.
- [65] T. D. Barfoot, *State estimation for robotics*. Cambridge University Press, 2017.
- [66] B. Pfrommer and K. Daniilidis, “Tagslam: Robust slam with fiducial markers,” *arXiv preprint arXiv:1910.00679*, 2019.
- [67] G. P. Meyer, A. Laddha, E. Kee, C. Vallespi-Gonzalez, and C. K. Wellington, “Lasernet: An efficient probabilistic 3d object detector for autonomous driving,” in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 677–12 686.

- [68] P. Kovesi, “Good colour maps: How to design them,” *arXiv preprint arXiv:1509.03700*, 2015.
- [69] R. B. Rusu and S. Cousins, “3d is here: Point cloud library (pcl),” in *Proc. IEEE International Conference on Robotics and Automation*, 2011, pp. 1–4.
- [70] R. A. Haddad, A. N. Akansu *et al.*, “A class of fast gaussian binomial filters for speech and image processing,” *IEEE Transactions on Signal Processing*, vol. 39, no. 3, pp. 723–727, 1991.
- [71] J. Serafin and G. Grisetti, “Nlcp: Dense normal based point cloud registration,” in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015, pp. 742–749.
- [72] K. S. Arun, T. S. Huang, and S. D. Blostein, “Least-squares fitting of two 3-d point sets,” *IEEE Transactions on pattern analysis and machine intelligence*, no. 5, pp. 698–700, 1987.
- [73] K. Koide, S. Oishi, M. Yokozuka, and A. Banno, “Scalable fiducial tag localization on a 3d prior map via graph-theoretic global tag-map registration,” in *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2022, pp. 5347–5353.
- [74] 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.
- [75] R. B. Rusu, “Semantic 3d object maps for everyday manipulation in human living environments,” *KI-Künstliche Intelligenz*, vol. 24, no. 4, pp. 345–348, 2010.
- [76] P. Schneider and D. H. Eberly, *Geometric tools for computer graphics*. Elsevier, 2002.

- [77] D. Girardeau-Montaut *et al.*, “Cloudcompare,” *France: EDF R&D Telecom ParisTech*, vol. 11, no. 5, 2016.
- [78] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert, “isam2: Incremental smoothing and mapping with fluid relinearization and incremental variable reordering,” in *Proc. of IEEE International Conference on Robotics and Automation*, 2011, pp. 3281–3288.
- [79] F. Dellaert and M. Kaess, *Factor Graphs for Robot Perception*. Foundations and Trends in Robotics, Vol. 6, 2017. [Online]. Available: <http://www.cs.cmu.edu/~kaess/pub/Dellaert17fnt.pdf>
- [80] Supervisely. Supervisely Computer Vision platform. (2023). [Online]. Available: <https://supervisely.com>
- [81] Y. Liu, J. Shan, and H. Schofield, “Fiducial tag localization on a 3d lidar prior map,” 2024. [Online]. Available: <https://arxiv.org/abs/2209.01072>
- [82] S. Sato, Y. Yao, T. Yoshida, T. Kaneko, S. Ando, and J. Shimamura, “Unsupervised intrinsic image decomposition with lidar intensity,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 13 466–13 475.
- [83] S. Huang, Z. Gojcic, Z. Wang, F. Williams, Y. Kasten, S. Fidler, K. Schindler, and O. Litany, “Neural lidar fields for novel view synthesis,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 18 236–18 246.
- [84] Z. Zheng, F. Lu, W. Xue, G. Chen, and C. Jiang, “Lidar4d: Dynamic neural fields for novel space-time view lidar synthesis,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 5145–5154.

List of Publications

1. **Y. Liu**, H. Schofield and J. Shan, "Navigation of a Self-Driving Vehicle Using One Fiducial Marker," IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (**MFI**), Karlsruhe, Germany, 2021, pp. 1-6, DOI: 10.1109/MFI52462.2021.9591194.
2. **Y. Liu**, H. Schofield and J. Shan, "Intensity Image-Based LiDAR Fiducial Marker System," in IEEE Robotics and Automation Letters (**RA-L**), vol. 7, no. 3, pp. 6542-6549, 2022, DOI: 10.1109/LRA.2022.3174971.
3. **Y. Liu**, A. Haridevan, H. Schofield and J. Shan, "Application of Ghost-DeblurGAN to Fiducial Marker Detection," IEEE/RSJ International Conference on Intelligent Robots and Systems (**IROS**), Kyoto, Japan, 2022, pp. 6827-6832, DOI: 10.1109/IROS47612.2022.9981701.
4. **Y. Liu**, K. Zhu, G. Wu, Y. Ren, B. Liu, Y. Liu, and J. Shan, "MV-DeepSDF: Implicit Modeling with Multi-Sweep Point Clouds for 3D Vehicle Reconstruction in Autonomous Driving," IEEE/CVF International Conference on Computer Vision (**ICCV**), Paris, France, 2023, pp. 8272-8282, DOI: 10.1109/ICCV51070.2023.00763.
5. **Y. Liu***, Z. Yang*, G. Wu, Y. Ren, K. Lin, B. Liu, Y. Liu, and J. Shan, "VQA-Diff:

- Exploiting VQA and Diffusion for Zero-Shot Image-to-3D Vehicle Asset Generation in Autonomous Driving," European Conference on Computer Vision (**ECCV**), Milan, Italy, 2024. pp. 323-340, DOI: https://doi.org/10.1007/978-3-031-72848-8_19.
6. **Y. Liu**, J. Shan, and H. Schofield, "Improvements to Thin-Sheet 3D LiDAR Fiducial Tag Localization," **IEEE ACCESS**, vol. 12, pp. 124907-124914, 2024, DOI: 10.1109/ACCESS.2024.3451404.
 7. **Y. Liu**, J. Shan, A. Haridevan, and S. Zhang, "L-PR: Exploiting LiDAR Fiducial Marker for Unordered Low Overlap Multiview Point Cloud Registration," **IEEE Transactions on Instrumentation & Measurement (TIM)**, in press, 2025
 8. S. Zhang, J. Shan, and **Y. Liu**, "Approximate Inference Particle Filtering for Mobile Robot SLAM," in **IEEE Transactions on Automation Science and Engineering (TASE)**, pp. 1-12, 2024, DOI: 10.1109/TASE.2024.3475735.
 9. Z. Yang*, **Y. Liu***, G. Wu, T. Cao, Y. Ren, Y. Liu, and B. Liu, "Learning Effective NeRFs and SDFs Representations with 3D GANs for Object Generation," **NeurIPS Workshop**, 2024 (* co-first author).
 10. S. Zhang, J. Shan, and **Y. Liu**, "Variational Bayesian Estimator for Mobile Robot Localization With Unknown Noise Covariance," in **IEEE/ASME Transactions on Mechatronics (T-MECH)**, vol. 27, no. 4, pp. 2185-2193, 2022, DOI: 10.1109/TMECH.2022.3161591.
 11. S. Zhang, J. Shan, and **Y. Liu**, "Particle Filtering on Lie Group for Mobile Robot Localization With Range-Bearing Measurements," in **IEEE Control Systems Letters (L-CSS)**, vol. 7, pp. 3753-3758, 2023, DOI: 10.1109/LCSYS.2023.3340419.

12. T. Liao, A. Haridevan, **Y. Liu**, and J. Shan, “Autonomous Vision-Based UAV Landing with Collision Avoidance Using Deep Learning,” Science and Information Conference (**SAI**), pp. 79–87, 2022, DOI: 10.1007/978-3-031-10464-0_6