

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

**SINGLE FRAME TARGETLESS LIDAR-CAMERA CALIBRATION USING
REGION OF INTEREST BASED EDGE FEATURES**



M.Sc. THESIS

Mahsun BİNGÖL

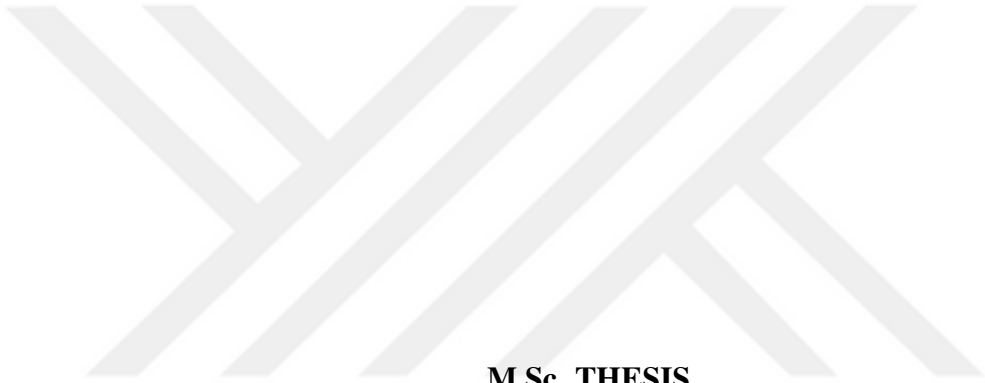
Department of Mechatronics Engineering

Mechatronics Engineering Programme

JANUARY 2026

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL

**SINGLE FRAME TARGETLESS LIDAR-CAMERA CALIBRATION USING
REGION OF INTEREST BASED EDGE FEATURES**



M.Sc. THESIS

**Mahsun BİNGÖL
(518221009)**

Department of Mechatronics Engineering

Mechatronics Engineering Programme

Thesis Advisor: Prof. Dr. Volkan SEZER

JANUARY 2026

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ LİSANSÜSTÜ EĞİTİM ENSTİTÜSÜ

**BÖLGE İLGİ TABANLI KENAR ÖZELLİKLERİ KULLANILARAK
TEK KARELİ HEDEFSİZ LİDAR-KAMERA KALİBRASYONU**

YÜKSEK LİSANS TEZİ

**Mahsun BİNGÖL
(518221009)**

Mekatronik Mühendisliği Ana Bilim Dalı

Mekatronik Mühendisliği Programı

Tez Danışmanı: Prof. Dr. Volkan SEZER

OCAK 2026

Mahsun BİNGÖL, a M.Sc. student of ITU Graduate School, student ID 518221009, successfully defended the thesis entitled “SINGLE FRAME TARGETLESS LIDAR-CAMERA CALIBRATION USING REGION OF INTEREST BASED EDGE FEATURES”, which he prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor : **Prof. Dr. Volkan SEZER**
Istanbul Technical University

Jury Members : **Doç. Dr. Osman Taha ŞEN**
Istanbul Technical University

Dr. Öğr. Üyesi Sinan Öncü
Boğaziçi University

Date of Submission : **02 January 2026**

Date of Defense : **16 January 2026**





*From there we came out to see the stars again.
— Dante Alighieri, Inferno*



FOREWORD

I would like to express my deepest appreciation to my thesis advisor, Prof. Dr. Volkan SEZER, for his guidance, encouragement, and continuous support throughout this study. His expertise, constructive feedback, and academic discipline greatly influenced both the technical quality of this thesis and my approach to scientific research.

I am profoundly grateful to my parents and my sister for their unconditional love, understanding, and support. Their constant encouragement and belief in me have been a source of motivation throughout my academic journey.

Finally, I would like to thank my fiancée, Şevval ÇETİNKAYA, who has been by my side throughout this entire process. Her patience, understanding, encouragement, help, and emotional support helped me overcome many challenging moments. This work is as much a result of her support as it is of my own efforts.

January 2026

Mahsun BİNGÖL
(Mechatronics Engineer)

TABLE OF CONTENTS

	Page
FOREWORD	ix
TABLE OF CONTENTS	xi
ABBREVIATIONS	xiii
SYMBOLS	xv
LIST OF TABLES	xvii
LIST OF FIGURES	xxi
SUMMARY	xxiii
ÖZET	xxv
1. INTRODUCTION	1
1.1 Background	1
1.2 Research Question	3
1.3 Research Objective	4
1.4 Outline of the Thesis	4
2. LITERATURE STUDY	7
2.1 LiDAR-Camera Calibration and Projection Model	7
2.1.1 Extrinsic transformation	7
2.1.2 Intrinsic camera projection	8
2.1.3 Overall projection model	9
2.2 Related Works	10
3. METHODS AND IMPLEMENTATION	21
3.1 LiDAR Point Cloud Edge Extraction Processing	24
3.2 Image Processing	26
3.3 Region of Interest (ROI) Processing	27
3.3.1 Manually selecting the ROI	27
3.3.2 Automatically selecting the ROI	28
3.4 Cost Function	30
3.5 Miss-calibration	31
3.6 Optimization	37
3.6.1 Greedy hill-climb grid search	38
3.6.2 Particle swarm optimization (PSO)	38
4. RESULT AND DISCUSSION	43
4.1 Dataset	43
4.2 Tools and Libraries	43
4.3 Experimental Results	45
4.3.1 Time performance analysis	46
4.3.2 Calibration methods comparison	50
5. CONCLUSIONS AND FUTURE WORKS	69
REFERENCES	73
CURRICULUM VITAE	77



ABBREVIATIONS

SAE	: Society of Automotive Engineers
ADS	: Automated Driving System
DDT	: Dynamic Driving Task
ODD	: Operational Design Domain
GPS	: Global Positioning System
IMU	: Inertial Measurement Unit
RADAR	: Radio Detection and Ranging
LiDAR	: Light Detection and Ranging
PID	: Proportional-Integral-Derivative
MPC	: Model Predictive Control
ROI	: Region of Interest
KITTI	: Karlsruhe Institute of Technology and Toyota Technological Institute
SGM	: Semi-Global Matching
ICP	: Iterative Closest Point
BA	: Bearing Angle
PnP	: Perspective- n -Point
RANSAC	: Random Sample Consensus
SE(3)	: Special Euclidean Group in three dimensions
kd-tree	: k-dimensional tree
RGB	: Red Green Blue
Canny	: Canny edge detection algorithm
OpenCV	: Open Source Computer Vision Library
YOLOv7-X	: You Only Look Once, version 7 (extended model)
BDD100K	: Berkeley DeepDrive (100K) dataset
ROS	: Robot Operating System
IoU	: Intersection over Union
PSO	: Particle Swarm Optimization
TVIW	: Time-Varying Inertia Weight
TVAC	: Time-Varying Acceleration Coefficients
RNG	: Random Number Generator
STL	: Standard Template Library
PCL	: Point Cloud Library
NDT	: Normal Distributions Transform
PCD	: Point Cloud Data file format
PLY	: Polygon File Format
OpenMP	: Open Multi-Processing
Hz	: Hertz (sampling frequency)
LTS	: Long Term Support
GCC	: GNU Compiler Collection
SO(3)	: Special Orthogonal Group in 3D (rotation group)



SYMBOLS

$\mathbf{R}$	: Rotation matrix
$\mathbf{t}$	: Translation vector
$\mathbf{T}$	: Extrinsic transformation matrix
$\mathbf{X}_L$	: Homogeneous 3D point in the LiDAR frame, $\mathbf{X}_L=[x_L, y_L, z_L, 1]^T$
$\mathbf{X}_C$	: 3D point in the camera frame, $\mathbf{X}_C=[x_C, y_C, z_C]^T$
$\mathbf{T}_{CL}$	: Extrinsic rigid-body transformation from LiDAR to camera frame
$\mathbf{R}_{CL}$	: Rotation matrix of the extrinsic transformation, $\mathbf{R}_{CL} \in \text{SO}(3)$
$\mathbf{t}_{CL}$	: Translation vector of the extrinsic transformation, $\mathbf{t}_{CL} \in \mathbb{R}^3$
$\mathbf{K}$	: Camera intrinsic matrix
f_x	: Focal length in x-direction (pixel units)
f_y	: Focal length in y-direction (pixel units)
c_x	: Principal point x-coordinate
c_y	: Principal point y-coordinate
u	: Image pixel coordinate along horizontal axis
v	: Image pixel coordinate along vertical axis
f_u	: Focal length along u axis
f_v	: Focal length along v axis
u_0	: Principal point along u axis
v_0	: Principal point along v axis
t_x	: Translation component along x-axis
t_y	: Translation component along y-axis
t_z	: Translation component along z-axis
x, y, z	: 3D point coordinates in the LiDAR frame
x_L, y_L, z_L	: LiDAR-frame coordinates (as in $\mathbf{X}_L$)
x_C, y_C, z_C	: Camera-frame coordinates (as in $\mathbf{X}_C$)
$\mathbf{X}_p^i$	: LiDAR edge response at point index p on scan line i
$\mathbf{P}_p^i$	: LiDAR point at index p on scan line i
$\mathbf{P}_{p-1}^i$	: Previous neighboring LiDAR point
$\mathbf{P}_{p+1}^i$	: Next neighboring LiDAR point
p	: Point index along a scan line
i	: Scan line (laser beam) index
r	: LiDAR range measurement (meters)
(u_p, v_p)	: Projected pixel coordinates of LiDAR point p
$D(u_p, v_p)$	: Bilinearly interpolated edginess value
N	: Number of valid projected LiDAR points in the ROI
J_c	: Single-ROI calibration cost
J_c^k	: Cost value for the k-th ROI
J_{avg}	: Average calibration cost over valid ROIs
N_{ROI}	: Number of automatically selected ROIs
N_{valid}	: Number of valid ROIs

F_C	: Fraction of perturbed cases with worse cost
T_{center}	: Center (reference) extrinsic transformation
T_{nbr}	: Perturbed neighboring transformation
D	: Gaussian probability density function value
K_{max}	: Maximum iteration count
$N(T)$	: Discrete neighborhood of transform T
T_{best}	: Best candidate transform
$J_c(T)$	: Cost function evaluated at T
w	: PSO inertia weight
w_{max}	: Maximum inertia weight
w_{min}	: Minimum inertia weight
c_1	: Cognitive acceleration coefficient
c_2	: Social acceleration coefficient
r_1, r_2	: Random scalars in $[0, 1]$
v_i	: Velocity vector of particle i
x_i	: Position vector of particle i
$p_{\text{best}, i}$	: Personal best position of particle i
$p_{\text{best}, g}$	: Global best position
N_p	: Number of particles
K	: Number of PSO iterations
t	: Iteration index
T	: Maximum iteration index
$v_{\text{max}, \text{trans}}$	: Maximum translation velocity
$v_{\text{max}, \text{rot}}$	: Maximum rotation velocity
k_{trans}	: Translation velocity scaling factor
k_{rot}	: Rotation velocity scaling factor
N_{threads}	: Number of parallel threads
R_{est}	: Estimated rotation matrix
R_{gt}	: Ground-truth rotation matrix
t_{est}	: Estimated translation vector
t_{gt}	: Ground-truth translation vector
$\ \cdot\ _2$	: Euclidean (L2) norm
X, Y, Z	: Cartesian translation axes
p_i	: Perturbation magnitude at level i
p_i^{rot}	: Rotational perturbation (degrees)
p_i^{trans}	: Translational perturbation (meters)
m	: Calibration method index
M	: Set of calibration methods
$T_m(p_i^{\text{trans}})$	: Translation error of method m
$R_m(p_i^{\text{rot}})$	: Rotation error of method m
R	: Set of all error ratios
S_m	: Calibration Success Score
m^*	: Best-performing method
s	: Seconds

LIST OF TABLES

	<u>Page</u>
Table 2.1: Comparison of LiDAR–camera calibration methods in terms of advantages and disadvantages.	11
Table 3.1: Perturbed translation parameters in meters for correct and incorrect calibration cases.	32
Table 3.2: Gaussian distribution parameters (μ and σ) for correct and incorrect calibration cases: interval in μ indicates 90% confidence interval.	37
Table 4.1: KITTI raw dataset statistics.	44
Table 4.2: Development environment specifications.	45
Table 4.3: Calibration Success Score for X-Axis Translation Error.	61
Table 4.4: Calibration Success Score for Y-Axis Translation Error.	61
Table 4.5: Calibration Success Score for Z-Axis Translation Error.	61
Table 4.6: Calibration Success Score for Roll-Axis Rotation Error.	62
Table 4.7: Calibration Success Score for Pitch-Axis Rotation Error.	62
Table 4.8: Calibration Success Score for Yaw-Axis Rotation Error.	63
Table 4.9: Translation calibration errors under different translation perturbation magnitudes using PSO-based methods.	64
Table 4.10: Translation calibration errors under different translation perturbation magnitudes using grid search optimization.	64
Table 4.11: Translation calibration errors using the Levinson et al. baseline method.	65
Table 4.12: Calibration errors under roll, pitch, and yaw perturbations using PSO-based optimization.	65
Table 4.13: Calibration errors under roll, pitch, and yaw perturbations using grid search optimization.	66
Table 4.14: Calibration errors under roll, pitch, and yaw perturbations using the Levinson et al. baseline method.	67



LIST OF FIGURES

	<u>Page</u>
Figure 1.1: Autonomous Vehicle System Architecture of A1 vehicle [2].....	3
Figure 2.1: Projection from 3D Lidar point into 2D image pixel.....	8
Figure 2.2: LiDAR - Camera Calibration Classification.	10
Figure 2.3: Geometrical constraints [7].....	13
Figure 2.4: Calibration example from [8].	14
Figure 2.5: Extraction pipeline for four central points from LiDAR (top) / stereo (bottom) [10].	14
Figure 2.6: Tracking calibration errors in real time. Initial calibration (Top), perturbed (Middle), recalibrated (Bottom) [12].	16
Figure 2.7: Depth-discontinuous edge and depth-continuous edge [14].	18
Figure 2.8: Depth-continuous edge extraction [14].	18
Figure 2.9: Comparison between depth-discontinuous edges and depth-continuous edges [14].	19
Figure 3.1: Block diagram of the proposed method.....	23
Figure 3.2: (a) Raw LiDAR point cloud (b) Projected LiDAR point cloud - Filtered.	24
Figure 3.3: Projected LiDAR point cloud from camera perspective.	24
Figure 3.4: Projected LiDAR point cloud from camera perspective (zoomed in).	25
Figure 3.5: Edged LiDAR point cloud from camera perspective (zoomed in)....	26
Figure 3.6: Edged LiDAR point cloud on image.....	26
Figure 3.7: (a) RGB image (b) Edginess map with $\beta = 10$	27
Figure 3.8: Manually selected ROI on the image.	28
Figure 3.9: (a) RGB image within the ROI (b) Edginess map within the ROI....	28
Figure 3.10: YOLOv7-X object detections on the image.	29
Figure 3.11: Percentage of the worse cases (FC) of correct and incorrect calibration parameters with small perturbation for multiple ROI selection in single frame.	33
Figure 3.12: Percentage of the worse cases (FC) of correct and incorrect calibration parameters with small perturbation for multiple ROI selection vs no ROI selection in single frame.	33
Figure 3.13: Calibration visualization between correct and miss-calibration scenarios.....	34
Figure 3.14: Histogram of FC values with Gaussian fit for correct calibration. ...	35
Figure 3.15: Histogram of FC values with Gaussian fit for incorrect calibration 1.	35
Figure 3.16: Histogram of FC values with Gaussian fit for incorrect calibration 2.	35
Figure 3.17: Histogram of FC values with Gaussian fit for incorrect calibration 3	36

Figure 3.18: Histogram of FC values with Gaussian fit for incorrect calibration 4.	36
Figure 3.19: Histogram of FC values with Gaussian fit for incorrect calibration 5.	36
Figure 4.1: Cumulative computation time comparison of LiDAR–camera calibration with automatic ROI selection and without ROI selection under increasing translation perturbations along the X-axis (3 cm, 5 cm, and 10 cm).	46
Figure 4.2: Cumulative computation time comparison of LiDAR–camera calibration with automatic ROI selection and without ROI selection under increasing translation perturbations along the Y-axis (3 cm, 5 cm, and 10 cm).	47
Figure 4.3: Cumulative computation time comparison of LiDAR–camera calibration with automatic ROI selection and without ROI selection under increasing translation perturbations along the Z-axis (3 cm, 5 cm, and 10 cm).	48
Figure 4.4: Average iteration time and corresponding speedup factor comparison for LiDAR–camera calibration under translation perturbations along the X-axis (3 cm, 5 cm, and 10 cm), with and without automatic ROI selection.	48
Figure 4.5: Average iteration time and corresponding speedup factor comparison for LiDAR–camera calibration under translation perturbations along the Y-axis (3 cm, 5 cm, and 10 cm), with and without automatic ROI selection.	49
Figure 4.6: Average iteration time and corresponding speedup factor comparison for LiDAR–camera calibration under translation perturbations along the Z-axis (3 cm, 5 cm, and 10 cm), with and without automatic ROI selection.	49
Figure 4.7: Overall comparison of average iteration time and corresponding speedup factors for LiDAR–camera calibration across all translation axes (X, Y, and Z), with and without automatic ROI selection.	50
Figure 4.8: Comparison of translation and rotation total errors with respect to the ground truth after applying X-axis perturbations of 0.03m, 0.05m, and 0.1m followed by extrinsic recalibration using different methods.	51
Figure 4.9: Comparison of translation and rotation total errors with respect to the ground truth after applying Y-axis perturbations of 0.03m, 0.05m, and 0.1m followed by extrinsic recalibration using different methods.	52
Figure 4.10: Comparison of translation and rotation total errors with respect to the ground truth after applying Z-axis perturbations of 0.03m, 0.05m, and 0.1m followed by extrinsic recalibration using different methods.	53

Figure 4.11: Comparison of translation and rotation total errors with respect to the ground truth after applying Roll-axis perturbations of 0.03° , 0.1° , 0.3° , and 1° followed by extrinsic recalibration using different methods.	55
Figure 4.12: Comparison of translation and rotation total errors with respect to the ground truth after applying Pitch-axis perturbations of 0.03° , 0.1° , 0.3° , and 1° followed by extrinsic recalibration using different methods.	56
Figure 4.13: Comparison of translation and rotation total errors with respect to the ground truth after applying Yaw-axis perturbations of 0.03° , 0.1° , 0.3° , and 1° followed by extrinsic recalibration using different methods.	57





SINGLE FRAME TARGETLESS LIDAR-CAMERA CALIBRATION USING REGION OF INTEREST BASED EDGE FEATURES

SUMMARY

With the increasing adoption of autonomous systems in both autonomous driving and robotics, accurate perception and sensor integration have become critical for safe and reliable operation. Among multi-sensor configurations, the extrinsic calibration between LiDAR and camera sensors plays a fundamental role in sensor fusion, object detection, and environment perception. This thesis addresses the LiDAR-camera extrinsic calibration problem and proposes a targetless, single-frame calibration framework based on region-of-interest (ROI) selection, designed to operate without calibration targets and to support online calibration scenarios.

The proposed framework builds upon edge-based calibration principles commonly used in the literature, but extends them through systematic integration of ROI-based data selection and multiple optimization strategies. Both manual and automatic ROI selection schemes are investigated. In the manual approach, a single rectangular ROI is defined by the user, whereas in the automatic approach, multiple ROIs are generated using YOLOv7-X-based object detection. Only ROIs containing reliable projected LiDAR points are retained, and in the multi-ROI case, the overall cost is computed as the average of ROI-level costs.

LiDAR point clouds and camera images are processed in a time-synchronized manner. LiDAR edge points are extracted based on depth discontinuities along scan lines, while camera images are processed through a multi-stage pipeline to produce a continuous edginess map that assigns higher values to pixels closer to image edges. A scalar cost function is defined by measuring the alignment between projected LiDAR edge points and image edginess values within the selected ROIs, where each projection contributes proportionally to the product of LiDAR depth discontinuity and image edginess.

Two optimization strategies are employed to refine the extrinsic parameters: a greedy grid-search method based on discrete neighborhood evaluation, and a deterministic Particle Swarm Optimization (PSO) algorithm with time-varying inertia and acceleration coefficients. To ensure reproducibility and efficiency, deterministic seeding and multi-threaded particle evaluation are used. Additionally, a calibration validity assessment mechanism is introduced to detect incorrect calibrations by analyzing cost distributions under small perturbations around the estimated solution.

Experiments conducted on the KITTI dataset demonstrate that ROI-based processing significantly reduces computational cost while improving calibration robustness. Automatic ROI selection reduces the average per-iteration computation time by approximately %25-%32 across all translation axes, achieving speedup factors up

to 1.70 $\times$. In terms of accuracy, ROI-based methods consistently outperform the baseline approach, achieving sub-centimeter translation errors and sub-degree rotation errors under small and moderate perturbations. Automatic ROI combined with PSO is particularly effective in stabilizing rotational estimates under translation and angular perturbations by exploiting geometric constraints from multiple scene objects.

Overall, the results confirm that the proposed ROI-supported, optimization-based, single-frame LiDAR–camera extrinsic calibration framework provides a computationally efficient, accurate, and stable solution suitable for both offline calibration and online calibration update scenarios in autonomous driving systems.



BÖLGE İLGİ TABANLI KENAR ÖZELLİKLERİ KULLANILARAK TEK KARELİ HEDEFSİZ LİDAR-KAMERA KALİBRASYONU

ÖZET

Günümüzde otonom sistemlerin yaygınlaşması ile birlikte hem otonom araçlarda hem de robotikte sensörlerin önemi artmıştır. Farklı amaçlara hizmet eden çeşitli alanlarla kullanılan birçok sensör bulunmaktadır. Bu sensörler, çevresel verileri toplamak, nesneleri algılamak ve sistemlerin çevreleriyle etkileşim kurmasını sağlamak için kullanılır. Örneğin, LIDAR sensörleri, otonom araçların çevresini haritalamak ve engelleri tespit etmek için yaygın olarak kullanılırken, ultrasonik sensörler robotik uygulamalarda yakınlık algılama için tercih edilmektedir. Ayrıca, kamera tabanlı sensörler görsel veri toplamak ve nesne tanıma görevlerinde önemli bir rol oynamaktadır. Sensör teknolojilerindeki ilerlemeler, otonom sistemlerin daha güvenli, verimli ve etkili bir şekilde çalışmasını mümkün kılmaktadır. Bu nedenle, sensörlerin doğru seçimi ve entegrasyonu, otonom sistemlerin başarısı için kritik öneme sahiptir.

Bu tezde, otonom sürüş ve algılayıcı füzyonu uygulamalarında kritik öneme sahip olan LiDAR-kamera dışsal (extrinsic) kalibrasyon problemi ele alınmış ve hedef (kalibrasyon tahtası vb.) gerektirmeyen, tek kare (single-frame) üzerinde çalışabilen ROI (Region of Interest) tabanlı bir kalibrasyon çerçevesi önerilmiştir. Literatürde yaygın olarak kullanılan kenar temelli kalibrasyon yaklaşımı esas alınmış; ancak veri kullanım biçimi, görüntü işleme akışı ve optimizasyon stratejileri bakımından yöntem genişletilerek manuel ROI ve otomatik ROI seçimleri ile farklı optimizasyon tekniklerinin etkileri sistematik olarak incelenmiştir. Çalışmada KITTI veri seti kullanılarak gerçek dışsal parametrelere farklı seviyelerde bozunumlar uygulanmış ve ROI seçiminin hem hesaplama verimliliği hem de kalibrasyon performansı üzerindeki etkisi nicel olarak değerlendirilmiştir.

Önerilen yöntemde, LiDAR nokta bulutu ile kamera görüntüsü zaman uyumlu biçimde ele alınmış ve LiDAR noktaları mevcut dışsal parametreler ile görüntü düzlemine projekte edilerek yalnızca görüntüyle kesişen noktalar üzerinden işlem yapılmıştır. LiDAR tarafında kenar noktaları, tarama çizgisi (scan line) boyunca komşu noktalar arasındaki derinlik süreksizliklerinden yararlanılarak çıkarılmış; kamera tarafında ise RGB görüntüden çok aşamalı bir görüntü işleme zinciri ile kenar temsili elde edilmiş ve kenarlara yakın pikselleri yüksek değerle ifade eden sürekli bir “edginess” haritası oluşturulmuştur. Kalibrasyon maliyet fonksiyonu, ROI içindeki projekte LiDAR kenar noktalarının görüntüdeki edginess değerleri ile olan hizalanmasını ölçmek üzere tanımlanmış; her bir projeksiyonun katkısı LiDAR derinlik süreksizliği ile görüntü edginess değerinin çarpımı olarak modellenerek tek bir skaler maliyet değerine indirgenmiştir. ROI seçimi iki farklı şekilde ele alınmıştır: (i) kullanıcı tarafından manuel olarak belirlenen tekil dikdörtgen ROI ve (ii) YOLOv7-X tabanlı nesne algılama ile otomatik olarak belirlenen çoklu ROI. Otomatik ROI yaklaşımında,

projekte LiDAR noktası içermeyen veya güvenilir olmayan tespitler elenmiş; çoklu ROI durumunda maliyet değerleri ROI bazında hesaplanarak ortalama maliyet üzerinden optimizasyon yürütülmüştür.

Dışsal parametrelerin iyileştirilmesi için iki farklı optimizasyon stratejisi uygulanmıştır. Birinci yöntem, altı serbestlik derecesi üzerinde sabit adımlı ayırık komşuluk oluşturarak en iyi komşu çözümü seçen açgözlü (greedy) tepe tırmanma tabanlı grid-search yaklaşımıdır. İkinci yöntem ise, zamanla değişen atalet ağırlığı ve hızlanma katsayılarını kullanan deterministik Parçacık Sürü Optimizasyonu (PSO) algoritmasıdır. PSO'da tekrarlanabilirlik amacıyla deterministik tohumlama uygulanmış ve parçacık değerlendirmeleri çok iş parçacıklı yapı ile hızlandırılmıştır. Ayrıca, çevrimiçi kullanım senaryolarını desteklemek amacıyla yanlış kalibrasyon durumunu tespit etmeye yönelik bir mekanizma geliştirilmiş; mevcut kalibrasyon çevresinde küçük bozunumlarla üretilen komşu dönüşümlerin maliyet dağılımı analiz edilerek doğru ve yanlış kalibrasyonların ayrıştırılması hedeflenmiştir.

Deneysel sonuçlar, ROI tabanlı işlemenin hem hesaplama maliyetini anlamlı biçimde azalttığını hem de kalibrasyonun sayısal olarak daha kararlı sonuçlar ürettiğini açıkça göstermektedir. Otomatik ROI seçimi kullanıldığında, iterasyon başına ortalama hesaplama süresi X ekseninde yaklaşık 0.31, s seviyesinden 0.21, s seviyesine düşmüş ve $1.51 \times$ hızlanma sağlanmıştır. Benzer şekilde, Y ve Z eksenlerinde ortalama iterasyon süreleri sırasıyla 0.28, s'den 0.21, s'ye gerileyerek $1.32 \times$ ve $1.35 \times$ hızlanma elde edilmiştir. Pertürbasyon seviyeleri ayrı ayrı incelendiğinde, X ekseninde hızlanma faktörünün $1.37 \times$ ile $1.70 \times$ arasında değiştiği, Y ve Z eksenlerinde ise hızlanmanın daha stabil biçimde $1.29 \times$ - $1.37 \times$ aralığında kaldığı gözlenmiştir. Bu bulgular, ROI ile sınırlandırılmış maliyet hesabının, arama uzayını küçülterek per-iterasyon hesaplama yükünü eksen bağımsız biçimde azalttığını ortaya koymaktadır.

Kalibrasyon doğruluğu açısından bakıldığında, ROI tabanlı yöntemlerin özellikle küçük ve orta büyüklükteki pertürbasyonlar altında belirgin üstünlük sağladığı görülmektedir. X eksenini çeviri pertürbasyonlarında, 0.03, m seviyesinde en düşük çeviri toplam hatası otomatik ROI ve PSO birleşimi ile 0.0194, m olarak elde edilmiş; bu değer manuel ROI + PSO için 0.0204, m ve otomatik ROI + grid-search için 0.0220, m seviyesinde kalmıştır. Aynı koşulda referans yöntem olan Levinson yaklaşımı 0.0870, m hata üreterek, önerilen yönteme kıyasla yaklaşık 4.5 kat daha yüksek hata sergilemiştir. Daha büyük bir pertürbasyon olan 0.10, m seviyesinde dahi, ROI tabanlı yöntemler 0.010-0.017, m aralığında çeviri hataları üretirken, referans yöntemin hatası 0.1175, m seviyesine yükselmiştir. Bu sonuçlar, ROI kısıtının büyük bozunumlar altında dahi çeviri doğruluğunu korumaya yardımcı olduğunu göstermektedir.

Çeviri pertürbasyonları altında ölçülen dönme (rotation) hataları incelendiğinde, ROI seçiminin kullanılan optimizasyon yöntemiyle birlikte farklı davranış örüntüleri sergilediği görülmektedir. X eksenini çeviri pertürbasyonu (0.03 m) altında, en düşük toplam dönme hatası manuel ROI + PSO yöntemi ile 0.0357° olarak elde edilirken, otomatik ROI + PSO ve otomatik ROI + grid-search yöntemleri sırasıyla 0.0640° ve 0.0805° dönme hataları üretmiştir. Buna karşın Y eksenini çeviri pertürbasyonlarında, otomatik ROI + PSO yöntemi dönme doğruluğunda belirgin bir üstünlük göstermiş; 0.05 m seviyesinde 0.0157° ve 0.10 m seviyesinde 0.0079° gibi oldukça düşük toplam dönme hatalarına ulaşılmıştır. Bu sonuçlar, otomatik ROI yaklaşımının birden

fazla nesneden elde edilen geometrik kısıtları birlikte kullanarak, çeviri kaynaklı bozunumlar altında dahi dönme parametrelerini daha etkin biçimde sınırlandırabildiğini göstermektedir.

Z eksen ve dönel (roll, pitch, yaw) pertürbasyonları altında elde edilen sonuçlar, çoklu ROI kullanımının kalibrasyon kararlılığı üzerindeki etkisini daha net ortaya koymaktadır. Örneğin pitch pertürbasyonlarında 0.03° seviyesinde otomatik ROI + PSO yöntemi 0.012, m çeviri hatası üretirken, manuel ROI + PSO 0.051, m ve referans yöntem 0.042, m hata üretmiştir. Benzer şekilde roll pertürbasyonlarında 0.10° seviyesinde otomatik ROI + PSO 0.015, m çeviri hatası ile en iyi sonucu verirken, manuel grid-search yöntemi 0.046, m seviyesinde kalmıştır. Bu sayısal bulgular, otomatik ROI kullanımının özellikle küçük ve orta açısal bozunumlarda hem çeviri hem de dönme bileşenlerinde daha dengeli bir hata dağılımı sağladığını göstermektedir.

Yanlış kalibrasyon tespiti kapsamında yapılan değerlendirmelerde ise, doğru kalibrasyon durumlarında komşu bozunumlar altında elde edilen başarı oranının ortalama olarak 0.90 'ın üzerinde olduğu, yanlış kalibrasyon senaryolarında ise bu değer belirsiz biçimde düştüğü gözlenmiştir. Başarı oranı dağılımlarının istatistiksel olarak incelenmesi, doğru kalibrasyon için dar varyanslı ve yüksek ortalamalı bir yapı sergilerken, yanlış kalibrasyon durumlarında dağılımın hem ortalama hem de standart sapma açısından belirgin biçimde ayrıştığını ortaya koymuştur. Bu sonuçlar, geliştirilen değerlendirme yaklaşımının kalibrasyon doğruluğunu yalnızca görsel olarak değil, nicel ve istatistiksel olarak da güvenilir biçimde izleyebildiğini göstermektedir.

Genel olarak elde edilen sayısal sonuçlar, manuel veya otomatik ROI seçimi ile desteklenen ve grid-search ya da PSO tabanlı optimizasyon yöntemleri kullanan tek kareye dayalı LiDAR-kamera dışsal kalibrasyon çerçevesinin; hesaplama süresini ortalama olarak %25-%32 oranında azaltırken, çeviri hatalarını santimetre altı seviyelerde ve dönme hatalarını ondalık derece mertebesinde tutabildiğini göstermektedir. Bu yönüyle, önerilen ROI destekli ve optimizasyon temelli kalibrasyon çerçevesi, çevrimiçi ve çevrimdışı kalibrasyon güncellemesi gerektiren otonom sürüş sistemleri için hem hesaplama açısından verimli hem de doğruluk ve kararlılık açısından uygulanabilir bir çözüm sunmaktadır.



1. INTRODUCTION

In this chapter, the background and motivation for the research are presented in Section 1.1. The research questions and objectives are defined in Sections 1.2 and 1.3, respectively. Finally, an outline of the thesis structure is provided in Section 1.4.

1.1 Background

Today, autonomous driving is becoming a proven concept. We can already see an increasing number of vehicles developed by different brands using it on the roads in many countries. This thesis adopts the taxonomy defined in the SAE J3016 standard, titled “*Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*”, which provides a widely accepted framework for classifying driving automation systems and clarifying the roles of the human driver and the automated driving system (ADS). The standard introduces the concept of the dynamic driving task (DDT) and establishes consistent terminology to avoid ambiguity in the use of terms such as autonomous and automated [1].

SAE J3016 defines six levels of driving automation, ranging from Level 0 (no driving automation) to Level 5 (full driving automation). Levels 0–2 require continuous human supervision, even when certain vehicle control functions are automated, and are commonly associated with advanced driver assistance systems. In contrast, Levels 3–5 progressively shift responsibility for the entire driving task to the ADS within a defined operational design domain (ODD), with Level 5 representing full automation under all conditions. This taxonomy provides a standardized context for discussing perception, sensor fusion, and calibration requirements in automated driving systems [1].

The autonomous vehicle A1 was developed by Jo et al. as a full-scale research platform for the 2012 Korean Autonomous Vehicle Competition [2]. A1 integrates multiple heterogeneous sensors, distributed computing units, and real-time control systems to demonstrate a complete autonomous driving pipeline in urban and structured

environments. Rather than proposing a new theoretical framework, the authors use A1 as a practical case study to illustrate how standard autonomous driving functionalities can be systematically organized and implemented within a real vehicle.

In this context, Figure 1.1 presents the structure of the autonomous driving algorithms implemented on A1, which serves as a representative and widely accepted functional decomposition of an autonomous driving system. The architecture follows the classical separation into localization, perception, planning, vehicle control, and system management, reflecting the canonical perception–decision–control paradigm commonly adopted in autonomous driving research. Although the diagram is specific to the A1 platform, the depicted structure is not vehicle-specific; instead, it provides a generic reference architecture that captures the fundamental algorithmic flow of autonomous driving systems, from sensor data processing to motion execution and supervisory control.

Localization: The localization module estimates the vehicle’s precise position and orientation within its environment using data from GPS (Global Positioning System), IMU (Inertial Measurement Unit), wheel encoders, and other sensors. It employs techniques such as sensor fusion, Kalman filtering, and map matching to achieve accurate and robust localization, which is critical for safe navigation.

Perception: The perception module processes raw sensor data such as RADAR (Range and Detection And Ranging), LIDAR (Light Detection and Ranging), cameras, and ultrasonic sensors to detect and classify objects in the environment, including other vehicles, pedestrians, traffic signs, and lane markings. It employs techniques such as sensor fusion, computer vision, and machine learning to create a comprehensive understanding of the vehicle’s surroundings.

Planning: The planning module generates a safe and efficient trajectory for the vehicle to follow, considering the perceived environment, traffic rules, and the vehicle’s dynamic constraints. It includes path planning, behavior planning, and decision-making components to handle various driving scenarios.

Vehicle Control: The vehicle control module translates the planned trajectory into low-level control commands for the vehicle’s actuators, such as steering, throttle, and

brakes. It employs control algorithms such as PID (Proportional-Integral-Derivative) control, MPC (Model Predictive Control), and adaptive control to ensure smooth and accurate execution of the planned path.

System Management: The system management module oversees the overall operation of the autonomous driving system, handling tasks such as system monitoring, fault detection, and recovery. It ensures that all components function correctly and coordinates their interactions to maintain safe and reliable vehicle operation.

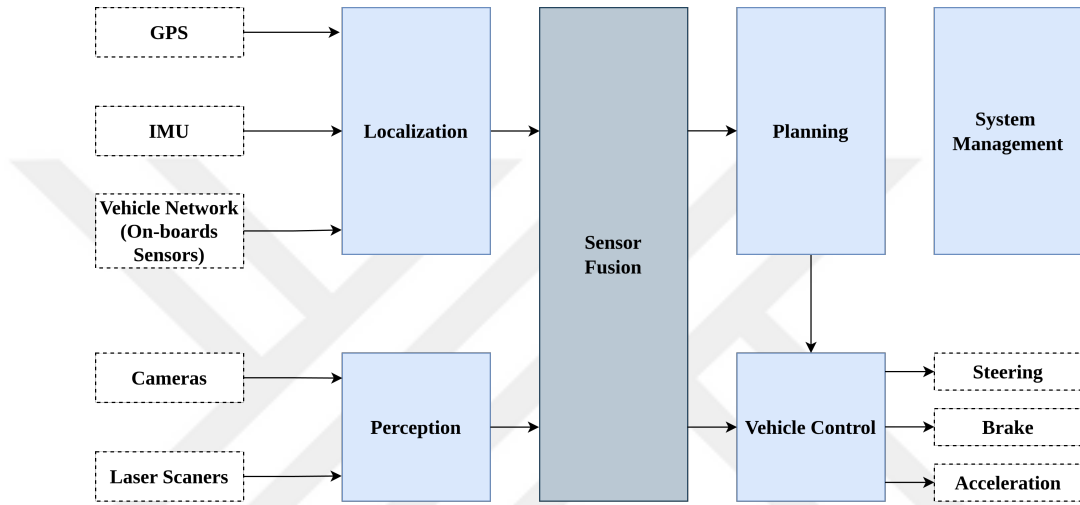


Figure 1.1 : Autonomous Vehicle System Architecture of A1 vehicle [2].

1.2 Research Question

LiDAR and cameras are two sensors frequently used in autonomous driving. Both sensors have advantages and disadvantages compared to each other, as well as different application areas. Therefore, it is necessary to fusion these two sensors. However, calibration of the two sensors is required before sensor fusion. Calibration performance is critically important because it also determines the sensor fusion performance. The research goal of this thesis is this calibration. The research questions can be stated as follows:

- How to calibrate LiDAR and camera without a target?
- How to calibrate LiDAR and camera online and offline?
- How does specifying a region of interest for calibration affect calibration performance?

- How accurate is the calibration algorithm?
- What is the performance compared with other different methods?

The principal focus of the thesis project is the extrinsic calibration of LiDAR and camera systems. Section 2.1 will present the mathematical model.

1.3 Research Objective

The objective of this thesis is to first determine the extrinsic transformation between LiDAR and camera coordinates through calibration. The expected scientific results are that the rotation and translation calibration parameters obtained after calibration should have a low error rate when compared to the ground truth values obtained from dataset.

The innovative aspect is the selection of region of interest for the calibration process. For offline calibration, the region of interest is selected by human interaction, while for online calibration, it is selected using an object detection algorithm that identifies edge-rich regions. Studies in the literature perform calibration either using a specific target, such as a chessboard, or using the entire image and projected LiDAR pointcloud in targetless methods. While calibration methods using targets provide high accuracy, they are quite limiting in practice. In addition, the proposed region of interest selection increases calibration speed. Furthermore, the aim is to expand the application area by proposing both online and offline calibration methods.

1.4 Outline of the Thesis

This thesis is organized into five main sections, each addressing a specific aspect of the proposed LiDAR–camera calibration framework.

Section 1 introduces the motivation and background of the study, defines the research questions and objectives, and provides an overview of the thesis structure.

Section 2 presents the theoretical foundations and related work. It first reviews the LiDAR–camera calibration problem, including extrinsic transformation, camera projection models, and the overall projection pipeline. Subsequently, existing calibration approaches in the literature are discussed to position the proposed method within the current state of research.

Section 3 details the proposed methods and their implementation. This section explains the LiDAR point cloud edge extraction process, image edge processing, and region of interest (ROI) selection strategies, including both manual and automatic approaches. The formulation of the cost function, the handling of miss-calibration, and the employed optimization techniques—namely greedy hill-climbing grid search and particle swarm optimization—are described in detail.

Section 4 presents the experimental setup and results. The dataset, software tools, and libraries used in the experiments are introduced, followed by a comprehensive performance analysis. Calibration accuracy, computational efficiency, and comparative evaluations of different calibration strategies are reported and discussed.

Finally, Section 5 concludes the thesis by summarizing the main contributions and findings of the study and outlines potential directions for future work.



2. LITERATURE STUDY

In this chapter, a comprehensive review of existing literature on LiDAR–camera extrinsic calibration methods is presented. Section 2.1 introduces the mathematical model underlying LiDAR–camera calibration and projection. Section 2.2 surveys various calibration techniques, categorizing them based on their methodologies, advantages, and limitations. The discussion highlights key contributions in both target-based and targetless calibration approaches, as well as offline and online methods. This literature review establishes the context for the proposed calibration framework detailed in subsequent chapters.

2.1 LiDAR-Camera Calibration and Projection Model

LiDAR–camera calibration aims to estimate the rigid-body transformation between the three-dimensional LiDAR coordinate frame and the two-dimensional camera image plane. As illustrated in Figure 2.1, this process enables a point measured in the LiDAR coordinate frame to be projected onto the camera image as a pixel location. Accurate calibration is essential for reliable sensor fusion, as even small errors in the transformation parameters may lead to significant misalignment between modalities.

Let $\mathbf{X}_L = [x_L, y_L, z_L, 1]^\top$ denote a point expressed in the LiDAR coordinate frame. To project this point onto the image plane, two successive transformations are applied: an extrinsic transformation between the LiDAR and camera coordinate frames, followed by an intrinsic camera projection.

2.1.1 Extrinsic transformation

The extrinsic calibration describes the relative pose between the LiDAR and camera coordinate frames and is represented by a rigid-body transformation matrix

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

where $\mathbf{R}_{CL} \in SO(3)$ is the rotation matrix and $\mathbf{t}_{CL} \in \mathbb{R}^3$ is the translation vector from the LiDAR frame to the camera frame. Applying this transformation yields the point coordinates in the camera frame,

$$\mathbf{X}_C = \mathbf{T}_{CL}\mathbf{X}_L, \quad (2.2)$$

where $\mathbf{X}_C = [x_C, y_C, z_C, 1]^\top$ represents the point in the camera coordinate frame.

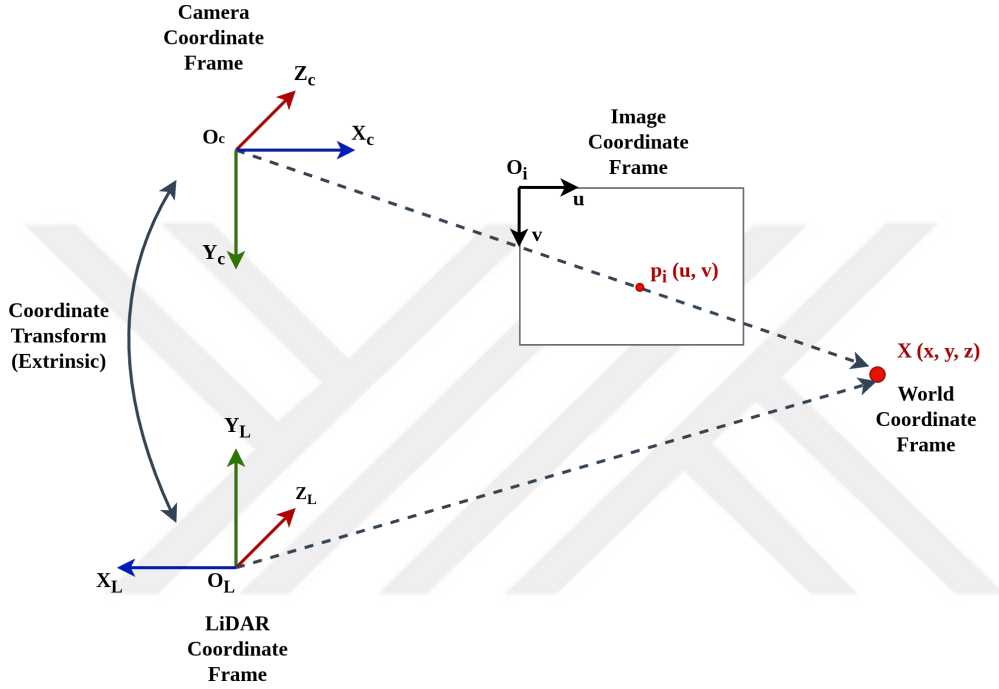


Figure 2.1 : Projection from 3D Lidar point into 2D image pixel.

As shown in Figure 2.1, this transformation aligns the LiDAR point cloud with the camera coordinate system, enabling geometric consistency between sensors.

2.1.2 Intrinsic camera projection

The intrinsic parameters characterize the internal geometry of the camera and map three-dimensional points in the camera frame onto the two-dimensional image plane.

The intrinsic matrix is defined as

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \quad (2.3)$$

where f_x and f_y denote the focal lengths in pixel units, and (c_x, c_y) represents the principal point.

Given a point $\mathbf{X}_C = [x_C, y_C, z_C]^\top$ in the camera coordinate frame, its projection onto the image plane is computed as

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \mathbf{K} \begin{bmatrix} x_C/z_C \\ y_C/z_C \\ 1 \end{bmatrix}, \quad (2.4)$$

where (u, v) denotes the image pixel coordinates and λ is a scaling factor corresponding to depth.

2.1.3 Overall projection model

Combining the extrinsic transformation between the LiDAR and camera coordinate frames with the intrinsic camera projection, the mapping from a three-dimensional LiDAR point to a two-dimensional image pixel can be expressed in a single matrix formulation. Let a point in the LiDAR coordinate frame be represented as $\mathbf{X}_L = [x, y, z, 1]^\top$. The corresponding pixel coordinates (u, v) on the image plane are obtained as

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_u & \gamma & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}, \quad (2.5)$$

where the intrinsic matrix contains the focal lengths f_u and f_v , the principal point coordinates (u_0, v_0) , and the skew parameter γ that is generally small and often assumed to be zero. The extrinsic matrix consists of the rotation matrix $\mathbf{R} = [r_{ij}] \in SO(3)$ and the translation vector $\mathbf{t} = [t_1, t_2, t_3]^\top$, which together define the rigid-body transformation from the LiDAR coordinate frame to the camera coordinate frame.

This compact formulation directly corresponds to the geometric relationship illustrated in Figure 2.1, where a three-dimensional LiDAR point is first transformed into the camera coordinate frame and subsequently projected onto the two-dimensional image plane through the camera intrinsics. The objective of LiDAR–camera calibration is to accurately estimate the extrinsic parameters $\mathbf{R}$ and $\mathbf{t}$, assuming the intrinsic parameters are known from prior camera calibration.

2.2 Related Works

The methodologies for LiDAR-camera extrinsic calibration systems can be classified into several categories: offline/online calibration, manual/automatic calibration, and target-based/targetless calibration. The objective of all approaches is to determine the extrinsic matrix, which contains rotation and translation matrices. The technique involves identifying projected 3D LiDAR points within the camera coordinate frame and their associated 2D picture pixels. Consequently, given a known camera intrinsic matrix, an ideal extrinsic matrix can be determined using an optimization technique.

The diagram in Figure 2.2 represents two primary classifications depending on the calibration method: target-based and targetless. The approach distinction referenced here addresses the topic of whether the LiDAR-camera calibration operation will use a target or be conducted without one. Upon resolving this matter, a determination must be made regarding whether the calibration procedure will be executed manually, involving human interaction, or automatically, devoid of any human interaction.

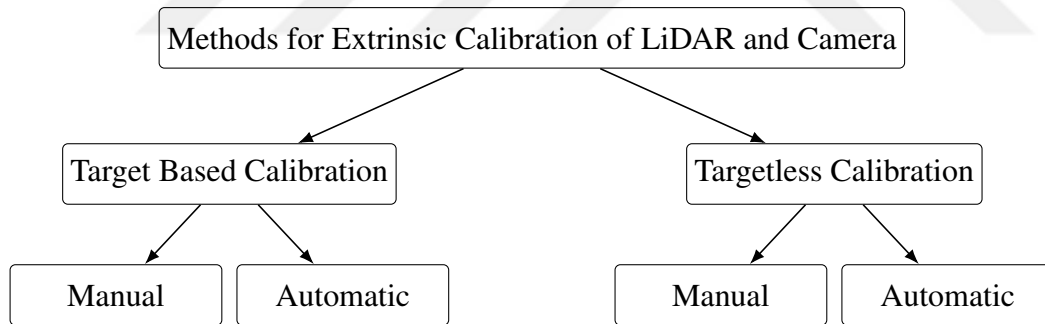


Figure 2.2 : LiDAR - Camera Calibration Classification.

LiDAR-camera calibration techniques possess both advantages and disadvantages in comparison to one another. The choice of approach is determined by the application domains, the platform for sensor placement, and the project specifications. The table 2.1 adapted from [3] delineates the advantages and disadvantages of calibration methods.

Table 2.1 : Comparison of LiDAR–camera calibration methods in terms of advantages and disadvantages.

Calibration method	Advantages	Disadvantages
Manual target-based	● Precise calibration results since the target geometry is known a priori. ● Straightforward correspondence registration, reducing outlier effects. ● Accurate data association across sensor modalities.	● Data acquisition is labor-intensive and time-consuming. ● Strongly dependent on operator expertise. ● Re-calibration is not feasible during motion or after mechanical disturbances. ● Restricted to static scenes with calibration targets.
Manual targetless	● Allows re-calibration during motion under mechanical vibrations and environmental changes. ● Uses natural environmental features instead of artificial targets. ● Suitable for online calibration.	● Highly dependent on operator skill. ● Performance degrades in feature-sparse or uninformative environments.
Automatic target-based	● Time-efficient due to automatic feature extraction and matching. ● High accuracy with known target geometry. ● Robust registration with fewer outliers.	● Still requires calibration targets and controlled data collection. ● Cannot adapt to calibration drift during motion. ● Limited flexibility in dynamic environments.
Automatic targetless	● Fully automatic and time-efficient. ● Enables online re-calibration under disturbances and environmental changes. ● Does not require artificial targets. ● Suitable for real-world autonomous driving scenarios.	● Performance degrades in scenes with insufficient geometric/visual features.

Furthermore, the literature differentiates between two separate kinds of calibration: online and offline. Targetless calibration methods are classified as online, whereas target-based approaches are known as offline calibration. Offline calibration methods produce more accurate results but are employed just when the robot is stationary. Conversely, online methods are executed during the robot's/vehicle's movement; nevertheless, their precision is considerably lower than that of offline methods. These online methods are generally intended to modify the extrinsic matrix, which varies over time due to mechanical or environmental factors affecting sensor positioning, following an initial calibration conducted using an offline method.

In an early study [4], introduced a two-stage approach for approximating the calibration parameters between a camera and a 2D laser scanner, using the plane constraints created by a chessboard presented in various orientations.

Calibration methods using targets have been refined over the years [5]–[7]. This approach, which entails performing studies with one or more targets, has gained increasing popularity and allows for the estimation of calibration parameters with substantial accuracy.

As a notable example of automatic calibration [7], presents a method that addresses the issue of extrinsic calibration between a camera and a 3D LiDAR using a checker board within a single frame. The extrinsic matrix is derived from 3D line-plane correspondences, utilising both the boundary lines extracted from the image and the plane of the board. The authors improve boundary sampling by positioning the target nearer to the LiDAR and demonstrate that parallel-sided, planar targets are subject to identical constraints. The geometrical constraints are illustrated in Figure 2.3. The method is validated using both simulated and real-world data, yielding accurate calibration results.

In [8], a method is presented for the calibration of a camera and a 3D range sensor (LiDAR/Kinect) that performs in a single shot and is entirely automated. The technique autonomously identifies many checkerboards located within the scene of the image, splits the range data into planar components, and computes the initial extrinsic matrix by leveraging the geometric correlations among these planes. The matrix is further

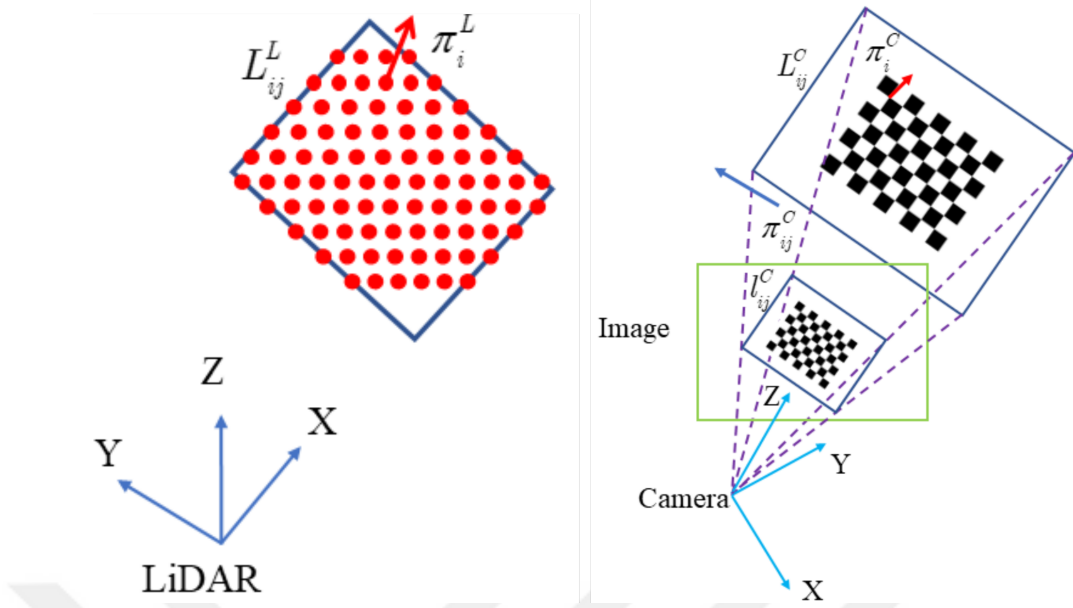
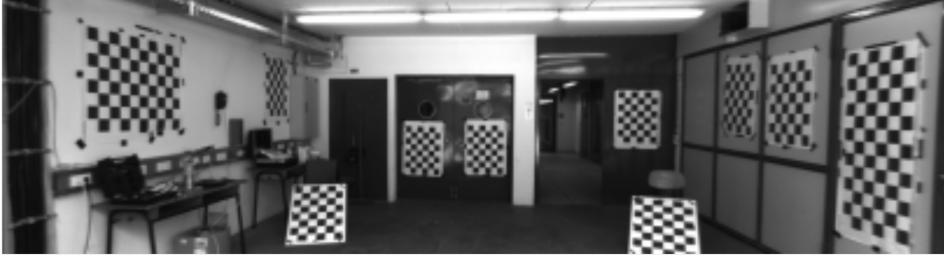


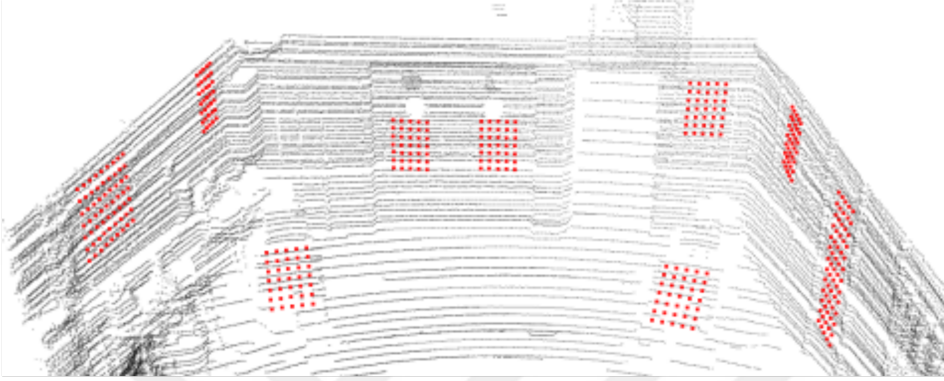
Figure 2.3 : Geometrical constraints [7].

improved using point-plane error. The primary advantages of the approach include its lack of necessity for user input, capability to yield results in a single attempt, reliability across various sensor types, rapid execution, and potential enhancement of both internal and external parameters based on the application. This technique serves as the sanctioned calibration method for the KITTI [9] dataset. An example of the input datas and calibration result with Velodyne HDL-64E is illustrated in Figure 2.4. In addition to checkerboards, there are techniques employing various patterns such as triangles, diamonds, and circles.

Guindel et al. [10] suggested an intelligent pattern that facilitates the calibration between the stereo camera and LiDAR, even for low-resolution LiDAR systems.. It is a plane featuring four symmetrical circular apertures. The objective is to ascertain four pairs of highly precise circle centres from both stereo and LiDAR coordinates to determine the extrinsic calibration parameters. Figure 2.5 illustrates the processes for extracting four central points from both LiDAR and stereo data. The stereo system employs depth estimation to convert images into a point cloud, which is far denser than LiDAR points. The Semi-Global Matching (SGM) approach is employed for stereo to provide precise depth estimate. Following coarse target segmentation and subsequent fine circle segmentation, the transformation parameters can be determined by minimising point-to-point distances using the Iterative Closest Point (ICP) technique.



(a) Input camera image



(b) Input range data and calibration result

Figure 2.4 : Calibration example from [8].

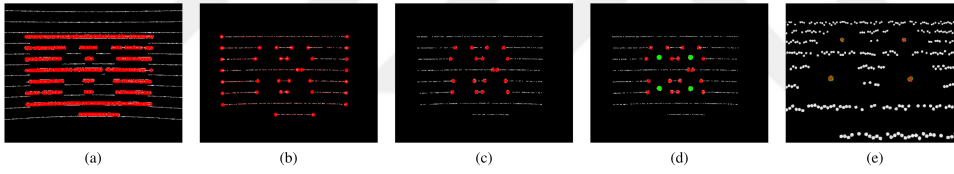


Fig. 3. Segmentation pipeline for extraction of the reference points from the lidar point cloud.

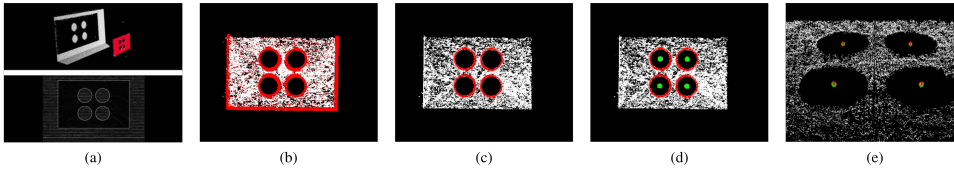


Figure 2.5 : Extraction pipeline for four central points from LiDAR (top) / stereo (bottom) [10].

Recent research has highlighted the importance of targetless methodologies, with one of the earliest contributions coming from [11], introduce an auto-calibration approach designed to address the external calibration issue between the camera and the 3D laser range sensor, depending only on data acquired from natural environments and avoiding the need for any calibration pattern. The authors solve the issue of visually ambiguous and feature-deficient raw depth data from the laser scanner by presenting a novel representation named “Bearing Angle (BA)”. BA pictures enhance

the visibility of depth discontinuities and changes in surface orientation, facilitating the user's ability to identify dependable point correspondences between the camera image and the laser data. Upon the manual determination of a minimum of four laser-camera alignments, the external calibration issue is transformed into a PnP formulation, with the solution derived by a linear PnP algorithm, subsequently refined by a Levenberg-Marquardt-based non-linear optimization. The method provides a practical and portable solution by necessitating the observation of only one laser-camera pair, obviating the requirement for an additional target in the scene, and being compatible with both omnidirectional and perspective cameras. Experimental findings show that eight to ten point correspondences distributed throughout the scene yield a calibration that is notably stable and produces minimal pixel error.

In edge-based calibration experiments [12]–[15], edge features are derived from image and LiDAR data points. The cost function generated from this edge information is subsequently minimised through various optimization algorithms. The study [12] aims to update the extrinsic matrix in real time when detecting miscalibrations between two sensors by probabilistic computation. The proposed technique aims to optimize the cost function, which is predicated on the natural alignment between the generated edge features from images and LiDAR point clouds. This goal is achieved via the execution of grid search optimization. The system presents two complimentary algorithms that operate online, devoid of targets or human interaction, using data collected from actual road environments: (1) detection of sudden miscalibration and (2) ongoing online calibration monitoring. This methods allows the robot to autonomously recalibrate in practical scenarios, including positional alterations of sensors over time, loosening screws, or minor displacements caused by vibration or impact. Figure 2.6 shows the real-time monitoring of calibration errors. In the first frame of the 400-frame sequence, the camera-laser transformation is perfectly tuned. Consequently, desired disturbances are applied to two translational and two rotational parameters during the sequence, leading to an increasingly severe miscalibration. Notwithstanding this synthetic degradation, the proposed online tracking system consistently updates the transformation estimates at each frame, effectively minimizing the generated drift. At the conclusion of the sequence, the algorithm reconstructs the synthetic offsets

with exceptional precision, showcasing its robustness and capacity to uphold precise calibration under dynamic situations.

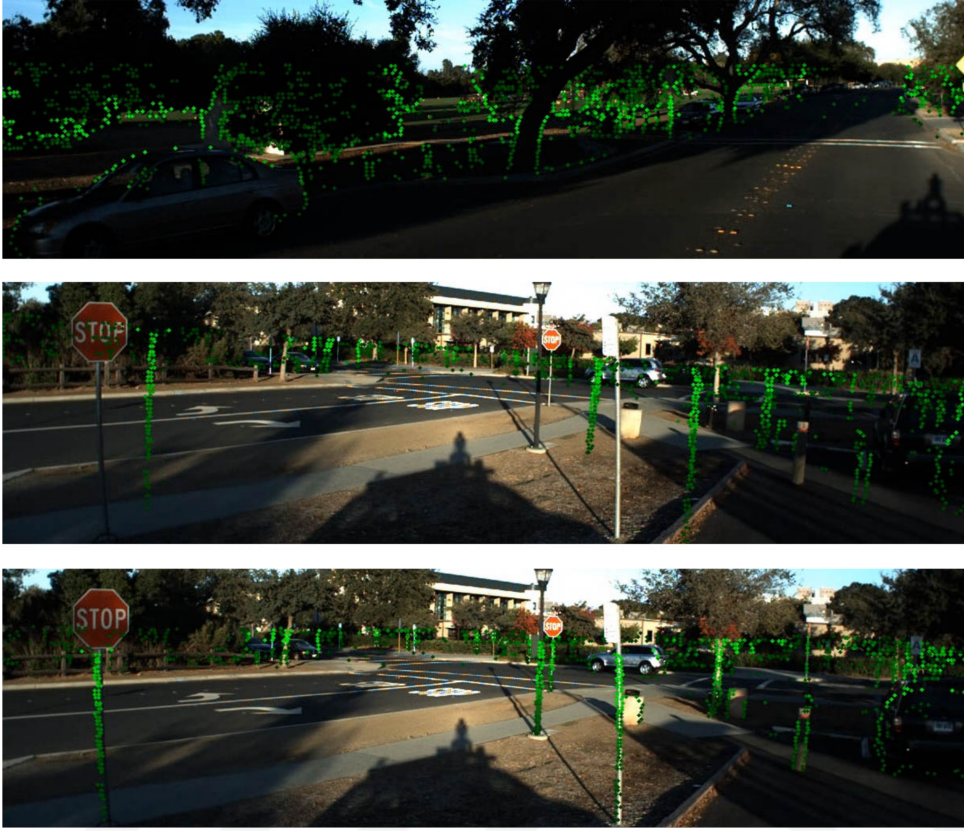


Figure 2.6 : Tracking calibration errors in real time. Initial calibration (Top), perturbed (Middle), recalibrated (Bottom) [12].

In the master’s thesis of Yi Yang [13], a similar approach to [12] is used, in which an automatic extrinsic calibration method between a 3D LiDAR and a camera is proposed based on edge alignment in natural scenes. However, unlike [12], Yang employs an improved edge extraction method for LiDAR point clouds and applies a gradient descent method to iteratively optimize the cost function.

The article [14], introduces an automated external calibration method for high-resolution LiDAR and camera systems, achieving pixel-level precision without the necessity of calibration targets. The authors analyse the principles of physical measurement used in modern LiDAR sensors and illustrate that depth-discontinuous edges, commonly referenced in literature, are unreliable in high-resolution scans. The effects of foreground inflation and bleeding-point, resulting from the divergence of the laser beam, compromise the accuracy of edge positioning. The article presents a depth-continuous,

which is illustrate in Figure 2.7, edge extraction method that is shown in Figure 2.8, that segments the point cloud into small voxel blocks, conducts RANSAC-based plane fitting within each voxel, then extracts the intersecting lines of the planes to address this issue. Edge detection utilising the Canny algorithm is executed on the camera side, with the resulting edges saved in a kd-tree for efficient matching. The acquired LiDAR-camera edge correspondences are resolved using maximum likelihood optimization on SE(3) with a comprehensive noise model. Comprehensive indoor and outdoor testing show that the approach is both resilient and exceptionally precise. The suggested method consistently converged, even within a broad beginning error range ($\pm 5^\circ$ rotation, ± 10 cm translation), and achieved pixel-level reprojection error using a single LiDAR-camera data pair. Moreover, the method produces favourable outcomes not only with solid-state LiDARs but also with spinning LiDARs when sufficient data aggregation is attained. The authors assert that the method's efficacy is dependent on the diversity of edges within the picture, highlighting that calibration may become unstable in environments populated solely by cylindrical objects or where edges are predominantly aligned in a singular orientation. The comparison between extracted depth-continuous edges and depth-discontinuous edges is shown in Figure 2.9.

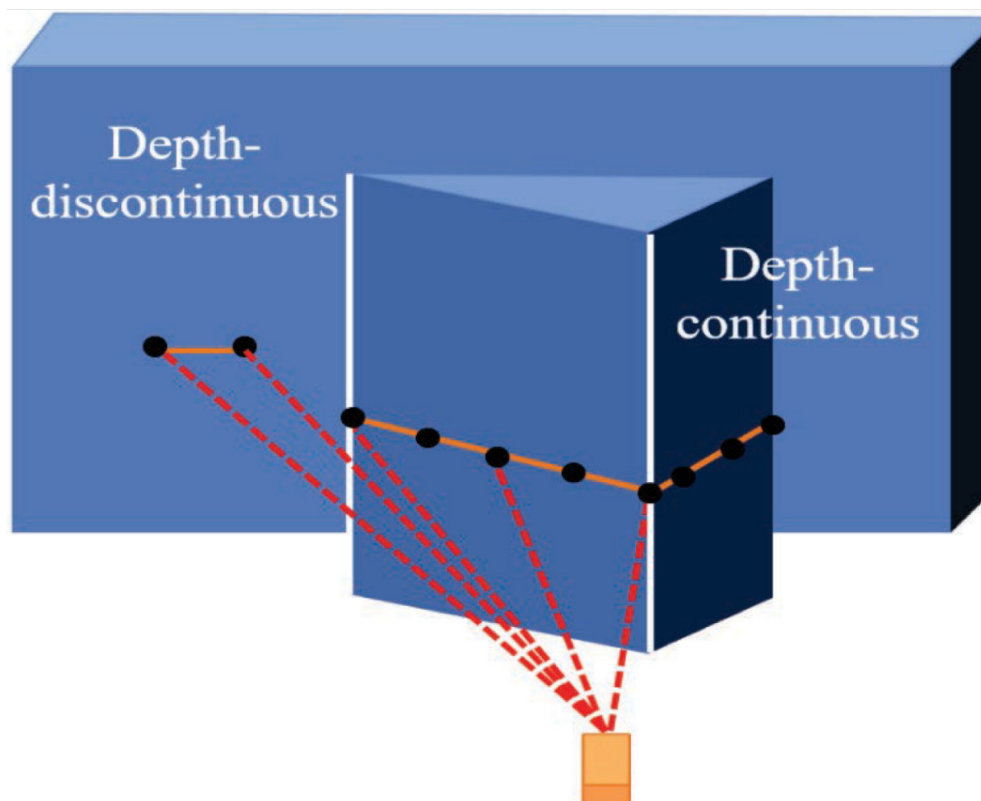


Figure 2.7 : Depth-discontinuous edge and depth-continuous edge [14].

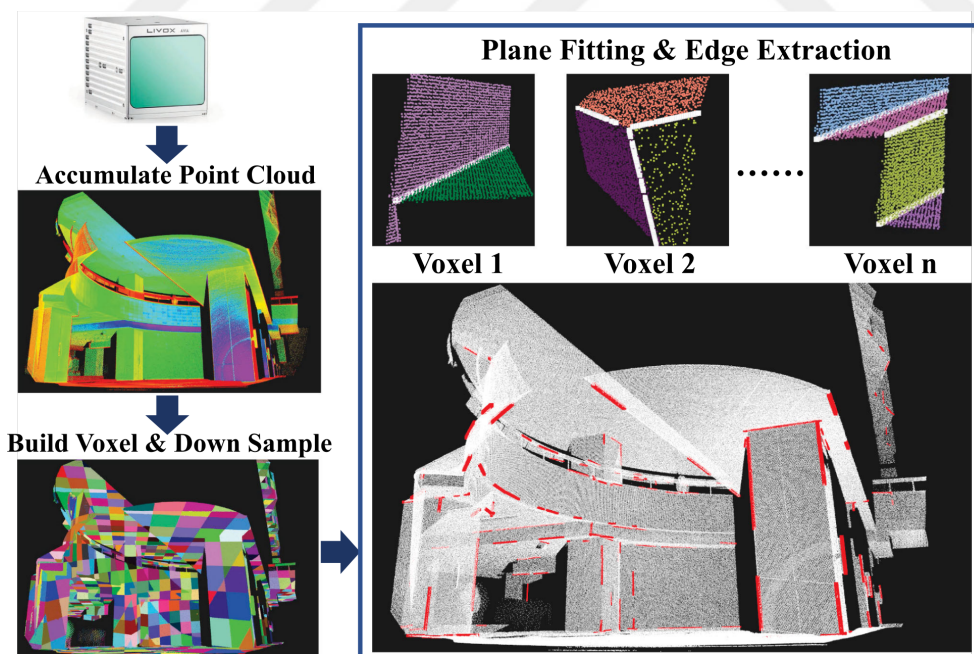


Figure 2.8 : Depth-continuous edge extraction [14].

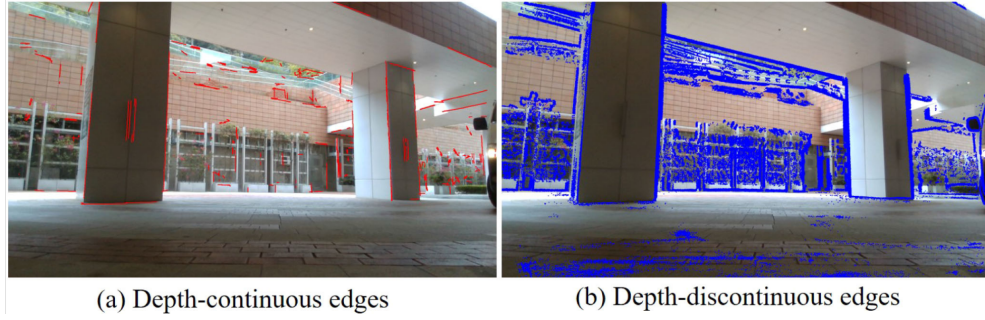


Figure 2.9 : Comparison between depth-discontinuous edges and depth-continuous edges [14].

In addition to edge detection, [16] introduced an approach that uses all overlapping data. The algorithm optimizes the mutual information between the colour image and the reflectivity values from the LiDAR, projected onto the image plane.



3. METHODS AND IMPLEMENTATION

This my thesis, proposed method has a similar structure to the calibration method in [12] but with improvement of data usage, image processing and compare with different optimization technique. In my method utilised data from the KITTI dataset to examine the impact of user-defined or automatically determined rectangular regions of interest (ROI) on LiDAR-camera calibration. Furthermore, calibration results obtained after applying perturbations to ground truth extrinsic parameters at various levels were presented and analyzed. The main contributions of this thesis can be summarized as follows:

- The edge detection was performed only within the manually or automatically determined region of interest (ROI), rather than on the entire image and LiDAR data. The ROI selection method aims to exclude noisy and non-edge regions from the calibration.
- Unlike the previously stated research in the literature that utilise a sliding window and necessitate extensive datasets, the calibration executed on a single frame through ROI selection facilitates the updating of calibration parameters with just one camera-LiDAR data pair.
- Since the area within the ROI is only consideration of the calibration process, rather than the complete image and its associated LiDAR point cloud, the computation and optimization of costs can be executed more efficiently.

Figure 3.1 illustrates the comprehensive architecture of the proposed LiDAR-camera calibration framework. The system has three primary stages: preprocessing, feature extraction, and sensor calibration. The pipeline is designed to process raw LiDAR point clouds and camera images, converting them into edge-based representations allowing for calibration via cost-based optimization.

During the preprocessing phase, raw point clouds and pictures are initially synchronised and subsequently directed to two essential modules: Intrinsic Matrix Calculation

and Cloud Projection. The camera's intrinsic parameters are either obtained and subsequently provided to the cloud-projection module, where the LiDAR points are projected onto the image plane using the current extrinsic parameters. This projection provides a base for subsequent comparisons between LiDAR-edge structures and image edges. The preprocessing phase establishes the groundwork for the whole pipeline by guaranteeing uniform geometric alignment among sensing modalities.

Subsequent to preprocessing, the outputs are directed to the feature extraction phase, which comprises two concurrent operations. The initial method is Point Cloud Edge Extraction, utilised for identifying edge structures within LiDAR data. The second is Image Edge Detection, utilised to identify edges in the camera image. These two feature sets denote the geometric cues to be aligned during the calibration process.

The concluding phase, Sensor Calibrator, is tasked with assessing and rectifying extrinsic alignment. The process commences with the identification of mis-calibration, which determines if the existing extrinsic parameters diverge markedly from a feasible solution. The ROI selection module, either autonomously or manually, identifies the most useful image patches for calibration, hence enhancing robustness against noise and occlusion. The cost computation module assesses the alignment between anticipated LiDAR edges and image edges based on the ROI and a specified cost function. The optimization module iteratively refines the extrinsic matrix by maximising the edge-alignment cost, yielding improved calibration parameters.

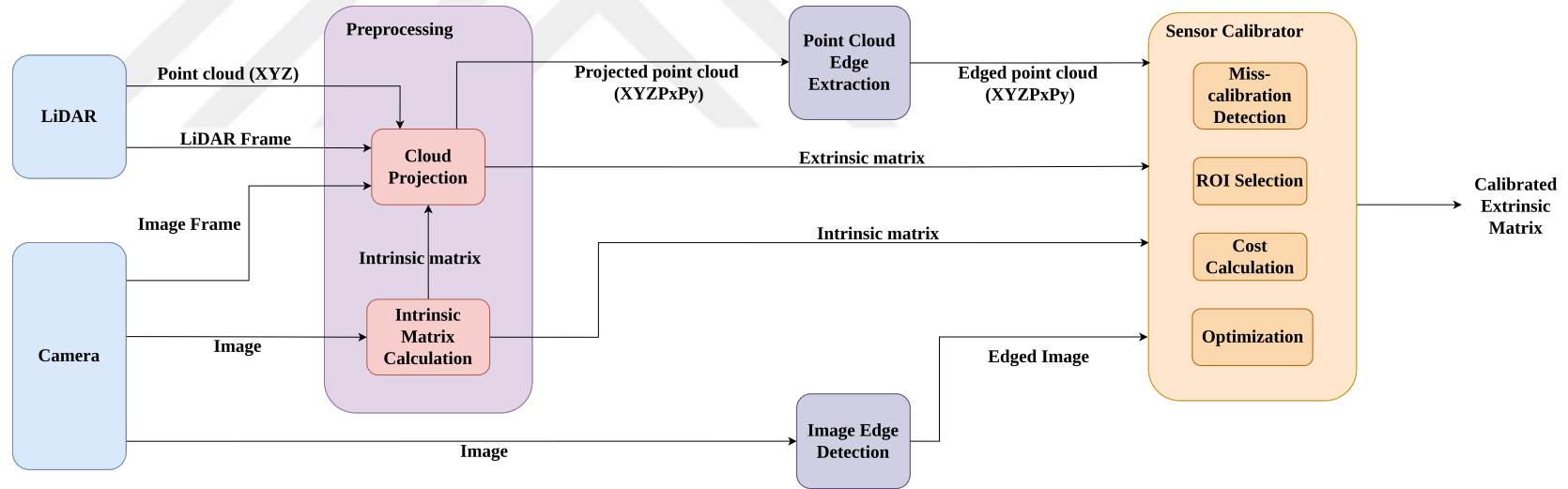
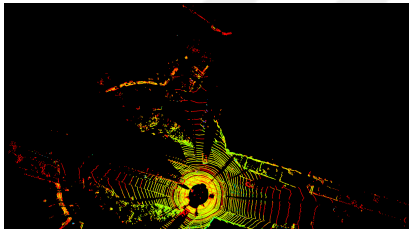


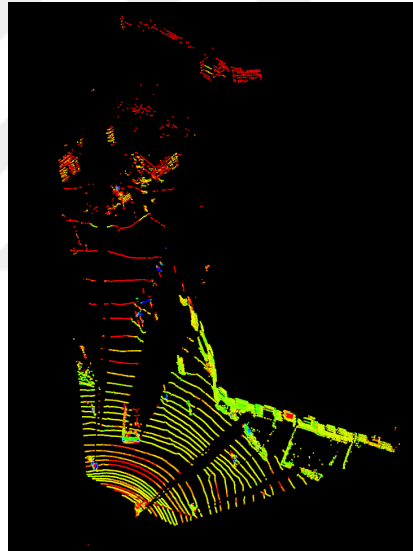
Figure 3.1 : Block diagram of the proposed method.

3.1 LiDAR Point Cloud Edge Extraction Processing

The processing of LiDAR point cloud edge extraction requires multiple stages. Initially, during the preprocessing phase, the 360-degree LiDAR data taken from the KITTI dataset is projected onto the camera coordinate system. The projection procedure utilises the extrinsic matrix and intrinsic camera parameters acquired from the prior offline calibration between the camera and LiDAR sensors. The objective is to utilise solely the points that intersect with the image. The residual points are discarded. Figure 3.2 illustrates the raw LiDAR point cloud and the projected LiDAR point cloud onto the camera coordinate frame. Figure 3.3 shows the projected LiDAR point cloud from the camera's perspective.



(a)



(b)

Figure 3.2 : (a) Raw LiDAR point cloud (b) Projected LiDAR point cloud - Filtered.

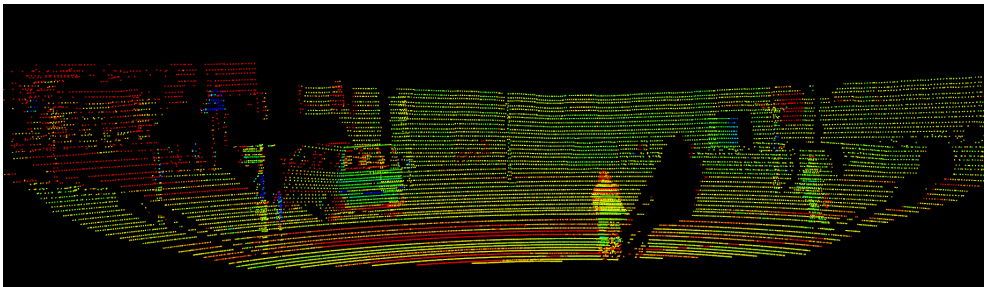


Figure 3.3 : Projected LiDAR point cloud from camera perspective.

The following step involves edge extraction. This stage utilises the methodology first proposed by [12] for LiDAR edge detection. This method employs depth-discontinuity logic, processing each laser beam independently, and identifies locations that are nearer than at least one of their neighbours as probable edge points. Points that are distanced from their neighbours are less likely to align with image edges and are consequently excluded.

$$X_p^i = (P_{p-1}^i \cdot r - P_p^i \cdot r, P_{p+1}^i \cdot r - P_p^i \cdot r, 0)^\gamma \quad (3.1)$$

In the Equation 3.1, the $\cdot r$ suffix denotes the LiDAR range measurement in meters corresponding to that specific point. The symbol P_p^i represents the LiDAR point at index p on scan line i , while P_{p-1}^i and P_{p+1}^i denote its immediate neighbors in the same scan line. The indices $p-1$, p , $p+1$ therefore correspond to consecutive points along the beam. The term X_p^i is the edge response value calculated for point p . The value of γ is set to 0.5, and, to enhance efficiency, all points exhibiting a depth discontinuity of less than 30cm are filtered out. The results of the LiDAR point cloud edge detection method can be seen in the below Figure 3.4 shows the projected LiDAR point cloud from the camera's perspective - zoomed in, and Figure 3.5 shows the edged LiDAR point cloud from the camera's perspective - zoomed in. The edged LiDAR points on image also shown in Figure 3.6.

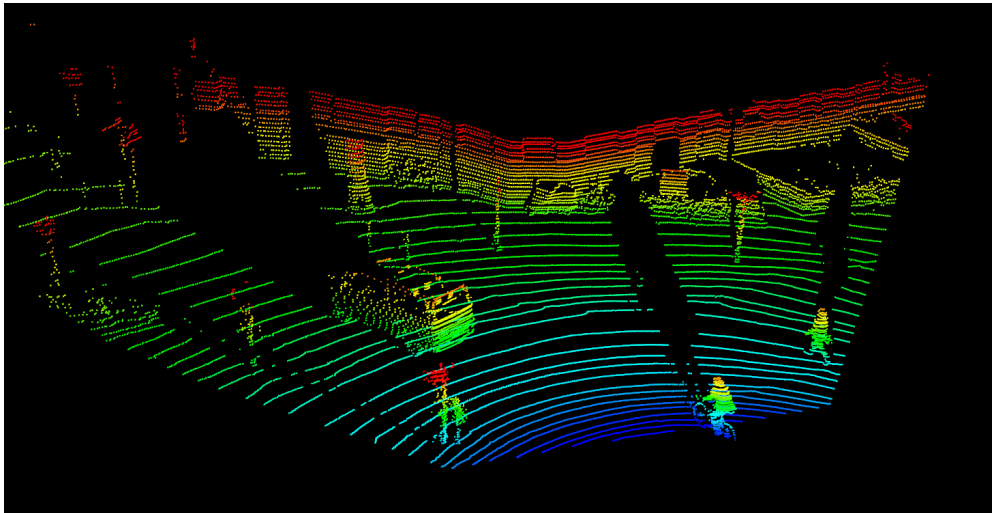


Figure 3.4 : Projected LiDAR point cloud from camera perspective (zoomed in).

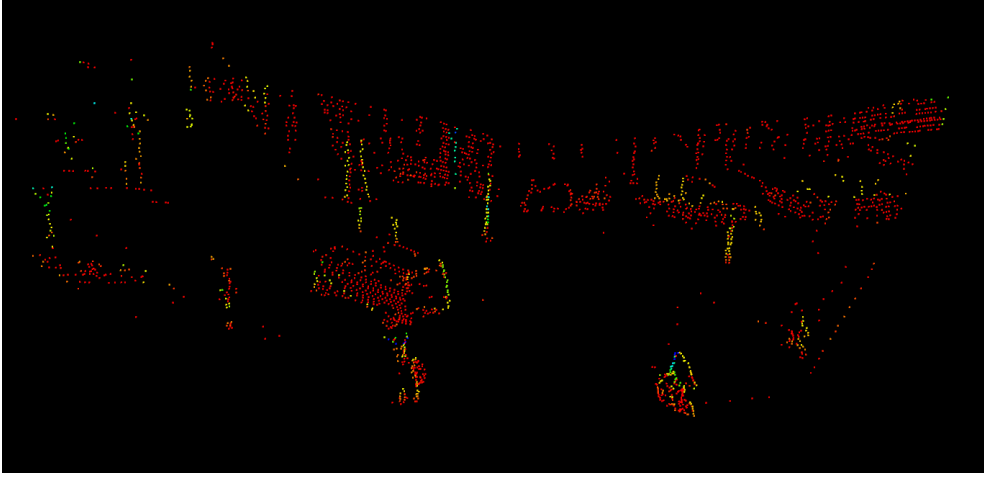


Figure 3.5 : Edged LiDAR point cloud from camera perspective (zoomed in).



Figure 3.6 : Edged LiDAR point cloud on image.

3.2 Image Processing

To extract edges from the image domain, a multi-stage processing pipeline was implemented. First, the RGB image is converted to grayscale. To suppress noise and avoid spurious gradients, a Gaussian blur is applied. Then, the Canny edge detection algorithm [17] is performed to obtain binary edge maps. To improve the robustness of edge localization, the resulting edge image is inverted and processed with a distance transform (L2 metric, 5×5 neighborhood). This step assigns higher values to pixels located closer to edges. The transformed image is then normalized into the range $[0, \beta]$, where β is a tunable parameter controlling edge sensitivity. Finally, the normalization is inverted so that pixels corresponding to edges obtain high values, producing an “edginess map” that is used in the cost function. All preprocessing steps apart from

Canny edge detection were implemented using the OpenCV library [18]. Figure 3.7 illustrates the RGB image and the resulting edginess map with $\beta = 10$.



(a)



(b)

Figure 3.7 : (a) RGB image (b) Edginess map with $\beta = 10$.

3.3 Region of Interest (ROI) Processing

The objective of the Region of Interest (ROI) processing stage is to concentrate on edge regions, hence selecting these areas over the processing of the full picture and projected LiDAR point cloud. Two different methodologies are proposed for ROI selection: manually selecting the ROI and automatically selecting the ROI.

3.3.1 Manually selecting the ROI

In this section, selecting the calibration working area requires the user to determine the region of interest by referring to the illustration provided in Figure 3.6. Specifically, the user must consider the following questions: Where is the edge-rich region? Which region should be selected to improve calibration accuracy? And in which area can the distortion be observed most clearly? The user then manually selects a rectangular ROI on the image that contains significant edge features while excluding regions with minimal or no edge information. This manual selection enables a more targeted

calibration process by focusing on areas that provide the most informative cues for aligning the LiDAR and camera data. Figure 3.8 illustrates an example of a manually selected ROI on the image. Within this ROI, an examples of image processing are presented in Figure 3.9.

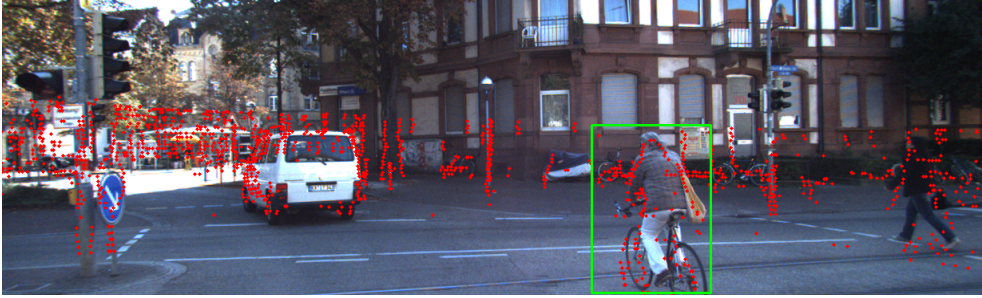


Figure 3.8 : Manually selected ROI on the image.



Figure 3.9 : (a) RGB image within the ROI (b) Edginess map within the ROI.

3.3.2 Automatically selecting the ROI

During the manual selection of regions of interest, it has been observed that the edges of objects on the road, including vehicles, traffic signs, poles, and pedestrians, are more pronounced in the projected point cloud. The YOLOv7-X model [19] was employed utilising class definitions derived from the Berkeley DeepDrive (BDD100K) dataset [20].

An image object detection procedure was executed on the RGB image by developing a ROS wrapper for the YOLOv7-X model, and one or more regions of interest were designated on the image. Figure 3.10 illustrates the results of the YOLOv7-X object detection on the image. The bounding boxes of these detected objects are then utilised to define the regions of interest for the calibration process.



Figure 3.10 : YOLOv7-X object detections on the image.

Major improvements have been implemented in automatic ROI selection utilising the YOLOv7-X model. Initially, bounding boxes without projected lidar points are not subjected to processing. In the illustration presented in Figure 3.10, the traffic sign is disregarded as it is positioned above the lidar points cloud. Secondly, the bounding box surrounding the detected object has been correspondingly expanded to accommodate distortions of ± 10 cm and ± 1 degree. The objective is to ensure that projected LiDAR points remain within the bounding box despite the perturbed extrinsic matrix, thus establishing a calibration workspace for each ROI. Finally, to eliminate distant detections with few points and unable to yield meaningful edginess values for image edge detection, the confidence threshold and intersection over union (IoU) threshold values were established at 0.8 and 0.65, respectively. This enabled the designation of ROIs with relatively greater proximity and reliability. This enhancement, which is vital for calibration robustness, guaranteed that extraneous and noise ROIs were excluded from the cost computation. Subsequent to object detection, only the lidar point cloud within the regions of interest is processed, similar to manual ROI selection, and calibration is conducted within that area. During the process of manual ROI, a singular cost value was produced; in this instance, the cost values from each ROI are aggregated and divided by the total number of ROIs to derive an average value. The cost employed in the optimization process is this average cost value. The subsequent

steps after selecting ROI like edge detection processes follow the same procedure as with manual ROI.

3.4 Cost Function

After selecting the ROI in the image, the chosen region is employed to evaluate the cost function which is shown in the Equation 3.2 that measures the alignment between the LiDAR edge points and the image edges. The cost value indicates how much correlation between these two data sets. Since both camera and Lidar are observed the same scene, they are supposed to have a strong correlation. More accurate the calibration matrix is, larger the value will be. In contrast to the studies that average the cost function over a moveable window of frames, proposed method does calibration using a single frame and the chosen region of interest.

For each LiDAR point p within the ROI, the three-dimensional point is projected onto the image plane using the intrinsic–extrinsic transformation. The projected coordinates (u_p, v_p) are verified to fall inside the ROI, and the local edge response $D(u_p, v_p)$ is obtained from the edge-processed image using bilinear interpolation. Each contribution to the cost is then computed as the product of the LiDAR point’s depth discontinuity value r_p and the interpolated edge response. where N is the number of valid projected points within the ROI, $D(u_p, v_p)$ is the bilinearly interpolated edge value at the projected pixel, and r_p denotes the LiDAR depth discontinuity associated with point p . This scalar measure J_c is subsequently optimized using the methods described in Section 3.6 to refine the extrinsic calibration parameters.

$$J_c = \frac{1}{N} \sum_{p=1}^N (D(u_p, v_p) \cdot \Delta r_p) \quad (3.2)$$

Cost function for automatic ROI selection

When multiple ROIs are automatically selected, the calibration cost is evaluated independently for each valid ROI. Let J_c^k denote the cost computed for the k -th ROI using the formulation given in Equation 3.2. Only ROIs that contain a sufficient number of projected LiDAR points are considered valid for evaluation.

The overall calibration cost is defined as the average cost over all valid ROIs and is expressed as

$$J_{\text{avg}} = \frac{1}{N_{\text{valid}}} \sum_{k=1}^{N_{\text{ROI}}} J_c^k, \quad (3.3)$$

where N_{ROI} denotes the total number of automatically selected regions of interest and N_{valid} represents the number of ROIs that contain sufficient projected LiDAR points.

By averaging costs across multiple regions, the calibration process becomes more robust to localized variations and benefits from diverse geometric constraints provided by multiple object instances in the scene. The resulting scalar measure J_{avg} is subsequently optimized using the methods described in Section 3.6 to refine the extrinsic calibration parameters.

3.5 Miss-calibration

Miss-calibration is essential for autonomous driving. It is important to make sure that sensor calibration (specifically LiDAR-camera calibration in this thesis) has been executed accurately. Miss-calibration detection must be conducted if sensor locations change during online driving, following offline driving, or if deviations arise due to vibrations. It is added to system to verify the correctness of the calibration parameters using a grid search.

Given a small perturbation (positive and negative with a fixed perturbation) applied to each of the six extrinsic parameters, namely the translation components $[x, y, z]$ and the rotation angles $[\theta_{\text{roll}}, \theta_{\text{pitch}}, \theta_{\text{yaw}}]$, a total of $3^6 = 729$ cost values are evaluated. These values include the cost corresponding to the original calibration parameters as well as the costs of the remaining 728 perturbed cases. For the translation parameters, a perturbation magnitude of 0.01 m is used, while for the rotation parameters, the perturbation magnitude is set to 1° . The percentage of perturbations yielding a worse cost value than the original calibration, denoted by F_C , is computed as,

$$F_C(T_{\text{center}}) = \frac{\#\{T_{\text{nbr}} \mid J_c(T_{\text{nbr}}) < J_c(T_{\text{center}})\}}{729 - 1} \quad (3.4)$$

where T_{center} represents the original extrinsic transformation, T_{nbr} denotes each of the 728 perturbed transformations, and $J_c(\cdot)$ is the cost function defined in Equation 3.2 or Equation 3.3 depending on the ROI selection method used.

F_C , is used to evaluate the correctness of the calibrations. The experiment utilising the KITTI dataset demonstrates an important difference in F_C between correct and incorrect calibration values, as illustrated in Figure 3.11. Figure 3.11 presents a comparison between the accurate calibration parameters and five different incorrect values for automatic ROI selection in other word multiple ROI selection. The accurate calibration parameters are derived from the KITTI offline calibration results, which are considered the ground truth. Five additional sets of incorrect parameters are randomly chosen, exhibiting variances in their translation values $[X, Y, Z]$ compared to those of KITTI. The variations span from 0.5 m to 1.5 m over three translational dimensions $[X, Y, Z]$. Table 3.1 reports the translational perturbations applied to generate the calibration cases shown in Fig. 3.11. For the correct calibration case, all translation parameters are set to zero, indicating that no perturbation is applied and the parameters correspond to the ground-truth extrinsic calibration. For the incorrect calibration cases, the translation parameters are deliberately perturbed by the amounts listed in the table, where positive and negative values indicate deviations from the ground-truth calibration along the corresponding axes. The perturbation magnitudes are adapted from [13].

Table 3.1 : Perturbed translation parameters in meters for correct and incorrect calibration cases.

Unit (meter)	X	Y	Z
<i>Correct</i>	0.000	0.000	0.000
<i>Incorrect₁</i>	0.941	1.075	1.272
<i>Incorrect₂</i>	0.491	0.505	0.542
<i>Incorrect₃</i>	-0.509	-0.495	-0.458
<i>Incorrect₄</i>	-1.500	-1.500	-1.500
<i>Incorrect₅</i>	1.500	1.500	1.500

In Figure 3.11, usually the correct calibration parameters exhibit a higher F_C value of approximately 0.9, indicating that 90% of the perturbed cases yield worse cost values compared to the original calibration. In contrast, all incorrect calibration parameters demonstrate much lower F_C values. This stark difference in F_C values between correct and incorrect calibrations highlights the correctness of the miss-calibration detection method in distinguishing accurate sensor alignments from incorrect ones.

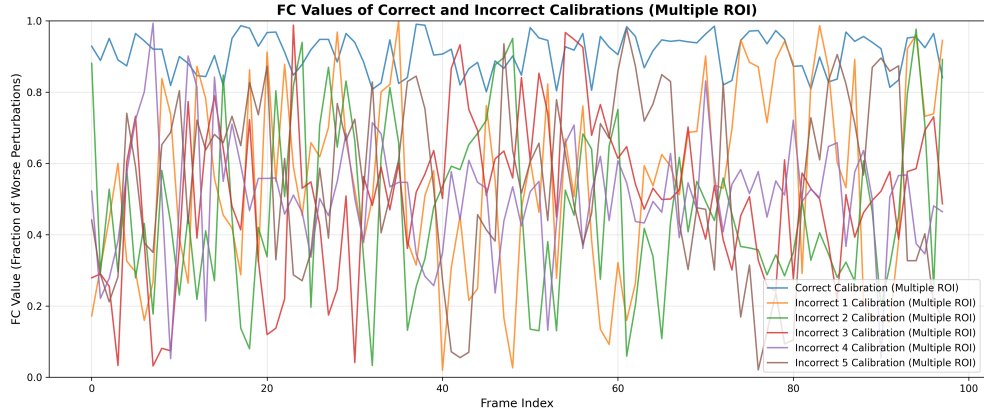


Figure 3.11 : Percentage of the worse cases (F_C) of correct and incorrect calibration parameters with small perturbation for multiple ROI selection in single frame.

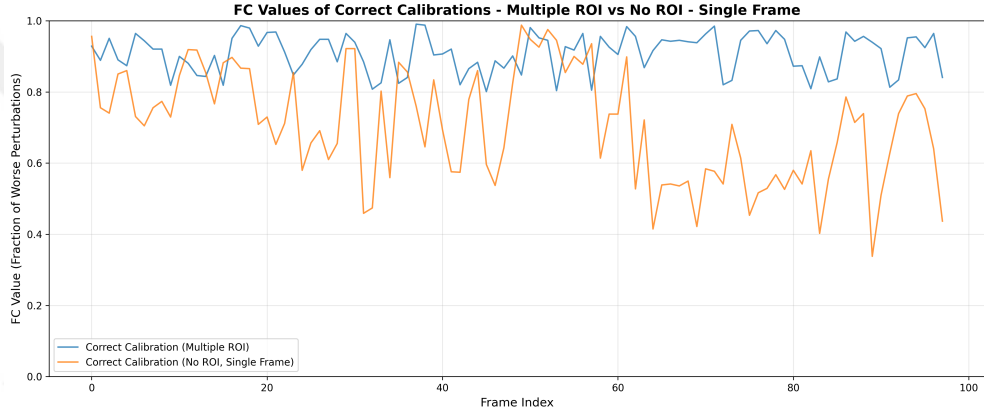


Figure 3.12 : Percentage of the worse cases (F_C) of correct and incorrect calibration parameters with small perturbation for multiple ROI selection vs no ROI selection in single frame.

Figure 3.12 presents a comparison of F_C values between the proposed automatic ROI selection method and a baseline approach that does not utilise any ROI selection, both evaluated on a single frame. The results indicate that the automatic ROI selection method consistently achieves higher F_C values for the correct calibration parameters compared to the no-ROI baseline. This suggests that focusing on edge-rich regions through ROI selection enhances the sensitivity of the miss-calibration detection process, leading to more reliable identification of accurate sensor alignments.

Figure 3.13 is, a one frame from KITTI dataset, an example illustration of the projection of edged LiDAR points onto the RGB image for both correct calibration (Figure 3.13a) and miss-calibration (Figure 3.13b). In the correct calibration scenario, the edged LiDAR points align well with the corresponding edges in the RGB image, indicating



(a) Correct calibration



(b) Miss-calibration

Figure 3.13 : Calibration visualization between correct and miss-calibration scenarios. accurate sensor alignment. Conversely, in the miss-calibration case, there is a noticeable misalignment between the edged LiDAR points and the image edges, highlighting the discrepancies caused by incorrect calibration parameters. This visual comparison underscores the importance of accurate calibration for correctness in sensor fusion in autonomous systems.

Although correct calibration parameters usually yield high F_C values (approximately 0.9), this qualitative observation alone is not sufficient to reliably characterize calibration correctness. Therefore, the distribution of F_C values is modeled using a Gaussian distribution which is shown in Equation 3.5, enabling a quantitative and statistically grounded criterion to determine whether a given calibration is correct or miss-calibrated. A conventional Gaussian distribution is utilised to fit the histogram distribution for the purpose of estimating the optimal mean μ and standard deviation σ . These two metrics exhibit significant differences between correct and incorrect calibration scenarios. Consequently, they may serve as distinguishing criteria for identifying miscalibration.

$$D = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \quad (3.5)$$

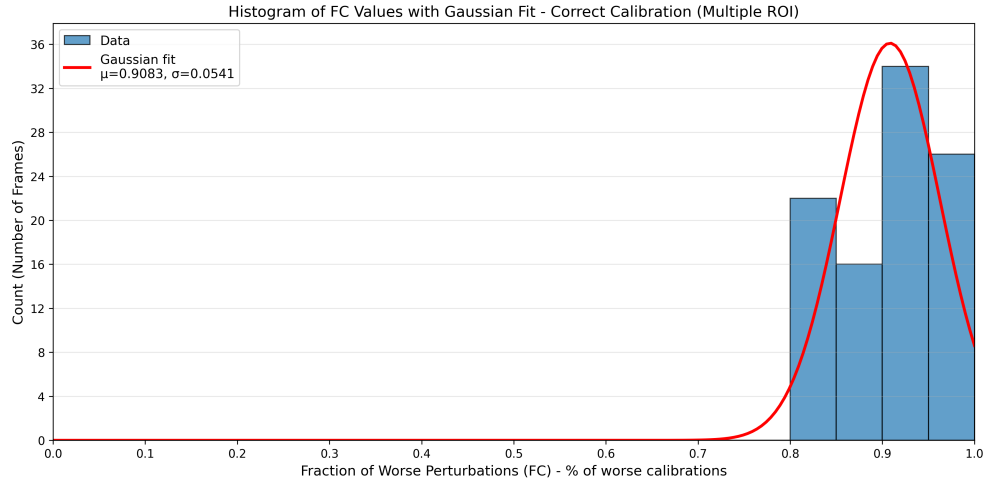


Figure 3.14 : Histogram of FC values with Gaussian fit for correct calibration.

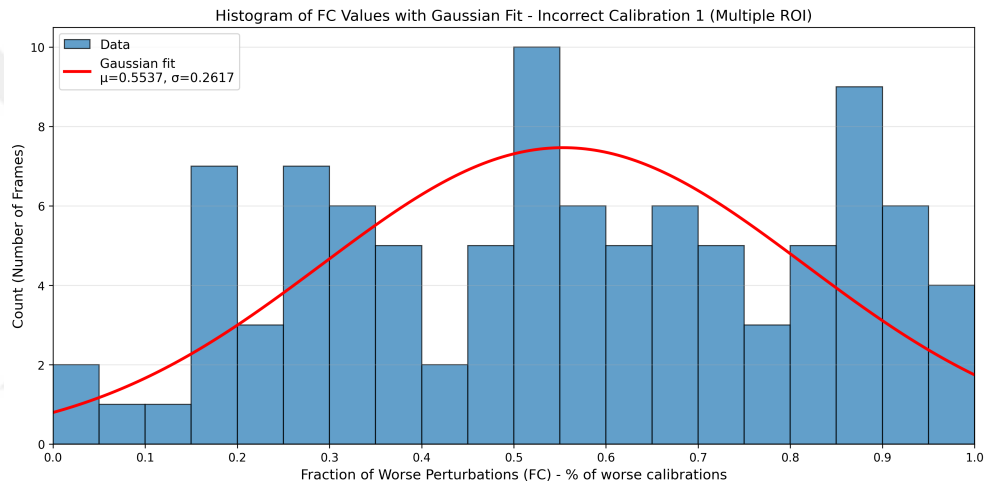


Figure 3.15 : Histogram of FC values with Gaussian fit for incorrect calibration 1.

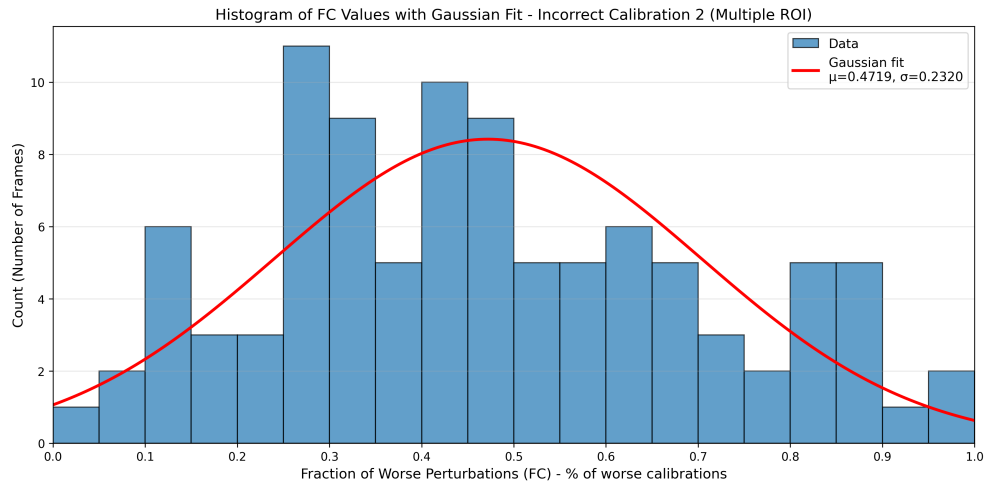


Figure 3.16 : Histogram of FC values with Gaussian fit for incorrect calibration 2.

The histogram plots generated from the correct and incorrect calibration data presented in Figure 3.11 are shown in Figures 3.14, 3.15, 3.16, 3.17, 3.18, and 3.19. As shown in

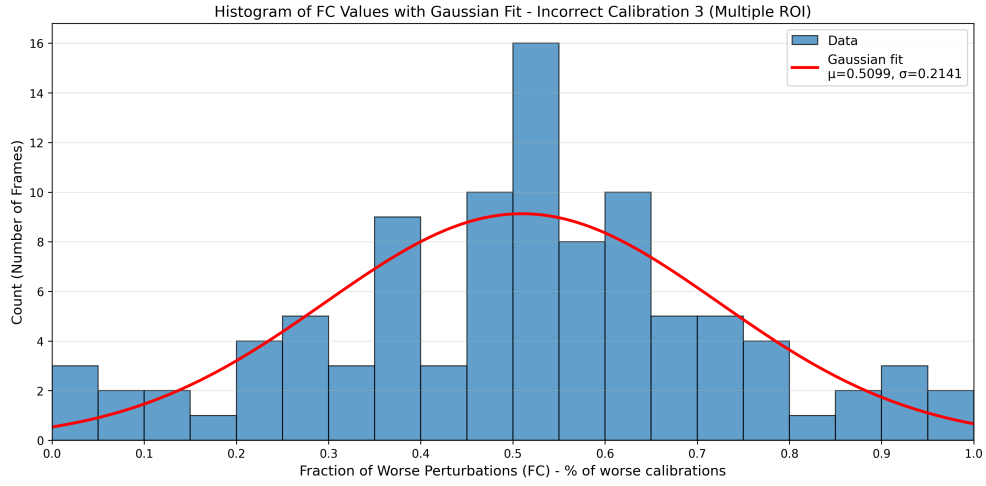


Figure 3.17 : Histogram of FC values with Gaussian fit for incorrect calibration 3

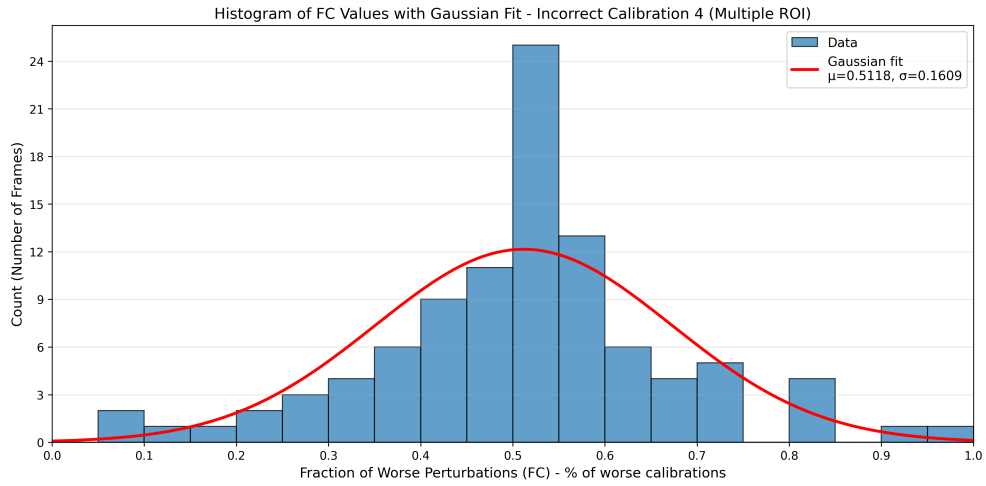


Figure 3.18 : Histogram of FC values with Gaussian fit for incorrect calibration 4.

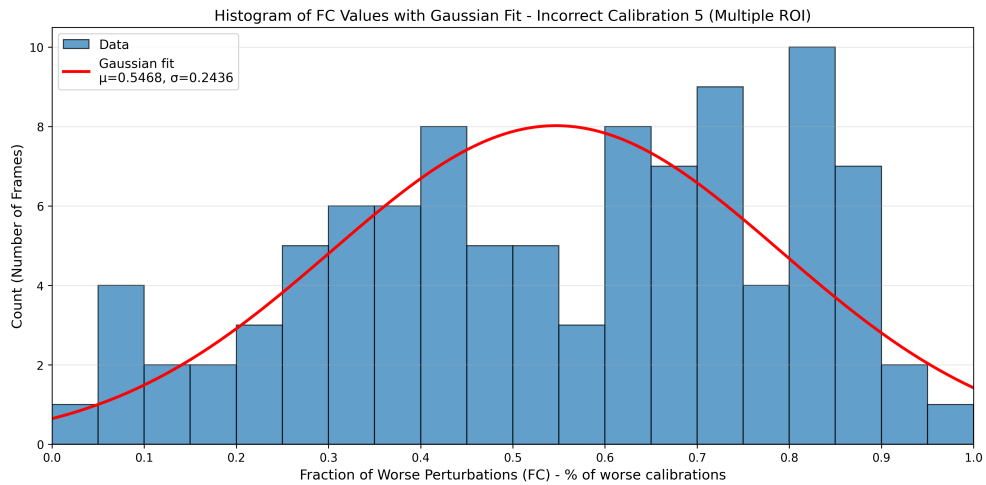


Figure 3.19 : Histogram of FC values with Gaussian fit for incorrect calibration 5.

these figures, the Gaussian fit effectively captures the distribution of F_C values for both correct and incorrect calibration scenarios. Notably, there is a significant difference in

the estimated mean μ and standard deviation σ between the two cases. The correct calibration scenario exhibits a higher mean and lower standard deviation, indicating a more concentrated distribution of F_C values around the optimal value. In contrast, the incorrect calibration scenarios show lower means and higher standard deviations, reflecting a broader spread of F_C values. These differences in statistical parameters can be leveraged as reliable criteria for distinguishing between correct and miss-calibrated sensor alignments. The complete results of μ and σ of the fitted Gaussian distribution of six groups are shown in Table 3.2.

Table 3.2 : Gaussian distribution parameters (μ and σ) for correct and incorrect calibration cases: interval in μ indicates 90% confidence interval.

Calibration Case	μ	σ
Correct	0.908303 [0.899221, 0.917385]	0.054138
Incorrect 1	0.553656 [0.509751, 0.597560]	0.261713
Incorrect 2	0.471883 [0.432960, 0.510806]	0.232021
Incorrect 3	0.509854 [0.473933, 0.545774]	0.214122
Incorrect 4	0.511802 [0.484817, 0.538787]	0.160857
Incorrect 5	0.546843 [0.505976, 0.587711]	0.243608

In both Table 3.2 and histograms in Figure 3.14 to Figure 3.19, the correct calibration values indicate that μ exceeds 0.9. In cases of incorrect calibrations, however, μ is less than 0.55. Therefore, μ can quantitatively assess if a set of calibration parameters is miscalibrated.

3.6 Optimization

Two different methods were used for optimizing the extrinsic parameters: a greedy hill-climbing strategy with discrete perturbations and Particle Swarm Optimization (PSO) [21].

3.6.1 Greedy hill-climb grid search

The method of Greedy Hill-Climb Grid Search aims to maximize the cost function defined in Section 3.4 by iteratively adjusting the six extrinsic parameters: three translations (X, Y, Z) and three rotations (Roll, Pitch, Yaw).

The algorithm starts from an initial estimate of the extrinsic transformation matrix T . At each iteration, a discrete neighborhood of candidate transformations $\mathcal{N}(T)$ is generated by perturbing each of the six extrinsic parameters (X, Y, Z, Roll, Pitch, Yaw) using a fixed step size of $\Delta_t = 0.01$ m for translation and $\Delta_r = 0.01^\circ$ for rotation. This discrete perturbation scheme yields a total of $3^6 = 729$ candidate transformations, including the current estimate.

Each candidate transformation $T_i \in \mathcal{N}(T)$ is evaluated using the cost function $J_c(T_i)$ or $J_{avg}(T_i)$ depending on the ROI selection method defined in Section 3.4. The candidate achieving the maximum cost value, denoted as T_{best} , is identified. If the cost improvement of T_{best} over the current estimate exceeds a small numerical tolerance $\varepsilon = 10^{-12}$, the algorithm performs a greedy update by setting $T \leftarrow T_{\text{best}}$.

In addition, the F_C is computed as the ratio of candidate transformations whose cost values are worse than that of the current estimate. If F_C exceeds a predefined threshold of $\tau = 0.9$, indicating that at least 90% of the neighborhood yields inferior cost values, the algorithm terminates early, assuming convergence. Otherwise, the iterative process continues until convergence is achieved or a maximum number of $K_{\text{max}} = 100$ iterations is reached. The final refined extrinsic transformation matrix T is returned as the output. The complete pseudo-code of the proposed Greedy Hill-Climb Grid Search optimization is provided in Algorithm 1.

3.6.2 Particle swarm optimization (PSO)

The extrinsic parameters are optimized using a deterministic PSO algorithm. The choice of PSO over other optimization methods is motivated by its effectiveness in handling multi-dimensional, non-convex optimization problems that are inherent in sensor calibration tasks.

Algorithm 1 Greedy Hill-Climb Grid Search for Extrinsic Calibration.

Require: Initial extrinsic transform T , step sizes $\Delta_t = 0.01$ m, $\Delta_r = 0.01^\circ$, tolerance $\varepsilon = 10^{-12}$, threshold $\tau = 0.9$, max iterations $K_{\max} = 100$

Ensure: Refined extrinsic transform T

```
1: for  $k = 1$  to  $K_{\max}$  do
2:   Generate candidate set  $\mathcal{N}(T)$  by applying  $\{-\Delta, 0, +\Delta\}$  perturbations to each
     of the six parameters
     Total candidates:  $|\mathcal{N}(T)| = 3^6 = 729$ , including  $T_{\text{center}} = T$ 
3:   for all  $T_i \in \mathcal{N}(T)$  do
4:      $J_i \leftarrow J_c(T_i)$ 
5:   end for
6:    $J_{\text{center}} \leftarrow J_c(T_{\text{center}})$ 
7:    $T_{\text{best}} \leftarrow \arg \max_{T_i \in \mathcal{N}(T)} J_c(T_i)$ 
8:    $J_{\text{best}} \leftarrow J_c(T_{\text{best}})$ 
9:    $F_C \leftarrow \frac{|\{T_i \in \mathcal{N}(T) \setminus \{T_{\text{center}}\} : J_c(T_i) < J_{\text{center}}\}|}{|\mathcal{N}(T)| - 1}$ 
10:  if  $J_{\text{best}} > J_{\text{center}} + \varepsilon$  then
11:     $T \leftarrow T_{\text{best}}$ 
12:  end if
13:  if  $F_C \geq \tau$  then
14:    break
15:  end if
16: end for
17: return  $T$ 
```

Among various PSO variants documented in literature [22], it is adopted a hybrid approach combining Time-Varying Inertia Weight (TVIW) with Time-Varying Acceleration Coefficients (TVAC). This strategy leverages the strengths of both mechanisms to achieve a balance between exploration and exploitation throughout the optimization process.

The inertia weight w decreases linearly from an initial value $w_{\max}$ to a final value $w_{\min}$ across iterations:

$$w(t) = w_{\max} - \left(\frac{w_{\max} - w_{\min}}{T} \right) \cdot t \quad (3.6)$$

where t is the current iteration and T is the maximum number of iterations.

The acceleration coefficients c_1 and c_2 vary inversely with iteration progress. Initially, c_1 (cognitive parameter) is large to encourage individual exploration and discovery of promising regions. As iterations progress, c_2 (social parameter) increases, promoting

convergence toward the global best solution:

$$c_1(t) = c_{1,\max} - \left(\frac{c_{1,\max} - c_{1,\min}}{T} \right) \cdot t \quad (3.7)$$

$$c_2(t) = c_{2,\min} + \left(\frac{c_{2,\max} - c_{2,\min}}{T} \right) \cdot t \quad (3.8)$$

This hybrid approach ensures that the algorithm maintains adequate diversity during early iterations while converging stably in later iterations. The decreasing inertia weight helps stabilize the solution, while the shifting acceleration coefficients naturally transition the swarm from exploring diverse solutions to exploiting promising regions.

The velocity update equation for particle i at iteration t is:

$$\begin{aligned} \vec{v}_i(t+1) = & w(t) \cdot \vec{v}_i(t) \\ & + c_1(t) \cdot r_1 \cdot (\vec{p}_{\text{best},i} - \vec{x}_i(t)) \\ & + c_2(t) \cdot r_2 \cdot (\vec{p}_{\text{best},g} - \vec{x}_i(t)) \end{aligned} \quad (3.9)$$

where $\vec{v}_i$ is the velocity of particle i , $\vec{x}_i$ is its position (6-DOF extrinsic parameters), $\vec{p}_{\text{best},i}$ is the particle's personal best position, $\vec{p}_{\text{best},g}$ is the global best position found by the swarm, and $r_1, r_2 \in [0, 1]$ are random values generated deterministically for reproducibility.

The PSO algorithm is implemented with deterministic randomization to ensure reproducibility. Each thread maintains its own random number generator seeded with a deterministic value derived from the iteration number and thread identifier:

$$\text{seed} = 42 + \text{iteration} \times \text{numThreads} + \text{threadID} \quad (3.10)$$

This approach preserves the stochastic nature of PSO while allowing results to be consistent across multiple runs. Multi-threading is employed to accelerate particle evaluation, with the number of active threads set to the number of hardware threads available minus one to maintain system responsiveness.

Velocity clamping is applied separately for translation and rotation components to maintain physical plausibility:

$$v_{\max,\text{trans}} = \frac{\Delta_{\text{trans}}}{k_{\text{trans}}}, \quad v_{\max,\text{rot}} = \frac{\Delta_{\text{rot}}}{k_{\text{rot}}} \quad (3.11)$$

where Δ_{trans} and Δ_{rot} represent the search ranges for translation and rotation, respectively, and $k_{\text{trans}}, k_{\text{rot}}$ are scaling factors that control the exploration scope.

The algorithm terminates after a fixed number of iterations, and the global best position found during the optimization process is selected as the final calibrated extrinsic transformation matrix. The complete pseudo-code of the proposed PSO optimization for extrinsic calibration is provided in Algorithm 2.



Algorithm 2 Particle Swarm Optimization (PSO) for Extrinsic Calibration.

Require: Initial parameter vector $\mathbf{x}_0 = [x, y, z, \phi, \theta, \psi]^\top$, $\Delta_t \in [t_{\min}, t_{\max}]$ with $t_{\min} = -0.15$, $t_{\max} = 0.15$, $\Delta_r \in [r_{\min}, r_{\max}]$ with $r_{\min} = -1.0^\circ$, $r_{\max} = 1.0^\circ$, $N_p = 32$, $K = 50$, $w_{\min} = 0.3$, $w_{\max} = 0.6$, $c_{1,\min} = 1.0$, $c_{1,\max} = 2.5$, $c_{2,\min} = 1.5$, $c_{2,\max} = 2.5$, $k_{\text{Trans}} = 30$, $k_{\text{Rot}} = 200$, seed rule: seed = $42 + k \cdot N_{\text{threads}} + \text{threadID}$

Ensure: Best parameter vector $\vec{p}_{\text{best},g}$ and corresponding transform $T(\vec{p}_{\text{best},g})$

```
1: Compute velocity limits:  $v_{\max}^t \leftarrow \frac{t_{\max} - t_{\min}}{k_{\text{Trans}}}$ ,  $v_{\max}^r \leftarrow \frac{r_{\max} - r_{\min}}{k_{\text{Rot}}}$ 
2: for  $i = 1$  to  $N_p$  do
3:   Initialize particle position  $\mathbf{x}_i \leftarrow \mathbf{x}_0$ 
4:   Initialize velocity  $\mathbf{v}_i \leftarrow \mathbf{1}_6$ 
5:    $\vec{p}_{\text{best},i} \leftarrow \mathbf{x}_i$ ,  $J(\vec{p}_{\text{best},i}) \leftarrow -\infty$ 
6: end for
7:  $\vec{p}_{\text{best},g} \leftarrow \mathbf{x}_0$ ,  $J(\vec{p}_{\text{best},g}) \leftarrow -\infty$ 
8: for  $k = 0$  to  $K - 1$  do
9:    $s \leftarrow \frac{k}{K - 1}$ 
10:   $w \leftarrow w_{\max} - (w_{\max} - w_{\min})s$ 
11:   $c_1 \leftarrow c_{1,\max} - (c_{1,\max} - c_{1,\min})s$ 
12:   $c_2 \leftarrow c_{2,\min} + (c_{2,\max} - c_{2,\min})s$ 
13:  for  $i = 1$  to  $N_p$  do
14:    Set RNG seed =  $42 + k \cdot N_{\text{threads}} + \text{threadID}$ 
15:    Evaluate objective:  $J_i \leftarrow J_c(T(\mathbf{x}_i))$ 
16:    if  $J_i > J(\vec{p}_{\text{best},i})$  then
17:       $\vec{p}_{\text{best},i} \leftarrow \mathbf{x}_i$ 
18:       $J(\vec{p}_{\text{best},i}) \leftarrow J_i$ 
19:    end if
20:    if  $J_i > J(\vec{p}_{\text{best},g})$  then
21:       $\vec{p}_{\text{best},g} \leftarrow \mathbf{x}_i$ 
22:       $J(\vec{p}_{\text{best},g}) \leftarrow J_i$ 
23:    end if
24:    for  $j = 1$  to 6 do
25:      Sample  $r_1, r_2 \sim \mathcal{U}(0, 1)$ 
26:       $v_{i,j} \leftarrow w v_{i,j} + c_1 r_1 (p_{\text{best},i,j} - x_{i,j}) + c_2 r_2 (p_{\text{best},g,j} - x_{i,j})$ 
27:      if  $j \leq 3$  then
28:         $v_{i,j} \leftarrow \text{clip}(v_{i,j}, -v_{\max}^t, v_{\max}^t)$ 
29:      else
30:         $v_{i,j} \leftarrow \text{clip}(v_{i,j}, -v_{\max}^r, v_{\max}^r)$ 
31:      end if
32:    end for
33:    Update position:  $\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{v}_i$ 
34:  end for
35: end for
36: return  $\vec{p}_{\text{best},g}$  and  $T(\vec{p}_{\text{best},g})$ 
```

4. RESULT AND DISCUSSION

In this chapter, the dataset used is described, followed by an overview of the tools and libraries employed in the implementation. Finally, the experimental results are presented and discussed in detail.

4.1 Dataset

In this thesis, the KITTI Vision Benchmark Suite [9] was utilized for training and evaluation. The KITTI dataset is one of the most widely used benchmark datasets in autonomous driving research, providing synchronized sensor data captured from a moving vehicle in urban, rural, and highway environments.

The KITTI raw dataset consists of recordings collected using a sensor-equipped vehicle in Karlsruhe, Germany. The data acquisition platform includes the following sensors:

- Stereo Camera System: Two PointGrey Flea2 color cameras (FL2-14S3C-C) and two PointGrey Flea2 grayscale cameras (FL2-14S3M-C) with a resolution of 1392×512 pixels, capturing images at 10 Hz.
- LiDAR: Velodyne HDL-64E rotating 3D laser scanner with 64 beams, providing approximately 100,000 points per frame at 10 Hz with a range of up to 120 meters.
- GPS/IMU: OXTS RT3003 inertial and GPS navigation system providing accurate localization and ground truth pose information at 100 Hz.

The KITTI raw dataset contains approximately 6 hours of recorded driving data. Table 4.1 summarizes the key statistics of the dataset used in this study.

4.2 Tools and Libraries

This section describes the software tools, programming languages, and libraries used throughout the development and implementation of the proposed system.

Table 4.1 : KITTI raw dataset statistics.

Property	Value
Total sequences	61
Total frames	~43,000
Image resolution	1392 × 512 pixels
LiDAR points per frame	~100,000
Sensor frequency	10 Hz
Recording duration	~6 hours
Environment types	Urban, rural, highway

The Robot Operating System (ROS) [23] was employed as the primary middleware framework for this thesis. ROS provides a flexible and modular architecture for developing robotic applications. ROS Noetic distribution running on Ubuntu 20.04 LTS was used as the development environment. Two primary programming languages were utilized in this work:

C++ was chosen as the main implementation language due to its high performance and low-level memory control, which are essential for real-time robotic applications. The C++14 standard was used, leveraging features such as smart pointers, lambda expressions, and the Standard Template Library (STL) for efficient data structure management.

Python was employed for rapid prototyping, data preprocessing, and visualization tasks. Key Python libraries used include NumPy for numerical computations, Matplotlib for generating plots, and various ROS Python bindings for node development and testing.

The Point Cloud Library (PCL) [24] was extensively used for 3D point cloud processing. PCL is an open-source library providing state-of-the-art algorithms for:

- Filtering: Downsampling (VoxelGrid), outlier removal (StatisticalOutlierRemoval, RadiusOutlierRemoval), and passthrough filtering.
- Segmentation: Plane segmentation using RANSAC, Euclidean cluster extraction.
- Registration: Iterative Closest Point (ICP) and Normal Distributions Transform (NDT) for point cloud alignment.
- Feature Extraction: Normal estimation and geometric descriptors.
- I/O Operations: Reading and writing PCD and PLY file formats.

PCL version 1.10 was used in conjunction with ROS Noetic, enabling seamless integration through the `pcl_ros` package [25]. OpenCV (Open Source Computer Vision Library) [18] was utilized for image processing and computer vision tasks.

OpenMP (Open Multi-Processing) [26] was employed for parallelizing computationally intensive operations. OpenMP provides a portable, shared-memory parallel programming API through compiler directives, enabling:

- Loop Parallelization: Distributing loop iterations across multiple CPU cores using `#pragma omp parallel` for directives.
- Task Parallelism: Concurrent execution of independent code sections.
- Reduction Operations: Efficient parallel aggregation of results.

OpenMP was particularly useful for accelerating point cloud processing operations such as nearest neighbor searches, normal estimation, and filtering, resulting in significant performance improvements on multi-core systems.

Table 4.2 summarizes the software environment used in this thesis.

Table 4.2 : Development environment specifications.

Component	Version/Specification
Operating System	Ubuntu 20.04 LTS
ROS Distribution	Noetic Ninjemys
C++ Standard	C++14
Python Version	3.8
PCL Version	1.10
OpenCV Version	4.2
OpenMP Version	4.5
CMake Version	3.16
Compiler	GCC 9.4.0

4.3 Experimental Results

This section presents the experimental results obtained from evaluating the proposed system on the KITTI dataset. The performance metrics, qualitative results, and comparative analysis with existing methods are discussed in detail.

4.3.1 Time performance analysis

This section explains the impact of ROI selection on the speed of LiDAR-camera calibration. Due to the necessity of human interaction and user dependency in manual ROI selection, automatic ROI selection is compared with operations executed on all images and projected point clouds without any ROI selection. To prevent a rise in the number of tests, perturbations were exclusively applied and evaluated on the translation (XYZ) axes.

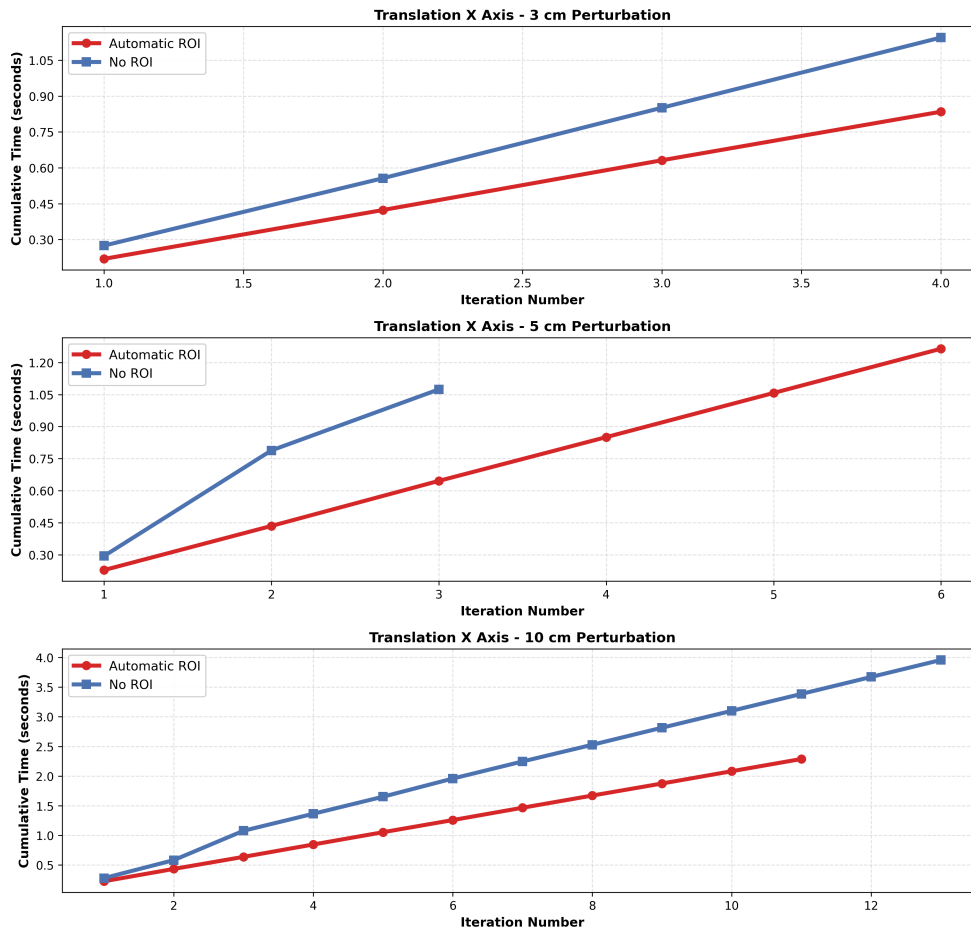


Figure 4.1 : Cumulative computation time comparison of LiDAR-camera calibration with automatic ROI selection and without ROI selection under increasing translation perturbations along the X-axis (3 cm, 5 cm, and 10 cm).

Figures 4.1, 4.2, and 4.3 illustrate the cumulative computation time for LiDAR-camera calibration with and without automatic ROI selection under varying translation perturbations along the X, Y, and Z axes. Across all translation axes and perturbation

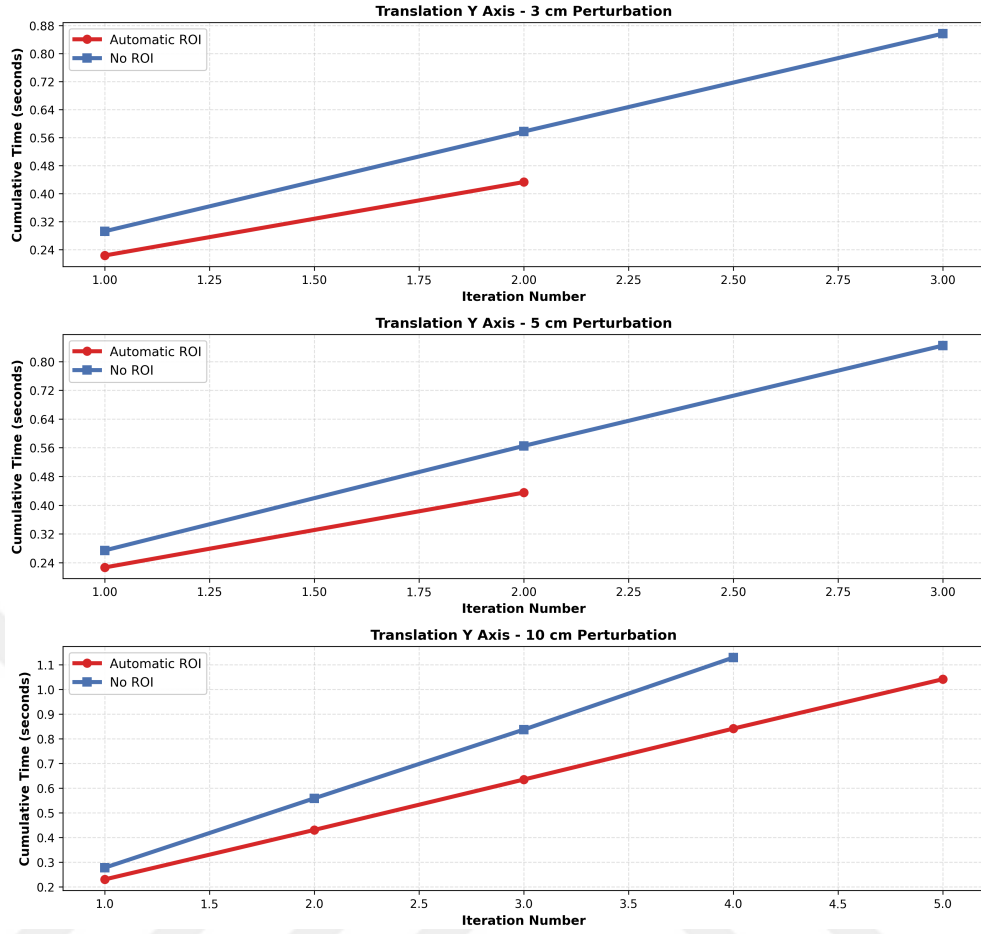


Figure 4.2 : Cumulative computation time comparison of LiDAR–camera calibration with automatic ROI selection and without ROI selection under increasing translation perturbations along the Y-axis (3 cm, 5 cm, and 10 cm).

magnitudes, the results consistently demonstrate that automatic ROI selection leads to a notable reduction in cumulative computation time compared to processing the full image and projected point cloud without ROI constraints. As the perturbation magnitude increases, the computational burden of the no-ROI approach grows more rapidly due to the expanded search space and increased number of iterations, resulting in a steeper time accumulation. In contrast, the automatic ROI-based method maintains a more controlled growth in computation time by restricting the optimization to geometrically informative regions, thereby reducing per-iteration processing cost. It should be noted that the time performance of the automatic ROI approach is directly influenced by both the number and the spatial extent of the selected ROIs. In the presented experiments, calibration was performed using six automatically detected ROI regions per frame, where fewer and more compact ROIs would further reduce the overall computation time.

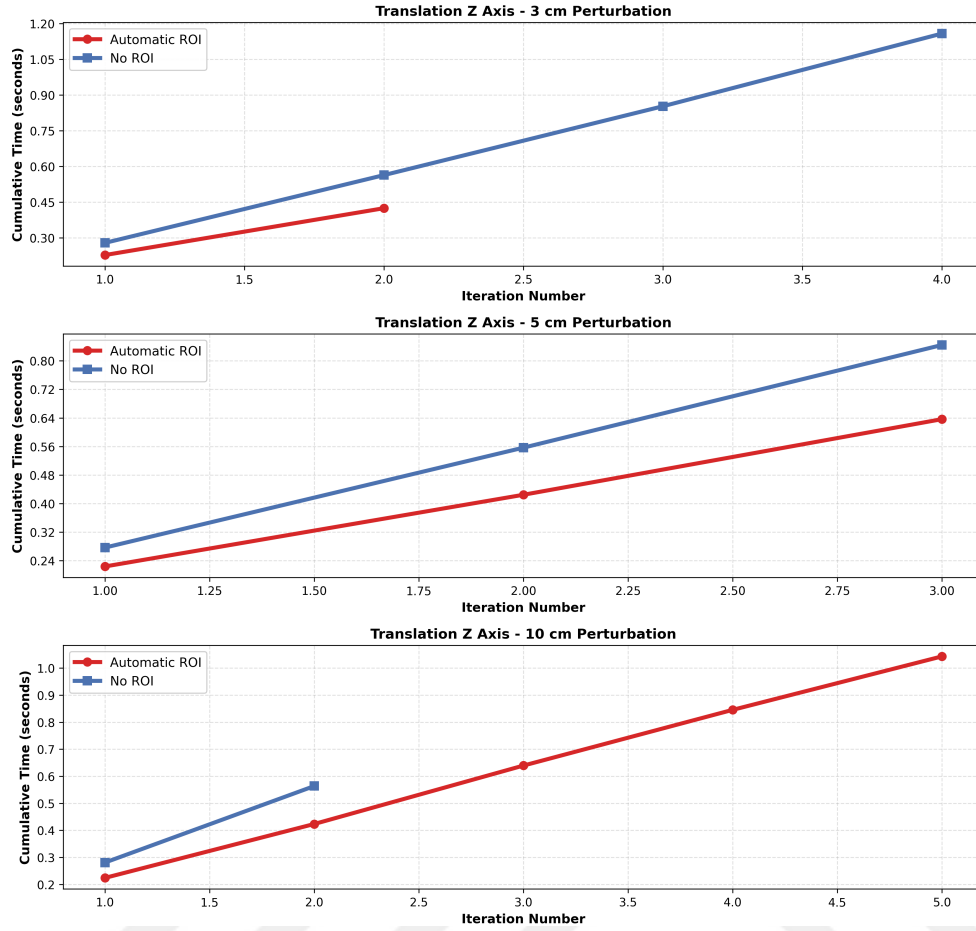


Figure 4.3 : Cumulative computation time comparison of LiDAR–camera calibration with automatic ROI selection and without ROI selection under increasing translation perturbations along the Z-axis (3 cm, 5 cm, and 10 cm).

This dependency highlights that the proposed method offers a flexible trade-off between computational efficiency and geometric coverage, while consistently outperforming the no-ROI baseline across all translation axes.

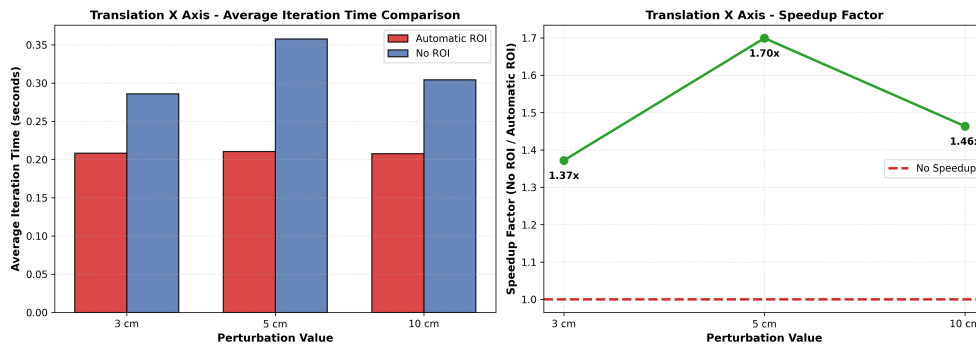


Figure 4.4 : Average iteration time and corresponding speedup factor comparison for LiDAR–camera calibration under translation perturbations along the X-axis (3 cm, 5 cm, and 10 cm), with and without automatic ROI selection.

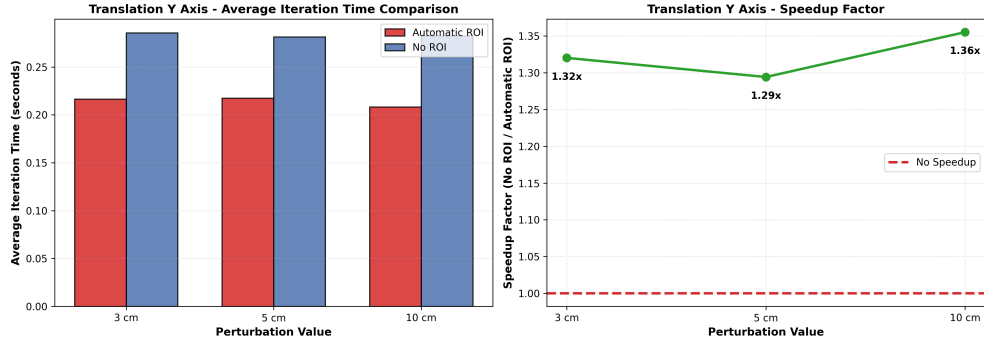


Figure 4.5 : Average iteration time and corresponding speedup factor comparison for LiDAR-camera calibration under translation perturbations along the Y-axis (3 cm, 5 cm, and 10 cm), with and without automatic ROI selection.

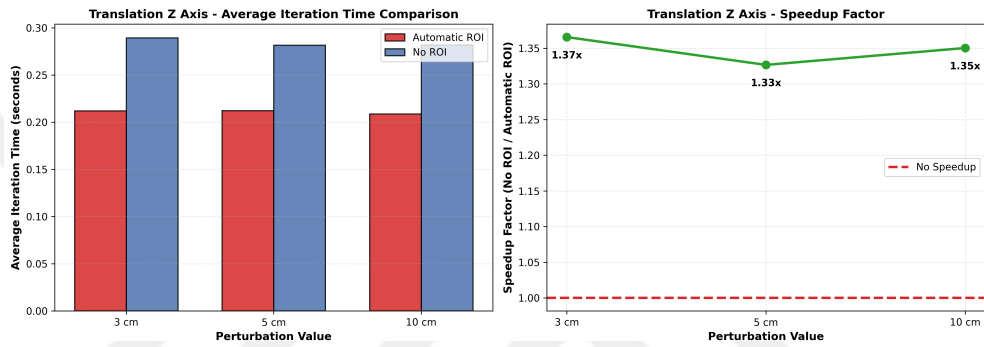


Figure 4.6 : Average iteration time and corresponding speedup factor comparison for LiDAR-camera calibration under translation perturbations along the Z-axis (3 cm, 5 cm, and 10 cm), with and without automatic ROI selection.

Figures 4.4, 4.5, and 4.6 show that across all translation axes (X, Y, and Z), automatic ROI selection consistently improves the calibration speed compared to the no-ROI baseline for all evaluated perturbation magnitudes (3 cm, 5 cm, and 10 cm). Along the X-axis, the speedup factor varies between approximately 1.37 $\times$ and 1.70 $\times$, with the highest acceleration observed at the 5 cm perturbation level. For translations along the Y-axis, the speedup remains more uniform, ranging from approximately 1.29 $\times$ to 1.36 $\times$ across all perturbations. Similarly, along the Z-axis, automatic ROI selection yields a stable speedup between approximately 1.33 $\times$ and 1.37 $\times$.

Figure 4.7 presents an aggregated comparison of average iteration time and overall speedup factors across all translation axes. The results show that automatic ROI selection consistently improves calibration speed for each axis, reducing the average iteration time from approximately 0.31 s to 0.21 s along the X-axis, from 0.28 s to 0.21 s along the Y-axis, and from 0.28 s to 0.21 s along the Z-axis. These reductions correspond to overall speedup factors of approximately 1.51 $\times$, 1.32 $\times$, and 1.35 $\times$ for the X, Y, and Z

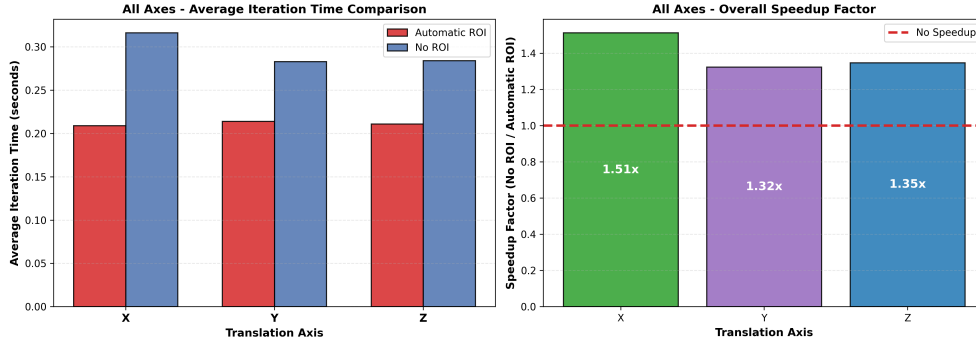


Figure 4.7 : Overall comparison of average iteration time and corresponding speedup factors for LiDAR–camera calibration across all translation axes (X, Y, and Z), with and without automatic ROI selection.

axes, respectively. The higher acceleration observed along the X-axis indicates a greater benefit from ROI-based processing in scenarios with more pronounced horizontal variation, while the more uniform speedup along the Y and Z axes reflects stable performance gains across different spatial directions. Overall, this analysis confirms that automatic ROI selection provides a consistent and axis-independent improvement in calibration speed, yielding a clear advantage over the no-ROI baseline when averaged across multiple perturbation levels.

4.3.2 Calibration methods comparison

This section presents a comparative analysis of the proposed LiDAR–camera calibration method against existing state-of-the-art approaches. The evaluation focuses on calibration accuracy and robustness to perturbations.

Figures 4.8, 4.9, and 4.10 illustrate the overall comparison of translation calibration errors along the X, Y, and Z axes for different methods under varying perturbation magnitudes. To ensure a geometrically meaningful and coordinate-invariant evaluation of rotational accuracy, the rotation total error is defined as the geodesic distance on the special orthogonal group $SO(3)$ between the estimated and ground-truth rotation matrices. This metric corresponds to the minimum rotation angle required to align the estimated orientation with the reference one and avoids the singularities and axis dependency associated with Euler-angle-based error representations. Given two rotation matrices $\mathbf{R}_{\text{est}}$ and $\mathbf{R}_{\text{gt}}$, the rotation error is computed as the angular distance of the relative rotation on the $SO(3)$ manifold.

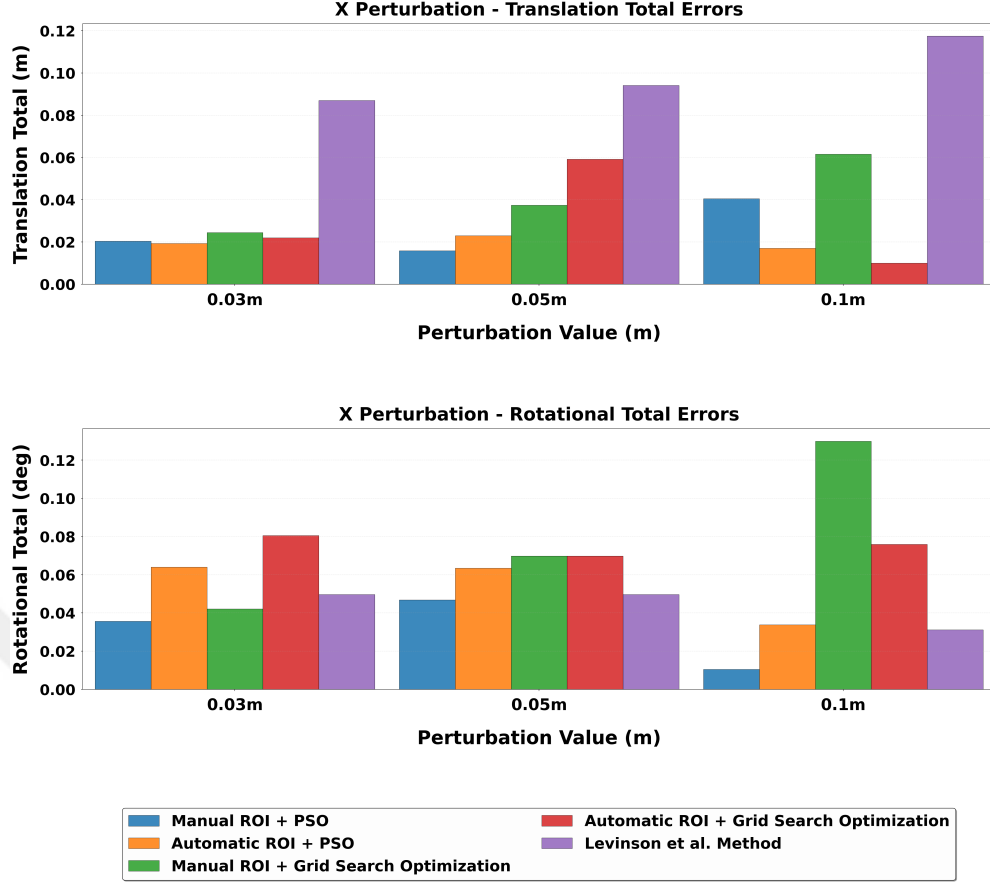


Figure 4.8 : Comparison of translation and rotation total errors with respect to the ground truth after applying X-axis perturbations of 0.03m, 0.05m, and 0.1m followed by extrinsic recalibration using different methods.

In contrast, the translation total error is quantified using the Euclidean norm of the translation difference vector, which provides a direct and physically interpretable measure of positional discrepancy in three-dimensional space. For an estimated translation vector $\mathbf{t}_{\text{est}}$ and the corresponding ground-truth translation $\mathbf{t}_{\text{gt}}$, the translation error is computed as $\|\mathbf{t}_{\text{est}} - \mathbf{t}_{\text{gt}}\|_2$, capturing the overall magnitude of the translation error independently of its direction.

Under X-axis translation perturbations, the translation total errors exhibit clear differences among the evaluated methods. For the smallest perturbation of 0.03 m, the lowest translation error is achieved by Automatic ROI + PSO (0.0194 m), closely followed by Manual ROI + PSO (0.0204 m) and Automatic ROI + Grid Search (0.0220 m). In contrast, the Levinson et al. method produces a substantially larger residual of 0.0870 m, highlighting its lower robustness in this setting. At 0.05 m, the best translation accuracy is obtained with Manual ROI + PSO (0.0159 m), whereas

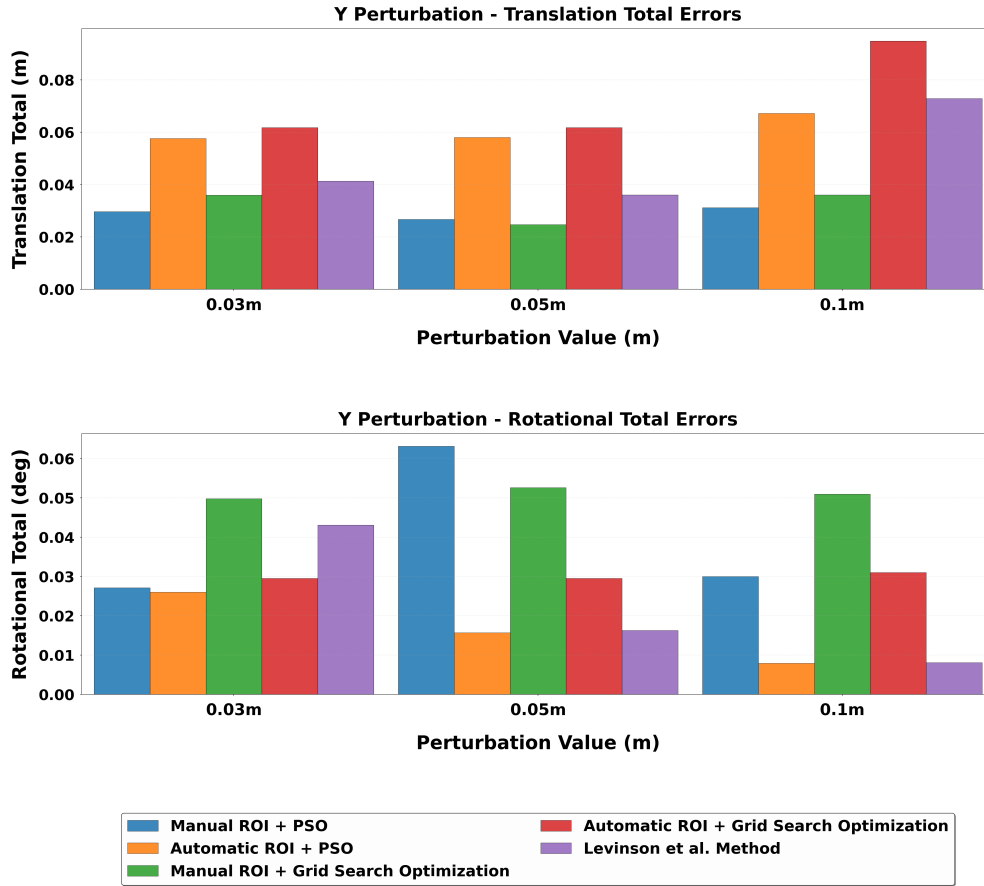


Figure 4.9 : Comparison of translation and rotation total errors with respect to the ground truth after applying Y-axis perturbations of 0.03m, 0.05m, and 0.1m followed by extrinsic recalibration using different methods.

grid-search-based approaches, particularly Automatic ROI + Grid Search, show a noticeable increase (0.0593 m). For the largest perturbation of 0.10 m, Automatic ROI + Grid Search achieves the lowest translation residual (0.0100 m), with Automatic ROI + PSO remaining competitive (0.0171 m). Meanwhile, Manual ROI + PSO degrades to 0.0405 m, and Levinson et al. again yields the highest error (0.1175 m).

The rotational behavior differs from the translation trend. At 0.03 m, the smallest rotation residual is achieved by Manual ROI + PSO (0.0357°), while automatic ROI variants produce higher values (0.0640° for Automatic ROI + PSO and 0.0805° for Automatic ROI + Grid Search). A similar pattern holds at 0.05 m, where Manual ROI + PSO remains the most accurate (0.0469°). At 0.10 m, Manual ROI + PSO clearly outperforms all other methods with a very low rotation residual of 0.0105° , whereas methods that minimize translation error more aggressively (e.g., Automatic ROI +

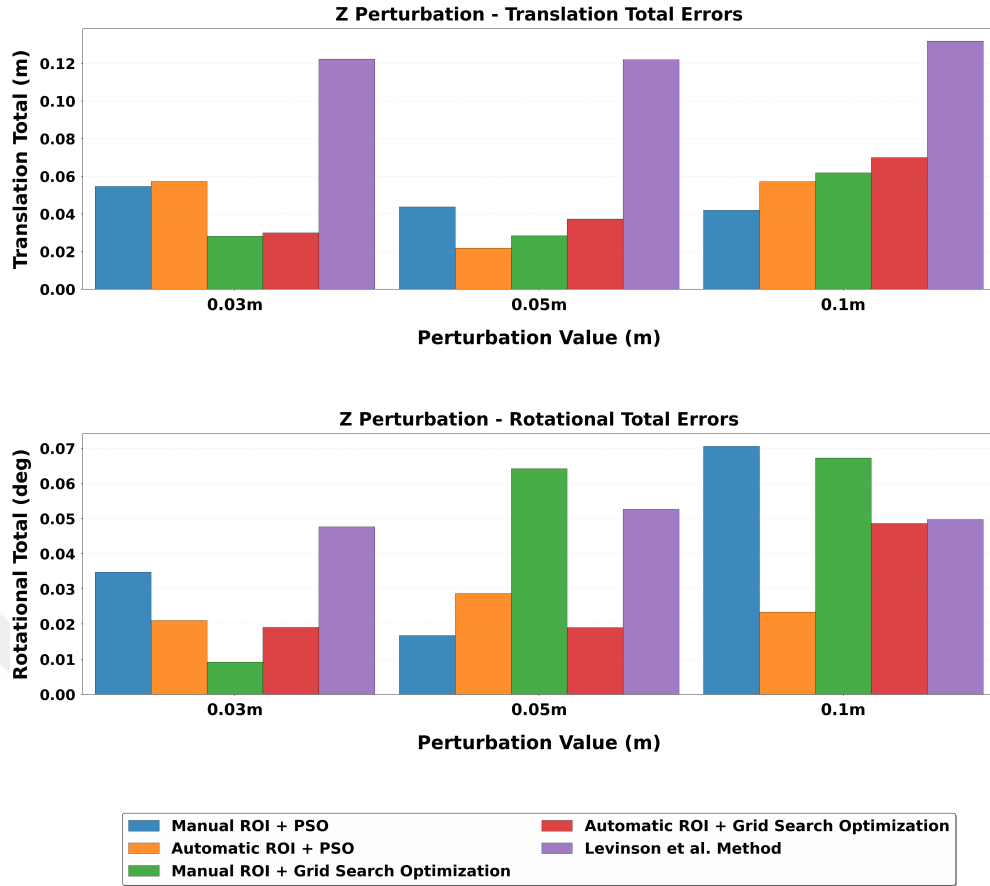


Figure 4.10 : Comparison of translation and rotation total errors with respect to the ground truth after applying Z-axis perturbations of 0.03m, 0.05m, and 0.1m followed by extrinsic recalibration using different methods.

Grid Search) exhibit higher rotational total errors (0.0760°). This demonstrates a clear translation–rotation trade-off along the X-axis.

For Y-axis perturbations, manual ROI-based methods consistently yield lower translation errors. At 0.03 m, Manual ROI + PSO achieves the lowest translation residual (0.0297 m), outperforming Manual ROI + Grid Search (0.0360 m) and Levinson et al. (0.0413 m). Automatic ROI methods are noticeably higher at this level (0.0576 m for PSO and 0.0617 m for Grid Search). At 0.05 m, Manual ROI + Grid Search slightly outperforms other approaches with 0.0247 m, closely followed by Manual ROI + PSO (0.0267 m). At 0.10 m, Manual ROI + PSO again yields the lowest translation residual (0.0312 m), while automatic ROI approaches show a marked increase, particularly Automatic ROI + Grid Search (0.0948 m).

In contrast to translation, rotational accuracy along the Y-axis strongly favors Automatic ROI + PSO. At 0.03 m, it produces a rotation residual of 0.0260° , comparable to

Manual ROI + PSO (0.0271°). At 0.05 m, Automatic ROI + PSO clearly outperforms all other methods with 0.0157° , while Manual ROI + PSO increases significantly to 0.0631° . At 0.10 m, the best rotation result is again achieved by Automatic ROI + PSO (0.00793°), marginally outperforming Levinson et al. (0.00804°). These results indicate that, for Y-axis perturbations, automatic ROI selection combined with PSO is particularly effective in minimizing rotational misalignment.

For Z-axis perturbations, translation total errors are generally higher, and the relative ranking of methods varies with perturbation magnitude. At 0.03 m, the lowest translation residual is obtained with Manual ROI + Grid Search (0.0282 m), followed closely by Automatic ROI + Grid Search (0.0300 m). PSO-based methods are less accurate at this level (0.0546 m for Manual ROI + PSO and 0.0573 m for Automatic ROI + PSO). At 0.05 m, Automatic ROI + PSO achieves the best translation accuracy (0.0220 m), outperforming both grid-search-based approaches. At 0.10 m, Manual ROI + PSO yields the lowest translation residual (0.0420 m), whereas grid-search-based methods increase substantially (0.0618 m and 0.0699 m). Across all Z-axis perturbations, Levinson et al. consistently produces the largest translation errors (0.1223–0.1318 m).

Rotational behavior along the Z-axis shows notable variability. At 0.03 m, the best rotation accuracy is achieved by Manual ROI + Grid Search (0.00921°), while PSO-based methods are higher. At 0.05 m, Manual ROI + PSO becomes the best-performing method with 0.0167° , whereas Manual ROI + Grid Search increases sharply to 0.0642° . At 0.10 m, the lowest rotation residual is obtained with Automatic ROI + PSO (0.0234°), while Manual ROI + PSO degrades significantly (0.0706°). This confirms that optimal performance along the Z-axis is highly dependent on both the perturbation magnitude and the chosen optimization strategy.

Figures 4.11, 4.12, and 4.13 present the overall comparison of calibration errors under roll, pitch, and yaw perturbations for different methods.

Under pitch perturbations, the translation total errors across all evaluated methods range approximately between 0.01 m and 0.13 m. At the smallest perturbation level of 0.03° , the proposed Automatic ROI + PSO approach achieves the lowest translation error of

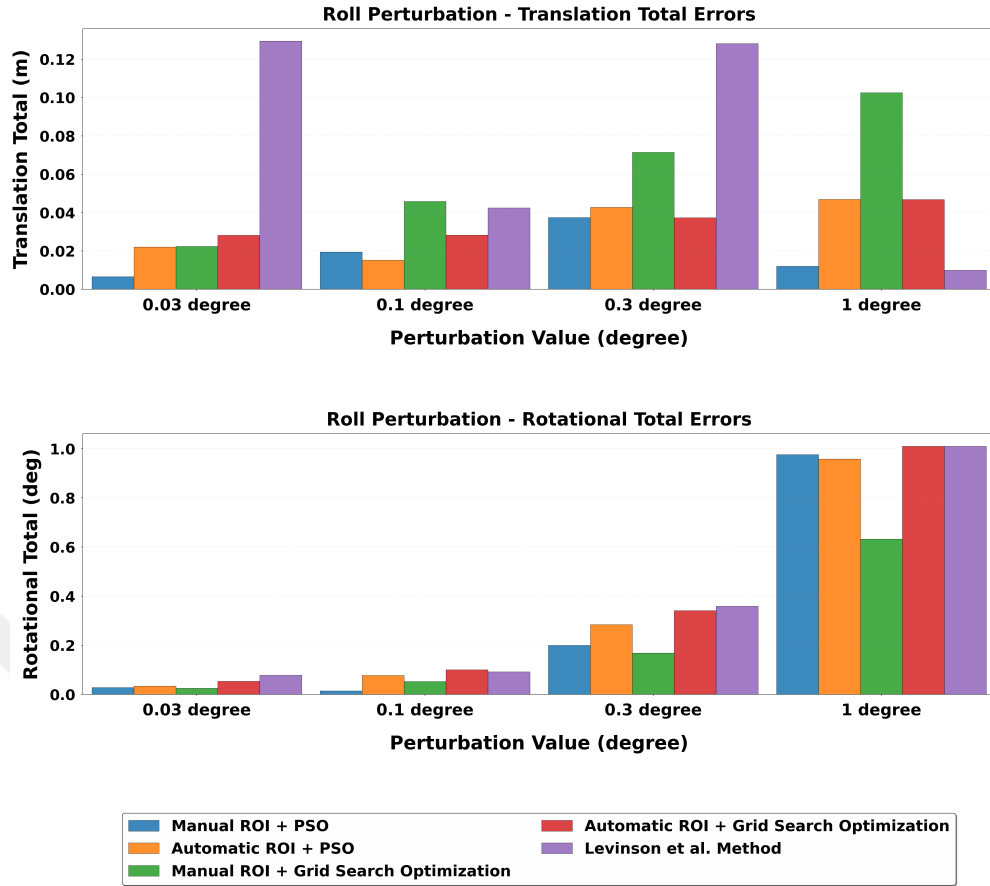


Figure 4.11 : Comparison of translation and rotation total errors with respect to the ground truth after applying Roll-axis perturbations of 0.03° , 0.1° , 0.3° , and 1° followed by extrinsic recalibration using different methods.

approximately 0.012 m, outperforming Manual ROI + PSO (approximately 0.05 m) and the Levinson et al. method (approximately 0.042 m) by a factor of nearly four.

At a moderate perturbation level of 0.1° , the lowest translation error is obtained using Manual ROI + Grid Search Optimization (approximately 0.044 m), suggesting that manually defined ROIs can better constrain the search space when pitch deviations increase. For larger perturbations of 0.3° , Manual ROI + PSO yields the best performance with a translation error of approximately 0.043 m, while the Levinson et al. method degrades significantly, reaching approximately 0.13 m. At the largest perturbation level of 1° , all methods exhibit increased translation errors; however, Automatic ROI + PSO maintains a comparatively moderate error level of approximately 0.08 m, whereas the Levinson et al. method remains the least stable. Overall, these results demonstrate that Automatic ROI + PSO is particularly robust for small pitch

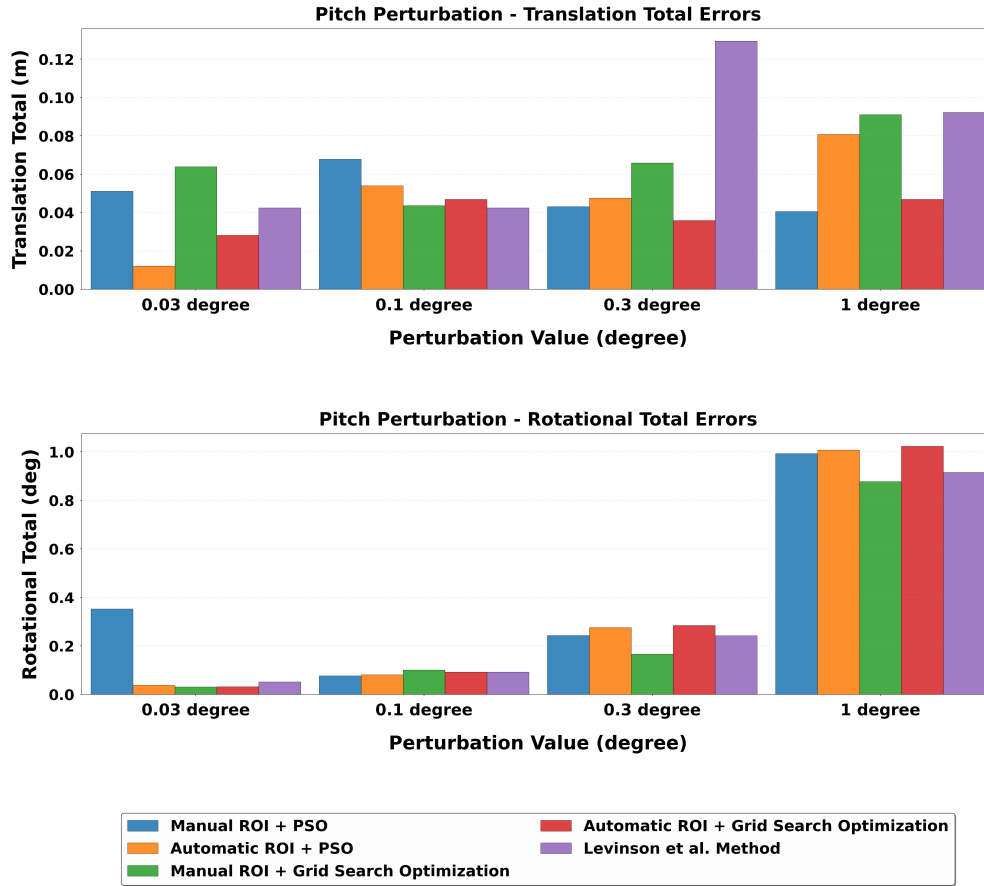


Figure 4.12 : Comparison of translation and rotation total errors with respect to the ground truth after applying Pitch-axis perturbations of 0.03° , 0.1° , 0.3° , and 1° followed by extrinsic recalibration using different methods.

perturbations, while manual ROI-based methods become more competitive as the perturbation magnitude increases.

Rotational total errors under pitch perturbations increase monotonically with perturbation magnitude for all methods, ranging from approximately 0.03° to values exceeding 1° . At 0.03° , Automatic ROI + PSO and Manual ROI + Grid Search Optimization achieve very low rotational errors below 0.05° , whereas Manual ROI + PSO exhibits a noticeably higher error of approximately 0.35° .

As the perturbation increases to 0.1° , all methods converge to a narrow error band between approximately 0.07° and 0.10° , indicating similar rotational recovery performance. At 0.3° , Manual ROI + Grid Search Optimization yields the lowest rotational error of approximately 0.17° , while Automatic ROI + PSO and Automatic ROI + Grid Search Optimization exhibit higher values around 0.27° – 0.29° . At the largest perturbation level of 1° , rotational errors approach 1° for all approaches,

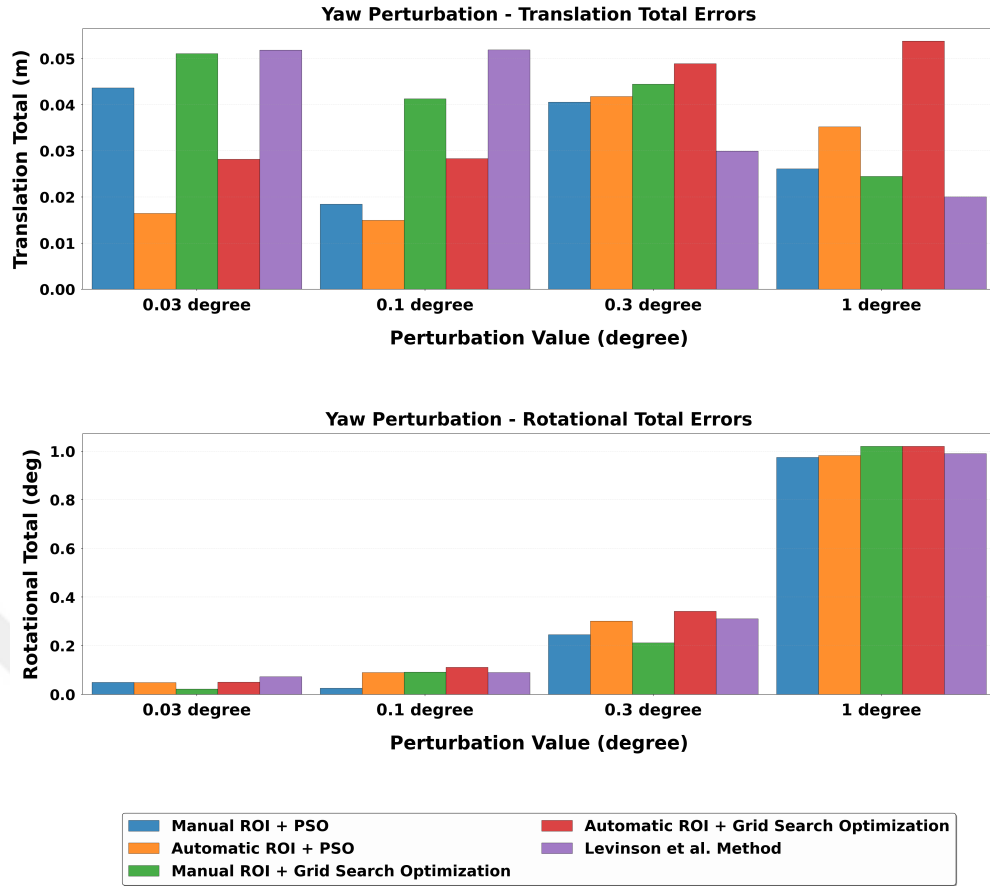


Figure 4.13 : Comparison of translation and rotation total errors with respect to the ground truth after applying Yaw-axis perturbations of 0.03° , 0.1° , 0.3° , and 1° followed by extrinsic recalibration using different methods.

reflecting the intrinsic difficulty of recovering large pitch misalignments from a single frame. Nevertheless, Automatic ROI + PSO remains comparable to manual methods and does not exhibit catastrophic divergence, indicating stable convergence behavior even under severe perturbations.

For roll perturbations, translation total errors generally lie between approximately 0.006 m and 0.13 m. At a perturbation level of 0.03° , Manual ROI + PSO achieves the lowest translation error of approximately 0.006 m, while the Levinson et al. method shows a significantly higher error of approximately 0.13 m, indicating poor sensitivity to small roll deviations.

At 0.1° , Automatic ROI + PSO produces the lowest translation error of approximately 0.015 m, outperforming Manual ROI + Grid Search Optimization (approximately 0.046 m). At 0.3° , Manual ROI + PSO again yields the best performance with a translation error of approximately 0.038 m, whereas the Levinson et al. method remains

unstable with errors exceeding 0.12 m. At the largest perturbation level of 1° , Automatic ROI + PSO maintains a relatively low translation error of approximately 0.047 m, while Manual ROI + Grid Search Optimization deteriorates to values above 0.10 m. These results indicate that Automatic ROI selection combined with PSO provides increased robustness to roll-induced translation coupling, particularly for moderate and large perturbations.

Rotational total errors under roll perturbations increase consistently with perturbation magnitude. At small perturbations of 0.03° and 0.1° , all methods achieve low rotational errors below 0.1° , with Manual ROI + PSO and Automatic ROI + PSO exhibiting the most stable behavior. At 0.3° , Manual ROI + Grid Search Optimization achieves the lowest rotational error of approximately 0.17° , while Automatic ROI + PSO and Automatic ROI + Grid Search Optimization reach moderately higher values of approximately 0.28° – 0.34° .

At the largest perturbation of 1° , rotational errors approach or exceed 1° for most methods, indicating that roll perturbations strongly couple into rotational recovery errors. Nevertheless, Automatic ROI + PSO remains comparable to manual baselines, confirming that the automatic ROI strategy does not introduce additional instability under large roll perturbations.

Translation total errors under yaw perturbations remain relatively constrained compared to pitch and roll perturbations, ranging from approximately 0.015 m to 0.055 m. At 0.03° , Automatic ROI + PSO yields the lowest translation error of approximately 0.016 m, significantly outperforming Manual ROI + PSO, which reaches approximately 0.044 m. A similar trend is observed at 0.1° , where Automatic ROI + PSO maintains a translation error of approximately 0.015 m, compared to values exceeding 0.04 m for Manual ROI + Grid Search Optimization.

At 0.3° , translation errors across all methods converge to a narrow band between approximately 0.03 m and 0.05 m, suggesting reduced sensitivity of translation estimation to yaw perturbations. At the largest perturbation of 1° , Manual ROI + Grid Search Optimization produces the lowest translation error of approximately 0.025 m, while Automatic ROI + Grid Search Optimization degrades to values above 0.05 m.

Overall, Automatic ROI + PSO demonstrates consistently competitive translation accuracy for yaw perturbations, particularly at small and moderate magnitudes.

Rotational total errors for yaw perturbations increase sharply with perturbation magnitude for all evaluated methods. At 0.03° and 0.1° , rotational errors remain below 0.1° , with Manual ROI + PSO achieving the lowest values. At 0.3° , Manual ROI + Grid Search Optimization yields the best performance with a rotational error of approximately 0.21° , while Automatic ROI + PSO and Automatic ROI + Grid Search Optimization reach approximately 0.30° – 0.34° .

At the largest perturbation level of 1° , all methods converge to rotational errors close to 1° , indicating that yaw misalignments dominate the rotational recovery limits in single-frame calibration. Importantly, the proposed Automatic ROI + PSO method remains comparable to both manual and baseline approaches and does not exhibit excessive divergence, confirming its stability across the full range of yaw perturbations.

Calibration success score computation

To quantitatively compare different calibration strategies under varying perturbation magnitudes, a *Calibration Success Score* is introduced to summarize the overall error behavior of each method across all perturbation levels. Let the perturbation sets be defined as $p_i^{\text{rot}} \in \{0.03^\circ, 0.1^\circ, 0.3^\circ, 1^\circ\}$ for rotational perturbations and $p_i^{\text{trans}} \in \{0.03\text{ m}, 0.05\text{ m}, 0.1\text{ m}\}$ for translational perturbations, with $i = 1, \dots, N$. Let $m \in \mathcal{M}$ denote a calibration method. For each perturbation level p_i^{trans} , method m yields a translation total error $T_m(p_i^{\text{trans}})$ (in meters), and for each p_i^{rot} , a rotation total error $R_m(p_i^{\text{rot}})$ (in degrees), where lower values correspond to higher calibration accuracy.

Step 1: Error-to-Perturbation Ratio: For scale-aware comparison, the error at each perturbation level is normalized by the corresponding perturbation magnitude:

$$\rho_m(p_i) = \begin{cases} \frac{T_m(p_i^{\text{trans}})}{p_i^{\text{trans}}}, & \text{translation errors,} \\ \frac{R_m(p_i^{\text{rot}})}{p_i^{\text{rot}}}, & \text{rotation errors.} \end{cases} \quad (4.1)$$

A smaller ratio $\rho_m(p_i)$ indicates lower sensitivity of method m to the applied perturbation.

Step 2: Global Min–Max Normalization: To ensure fair comparison across all methods, the ratio values corresponding to the same error type are pooled:

$$\mathcal{R} = \bigcup_{m \in \mathcal{M}} \{\rho_m(p_i)\}_{i=1}^N. \quad (4.2)$$

Let $\rho_{\min} = \min(\mathcal{R})$ and $\rho_{\max} = \max(\mathcal{R})$. Each ratio is then normalized using global min–max scaling:

$$\tilde{\rho}_m(p_i) = \begin{cases} \frac{\rho_m(p_i) - \rho_{\min}}{\rho_{\max} - \rho_{\min}}, & \text{if } \rho_{\max} > \rho_{\min} \\ 0, & \text{if } \rho_{\max} = \rho_{\min} \end{cases} \quad (4.3)$$

where $\tilde{\rho}_m(p_i) \in [0, 1]$, and smaller values correspond to better relative performance.

Step 3: Calibration Success Score: The Calibration Success Score for method m is computed as the mean of the normalized ratios over all perturbation levels:

$$S_m = \frac{1}{N} \sum_{i=1}^N \tilde{\rho}_m(p_i), \quad (4.4)$$

where N is the number of perturbation levels and $\tilde{\rho}_m(p_i)$ is the globally min–max normalized error-to-perturbation ratio for method m at perturbation p_i as defined in Equation (4.3).

Since lower normalized ratios indicate better calibration performance, the optimal method is determined by

$$m^* = \arg \min_{m \in \mathcal{M}} S_m, \quad (4.5)$$

where $\mathcal{M}$ is the set of all evaluated methods.

Accordingly, a lower Calibration Success Score reflects a method's ability to consistently achieve smaller errors relative to the applied perturbation magnitude across all tested levels.

Table 4.3 summarizes the Calibration Success Scores for different calibration methods based on their X-axis translation errors. The proposed Automatic ROI + PSO method achieves the lowest score of 0.116, indicating superior performance in maintaining low translation errors relative to perturbation magnitudes. Manual ROI + PSO follows closely with a score of 0.131, demonstrating that both ROI-based strategies effectively enhance calibration robustness. In contrast, grid-search-based methods yield higher

Table 4.3 : Calibration Success Score for X-Axis Translation Error.

Method	Calibration Success Score
Automatic ROI + PSO	0.116
Manual ROI + PSO	0.131
Automatic ROI + Grid Search Optimization	0.205
Manual ROI + Grid Search Optimization	0.224
Levinson et al. Method	0.673

scores (0.205 and 0.224), reflecting their comparatively lower accuracy under varying perturbations. The Levinson et al. method exhibits the highest score of 0.673, highlighting its limited effectiveness in this evaluation.

Table 4.4 : Calibration Success Score for Y-Axis Translation Error.

Method	Calibration Success Score
Manual ROI + PSO	0.172
Manual ROI + Grid Search Optimization	0.213
Levinson et al. Method	0.361
Automatic ROI + PSO	0.538
Automatic ROI + Grid Search Optimization	0.631

Table 4.4 presents the Calibration Success Scores for various calibration methods based on their Y-axis translation errors. The Manual ROI + PSO method achieves the lowest score of 0.172, indicating its effectiveness in minimizing translation errors relative to perturbation magnitudes. Manual ROI + Grid Search Optimization follows with a score of 0.213, demonstrating that manual ROI selection consistently enhances calibration accuracy. The Levinson et al. method attains a moderate score of 0.361, while both automatic ROI-based methods yield higher scores (0.538 and 0.631), suggesting that manual ROI selection is particularly beneficial for Y-axis calibration robustness.

Table 4.5 : Calibration Success Score for Z-Axis Translation Error.

Method	Calibration Success Score
Manual ROI + Grid Search Optimization	0.079
Automatic ROI + Grid Search Optimization	0.108
Automatic ROI + PSO	0.151
Manual ROI + PSO	0.169
Levinson et al. Method	0.599

Table 4.5 summarizes the Calibration Success Scores for different calibration methods based on their Z-axis translation errors. The Manual ROI + Grid Search Optimization method achieves the lowest score of 0.079, indicating its superior performance in minimizing translation errors relative to perturbation magnitudes. Automatic ROI + Grid Search Optimization follows with a score of 0.108, demonstrating that grid-search-based approaches benefit from both manual and automatic ROI selection. The proposed Automatic ROI + PSO method attains a competitive score of 0.151, while Manual ROI + PSO scores 0.169. The Levinson et al. method exhibits the highest score of 0.599, highlighting its limited effectiveness in this evaluation.

Table 4.6 : Calibration Success Score for Roll-Axis Rotation Error.

Method	Calibration Success Score
Manual ROI + Grid Search Optimization	0.202
Manual ROI + PSO	0.219
Automatic ROI + PSO	0.327
Automatic ROI + Grid Search Optimization	0.443
Levinson et al. Method	0.524

Table 4.6 presents the Calibration Success Scores for various calibration methods based on their roll-axis rotation errors. The Manual ROI + Grid Search Optimization method achieves the lowest score of 0.202, indicating its effectiveness in minimizing rotational errors relative to perturbation magnitudes. Manual ROI + PSO follows closely with a score of 0.219, demonstrating that both ROI-based strategies enhance calibration accuracy. The proposed Automatic ROI + PSO method attains a moderate score of 0.327, while Automatic ROI + Grid Search Optimization scores 0.443. The Levinson et al. method exhibits the highest score of 0.524, highlighting its limited effectiveness in this evaluation.

Table 4.7 : Calibration Success Score for Pitch-Axis Rotation Error.

Method	Calibration Success Score
Manual ROI + Grid Search Optimization	0.028
Automatic ROI + Grid Search Optimization	0.039
Automatic ROI + PSO	0.039
Levinson et al. Method	0.048
Manual ROI + PSO	0.270

Table 4.7 summarizes the Calibration Success Scores for different calibration methods based on their pitch-axis rotation errors. The Manual ROI + Grid Search Optimization method achieves the lowest score of 0.028, indicating its superior performance in minimizing rotational errors relative to perturbation magnitudes. Both Automatic ROI + Grid Search Optimization and Automatic ROI + PSO attain competitive scores of 0.039, demonstrating that automatic ROI selection effectively enhances calibration accuracy for pitch rotations. The Levinson et al. method scores 0.048, while Manual ROI + PSO exhibits a significantly higher score of 0.270, suggesting that this approach is less effective for pitch-axis calibration in this evaluation.

Table 4.8 : Calibration Success Score for Yaw-Axis Rotation Error.

Method	Calibration Success Score
Manual ROI + Grid Search Optimization	0.271
Manual ROI + PSO	0.309
Automatic ROI + PSO	0.404
Automatic ROI + Grid Search Optimization	0.456
Levinson et al. Method	0.500

Table 4.8 presents the Calibration Success Scores for various calibration methods based on their yaw-axis rotation errors. The Manual ROI + Grid Search Optimization method achieves the lowest score of 0.271, indicating its effectiveness in minimizing rotational errors relative to perturbation magnitudes. Manual ROI + PSO follows with a score of 0.309, demonstrating that both ROI-based strategies enhance calibration accuracy. The proposed Automatic ROI + PSO method attains a moderate score of 0.404, while Automatic ROI + Grid Search Optimization scores 0.456. The Levinson et al. method exhibits the highest score of 0.500, highlighting its limited effectiveness in this evaluation.

Tables 4.9, 4.10, and 4.11 summarize the translation calibration errors for various methods under different perturbation magnitudes along the X, Y, and Z axes.

Tables 4.12, 4.13, and 4.14 present the calibration errors under roll, pitch, and yaw perturbations for the PSO-based method, grid search optimization, and the Levinson et al. baseline method, respectively.

Table 4.9 : Translation calibration errors under different translation perturbation magnitudes using PSO-based methods.

Method	Axis	Perturbation	Translation Total Error (m)	Rotation Total Error (degree)
Manual ROI + PSO	X	0.03 m	0.0204	0.0357
Manual ROI + PSO	X	0.05 m	0.0159	0.0469
Manual ROI + PSO	X	0.10 m	0.0405	0.0105
Automatic ROI + PSO	X	0.03 m	0.0194	0.0640
Automatic ROI + PSO	X	0.05 m	0.0229	0.0634
Automatic ROI + PSO	X	0.10 m	0.0171	0.0338
Manual ROI + PSO	Y	0.03 m	0.0297	0.0271
Manual ROI + PSO	Y	0.05 m	0.0267	0.0631
Manual ROI + PSO	Y	0.10 m	0.0312	0.0300
Automatic ROI + PSO	Y	0.03 m	0.0576	0.0260
Automatic ROI + PSO	Y	0.05 m	0.0581	0.0157
Automatic ROI + PSO	Y	0.10 m	0.0672	0.0079
Manual ROI + PSO	Z	0.03 m	0.0546	0.0348
Manual ROI + PSO	Z	0.05 m	0.0438	0.0167
Manual ROI + PSO	Z	0.10 m	0.0420	0.0706
Automatic ROI + PSO	Z	0.03 m	0.0573	0.0211
Automatic ROI + PSO	Z	0.05 m	0.0220	0.0287
Automatic ROI + PSO	Z	0.10 m	0.0572	0.0234

Table 4.10 : Translation calibration errors under different translation perturbation magnitudes using grid search optimization.

Method	Axis	Perturbation	Translation Total Error (m)	Rotation Total Error (degree)
Manual ROI + Grid Search	X	0.03 m	0.0245	0.0421
Manual ROI + Grid Search	X	0.05 m	0.0374	0.0698
Manual ROI + Grid Search	X	0.10 m	0.0616	0.1299
Automatic ROI + Grid Search	X	0.03 m	0.0220	0.0805
Automatic ROI + Grid Search	X	0.05 m	0.0593	0.0699
Automatic ROI + Grid Search	X	0.10 m	0.0100	0.0760
Manual ROI + Grid Search	Y	0.03 m	0.0360	0.0498
Manual ROI + Grid Search	Y	0.05 m	0.0247	0.0526
Manual ROI + Grid Search	Y	0.10 m	0.0361	0.0510
Automatic ROI + Grid Search	Y	0.03 m	0.0617	0.0295
Automatic ROI + Grid Search	Y	0.05 m	0.0617	0.0295
Automatic ROI + Grid Search	Y	0.10 m	0.0948	0.0310

Table 4.11 : Translation calibration errors using the Levinson et al. baseline method.

Axis	Perturbation	Translation Total Error (m)	Rotation Total Error (degree)
X	0.03 m	0.0870	0.0497
X	0.05 m	0.0942	0.0497
X	0.10 m	0.1175	0.0312
Y	0.03 m	0.0413	0.0431
Y	0.05 m	0.0361	0.0163
Y	0.10 m	0.0728	0.0080
Z	0.03 m	0.1223	0.0477
Z	0.05 m	0.1219	0.0527
Z	0.10 m	0.1318	0.0498

Table 4.12 : Calibration errors under roll, pitch, and yaw perturbations using PSO-based optimization.

Method	Axis	Perturbation	Translation Total Error (m)	Rotation Total Error (degree)
Manual ROI + PSO	Roll	0.03°	0.006538	0.028188
Manual ROI + PSO	Roll	0.10°	0.019464	0.013931
Manual ROI + PSO	Roll	0.30°	0.037466	0.199637
Manual ROI + PSO	Roll	1.00°	0.012023	0.975969
Automatic ROI + PSO	Roll	0.03°	0.022074	0.033386
Automatic ROI + PSO	Roll	0.10°	0.015200	0.076896
Automatic ROI + PSO	Roll	0.30°	0.042803	0.284174
Automatic ROI + PSO	Roll	1.00°	0.046910	0.957762
Manual ROI + PSO	Pitch	0.03°	0.051161	0.352400
Manual ROI + PSO	Pitch	0.10°	0.067826	0.076280
Manual ROI + PSO	Pitch	0.30°	0.043093	0.242930
Manual ROI + PSO	Pitch	1.00°	0.040629	0.992903
Automatic ROI + PSO	Pitch	0.03°	0.012100	0.037255
Automatic ROI + PSO	Pitch	0.10°	0.053997	0.080387
Automatic ROI + PSO	Pitch	0.30°	0.047572	0.275397
Automatic ROI + PSO	Pitch	1.00°	0.080825	1.007051
Manual ROI + PSO	Yaw	0.03°	0.043650	0.049443
Manual ROI + PSO	Yaw	0.10°	0.018430	0.025619
Manual ROI + PSO	Yaw	0.30°	0.040555	0.245730
Manual ROI + PSO	Yaw	1.00°	0.026089	0.974385

Continued on next page

Table 4.12 – Continued from previous page

Method	Axis	Perturbation	Translation Total Error (m)	Rotation Total Error (degree)
Automatic ROI + PSO	Yaw	0.03°	0.016447	0.048935
Automatic ROI + PSO	Yaw	0.10°	0.014980	0.089963
Automatic ROI + PSO	Yaw	0.30°	0.041769	0.301489
Automatic ROI + PSO	Yaw	1.00°	0.035218	0.982335

Table 4.13 : Calibration errors under roll, pitch, and yaw perturbations using grid search optimization.

Method	Axis	Perturbation	Translation Total Error (m)	Rotation Total Error (degree)
Manual ROI + Grid Search	Roll	0.03°	0.022406	0.025000
Manual ROI + Grid Search	Roll	0.10°	0.045782	0.052154
Manual ROI + Grid Search	Roll	0.30°	0.071417	0.168310
Manual ROI + Grid Search	Roll	1.00°	0.102481	0.631955
Automatic ROI + Grid Search	Roll	0.03°	0.028163	0.053288
Automatic ROI + Grid Search	Roll	0.10°	0.028286	0.100259
Automatic ROI + Grid Search	Roll	0.30°	0.037279	0.340910
Automatic ROI + Grid Search	Roll	1.00°	0.046856	1.009807
Manual ROI + Grid Search	Pitch	0.03°	0.063947	0.030646
Manual ROI + Grid Search	Pitch	0.10°	0.043684	0.100280
Manual ROI + Grid Search	Pitch	0.30°	0.065832	0.165919
Manual ROI + Grid Search	Pitch	1.00°	0.091055	0.877002
Automatic ROI + Grid Search	Pitch	0.03°	0.028195	0.031798
Automatic ROI + Grid Search	Pitch	0.10°	0.046854	0.091724
Automatic ROI + Grid Search	Pitch	0.30°	0.035880	0.283884
Automatic ROI + Grid Search	Pitch	1.00°	0.046958	1.022978
Manual ROI + Grid Search	Yaw	0.03°	0.051060	0.021654
Manual ROI + Grid Search	Yaw	0.10°	0.041306	0.091966
Manual ROI + Grid Search	Yaw	0.30°	0.044472	0.211661
Manual ROI + Grid Search	Yaw	1.00°	0.024473	1.020424
Automatic ROI + Grid Search	Yaw	0.03°	0.028197	0.050603
Automatic ROI + Grid Search	Yaw	0.10°	0.028309	0.111666
Automatic ROI + Grid Search	Yaw	0.30°	0.048886	0.342148
Automatic ROI + Grid Search	Yaw	1.00°	0.053767	1.020423

Table 4.14 : Calibration errors under roll, pitch, and yaw perturbations using the Levinson et al. baseline method.

Axis	Perturbation	Translation Total Error (m)	Rotation Total Error (degree)
Roll	0.03°	0.129394	0.078146
Roll	0.10°	0.042428	0.091754
Roll	0.30°	0.128159	0.358824
Roll	1.00°	0.010001	1.010128
Pitch	0.03°	0.042468	0.051530
Pitch	0.10°	0.042426	0.092301
Pitch	0.30°	0.129366	0.242249
Pitch	1.00°	0.092292	0.915223
Yaw	0.03°	0.051855	0.072548
Yaw	0.10°	0.051908	0.090342
Yaw	0.30°	0.029943	0.311062
Yaw	1.00°	0.020028	0.990189



5. CONCLUSIONS AND FUTURE WORKS

This thesis presented a targetless LiDAR-camera extrinsic calibration framework that jointly studies the impact of (i) ROI selection strategy (manual vs. automatic) and (ii) optimization method (greedy hill-climb grid search vs. PSO) under controlled translation and rotation perturbations. In contrast to many prior works that report results for a single configuration or limited perturbations, this study provides a systematic and quantitative comparison across multiple axes and perturbation magnitudes on the KITTI dataset. Rather than introducing a single fixed calibration algorithm, the proposed approach is formulated as a flexible ROI-based calibration framework in which both manual and automatic ROI selection strategies can be combined with different optimization methods and systematically evaluated against a classical targetless baseline.

From a computational perspective, restricting the calibration to edge-rich ROIs provides a clear efficiency advantage over the no-ROI baseline. In the time-performance experiments under translation perturbations, automatic ROI selection reduced the average iteration time from approximately 0.31 s to 0.21 s along the X-axis, from 0.28 s to 0.21 s along the Y-axis, and from 0.28 s to 0.21 s along the Z-axis, corresponding to speedup factors of $1.51\times$, $1.32\times$, and $1.35\times$, respectively. These results confirm that automatically constraining the optimization to geometrically informative image regions lowers computational cost in a largely axis-independent manner. Overall, the runtime improvement is primarily driven by spatially reducing the processed image and point-cloud content, while the realized speedup also depends on the number and extent of the detected ROIs.

In terms of calibration accuracy, the experiments highlight that performance depends on both the perturbation regime and the chosen configuration, and that no single setting dominates across all scenarios. For translation perturbations along the X-axis, the best overall robustness—as measured by the Calibration Success Score—is achieved by *Automatic ROI + PSO* ($S_m = 0.116$), closely followed by *Manual ROI + PSO*

(0.131), whereas the reference method remains substantially worse (0.673). Along the Y-axis, however, the strongest translation robustness is obtained with manual ROI selection: *Manual ROI + PSO* yields the lowest score (0.172), followed by *Manual ROI + Grid Search* (0.213), while automatic ROI configurations are less favorable in translation recovery (0.538 and 0.631). Along the Z-axis, the best translation robustness is achieved by *Manual ROI + Grid Search* (0.079), with *Automatic ROI + Grid Search* (0.108) and *Automatic ROI + PSO* (0.151) remaining competitive; the reference method again exhibits the weakest stability (0.599). These results indicate that automatic ROI selection is highly beneficial for efficiency and can be competitive in translation recovery, but manual ROI can still provide stronger constraints in specific axes, particularly when scene knowledge enables selecting highly informative regions.

For rotational perturbations, grid-search-based approaches combined with manual ROI selection are particularly effective under small perturbations, where deterministic local exploration is sufficient. This is reflected in the pitch-axis rotation score, where *Manual ROI + Grid Search* achieves the lowest value (0.028), while *Automatic ROI + PSO* and *Automatic ROI + Grid Search* remain close (0.039). For roll and yaw rotations, *Manual ROI + Grid Search* again yields the best success scores (0.202 for roll and 0.271 for yaw), with *Manual ROI + PSO* as a consistent second-best baseline (0.219 roll, 0.309 yaw). Across roll and yaw, the automatic ROI configurations remain competitive but generally do not surpass the best manual-ROI setting in the tested single-frame regime. The reference method systematically reports higher (worse) scores (e.g., 0.524 for roll and 0.500 for yaw), confirming reduced stability under the evaluated perturbations.

Taken together, the results support three practical conclusions. First, ROI-based processing is a primary driver of computational efficiency, and automatic ROI selection provides a reliable mechanism to obtain speedups without user intervention. Second, accuracy benefits are regime-dependent: PSO is advantageous in several translation-perturbation settings (most notably X-axis), while grid search with carefully chosen manual ROIs excels under small-to-moderate rotational perturbations and in Z-axis translation robustness. Third, automatic ROI selection remains a strong candidate for deployable calibration pipelines because it removes manual tuning and consistently

reduces runtime; however, when prior scene knowledge is available, manual ROI selection can still offer superior geometric constraints in certain axes.

In conclusion, this thesis demonstrates that both manual and automatic ROI selection are valid and effective components of a targetless LiDAR–camera calibration framework, and that combining ROI selection with complementary optimizers (grid search and PSO) yields distinct advantages depending on the expected miscalibration type and magnitude. A flexible deployment strategy that adapts (i) ROI mode and (ii) optimizer choice to the anticipated perturbation regime is therefore well suited for practical systems.

Future Works

Building upon the findings of this thesis, several directions can be pursued to further improve robustness and deployability. First, the automatic ROI stage can be extended by incorporating additional object classes and/or structural priors to better capture edge-rich regions beyond vehicles (e.g., poles, fences, or road boundaries) and to stabilize performance in axes where manual ROIs currently dominate. Second, hybrid optimization strategies that combine coarse-to-fine behavior—for instance, performing a short grid-search phase for coarse alignment followed by PSO for refinement—may improve convergence speed while retaining robustness under larger perturbations.



REFERENCES

- [1] (2021). *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*, revised April 2021, Issued January 2014.
- [2] **Jo, K., Kim, J., Kim, D., Jang, C. and Sunwoo, M.** (2015). Development of Autonomous Car—Part II: A Case Study on the Implementation of an Autonomous Driving System Based on Distributed Architecture, *IEEE Transactions on Industrial Electronics*, 62(8), 5119–5132.
- [3] **Khurana, A. and Nagla, K.S.** (2021). Extrinsic Calibration Methods for Laser Range Finder and Camera: A Systematic Review, *MAPAN*, 36, 669–690.
- [4] **Zhang, Q. and Pless, R.** (2004). Extrinsic calibration of a camera and laser range finder (improves camera calibration), *Proc. 2004 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Sendai, Japan, pp.2301–2306.
- [5] **Beltran, J., Riveiro, B. and Arias, P.** (2022). A Novel Method for the Extrinsic Calibration of a 3D LiDAR and a Camera Based on the 3D-3D Point Correspondences, *Sensors*, 22, 5375.
- [6] **Verma, A., Kumar, S., Ranjan, A., Bhowmick, S. and Singh, S.K.** (2019). A Review of Camera and LIDAR Calibration Methods for Autonomous Vehicles, *IEEE Sensors Journal*, 19, 10607–10625.
- [7] **Zhou, L., Li, Z. and Kaess, M.** (2018). Automatic extrinsic calibration of a camera and a 3D LiDAR using line and plane correspondences, *Proc. 2018 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Madrid, Spain, pp.5562–5569.
- [8] **Geiger, A., Moosmann, F., Car, O. and Schuster, B.** (2012). Automatic camera and range sensor calibration using a single shot, *Proc. 2012 IEEE Int. Conf. on Robotics and Automation (ICRA)*, Saint Paul, MN, USA, pp.3936–3943.
- [9] **Geiger, A., Lenz, P., Stiller, C. and Urtasun, R.** (2013). Vision meets robotics: The KITTI dataset, *International Journal of Robotics Research*, 32(11), 1231–1237.
- [10] **Guindel, C., Beltrán, J., Martín, D. and García, F.** (2017). Automatic extrinsic calibration for lidar-stereo vehicle sensor setups, *Proc. 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, Yokohama, Japan, pp.1–6.

- [11] **Scaramuzza, D., Harati, A. and Siegwart, R.** (2007). Extrinsic self calibration of a camera and a 3D laser range finder from natural scenes, *Proc. 2007 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, San Diego, CA, USA, pp.4164–4169.
- [12] **Levinson, J. and Thrun, S.** (2013). Automatic online calibration of cameras and lasers, *Proc. Robotics: Science and Systems (RSS)*, Berlin, Germany.
- [13] **Yang, Y.** (2019). *Automatic Online Calibration Between Lidar and Camera*, <https://www.diva-portal.org/smash/get/diva2:1415878/FULLTEXT01.pdf>, independent thesis Advanced level (degree of Master (Two Years)), 20 credits / 30 HE credits.
- [14] **Yuan, C., Liu, X., Hong, X. and Zhang, F.** (2021). Pixel-level extrinsic self calibration of high resolution LiDAR and camera in targetless environments, *IEEE Robotics and Automation Letters*, 6(4), 7517–7524.
- [15] **Vaida, A.S. and Nedevschi, S.** (2019). Automatic Extrinsic Calibration of LIDAR and Monocular Camera Images, *Proc. 2019 IEEE 15th International Conference on Intelligent Computer Communication and Processing (ICCP)*, Cluj-Napoca, Romania, pp.117–124.
- [16] **Pandey, G., McBride, J.R., Savarese, S. and Eustice, R.M.** (2015). Automatic extrinsic calibration of vision and lidar by maximizing mutual information, *Journal of Field Robotics*, 32(5), 696–722.
- [17] **Canny, J.** (1986). A computational approach to edge detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6), 679–698.
- [18] **Bradski, G.** (2000). The OpenCV Library, *Dr. Dobb's Journal of Software Tools*.
- [19] **Wang, C.Y., Bochkovskiy, A. and Liao, H.Y.M.** (2022). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors, *arXiv preprint arXiv:2207.02696*.
- [20] **Yu, F., Chen, H., Wang, X., Xian, W., Chen, Y., Liu, F., Madhavan, V. and Darrell, T.** (2020). BDD100K: A Diverse Driving Video Database with Scalable Annotation Tooling, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp.12822–12831.
- [21] **Kennedy, J. and Eberhart, R.** (1995). Particle swarm optimization, *Proceedings of ICNN'95 - International Conference on Neural Networks*, volume 4, pp.1942–1948 vol.4.
- [22] **Ratnaweera, A., Halgamuge, S. and Watson, H.** (2004). Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients, *IEEE Transactions on Evolutionary Computation*, 8(3), 240–255.

- [23] **Quigley, M., Conley, K., Gerkey, B., Faust, J., Foote, T., Leibs, J., Wheeler, R. and Ng, A.Y.** (2009). ROS: An open-source Robot Operating System, *ICRA Workshop on Open Source Software*, volume 3, p. 5.
- [24] **Rusu, R.B. and Cousins, S.** (2011). 3D is here: Point Cloud Library (PCL), *IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, pp.1–4.
- [25] **ROS Wiki** (2025). *pcl_ros: ROS Interface for the Point Cloud Library*, https://wiki.ros.org/pcl_ros, accessed: 2025-01-02.
- [26] **Dagum, L. and Menon, R.** (1998). OpenMP: An industry standard API for shared-memory programming, *IEEE Computational Science and Engineering*, 5(1), 46–55.





CURRICULUM VITAE

Name SURNAME: Mahsun BİNGÖL

EDUCATION:

- **B.Sc.:** 2021, Yıldız Technical University, Faculty of Mechanical Engineering, Department of Mechatronics Engineering

PROFESSIONAL EXPERIENCE AND REWARDS:

- 2021-Present AVL Research and Engineering, Software and Functions Senior Engineer

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

- **Bingöl, M., Sezer, V. (2025).** Single-Shot ROI-Based Targetless LiDAR–camera Calibration via Edge Features. *International Conference on Electrical and Electronics Engineering (ELECO)*, Bursa, Turkey, 27-29 November, 2025.