

Real-Time Stairs Detection and Terrain Classification: Evaluating LiDAR Sensor Performance

by

Shovon Sudan Saha

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Mechanical and Mechatronics Engineering

Waterloo, Ontario, Canada, 2025

© Shovon Sudan Saha 2025

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

The growing use of wearable and autonomous systems in dynamic environments requires reliable terrain classification and obstacle detection to ensure user safety, adaptability to environmental conditions, and optimize system performance. This thesis aims to develop a complete real-time processing pipeline for terrain classification and stairs detection using LiDAR sensors, focusing on the comparative performance of both low- and high-accuracy sensors: the Cyglidar D1 and RoboSense RS-LiDAR-M1 v2.

The research introduces a novel end-to-end pipeline that integrates innovative feature extraction methods with efficient classification algorithms, optimized for real-time processing of 3D point clouds. The study utilizes Support Vector Machines (SVM) with linear and non-linear kernels to classify terrain or stairs, deploying a combination of wearable and tripod-mounted configurations to collect data. These models were implemented on the Jetson Nano, an embedded platform, to evaluate real-time feasibility. Key techniques included normal vector extraction as part of the feature extraction process, noise simulation to account for lower-cost sensor conditions, and comprehensive real-time performance analysis, enabling robust classification across diverse scenarios.

The results indicated that while the Cyglidar D1 sensor achieved a commendable F1 score of 0.96 in stairs detection, its performance in terrain classification was hindered by noise and overlapping features, particularly between plain and grassy terrains. On the other hand, the RoboSense RS-LiDAR-M1 v2 excelled, achieving an F1 score of 0.99 in terrain classification with minimal processing delay. The system proved robust against noise, maintaining an F1 score of 0.99 even with 10 mm of added Gaussian noise.

These findings validate the effectiveness of the proposed pipeline, demonstrating its ability to balance sensor cost, noise tolerance, and algorithmic efficiency while delivering robust performance in real-time applications. The research provides valuable insights into the trade-offs between sensor accuracy and cost, contributing to advancements in LiDAR-based terrain classification and obstacle detection systems. Moreover, it underscores the importance of preprocessing and sensor selection in enhancing classification reliability in real-world scenarios.

Keywords: Terrain Classification, Stairs Detection, LiDAR Sensors, Real-time Systems, Support Vector Machines, 3D Point Cloud Processing, Embedded Systems.

Acknowledgements

I would like to express my deepest gratitude to my supervisor, Dr. James Tung, for his unwavering support and guidance throughout my research. His commitment to fostering an open dialogue allowed me to explore and define my research interests, making the journey of venturing into this new domain both exciting and enriching. Dr. Tung provided invaluable feedback and patiently supported me every step of the way, encouraging me to think critically and independently while ensuring that I had the resources and mentorship needed to succeed. His patience and encouragement were crucial as I navigated the challenges of this research, and his dedication to my academic growth has left a lasting impact on both my work and my approach to research.

I am also profoundly grateful to my co-supervisor, Dr. Ehsan Hashemi, for his constant support and for providing the technical insights that greatly enhanced the quality of this work.

This thesis would not have been possible without the collaborative environment in the Lab of Neural and Rehabilitation Engineering of the Department of Mechanical and Mechatronic Engineering at the University of Waterloo. I extend my sincere thanks to my colleagues and fellow researchers in the department, whose collaboration and discussions have contributed significantly to this research.

I am especially grateful to Christian Mele for his assistance with data collection and technical support, which were pivotal in the successful completion of this thesis.

My heartfelt gratitude goes to my family and friends for their unwavering support and encouragement throughout this journey. Their patience and belief in me kept me motivated during the most challenging times. Firstly, I would like to express my eternal gratitude to my parents and my brother, without whom this post-graduate endeavor might not have been possible. My parents have sacrificed so much to allow me to attend such an esteemed world-class educational institution and pursue my dream program.

I also want to express my deepest appreciation to my close friend, Shuveccha, for her unwavering support throughout these two years of my thesis. She not only kept me motivated but also provided support whenever and however she could, whether by caring for me in every way possible or helping me discuss and refine the structure of my thesis. Her presence and encouragement have been invaluable.

Finally, I would like to acknowledge the financial support provided by the University of Waterloo, which made this research possible. Without their resources and opportunities, this thesis would not have come to fruition.

Table of Contents

Author's Declaration	ii
Abstract	iii
Acknowledgements	iv
List of Figures	viii
List of Tables	xi
1 Introduction	1
1.1 Problem Statement	1
1.2 Research Objective	5
1.3 Thesis Structure	6
2 Background/Literature Review	8
2.1 Stairs Detection	8
2.2 Terrain Classification	11
2.3 Conclusion	16
3 Algorithm Development	18
3.1 Research Design	18

3.1.1	System Architecture in Brief	19
3.2	Algorithm Development for Stairs Detection	21
3.2.1	Training Dataset	23
3.2.2	Algorithm	26
3.3	Algorithm Development for Terrain Classification	36
3.4	Conclusion	41
4	Experimental Setup & Datasets	43
4.1	System Architecture	44
4.1.1	Low-Accuracy LiDAR: Cygbot Cyglidar D1	44
4.1.2	High-Accuracy LiDAR: RoboSense RS-LiDAR-M1 v2	47
4.1.3	Data Acquisition Speed and Real-Time Efficiency	53
4.2	Experimental Setup	55
4.2.1	Low-Accuracy LiDAR Setup: Cyglidar D1	55
4.2.2	High-Accuracy LiDAR Setup: RoboSense RS-LiDAR-M1 v2	58
4.3	Conclusion	62
5	Results	65
5.1	Stairs Detection using Cyglidar	66
5.1.1	Performance Metrics Across 10 K-Folds	67
5.2	Terrain Classification using the Lower Accuracy LiDAR (Cygliclar)	74
5.2.1	Raw and Processed Point Cloud	74
5.2.2	Normal Extraction and Scatter Plot	74
5.2.3	Results using SVM with Non-linear (RBF) Kernel	78
5.2.4	Discussion	79
5.3	Terrain Classification using the Higher Accuracy LiDAR (RoboSense)	81
5.3.1	Raw and Processed Point Cloud	81
5.3.2	Normal Extraction and Scatter Plot	81

5.3.3	Results using SVM with Linear Kernel	83
5.3.4	Performance Metrics Across 5 K-Folds for Terrain Classification . .	86
5.4	Terrain Classification using the Higher Accuracy LiDAR (Robosense) with Added Noise	88
5.4.1	Results using SVM with Linear Kernel on Noisy Data	91
5.4.2	Performance Metrics Across 5-Fold Cross-Validation for Terrain Clas- sification with 10mm Gaussian Noise	93
5.5	Conclusion	96
6	Conclusion and Future Work	97
6.1	Summary of Findings	97
6.2	Limitations	99
6.3	Recommendations	101
6.4	Future Work	104
6.5	Conclusion	107
	References	109

List of Figures

2.1	Mobile wheeled robot with LiDARs and computing unit [38]	10
2.2	Stairs detection in real-world and plane modeling [71]	12
2.3	System architecture for terrain classification using CNNs, involving classifiers for environment type, terrain visibility, and terrain classification within a cropped image center [40]	13
2.4	Terrain classification performed on-board: (a) Data collected in a forested region. (b) Data collected on a trail. [29]	14
3.1	Software ROS Architecture Flow for Stairs Detection and Real-time Terrain Classification.	20
3.2	Stair detection threshold arc used to define collection labelling	24
3.3	Diverse Staircase Dataset for LiDAR Training	24
3.4	Example collection scene in RViz displaying LiDAR scenes for a metal stairwell.	26
3.5	Flowchart of the stair detection algorithm development process using LiDAR data, from data acquisition to model evaluation.	27
3.6	Voxelization Process: Original points (black dots) and their centroids (red stars) within each voxel.	29
3.7	Flowchart of the terrain classification algorithm development process using LiDAR data.	37
4.1	Cygbot Cyglidar D1 solid-state LIDAR with accessories [9].	45
4.2	RoboSense RS-LiDAR-M1 LiDAR [52].	48
4.3	Topology of LiDAR and PC [51].	51

4.4	Essential components Involved connecting the RoboSense LiDAR [51]. . . .	52
4.5	Data acquisition setup with tripod and walker	57
4.6	a) Cyglidar mounted on tripod b) Cyglidar in wearable-format, both connected with the mobile workstation on walker	58
4.7	Cygbot Cyglidar D1 collection mount design with accompanying phone for validation	59
4.8	(a) Point cloud of the stairs captured by Cyglidar D1 in tripod setup, (b) Point cloud of grassy terrain captured by Cyglidar D1 in wearable setup. .	59
4.9	Tripod setup for the RoboSense RS-LiDAR-M1 v2 sensor, featuring a 3D printed adapter for secure mounting, a spirit level for precise alignment, and a stable tripod for stationary data acquisition.	61
4.10	(a) RoboSense LiDAR setup facing a staircase, (b) Corresponding point cloud of the stairs, (c) RoboSense LiDAR setup facing grassy terrain, (d) Corresponding point cloud of the grassy terrain.	63
5.1	Frequency analysis of the top 6 most misclassified scenarios for false negatives and false positives.	70
5.2	Examples of False Positives in Fast Walking and Office Desk Reflective Scenes.	71
5.3	Examples of False Negatives: Climbing Stairs and Plain Surface in Sunlight.	72
5.4	Cyglidar D1 point cloud data: (a) Raw point cloud for plain terrain, (b) Raw point cloud for grassy terrain, (c) Downsampled point cloud after passthrough filtering for plain terrain, (d) Downsampled point cloud after passthrough filtering for grassy terrain.	75
5.5	Normals extracted from downsampled Cyglidar D1 point clouds for (a) plain terrain, and (b) grassy terrain.	76
5.6	Scatter plot of NormalX vs. NormalY for plain and grassy terrains (components in radians).	77
5.7	(a) View of the plain terrain, (b) Experimental setup for grassy terrain, (c) Raw point cloud data for plain terrain, and (d) Raw point cloud data for grassy terrain.	82
5.8	Normals extracted from the downsampled RoboSense RS-LiDAR-M1 v2 point clouds for (a) plain terrain, and (b) grassy terrain.	83

5.9	Scatter plot of NormalX vs. NormalY for plain and grassy terrains using RoboSense RS-LiDAR-M1 v2.	84
5.10	Processed point clouds with added Gaussian noise: (a) 6 mm noise, (b) 10 mm noise.	89
5.11	Scatter plot of NormalX vs. NormalY for plain and grassy terrains with 10 mm Gaussian noise added.	90

List of Tables

5.1	Final RBF Function Parameters	66
5.2	SVM 10 K-Fold Total Performance	67
5.3	SVM classifier performance	67
5.4	Performance Metrics Across 10 K-Folds for Stairs Detection using Cyglidar	67
5.5	Real-time performance of SVM with RBF kernel on Jetson Nano.	79
5.6	Precision, Recall, and F1 Score for SVM with RBF kernel on Jetson Nano.	79
5.7	Performance metrics for SVM with RBF kernel on Jetson Nano.	80
5.8	Real-time performance results using SVM with Linear Kernel on Jetson Nano.	85
5.9	Precision, Recall, F1 Score, and Support for SVM with Linear Kernel. . . .	85
5.10	Confusion Matrix for SVM with Linear Kernel.	86
5.11	Performance Metrics Across 5 K-Folds for Terrain Classification	87
5.12	Real-time performance results using SVM with Linear Kernel on Jetson Nano on Noisy LiDAR Data.	91
5.13	Precision, Recall, and F1 Score for SVM with Linear Kernel for Noisy LiDAR Data.	91
5.14	Confusion Matrix for SVM with Linear Kernel on Noisy Data.	92
5.15	Performance Metrics Across 5-Fold Cross-Validation for Terrain Classifica- tion (10mm Gaussian Noise)	93

Chapter 1

Introduction

1.1 Problem Statement

With rapid advancement of medicine, patient and therapeutic care, the general population has benefited from longer life expectancy. The growing demographic of the elderly and individuals with disabilities underscores an urgent need for advanced assistive technologies addressing mobility disabilities, such as walkers and exoskeletons. Although mobility assistive technologies have seen significant advancements, their effectiveness in complex environments remains limited. These environments often feature different terrain types and obstacles such as stairs, which pose navigational challenges that conventional mobility aids struggle to handle effectively. These challenges can compromise safety and hinder user independence, which requires a technological breakthrough to enhance navigation aids.

Globally, more than 2.5 billion people require one or more assistive products, such as wheelchairs or hearing aids, according to the World Health Organization [69]. As the demand for assistive devices continues to increase, the need for these technologies to become adaptable to our environments becomes increasingly evident. Human beings are blessed with a sophisticated cognitive system that enables seamless navigation in our daily environments. Unfortunately, people with cognitive and / or mobility impairments are challenged by difficulty in detecting and navigating changes in planes, commonly found obstacles such as stairs or different terrains in their neighborhood [59], all of which would require a change in gait pattern for humans or exoskeletons.

In response to these growing challenges, recent developments in visual and inertial-based methods for environmental perception in assistive devices have seen significant advancements, particularly in the integration of sophisticated computer vision algorithms,

deep learning models, and sensor fusion techniques. These technologies leverage RGB-D sensors and depth cameras to create accurate 3D mappings of the environment, improving the spatial awareness of assistive devices [39][60]. Inertial measurement units (IMU) are increasingly utilized to track user movements and orientation, often in combination with visual data, to provide a more robust and reliable perception of the environment [50]. Wearable devices, such as smart glasses and chest-mounted cameras, are now equipped with visual and inertial sensors, providing users with comprehensive environmental awareness [60]. In addition, innovations in smart canes, which integrate multiple sensors, including cameras, ultrasonic sensors, and IMUs, provide enhanced environmental information to aid navigation [39][45]. Many modern assistive technologies also capitalize on the capabilities of smartphones, using their powerful sensors and processing capabilities to create portable, cost-effective solutions [39][60]. These advances aim to improve obstacle detection, navigation assistance and general spatial awareness, with ongoing research focusing on improving accuracy, reliability, user friendliness, and real-time processing capabilities [60][45].

Building on these advancements, this research is driven by the critical need to improve the autonomy and safety of mobility aids through the integration of sophisticated sensing technologies, such as LiDAR (Light Detection and Ranging). The motivation comes from the potential of LiDAR technology to transform mobility aids into more intelligent, reliable, and accessible tools. These enhancements are crucial to enabling users, particularly those with mobility impairments, to navigate more safely and independently in diverse, dynamic, and unpredictable environments.

This is especially relevant in the context of wearable mobility assistive technology in the lower extremities (e.g. exoskeletons, orthoses, prostheses), which are active or passive devices worn by users with some form of mobility impairment to aid and support gait and stability [21]. Active lower limb devices deliver a combination of preset and online adjusted assistance profiles to the user to aid or replace gait trajectories. Altering assistance profiles to avoid or traverse terrain of different types and topology alternative to traditional flat ground requires extensive knowledge of the surrounding environment and is a constant challenge in robotics and control. Specifically, for wearable robotics, identification and navigation of the plane / stairs are a complex issue when determining efficient and effective methods of identification and traversal [28].

As most of the state-of-the-art algorithms involve some sort of camera (RGB-d, in most cases) to detect those obstacles, there exists a privacy concern limiting adoption of such systems in exoskeletons and rehabilitation devices designed to assist individuals with mobility impairments [44]. Although exoskeletons have the potential to significantly improve the quality of life for individuals with disabilities, many users are hesitant to use

them due to concerns about privacy [24]. Preserving user privacy is crucial when developing any assistive devices, exoskeletons as such, for several reasons. Firstly, exoskeletons are often used to assist individuals with physical disabilities or injuries, and these individuals may have sensitive medical or personal information that needs to be protected. Secondly, exoskeletons can contain sensors and cameras that gather data about the user, such as their movements and biometric information. This data could potentially be used for purposes beyond the intended use of the exoskeleton, such as marketing or research, without the user’s consent. Finally, preserving user privacy builds trust between the user and the designer, which is essential for successful adoption of the technology.

Stairs pose a complex problem for assisted mobility, requiring detailed knowledge of the scene and surfaces with the explicit intent of ascending or descending while navigating a potentially precarious fall risk. This complexity is highlighted by the multifaceted challenges in stair detection and recognition for assistive devices [26], where accurate scene understanding and surface analysis are critical. Recent research has emphasized the importance of recognizing human motion intent when approaching staircases [13], leveraging deep learning-based techniques for stair detection [73] and the use of 3D point cloud data to address these challenges [60]. The integration of multiple sensors and fusion strategies has also been explored to enhance locomotion activity recognition in complex environments such as staircases [18]. With the growing demand for assistive devices, there is an increased potential for interaction with stairs in unmapped environments, making accurate identification, evaluation, and characterization of terrains and stairs crucial to user safety.

The focus of this thesis is centred on the identification of key environmental features, such as terrain and stairs, that pose mobility challenges, using onboard solid-state light detection and ranging (LiDAR) sensors. Particular emphasis is placed on evaluating the performance of LiDAR systems in real-time embedded applications for detecting stairs as a critical environmental feature.

Although stairs detection and characterization have been extensively investigated in robotics and wearable devices, little work has been done on the detection of different terrains, especially in the domain of assistive devices. Stair detection has been examined in mobile robotics, where the sensor stack is in parallel with the ground, simplifying algorithm development. The state-of-the-art algorithms in those robots currently available rely heavily on the high-end sensor stack (LiDAR, RGB-d, Depth Sensing, etc.) combined with access to advanced computing platforms due to heavy computational cost. The high sensor cost and computational requirements limit the deployable capability of such systems regardless of their detection accuracy/capability. **This gap in the deployability of terrain detection is addressed in this thesis with an affordable sensor and a smaller computing stack.**

Using LiDAR sensors as a sensing modality can provide several advantages over video modalities for assistive mobility devices (e.g., exoskeletons). One of the main advantages is that LiDARs can capture depth information without requiring cameras or other sensors that could potentially invade the user’s privacy. This capability is well established in the scientific community, as evidenced by several high-impact studies. Schwarz et al. (2010) provides a comprehensive overview of LiDAR technology, highlighting its ability to map the world in 3D without relying on traditional imaging [56]. Behroozpour et al. (2017) further elaborate on LiDAR system architectures, emphasizing their standalone depth-sensing capabilities [3]. The fundamental principle of LiDAR, as explained by NOAA, involves using pulsed laser light to measure variable distances, generating precise three-dimensional information about the characteristics of the Earth’s surface without the need for camera-based imaging [41]. Additionally, Shan and Toth (2018) detail how LiDAR systems use time-of-flight measurements of laser pulses to directly capture depth information, independent of traditional photographic methods [58]. Another advantage of using LiDARs is that they can work in low-light or complete darkness, which can be useful in situations where lighting conditions are poor or where users need to move around in dark environments. Additionally, LiDARs can capture more accurate and detailed 3D information than cameras, which can improve the performance of the exoskeleton by providing more precise data about the user’s surroundings.

Although cameras have been commonly used for environmental perception in exoskeletons, LiDAR technology offers several advantages that can enhance exoskeleton performance and functionality. LiDAR sensors provide more accurate and detailed 3D information about the surrounding environment, creating precise point clouds that can improve the spatial awareness of an exoskeleton and the ability to navigate complex terrains [33]. This enhanced 3D perception can lead to better support and stability for exoskeleton users, especially on uneven surfaces or stairs [33]. LiDAR also performs better in low light or dark conditions compared to cameras, as it actively emits laser pulses rather than relying on ambient light [70]. Furthermore, LiDAR offers superior privacy protection by capturing spatial data without identifying individuals [15]. The integration of LiDAR technology into exoskeletons can enable more robust environmental perception and adaptive control strategies, potentially improving user safety and the overall effectiveness of rehabilitation and assistance applications [15][70]. Recent research has focused on refining the algorithms that process LiDAR data for more effective terrain classification. Techniques such as machine learning and deep learning are being employed to improve the accuracy of these classifications. These algorithms are trained in vast datasets of varied terrains to learn distinguishing features that are not immediately apparent to human operators or simpler detection systems.

Incorporating both stairs detection and terrain classification into the functionality of LiDAR-equipped assistive devices can significantly broaden their scope and effectiveness of application. By advancing methods to interpret and interact with their environment more intelligently, developers can create solutions that offer unprecedented support and safety for individuals with mobility challenges, fulfilling a critical need in the assistive technology market.

Advancing on existing systems utilizing high-cost sensors and computationally intensive algorithms, this research introduces a comprehensive pipeline for real-time processing of 3D point clouds using relatively low-cost LiDAR technology. This pipeline includes novel feature extraction techniques tailored to classify environmental features, specifically terrains and stairs. By addressing privacy concerns, computational constraints, and deployment challenges, the pipeline advances the potential of assistive devices to adapt to environmental conditions, affording greater adaptability in real-world scenarios.

1.2 Research Objective

The overarching objective of this thesis is to address the limitations of current assistive mobility technologies by developing a reliable, low-cost LiDAR-based system for stairs and terrain classification. This system is designed to enhance the autonomy, safety, and performance of assistive devices in diverse real-world settings while addressing key challenges such as privacy concerns, computational constraints, and adaptability to complex environments. The system will be evaluated under different environmental conditions, including scenarios with noise-simulated data, to ensure robustness and applicability across diverse environments. To achieve this overarching objective, the research has been structured around the following key sub-objectives:

1. Develop a Low-Cost, Low-Computation Algorithm for Stair Detection:

Addressing the problem of safe navigation in environments with stairs, this objective focuses on designing an algorithm that uses LiDAR data to accurately identify ascending stairs in real time. The algorithm must be computationally efficient to run on resource-constrained platforms, such as those used in assistive devices, while maintaining high accuracy. This objective directly responds to the need for safer, more reliable mobility aids capable of operating under limited hardware resources.

2. Investigate Terrain Classification Capabilities:

To meet the challenge of navigating varied terrains (as highlighted in the problem statement), this objective involves the development of a robust algorithm capable of reliably classifying terrain types. By simulating the performance of lower-accuracy LiDAR sensors through noise injection, the study explores the feasibility of low-cost solutions for real-world applications. This objective addresses both computational efficiency and affordability, ensuring the system can adapt to dynamic environments without sacrificing performance.

1.3 Thesis Structure

The remainder of this thesis is structured as follows:

- **Chapter 2: Background and Literature Review**

This chapter delves into the fundamental concepts of LiDAR technology, focusing on its applications in terrain classification and stairs detection. It provides a comprehensive review of existing methodologies and approaches that have been employed in related research, analyzing their effectiveness and identifying the gaps that this study aims to address. The chapter sets the stage by contextualizing the current research within the broader scope of existing knowledge.

- **Chapter 3: Algorithm Development**

Chapter 3 outlines the step-by-step development of the algorithms used in this research. It reports the mathematical foundations, logic, and pseudocode of underlying the algorithms, emphasizing the innovative approaches introduced in this thesis. The chapter also touches on the hardware architecture that supports these algorithms, offering insights into the integration of software and hardware components to achieve the research goals.

- **Chapter 4: Experimental Setup**

This chapter describes the experimental setups designed to test and validate the developed algorithms. It details the specific hardware configurations, including the LiDAR sensors and associated components, as well as the environments in which data was collected. The chapter is divided into sections focusing on stairs detection and terrain classification, each discussing the methodology employed to gather and preprocess the data for subsequent analysis.

- **Chapter 5: Results and Discussion**

Chapter 5 presents the results obtained from the experimental setups, offering a detailed analysis of the performance of the developed algorithms. The chapter discusses the accuracy, efficiency, and potential limitations of the algorithms, comparing them with results from existing studies. The discussion also explores the broader implications of the findings, considering the impact on future research and practical applications.

- **Chapter 6: Conclusion and Future Work**

The final chapter summarizes the key contributions and findings of this research. It reflects on the objectives set out at the beginning of the thesis, evaluating how well they have been achieved. The chapter concludes with a discussion of the potential for future work, suggesting ways in which the research could be expanded or refined to address any limitations identified during the study.

Chapter 2

Background/Literature Review

To develop a reliable, low-cost terrain classification system using LiDARs that enhances the reliability, performance, and safety of assistive devices in real-world settings, it is crucial to understand the current state of research in this domain. The literature review is divided into two key sections: stairs detection and terrain classification. The first section focuses on examining various approaches and methodologies that have been used to identify stairs using LiDAR and other sensor technologies. The second section delves into terrain classification, exploring the methods and algorithms used to differentiate between different types of terrains, which are essential for enhancing the navigational capabilities of assistive devices. By reviewing these areas, the thesis aims to build on existing knowledge and address gaps in the literature, ultimately contributing to the development of more robust and efficient assistive technologies.

2.1 Stairs Detection

As a common obstacle that must be navigated safely and effectively, stair negotiation is a major challenge in the deployment of assistive technologies, such as rehabilitation exoskeletons. The detection of stairs and planes has been a well-explored area in the field of autonomous robotics, where accurate environmental perception is essential for safe navigation. Although much of the existing research in this domain relies on cameras or a combination of cameras and other sensors, the context of assistive devices demands alternative approaches, especially those prioritizing user privacy. In such scenarios, the use of LiDAR technology becomes increasingly relevant due to its ability to capture detailed depth information without compromising privacy.

Recent advances have focused on using LiDAR for stairs detection to prevent accidents, particularly among visually impaired individuals. As a key example, Matsumura et al. introduced a novel approach that employs deep learning techniques to analyze 3D point cloud data for stair detection. Their study utilized a convolutional neural network (CNN) to classify the point clouds into stairs and non-stairs categories, achieving high accuracy with a low false-positive rate. The proposed system was tested in various environments, demonstrating its potential to improve the safety and mobility of visually impaired users by alerting them to the presence of stairs [38].

The use of deep learning for stair detection marks a significant shift from traditional methods, which often relied on geometric features or simpler classifiers. By incorporating deep learning algorithms capable of understanding complex patterns within point-cloud data, researchers are moving closer to developing robust, reliable systems that can operate effectively in real-world scenarios. For example, the system developed by Matsumura et al. not only identifies stair structures, but also evaluates their characteristics, such as ascent or descent, providing critical contextual information that can be used to prevent falls and other accidents [38].

The deep learning-based stairs detection system described by Matsumura et al. leverages advanced neural networks to achieve high accuracy and low false-positive rates in identifying stairs within 3D point cloud data. While this approach demonstrates significant potential, particularly in enhancing the safety and mobility of visually impaired individuals, it is not without limitations. One primary concern is the system’s dependency on large, well-annotated training datasets, which may limit its applicability in diverse real-world scenarios where such data are not readily available. Additionally, the computational demands of deep learning models may restrict their use in low-power or resource-constrained devices, often found in assistive technologies. Environmental variations, such as changes in lighting or unexpected obstacles, can also impact the reliability and consistency of the system’s performance. Therefore, while the system presents a promising solution, these potential drawbacks must be carefully considered when evaluating its suitability for deployment in various assistive contexts.

LiDAR technology has frequently been integrated into mobile robotic platforms, often mounted on wheeled robots or vehicles, to enable real-time environmental perception and navigation. For instance, Bansal et al. proposed an architecture for streaming and processing 3D point-cloud data from a moving robot equipped with dual LiDAR sensors, one oriented parallel to the ground and the other perpendicular to it. This setup, illustrated in Figure 2.1, uses a multithreaded streaming pipeline to handle LiDAR data in real-time, facilitating accurate characterization of complex 3D structures such as doors and stairs. The system’s efficacy in detecting and modeling these structures has been validated in live

streaming scenarios, showcasing its potential for dynamic, real-world applications [2].



Figure 2.1: Mobile wheeled robot with LiDARs and computing unit [38]

However, real-time processing of high-resolution LiDAR data presents significant computational challenges. The requirement for substantial computational power may pose limitations for deploying such systems on low-power devices or in environments with limited resources. Furthermore, system performance can be affected by environmental factors, including variable lighting conditions or sensor obstructions, which might introduce noise or inaccuracies in the LiDAR data. Furthermore, when applied to assistive devices, such as exoskeletons, the challenge extends to stabilizing the sensor data, especially given the continuous movement and variability in a user’s gait. This complexity adds another layer of difficulty in ensuring system reliability and consistency in real-world applications.

As a key challenge in real-world mobility, determining relevant information on the presence and geometry of stairs is a fundamental problem in control of exoskeletons. Given the impracticality of preloading all possible stair geometries into an exoskeleton’s control system, Zhao et al. proposed a method combining LiDAR and RGB-D camera data to dynamically detect and model stairs in real-time. Their approach focuses on identifying the position, orientation, and geometric parameters of stairs, including the number of steps, the height of the riser and the depth of the tread, which are critical for safe navigation (Fig.

2.2) [71]. The system segments the stairs into planes using different colors, facilitating the visualization and interpretation of the detected structures. This method presents a significant advancement in enabling exoskeletons to adapt to various stair configurations in real-time, thus enhancing their functionality for rehabilitation purposes, particularly for patients recovering from hemiplegia (i.e., paralysis affecting one side of the body, including the lower face, arm, and/or leg [71]).

Despite being designed for real-time operation, the algorithm presented in this study is not immune to privacy concerns associated with RGB video streams capturing identifiable features (e.g., faces). Furthermore, the fusion of LiDAR and vision data requires significant computational power and may be sensitive to environmental factors, such as lighting conditions and occlusions affecting accuracy and reliability. Additionally, the effectiveness of the system may be limited by variations in the geometries of different staircases, which can make it difficult to generalize the model to different environments.

2.2 Terrain Classification

In the domain of terrain classification, deep learning methods (e.g., CNNs), have become increasingly prevalent due to their ability to process and analyze complex environmental data. Moore et al. (2023) made significant strides in this area by leveraging CNNs to classify terrains based on video data obtained from online sources. Considering a lack of annotated datasets, their research focused on creating a novel dataset by mining images from videos and then training CNN models to accurately classify various types of terrain. The innovative aspect of this study lies in the automated data curation tool developed by the authors, which enabled efficient acquisition and annotation of video data, facilitating the training of their deep learning models [40].

In this thesis, a novel approach is proposed by utilizing low-cost LiDAR data for terrain classification. Unlike Moore et al., who rely on video data, this research employs LiDAR, which provides 3D spatial information that may be less affected by variable conditions such as lighting and image quality challenges that often arise in video-based data collection. The focus on LiDAR also addresses privacy concerns, as it does not capture identifiable personal information, which is a potential risk when using video data. Moore et al. address privacy concerns by proposing the use of proposed automated filtering methods to remove sensitive information from video data [40]. While this may mitigate data collection concerns, perceived privacy risks associated with the sensing modality (i.e., video) remains an ongoing issue [61].

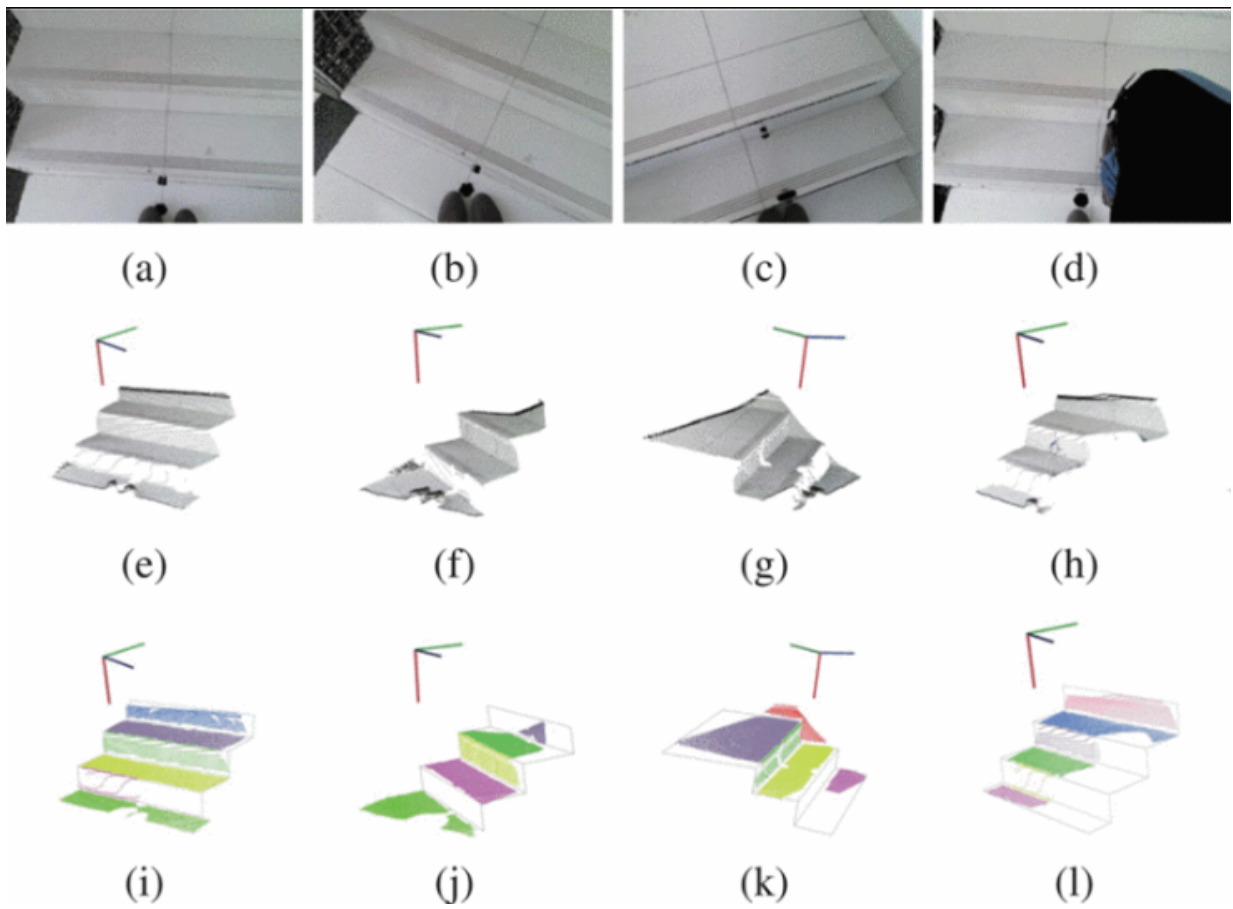


Figure 2.2: Stairs detection in real-world and plane modeling [71]

Figure 2.3 illustrates the proposed architecture for processing video data through CNNs to classify different terrains. The architecture consists of multiple stages: the first classifier (#1) identifies whether the image is from an indoor or outdoor environment. If the image passes this stage, the subsequent classifiers (#2 and #3) determine whether the terrain is visible within the central 500×500 pixel crop of the image. If the terrain is visible, the final classifiers (#4 and #5) proceed to classify the type of terrain depicted in the center crop. This systematic approach exemplifies the complexity and robustness of Moore et al.'s system in handling diverse environmental data, making it a key reference for understanding the application of deep learning in terrain classification [40].

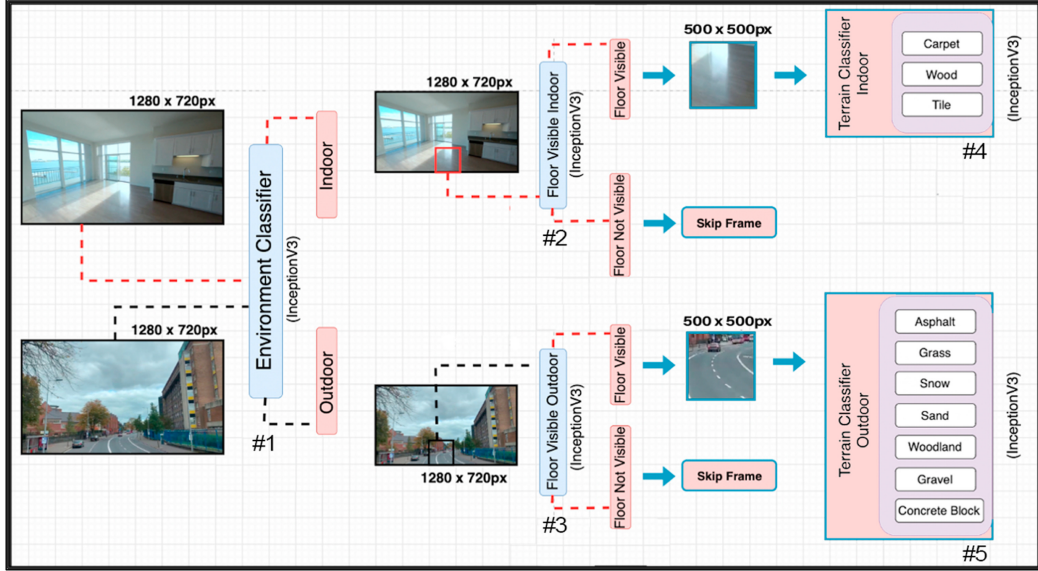


Figure 2.3: System architecture for terrain classification using CNNs, involving classifiers for environment type, terrain visibility, and terrain classification within a cropped image center [40]

Lalonde et al. (2006) investigated the challenges and solutions for improving ground robot navigation through effective terrain classification. Their work focused on the classification and segmentation of natural terrains using high-resolution LiDAR data, employing both geometric and statistical analyses to categorize various terrain types (e.g., grass, dirt, and gravel). The detailed point clouds generated by LiDAR allow for precise terrain mapping, which is crucial for autonomous ground robots operating in complex outdoor environments [29].

As shown in Figure 2.4, Lalonde et al. classified terrain in real-time using the onboard

computing stack of a mobile robot with 3D LiDAR data. This classification system is essential for allowing the robot to adapt its navigation strategies based on the detected terrain, highlighting the practical implications of their research in autonomous systems. The figure also exemplifies the potential of LiDAR technology to provide reliable terrain data that can be leveraged not only within robotics but also in assistive technologies, where safe and effective navigation is paramount [29].

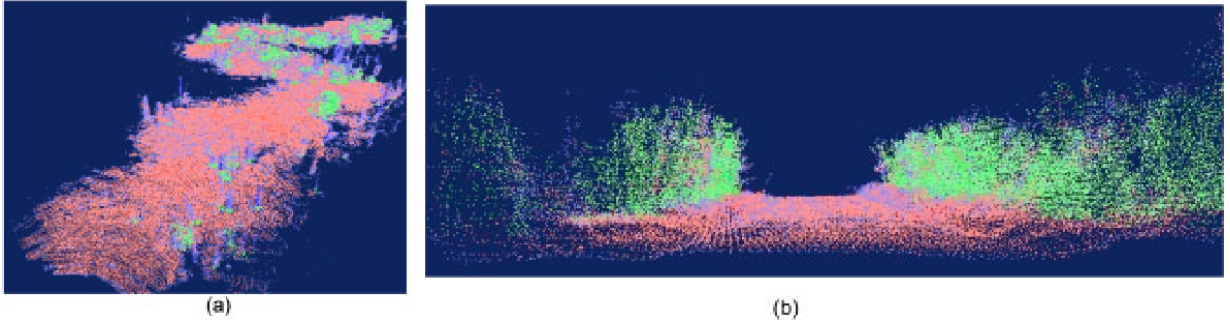


Figure 2.4: Terrain classification performed on-board: (a) Data collected in a forested region. (b) Data collected on a trail. [29]

Lalonde et al. extracted key geometric and statistical features from the LiDAR data to characterize different terrains, including:

- **Surface Roughness:** Calculated by analyzing elevation variability within a local neighborhood of points, enabling the distinction between rough terrains like rocks and smoother surfaces such as grass.
- **Point Density:** Utilized to infer terrain type based on the density of points within the cloud, where denser point clouds may indicate more complex terrains.
- **Local Planarity:** Assesses how well points fit a plane, aiding in the identification of flat ground versus irregular surfaces.
- **Height Variance:** Differentiates terrains with varying elevation levels, providing further granularity in classification.

The extracted features are fed into a Gaussian Mixture Model (GMM), a probabilistic model well suited for terrain classification due to its ability to represent the probability distribution of these features in a continuous space. The GMM, trained on labeled terrain data, learns the statistical properties of each terrain class and is later used to classify new

unlabeled LiDAR data by calculating the likelihood of the observed features belonging to each class [29]. Lalonde et al. also examined terrain segmentation methods, which are crucial to isolating different terrain types within a single LiDAR scan. The reported segmentation process involves clustering LiDAR points into distinct regions based on their spatial proximity and feature similarity, allowing for the individual classification of each segment. This is particularly useful in complex environments where multiple terrain types might be present in a single scan [29].

The study’s emphasis on high-resolution LiDAR and advanced probabilistic modeling provides a robust framework for terrain classification in autonomous ground robots. However, the approach requires significant computational resources and high-accuracy sensors, making it less accessible for applications where cost and computational efficiency are critical. In contrast, the current research explores the use of low-cost LiDAR systems, coupled with noise-injected simulations, to develop scalable and practical solutions for terrain classification in assistive devices. This research aims to ensure that these methods are not only accurate, but also feasible for real-world applications in human-centric environments, where both cost and computational efficiency are paramount [29].

The challenges and advancements in LiDAR-based terrain classification for various other applications, including agriculture, robotics, and autonomous navigation, can also be found in the literature. The UAS-LiDAR study compared traditional ground classification algorithms with deep learning methods, demonstrating the superior performance of machine learning approaches in the handling of high-density point clouds [14]. The Natural Terrain Classification Paper introduced a method for segmenting 3-D point clouds to classify vegetation, solid surfaces, and linear structures, addressing the complexities of processing 3-D data in real-time [65]. Another study focused on object detection and terrain classification in agricultural fields using 3D LiDAR data, achieving high accuracy in distinguishing between ground, vegetation, and objects using SVM classifiers [27]. These works underscore the evolving landscape of LiDAR-based terrain classification and the increasing role of machine learning in improving accuracy and efficiency.

To summarize, the advancements in LiDAR-based terrain classification, as explored in this section, highlight the efforts in precise environmental perception for both autonomous systems and assistive technologies. The integration of LiDAR sensors, particularly in applications like mobile robots, agriculture, and real-time navigation, has demonstrated notable progress in handling complex terrains and environments. While higher-resolution sensors such as the RoboSense RS-LiDAR-M1 v2 excel in distinguishing between different terrain types, even low-cost sensors have shown potential when combined with robust algorithms and preprocessing techniques. The studies discussed emphasize the ongoing challenges related to computational resources, noise, and real-time processing but also point to the

growing effectiveness of machine learning techniques in addressing these challenges. This thesis builds on these findings by investigating scalable, practical solutions for terrain classification in assistive devices, ensuring that cost, efficiency, and performance are balanced to meet real-world demands.

2.3 Conclusion

The literature reviewed in this chapter underscores the significant advancements and ongoing challenges in the fields of stairs detection and terrain classification, particularly within the context of assistive technologies. The reviewed studies highlight the critical importance of accurate environmental perception for enhancing the safety and functionality of mobility aids like rehabilitation exoskeletons and other assistive devices.

In the domain of stairs detection, the evolution from traditional geometric approaches to more sophisticated methods like deep learning has been particularly noteworthy. Research such as that by Matsumura et al. demonstrates how convolutional neural networks (CNNs) can effectively process 3D LiDAR data to identify and characterize stairs, thereby offering promising solutions to improve the mobility and safety of visually impaired individuals. These advancements also illustrate the potential for integrating advanced sensing and computational technologies into assistive devices to better navigate complex environments. However, challenges remain, including the high computational costs associated with deep learning models and the need for extensive annotated datasets, which may limit the scalability of these methods in diverse real-world scenarios.

For terrain classification, the contrast between video-based and LiDAR-based approaches has been thoroughly explored. Studies by Moore et al. and Lalonde et al. illustrate the strengths and limitations of using different sensing modalities for environmental perception. While video-based methods benefit from well-developed deep learning frameworks, they are susceptible to variations in lighting and other environmental conditions. In contrast, LiDAR-based methods offer more consistent and reliable data for terrain classification, as evidenced by Lalonde et al.'s work on ground robots. The use of LiDAR, coupled with techniques such as PCA and Gaussian Mixture Models (GMMs), enables precise classification of complex terrain types, though often at the cost of increased computational demands.

In addition, more recent studies have highlighted the growing role of machine learning to process LiDAR data for terrain classification tasks across various applications. Research by Fareed et al. and Kragh et al. have demonstrated the potential of LiDAR in agriculture

and autonomous navigation, using high-density point clouds to accurately classify different terrain types. These studies emphasize the computational challenges inherent in real-time LiDAR data processing, particularly on resource-constrained platforms such as mobile robots and assistive devices. The need for efficient algorithms that can handle large, noisy datasets while maintaining high classification accuracy has become a key motivation for continued research in this area.

The motivation for developing efficient algorithms stems from the computational challenges identified in these studies. As highlighted by the concept of algorithmic efficiency, there is a constant need to optimize resource usage, particularly for embedded systems with limited computational power. Real-time processing of high-density point clouds, often exceeding 100,000 points per second, poses significant challenges for mobile and embedded platforms. The need for algorithms that can handle diverse terrain types, sensor configurations, and maintain accuracy while reducing computational demands is crucial. Efficient algorithms that can run on embedded computing units like NVIDIA Jetson Nano would enable real-time terrain classification in resource-constrained environments, expanding the applicability of this technology in robotics, agriculture, and environmental monitoring. This push for efficiency aligns with the broader trend in computer science of maximizing performance while minimizing resource consumption, as emphasized by Donald Knuth's statement on the importance of even small improvements in software engineering.

In summary, the review highlights that while substantial progress has been made in both stairs detection and terrain classification, there is a clear need for methods that balance accuracy, computational efficiency, and scalability. The findings from the literature provide a solid foundation for the development of new algorithms that leverage the strengths of existing methods while addressing their limitations. The research presented in this thesis aims to contribute to this ongoing effort by developing robust, cost-effective systems for terrain classification and stairs detection, ultimately enhancing the safety and reliability of assistive devices in real-world settings.

Chapter 3

Algorithm Development

The Algorithm Development chapter outlines the systematic approach undertaken to develop an efficient staircase detection and terrain classification system using LiDAR technology. The primary aim of this research is to explore the feasibility and effectiveness of using low-accuracy LiDAR sensors for these tasks, providing insights into how cost-effective solutions can be implemented in assistive technologies. To ensure a comprehensive evaluation, this thesis also benchmarks the performance of these systems against high-accuracy LiDAR data, assessing potential improvements achievable with more precise (and higher-cost) sensing technologies.

This chapter begins with an overview of the research design, followed by an in-depth discussion of the system architecture, which highlights the hardware setup and configurations used for data collection. The latter sections delve into the algorithm development processes for both staircase detection and terrain classification, including the mathematical foundations of the methods employed and pseudo-codes where appropriate.

3.1 Research Design

The research design of this study is structured to achieve dual objectives: developing and validating algorithms for stairs detection and terrain classification using LiDAR technology, with a focus on cost-effective and computationally efficient solutions. This research employs an experimental and computational approach that facilitates controlled, systematic testing of algorithms developed under various conditions to examine their applicability in real-world scenarios.

The core hypothesis of this research posits that low-cost LiDAR systems can be used effectively for the reliable detection of stairs and the classification of terrains, despite their inherent limitations in resolution and accuracy. By examining approaches that mitigate these limitations, this study aims to expand the accessibility and affordability of advanced mobility aids, providing practical solutions to improve the autonomy and safety of individuals using assistive devices.

3.1.1 System Architecture in Brief

The system architecture designed for this research integrates both low- and high-accuracy LiDAR sensors, enabling a thorough evaluation of terrain classification and stairs detection algorithms under various environmental conditions. The modularity of the system allows for seamless integration of these sensors into existing rehabilitation devices, requiring minimal hardware or software modifications. The LiDAR sensors utilized in this study are as follows:

- **Low-Accuracy LiDAR:** The Cygbot Cyglidar D1 was selected due to its cost-effectiveness, lightweight design, and ease of integration [9]. This sensor operates using dual 2D and 3D solid-state TOF (Time-of-Flight) technology, providing reliable performance for both static and dynamic environments. It features a horizontal field of view (FOV) of 120° and a vertical FOV of 65°, with a measurement speed of 15 Hz. The Cyglidar D1 detects objects at distances ranging from 0.2 to 8 meters for 2D and from 0.5 to 2 meters for 3D, with a $\pm 1\%$ accuracy. Its compact form factor, weighing only 31g, makes it particularly suitable for wearable applications. The detailed setup for both the tripod-mounted and wearable configurations will be discussed in **Chapter 4**.
- **High-Accuracy LiDAR:** The RoboSense RS-LiDAR-M1 v2 served as the high-accuracy sensor benchmark [52]. Utilizing advanced MEMS (Micro-Electro-Mechanical Systems) technology, the RS-LiDAR-M1 v2 generates high-resolution 3D point clouds, offering a horizontal FOV of 120° and a vertical FOV of 65°, with a measurement speed of 20 Hz. This sensor detects objects within a range of 0.2 to 200 meters, with a fine resolution of 0.1° in both horizontal and vertical axes, ensuring precise environmental data collection. The sensor operates in temperatures ranging from -20°C to 60°C and weighs 0.9 kg, making it more suitable for tripod-mounted configurations, especially for applications requiring detailed, accurate measurements. The computational unit was selected to handle the high data throughput of this sensor, ensuring real-time data acquisition and processing.

Figure 3.1 illustrates the overall ROS-based system architecture and the flow of data from the LiDAR sensors to the terrain classification and stairs detection algorithms. This architecture integrates two types of LiDARs, Cyglidar D1 and RoboSense RS-LiDAR-M1, into a unified framework that operates within the Robot Operating System (ROS). The flow of data from sensor drivers to the final prediction stage demonstrates the modularity and flexibility of the system, allowing for easy adaptation to different sensor configurations and algorithms.

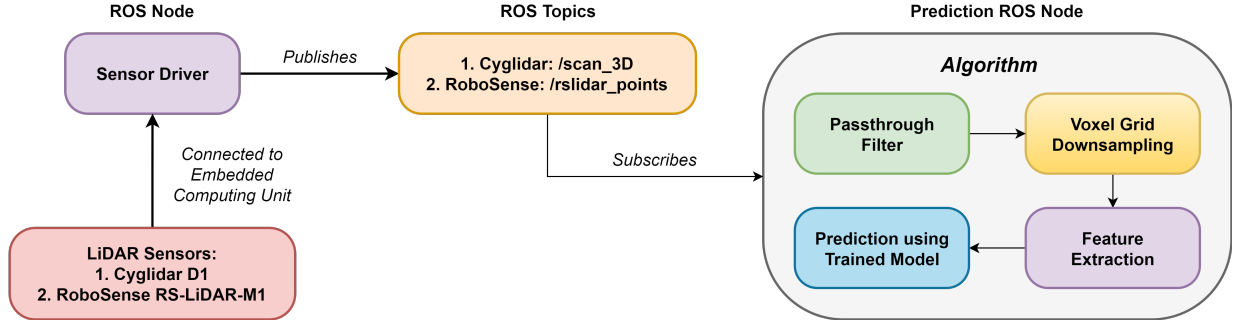


Figure 3.1: Software ROS Architecture Flow for Stairs Detection and Real-time Terrain Classification.

The figure is composed of two primary components:

- **LiDAR Sensors:** The sensors explained above are connected here with the embedded computing unit of the data acquisition system. The computing unit is powered by a battery to ensure portability of the system.
- **Sensor Driver ROS Node:** The sensor driver manages data acquisition from the LiDAR sensors. It subscribes to specific *ROS topics* based on the sensor type: the Cyglidar D1 data is streamed through the topic ‘/scan_3D’, while the RoboSense RS-LiDAR-M1 data is streamed through ‘/rslidar_points’. These drivers ensure proper handling of incoming point cloud data, converting it into a format that can be processed by the subsequent algorithms.
- **Prediction ROS Node:** The algorithm component in the prediction node is the main component running on the computing unit to predict the classification of stairs/terrains. The algorithm shown in the figure provides a brief high-level overview of the processing pipeline, emphasizing the main components of the algorithm developed in this research. This includes:

- **Passthrough Filter:** This filter processes the raw point cloud data by limiting the range of points based on their spatial coordinates. This step removes points outside the region of interest, such as points too far away or too close, ensuring that only relevant data are processed.
- **Voxel Grid Downsampling:** To further reduce the computational load, voxel grid downsampling is applied, which reduces the density of the point cloud while retaining essential geometric features. This step is particularly useful in resource-constrained environments like the Jetson Nano, ensuring efficient real-time processing.
- **Feature Extraction:** Depending on the application (terrain classification or stairs detection), specific features are extracted from the downsampled point cloud. For terrain classification, surface normals and geometric properties are key features, while for stairs detection, planar segmentation and elevation characteristics are more relevant. These features are critical for accurate predictions and will be elaborated in subsequent sections of this chapter.
- **Prediction using Trained Model:** Once the relevant features are extracted, they are passed to a pre-trained machine learning model (SVM) that classifies the terrain or detects stairs. The model outputs the prediction in real-time, providing immediate feedback to the system.

It is important to note that the algorithm depicted in the flowchart is presented in a generalized form. The specific algorithm used varies slightly depending on whether it is applied to stairs detection or terrain classification. For instance, terrain classification primarily focuses on analyzing surface normals and spatial consistency, whereas stairs detection emphasizes geometric features such as step height and depth. The differences in features and processing steps will be discussed in more detail in later sections of this chapter.

The architecture is optimized for deployment on embedded platforms such as the NVIDIA Jetson Nano, ensuring a balance between computational efficiency and classification accuracy. The design is adaptable to different LiDAR sensors and applications, making it versatile for various real-time assistive technologies and mobile robotics use cases.

3.2 Algorithm Development for Stairs Detection

The development of the stair detection algorithm focuses on harnessing LiDAR data to accurately identify stair-like features in various environments. This process utilizes Principal

Component Analysis (PCA) to effectively extract significant features from the data, which are crucial for the subsequent application of a Support Vector Machine (SVM) model for classification. The SVM model is trained using a carefully curated dataset that includes labeled instances of both stairs and non-stairs, towards building the capability of distinguishing between these classes with high precision.

For this research, data were collected using both low-accuracy and high-accuracy LiDAR sensors to provide a comprehensive understanding of how different technologies perform under similar testing conditions. It is essential to note that this thesis primarily aims to investigate whether lower-accuracy LiDAR can effectively detect stairs. This approach aligns with the broader research goal of creating a cost-effective solution, not in terms of cost, but also in terms of computational power required to process the data and run the trained model. By focusing on lower-accuracy LiDAR, the study aims to demonstrate that effective stairs detection can be achieved without the need for expensive high-end hardware, thereby reducing the overall resource footprint of the technology.

This investigation is particularly pertinent given the varied stair structures encountered in typical environments, which present challenges in terms of generalizable detection and classification. The research carefully evaluates the performance of lower-accuracy LiDAR in detecting these structures against a backdrop of potentially noisy data. The findings of these evaluations are expected to contribute significantly to the field by providing insight into the minimum specifications required for reliable stairs detection systems, and a wider range of applications.

In sum, the stair detection algorithm serves as a key component of the research, pushing the boundaries of what can be achieved with less expensive LiDAR systems. This endeavor not only investigates the limits of lower-cost technology in performing complex recognition tasks but also sets a foundation for future studies aimed at optimizing automated detection systems in varied architectural environments.

In this preliminary investigation, the focus is on identifying ascending stairs, although the framework can be adapted to recognize other stair structures with appropriate adjustments to the training data. Successful stairs identification requires clearly defined test conditions that outline when a set of LiDAR data should be classified as ascending stairs.

In this study, the focus is limited to ascending stairs due to the distinct challenges associated with detecting descending stairs. Descending stairs introduce complexities such as reduced vertical field of view, additional noise from elevation changes, and higher variability in user movement patterns. These factors complicate reliable feature extraction and increase the risk of misclassification. Moreover, descending stairs pose greater safety implications; a failure to detect descending stairs accurately could result in significant hazards,

such as falls. By focusing on ascending stairs, which offer a more controlled scenario, this preliminary investigation ensures robustness in the detection framework while laying the groundwork for future expansions to include descending stairs.

The detection process is influenced by two main factors: the angle of approach and the range of approach to the stairs. For angle of approach, it is assumed that an individual walking parallel to a set of stairs without intent to ascend (i.e. walking around stairs) should not trigger a stair detection event. Therefore, the system is designed to recognize stairs only when the individual approaches them within a specific angular range. This range is defined as a 45-degree arc emanating from the point perpendicular to the first stair. LiDAR data collected within this arc is labeled as ascending stairs during training, ensuring that the system can accurately detect stairs when approached directly. Figure 3.2 illustrates the collection arc used in this study.

In addition, the range threshold is set to optimize the detection capabilities of the low-accuracy LiDAR, which has a maximum scanning range of 2 meters. To ensure that stairs are detected in a timely manner, the range threshold is set at 1.8 meters, approximately 2-3 steps away. This buffer below the maximum range accounts for potential inaccuracies or noise in the data, allowing the algorithm to begin processing the point cloud as soon as the LiDAR detects an object within this range. This ensures that the first step of the staircase is captured effectively, providing the system with sufficient data to accurately identify the staircase structure as the user approaches.

3.2.1 Training Dataset

In this research, a meticulously curated and diverse set of LiDAR data was collected from an extensive array of staircases to ensure that the detection algorithms could be trained to achieve high levels of accuracy and robustness in varied real-world conditions. The dataset was designed to cover a broad spectrum of scenarios including stairs constructed from a variety of materials, such as wood, metal, concrete, and carpet. Each of these materials presents unique challenges for LiDAR-based detection due to their differing light reflectance properties, which can significantly impact the accuracy of depth and shape perception in the collected data.

In addition, the architectural design of stairs varied significantly, including varying numbers of steps, stair widths, and a range of angles of inclination. Acquiring a diverse dataset was crucial for training the algorithms to handle the wide array of stair structures encountered in everyday environments, from narrow, steep staircases in older buildings to wider, more gradual ones. As illustrated in Fig. 3.3, the training data included a wide

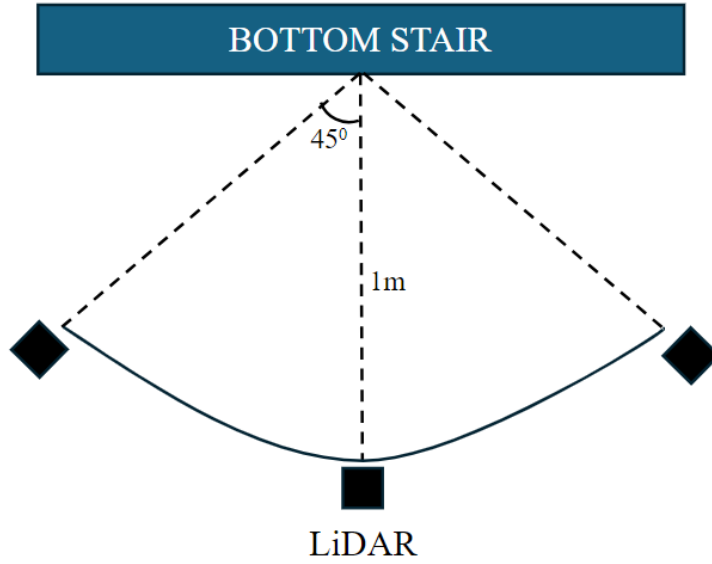


Figure 3.2: Stair detection threshold arc used to define collection labelling

selection of stairs, showcasing a variety of materials and designs from wooden and metal stairs to carpeted and industrial ones. Each type was chosen for its unique design and material characteristics to enhance the diversity of data and robustness of the stairs detection algorithms. By incorporating a broad spectrum of staircases, the aim was to develop algorithms capable of generalizing effectively across different conditions and configurations reflecting diverse real-world scenarios.

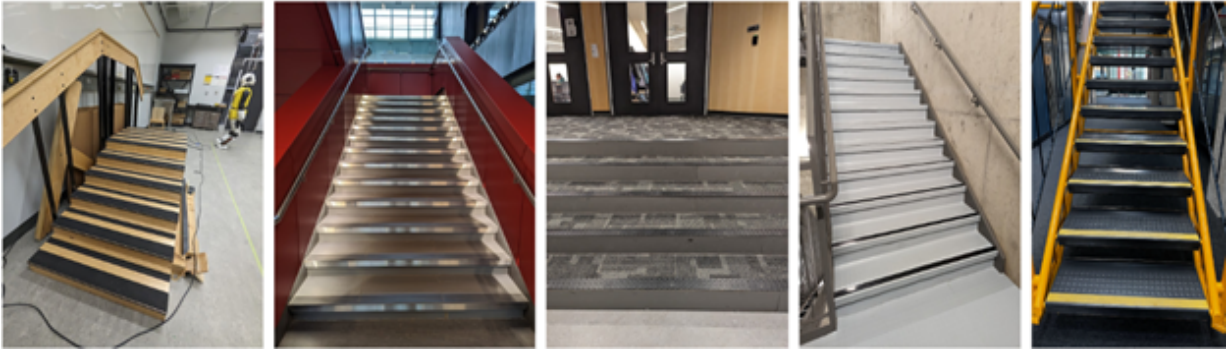


Figure 3.3: Diverse Staircase Dataset for LiDAR Training

To further enhance the comprehensiveness of the dataset, LiDAR data was captured from multiple distances and angles within the detection threshold arc shown in Fig. 3.2.

This approach was crucial for simulating real-world scenarios where users might approach stairs from various directions and distances. By varying the data collection angles and distances in training data, the algorithms are more likely to generalize across different approach trajectories, a key factor in real-world applicability.

Moreover, to account for the dynamic nature of human movement, data were captured using a wearable LiDAR format, where the sensor was mounted on a person walking towards the stairs, ascending the steps, and navigating the initial climb. This introduced natural variability in the data due to human movements, such as vibrations and shifts in orientation. This variability is essential for training algorithms that need to perform reliably in practical, everyday settings, where stability cannot be guaranteed.

Additionally, to simulate both static and dynamic use cases, data were collected using a stationary LiDAR mounted on a tripod, as well as in a wearable format. The stationary setup provided clean, stable data, ideal for capturing the precise details of stair geometry, while the wearable setup introduced real-world complexities, such as shaking and vibrations associated with a person in motion. By combining data from these varied conditions, the study aimed to develop algorithms that are not only accurate but also robust enough to handle the challenges posed by dynamic environments.

The collection periods ranged from 30 seconds to 5 minutes. These periods were chosen to ensure a variance in the information collected, accounting for natural movements and adjustments, such as breathing. Data were gathered in both wearable and static tripod formats. The LiDAR stream, received as 'sensor_msgs::PointCloud2' according to the ROS framework, was stored in ROSbags. During training, each frame was analyzed, with additional frames providing stronger training against noise. This approach to data collection was chosen to distinguish between actual stair patterns and noise. All together, the dataset comprises of 274,510 frames, equivalent to approximately 305.01 minutes of LiDAR data. An example of the collected scenes is shown in Fig. 3.4.

This comprehensive approach to data collection, which spans different materials, structural designs, distances, angles, and dynamic conditions, was critical to generate and study algorithms for deployment in assistive technologies. Such technologies, including exoskeletons and mobility aids, rely heavily on precise and reliable environmental perception to improve user safety and autonomy. Representative examples of data collection scenarios, highlighting the diversity and thoroughness of the process, are depicted in the accompanying figure, showcasing the depth of the research effort in preparing a dataset that can support robust and reliable staircase detection in real-world applications.

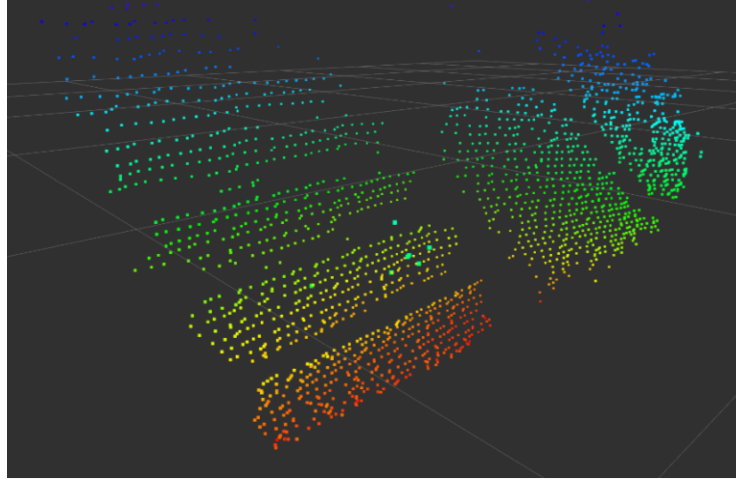


Figure 3.4: Example collection scene in RViz displaying LiDAR scenes for a metal stairwell.

3.2.2 Algorithm

The flowchart presented in Figure 3.5 outlines the step-by-step methodology for the staircase detection algorithm utilizing LiDAR data. The process includes data acquisition, pre-processing, feature extraction, and machine learning model training. The following is a detailed breakdown of each stage, including the mathematical and computational techniques involved.

I Stairs & Non-Stairs Data ROSbags: The data acquisition process begins with collecting LiDAR data in ROSbags (Robot Operating System bags), which store message data such as sensor readings, allowing for organized and reusable datasets. ROS is an open-source framework that provides a collection of tools, libraries, and conventions for developing robot applications. Within this ecosystem, ROSbags serve as a flexible, convenient file format for storing and managing message data from various sources, including sensor readings, robot state information, and control commands [63, 37]. ROSbags offer a structured way to organize complex, multi-modal data and provide reusability, time synchronization, and efficient storage [63, 67]. The use of ROSbags ensures flexibility, allowing developers to replay recorded data, crucial for iterative development, testing, and debugging systems [63, 67]. Importantly, ROSbags maintains temporal relationships between different data streams and support a wide variety of ROS message types, making them versatile for various robotics applications [63, 49].

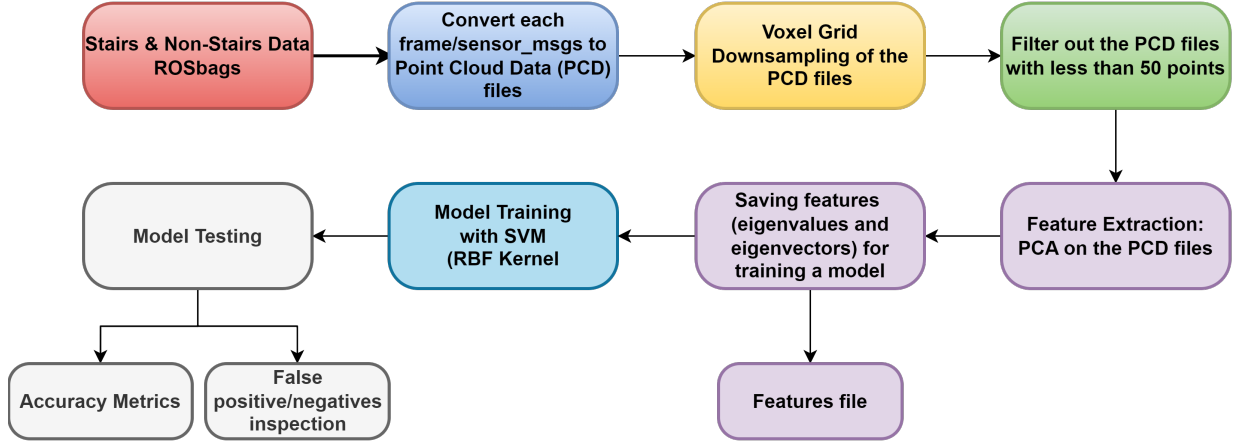


Figure 3.5: Flowchart of the stair detection algorithm development process using LiDAR data, from data acquisition to model evaluation.

II Convert Each Frame/Sensor_msgs to Point Cloud Data (PCD) Files: The collected ROSbag data is then converted into Point Cloud Data (PCD) files. PCD is a widely used data format specifically designed for handling 3D point clouds, which represent spatial information captured by sensors like LiDAR [53, 11]. The PCD format provides a structured way to store and manipulate 3D data, facilitating processing and analysis, particularly in robotics and computer vision applications [53]. This conversion step is crucial to standardize the data, and ensures compatibility with the Point Cloud Library (PCL) and other processing frameworks, thereby facilitating efficient data handling, storage, and further analysis [11].

III Voxel Grid Downsampling of the PCD Files: Voxel grid downsampling is a widely used method in point cloud processing to reduce the number of points while preserving the overall structure of the data. A voxel grid is a 3D grid that divides the space into smaller, equally sized cubes (voxels). All the points within a voxel are approximated by their centroid, which significantly reduces the number of points, enhancing computational efficiency.

Mathematical Basis: In this study, the Point Cloud Library (PCL) is used for voxel grid downsampling [48]. PCL offers efficient implementations of various algorithms, including voxel grid filtering, which reduces the computational load without losing critical spatial details [11].

Equation: For a point cloud $\mathcal{P} = \{p_1, p_2, \dots, p_n\}$, where each point $p_i = (x_i, y_i, z_i)$, voxel grid downsampling can be represented by dividing the space into voxels of size

$\delta x, \delta y, \delta z$. The downsampled point $p'_j = (\bar{x}_j, \bar{y}_j, \bar{z}_j)$ within each voxel V_j is computed as:

$$\bar{x}_j = \frac{1}{|V_j|} \sum_{p_i \in V_j} x_i, \quad \bar{y}_j = \frac{1}{|V_j|} \sum_{p_i \in V_j} y_i, \quad \bar{z}_j = \frac{1}{|V_j|} \sum_{p_i \in V_j} z_i \quad (3.1)$$

This equation (Equation 3.1) represents the process of averaging the coordinates of all points within a voxel to compute the centroid, which then serves as the representative point for that voxel. The variables $\bar{x}_j$, $\bar{y}_j$, and $\bar{z}_j$ are the coordinates of the centroid of voxel V_j , and $|V_j|$ is the number of points within that voxel. The purpose of this operation is to reduce the number of points in the point cloud while preserving the overall structure and important spatial features of the data.

Figure: The figure below demonstrates the voxelization process, where each grid cell represents a voxel, and the black dots are the original points. The red stars indicate the centroids of the points within each voxel, which are used to represent the voxel in the downsampled point cloud. The figure is adapted from the Libpointmatcher documentation [34].

Using the Point Cloud Library (PCL), the downsampling process is efficiently implemented. The integration of OpenMP for parallel processing further accelerates this computation. By distributing the work across multiple cores, OpenMP allows the voxel grid downsampling to be performed in parallel, which is especially beneficial when dealing with large datasets or when real-time processing is required. This parallelized approach significantly reduces computation time, making the system more responsive and capable of handling high-throughput data streams in real-time applications [48].

OpenMP Parallelization: To expedite the downsampling process, OpenMP (Open Multi-Processing) was employed to parallelize the computation across multiple CPU cores [7]. OpenMP is an API that supports multi-platform shared memory multiprocessing programming. It enables the distribution of loop iterations across threads, thereby dividing the workload and significantly reducing execution time.

In the context of voxel grid downsampling, the point-cloud data is divided into smaller segments, each processed in parallel. This is achieved by parallelizing the loop that iterates over the point-cloud data points and assigns them to their respective voxels. Each thread independently computes the centroid for its assigned voxel, which is then synchronized across the threads to form the final downsampled point cloud. The parallelized approach is crucial for real-time applications, where processing speed is paramount [35].

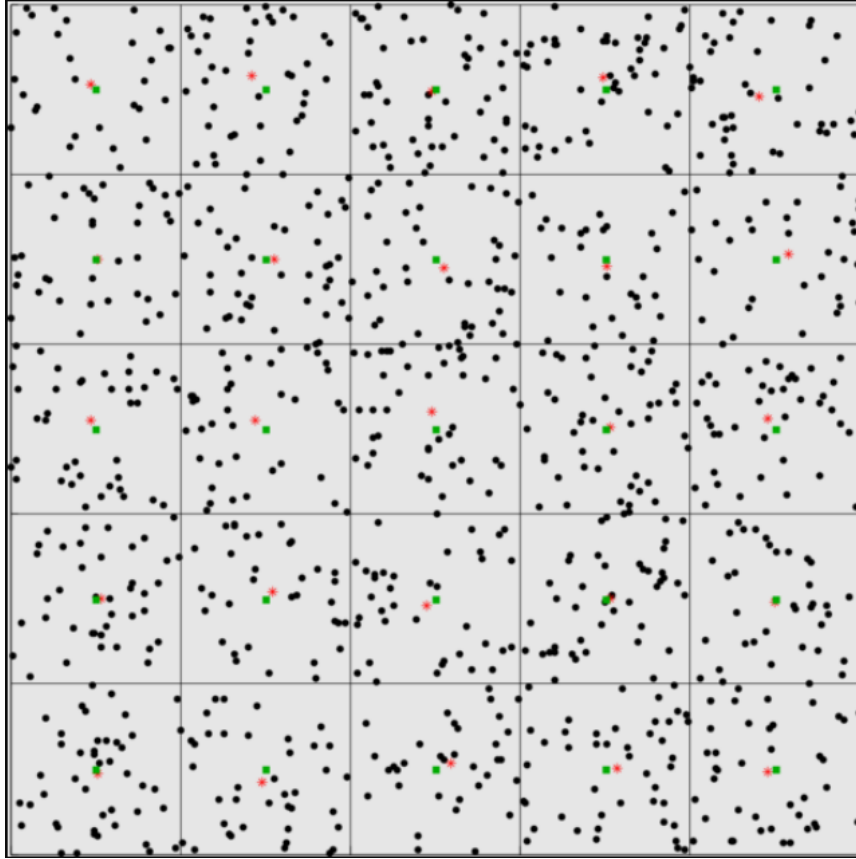


Figure 3.6: Voxelization Process: Original points (black dots) and their centroids (red stars) within each voxel.

OpenMP’s directive-based model makes it easier to implement parallelism in existing code without extensive modifications. By using OpenMP, the voxel grid downsampling process achieves a significant speedup, making it viable for deployment in real-time systems where large point cloud datasets need to be processed quickly and efficiently.

IV Filter Out the PCD Files with Less Than 50 Points: After downsampling, point clouds with fewer than 50 points are filtered out. This threshold is essential to eliminate frames with insufficient data that may negatively impact the accuracy of the classification model. Removing these small point clouds ensures that the subsequent stages of the process work with meaningful and relevant data.

V Feature Extraction, PCA on the PCD Files: Principal Component Analysis (PCA) is a powerful statistical method used for dimensionality reduction. In the context of point clouds, PCA is employed to extract the principal components that represent the main axes of variation in the data.

Mathematical Basis: The mathematical foundation of PCA begins with the computation of the covariance matrix of the point cloud data. The covariance matrix C captures the variance and correlation between the different dimensions (X, Y, Z) of the point cloud. For a given point cloud $\mathcal{P} = \{p_1, p_2, \dots, p_n\}$, where each point $p_i = (x_i, y_i, z_i)$, the covariance matrix C is calculated as follows:

$$C = \frac{1}{n-1} \sum_{i=1}^n (p_i - \bar{p})(p_i - \bar{p})^T \quad (3.2)$$

where $\bar{p}$ is the mean of all points in the cloud.

The next step involves solving the eigenvalue problem for the covariance matrix:

$$Cv = \lambda v \quad (3.3)$$

In this equation, v represents the eigenvectors and λ represents the corresponding eigenvalues. The eigenvectors v are the principal components that define the new coordinate system, oriented along the directions of maximum variance in the data. The eigenvalues λ indicate the amount of variance captured by each principal component [23].

By projecting the original point cloud data onto these principal components, PCA effectively reduces the dimensionality of the data [68]. In terrain classification, the first few principal components usually capture most of the variance, allowing the data

to be represented in a lower-dimensional space without significant loss of information. This reduced representation is commonly used as input features for machine learning models, such as Support Vector Machines (SVMs).

Applications of PCA in literature: PCA has been widely used in the processing and analysis of 3D point clouds [68]. It is particularly useful in applications where high-dimensional data needs to be simplified for further analysis. For instance, in the context of terrain classification, PCA helps focus on the most informative features of the landscape, enhancing the accuracy and efficiency of the classification algorithms.

PCA has been extensively used in LiDAR data processing for dimensionality reduction techniques and to enhance efficiency and accuracy. Duan et al. (2021) addressed the inherent noise in LiDAR-generated point clouds, proposing a novel method that takes advantage of PCA to reduce the dimensionality of the data from 3D to 2D. By focusing on the principal components that capture the most significant variance, this approach effectively simplifies the noise reduction process. The transition to 2D data allows for more efficient clustering and filtering, ultimately preserving essential environmental features while significantly lowering computational complexity. This method not only improves the precision and recall of noise reduction algorithms, but also reduces the computational burden, making it more suitable for real-time applications [12].

Hackel et al. (2017) introduced a large-scale 3D point cloud classification dataset designed to push the boundaries of deep learning methods in this domain. Although the paper does not explicitly detail the use of PCA, it is implied that dimensionality reduction techniques like PCA could be instrumental in handling the massive scale of the dataset, which comprises over 4 billion manually labeled points. By reducing the dimensionality of the data, PCA would help streamline the learning process, allowing deep learning models to focus on the most relevant features and improving their generalization across diverse urban and rural scenes. The paper emphasizes the need for such techniques to make deep learning models more computationally feasible and effective in processing large, complex 3D datasets [19].

Together, these studies highlight the critical role of Principal Component Analysis (PCA) in the processing and classification of 3D point-cloud data. PCA is instrumental in reducing data complexity, facilitating noise reduction, and enhancing the performance of machine learning models. By simplifying the data to its most significant components, PCA enables more accurate and efficient data processing, making it highly applicable to real-world scenarios, whether it be noise filtering for LiDAR data or in the classification of extensive 3D datasets.

In this research, PCA is applied to the Point Cloud Data (PCD) files to extract the

principal components that capture the most crucial variations in the data. These components are then used as features for subsequent machine learning models. This approach not only reduces computational complexity, but likely also bolsters the system’s robustness across a range of environmental conditions, ensuring reliable performance in diverse real-world applications.

VI Saving Features (Eigenvalues and Eigenvectors) for Training a Model: The eigenvalues, which quantify the variance along each principal component, and the eigenvectors, which define the directions of these components, are systematically stored in a feature matrix. This matrix serves as a compact and powerful representation of the original data, significantly reducing the computational burden while preserving the key information necessary for accurate classification.

During the training phase, this feature matrix is utilized as input for machine learning algorithms. Optimized storage of these PCA-derived features not only enhances data handling efficiency but also ensures quick retrieval, which is vital for real-time processing [68].

VII Model Training with SVM (RBF Kernel): Support Vector Machines (SVM) are a class of supervised learning algorithms widely used for classification tasks, particularly in challenging applications. The core idea behind SVM is to find an optimal hyperplane that maximizes the margin between different classes in a high-dimensional feature space, which is crucial for tasks requiring high accuracy in distinguishing between complex structures like stairs and non-stairs [8, 66].

Initially, the SVM was tested with a linear kernel to determine if a straightforward linear separation could accurately classify the data. The linear kernel attempts to separate the data using a straight line (or hyperplane) in the feature space, mathematically represented as:

$$K(x_i, x_j) = x_i^T x_j \quad (3.4)$$

However, due to the complex, non-linear nature of stairs detection in 3D point clouds, a linear SVM was implemented and deemed inadequate in performance. The classification boundary required non-linear separation to capture the intricate relationships between the eigenvalues and eigenvectors derived from the point cloud data.

To address this, the Radial Basis Function (RBF) kernel was employed, which is particularly effective for nonlinear classification problems. The RBF kernel is defined as:

$$K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2) \quad (3.5)$$

Where:

- x_i and x_j are the feature vectors in the input space, derived from the eigenvalues and eigenvectors of the PCA analysis on the point cloud data.
- γ (gamma) is a hyperparameter that controls the influence of each training example. A lower value γ results in a smoother decision boundary, while a higher value γ allows a more complex adaptive boundary that fits the training data more closely.

The SVM with the RBF kernel transforms the input data, consisting of all eigenvalues and eigenvectors for the 3D axes, into a higher-dimensional space where a linear separation becomes possible. This transformation is particularly beneficial when the stairs and non-stair classes are not linearly separable in the original feature space [57, 6]. By leveraging the RBF kernel, the model captures the non-linear relationships between the features, leading to improved classification performance.

The decision function for the SVM classifier with the RBF kernel is expressed as follows:

$$f(x) = \text{sign} \left(\sum_{i=1}^n \alpha_i y_i K(x_i, x) + b \right) \quad (3.6)$$

Where:

- α_i are the Lagrange multipliers that determine the weight of each support vector.
- y_i are the class labels (stair or non-stair).
- b is the bias term.

This decision function enables the SVM to classify new data points by evaluating their position relative to the hyperplane, considering the non-linear transformations introduced by the RBF kernel. The results of using both linear and RBF kernels will be discussed in detail in Chapter 4, where their performance in stair detection is analyzed, and demonstrates the advantages of using the RBF kernel for this application.

Hyperparameter Tuning

To optimize the SVM model for stairs detection, a comprehensive hyperparameter tuning process was conducted using the *RandomizedSearchCV* method. This involved testing a wide range of values for several key parameters using a 5-fold cross-validation approach, ensuring the model's robustness across different subsets of the dataset.

Tested Hyperparameters:

- **Kernel:** 'linear', 'rbf', 'poly', 'sigmoid'
- **C (Regularization Parameter):** 0.001, 0.01, 0.1, 1, 10, 100, 1000
- **Gamma:** 'scale', 'auto', 0.0001, 0.001, 0.01, 0.1, 1
- **Degree:** 2, 3, 4, 5 (Only applicable for polynomial kernels)
- **coef0:** 0.0, 0.1, 0.5, 1.0 (Only applicable for polynomial and sigmoid kernels)

The parameter grid was designed to cover a broad spectrum of kernel functions and regularization strengths. By testing different combinations of gamma values, degrees (for polynomial kernels), and coefficients, the aim was to identify the optimal configuration for distinguishing between stairs and non-stairs.

RandomizedSearchCV was selected over GridSearchCV to allow for a more efficient exploration of the hyperparameter space, as it randomly samples a fixed number of parameter combinations rather than exhaustively testing all possible combinations. This approach helped reduce computational overhead while still providing reliable results. By using cross-validation, the model's performance was evaluated on different data folds, ensuring that it generalized well and avoided overfitting.

The hyperparameter tuning process was critical in selecting the best parameters, which were later applied in the final model for more detailed analysis in the results chapter.

Support Vector Machines: Hard Margin vs. Soft Margin

For the SVM classifier to find an optimal hyperplane that separates different classes, two main approaches can be utilized: hard margin SVM and soft margin SVM.

- **Hard Margin SVM:** In hard margin SVM, the goal is to find a hyperplane that perfectly separates the classes without allowing any misclassification. This approach is effective when the dataset is linearly separable and free from noise, as it maximizes the margin between the closest data points (support vectors) and

the hyperplane. However, hard margin SVMs are sensitive to outliers, as even a single misclassified point can lead to an infeasible solution.

- **Soft Margin SVM:** In soft margin SVM, the model introduces slack variables to allow for some misclassification or margin violations. This makes the algorithm more robust to noisy or overlapping datasets by striking a balance between maximizing the margin and minimizing classification errors. The trade-off is controlled by the regularization parameter C , which adjusts the model’s tolerance for misclassification.

In this thesis, a **hard margin SVM** was chosen for terrain classification and stairs detection. This decision was based on the assumption that the datasets used (e.g., RoboSense LiDAR data) exhibit high-quality, linearly separable features after preprocessing and noise injection. By using a hard margin SVM, the classifier maximizes the separation between terrain classes, which is particularly advantageous when the dataset is well-conditioned.

However, it is important to note the potential limitations of this approach. In scenarios involving higher levels of noise or overlapping data distributions, a soft margin SVM may provide better generalization by mitigating the effects of outliers and margin violations. Future work could involve exploring soft margin SVMs to evaluate their performance under varying noise conditions.

The choice of a hard margin SVM aligns with the primary objective of this research: achieving high classification accuracy with minimal computational overhead, as hard margin SVMs eliminate the need for additional regularization tuning. This simplification was especially critical for real-time applications on embedded systems like the Jetson Nano, where computational resources are limited.

VIII Model Testing and Accuracy Evaluation: The evaluation of the trained SVM model was conducted in two distinct phases to thoroughly assess its performance. In the first phase, the model was tested offline using a high-end computer (detailed specifications provided in the following chapter) to understand its generalization capabilities on unseen data. This offline testing phase was crucial to determine how well the model performed with diverse datasets and in different scenarios that simulate real-world environments. The trained model’s accuracy, precision, recall, and F1 scores were calculated to provide an initial understanding of its effectiveness.

In the second phase, the trained model was deployed on an NVIDIA Jetson Nano to evaluate its real-time (or online) performance. This phase aimed to test the algorithm

in a live environment where computational resources are limited compared to the high-end offline setup. By running the model on the Jetson Nano, the practical feasibility of the algorithm for deployment in assistive technologies was assessed.

Accuracy metrics and inspection of false positives and negatives played a critical role in further refinement of the model. During both phases, performance was evaluated using standard accuracy metrics, including precision, recall, and the F1 score. Additionally, detailed logs were maintained to track the model’s behavior, especially focusing on instances of false classifications. These logs provided insight into specific frames where the model failed to classify correctly, allowing for an in-depth analysis of these errors. The information saved in these logs is crucial for iterative refinement of the model, enabling targeted improvements to enhance accuracy and reducing false classifications in future iterations.

3.3 Algorithm Development for Terrain Classification

The algorithm developed for terrain classification is designed to differentiate between two types of terrains: plain and grass. This classification is based on the orientation of the normal vectors extracted from the point clouds generated by the LiDAR sensors. Data were collected using Cyglidar D1 and RoboSense M1 sensors, to permit evaluation across different sensors.

The algorithm process is visually represented in the flowchart (Fig. 3.7). This process involves several critical steps, including data collection, Gaussian noise addition, passthrough filtering, voxel grid downsampling, normal extraction, feature selection, model training, and cross-validation. Each step is carefully designed to ensure that the algorithm can accurately classify the terrain, even in the presence of noise and varying environmental conditions.

The proposed algorithm follows these key steps:

- I **Data Collection:** LiDAR data for both plain and grass terrains were collected using Cyglidar D1 and RoboSense M1 sensors. The collected data is stored in ROSbags, providing the flexibility to replay the data multiple times during the algorithm’s development and testing phases [67].
- II **Gaussian Noise Addition:** The addition of Gaussian noise to the point cloud data simulates the conditions of a low-accuracy LiDAR sensor. Low-accuracy LiDARs

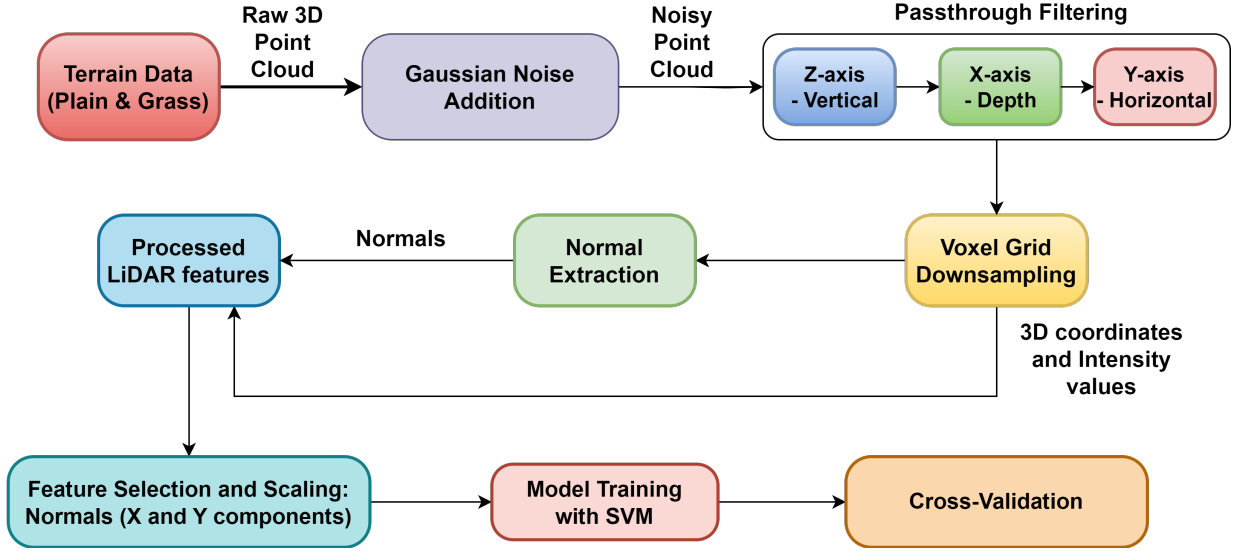


Figure 3.7: Flowchart of the terrain classification algorithm development process using LiDAR data.

typically produce sparser point clouds compared to high-accuracy sensors due to lower resolution and precision [62, 48]. To accurately replicate these conditions, the process involves two critical steps: voxel grid downsampling and the subsequent addition of Gaussian noise.

Point-Cloud Density Reduction: To simulate the output of a lower-resolution LiDAR sensor, voxel grid downsampling is applied to the high-density point cloud data. This process involves dividing the point cloud space into cubic voxels of fixed size and then averaging the coordinates of all points within each voxel to represent the voxel by a single point—its centroid. For this implementation, a voxel size of $0.05 \times 0.05 \times 0.05$ meters (5 cm cubic dimensions) is chosen to achieve the desired reduction in point density.

The downsampling operation is mathematically described by Equation 3.1, where the coordinates of the downsampled point $\bar{x}_j$, $\bar{y}_j$, and $\bar{z}_j$ within each voxel V_j are computed by averaging the coordinates of all points within that voxel. A voxel size of 5 cm was selected empirically, this process effectively reduces the number of points in the point cloud, simulating the effect of a low-resolution LiDAR while maintaining the essential geometric structure necessary for further analysis [11].

Mathematical Basis of Gaussian Noise: Following downsampling, Gaussian noise

is added to the point cloud data. The Gaussian distribution, also known as the normal distribution, is defined mathematically by the probability density function (PDF):

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \quad (3.7)$$

where:

- x is the variable,
- μ is the mean of the distribution (typically set to 0 for noise addition),
- σ is the standard deviation, which determines the spread or 'width' of the distribution.

In this context, the Gaussian noise is added to each point's coordinates in the down-sampled point cloud. The noise is generated using the standard normal distribution with a mean of 0 and varying levels of standard deviation (σ), which corresponds to the intensity of the added noise. The standard deviation values used range from 4 to 10 mm, incremented by 2 mm to simulate different levels of noise. The mathematical operation to add Gaussian noise can be expressed as:

$$x_{\text{noisy}} = x + \mathcal{N}(0, \sigma_x) \quad (3.8)$$

$$y_{\text{noisy}} = y + 2\mathcal{N}(0, \sigma_y) \quad (3.9)$$

$$z_{\text{noisy}} = z + \mathcal{N}(0, \sigma_z) \quad (3.10)$$

Where $\mathcal{N}(0, \sigma)$ represents the normal distribution with a mean of 0 and standard deviation σ . In the provided code, this process is implemented by generating random values from the normal distribution and adding these values to the x, y, and z coordinates of each point in the cloud. The inclusion of different scaling factors for the y-coordinate, such as the factor of 2 in the noise addition, is intended to reflect the inherent variability in different axes due to sensor characteristics or specific environmental factors.

This method permits examination under varying levels of noise in the data, which is critical for ensuring robustness and accuracy in terrain classification tasks under different operational conditions [35, 5].

III Passthrough Filtering: A passthrough filter is applied sequentially along the Z, X, and Y axes to isolate the region of interest within the LiDAR data. This filtering step effectively removes irrelevant data points, ensuring that only the points corresponding to the terrain of interest are retained for further processing. The passthrough filtering operation can be mathematically described as follows:

For the Z-axis:

$$P(z_i) = \begin{cases} 1 & \text{if } Z_{\min} \leq z_i \leq Z_{\max} \\ 0 & \text{otherwise} \end{cases} \quad (3.11)$$

For the X-axis:

$$P(x_i) = \begin{cases} 1 & \text{if } X_{\min} \leq x_i \leq X_{\max} \\ 0 & \text{otherwise} \end{cases} \quad (3.12)$$

For the Y-axis:

$$P(y_i) = \begin{cases} 1 & \text{if } Y_{\min} \leq y_i \leq Y_{\max} \\ 0 & \text{otherwise} \end{cases} \quad (3.13)$$

Here, $P(x_i)$, $P(y_i)$, and $P(z_i)$ represent the passthrough filter operations on the respective coordinates x_i , y_i , and z_i of each point in the point cloud. The variables $[Z_{\min}, Z_{\max}]$, $[X_{\min}, X_{\max}]$, and $[Y_{\min}, Y_{\max}]$ define the filtering ranges along the Z, X, and Y axes, respectively, within which points are retained. Points outside these specified ranges are excluded from further analysis [47].

The filtering process is executed sequentially—starting with the Z-axis, followed by the X-axis, and finally the Y-axis. This order is strategically chosen to maximize the removal of unwanted points at each step, making the algorithm more computationally efficient. By reducing the number of points that need to be processed in each subsequent step, the computational load decreases significantly, which influences real-time processing performance. The use of the Point Cloud Library (PCL) to implement this filtering process provides a robust and efficient solution, enabling the handling of large datasets with high computational efficiency [47].

IV Voxel Grid Downsampling: Voxel grid downsampling is applied to reduce the density of the point clouds while preserving their overall structure. This step involves dividing the space into smaller, equally sized cubes (voxels) and approximating all points within a voxel by their centroid [48]. The mathematical process for this downsampling, as previously described in Equation 3.1 in Section 3.2.4 (III) of the stairs detection section, involves averaging the coordinates of all points within a voxel to

determine the centroid, which then represents the entire voxel in the downsampled point cloud.

V Normal Extraction: Normal vectors are extracted from the downsampled point cloud data to analyze the surface orientation. This process is critical for terrain classification as it provides information about the geometric structure of the surface, which is essential for distinguishing between different types of terrain.

Mathematical Basis: The normal extraction process is based on Principal Component Analysis (PCA), which is used to compute the normal vectors at each point in the point cloud. For each point, the covariance matrix of its neighboring points is calculated, capturing the variance and correlation between the coordinates. This covariance matrix C is then used to solve the eigenvalue problem, as previously mentioned in Equation 3.3:

$$C\mathbf{v} = \lambda\mathbf{v}$$

Here, $\mathbf{v}$ represents the eigenvectors, and λ represents the corresponding eigenvalues. The eigenvector associated with the smallest eigenvalue is selected as the normal vector for that point, as it indicates the direction of least variance, which is perpendicular to the surface.

In this study, we focus specifically on the X and Y components of the normal vectors, corresponding to the depth direction (X) and the left-right direction (Y). These components are sufficient to distinguish between plain and grass terrains, as they reflect the surface’s inclination and texture, which vary distinctly between these two terrain types.

To handle the large volume of data and ensure efficient computation, the normal extraction process is parallelized using OpenMP [4]. Please refer to Section 3.2.4 (III) (OpenMP Parallelization) section. The Point Cloud Library (PCL) is utilized for implementing this process, leveraging its robust and optimized algorithms for point cloud processing [54].

VI Feature Selection and Scaling: After extracting the normal vectors from the point cloud data, the X and Y components are specifically chosen for further processing. This selection is based on their relevance to the terrain classification task, as these components effectively capture the critical variations in surface orientation [17, 22].

To ensure that the selected features contribute optimally to the classification model, the data is scaled to standardize the range of the features. This step is crucial because it prevents features with larger numeric ranges from dominating those with smaller

ranges, thereby allowing all features to contribute equally to the model’s decision-making process [46]. The scaling process typically involves normalizing the data towards a mean of zero and a standard deviation of one, which enhances the performance and convergence speed of machine learning algorithms like Support Vector Machines (SVMs).

This focused selection and careful scaling of features not only streamline the computational process but also enhance the model’s ability to accurately classify different terrains based on the most informative aspects of the data.

VII Model Training with SVM: The selected features, derived from the X and Y components of the normal vectors, are used to train a Support Vector Machine (SVM) model. Given the complexities involved in terrain classification, we initially hypothesized that a non-linear kernel, such as the Radial Basis Function (RBF) kernel, would be more effective. This assumption was based on our prior work with stair detection, where non-linear kernels demonstrated superior performance in handling complex feature spaces [55].

However, after visualizing the feature distribution and considering the computational demands of real-time applications, we explored the use of a linear kernel as an alternative. The linear kernel offered a more straightforward decision boundary, which, although less flexible than the RBF kernel, resulted in significantly reduced computational overhead [8].

The trade-off between computational efficiency and classification accuracy was carefully evaluated, leading to the conclusion that the linear kernel provided a practical balance for real-time terrain classification, especially when slight compromises in accuracy could be tolerated. The detailed comparison of the SVM performance with both RBF and linear kernels, including their respective accuracy metrics, is discussed in the subsequent chapter.

3.4 Conclusion

This chapter presented the systematic approach used to develop algorithms for stairs detection and terrain classification using LiDAR technology. This involved the design and testing of algorithms that could reliably operate with low- and high-accuracy LiDAR sensors. Each algorithm was meticulously crafted, starting from data collection, followed by extensive preprocessing, feature extraction, and finally, the application of machine learning models for classification.

The chapter detailed the steps taken to ensure the robustness and efficiency of the algorithms, such as employing parallel processing techniques like OpenMP to optimize computational time. Decisions made during algorithm development, such as selecting appropriate machine learning models and kernels, were guided by a combination of empirical testing and theoretical considerations.

This methodology not only ensures the applicability of the algorithms in various settings but also lays a strong foundation for the experimental results and discussion that will follow in the next chapter. The performance of these algorithms, particularly in terms of accuracy, computational efficiency, and real-time applicability, will be critically evaluated and discussed in relation to research objectives.

Chapter 4

Experimental Setup & Datasets

This chapter provides a comprehensive overview of the system architecture, hardware configurations, and data collection protocols employed for the development and testing of the algorithms used in stairs detection and terrain classification. The purpose of this section is to establish a detailed understanding of the experimental setup, which forms the foundation for the algorithm development and validation processes.

The chapter first reports the system architecture, detailing the integration of both low-accuracy and high-accuracy LiDAR sensors within the experimental framework. The Cygbot Cyglidar D1 and RoboSense RS-LiDAR-M1 v2 are highlighted as the primary sensors used, each selected for their specific attributes that cater to different aspects of the research. The hardware specifications and configurations are then reported, including the processing/compute units, data storage, and additional components. The hardware setup is optimized to balance performance with the need for portability and adaptability in various testing environments.

The final sections focus on the data collection protocol, detailing the methods used to gather and preprocess data for model training and testing. An emphasis is placed on procedures to ensure data consistency in different testing scenarios, including descriptions of the environments in which the data were collected, types of terrain and stairs encountered, and data annotation and preprocessing.

4.1 System Architecture

The system architecture details the hardware and software configurations necessary for data collection and processing. By integrating two different LiDAR systems, low- and high-end sensors, this research provides a comprehensive examination of their capabilities and limitations. The architecture is designed to be flexible and adaptable, facilitating seamless integration into existing assistive technologies with minimal modifications.

4.1.1 Low-Accuracy LiDAR: Cygbot Cyglidar D1

I Sensor Specification: The Cygbot Cyglidar D1 is a low-cost, low-accuracy LiDAR sensor designed for consumer-level applications [9]. It features a compact form factor [Fig. 4.1, weighing only 31g, ideal for integration into wearable systems. The Cyglidar D1 operates at a wavelength of 940nm, ensuring eye safety while providing reliable measurements over a wide range of conditions. The 940nm wavelength is within the Near-Infrared (NIR) range, which is less hazardous to human eyes compared to shorter wavelengths, making it suitable for applications where user safety is required [32].

The horizontal field of view for 2D/3D is 120°, while the vertical field of view is 65°, offering comprehensive coverage suitable for assistive technologies. The measurement speed for 2D and 3D scans is 15 Hz, with a reported distance accuracy of $\pm 1\%$ over a detection range of 200mm to 7,000mm in 2D mode and 50mm to 2,000mm in 3D mode. The resolution of the sensor is 0.75° (angle) for 2D and 160 x 60 pixels for 3D, making it capable of capturing detailed spatial information necessary for navigation and obstacle detection tasks [9].

The Cyglidar D1 is designed with simplicity in mind, using a standard UART TTL 3.3V interface with a communication speed of up to 3,000,000 bps, easily integrated with various computing platforms, including the NVIDIA Jetson Nano. The sensor is powered by a 5V, 500mA input source, making it efficient in terms of power consumption. Additionally, it operates in a temperature range of -10°C to 50°C, making it suitable for a variety of environmental conditions [9].

Although the field of view of this sensor is not 360°, it is suitable for the proposed application because a wide horizontal view in front of the user can be captured using this Cyglidar D1. Moreover, solid-state LiDAR has several advantages over mechanical LiDAR, including:

- **Reliability:** With no moving parts, solid-state LiDARs (e.g., Cyglidar D1) are less prone to wear and tear and more reliable in harsh environmental conditions.

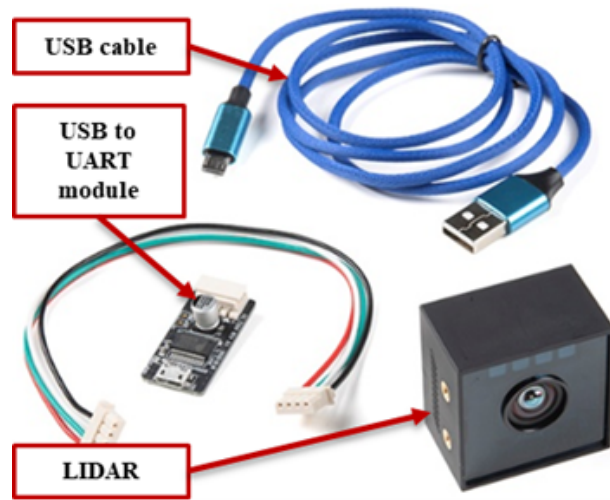


Figure 4.1: Cygbot Cyglidar D1 solid-state LIDAR with accessories [9].

Unlike mechanical LiDARs, which have rotating components that can introduce noise and degrade over time, solid-state LiDARs maintain consistent performance with minimal maintenance [32, 64].

- **Size and Weight:** Weighing only 31g, the Cyglidar D1 is exceptionally lightweight, partly due to the absence of mechanical parts. This makes it easier to integrate into compact systems, such as wearable exoskeletons, without adding significant weight [72].
- **Speed:** The Cyglidar D1 operates at 15 Hz, which means it can capture 15 frames of LiDAR data per second. While this is sufficient for slower-paced environments or static applications, it may not be ideal for dynamic scenarios where the user is moving quickly. In such cases, higher-speed data acquisition rates, such as 30 Hz or more, are often required to ensure real-time processing and accurate terrain classification or stairs detection. Faster acquisition speeds help capture more data points per second, reducing the chances of missing important features or transitions in the environment [32, 43]. Therefore, while the Cyglidar D1 is advantageous due to its compactness and solid-state design, it might face limitations in more demanding real-time applications.
- **Range:** Despite its compact size, the Cyglidar D1 offers an impressive range, which is beneficial for detecting changes in terrain or obstacles from a distance. This allows the system to adjust its response time accordingly, enhancing safety and performance [72].

- **Cost:** Solid-state LiDARs are generally more cost-effective to manufacture than mechanical LiDARs, as they have fewer components and can be produced at scale. This makes them an affordable option for integrating into consumer-level assistive devices [72].

II Computing Unit Used: Processing 3D-point cloud data generated by the LiDAR is performed on an NVIDIA Jetson Nano computing platform. This microprocessor offers several advantages for onboard computing of 3D point cloud data, making it an excellent choice for robotics and mobile systems.

The Cyglidar D1’s data processing in this study was handled solely by the CPU, as the 3D point cloud processing was performed using the PCL, which currently lacks native support for GPU acceleration. Although CPU processing was sufficient for the real-time applications discussed here, integrating GPU-based processing could potentially reduce computation times for more complex or large-scale tasks.

- **Real-time Processing:** The Jetson Nano’s quad-core ARM Cortex-A57 CPU is responsible for the real-time processing of 3D point cloud data. The Cortex-A57 architecture is based on the advanced 64-bit architecture of ARM, which is optimized for high-performance applications while maintaining energy efficiency. This makes it particularly suitable for embedded systems like the Jetson Nano, where balancing performance and power consumption is crucial.

The Cortex-A57 features advanced superscalar execution, which allows the CPU to execute multiple instructions per clock cycle. This improves overall efficiency in handling data-intensive tasks such as 3D point cloud processing, where a large number of data points must be processed in parallel. Additionally, its out-of-order execution capability enables more efficient use of the CPU’s resources, allowing for higher throughput in real-time tasks like terrain classification and stairs detection.

With its multi-core architecture, the ARM Cortex-A57 CPU can perform parallel data processing, making it possible to handle multiple point cloud frames simultaneously. This is particularly beneficial for applications requiring immediate analysis and decision-making, such as real-time obstacle detection and navigation. The combination of advanced processing techniques, 64-bit capabilities, and energy efficiency makes the Cortex-A57 a suitable choice to process 3D point cloud data in this LiDAR-based system, ensuring responsive performance without the need for GPU acceleration [42].

- **Low Latency:** The high processing power of the Jetson Nano enables it to minimize latency in data processing, ensuring that the system can quickly respond

to changes in the environment. This is especially important in dynamic scenarios, such as reacting to unexpected obstacles [25].

- **Compact Size and Energy Efficiency:** The Jetson Nano’s small form factor (measuring 69.6 x 45mm) and low power consumption (5W to 10W) make it ideal for integration into compact, energy-constrained systems. This is particularly beneficial for mobile platforms, such as drones, robots, or wearable devices, where space and power are at a premium [42].
- **Cost-effectiveness:** Compared to other high-performance computing platforms, the Jetson Nano is relatively affordable, making it accessible to a wide range of users and applications. Its cost-effectiveness, combined with its processing power, makes it an attractive option for developers working on budget-sensitive projects that still require robust computing capabilities [16].

4.1.2 High-Accuracy LiDAR: RoboSense RS-LiDAR-M1 v2

I Sensor Specification: The RoboSense RS-LiDAR-M1 v2 is a high-performance, solid-state LiDAR sensor known for its exceptional accuracy and range capabilities [52]. This LiDAR system utilizes MEMS (Micro-Electro-Mechanical Systems) technology, providing precise and efficient scanning capabilities. With a detection range of up to 200 meters and an angular resolution of 0.1 degrees, this sensor is designed to capture detailed and precise environmental data. The RS-LiDAR-M1 v2 operates at a scanning frequency of up to 20 Hz, providing high-density point clouds that are crucial for advanced perception tasks.

Common uses for the RS-LiDAR-M1 v2 include autonomous driving, smart transportation systems, and robotics. Its high accuracy and reliability make it ideal for applications requiring precise object detection and environmental mapping, where safety and performance are critical. Figure 4.2 shows the RoboSense LiDAR used in this research [51].

Key Specifications:

i Measurement Capabilities:

- **Ranging Distance:** The RS-LiDAR-M1 can measure distances ranging from 0.5 meters up to 200 meters (with 150 meters achievable using a 10% NIST diffuse reflector).
- **Ranging Precision:** The system offers a high precision of ± 5 cm within a range of 10m to 100m, which is critical for applications demanding accuracy.

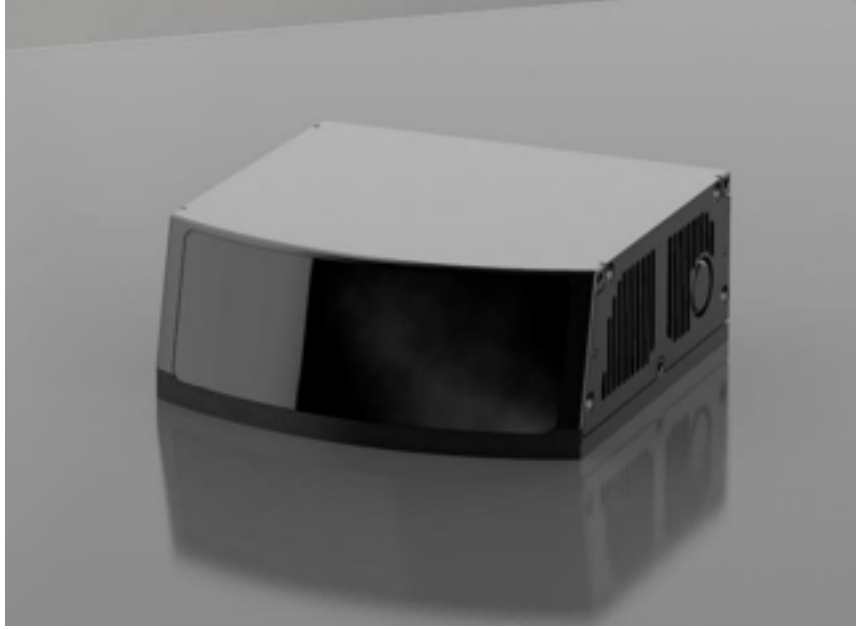


Figure 4.2: RoboSense RS-LiDAR-M1 LiDAR [52].

ii **Field of View (FOV):**

- **Horizontal FOV:** The LiDAR has a 120° horizontal field of view, covering a significant portion of the surroundings.
- **Vertical FOV:** It also features a 25° vertical FOV, with an angular resolution averaging 0.2° both horizontally and vertically, which allows for detailed environmental mapping.

iii **Output Specifications:**

- **Data Rate:** The RS-LiDAR-M1 supports a data rate of up to 1,500,000 points per second in dual return mode, providing dense point cloud data for detailed analysis.
- **Output Channels:** The device outputs data via 1000Base-T1 Gigabit Ethernet using UDP, ensuring fast and reliable communication with processing units.

iv **Laser Specifications:**

- **Safety:** The laser is classified as Class 1 eye-safe, which means it is safe for use in environments where humans are present. The RoboSense RS-LiDAR-M1 operates at a wavelength of 905 nm, commonly used in commercial LiDAR

systems due to compliance with stringent eye-safety standards. Eye safety is paramount, especially in environments where humans may be exposed to laser emissions from LiDAR systems. The 905 nm wavelength allows the LiDAR system to emit sufficient power for effective ranging while staying within the safety limits defined for human eye exposure. This makes RS-LiDAR-M1 an excellent choice for applications requiring both high performance and stringent safety measures [32].

- **Wavelength:** The operating wavelength is 905 nm, which is standard for many LiDAR systems [51].

The choice of the 905 nm wavelength in the RoboSense RS-LiDAR-M1 is based on its advantageous properties within the NIR range. This wavelength is not only effective in terms of performance, but is also compatible with mature silicon-based photodetector technologies, offering high detection efficiency. Although LiDAR systems using 1,550 nm wavelengths can achieve higher power output, they require expensive indium gallium arsenide (InGaAs) detectors and suffer from greater atmospheric water absorption. In contrast, 905-nm systems can utilize cost-effective and efficient silicon detectors, making this wavelength a popular choice in the industry. The combination of high efficiency, cost effectiveness, and atmospheric resilience solidifies the 905-nm wavelength as a reliable and practical option for high-performance LiDAR systems such as the RS-LiDAR-M1 [32].

v Mechanical and Environmental Specifications:

- **Power Consumption:** The device operates at a power consumption of 15W, which is relatively low considering its capabilities.
- **Operating Voltage:** It operates within a voltage range of 9 to 32V DC, making it versatile for integration into various systems.
- **Durability:** The sensor is rated IP67 and IP6K9K, indicating it is highly resistant to dust, water, and other environmental challenges, making it suitable for outdoor use.
- **Operating Temperature:** It can operate in extreme temperatures ranging from -40°C to 85°C, ensuring reliability in diverse climates.

These hardware specifications highlight the capabilities and suitability of RS-LiDAR-M1 for applications requiring high precision, extensive coverage, and reliable performance in challenging environments [51].

II Computing Unit Used: The computing unit for the RoboSense RS-LiDAR-M1 v2 is a ASUS ZenBook Flip S UX370 [1]. The specifications of the laptop include:

- Processor: Intel® Core™ i7-8550U (1.8GHz quad-core with Turbo Boost up to 4.0GHz)
- Memory: 16GB LPDDR3 2133MHz
- Storage: 512GB PCIe SSD
- Graphics: Intel® UHD Graphics 620
- Display: 13.3" 4K UHD (3840 x 2160) touchscreen
- Connectivity: Wi-Fi 802.11ac, Bluetooth® 4.2
- Operating System: Ubuntu 20.04 LTS

The laptop's powerful processor and ample memory support efficient data handling and processing, suitable for managing the high data throughput from the LiDAR sensor. The decision to use a laptop for this setup was driven by the need for rapid integration and ease of use.

III Components Involved and Connection: The setup for the RS-LiDAR-M1 v2 involves several key components and connections, as detailed in the user guide:

- LiDAR Sensor: RoboSense RS-LiDAR-M1 v2
- Computing Unit: ASUS ZenBook Flip S UX370
- Power Adapter: 12V DC power adapter to supply power to the LiDAR sensor
- Ethernet Cable: Connects the LiDAR sensor to the laptop for data transmission
- 3D-Printed Adaptor: Custom adaptor for mounting the LiDAR on the tripod
- Additional Interfaces: Include network interfaces for configuring and managing the sensor settings

These components work together to ensure the stable and efficient operation of the LiDAR system. The power adapter provides a consistent power supply, while the Ethernet connection enables high-speed data transfer between the sensor and the computing unit. The connection topology of the sensor to the computing unit is shown in figure 4.3 below [51].

The figure 4.3 above illustrates the connectivity and interface setup essential to integrate the LiDAR system with the associated computing unit. Here is a detailed explanation of the components and their connections:

- LiDAR Sensor: RoboSense LiDAR was used to collect the environmental data.

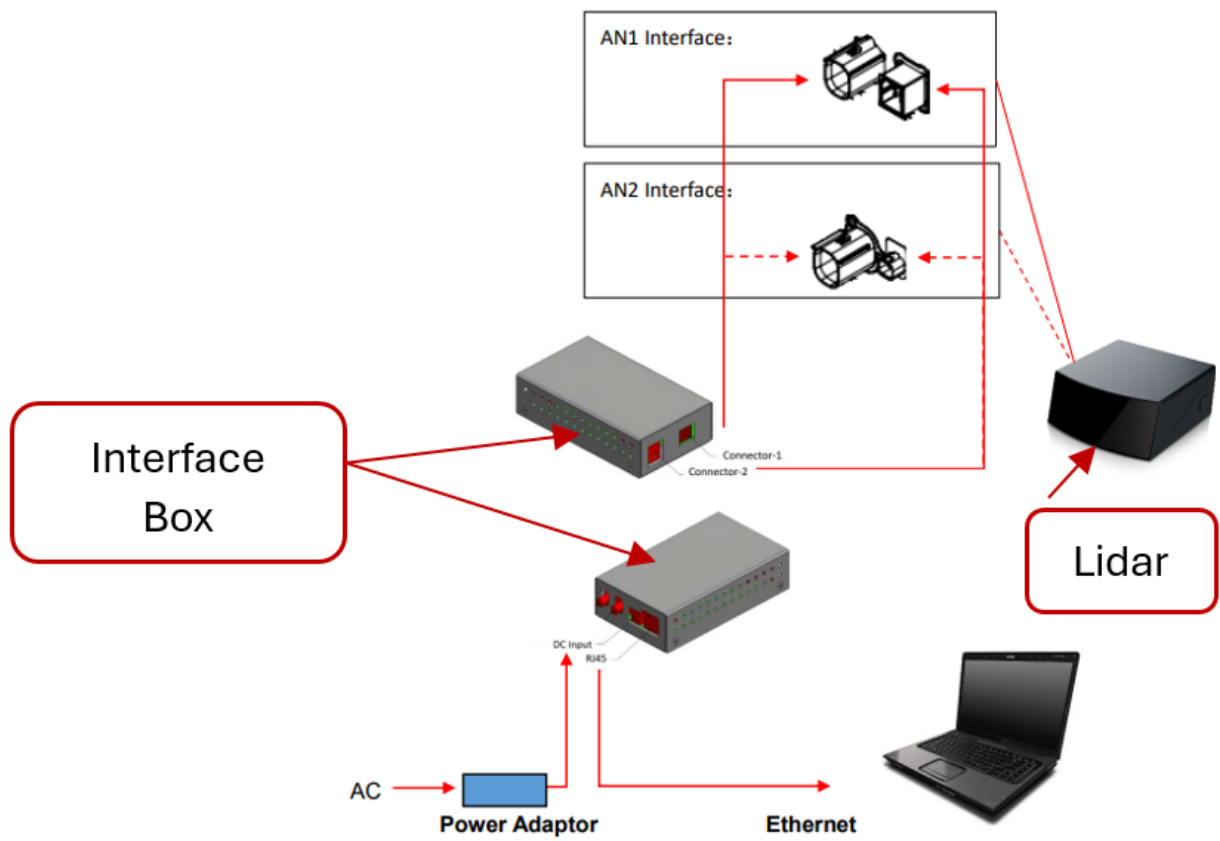


Figure 4.3: Topology of LiDAR and PC [51].

- Interface Box and AN1 and AN2 Interfaces: These represent the specific channels or ports on the LiDAR sensor allocated for data transmission.
- Ethernet Connection: The LiDAR sensor is connected to the laptop through an Ethernet cable. This setup provides a high-speed data transmission link, ensuring that large volumes of point cloud data are transferred efficiently and without latency.
- Power Adaptor: The LiDAR sensor requires a dedicated power supply to operate.
- Laptop (Computing Unit): This serves as the computing unit. It is used to receive and process data from the LiDAR sensor, and can be applied to control sensor operations.
- Connector 1 and Connector 2: These are used to detail additional connectivity that might be required for expanded functionality or secondary data processing channels.

This setup diagram is a blueprint for setting up the LiDAR system in real-world applications, ensuring all components are correctly connected and configured for optimal performance. The practical setup demonstrated in the figure below mirrors this diagram, but includes the physical objects used in real-life [51].



Figure 4.4: Essential components Involved connecting the RoboSense LiDAR [51].

In Figure 4.4, components essential for LiDAR data collection are shown. Central-top to the setup is the LiDAR sensor, visible with multiple cables protruding, which captures the environmental data crucial for analysis. Adjacent to the sensor, an Ethernet cable, distinguishable by its yellow color, is positioned to facilitate high-speed data transfer from the sensor to the computing unit once connected. The power adapter, necessary for converting AC to DC to power the LiDAR sensor, is also evident, with its cable ready to be hooked up to provide the needed electricity. Additional protective black covers are used over the two connectors to protect the sensitive connections and cabling, ensuring they remain secure from physical and electrical interference during the setup. This configuration underscores a meticulous preparation phase where each component is strategically placed to optimize functionality once fully assembled. Detailed connection requirements can be found in the RoboSense RS-LiDAR-M1 v2 User Guide [51].

4.1.3 Data Acquisition Speed and Real-Time Efficiency

When real-time terrain classification and stairs detection systems are implemented, the speed of data acquisition plays a critical role in determining the effectiveness and responsiveness of the system. The data acquisition rate (measured in Hz) refers to the frequency with which a LiDAR sensor can capture its environment and transmit the data to processing. In real-time applications, particularly those where the system must respond to dynamic environments, faster data acquisition rates are generally preferred to ensure that all environmental changes are captured accurately and in time.

Cyglidar D1: The Cyglidar D1 operates at a data acquisition speed of 15 Hz, meaning it can capture 15 frames per second. While this speed is sufficient for slower-paced applications or static scenarios, it may fall short in dynamic environments where a user is moving quickly or when fast-changing obstacles are present. For instance, when a person is walking rapidly, ascending stairs, or navigating uneven terrain, a faster data acquisition speed would allow the system to more accurately detect changes in the environment and adjust accordingly. In such real-time applications, missing data points due to a lower acquisition rate could result in delayed or inaccurate terrain classification or obstacle detection, potentially compromising user safety [43].

RoboSense RS-LiDAR-M1 v2: In contrast, the RoboSense RS-LiDAR-M1 v2 offers significantly higher data acquisition rates, capturing up to 30 Hz or more depending on the configuration. This higher rate allows the sensor to capture more detailed point clouds in less time, which is essential for applications where rapid decision-making is required. The

faster acquisition speed ensures that critical features, such as stairs or terrain transitions, are detected in real-time without missing key data. This makes the RoboSense LiDAR more suitable for high-speed or dynamic environments where the user may need immediate feedback to navigate safely.

Trade-offs and Importance for Real-Time Applications: The trade-off between the two LiDAR sensors primarily revolves around the balance between speed and application requirements. While the Cyglidar D1 is more compact and cost-effective, its slower data acquisition speed limits its performance in real-time, high-speed scenarios. On the other hand, the RoboSense RS-LiDAR-M1 v2, with its higher acquisition speed, provides the ability to capture more detailed data and respond faster, making it ideal for embedded systems where rapid feedback and decision-making are crucial. For real-time terrain classification and stairs detection, a data acquisition rate of 30 Hz or higher is generally recommended to maintain system responsiveness and ensure the safety and reliability of the application [10, 30].

However, the effectiveness of these data acquisition rates is closely tied to the system’s computational capabilities, which process these incoming data in real-time, as discussed in the following section.

Real-Time Efficiency

The real-time performance of the LiDAR-based system was a key consideration in both the stairs detection and terrain classification experiments. Given the high-density point clouds generated by sensors such as the RoboSense RS-LiDAR-M1 v2, ensuring timely data processing is essential for maintaining responsiveness in dynamic environments.

In this study, all LiDAR data processing was handled exclusively by a CPU, as the PCL does not yet offer native support for GPU acceleration. Although this CPU-only approach was sufficient for the tasks performed, integrating GPU support could potentially reduce computation times for more complex tasks, such as terrain classification and stairs detection. However, modifying the current software architecture to support GPU processing would require significant changes, as PCL is optimized for CPU-based operations.

The computational power utilization percentages and system load during data processing were not explicitly monitored or recorded throughout the experiments. Future work could include a detailed analysis of resource utilization to identify bottlenecks and areas for potential optimization, particularly for scaling up to larger or more complex scenarios.

In applications requiring more rapid data processing, GPU integration, potentially through alternative libraries or hybrid CPU-GPU solutions, could enhance system effi-

ciency. Careful consideration would be needed to maintain compatibility with PCL or develop a pipeline that leverages GPU-accelerated frameworks while continuing to use PCL for other tasks.

4.2 Experimental Setup

The subsequent section describes how the data was collected, including the scenes captured and their corresponding 3D point clouds from both sensors. This includes both low-accuracy and high-accuracy LiDARs, used in environments that span from controlled indoor settings for stairs detection to diverse outdoor conditions for terrain classification. The experimental setup employed for data collection, crucial for the development and validation of the stairs detection and terrain classification models discussed, is described. The setup is designed to ensure the acquisition of high-quality data using two LiDAR sensors: the low-accuracy Cygbot Cyglidar D1 and the high-accuracy RoboSense RS-LiDAR-M1 v2. The configurations, supporting components, and specific methodologies utilized for data collection are tailored to address the unique challenges associated with each application, i.e., stairs detection and terrain classification.

4.2.1 Low-Accuracy LiDAR Setup: Cyglidar D1

The Cyglidar D1 was utilized in two distinct configurations for data acquisition: i) mounted on a tripod for stationary data collection, and ii) in a wearable format for dynamic, user-centric data gathering. This dual configuration approach facilitated a comprehensive environmental scan, capturing both static features of stairs and the varying terrains encountered in real world settings. The tripod setup ensured stability and precise alignment for static data collection, enabling the collection of detailed point-cloud data crucial for stairs detection algorithms. In contrast, the wearable setup offered mobility, demonstrating the sensor’s versatility in capturing data from the user’s perspective—an approach particularly valuable for terrain classification tasks, such as distinguishing between plain and grassy surfaces.

In the experimental setup, the LiDAR sensor was mounted in both tripod and wearable formats, with the Cyglidar D1 specifically angled slightly downward at **approximately 30 degrees** to ensure the initial steps of the stairs were captured within 3-5 walking steps. This configuration was critical for acquiring detailed point cloud data of both stair structures and various terrain types, laying the groundwork for the algorithms used in stair

detection and terrain classification. In addition to that, a mobile workstation mounted on a walker was employed to manage and process data in real-time, ensuring that the LiDAR sensor remained stable and properly aligned to deliver consistent and reliable data capture.

The data acquisition process was designed to reliably collect LiDAR data, which is essential to develop and validate stair detection and terrain classification algorithms. This process unfolded in two phases: first, the tripod setup was employed for controlled data collection, followed by the wearable setup for capturing dynamic real-world data.

- **Tripod Setup:** The tripod set up served as the initial phase of data collection, providing a stable and repeatable apparatus essential for establishing baseline measurements and calibrating the sensors. The primary purpose of the tripod setup was to collect high-quality, consistent data from targeted planes, such as floors, walls, and staircases. This setup allowed for precise control over the sensor’s position and orientation, ensuring uniform data capture across various environments.

The LiDAR sensor was securely mounted on a tripod, providing a stable platform that minimized vibrations and other disturbances. This setup facilitated the collection of detailed point cloud data, crucial for algorithm development and testing. The tripod setup was enhanced by integrating a mobile workstation, comprising a monitor, keyboard, and mouse. Figure 4.5 demonstrates this sensor setup and the necessary accessories involved. This configuration enabled real-time monitoring and analysis of the data being collected, allowing researchers to make immediate adjustments to the sensor’s settings or position as needed.

During data collection, the tripod was positioned in various locations within the test environment to capture comprehensive data sets (Fig. 4.6a). The LiDAR sensor scanned the surroundings, generating point clouds that were stored for later processing and analysis. The point clouds generated from these setups are shown in Figures 4.8a and 4.8b.

- **Wearable Setup:** The wearable setup was developed to simulate real-world conditions, allowing the LiDAR sensor to be worn on the body to capture dynamic data as the user moved through different environments. This setup aimed to replicate the conditions under which assistive technologies would operate, capturing data from the user’s own perspective (i.e., egocentric view). This setup provided insights into how the sensor would perform in practical applications, where mobility and adaptability are crucial.

A GoPro chest strap and mount were modified to accommodate the LiDAR sensor at the user’s chest level. This positioning allowed the sensor to capture a natural field

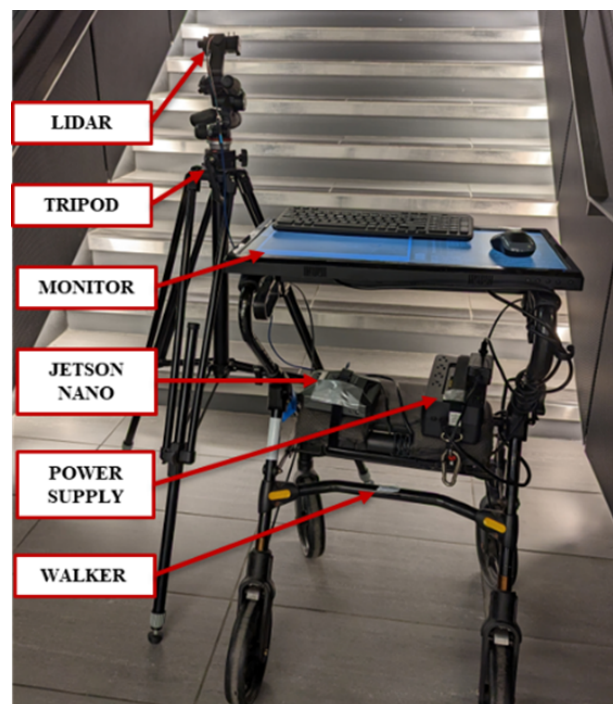


Figure 4.5: Data acquisition setup with tripod and walker

of view, as the user would experience. To ensure secure and adjustable mounting, a custom adaptor was designed and 3D printed. This adaptor integrated an adjustable phone mount with the GoPro buckle base, allowing for fine-tuning of the sensor’s orientation to optimize data capture. Figure 4.6b demonstrates the 3D printed adapter and the LiDAR worn by the GoPro chest strap.

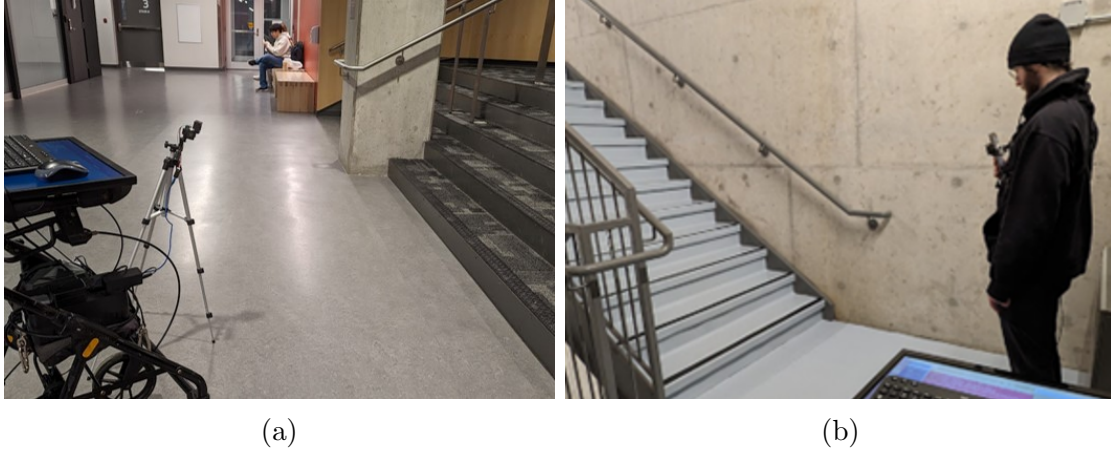


Figure 4.6: a) Cyglidar mounted on tripod b) Cyglidar in wearable-format, both connected with the mobile workstation on walker

The wearable set-up enabled dynamic data collection as the user moved through various terrains and encountered different obstacles. This setup was particularly useful for collecting data in environments where the tripod setup was impractical, such as outdoor settings with uneven surfaces. The point clouds generated from these setups are shown in Figures 4.8a and 4.8b.

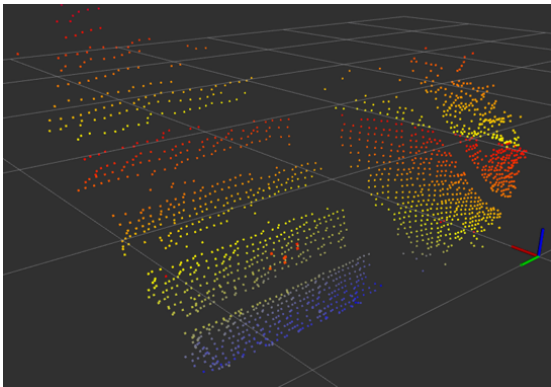
An additional adapter [Fig. 4.7] was designed to mount both the LiDAR sensor and a smartphone, which recorded video footage of the environment. This setup allowed for visual verification of the LiDAR data, ensuring that the sensor accurately captured the scenes it was intended to analyze.

4.2.2 High-Accuracy LiDAR Setup: RoboSense RS-LiDAR-M1 v2

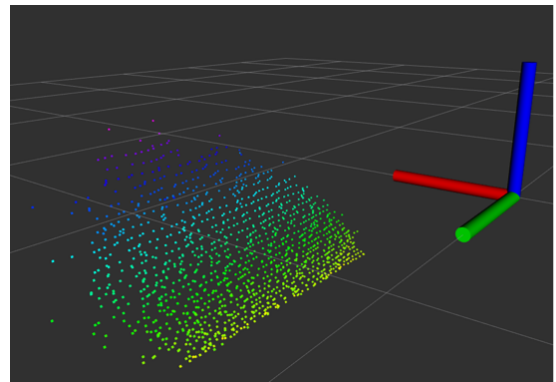
The RoboSense RS-LiDAR-M1 v2, a high-accuracy LiDAR sensor, was exclusively used in a tripod-mounted configuration due to its larger size and the need for precision in



Figure 4.7: Cygbot Cyglidar D1 collection mount design with accompanying phone for validation



(a) Point cloud of the stairs captured by Cyglidar D1



(b) Point cloud of grassy terrain captured by Cyglidar D1

Figure 4.8: (a) Point cloud of the stairs captured by Cyglidar D1 in tripod setup, (b) Point cloud of grassy terrain captured by Cyglidar D1 in wearable setup.

data collection. Unlike the Cyglidar D1, which was also utilized in a wearable format, the RoboSense LiDAR was deployed solely in a stationary setup. The associated sensor components (Fig. 4.4) were carefully connected and placed on a secure surface near the sensor, since the mobile workstation (Fig. 4.5) was not available at the data collection site.

The stationary tripod configuration allowed for alignment and consistency between data collection sessions. The RoboSense RS-LiDAR-M1 v2 was positioned in various environments, focusing on staircases and terrain surfaces, ensuring that the data captured would be representative of real-world conditions. The data collected with this sensor was intended to serve as a benchmark for comparing the performance of the algorithms developed using lower-cost, lower-accuracy LiDAR; Cyglidar D1.

The data processing for the RoboSense RS-LiDAR-M1 v2 was handled exclusively using a CPU, as the point cloud processing library (PCL) utilized for this study does not support GPU acceleration. While CPU processing was sufficient for the real-time tasks in this setup, there is potential for further optimization if GPU integration becomes feasible in the future.

In the following sections, the specific scenes captured, along with the corresponding 3D point clouds from both sensors, will be presented, providing a comprehensive overview of the data acquisition process that underpins this investigation.

Tripod Setup:

The RoboSense RS-LiDAR-M1 v2 was securely mounted on a tripod to ensure stability and precise alignment during data collection. As depicted in Figure 4.9, the setup includes several key components that contribute to the accuracy and reliability of the data acquisition process.

A 3D printed adapter was used to attach the LiDAR sensor to the tripod’s ball-head, allowing for adjustable positioning and ensuring that the sensor could be perfectly aligned with the environment being scanned. This setup was further stabilized using a spirit level, which ensured that the LiDAR was perfectly horizontal, thus avoiding any skewing of the point-cloud data.

This stationary setup allowed the sensor to be positioned in various environments with consistent orientation, focusing particularly on staircases and terrain surfaces. The high accuracy of RoboSense RS-LiDAR-M1 v2, combined with the precision of the tripod setup, was critical to capture high-quality 3D point clouds that serve as a benchmark for comparison with data obtained from the lower-cost Cyglidar D1.



Figure 4.9: Tripod setup for the RoboSense RS-LiDAR-M1 v2 sensor, featuring a 3D printed adapter for secure mounting, a spirit level for precise alignment, and a stable tripod for stationary data acquisition.

RoboSense LiDAR Tripod Setup: Data Collection for Stair and Terrain Detection

Figures 4.10 illustrate the experimental setup in which the RoboSense RS-LiDAR-M1 v2 is mounted on a tripod for data collection in different environments. The setup is designed to capture high-resolution 3D point clouds of both staircases and grassy terrains, which are essential for developing and testing the stairs detection and terrain classification algorithms discussed in this thesis.

In Figure 4.10a, the LiDAR sensor is positioned facing a staircase. This configuration is crucial for capturing the geometric details of the stairs, which are used to train the stair detection model. The sensor is carefully aligned and stabilized on the tripod to ensure that the point-cloud data accurately represents the stair structure. The spirit level attached to the setup ensures that the sensor remains perfectly level, reducing the potential for skewed data. The corresponding point cloud, shown in Figure 4.10c, provides a detailed 3D representation of the stairs, capturing their dimensions and surface characteristics essential for the algorithm’s accuracy.

Figure 4.10b shows the LiDAR sensor directed toward a grassy terrain. This setup is used to collect data that will be used in terrain classification, specifically to differentiate between plain surfaces and grassy areas. By capturing the terrain at various distances and angles, the LiDAR generates a comprehensive dataset that reflects the natural variability of outdoor environments, which is essential for the robustness of the classification algorithms. The resulting point cloud, depicted in Figure 4.10d, offers a detailed view of the grassy terrain, highlighting the texture and depth variations that are critical for effective terrain classification.

4.3 Conclusion

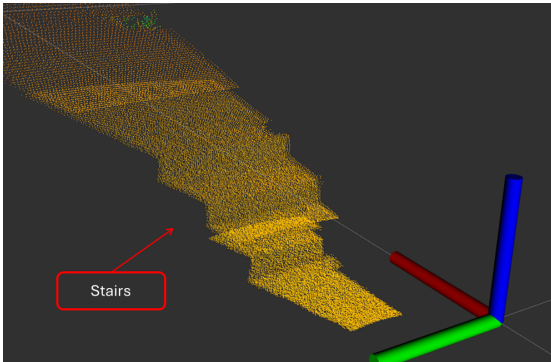
This chapter provided a detailed overview of the experimental setup and configurations employed for data collection using both the low-accuracy Cygbot Cyglidar D1 and the high-accuracy RoboSense RS-LiDAR-M1 v2. The dual format approach with Cyglidar D1, which utilizes both tripod-mounted and wearable setups, allowed for the comprehensive acquisition of point cloud data necessary for both stairs detection and terrain classification tasks. The tripod setup provided stability and precision in static environments, ensuring high-quality data collection for stair detection algorithms. Meanwhile, the wearable setup demonstrated the sensor’s versatility and effectiveness in dynamic, real-world scenarios, particularly in differentiating between plain and grassy surfaces.



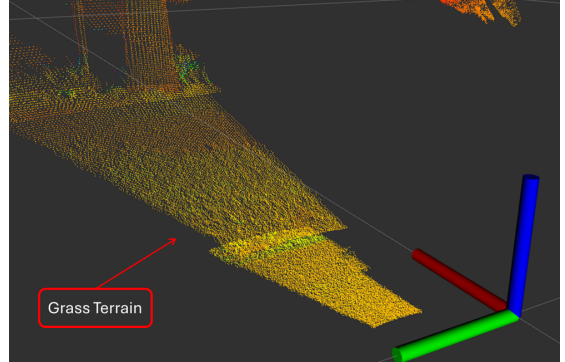
(a) RoboSense LiDAR facing stairs



(b) RoboSense LiDAR facing grassy terrain



(c) Point cloud of the stairs captured by RoboSense LiDAR



(d) Point cloud of grassy terrain captured by RoboSense LiDAR

Figure 4.10: (a) RoboSense LiDAR setup facing a staircase, (b) Corresponding point cloud of the stairs, (c) RoboSense LiDAR setup facing grassy terrain, (d) Corresponding point cloud of the grassy terrain.

The RoboSense RS-LiDAR-M1 v2, deployed exclusively in a tripod-mounted configuration, served as a benchmark for performance evaluation. Its high-resolution data captured under controlled conditions will be used to assess the effectiveness of the algorithms developed using the lower-cost Cyglidar D1 sensor.

In summary, the experimental setups described in this chapter are fundamental to the development and validation of the algorithms discussed in the following chapters. The collected data not only provides the foundation for accurate and reliable algorithm development, but also highlights the trade-offs between sensor accuracy, cost, and practical deployment in assistive technologies. This chapter thus sets the stage for the detailed analysis and discussion of the algorithmic results in the following sections.

Chapter 5

Results

This chapter evaluates the performance of the comprehensive real-time pipeline developed for stairs and terrain classification studies conducted using: 1) the low-accuracy Cyglidar D1 and 2) the high-accuracy RoboSense RS-LiDAR-M1 v2. The analysis is structured to first evaluate the performance of each sensor in its original state, followed by a detailed examination of how sensor accuracy and added noise impact the terrain classification results.

The results examining the efficacy of the developed pipeline in processing 3D point cloud data are discussed, starting with the raw point cloud data obtained from each sensor, progressing through the preprocessing steps (incl. downsampling, noise addition, and normal extraction steps), and concluding with the classification performance using Support Vector Machines (SVMs) with both linear and nonlinear kernels.

Each section provides visual representations of the intermediate steps, such as point clouds at various stages of processing, and the resulting classification boundaries. These visualizations are critical for understanding the influence of sensor accuracy and noise on classification results. The final results are analyzed to determine the efficacy of each approach and to compare the robustness of the algorithm at different levels of sensor noise and accuracy.

The chapter concludes by summarizing the key findings and providing insights into the implications of sensor selection and noise handling for terrain classification tasks. This sets the stage for discussion in the subsequent chapter, where the broader implications of these results are explored in the context of real-world applications.

5.1 Stairs Detection using Cyglidar

This section evaluates the performance of a SVM binary classifier with an RBF kernel for detecting ascending stairs using data collected from both the wearable and tripod configurations of the Cyglidar D1 sensor. As described in the [Chapter 3](#), the SVM model was trained to classify the LiDAR data in order to identify the presence of ascending stairs. The dual setup allowed for comprehensive data collection, capturing the variability in sensor perspective and distance from the stairs.

Given the limited size of the dataset, a k-fold cross-validation approach was employed to evaluate the model using multiple data subsets, thereby providing a robust estimate of model performance. This method enabled the use of the entire dataset for both training and testing, maximizing the utility of the available data.

To optimize the classifier’s performance, an extensive hyperparameter tuning process was performed. More details about training about finding the most optimized hyperparameters can be found in [chapter 3](#) in the stairs detection algorithm development section. The best-performing configuration, which provided the highest classification f1-score across the 10 k-folds, involved tuning the RBF kernel’s key parameters, specifically gamma and the regularization term (C). Table 5.1 summarizes the final parameters selected. The choice of hyperparameters played a significant role in fine-tuning the model’s ability to detect stair scenes, particularly in environments with varying sensor perspectives.

Table 5.1: Final RBF Function Parameters

Gamma	C Normalising term
0.1	100

To provide an assessment of the model’s performance, a confusion matrix was generated by summing the results from all 10 k-folds, as shown in Table 5.2. This matrix gives insight into the true positives, false positives, true negatives, and false negatives, allowing for a comprehensive evaluation of the classifier’s ability to detect stairs.

Table 5.3 summarizes the average precision, recall, and F1-score across all 10 k-folds. The high recall value indicates the classifier’s ability to minimize false negatives, which is critical for safely detecting stairs in real-time applications where missing a stairs could lead to serious accidents.

Table 5.2: SVM 10 K-Fold Total Performance

	p	n	total
p'	41615	540	P'
n'	1127	11620	N'
total	P	N	

Table 5.3: SVM classifier performance

SVM Performance	Value
f1-score	0.9696
Recall	0.9736
Precision	0.9872

5.1.1 Performance Metrics Across 10 K-Folds

In addition to the confusion matrix and average metrics discussed earlier, Table 5.4 provides a comprehensive breakdown of the performance metrics across the 10 k-fold cross-validation iterations. These detailed metrics enable an in-depth analysis of the classifier's stability and robustness.

Table 5.4: Performance Metrics Across 10 K-Folds for Stairs Detection using Cyglidar

Metric	Mean	Standard Deviation	Variance	Minimum	Maximum	Second Moment
Accuracy	0.968271	0.002764	0.000008	0.961020	0.971949	0.937556
Precision	0.952323	0.004556	0.000021	0.946013	0.960688	0.906941
Recall	0.908916	0.010235	0.000105	0.884317	0.925593	0.826234
F1-Score	0.930079	0.005807	0.000034	0.914125	0.936311	0.865081

The metrics in Table 5.4 reveal the consistency of the SVM classifier across the cross-validation folds. Below is a detailed discussion of the metrics:

- **Accuracy:** The model achieved an average accuracy of **96.83%** across the folds, with a minimal standard deviation of **0.28%**. This demonstrates the classifier's ability to generalize effectively to new data, with very little variance (**0.000008**) across folds, indicating consistent performance.
- **Precision:** The precision averaged at **95.23%**, reflecting the model's reliability in minimizing false positives. The small standard deviation (**0.45%**) shows that precision remains consistent across different folds, with the highest value being **96.07%**.
- **Recall:** With an average recall of **90.89%**, the model successfully captured most of the true positive cases. However, the higher standard deviation (**1.02%**) and variance (**0.000105**) compared to precision suggest that some folds may have involved more challenging cases or noise, leading to slight inconsistencies in identifying all positive instances.
- **F1-Score:** The harmonic mean of precision and recall, the F1-score, averaged at **93.01%**, with values ranging from **91.41%** to **93.63%**. This reflects a balanced performance, though some folds with higher variability in recall impacted the overall score.
- **Second Moment:** The second moment values indicate the distribution of the metrics. For example, the second moment for accuracy (**0.937556**) highlights a tight concentration of values near the mean, while recall (**0.826234**) exhibits more spread, underscoring greater variability in detecting positive cases.

Key Observations:

1. **Stability Across Folds:** The small variance and standard deviation values for most metrics highlight the model's robustness and consistent performance across folds.
2. **Precision vs. Recall Trade-Off:** The model demonstrated higher precision compared to recall, suggesting a conservative approach to positive predictions that prioritizes avoiding false positives. However, improving recall in future iterations could further enhance the classifier's ability to detect all positive instances.
3. **Impact of Data Complexity:** The variability in recall and F1-score across folds suggests that certain data subsets may have posed more challenges, likely due to edge cases, sensor noise, or variations in stair features.

4. **Real-World Applicability:** The high mean values for all metrics, coupled with low variability, confirm the classifier’s reliability for real-time stairs detection tasks using the Cyglidar D1 sensor.

The insights from this analysis, combined with the confusion matrix and average metrics, emphasize the model’s robustness for detecting stairs in diverse scenarios. These results also underscore the importance of further optimizing the classifier to address variability in recall and handle challenging cases more effectively.

Limitations

While the SVM model with an RBF kernel provided robust performance in detecting stairs, there are certain limitations inherent in the binary classification approach used. The model struggles with edge cases and transitions between stair and non-stair scenes. These limitations are particularly noticeable in dynamic environments where sensor noise or complex background structures (e.g., hallways with reflective surfaces) can introduce confusion.

Additionally, the Cyglidar D1, being a low-cost LiDAR sensor, introduces some level of noise and inaccuracy, particularly in fast-moving scenes or in areas with high reflectance. This inherent sensor limitation contributes to both false positives and false negatives, as illustrated in Figure 5.1. Addressing these limitations could involve incorporating temporal data to better capture motion dynamics, or using higher-quality sensors with less inherent noise.

Figures 5.2 and 5.3 further illustrate the classifier’s limitations in misclassifying point clouds in fast-moving scenes or complex reflective environments. Future work could involve improving the preprocessing steps to better manage sensor noise or integrating additional sensors to enhance detection accuracy in these challenging environments.

Error Analysis

The challenges observed in the error analysis of stairs detection align closely with the computational and deployment constraints highlighted in the literature. Matsumura et al. [38] introduced a deep learning-based approach leveraging convolutional neural networks (CNNs) for stairs detection, achieving high accuracy and a low false-positive rate. However, this approach requires significant computational resources and access to large, annotated training datasets, which limit its suitability for resource-constrained platforms such as the

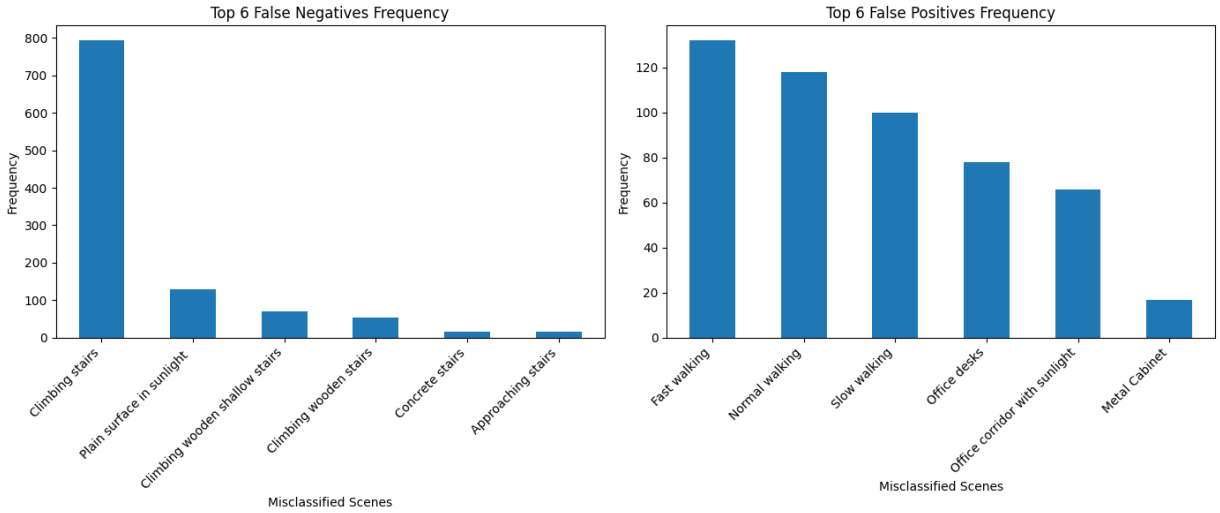
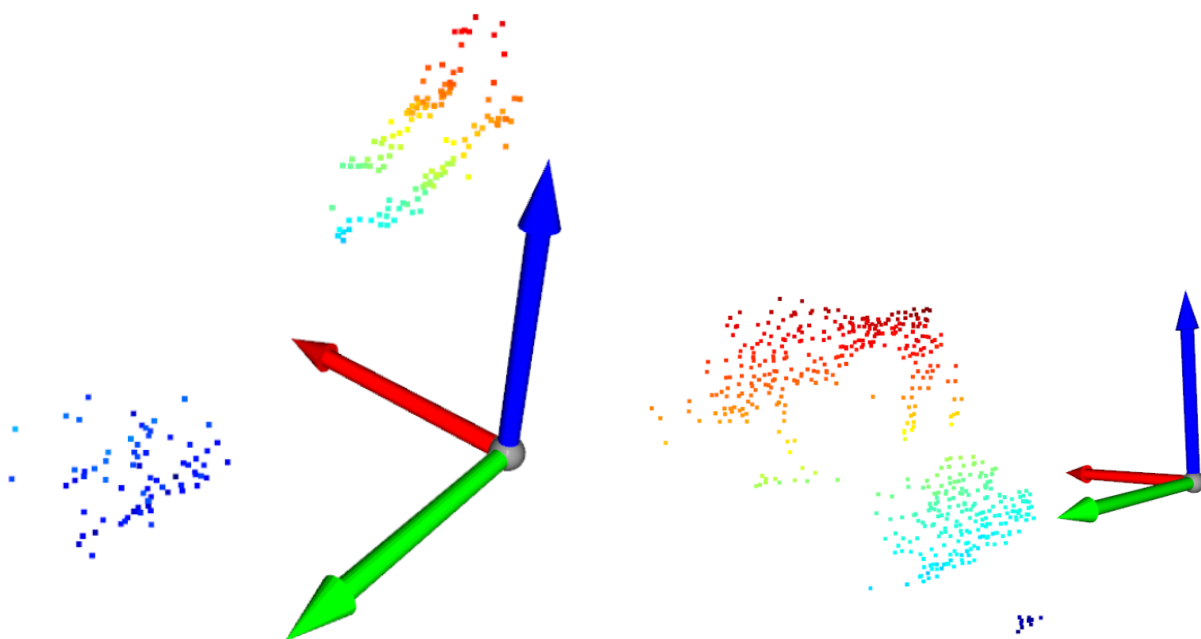


Figure 5.1: Frequency analysis of the top 6 most misclassified scenarios for false negatives and false positives.

Jetson Nano. Similarly, its generalizability across diverse environments is restricted due to its reliance on data that may not adequately capture real-world variability. In comparison, the SVM-based approach developed in this thesis, which employs RBF kernels, provides comparable accuracy (F1-score of 96.96%) while significantly reducing computational overhead. This efficiency is achieved by optimizing the preprocessing pipeline and utilizing feature-based learning, making it more practical for real-time deployment in assistive devices.

While the CNN approach proposed by Matsumura et al. demonstrates effectiveness in ideal conditions, environmental factors such as low illumination and occlusions can significantly impact its performance. Similar limitations were addressed by Zhao et al. [71], who fused LiDAR and RGB-D camera data to detect and model stairs in real-time. Their method enhanced the ability to identify stair geometries, such as tread and riser dimensions, and improved functionality in rehabilitation applications. However, the use of RGB-D cameras introduces privacy concerns and requires significant computational power for multimodal data processing.

The findings in this thesis emphasize that low-cost LiDAR sensors, such as the Cyglidar D1, are effective for basic stairs detection but encounter challenges in distinguishing complex or occluded stair scenes due to inherent noise and reduced resolution. This observation aligns with the computational trade-offs discussed by Bansal et al. [2], where real-time pro-



(a) False Positive: Fast Walking.

(b) False Positive: Reflective Office Desks.

Figure 5.2: Examples of False Positives in Fast Walking and Office Desk Reflective Scenes.

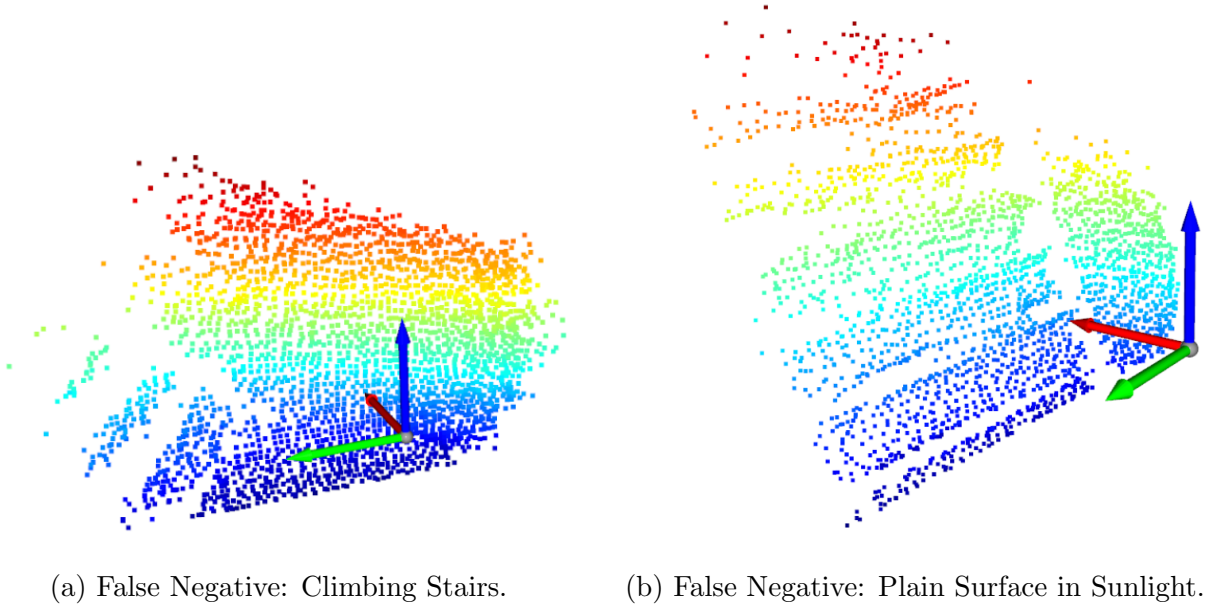


Figure 5.3: Examples of False Negatives: Climbing Stairs and Plain Surface in Sunlight.

cessing of LiDAR data for mobile robotics demanded substantial computational resources. Potential improvements to address

Discussion

The performance of the SVM classifier with an RBF kernel in detecting ascending stairs using the Cyglidar D1 sensor demonstrated impressive robustness, with a f1 score of 96.96% on the 10 k cross-validations. This level of precision is crucial for applications where reliable detection of stairs is paramount to ensuring user safety, particularly in mobility-assisted devices. The strong precision and recall metrics of the classifier further underscore its effectiveness in distinguishing between stairs and non-stairs scenes.

Despite the overall success of the classifier, certain limitations became apparent. False positives, where non-stair scenes were incorrectly classified as stairs, were primarily observed in scenarios involving rapid movement or environments with high reflectance, such as hallways with reflective surfaces. These errors, while not as critical as false negatives, still pose potential inefficiencies by diverting computational resources towards processing non-existent stairs. Reducing false positives through methods such as incorporating tem-

poral data or filtering multiple frame sequences could improve the algorithm’s robustness in dynamic environments. Examples of false positives are shown in Figure 5.2, where fast walking and reflective office desks were common causes of misclassification.

Conversely, false negatives present a more significant concern due to the safety risks they introduce. In instances where actual stairs were missed, the most common causes involved fast movements or challenging approach angles, where the sensor had insufficient time to capture a complete view of the stairs. These false negatives could potentially lead to hazardous situations in real-world applications. Figures 5.3a and 5.3b depict typical examples of failure conditions involving stair climbing or detecting plain surfaces in sunlight. These errors often occurred when the subject was either entering or leaving the sensor’s field of view, underscoring the importance of maintaining consistent visibility for accurate detection.

A frequency analysis was performed on the top six scenarios most frequently misclassified to gain insight into the reasons for false positives and false negatives. As shown in Figure 5.1, the majority of false negatives occurred during stair-climbing events, particularly when the scene was poorly illuminated or the sensor’s field of view was obstructed. False positives were most frequently observed during fast walking or when reflective surfaces, such as metal cabinets or desks, confused the sensor’s readings. These findings highlight the need for improved preprocessing techniques to better filter noise and reduce the impact of reflectance on the classifier’s performance.

While the SVM classifier with an RBF kernel performed well in static environments, the analysis indicates that dynamic environments, especially those involving rapid movement or highly reflective surfaces, still pose significant challenges. Integrating additional sensors, such as infrared cameras or depth sensors, could provide complementary data to mitigate these limitations. Furthermore, improving the classifier’s preprocessing pipeline, for instance by employing advanced filtering techniques to reduce noise, could significantly enhance detection accuracy.

Finally, it is important to note that the current algorithm operates on a frame-by-frame basis, which limits its ability to account for temporal context. A potential improvement could involve incorporating a memory unit or voting system that aggregates data over a sliding time window. This approach could increase confidence in stair detection by considering multiple frames in sequence, reducing the likelihood of false negatives or false positives arising from momentary noise or sensor inconsistencies. This temporal context could enable the classifier to make more informed decisions in real-time applications, where rapid and accurate response is critical.

5.2 Terrain Classification using the Lower Accuracy LiDAR (Cyglidar)

This section explores terrain classification using the Cyglidar D1, a lower accuracy LiDAR sensor. The challenges associated with this sensor, due to its limited resolution and precision, are highlighted, particularly in differentiating between plain and grassy terrains. The analysis includes visualizations of the raw point cloud, downsampled cloud, extracted normals, and scatter plots of the normals for both terrain types. The results obtained using an SVM with a non-linear (RBF) kernel are presented, followed by a discussion on the difficulties of achieving accurate terrain classification with this sensor.

The preprocessing pipeline developed in this thesis considers efficient noise handling and feature preservation, tailored specifically for resource-constrained embedded platforms. This contrasts with conventional high-resource preprocessing techniques used in prior studies.

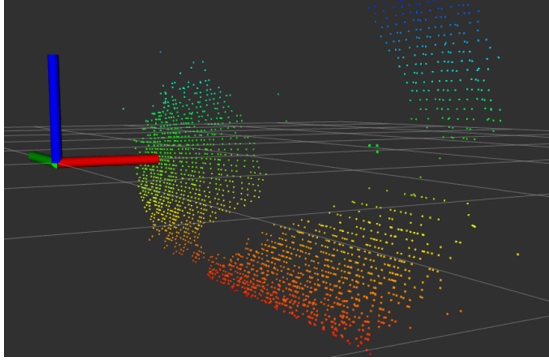
5.2.1 Raw and Processed Point Cloud

Figures 5.4a and 5.4b show the raw point cloud data obtained from the Cyglidar D1 sensor when the sensor was directed toward plain and grassy terrains, respectively.

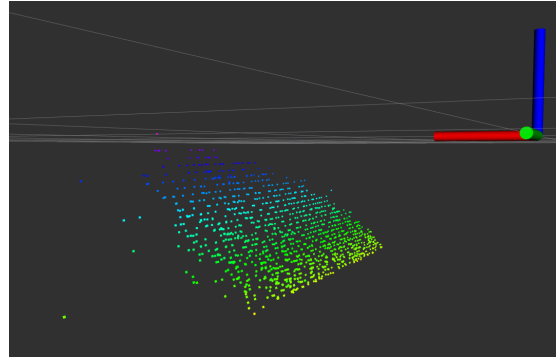
The subsequent figures 5.4c and 5.4d illustrate the downsampled point clouds after applying passthrough filtering. The passthrough filter effectively isolates the region of interest by cropping out unnecessary data points, focusing only on the relevant terrain. Downsampling further reduces the density of the point cloud, decreasing the computational load while retaining the essential features of the terrain that are necessary for accurate classification. This balance between data reduction and feature preservation is critical for efficient and effective terrain classification using lower-accuracy sensors such as the Cyglidar D1.

5.2.2 Normal Extraction and Scatter Plot

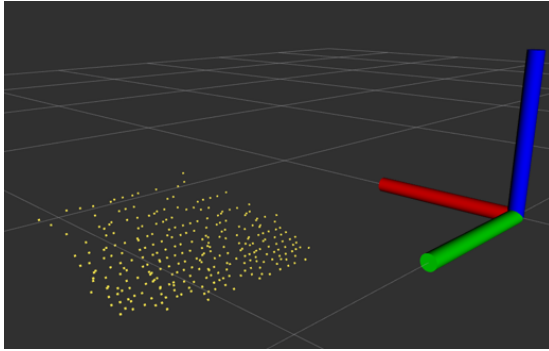
Figure 5.5 illustrates the extracted normal vectors from the downsampled point clouds for plain and grassy terrains. These normals provide critical information about the terrain surface orientation and serve as the primary features for subsequent classification. Note the components of the normals are expressed in radians.



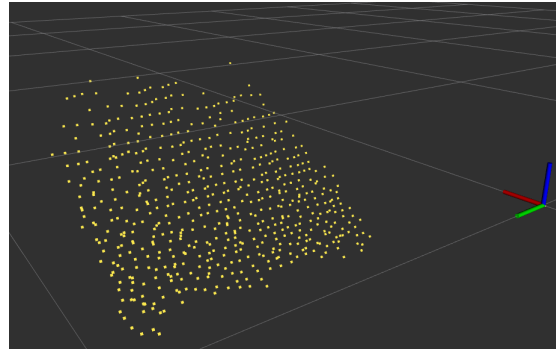
(a) Raw point cloud data from Cyglidar D1 (Plain Terrain)



(b) Raw point cloud data from Cyglidar D1 (Grassy Terrain)

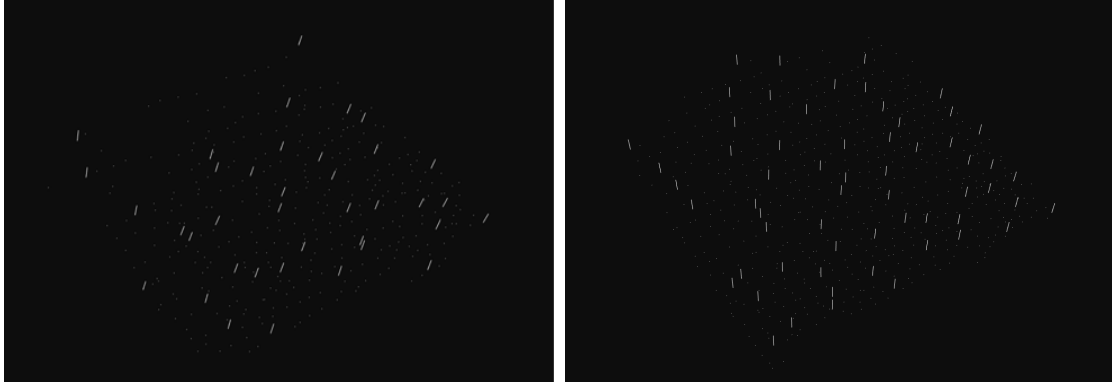


(c) Downsampled point-cloud (Plain Terrain)



(d) Downsampled point-cloud (Grassy Terrain)

Figure 5.4: Cyglidar D1 point cloud data: (a) Raw point cloud for plain terrain, (b) Raw point cloud for grassy terrain, (c) Downsampled point cloud after passthrough filtering for plain terrain, (d) Downsampled point cloud after passthrough filtering for grassy terrain.



(a) Normals extracted from plain terrain (b) Normals extracted from grassy terrain

Figure 5.5: Normals extracted from downsampled Cyglidar D1 point clouds for (a) plain terrain, and (b) grassy terrain.

The figure 5.5 illustrates the normal vectors extracted from the downsampled point clouds for both plain and grassy terrains. In the context of point-cloud data, normal vectors provide insights into the surface orientation of the terrain. The normals for plain terrain (Figure 5.5a) are relatively uniform and consistent, indicating a smooth and flat surface. This is expected, as plain surfaces generally have minimal variations in height and structure, resulting in normals that are closely aligned.

On the other hand, normals extracted from the grassy terrain (Figure 5.5b) show greater variability. The grass introduces a significant amount of surface roughness and irregularity, which is reflected in the scattered orientations of the normals. This variability is a key feature that distinguishes grassy (i.e., rough) from plain (i.e., smooth) terrain and forms the basis for terrain classification.

Following normal extraction, a scatter plot of the X and Y components of these normals is generated, as shown in Figure 5.6. The scatter plot provides a visual representation of how normal vectors differ between plain and grassy terrains. The plain terrain normals (represented by purple points) are tightly clustered, indicating that the surface has minimal deviation from the mean orientation. This tight clustering is indicative of the uniformity of the plain surface, where the normals do not vary much in direction. In contrast, the grassy terrain normals (represented by green points) are more widely dispersed. The spread of these points in the scatter plot reflects the natural variation in the orientation of the surface caused by the unevenness and texture of the grass. The wider spread of the normals of the grassy terrain in the X and Y directions highlights the increased complexity and irregularity of the surface.

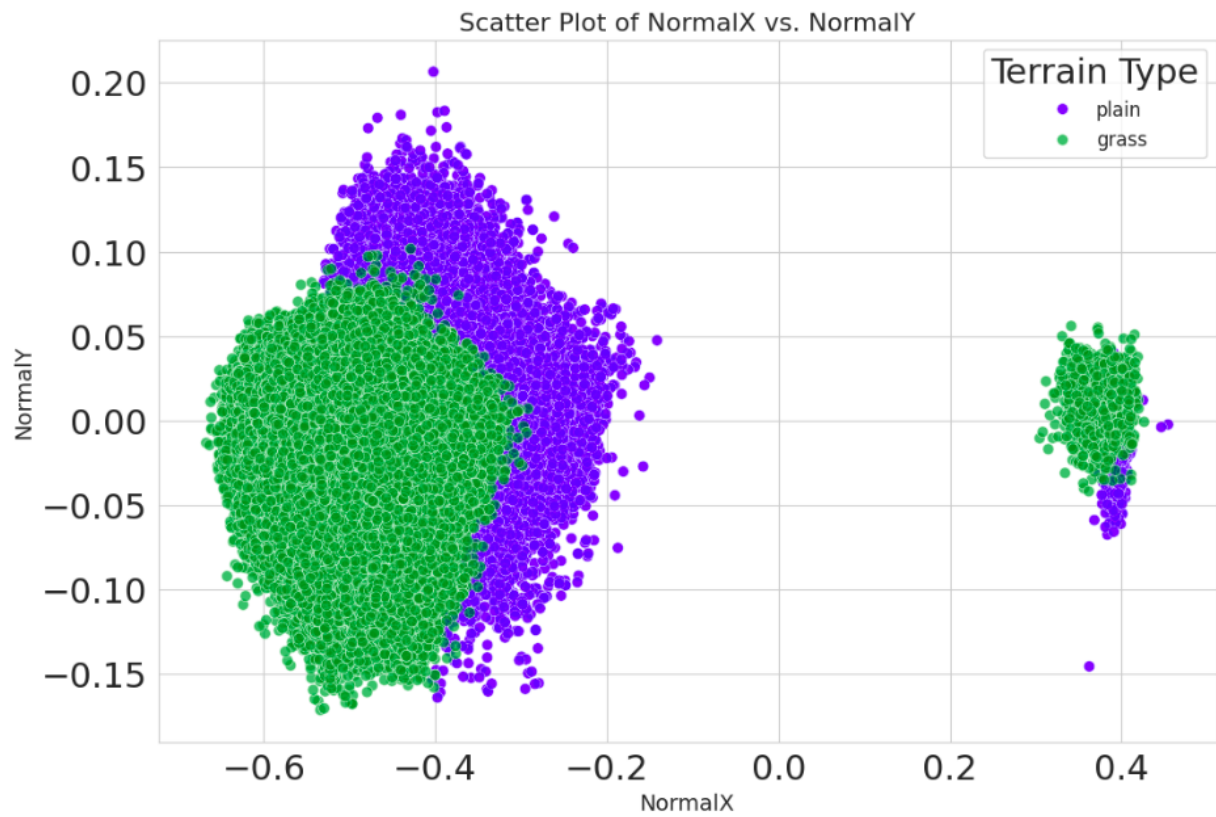


Figure 5.6: Scatter plot of NormalX vs. NormalY for plain and grassy terrains (components in radians).

This distinction between the clustering of normals for plain terrain and the dispersion for grassy terrain is crucial for the terrain classification algorithm. The differences in the normal vector distribution provide a reliable feature for distinguishing between these two types of surfaces, which is essential for applications such as assistive devices or autonomous navigation where terrain recognition is critical.

The distinct separation between clusters of normals in the scatter plot indicates the effectiveness of the novel preprocessing and feature extraction pipeline, enabling robust classification even with lower-accuracy sensors like the Cyglidar D1.

The analysis of these plots will be further discussed in the *Results using SVM with Non-linear (RBF) Kernel* section, where the effectiveness of the terrain classification model based on these features is examined.

5.2.3 Results using SVM with Non-linear (RBF) Kernel

The terrain classification model was evaluated using a Support Vector Machine (SVM) with a non-linear Radial Basis Function (RBF) kernel. The RBF kernel has been used to identify complex, non-linear decision boundaries, making it potentially more accurate in distinguishing between terrains that have overlapping features in the feature space.

In this experiment, the model was trained and validated using a 20-second validation dataset set aside for each terrain type. The real-time performance was tested on the Jetson Nano by running the model prediction ROS node in parallel with a rosbag that simulated data streaming from a sensor.

The results, shown in Table 5.7, indicate that while the RBF kernel can effectively capture the nuances in terrain classification, it requires substantial computational resources. The average total time taken for preprocessing, feature extraction, and prediction was approximately 390.52 ms per sample, which is relatively high for real-time applications. This longer processing time is primarily due to the computational complexity of the RBF kernel, which involves evaluating the Gaussian function for each support vector during prediction.

In terms of performance, the model achieved a precision of 1, a recall of 0.5861, and an F1 score of 0.7355, shown in Table 5.6. These metrics indicate that while the classifier was highly precise in identifying positive terrain features, it struggled with recall, meaning that a significant number of true positives (correctly identified terrain features) were missed. The F1 score, which is the harmonic mean of Precision and Recall, reflects the balance between these two metrics and highlights the trade-offs involved in using a complex kernel like the RBF.

Average Preprocessing Time (ms)	Average Feature Extraction Time (ms)	Average Prediction Time (ms)
3.60	1.94	384.98

Table 5.5: Real-time performance of SVM with RBF kernel on Jetson Nano.

Precision	Recall	F1 Score
1.0000	0.5861	0.7355

Table 5.6: Precision, Recall, and F1 Score for SVM with RBF kernel on Jetson Nano.

While the RBF kernel was able to handle non-linear terrain features effectively, the relatively low recall suggests that the model may struggle in more dynamic or varied environments. This can be attributed to the computational complexity and the limited generalizability of the model in real-time applications where rapid predictions are crucial. The high computational demand of the RBF kernel would introduce processing delays, making it less suitable for scenarios where rapid decision-making is essential, such as in autonomous navigation or assistive technologies.

5.2.4 Discussion

The performance of the SVM classifier using the non-linear RBF kernel on the Jetson Nano highlights important trade-offs between model complexity and real-time applicability. While the RBF kernel is capable of capturing complex non-linear decision boundaries, the computational cost associated with these capabilities proved to be a significant limitation in a real-time setting. The model demonstrated an average total processing time of 390.52 ms, which is substantial for real-time applications that require rapid decision-making.

In terms of classification performance, the precision of the model was excellent at 1.0, indicating that the classifier rarely made false positive errors in identifying the terrain features. However, a recall value of 0.5861 implies that the model failed to detect a significant number of true positive terrain features, indicating that despite its high specificity, it had difficulty correctly identifying all relevant features. This is further reflected in the F1 Score of 0.7355, which represents a balance between Precision and Recall. The results suggest that although the RBF kernel is theoretically better suited for complex decision boundaries, its practical performance in this context was limited by the computational overhead.

Moreover, the high computational complexity of the RBF kernel, requiring evaluation

of the Gaussian function for each support vector during prediction, contributes to its slower real-time performance. This delay could potentially hinder its use in scenarios where quick response times are critical, such as autonomous navigation or assistive devices. These results underscore the need for careful consideration of model complexity when deploying machine learning algorithms on resource-constrained platforms like the Jetson Nano.

The comparison of real-time performance metrics is presented in Table 5.7 and the precision, recall, and F1 score metrics in Table 5.6. The high Precision indicates that the model is highly specific but the relatively low Recall and F1 Score suggest that it may struggle to detect all true terrain features in a real-time environment, likely due to sensor noise and variability in the data.

Metric	RBF Kernel
Average Total Time Taken (ms)	390.52
Precision	1.0000
Recall	0.5861
F1 Score	0.7355

Table 5.7: Performance metrics for SVM with RBF kernel on Jetson Nano.

In conclusion, while the RBF kernel is effective in handling non-linear terrain features, its significant computational cost and lower recall make it less suitable for real-time terrain classification on embedded systems like the Jetson Nano. The limitations of the Cyglidar D1 LiDAR sensor, particularly its lower resolution and increased noise, further complicated the classifier’s performance, highlighting the critical role of sensor precision in achieving reliable terrain detection. Despite these challenges, the proposed framework successfully extracted meaningful features from the Cyglidar D1 data, demonstrating its adaptability to low-cost sensors and its potential for integration into resource-constrained assistive robotics systems.

The results validate the suitability of this approach for mobility-assistive applications, such as exoskeletons or smart walkers, where real-time terrain detection is essential to adapt to environmental conditions, contributing to user safety and autonomy. While the Cyglidar D1’s limitations introduce challenges in dynamic or noisy environments, the methodology’s ability to adapt and extract actionable features from low-cost sensors reinforces its practicality in cost-sensitive applications. Future improvements, such as advanced noise handling techniques and the incorporation of temporal smoothing, could enhance its robustness. These findings emphasize the system’s potential to advance mobility-assistive technologies by providing a scalable, efficient, and affordable solution for real-time terrain classification.

5.3 Terrain Classification using the Higher Accuracy LiDAR (RoboSense)

While kernel selection is crucial, the quality of the input data plays an important role in the effectiveness of the classification model. A sensor with higher accuracy, like the RoboSense RS-LiDAR-M1, would likely produce point clouds with better defined features and clearer distinctions between different terrain types. This would result in a more pronounced separation of clusters in the feature space, thereby improving classification accuracy and potentially reducing the time required for real-time processing. Building on the insights gained from the Cyglidar D1's performance, this section explores terrain classification using the RoboSense RS-LiDAR-M1 v2, a LiDAR sensor known for its higher accuracy and precision. By leveraging the improved data quality provided by this sensor, the goal is to achieve more reliable and accurate terrain classification.

The analysis in this section begins with an examination of the raw point cloud data captured by the RoboSense LiDAR, followed by steps for downsampling and normal extraction. The extracted normals are visualized through scatter plots, allowing for a direct comparison between the plain and grassy terrains. The results obtained using a linear kernel in an SVM are discussed in detail.

5.3.1 Raw and Processed Point Cloud

The experimental setup for terrain classification involved data collection for both grassy and plain terrains. Figure 5.7 shows the tripod setup for the grassy terrain and the view of the plain terrain where the RoboSense RS-LiDAR-M1 v2 was used to capture the point-cloud data.

5.3.2 Normal Extraction and Scatter Plot

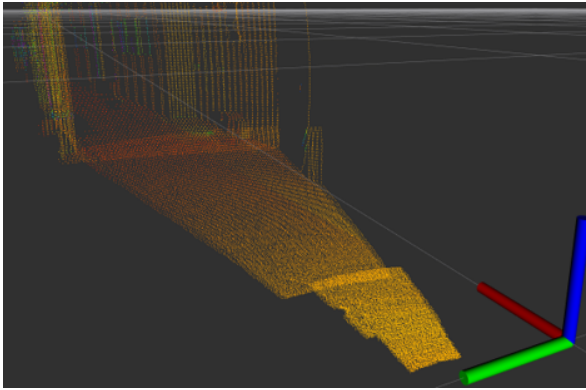
The processed downsampled clouds for both plain and grassy terrains were subjected to normal extraction, following the same preprocessing steps as previously described. These steps included applying a passthrough filter to isolate the region of interest, followed by voxel grid downsampling to reduce the point cloud density while preserving the essential geometric structure. The resulting normals, extracted from the downsampled clouds, provide critical information about the surface orientation of each terrain type.



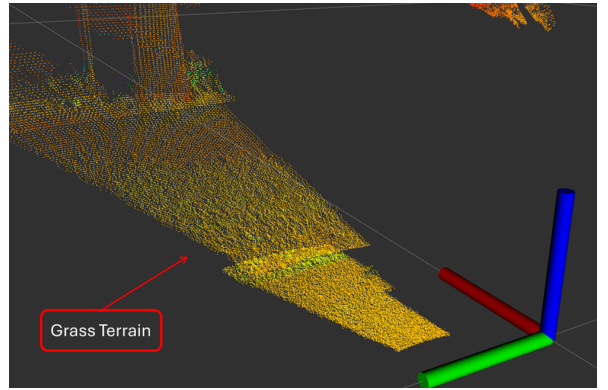
(a)



(b)



(c)



(d)

Figure 5.7: (a) View of the plain terrain, (b) Experimental setup for grassy terrain, (c) Raw point cloud data for plain terrain, and (d) Raw point cloud data for grassy terrain.

Figure 5.8 illustrates the normals extracted from the plain and grassy terrains. The normals for the plain terrain (Figure 5.8a) are relatively consistent and uniformly aligned, indicating a flat and smooth surface with minimal variation. In contrast, the normals for the grassy terrain (Figure 5.8b) exhibit more variability, reflecting the uneven and textured surface characteristic of grass.

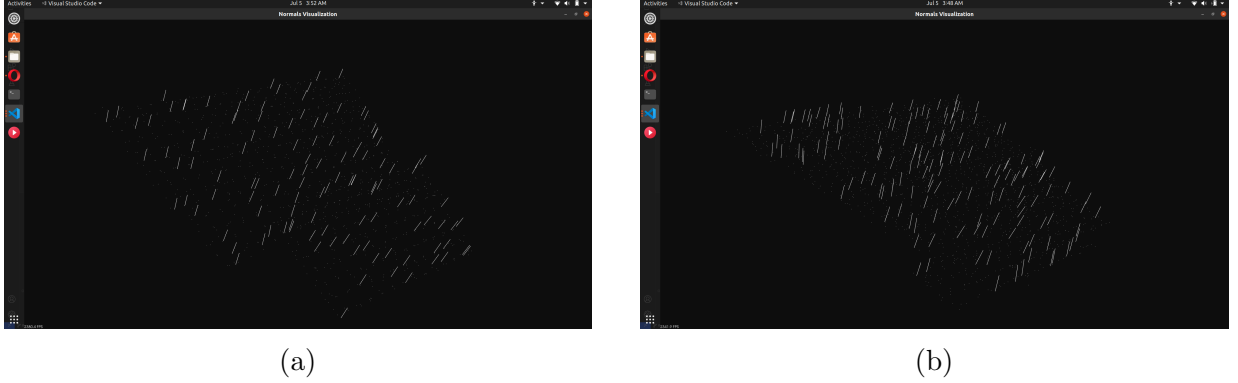


Figure 5.8: Normals extracted from the downsampled RoboSense RS-LiDAR-M1 v2 point clouds for (a) plain terrain, and (b) grassy terrain.

To further analyze the characteristics of the terrain, a scatter plot was generated to visualize the relationship between the X and Y components of the normal vectors for both terrains. As shown in Figure 5.9, the scatter plot reveals a compact and well-separated distribution of normal components for each type of terrain. Plain terrain normals (depicted in purple) are tightly clustered, reflecting the uniformity of the flat surface, whereas grassy terrain normals (depicted in green) display a distinct cluster, although with a wider spread due to the natural variability in grass texture.

The minimal overlap between the normal vector clusters for the two types of terrain suggests these features are highly discriminative, making the development of a classifier to distinguish between plain and grassy terrains relatively straightforward. This contrasts with the results obtained with the Cyglidar D1 sensor, where the significant overlap in normal components posed challenges for effective classification.

5.3.3 Results using SVM with Linear Kernel

The table below presents the real-time performance results of terrain classification using the SVM with a linear kernel on the Jetson Nano. Due to the clear separation between

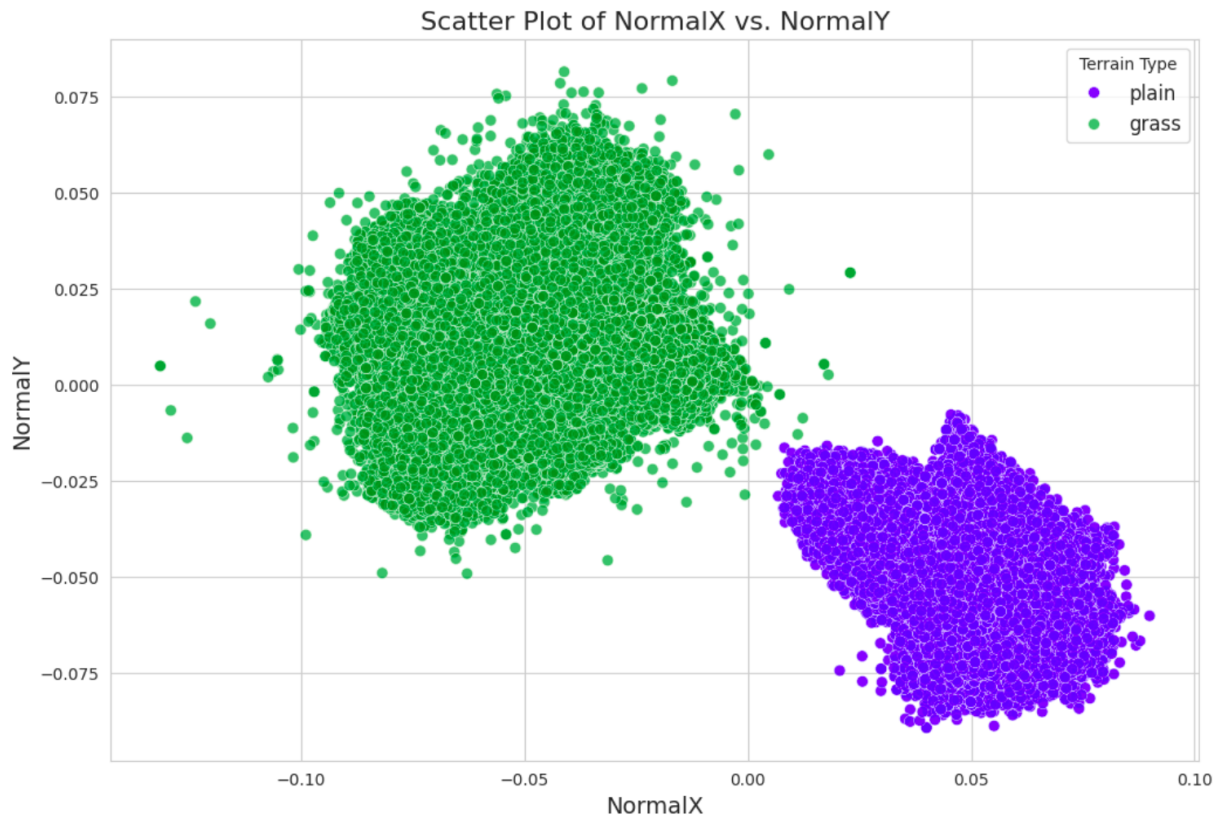


Figure 5.9: Scatter plot of NormalX vs. NormalY for plain and grassy terrains using RoboSense RS-LiDAR-M1 v2.

the classes in the normal vector components (as highlighted in the scatter plot analysis), a linear kernel was chosen for this classification task. This choice was made to simplify the model and reduce computational overhead while maintaining high accuracy.

The results demonstrate that the model performs exceptionally well in real-time conditions, achieving a very high precision of 1 for both terrain classes, a recall of 1 for both, and an F1 score of 1 for both terrain classes. This indicates that the system almost perfectly categorizes terrain types without misclassifying them. These metrics reflect the model’s robustness and reliability in detecting changes in terrain in real time.

In addition, the total processing time, including pre-processing, feature extraction, and prediction, is approximately 17.68 ms. This indicates that the system is capable of making terrain classification decisions nearly instantly, making it highly suitable for real-time applications in embedded systems.

The efficiency of the linear kernel, combined with the precision of the RoboSense LiDAR sensor, results in minimal prediction time, highlighting the feasibility of deploying this system in scenarios where fast and reliable terrain classification is critical.

Metric	Value (ms)
Preprocessing Time	2.03
Feature Extraction Time	2.40
Prediction Time	13.24
Total Time Taken	17.68

Table 5.8: Real-time performance results using SVM with Linear Kernel on Jetson Nano.

In addition to the processing time metrics, Table 5.9 summarizes the key performance metrics for the SVM with the linear kernel for both terrain classes, plain and grass:

Class	Precision	Recall	F1 Score	Support
Plain	1.0	1.0	1.0	156435
Grass	1.0	1.0	1.0	251749

Table 5.9: Precision, Recall, F1 Score, and Support for SVM with Linear Kernel.

The confusion matrix further confirms the high precision and recall values reported earlier. The model misclassified only 1 instance of grass as plain and 2 instances of plain as grass out of a total of over 400,000 instances. This demonstrates the model’s ability to accurately classify terrain types in real time using the RoboSense LiDAR sensor. The

Table 5.10: Confusion Matrix for SVM with Linear Kernel.

	p	n	total
p'	156434	1	P'
n'	2	251747	N'
total	P	N	

model’s ability to correctly classify all terrain frames as negatives indicates that every data point in this dataset was identified as negative, thereby enhancing the model’s robustness for this particular use case. This classification performance highlights the model’s high accuracy in classifying terrain types in real-time using the RoboSense LiDAR sensor. It demonstrates the robustness and reliability of the model for this specific task, as it correctly classified both positive and negative examples in the dataset. This results in nearly perfect performance, demonstrating the effectiveness and reliability of the SVM with the linear kernel in this application.

To sum up, the linear kernel streamlines the classification procedure while delivering outstanding accuracy in real-time applications. The precision, recall, and F1 score underscore the system’s dependability when using high-quality LiDAR data. This method’s effectiveness is notably boosted by the distinct separation between terrain features, enabling efficient classification without resorting to more complex and computationally intensive kernels.

5.3.4 Performance Metrics Across 5 K-Folds for Terrain Classification

Table 5.11 presents a detailed breakdown of the performance metrics across the 5 k-fold cross-validation iterations for terrain classification using the RoboSense LiDAR sensor. These metrics provide insight into the consistency and robustness of the model.

The table illustrates the uniformity of the model’s performance metrics across all folds,

Table 5.11: Performance Metrics Across 5 K-Folds for Terrain Classification

Metric	Mean	Standard Deviation	Variance	Minimum	Maximum	Second Moment
Accuracy	1.0000	0.0000	0.0000	1.0000	1.0000	1.0000
Precision	1.0000	0.0000	0.0000	1.0000	1.0000	1.0000
Recall	1.0000	0.0000	0.0000	1.0000	1.0000	1.0000
F1-Score	1.0000	0.0000	0.0000	1.0000	1.0000	1.0000

reflecting the reliability of the linear kernel SVM for this application. Below are the key observations derived from the results:

- **Accuracy:** The model consistently achieved an accuracy of **100%** across all folds. This exceptional accuracy confirms the model’s ability to classify terrain types (plain and grass) with complete reliability.
- **Precision:** Precision, which measures the ratio of true positives to all predicted positives, also reached **100%**, indicating the model’s effectiveness in avoiding false positives.
- **Recall:** The recall metric, indicative of the model’s ability to identify all relevant instances (true positives), similarly achieved a value of **100%**. This ensures that no true terrain instances were missed during classification.
- **F1-Score:** The F1-score, a harmonic mean of precision and recall, was also **100%**, reflecting the perfect balance between these two critical metrics.

Key Insights:

1. **Stability:** The metrics displayed no variability across the folds, as evidenced by the standard deviation and variance values of **0.0000**. This underscores the robustness of the model.
2. **Perfect Classification:** With no misclassifications in any fold, the confusion matrices for each fold exhibited complete separation of the terrain types.
3. **Effectiveness of Features:** The results highlight the effectiveness of the normal vector features (**NormalX** and **NormalY**) in enabling the model to achieve such high performance without the need for more complex kernels or additional features.

These results demonstrate the exceptional performance of the terrain classification pipeline, particularly when leveraging high-accuracy LiDAR data from the RoboSense sensor. Such precision and reliability make this system highly suitable for real-time applications in assistive technologies, where rapid and accurate terrain detection is critical.

5.4 Terrain Classification using the Higher Accuracy LiDAR (Robosense) with Added Noise

In this section, the RoboSense RS-LiDAR-M1 v2 sensor was used with artificially added Gaussian noise to simulate conditions analogous to those of a lower-accuracy, cost-effective LiDAR sensor. The primary objective of this analysis was to assess the resilience and robustness of the terrain classification algorithm in these degraded conditions, which are often encountered in real-world practical scenarios.

Gaussian noise was introduced into the point cloud data at varying levels, specifically 6 and 10 mm. This degradation was designed to test how the classification model performs when the quality of the data is intentionally compromised. The noise addition process, detailed in the Methods section, aimed to simulate the challenges faced by systems using low-cost LiDARs, which inherently produce noisier and less precise data.

The impact of this noise on the point clouds is visually represented in Figures 5.10a and 5.10b. As seen, the 6mm noise introduces some dispersion, but the overall structure of the terrain remains recognizable. However, the 10mm noise level results in a significantly more distorted point cloud, with greater dispersion and less coherence in the data points. These images clearly show how increasing noise levels lead to a rougher and less detailed representation of the environment, which in turn complicates the task of terrain classification.

Following noise addition, the data was processed to extract normal vectors from the downsampled point clouds. The scatter plot of the X and Y components of these normals, shown in Figure 5.11, provides insight into how noise affects the distribution of normals. With 10 mm of noise, the scatter plot reveals a much wider spread of the points, indicating a higher degree of variability in the surface orientation. This increased variability reflects the challenge of distinguishing between plain and grassy terrains when data are degraded. The overlap between the distributions for plain and grassy terrains is more pronounced than in the case of clean data, which can complicate the classification process.

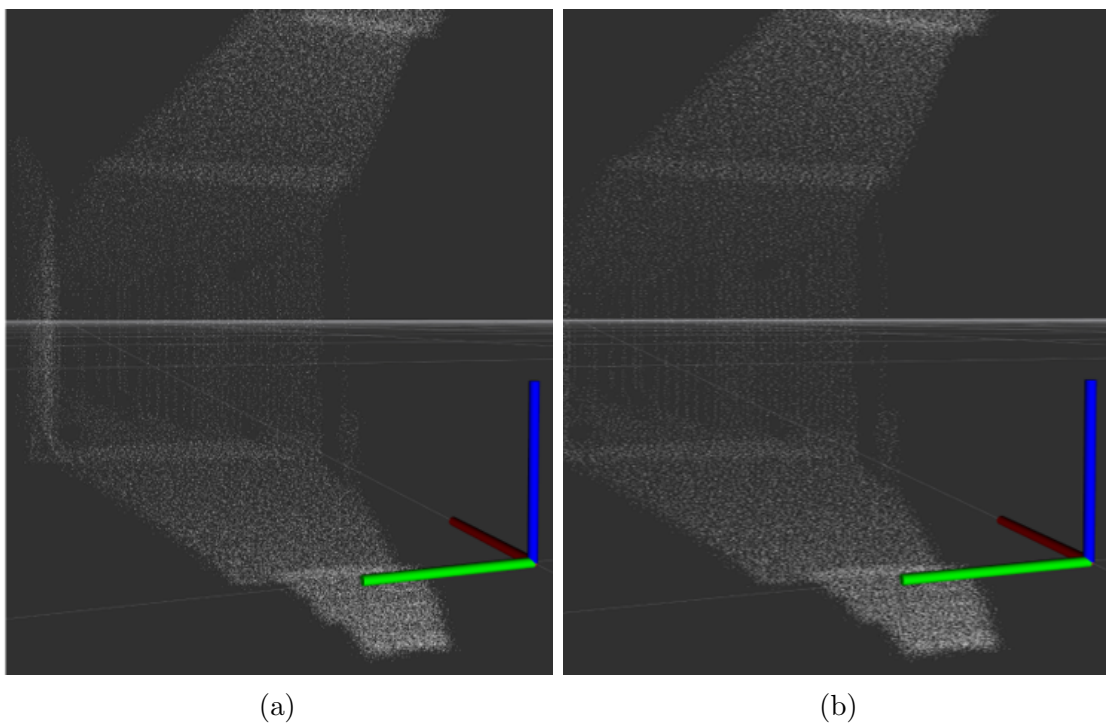


Figure 5.10: Processed point clouds with added Gaussian noise: (a) 6 mm noise, (b) 10 mm noise.

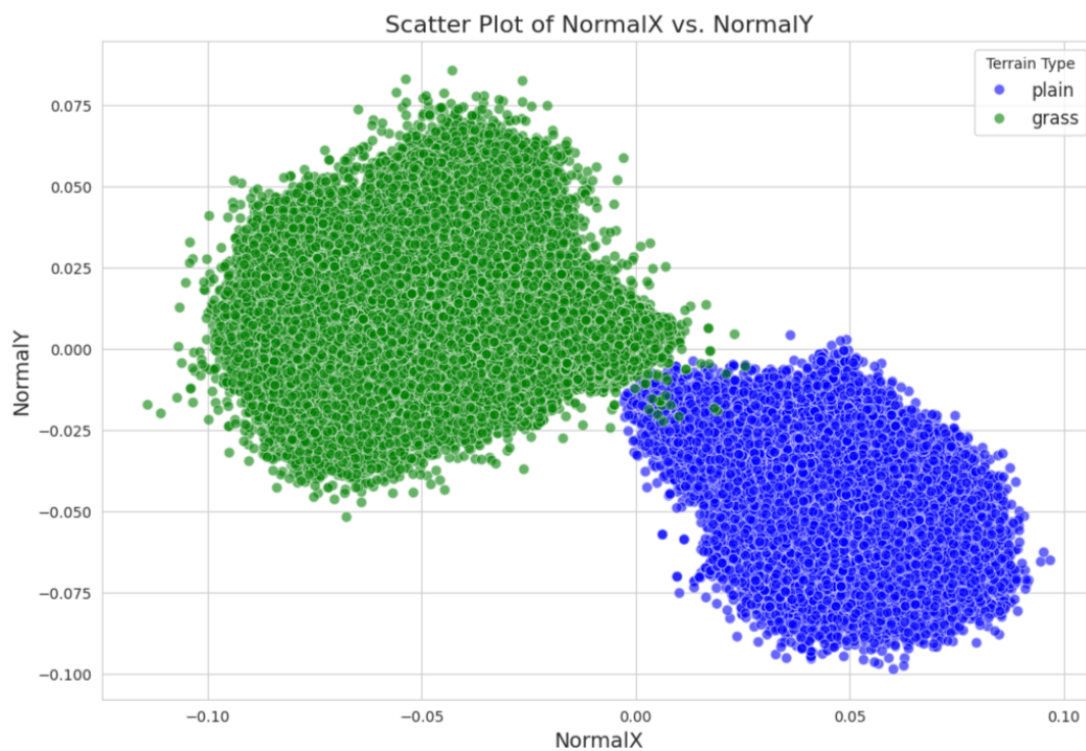


Figure 5.11: Scatter plot of NormalX vs. NormalY for plain and grassy terrains with 10 mm Gaussian noise added.

5.4.1 Results using SVM with Linear Kernel on Noisy Data

To evaluate the real-time capabilities of the terrain classification model under noisy conditions, the SVM with a linear kernel was utilized, similar to the previous section. This section presents the performance metrics obtained by running the model on the Jetson Nano using the point-cloud data **with 10 mm noise**.

The inclusion of noise introduces additional challenges for the classifier, primarily due to the increased overlap between the normal vector distributions of the two terrain types. Despite these difficulties, the linear kernel SVM model **trained solely on clean data** continued to show robust performance, though with a slight decline compared to the outcomes with clean data. The table below summarizes the results:

Metric	Value (ms)
Preprocessing Time	2.30
Feature Extraction Time	2.40
Prediction Time	15.81
Total Time Taken	20.51

Table 5.12: Real-time performance results using SVM with Linear Kernel on Jetson Nano on Noisy LiDAR Data.

The small increase in prediction time and overall processing time 5.12 indicates that the classifier can still operate efficiently under degraded conditions, making it suitable for real-world applications where sensor noise is often unavoidable. Moreover, these results highlight the potential of using higher accuracy LiDAR sensors, even with added noise, to achieve reliable terrain classification in embedded systems with constrained computational resources. The resilience of the classification algorithm in such scenarios underscores the importance of sensor precision and quality in real-time applications.

In addition to the processing time metrics, Table 5.13 summarizes the key performance metrics for the SVM with the linear kernel for noisy LiDAR data:

Class	Precision	Recall	F1 Score	Support
Plain	1.00	0.98	0.99	155821
Grass	0.99	1.00	0.99	234360

Table 5.13: Precision, Recall, and F1 Score for SVM with Linear Kernel for Noisy LiDAR Data.

Compared to the results obtained with clean high-accuracy LiDAR data, the total processing time increased slightly, primarily due to the more complex point cloud data generated by the noise addition. The F1 score remains high at 0.9936, which is only slightly lower than the 0.999974 obtained with clean data. The recall remains nearly perfect at 0.9999, showing a minimal reduction compared to 0.999948 for the clean LiDAR data. Precision dropped slightly to 0.9874 due to the slight increase in false positives introduced by the noise. Despite these minor reductions in performance, the model shows remarkable resilience in handling noisy data while maintaining real-time processing capability.

Table 5.14: Confusion Matrix for SVM with Linear Kernel on Noisy Data.

	p	n	total
p'	152841	2980	P'
n'	19	234341	N'
total	P	N	

Despite the introduction of noise, the model demonstrated an impressive ability to classify both plain and grass terrains with minimal misclassifications. The higher number of false positives reflects the challenge of distinguishing terrain types in noisy environments. Nevertheless, the high recall and F1 score emphasize the model's robustness, and its real-time capability was maintained with only a slight increase in processing time. These results highlight the practicality of the linear kernel SVM for real-time terrain classification using noisy LiDAR data, particularly when paired with a robust processing pipeline.

The results with the RoboSense RS-LiDAR-M1 v2 further underscore the potential of high-accuracy LiDAR sensors in achieving reliable real-time terrain classification. The pipeline's performance, characterized by near-perfect F1 scores, demonstrates its adaptability and robustness in handling noise, making it highly suitable for practical deployment in mobility-assistive technologies. This scalability emphasizes the system's ability to meet the demands of diverse and dynamic environments, further validating the approach for real-world applications.

5.4.2 Performance Metrics Across 5-Fold Cross-Validation for Terrain Classification with 10mm Gaussian Noise

The performance of the SVM classifier with a linear kernel was evaluated for terrain classification using the RoboSense LiDAR data after injecting 10mm of Gaussian noise. The results highlight the robustness of the classifier in handling noisy data, achieving consistently high metrics across all folds. Table 5.15 presents the detailed metrics obtained from the cross-validation process.

Table 5.15: Performance Metrics Across 5-Fold Cross-Validation for Terrain Classification (10mm Gaussian Noise)

Metric	Mean	Standard Deviation	Variance	Minimum	Maximum	Second Moment
Accuracy	0.9985	0.0001	0.0001	0.9983	0.9987	0.9970
Precision	0.9985	0.0001	0.0001	0.9984	0.9987	0.9970
Recall	0.9990	0.0001	0.0001	0.9988	0.9992	0.9980
F1-Score	0.9988	0.0001	0.0001	0.9986	0.9989	0.9975

The results presented in Table 5.15 show a minimal standard deviation and variance across folds, indicating the consistency of the classifier in terrain classification despite the introduction of noise. Below are the key observations:

- **Accuracy:** The classifier achieved an average accuracy of **99.85%**, with a standard deviation of **0.01%**, demonstrating its ability to generalize well across different data splits.
- **Precision:** An average precision of **99.85%** was observed, reflecting the model’s ability to avoid false positives even in noisy conditions.
- **Recall:** The recall metric, which measures the detection of true positives, averaged at **99.90%**, highlighting the classifier’s robustness in identifying all relevant instances despite noise.
- **F1-Score:** The F1-Score of **99.88%** indicates a strong balance between precision and recall, confirming the classifier’s suitability for noisy data scenarios.

The high performance across all metrics emphasizes the resilience of the linear kernel SVM in noisy environments. These findings validate the approach for real-world terrain classification tasks, particularly in scenarios involving mobility-assistive devices where noisy sensor data is common.

Error Analysis

While the SVM with a linear kernel performed well overall in noisy conditions, misclassifications did occur, as indicated by the confusion matrix in Table 5.14. Specifically, the model produced 2980 false positives (incorrectly classifying grass as plain) and 19 false negatives (incorrectly classifying plain as grass). This section analyzes the potential reasons for these errors and provides insight into the challenges posed by noisy data.

One key explanation for these misclassifications lies in the effect of noise on the normal vector distributions extracted from the point cloud. The introduction of 10 mm Gaussian noise tends to scatter the points slightly more than usual, which can affect the calculation of the surface normals. For flat, smooth surfaces such as plain terrain, the normals are typically tightly clustered and aligned. However, when noise is introduced, the calculated normals for plain terrain may become more spread out, resembling those from grassy terrain, which naturally exhibits more variance in its surface normals due to its inherent texture. This increased spread in normal vectors for plain terrain could result in misclassification, as the system may interpret this spread as characteristic of a grassy surface. This finding aligns with Lalonde et al.’s observations [29], where noise and variability in LiDAR data led to feature overlap, complicating terrain classification.

Sensor misalignment or environmental interference, such as uneven ground, can further distort point cloud data, exacerbating the overlap between the feature spaces of plain and grassy terrains. As Fareed et al. [14] highlighted, robust noise handling is critical when working with low-cost LiDAR sensors to mitigate the effects of degraded resolution. Similarly, Lalonde et al.’s work demonstrated that preprocessing techniques, such as local planarity assessment, could aid in distinguishing terrains even in the presence of noise, but such methods require computational resources that may not be feasible for resource-constrained systems like the Jetson Nano.

Additional factors contributing to these errors include:

- **Sensor Orientation and Positioning:** Depending on the position of the LiDAR relative to the surface, certain angles can amplify the effects of noise. For instance, if the LiDAR captures both plain and grassy terrain within the same frame, mixed signals may arise, leading to misclassification. These findings align with the challenges reported by Vandapel et al. [65], who observed difficulties in segmenting point clouds when multiple terrain types were present in a single scan.
- **Edge Cases:** In instances where the LiDAR captures frames near the transition between grassy and plain terrains, the system may struggle to determine the appropriate class. This scenario is consistent with observations by Lalonde et al. [29],

where boundaries between terrain types posed significant challenges in achieving accurate segmentation.

- **Vegetation and Obstacles:** Small patches of vegetation on plain terrain or reflective surfaces, such as wet patches, can introduce irregularities in the point cloud, deviating from the expected normal distribution. These findings mirror those of Kragh et al. [27], who highlighted the difficulties in distinguishing between ground, vegetation, and objects when using LiDAR data in agricultural fields.
- **Lighting and Atmospheric Conditions:** Although LiDAR is generally robust to lighting, environmental factors like fog, rain, or strong sunlight can degrade data quality. Moore et al.’s video-based approach [40] faced similar challenges, albeit exacerbated by image dependency, whereas LiDAR-based methods remain comparatively less sensitive but not immune to atmospheric variations.

Potential Mitigations

Several strategies could be considered to reduce these misclassifications and improve the overall performance of the classifier in noisy environments:

- **Enhanced Preprocessing:** Implementing additional noise filtering techniques, such as bilateral filtering, could help smooth out the point cloud data before normal estimation. This could reduce the scattering effect of noise on the normals and make the features more distinct for classification.
- **Adaptive Thresholding for Normals:** Developing an adaptive thresholding technique that dynamically adjusts the classification boundary based on the noise level or surface irregularity could help mitigate the overlap between plain and grassy terrain. This would allow the classifier to be more resilient to noise-induced distortions in the normal vectors.
- **Multimodal Sensor Fusion:** Integrating additional sensors, such as stereo cameras or IMUs, could help provide complementary data to the LiDAR, enhancing the overall robustness of the system. For example, an IMU could provide information about the orientation and movement of the device, which could help correct any misalignments in the LiDAR data. Kragh et al. [27] demonstrated the benefits of multimodal systems in handling complex terrains.

- **Temporal Smoothing:** Incorporating temporal smoothing, where the classification is averaged over several consecutive frames, as suggested by Lalonde et al., could help reduce the impact of noisy data in any single frame. This would allow the system to make more reliable predictions by considering the overall trend in the data rather than relying solely on individual frames.

5.5 Conclusion

The RoboSense RS-LiDAR-M1 v2 demonstrated significantly better performance in terrain classification, achieving near-perfect F1 scores, even under noisy conditions. This highlights the importance of sensor accuracy in producing cleaner point clouds for robust feature extraction and classification. In contrast, the Cyglidar D1, while cost-effective, exhibited limitations in handling noise and overlapping features, leading to lower recall and F1 scores in terrain classification tasks.

These results reveal a trade-off between sensor cost and classification performance, emphasizing the need for a preprocessing pipeline that can mitigate the impact of noise, particularly when using low-cost sensors like the Cyglidar D1. The pipeline’s flexibility in handling data from both sensors underscores its adaptability across hardware configurations.

The misclassifications observed in this analysis highlight the challenges of classifying terrain in real-time using noisy LiDAR data. Although the SVM of the linear kernel performed well in general, noise-induced distortions in the point cloud, particularly affecting the normals for plain terrain, contributed to some errors. By implementing additional noise handling techniques, improving sensor placement, or integrating complementary data sources, it may be possible to further reduce these errors and enhance the system’s performance in real-world scenarios.

Chapter 6

Conclusion and Future Work

This chapter synthesizes the findings from the terrain classification and stairs detection experiments, outlines the limitations encountered throughout the research, provides recommendations for LiDAR sensor specifications, and proposes directions for future work. The results obtained from the experiments using the Cyglidar D1 and RoboSense RS-LiDAR-M1 v2 sensors have provided significant insights into the challenges and possibilities of real-time terrain classification and stairs detection in embedded systems.

The results of this research are particularly relevant to the development of mobility-assistive technologies, such as smart walkers and exoskeletons, where robust real-time terrain classification and stairs detection are critical to ensuring user safety and autonomy.

6.1 Summary of Findings

This research explored the effectiveness of terrain classification and stairs detection using two different LiDAR sensors: the Cyglidar D1, a lower-accuracy sensor, and the RoboSense RS-LiDAR-M1 v2, a higher-accuracy sensor. The findings provide valuable insights into the challenges and opportunities presented by these sensors in real-time applications on embedded platforms like the Jetson Nano.

These findings provide crucial insights for assistive robotics, where real-time perception systems are necessary to navigate complex environments and enhance the mobility and independence of users with physical disabilities.

- **Stairs Detection with Cyglidar D1:** The detection of ascending stairs using the Cyglidar D1 sensor was a critical component of this research. Despite the sensor's

limitations in resolution and noise handling, the Support Vector Machine (SVM) classifier with a Radial Basis Function (RBF) kernel demonstrated that effective stair detection is possible even with lower-accuracy hardware. Both wearable and tripod configurations were utilized to capture diverse perspectives.

The classifier achieved an average F1 score of 96.96%, which is notable given the limitations of the sensor. However, certain conditions, such as rapid movement and environments with high reflectance (e.g., hallways with reflective surfaces), led to misclassifications. False positives primarily occurred in fast-moving scenarios, where objects like walls or corners were misinterpreted as stairs. False negatives—where actual stairs were missed—were more critical, occurring most often during quick approaches or when the sensor’s field of view was partially obstructed. These results indicate that while the Cyglidar D1 can effectively detect stairs in controlled environments, its performance is hampered in more dynamic conditions.

This underscores the importance of proper sensor placement and preprocessing techniques, both of which significantly influence classification accuracy. The Cyglidar D1’s limitations in resolution and noise handling were evident, but the classifier’s performance remained sufficient for detecting stairs in less complex environments.

- **Terrain Classification with Cyglidar D1:** The terrain classification experiments with the Cyglidar D1 sensor presented more significant challenges. The sensor’s lower accuracy resulted in substantial overlap between the normal vectors for plain and grassy terrains, making it difficult to achieve high classification results. The scatter plots of normal vectors illustrated this overlap, with the RBF kernel struggling to effectively separate the two classes.

Real-time performance was constrained due to the computational demands of the RBF kernel, which is computationally expensive and less suited for embedded systems like the Jetson Nano. Despite these limitations, the findings provide useful insights into the challenges of using lower-accuracy LiDAR sensors for terrain classification. The need for more distinct features, better preprocessing, and possibly alternative classification methods became evident when working with noisier and less detailed data.

- **Terrain Classification with RoboSense RS-LiDAR-M1 v2:** The transition to the higher-accuracy RoboSense RS-LiDAR-M1 v2 sensor significantly improved classification performance. This sensor’s higher resolution and precision led to a clearer separation of normal vectors between plain and grassy terrains, as evidenced by more compact and distinct scatter plots.

Even with a simpler linear SVM kernel, the classification F1 score was significantly higher—reaching 99.90%—and the processing times were much faster compared to the RBF kernel used with the Cyglidar D1. The linear kernel’s simplicity and efficiency made it particularly advantageous, allowing the system to operate in real-time on the Jetson Nano with minimal computational overhead. These results highlight the benefits of using a high-accuracy sensor for terrain classification in embedded systems, where both accuracy and processing speed are essential.

- **Impact of Noise on Terrain Classification:** To simulate conditions that might be encountered with lower-accuracy, cost-effective LiDAR sensors, Gaussian noise (up to 10 mm) was added to the RoboSense LiDAR data. This experiment aimed to assess the robustness of the terrain classification algorithm under degraded data conditions. Even with significant noise, the algorithm maintained a high F1 score of 99.36% and showed only a slight increase in processing times compared to clean data.

Scatter plots of normal vectors with added noise revealed increased overlap between the two terrain classes, reflecting the challenge of distinguishing between them when data quality is compromised. Nonetheless, the classifier’s ability to handle this noise while maintaining high F1 scores underscores its robustness and suitability for real-world applications, where sensor data may often be noisy or degraded.

6.2 Limitations

Despite the promising results obtained in this research, several limitations were encountered that should be acknowledged. These limitations are categorized into algorithmic, dataset, and hardware constraints that influenced the overall performance and generalizability of the findings.

- **Algorithmic Limitations:** The reliance on a binary classification model, particularly for stairs detection, constrained the algorithm’s ability to handle complex scenes with multiple transitions or edge cases. The use of an SVM with an RBF kernel was effective in certain scenarios but came with significant computational overhead. This high computational demand made it challenging to achieve real-time performance, especially on resource-constrained platforms like the Jetson Nano.

Additionally, the frame-by-frame processing approach did not fully capture the temporal context of the scenes, which is critical in dynamic environments. The lack of

temporal information likely contributed to the misclassification of fast-moving objects or transitions between stairs and non-stair scenes. Furthermore, focusing solely on SVM models limited the exploration of other classification techniques that might have provided better performance, especially for larger and more complex datasets. Algorithms such as deep learning models, random forests, or ensemble methods were not explored due to the computational constraints but could potentially offer improved accuracy and robustness.

- **Dataset Limitations:** The dataset used in this research, particularly for stairs detection, was relatively small and lacked diversity. This limitation affected the ability of the classifier to generalize across different environments and scenarios. The k-fold cross-validation technique was employed to maximize the utility of the available data, but the small dataset size inherently constrained the model’s exposure to a wide variety of scenes and conditions.

A more extensive and diverse dataset would be necessary to fully validate the effectiveness of the developed models. The dataset primarily included data from specific terrains and stairs, which may not represent the variability encountered in real-world applications. For instance, variations in stair design, material, and lighting conditions were not comprehensively captured, limiting the applicability of the results to other environments. The limited number of samples also restricted the ability to train more complex models that typically require larger datasets to achieve optimal performance.

- **Hardware Limitations:** The Jetson Nano, chosen for its balance between cost and processing power, imposed significant limitations on the complexity of the algorithms and the resolution of the point clouds that could be processed in real-time. While the platform was sufficient for demonstrating the feasibility of real-time terrain classification and stairs detection, its limited computational resources restricted the use of more advanced machine learning models, such as deep neural networks, which could potentially offer better performance.

The hardware constraints also influenced the preprocessing steps, such as the degree of downsampling applied to the point cloud data. While downsampling was necessary to reduce the computational load, it also led to a loss of detail that could have contributed to better classification accuracy. The Jetson Nano’s limited memory and processing power required careful optimization of the algorithms to ensure real-time performance, but this optimization often came at the cost of reduced accuracy or the exclusion of potentially useful features from the data.

Furthermore, the use of a single LiDAR sensor in each experiment limited the ability to compare results across different hardware setups. While the RoboSense RS-LiDAR-M1 v2 provided higher accuracy and better performance than the Cyglidar D1, the conclusions drawn about the effectiveness of these sensors are specific to the hardware used and may not generalize to other LiDAR models or configurations.

- **Reliance on Frame-by-frame Processing:** The algorithm’s reliance on frame-by-frame processing, while sufficient for controlled experiments, may not fully address the dynamic and complex environments encountered in mobility-assistive devices.

These limitations highlight areas where future research could focus on improving the generalizability and robustness of the terrain classification and stairs detection models, particularly in the context of real-time applications on resource-constrained hardware.

6.3 Recommendations

This section outlines recommendations based on the findings of this thesis, which examined the performance of LiDAR-based systems for stairs detection and terrain classification in real-time applications. Although the results demonstrated the potential of both low- and high-resolution LiDAR sensors, it is important to emphasize that these recommendations are made within the limited scope of this research. The recommendations provided here are based on the data and algorithms tested within this study and do not constitute comprehensive guidelines. Further testing with a broader range of datasets, sensors, and environments is necessary to fully validate these recommendations.

Based on practical experience with handling and operating the LiDAR sensors investigated, the following recommendations aim to provide insights into the trade-offs between different sensor attributes. Selecting the appropriate LiDAR sensor for real-time applications involves balancing several critical factors: resolution, noise tolerance, field of view, range, and cost. Each of these factors can significantly influence the system’s overall performance in detecting stairs or classifying terrain. The specific needs of a particular application, such as precision, speed, and environmental conditions, should guide the decision-making process when choosing a LiDAR sensor. Drawing from the experiences gained in this thesis. However, please note these recommendations based on preliminary investigations in a small set of devices and should be considered with care.

- **Resolution:** The results of this thesis indicate that sensor resolution plays a key role in distinguishing between complex features such as stairs detection and terrain-type

classification. However, based on specific application requirements, an ultra-high-resolution sensor may not always be necessary. A resolution closer to 0.2° , similar to the Robosense LiDAR, should provide sufficient detail for tasks such as stairs detection and terrain classification, especially when combined with robust post-processing algorithms. In environments where precision is less critical and the features are more pronounced (e.g., clear transitions between terrain types), this resolution can offer a good balance between cost and computational demands. The use of slightly lower resolution (closer to 0.2° than 0.75°) can still maintain acceptable performance, as long as efficient data filtering and feature extraction techniques are employed. Finally, a resolution ranging from 0.4° to 0.6° could be sufficient for the applications of stairs detection and terrain classification.

- **Noise Tolerance:** Noise tolerance is a critical factor for real-time systems, particularly in environments where sensor data may be compromised by reflections, movement, or environmental factors. In this thesis, noise tolerance was tested using Gaussian noise, with the algorithm maintaining high accuracy up to 10 mm of added noise. Based on this, it is recommended that sensors chosen for similar applications should be able to tolerate noise levels of 10 mm or less without significant performance degradation. However, further testing with other types of noise, such as Poisson or Salt-and-Pepper noise, may provide a more comprehensive understanding of noise impacts. Ideally, sensors should come equipped with internal noise reduction capabilities or be integrated with preprocessing techniques like Kalman filtering to enhance data reliability. Overall, it is the researchers' view that resolution plays a more significant role than noise tolerance.
- **Range and Field of View:** For stairs detection and terrain classification in real-time, the field of view (FOV) and range of the sensor are crucial for providing the system with enough context to detect changes in the environment in advance.

Range: While the RoboSense RS-LiDAR-M1 v2 offers a much longer range, a range of 5 to 7 meters should be sufficient to detect stairs or terrain transitions early enough to allow the system to respond in real-time for the specific application of stairs and terrain classification. This range provides enough lead time for decision-making and control without overburdening the system with excessive data from distant objects. It should be noted that the appropriate range may vary depending on the application, and systems operating in larger, more dynamic environments may require a longer range and wider FOV. For wearable systems or close-range applications, the Cyglidar D1's specifications are more than adequate.

Field of View: It is difficult to suggest a particular range of FOV, as it depends

on the application and setup of the sensor on the equipment. Considering LiDARs are usually less accurate/precise around its boundaries, the common practice of processing LiDAR data has been to crop towards the center of the frame, similar to a passthrough filter where the sensor frame is cropped from all three axes to focus on the region of interest.

Let's examine the FOV for the following applications:

- **Stairs Detection:** To effectively detect stairs, it's important to clearly define the problem's scope. For instance, if the goal is to identify stairs while a person ascends and then pauses at the mid-stair landing before looking at the stairs, a broader FOV might be essential, similar to what a RoboSense sensor offers. In contrast, for most scenarios, a FOV comparable to the Cyglidar D1, which is 60° by 120° , should suffice. This range allows for the detection of several stair steps and ensures the stairs are identified as soon as they enter the sensor's frame. Additionally, the detection success is influenced by the user's turning rate and the placement of the LiDAR sensor.
- **Terrain Classification:** For identifying various terrains, a sensor with a significantly smaller FOV can be sufficient since the area of interest is merely the section directly ahead of the user. Therefore, a sensor with a FOV comparable to Cyglidar D1 should be adequate.
- **Processing Speed:** Real-time systems depend heavily on the sensor's ability to capture and transmit data quickly. In this study, Cyglidar D1 operated at 15 Hz, which was sufficient for moderate-speed applications. For applications requiring faster response times, sensors that can capture data at higher frequencies (e.g., 20 Hz or more) may be necessary. Ensuring that the sensor's data output speed is compatible with the system's processing capabilities is essential to maintain the real-time nature of the application, particularly in dynamic environments or with moving objects.
- **Cost-Effectiveness:** High-end sensors such as RoboSense RS-LiDAR-M1 v2 deliver superior performance but come at a significant cost, which could be prohibitive in consumer-level or large-scale deployments. More affordable options, like the Cyglidar D1, provide adequate performance for applications where ultra-high precision is not necessary. As LiDAR technology evolves, the cost-performance trade-off is expected to narrow. With ongoing advancements in manufacturing and scaling of solid-state LiDARs, it is likely that future iterations of lower-cost sensors will offer higher precision and accuracy. Therefore, more affordable sensors with improved performance are expected to become available, further lowering the barriers to the

widespread adoption of LiDAR in real-time applications such as assistive devices and autonomous systems.

In conclusion, these recommendations, drawn from the limited experience of this research, suggest that selecting the right LiDAR sensor requires careful consideration of trade-offs between resolution, noise tolerance, field of view, range, and cost. Although high-performance sensors offer significant advantages in terms of accuracy and reliability, their cost may not always be justified, especially in applications where moderate accuracy is sufficient. Further testing across various conditions and with different sensors would provide more conclusive insights into the optimal configurations for real-time terrain classification and stairs detection systems.

6.4 Future Work

Future research directions have been identified based on the limitations and findings of this study. These areas offer potential improvements in the performance, robustness, and applicability of LiDAR-based terrain classification and stairs detection systems.

- **Alternative Preprocessing Techniques:** The preprocessing of LiDAR data plays a critical role in the overall effectiveness of classification algorithms. Future work could explore more sophisticated preprocessing methods, especially for lower-accuracy sensors like the Cyglidar D1. One potential avenue is time-based downsampling, which averages data points over time to reduce variability and create a more stable representation of the scene. This method could be particularly useful in dynamic environments where the scene changes rapidly. Moreover, implementing a voting or confidence-based system that evaluates multiple frames over a sliding time window could enhance the reliability of the classification by reducing the impact of transient noise and sensor errors.
- **Feature Engineering and Alternative Classification Methods:** Although this study focused on SVM classifiers with RBF and linear kernels, deeper investigation of alternative classification techniques is recommended. Methods such as k-Nearest Neighbors (kNN), maximum likelihood estimates, or even deep learning approaches could be explored to potentially enhance accuracy or reduce computational overhead. Additionally, feature engineering could be further refined by identifying new features that capture critical aspects of the terrain or stairs more effectively. Alternatively,

feature-free methods like Deep Neural Networks could be employed to automatically learn relevant features from the data, bypassing the need for manual feature extraction. Ensemble methods or hybrid models that combine the strengths of multiple classifiers could also be considered to improve robustness and generalization.

- **Optimization for Computational Efficiency in Onboard Processing:** One of the critical challenges in deploying real-time LiDAR-based terrain classification and stairs detection systems on embedded platforms is achieving computational efficiency without compromising accuracy. Onboard processing is constrained by limited computational resources, such as reduced memory, lower processing power, and limited energy availability, particularly on platforms like the Jetson Nano.

Future work should investigate optimization techniques that can make the existing framework more efficient for onboard processing. For instance, model pruning and quantization methods could be explored to reduce the computational complexity of the SVM classifiers without significant degradation in performance. These techniques involve simplifying the model structure by removing redundant parameters or using lower-precision representations of weights, thereby reducing memory usage and speeding up inference times.

Additionally, implementing parallel processing pipelines tailored for specific hardware architectures, such as GPUs or tensor cores, can significantly enhance computational throughput. Employing lightweight libraries and frameworks optimized for embedded devices, such as NVIDIA TensorRT or ONNX, could also improve runtime performance.

Another promising area is the exploration of low-complexity feature extraction methods that reduce the dimensionality of the input data. Simplifying preprocessing steps, such as normal vector estimation, or incorporating dimensionality reduction techniques like PCA, can alleviate the computational burden of real-time operations. Such methods should be carefully evaluated to ensure they preserve the critical features required for high classification accuracy.

Furthermore, the trade-off between model complexity and processing speed could be better balanced by implementing adaptive models. For example, the system could employ simpler linear classifiers during low-variability scenarios (e.g., flat terrain) and dynamically switch to more complex non-linear classifiers when intricate features (e.g., overlapping stairs or noisy terrain) are detected. This adaptive approach could improve computational efficiency by minimizing the use of resource-intensive algorithms during straightforward classification tasks.

Finally, optimizing energy consumption for onboard processing is a crucial consideration for long-term deployments, particularly in mobile applications such as assistive devices or autonomous navigation. Future research could focus on developing energy-efficient algorithms by optimizing data transfer and memory access patterns, as well as implementing power-aware scheduling strategies for computational tasks.

By addressing computational efficiency in onboard processing, these optimizations will enable the proposed system to meet the stringent real-time requirements of resource-constrained environments, paving the way for robust, scalable, and energy-efficient deployments in real-world applications.

- **Real-Time Temporal Context Integration:** One of the key limitations identified was the algorithm’s reliance on frame-by-frame classification, which does not fully exploit the temporal continuity of the data. Future research could focus on integrating temporal context into the classification process. This could be achieved through the use of memory units, recurrent neural networks, or a voting system that considers the classification results over a sequence of frames. Such approaches could enhance the system’s ability to accurately classify terrains and detect stairs by leveraging the consistency of features across multiple frames, thereby reducing the likelihood of false positives and negatives in dynamic environments.
- **Extended Dataset and Real-World Testing:** The relatively small and controlled dataset used in this study limits the generalizability of the findings. To address this, future work should involve the collection of a larger and more diverse dataset that includes various types of stairs and terrains under different environmental conditions. Real-world testing is crucial for evaluating the model performance in more challenging and unpredictable scenarios. Furthermore, incorporating data from different LiDAR sensors with varying specifications would help refine sensor recommendations and improve overall robustness of the system. Testing in diverse settings, such as outdoor environments with varying lighting and weather conditions, could provide deeper insights into the practical applicability and limitations of the system.
- **Inclusion of Descending Stairs:** This study primarily focused on detecting ascending stairs due to their relative simplicity in terms of data capture and feature extraction. However, the inclusion of descending stairs poses unique challenges that should be addressed in future work. Descending stairs introduce complexities such as more varied user movement patterns, which can affect the stability of the LiDAR data, and a narrower vertical field of view, making it difficult to extract consistent features. Additionally, the risk of misclassification increases as elevation changes introduce more noise into the point cloud data. Addressing these challenges is crucial,

as misclassification in descending stairs could lead to critical safety hazards, such as falls. Future work should aim to collect an extended dataset that includes descending stairs under various environmental conditions. Refinement of the model to handle these challenges will enhance its robustness and advance safely and effectiveness in real-world scenarios where both ascending and descending stairs are present.

- **Exploration of Soft Margin SVM:** While this study focused on the use of hard margin SVMs for terrain classification and stairs detection, future research should explore the implementation of soft margin SVMs. Soft margin SVMs introduce a regularization parameter (C) that allows the model to balance the trade-off between maximizing the margin and tolerating some misclassifications. This approach could enhance the classifier’s robustness in handling noisy or overlapping data, particularly in real-world environments where such variability is prevalent. Investigating the effect of different values of C on the model’s performance could provide valuable insights into the optimal configuration for specific applications. Additionally, the inclusion of soft margin SVMs could improve the system’s ability to generalize across datasets with higher levels of complexity or noise, further extending the applicability of the methodology to diverse environments.

6.5 Conclusion

This study presents a novel framework for terrain classification and stairs detection using LiDAR sensors, specifically designed for real-time embedded applications. Unlike previous approaches that often depend on high-end hardware or camera-based systems, this research demonstrates the practicality of utilizing low-cost LiDAR sensors. By addressing privacy concerns crucial in assistive robotics, it offers a more accessible and secure solution.

The thesis investigates the challenges and opportunities associated with LiDAR-based terrain classification and stairs detection, with a focus on the performance of various sensors and algorithms in real-time scenarios. It provides a comprehensive analysis of the strengths and limitations of both low- and high-accuracy LiDAR sensors, emphasizing the pivotal roles of sensor precision, cost, and computational efficiency in achieving reliable classification and system performance.

The findings provide a unique contribution to the field of assistive robotics by highlighting practical trade-offs between sensor cost, computational efficiency, and performance, which are rarely addressed in existing literature.

- **Stairs Detection:** The stairs detection experiments using the Cyglidar D1 sensor demonstrated that it is possible to achieve effective stair detection with lower-accuracy sensors, but with some limitations. The SVM classifier with an RBF kernel achieved a commendable accuracy of 96.96%, despite the challenges posed by the sensor’s resolution and noise. However, the system was prone to false positives in fast-moving scenarios and false negatives in cases where the stairs were partially obscured or approached quickly. These findings highlight the need for careful sensor selection and preprocessing techniques, as well as the importance of considering the requirements of the specific application, such as range and resolution, rather than defaulting to the highest performing sensor available.
- **Terrain Classification:** The terrain classification tasks revealed significant performance differences between Cyglidar D1 and RoboSense RS-LiDAR-M1 v2 sensors. Although the Cyglidar D1’s lower resolution posed challenges, leading to more overlap between terrain types, it still provided valuable data when coupled with advanced postprocessing techniques. On the other hand, the RoboSense RS-LiDAR-M1 v2 offered significantly clearer separation of terrain types due to its higher resolution, allowing faster and simpler classification with a linear SVM kernel. However, this comes with an increase in cost and computational overhead. Depending on the application, lower-resolution sensors such as the Cyglidar D1 may suffice, especially in environments with more pronounced terrain features or where cost and computational efficiency are prioritized.
- **Impact of Noise:** The results demonstrated the robustness of the classification algorithm in handling noise, particularly with up to 10mm added Gaussian noise. This capability ensures that the system can function reliably even in noisy environments, which is essential for real-world applications. However, further testing with other noise types would be beneficial in fully understanding the sensor limitations under various conditions.

In conclusion, this research has underscored the importance of carefully balancing sensor performance with cost and computational efficiency in LiDAR-based terrain classification and stairs detection tasks. Although high-end sensors such as the RoboSense RS-LiDAR-M1 v2 significantly improve accuracy and processing speed, more affordable sensors, such as the Cyglidar D1, can still perform adequately for specific applications with the right algorithmic support and pre-processing steps. The findings of this study provide a foundation for future work that may further explore how to maximize performance in cost-sensitive or resource-constrained environments.

References

- [1] ASUS. Zenbook flip s ux370, 2024. Accessed: 2024-09-02.
- [2] Mayank Bansal, Bogdan Matei, Ben Southall, Jayan Eledath, and Harpreet Sawhney. A lidar streaming architecture for mobile robotics with application to 3d structure characterization. In *2011 IEEE International Conference on Robotics and Automation*, pages 1803–1810. IEEE, 2011.
- [3] Behnam Behroozpour, Phillip AM Sandborn, Ming C Wu, and Bernhard E Boser. Lidar system architectures and circuits. *IEEE Communications Magazine*, 55(10):135–142, 2017.
- [4] OpenMP Architecture Review Board. *OpenMP Application Programming Interface Version 4.0*. OpenMP Architecture Review Board, 2013.
- [5] Graham M Brooker. Noise modeling for lidar sensor simulation in 3d environment. *Remote Sensing*, 7(2):1570–1594, 2015.
- [6] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27, 2011.
- [7] Barbara Chapman, Gabriele Jost, and Ruud van der Pas. *Using OpenMP: Portable Shared Memory Parallel Programming*. MIT Press, Cambridge, MA, 2008.
- [8] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.
- [9] Cygbot. 2d/3d dual solid-state tof lidar cyglidar d1. <https://www.cygbot.com/2d-3d-dual-solid-state-tof-lidar>, 2024. Available: <https://www.cygbot.com/2d-3d-dual-solid-state-tof-lidar>.

- [10] Anpei Dai et al. Hsc4d: Human-centered 4d scene capture in large-scale indoor-outdoor space using a lightweight hip-mounted lidar. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10510–10519, 2022.
- [11] PCL Documentation. Pcd file format, 2024.
- [12] Lianming Duan, Zhiyu Zhang, Shiyao Liu, Yong Yin, and Cheng Wang. Low-complexity object detection in 3d lidar point clouds. *Optics and Laser Technology*, 127:106172, 2020.
- [13] Su-Hong Eom, Jeon-Min Kang, Ga-Young Kim, and Eung-Hyuk Lee. A study on gathering staircase information for active staircase entry of wheelchair stair climbing assistive devices. In *ICINCO (1)*, pages 747–753, 2023.
- [14] Nadeem Fareed, Joao Paulo Flores, and Anup Kumar Das. Analysis of uas-lidar ground points classification in agricultural fields using traditional algorithms and pointcnn. *Remote Sensing*, 15(2):483, 2023.
- [15] Stephen Fox, Olli Aranko, Juhani Heilala, and Päivi Vahala. Exoskeletons: Comprehensive, comparative and critical analyses of their potential to improve manufacturing performance. *Journal of Manufacturing Technology Management*, 31(6):1261–1280, 2019.
- [16] A.C. Gonzalez and C. Costilla. Performance evaluation of the nvidia jetson nano through a real-time machine learning application. *Journal of Machine Learning Applications*, 7(1):15–24, 2020.
- [17] Isabelle Guyon and André Elisseeff. Introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.
- [18] Ahsan Habib, Md Milon Islam, Muhammad Nomani Kabir, Motasim Billah Mredul, and Mahmudul Hasan. Staircase detection to guide visually impaired people: A hybrid approach. *Revue d’Intelligence Artificielle*, 33(5), 2019.
- [19] Timo Hackel, Nikolay Savinov, Lubor Ladicky, Jan D Wegner, Konrad Schindler, and Marc Pollefeys. Semantic3d.net: A new large-scale point cloud classification benchmark. *arXiv preprint arXiv:1704.03847*, 2017.
- [20] Dirk Holz, Alexandru Eugen Ichim, Federico Tombari, Radu Bogdan Rusu, and Sven Behnke. *Registration with the Point Cloud Library: A Modular Framework for Aligning in 3-D*, volume 22. IEEE, 2015.

- [21] Weiguang Huo, Samer Mohammed, Juan C Moreno, and Yacine Amirat. Lower limb wearable robots for assistance and rehabilitation: A state of the art. *IEEE systems Journal*, 10(3):1068–1081, 2014.
- [22] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning*. Springer, 2013.
- [23] Ian T. Jolliffe and Jorge Cadima. *Principal Component Analysis*. Springer Series in Statistics. Springer, 2nd edition, 2016.
- [24] Alexandra Kapeller, Heike Felzmann, Eduard Fosch-Villaronga, and Ann-Marie Hughes. A taxonomy of ethical, legal and social implications of wearable robots: an expert perspective. *Science and engineering ethics*, 26:3229–3247, 2020.
- [25] Bimal Koirala, Deepak B. Bista, and Bimal Kumar Mishra. Real-time lidar processing using nvidia jetson platform for self-driving vehicles. In *2019 IEEE 10th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pages 0466–0471. IEEE, 2019.
- [26] Weifeng Kong, Zhiying Tan, Wenbo Fan, Xu Tao, Meiling Wang, Linsen Xu, and Xiaobin Xu. A review of the application of staircase scene recognition system in assisted motion. *Digital Signal Processing*, page 104362, 2023.
- [27] Mikkel Kragh, Rasmus N Jørgensen, and Henrik Pedersen. Object detection and terrain classification in agricultural fields using 3d lidar data. In *International conference on computer vision systems*, pages 188–197. Springer, 2015.
- [28] Andrew Garrett Kurbis, Brokoslaw Laschowski, and Alex Mihailidis. Stair recognition for robotic exoskeleton control using computer vision and deep learning. In *2022 International Conference on Rehabilitation Robotics (ICORR)*, pages 1–6. IEEE, 2022.
- [29] Jean-François Lalonde, Nicolas Vandapel, Daniel F Huber, and Martial Hebert. Natural terrain classification using three-dimensional ladar data for ground robot mobility. *Journal of field robotics*, 23(10):839–861, 2006.
- [30] Brock Laschowski, William McNally, Alexander Wong, and John McPhee. Exonet database: Wearable camera images of human locomotion environments. *Frontiers in Robotics and AI*, 7:562061, 2020.
- [31] Nanxi Li, Chong Pei Ho, Jin Xue, Leh Woon Lim, Guanyu Chen, Yuan Hsing Fu, and Lennon Yao Ting Lee. A progress review on solid-state lidar and nanophotonics-based lidar sensors. *Laser & Photonics Reviews*, 16(11):2100511, 2022.

- [32] You Li and Javier Ibanez-Guzman. Lidar for autonomous driving: The principles, challenges, and trends for automotive lidar and perception systems. *IEEE Signal Processing Magazine*, 37(4):50–61, 2020.
- [33] Zhijun Li, Xiang Li, Qinjian Li, Hang Su, Zhen Kan, and Wei He. Human-in-the-loop control of soft exosuits using impedance learning on different terrains. *IEEE Transactions on Robotics*, 38(5):2979–2993, 2022.
- [34] Libpointmatcher. *Voxelization Process Documentation*, 2024. <https://libpointmatcher.readthedocs.io/en/stable/DataPointsFilterDev/>.
- [35] Peter Lindstrom and Greg Turk. Fast and memory efficient polygonal simplification. In *Proceedings Visualization’98 (Cat. No. 98CB36276)*, pages 279–286. IEEE, 1998.
- [36] Maptelligent. The complete guide to lidar indoor mapping, 2023.
- [37] MathWorks. Working with rosbag logfiles, 2024.
- [38] Haruka Matsumura and Chinthaka Premachandra. Deep-learning-based stair detection using 3d point cloud data for preventing walking accidents of the visually impaired. *IEEE Access*, 10:56249–56255, 2022.
- [39] Mohamed Dhiaeddine Messaoudi, Bob-Antoine J Menelas, and Hamid Mcheick. Review of navigation assistive tools and technologies for the visually impaired. *Sensors*, 22(20):7888, 2022.
- [40] Jason Moore, Sam Stuart, Peter McMeekin, Richard Walker, Mina Nouredanesh, James Tung, Richard Reilly, and Alan Godfrey. Toward enhanced free-living fall risk assessment: data mining and deep learning for environment and terrain classification. *Intelligence-Based Medicine*, 8:100103, 2023.
- [41] National Oceanic and Atmospheric Administration. What is lidar?, 2024. Accessed: 2024-08-29.
- [42] NVIDIA. Nvidia jetson nano developer kit. <https://developer.nvidia.com/embedded/jetson-nano-developer-kit>, 2024.
- [43] Government of Canada. A radar-based terrain mapping approach for stair detection towards enhanced prosthetic foot control, 2023.

- [44] Julia Offermann-van Heek, Katrin Arning, and Martina Ziefle. All eyes on you! impact of location, camera type, and privacy-security-trade-off on the acceptance of surveillance technologies. In *Smart Cities, Green Technologies, and Intelligent Transport Systems: 6th International Conference, SMARTGREENS 2017, and Third International Conference, VEHITS 2017, Porto, Portugal, April 22-24, 2017, Revised Selected Papers 6*, pages 131–149. Springer, 2019.
- [45] Gabriel Iluebe Okolo, Turke Althobaiti, and Naeem Ramzan. Assistive systems for visually impaired persons: challenges and opportunities for navigation assistance. *Sensors*, 24(11):3572, 2024.
- [46] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Mathieu Brucher, Matthieu Perrot, and Edouard Duchesnay. Scikit-learn: Machine learning in python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
- [47] Point Cloud Library (PCL). *PCL Documentation: PassThrough Filter*, 2024.
- [48] Point Cloud Library (PCL). Voxel grid filtering. https://pointclouds.org/documentation/tutorials/voxel_grid.html, 2024. Available: https://pointclouds.org/documentation/tutorials/voxel_grid.html.
- [49] Rerun. Blog: Rosbag, 2024.
- [50] Karla Miriam Reyes Leiva, Milagros Jaén-Vargas, Benito Codina, and José Javier Serano Olmedo. Inertial measurement unit sensors in assistive technologies for visually impaired people, a review. *Sensors*, 21(14):4767, 2021.
- [51] RoboSense. Rs-lidar-m1 datasheet, 2024. Accessed: 2024-09-02.
- [52] RoboSense. Rs-lidar-m1 v2. <https://www.robosense.ai/en/RS-LiDAR-M1>, 2024. Available: <https://www.robosense.ai/en/RS-LiDAR-M1>.
- [53] Radu Bogdan Rusu and Steve Cousins. 3d is here: Point cloud library (pcl). *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.
- [54] Radu Bogdan Rusu, Zoltan Csaba Marton, Nico Blodow, Marius Dolha, and Michael Beetz. Towards 3d point cloud based object maps for household environments. *Robotics and Autonomous Systems*, 56(11):927–941, 2008.

- [55] Bernhard Scholkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2002.
- [56] Brent Schwarz. Mapping the world in 3d. *Nature Photonics*, 4(7):429–430, 2010.
- [57] Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, 2001.
- [58] Jie Shan and Charles K Toth. *Topographic laser ranging and scanning: principles and processing*. CRC press, 2018.
- [59] Anne Shumway-Cook, Aftab E Patla, Anita Stewart, Luigi Ferrucci, Marcia A Ciol, and Jack M Guralnik. Environmental demands associated with community mobility in older adults with and without mobility disabilities. *Physical therapy*, 82(7):670–681, 2002.
- [60] Ruxandra Tapu, Bogdan Mocanu, and Titus Zaharia. Wearable assistive devices for visually impaired: A state of the art survey. *Pattern Recognition Letters*, 137:37–52, 2020.
- [61] Adam K Taras, Niko Sünderhauf, Peter Corke, and Donald G Dansereau. Inherently privacy-preserving vision for trustworthy autonomous systems: Needs and solutions. *Journal of Responsible Technology*, 17:100079, 2024.
- [62] TechInsights. *Comparison of High-Accuracy vs Low-Accuracy LiDAR Sensors*, 2021.
- [63] Ternaris. Rosbags documentation, 2024.
- [64] Nam Van Dinh, Young-guk Ha, and Gon-Woo Kim. A universal control system for self-driving car towards urban challenges. In *2020 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pages 452–454. IEEE, 2020.
- [65] Nicolas Vandapel, Daniel F Huber, Anuj Kapuria, and Martial Hebert. Natural terrain classification using 3-d ladar data. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, volume 5, pages 5117–5122. IEEE, 2004.
- [66] Vladimir N. Vapnik. *An Overview of Statistical Learning Theory*. John Wiley & Sons, New York, 1998.
- [67] ROS Wiki. Rosbag, 2024.

- [68] Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.
- [69] World Health Organization. Assistive technology, 2024. Accessed: 2024-08-29.
- [70] Yumeng Yao, Dongqing Shao, Marco Tarabini, Seyed Alireza Moezi, Kun Li, and Paola Saccomandi. Advancements in sensor technologies and control strategies for lower-limb rehabilitation exoskeletons: A comprehensive review. *Micromachines*, 15(4):489, 2024.
- [71] Xiaoming Zhao, Weihai Chen, Xing Yan, Jianhua Wang, and Xingming Wu. Real-time stairs geometric parameters estimation for lower limb rehabilitation exoskeleton. In *2018 Chinese control and decision conference (CCDC)*, pages 5018–5023. IEEE, 2018.
- [72] Baoding Zhou, Doudou Xie, Shoubin Chen, Haoquan Mo, Chunyu Li, and Qingquan Li. Comparative analysis of slam algorithms for mechanical lidar and solid-state lidar. *IEEE Sensors Journal*, 23(5):5325–5338, 2023.
- [73] Yancong Zhu, Haojie Li, Shaojun Lyu, Xinying Shan, Yih-Kuen Jan, and Fengling Ma. Stair-climbing wheelchair proven to maintain user’s body stability based on anybody musculoskeletal model and finite element analysis. *PLoS one*, 18(1):e0279478, 2023.