Mobile Robot Navigation in an Outdoor Environment

By

V. SATISH KUMAR REDDY

*ENGG*01201004003

BHABHA ATOMIC RESEARCH CENTRE, MUMBAI

A thesis submitted to the

Board of Studies in Engineering Sciences

In partial fulfillment of requirements

For the Degree of

DOCTOR OF PHILOSOPHY

of

HOMI BHABHA NATIONAL INSTITUTE

JULY, 2017

Homi Bhabha National Institute[1]

Recommendations of the Viva Voce Committee

As members of the Viva Voce Committee, we certify that we have read the dissertation prepared by V. Satish Kumar Reddy entitled Mobile Robot Navigation in an Outdoor Environment and recommend that it may be accepted as fulfilling the thesis requirement for the award of Degree of Doctor of Philosophy.

Chairman – Dr. A. P. Tiwari		Date: 19.7.17
Guide / Convener – Dr. Prabir K. Pal		Date: 19.7.17
Examiner – Dr. Sumantra Dutta Roy	Sumantra Dutta Roy	Date: 19/07/2017.
Member 1- Dr. Anup Bhattacharjee		Date: 19/07/2017
Member 2- Dr. D. C. Kar		Date: 19.07.2017
Member 3- Dr. V. H. Patankar	V. H. Patankar	Date: 19.07.17

Final approval and acceptance of this thesis is contingent upon the candidate's submission of the final copies of the thesis to HBNI.

I/We hereby certify that I/we have read this thesis prepared under my/our direction and recommend that it may be accepted as fulfilling the thesis requirement.

Date: 19.7.17

Place: Mumbai

Co-guide (if applicable) **Guide**

[1] This page is to be included only for final submission after successful completion of viva voce.

STATEMENT BY AUTHOR

This dissertation has been submitted in partial fulfillment of requirements for an advanced degree at Homi Bhabha National Institute (HBNI) and is deposited in the Library to be made available to borrowers under rules of the HBNI.

Brief quotations from this dissertation are allowable without special permission, provided that accurate acknowledgement of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the Competent Authority of HBNI when in his or her judgement the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

VSK Reddy

V. Satish Kumar Reddy

DECLARATION

I, hereby declare that the investigation presented in the thesis has been carried out by me. The work is original and has not been submitted earlier as a whole or in part for a degree / diploma at this or any other Institution / University.

V. Satish Kumar Reddy

List of Publications based on this thesis

Journal Publications

1. Satish Kumar Reddy and Prabir K. Pal, "Segmentation of Ordered Point Cloud using a Novel Measure of Terrain Unevenness", Sensor Review, Emerald Publishing, Volume 37, Issue 1, Pages 88-100, 2017. DOI: dx.doi.org/10.1108/SR-04-2016-0078.

2. Satish Kumar Reddy and Prabir K. Pal, "Computing an unevenness field from 3D laser range data to obtain traversable region around a mobile robot", Robotics and Autonomous Systems, Elsevier, Volume 84, October 2016, Pages 48-63, ISSN 0921-8890, DOI: dx.doi.org/10.1016/j.robot.2016.07.002.

3. Satish Kumar Reddy and Prabir K. Pal, "Detection of traversable region around a mobile robot by computing terrain unevenness from the range data of a 3D laser scanner", International Journal of Intelligent Unmanned Systems, Emerald Publishing, Volume 4, Issue 2, Pages 107-128, 2016. DOI: http://dx.doi.org/10.1108/IJIUS-08-2015-0009.

Conference Publications

1. Satish K. Reddy and Prabir K. Pal, "Segmentation of point cloud from a 3D LIDAR using range difference between neighbouring beams," ACM International Conference on Advances In Robotics, AIR '15, Goa, India, Article 55, 6 pages, 2015. DOI: dx.doi.org/10.1145/2783449.2783504.

V. Satish Kumar Reddy

ACKNOWLEDGEMENTS

First, I would like to express my sincere gratitude towards my advisor Dr. Prabir K. Pal for providing a very interesting and relevant research topic. His foresight, knowledge, encouragement and patient guidance along with equanimity are responsible towards the completion of this thesis. I sincerely thank my other advisor Dr. S. V. G. Ravindranath for his time to time encouragement and advice in keeping me motivated.

I am very thankful to the Doctoral Committee chairman Dr. A. P. Tiwari and members Dr. Anup K. Bhatacherjee, Dr. D.C. Kar and Dr. V.H. Patankar for their perseverance in providing valuable suggestions and feedback while keeping a regular track on my progress. I would like to thank Dr. Kallol Roy for his Bayesian Estimation class which helped in understanding standard solutions towards robot navigation.

I would like to sincerely thank every person at the Division of Remote Handling and Robotics (DRHR) for their support and interaction during my long stay. Division Heads, Shri. Manjit Singh and Dr. D.N. Badodkar were kind in providing the required facilities and environment for carrying out research. Sarngadharan sir helped with knowledge on mobile robots and equipment for carrying out research. Dr. T.A. Dwarakanath helped with regular suggestions and advice on research. Dr. Gaurav Bhutani helped on almost everything starting from a quick initial introduction to OpenGL. I thank Jagadish Kota for his brotherly support and for eager participation in the outdoor experiments sometimes in extreme weather. Abhishek Kaushik provided good company as a fellow research student. Hemanta Swain was always positive on my research and helped with encouraging words and company. Namitha, Shishir and Jaju were helpful with quick knowledge and implementation. Bamne, Patil and Jacob from CDM were innovative in making the necessary mounts and fixtures on the robot.

I would like to thank Venkata P P K from the Computer Division for his continuous help and guidance on programming and graphics. His encouragement, motivation and the ability to see things differently made things easier. Preetham P. Shete's quick introduction on organizing large programs was very helpful. Several interactions with them over the lunch were very enjoyable.

I am indebted to BARC and HBNI for providing opportunities for students to do research on real time projects at their facilities. I thank IADD for allowing their electric vehicle for collection of data for some experiments.

I was fortunate to have several fellow research students as friends who kept through the good and difficult times. I would in particular acknowledge friends in Anushakti nagar and Kharghar for their constant company, support and enjoyment. Vidya Sagar provided with useful tips in Latex during thesis preparation. I thank several individuals and families who made my stay in Mumbai pleasurable; Lakshmi aunty and family for their initial support and care, N V Rao sir for his constant elderly support and Nithin P. Parkar sir for providing his nice house for a majority part of my stay in Mumbai.

Although mentioned at the last, Family is always a great source of strength and encouragement. I would like to express my heartful thanks to my parents and brother for their constant love, support and belief in me. During my PhD, I was fortunate to add two lovely persons into my family. While they deserved some more time and attention for a wife and a small daughter, they have always made me feel happy.

Contents

Abstract

Mobile robots are continuously evolving to perform operations with reduced levels of human intervention, allowing them to operate in remote or hazardous locations almost autonomously. There is also huge interest in development of autonomous vehicles for transporting humans to their destinations with increased comfort and safety. Reliable perception of environment surrounding a robot is a key prerequisite for its safe navigation. However, complex, unstructured and diverse outdoor environments make perception task far from trivial compared to the structured indoor environments. Sensors and algorithms are required to perceive and process data in three dimensions. Rotating Laser Scanners are commonly used sensors for obtaining range measurements to the surrounding objects with good spatial resolution. This huge data is needed to be processed quickly for extracting meaningful information about the surroundings.

This work exploits the structured nature of range data obtained from commonly used multi-beam laser scanners to develop algorithms for robot perception. A novel measure called unevenness is introduced that essentially captures the nature of surrounding terrain. It is computed at each point by comparing its range with those of its neighbours. It succinctly captures small undulations or discontinuities in the surface. Analytical expressions are derived for unevenness to study its sensitivity with range, and to decide on a scheme for setting thresholds for detecting obstacles at all ranges. Traversable region is obtained by connecting all the points with unevenness less than a threshold. Detection of subtle discontinuities using unevenness enables segmentation of discernible objects and features in robot's environment. Unevenness further contributes in detection of key points within a scan. Using only the key points instead of the entire point set not only accelerates registration but improves accuracy.

Unevenness contributes to different navigation tasks by processing data from a single laser scan. It provides a clear boundary between object and ground and between two objects leading to their easy identification. Unevenness detects traversable slopes while being unaffected by the roll and pitch of sensor caused by robot's motion. Traversable region detection and segmentation are performed well within real time. Registration using select key points using unevenness brings down registration times to nearly a tenth compared to registration using the entire point cloud. Experimental results are presented to demonstrate the efficacy of unevenness in all aspects of autonomous navigation.

List of Figures

List of Tables

Acronyms

e.g.	for example (latin: exempli gratia)
et al.	and others (latin: et alii)
i.e.	that is (latin: id est)
viz.	namely (latin: videlicet)
1D/2D/3D	1/2/3- dimensional
DARPA	Defense Advanced Research Projects Agency
DOF	degrees of freedom
FOV	Field of View
GICP	Generalized Iterative Closest Point
GP	Gaussian Process
GP-INSAC	Gaussian Process Incremental Sampling and Consensus
GPS	Global Positioning System
HMM	Hidden Markov Model
HSV	Hue Saturation Value
ICP	Iterative Closest Point

IMU	Inertial Measurement Unit
KNN	K Nearest Neighbour
Laser	Light amplification by stimulated emission of radiation
Lidar	Light Detection and Ranging
MRF	Markov Random Field
MRPT	Mobile Robot Programming Toolkit
NDT	Normal Distribution Transform
NDT-OM	Normal Distribution Transform Occupancy Map
PCA	Principal Component Analysis
PCL	Point Cloud Library
Radar	Radio Detection and Ranging
RANSAC	Random Sample Consensus
RBNN	Radially Bounded Nearest Neighbour
SEIF	Sparse Extended Information Filters
SLAM	Simultaneous Localization and Mapping
SRG-NDT	Segmented Region Growing Normal Distribution Transform
SVD	Singular Value Decomposition
SV-NDT	Super Voxel-Normal Distribution Transform
TOF	Time of Flight
UDP	User Datagram Protocol
UPD	Unevenness Point Descriptor

Nomenclature

a	minimum obstacle height constant
b	inter ring obstacle height
c_d	centroid of data scan
c_m	centroid of model scan
$C_{row,col}$	grid cell with indices row and col
d	horizontal distance between successive laser beams
e_1, e_2	limiting thresholds for obstacle growth
ϵ	difference in error between successive iterations
f	obstacle height proportionality constant
g	transverse obstacle height
H	sensor height above ground
$\mathbf{H}$	covariance matrix
i, j, m, n, u, v	indices
I	intensity
max	maximum value
p	point
P	point cloud

R	range
$\mathbf{R}$	Rotation matrix
$\mathbb{R}$	Real number space
S	Set of scan points
S_i	i^{th} segment
$\mathbf{t}$	translation vector
th	threshold
$\mathbf{N}$	number of points in a scan
w	weight vector
x, y, z	cartesian coordinates
XYZ	cartesian coordinate system
α	slope in radial direction
β	radial slope threshold
γ	height difference threshold
Γ	transverse unevenness
δ	change
ξ	key point selection threshold
$\sum$	summation
Ω	radial unevenness
θ	yaw or rotation angle
ϕ	pitch or elevation angle
ψ	roll angle

Chapter 1

Introduction

Mobile robotics is an important area of application of robots, where robots move to different locations to perform a set of assigned tasks. Developing autonomous capabilities in these robots enable them to operate with minimum or no human intervention, sometimes in locations which are hazardous or inaccessible to humans. Autonomous mobile robots are finding applications in industries, mining, rescue and recovery, and planetary explorations. In addition, over the past few years, there is a surge in research and development of driverless cars both in industry and academia. These are aimed to transport occupants safely to their destinations autonomously. Driverless cars relieve humans from the task of continuously driving their vehicles, allowing them to either relax or perform other important tasks. An automated systematic control of the vehicle is expected to enhance the safety and comfort levels of its occupants. It is also likely to improve traffic management on busy roads, thus preventing accidents. For autonomous operations, a robot needs to build a reliable model of its surrounding environment. Once the robot perceives its surroundings, it can plan a safe trajectory towards the goal.

Perception is the task of acquiring knowledge about the robot's surroundings. This is done by taking measurements from various sensors and then extracting meaningful information from them [1]. Perception leads to both understanding the nature of robot's surroundings and locating the robot with respect to its surrounding objects. As such, locating the robot in its environment is the most fundamental problem to provide a mobile robot with autonomous capabilities[2]. Further, the uncertainties in detecting position and identity of objects in the environment increase in the presence of unstructured objects such as trees, bushes, pedestrians etc. in a typical outdoor environment. The present research is aimed at developing algorithms for improving the three-dimensional (3D) perception of a wheeled mobile robot in outdoor environments using range data from Multi-beam rotating laser scanners, where no prior map is available to the robot. The range data from a 3D laser scanner is a set of points sampling the surfaces surrounding the robot. The systematic way in which range data is acquired by the laser scanner is exploited by our algorithms to build robot's perception.

Although Global Positioning System (GPS) appears to directly provide robot's location in outdoor environments, GPS alone is not adequate [3]. GPS has localization errors up to few meters, which could turn crucial in some situations. Multi path errors and requirement of clear sky view make GPS less dependable. This makes it essential for a robot to establish its location using the robustly detected features within the environment. This is the primary step required by a robot to complete its navigational tasks.

Mobile robot navigation in an outdoor environment poses increased difficulty compared to that in an indoor environment because of the environmental complexity. Features present in indoor environments are regular, environment is structured, and the robot moves in a plane, which usually makes it sufficient to perceive and represent the environment in two dimensions (2D). Traversable regions of an indoor environment are usually a plane at a single level. Features like walls, doors and other obstacles are represented in this plane as lines and arcs. However, the environments encountered by outdoor mobile robots are rarely structured and features not regular in shape. Also, the robot's orientation keeps changing in three dimensions (3D) as it moves on an uneven terrain, so it keeps seeing different sections of the surroundings with a 2D scanner. This makes a simple 2D perception inadequate for representing outdoor environments. It is also difficult to model the variety of objects or features that are present in outdoor environments. The unstructured nature of robot's surroundings, uneven terrains and the ability or requirement of robots to traverse such terrains in six Degrees of Freedom (6DOF) make it essential for outdoor mobile robots to perceive environments in 3D. Another characteristic of outdoor navigation is the need for representing large areas in which the robot moves and the scalability of the maps as the area of operation increases.

The contribution of this work lies in exploiting the structured nature of range data provided by rotating Multi-beam laser scanners when sampling the robot's surroundings. This is accomplished by introducing a novel measure of unevenness that is computed at each range point using the neighbouring range data from the laser scanner. A simple mathematical model is derived and analyzed to project unevenness as a point attribute for developing robot perception. Unevenness is obtained by comparing the expected and measured range differences between neighbouring range points in a laser scan. Unevenness largely overcomes the shortcomings due to the sparse sampling of surroundings along the elevation angle. Unevenness, introduced in this work, captures surface discontinuities between two neighbouring range points very effectively in spite of data sparsity and even when these two points are from different surfaces but close to one another. Unevenness also allows us (using a trick) to overcome the effect of small roll and pitch experienced by the robot in motion on an uneven terrain. It also works well in environments containing slopes, making it suitable for wheeled mobile robot navigation. Since the methods are processed at a point level, there is a clear distinction between an object and ground and between two objects.

This thesis contributes to three important stages of robot navigation. Unevenness is first used to obtain traversable regions around a robot after detecting obstacles. This helps the robot to traverse its environment by avoiding obstacles. Points belonging to the non-traversable regions are then segmented to bring out important discernible objects in the robot's environment. Unevenness is further utilized in detection of key points within a scan which are used for registration of scans that are taken at different locations from a moving robot. This improves the quality and speed of scan registration leading to more accurate robot localization.

1.1 Background

Progress of autonomous capabilities in robots can be traced back to Autonomous Guided Vehicles (AGVs) that moved or glided over metal wires using magnetic induction [4]. Wires are laid under floor following the desired trajectories. Change in trajectories would require re-laying of the wires underground. AGVs that could track painted lines or stripes provided some flexibility, where it is possible to repaint the trajectories over the floor [5]. However, this required maintaining the floor and paint surfaces in a good shape for robust navigation. Further flexibility in navigation is attained by using artificial markers called beacons placed at different locations around the area of robot operation [4]. These markers are continuously observed by the moving robots from different locations. By locating reflective markers at different locations around an operating hall, an AGV with laser range finders can localize itself in the operating area. This approach allows flexibility in altering the trajectories on-the-fly using software. All these approaches are suitable only for industrial applications and within a limited area of operation because additional infrastructure (wires, lines, beacons) is required for implementing robot navigation.

It must also be noted that the robot trajectories must be obstacle free for safe navigation.

Unlike in industrial applications, a dedicated navigation infrastructure is generally not available for myriad of applications where autonomous mobile robots can be potentially employed. All these environments are too complex to be modelled. Driverless cars should understand their environments with dynamic traffic participants for safe navigation. The DARPA Grand challenge 2005 in off-road desert-like environment, and the Urban Challenge in 2007 in a simulated urban environment provided impetus for development of driverless cars. Figure 1.1 shows the competition winning cars Stanley [6] and Boss [7] respectively. Interest is still growing with several major corporations like Google and almost all the technology and automobile companies getting involved in development of driverless cars. However, several challenges both technical and legal are to be overcome before they are employed safely on public roads.

Figure 1.1. DARPA Competition Winners, Stanley (left) at 2005 Grand Challenge and Boss (Right) at 2007 Urban Challenge.

Modelling the entire operating area by taking true measurements prior to navigation is far from practical. In addition, autonomous robots may have to operate in completely new locations or in areas that are constantly changing. For this reason, it is important for robots to develop perception of environment at the time of navigation. From the perceived environment, a robot can plan its navigation by using the detected features. It has to primarily find a traversable region, which is free of obstacles, for planning its motion. Environment surrounding a robot has to be built in the form of a map which is a compact representation that a robot can understand. Once a map is available, the robot can localize itself in its environment based on the sensor measurements and carry out navigation. Maps can be classified as either metric or topological. Metric maps have geometrical features present in the map with information on the distances between features. Topological maps [8] are higher level representations containing cognizable features forming graph nodes with edges between the nodes representing connectivity between nodes. Another representation is hybrid maps, which utilize both metric for local representation and topological for higher or global level representation [9, 10].

Robot navigation can be local and reactive, wherein a robot traverses a smooth traversable path by avoiding obstacles. However, this does not guarantee that the robot will reach its goal. It may lead to local minimum or to oscillations between two locations. Dynamic window approach [11] incorporates the dynamics of the robots along with overall goal to reduce the search space to admissible velocities. Similarly, a tentacle based navigation scheme [12] evaluates each of a set of pre-calculated trajectories, called tentacles, depending on the vehicle speed. A clear trajectory, preferably in the direction of current motion, is blended towards a global trajectory defined with GPS waypoints. Lack of reliability of GPS is a drawback for this approach.

A map can first be built by manually driving the robot in the operating environment. Using this map, the robot can then navigate autonomously [13]. Thrun [14] provides the history of robotic mapping, and surveys various probabilistic models that are used for 2D mapping. In unknown environments, given the sensor measurements, robot location should be known for constructing a map. But robot location cannot be established without having a map. This mutual dependence makes the problem difficult to solve. Popular approach for solving this problem is called Simultaneous Localization and Mapping (SLAM), first introduced by Smith and Cheeseman [15]. It concurrently maps the environment and locates the robot in it. There are numerous solutions to the SLAM problem, most of them for 2D maps, including Extended Kalman Filter [16], Particle Filter [17], Sparse Extended Information Filter(SEIF) [18], FastSLAM [19] and GraphSLAM [20]. However, most of the implementations are limited to small scale indoor environment with 3DOF (x, y, θ) where x and y are the spatial coordinates and θ is the bearing; these algorithms are not easily scaled to the 3D environment as they require complex representation of the environment and much more memory. Robot pose should be in 6DOF $(x, y, z, pitch, roll, yaw)$. However, if proper data association and correspondence between features in successive scans are achieved, SLAM can be performed in extensive outdoor environments [21]. For this, robust feature detection is essential. Nüchter et al. [22] built 6D SLAM by incrementally registering the successive 3D scans using the standard registration algorithms along with odometry, and then relaxing the constraints when either taking the known measurements or when visiting the same spot again called loop closing. Borrmann et al.[23] similarly extend the graph based 2D SLAM [20] to obtain a global representation of the environment by building a network using a distance heuristic along with odometry based extrapolation for obtaining 6DOF pose in 3D data. It also registers successive scans using Iterative Closest Point [24] algorithm.

In order to overcome the complexities of 3D outdoor environments, particularly because of the huge data requirement needing large memory, high computational requirements and also to build a convenient navigation scheme, researchers have represented outdoor environment using different types of maps. In structured environments, where robots traverse on plane surfaces, environments are sufficiently well represented in 2D with obstacles and landmarks represented by simple features like lines. This leads to 3D perception of environment by using simple 2D maps for navigation [25]. Here, the complexity of map representation is reduced. Occupancy grids [26, 27] commonly represent robot environment in 2D with equally spaced grid cells classified as either obstacle or free based on the information of points present in individual cells. All the data from different sensors or the data from the same sensor taken at different robot locations can be fused onto the grid-cells [26]. However, 2D representation leads to erroneous assumptions in non-flat terrains, making it essential to represent the environment in complete 3D. Occupancy grids are extended to 3D environment where 3D space is discretized into smaller cubic cells called voxels [28] to represent free and occupied spaces. A 2.5D representation is commonly used as a trade off between completeness of 3D and simplicity of 2D representation, where each grid cell contains some information about the column above it. For example, elevation maps [29] are represented with values of highest point in each of the grid cells. This representation is used by most teams participating in DARPA Urban Challenge [30]. In addition to the regular rectangular grids, and in order to deal with data sparsity, some methods use polar grids [31] or circular grids [32], where size of cells increase with distance from the robot. Elevation maps however classify potentially traversable regions under over-hanging objects as obstacles, which is not desirable. This may, for example, prevent a robot from seeing the regions under trees or bridges as traversable. Map registration using data from different locations is also difficult using elevation maps. Overhanging obstacles can be dealt with by employing a safe height [31, 33] or by using extended elevation maps [34] where points above a free space are safely discarded. Multi-level surface maps [35] discretize points vertically projected on to a cell into discrete height levels thus allowing robots to traverse on multiple surfaces like over and under the same bridge. In addition to these, compact representations include Octomaps [36], where the regions are represented using octrees [37]. The advantage here is that the maps are compact and not limited to a fixed area. They can easily be extended as the robot moves. Recently Saarinen et al.[38] employed 3D occupancy grids using normal distributions in each cell, calling the scheme as 3D Normal Distribution Transform

Occupancy Map (3D NDT-OM), which combines the compactness of the Normal distribution transform [39] with the robustness of Occupancy grids.

In the approaches described using occupancy grids, a set of points within a grid cell are grouped together and collectively classified. This results in a lack of clear boundary between different objects or between an obstacle and ground. But, for an outdoor mobile robot, it is desirable for the features to have clear boundaries. This helps the robot to accurately segment and classify features containing only the points belonging to them. This is essential for robust data association in a scan and for feature correspondence between the scans. Objects are also needed to be tracked to understand the behaviour of traffic participants. For bringing clear boundaries between objects, one has to use attributes at a point level. But given that a point in isolation cannot give much meaningful information, points are needed to be equipped with local description about the surroundings. Point level attributes in robot literature include Intensity [40], Normals [41], Difference of Normals [42], Local Convexity [43], Persistent Feature Histograms [44] and Unevenness Point Descriptor (UPD) [45]. Of these, Intensity is not linear, thus making it difficult to use. Intensity values returned to the sensor also depend on the angle at which the laser strikes the surface. While Normals, Difference of Normals, Local convexity and UPD take into account both terrain irregularity and inclination, they depend on neighbouring points which results in a smooth transition between ground and objects.

Given this background, a robot should develop a good perception in order to solve the navigation problem. There is an incentive in developing robust schemes that detect clear feature boundaries between ground and obstacles and also between two objects. Unevenness, proposed in our thesis, does this by developing perception at a point level without using grid based maps. Unevenness uses the range data from a point's immediate neighbours, and information that is available in the structured scan data. Unevenness as an attribute has a potential to contribute to different stages in robot navigation as detailed in the subsequent chapters along with the state-of-art techniques. Unevenness develops reasonably good perception in spite of the data being sparse in the radial direction.

1.2 Research Methodology

Following are the underlying principles and assumptions implicit in the present research work for outdoor mobile robot navigation.

- Robot perception is developed by using data from a multi-beam laser scanner that quickly scans the robot surroundings and provides structured range data leading to an organized point cloud with neighbourhood information.
- Experiments are carried out mainly on real data collected using our outdoor mobile robot, and not on simulated data. A Velodyne HDL-32 scanner is placed at a height of $1.3m$ above ground level to collect data at various locations in our campus. Thresholds are validated using obstacle boxes of different heights by placing them on a near flat surface of our lab terrace. Data is collected by placing the scanner at the same $1.3m$ height using a stationary stand. In addition, a Sick LMS291 laser scanner is mounted on the robot at $0.6m$ height and rotated in the yaw direction in a stop-scan-go fashion, and its range data collected and processed. The purpose is to show that the proposed methods are not specific to a single type of scanner.
- From the range data of a single $360°$ scan, we compute an unevenness field around the robot. A threshold is applied to detect obstacles that need to be avoided. A path for the robot can be planned within the traversable region. Once the traversable points are removed, only obstacles remain. We again use unevenness to segment these obstacles into distinct features. At a third stage,

some of these features are selectively used for scan matching using Iterative Closest Point (ICP) algorithm for localization of the robot.

- Robot is considered to be a point object. The thresholds for detection of obstacles can be set according to the capabilities of the operating robot. For example, robot width along with the wheel base and ground clearance determines the thresholds for obstacle detection and in planning a safe traversable path.

- The nature of sampling by the laser scanner is such that data gets sparser with distance resulting in smaller far away objects not being sampled by any of the laser beams. However, these objects are eventually detected when the robot moves close to them. Given the quick rate of data acquisition and slow robot speed in our case, this poses no threat to the robot.

- Since the outdoor environments are complex and contain a variety of objects, we focus on finding sufficient number of robust features required for safe navigation instead of obtaining perfect perception.

- Emphasis is placed on providing quantitative results for individual features instead of that for the whole scan, given the difficulty of evaluating the large number of points in a scan. However qualitative evaluations are included for entire scans.

- We tried to develop algorithms that can be executed within the time interval between two successive scans for the scanner rotating at 10Hz. This we consider as real time.

- The methodologies try to clearly identify the objects and features at a point level instead of coarsely identifying the regions in the environment.

- Since our robot moves at a slow speed ($0.2m/s$), and given the high rate at which the scanner provides range data, motion corrections and sensor tilt (pitch and roll) compensations are not done prior to application of our algorithms. This does not adversely affect detection as the robot movement is very minute during the interval between reading the ranges of neighbouring points. Once the perception is developed, the exact location of the points can be corrected using data from the inertial sensors.

- Single 360° scans are processed individually as and when they are acquired. Obstacle detection, traversability, segmentation and also the detection of key points are done on individual scans.

- Results, where appropriate, are shown with individual features instead of the entire scan for highlighting our contributions.

1.3 Thesis Overview

This thesis presents methods that improve 3D perception of a mobile robot in an outdoor environment. 3D perception around a given robot position in an unknown environment is developed by using a simple unevenness attribute obtained by using only the range data from a rotating Multi-beam laser scanner. Unevenness with an ability to capture subtle surface discontinuities is shown to clearly bring out features at a point level and contribute to different stages of robot navigation. This level of perception is obtained in spite of data being sparse in the radial direction and in complex outdoor environments. Algorithms developed assume no prior knowledge of the environment; they are model free, and the proposed methods require no prior training or machine learning. Chapters following the introduction with the general background information on robot navigation are organized with each chapter describing a single navigation stage along with the state of the art.

This chapter provides an overview of robot navigation problem and the need for reliable perception for mobile robots in order to navigate in complex outdoor environments.

In chapter two, information on the types of sensors used with mobile robots is presented. Laser scanning systems used for obtaining 3D laser scans, particularly the Velodyne HDL-32 laser scanner, primarily used in our experiments, is described in detail. The systematic nature in which the sensor scans the environment leading to an organized range data is explained.

In chapter three, the novel point level attribute, which we call unevenness, is introduced. This is central to our work. Unevenness is characterized for different obstacle heights and slopes across the ranges. Unevenness is shown to detect traversable regions around the robot and it clearly brings out even smallest of obstacle features from ground. It detects both positive and negative obstacles. Obstacles are detected in spite of presence of slopes, and even when the robot experiences pitch and roll. Immediate traversable region around the current robot location is obtained within a short period, making this approach real time.

In chapter four, segmentation of discernible objects present in robot's environment is presented. Ability of unevenness in bringing out surface discontinuities in the individual objects is demonstrated. Detection of finer edges in the transverse direction in addition to regular radial edges is shown to separate out even objects which are close by. A point level region growing algorithm is used to grow segments. Growth is restricted based on the unevenness values of individual points and detection of edges using unevenness. Even though unevenness exploits the structured nature of a scan, in theory it can only segment out rigid objects. It is shown that a reasonable set of discernible features that are needed for robot navigation are properly segmented, including moving objects like vehicles, pedestrians and even bicycles. Segmentation is performed in real time.

In chapter five, unevenness is used for improving quality and speed of scan registration between successive scans obtained from a moving robot. Registration aggregates point clouds obtained from different locations leading to robot localization. Robot pose is obtained as a result of convergence to a correct rigid transformation between successive scans. Unevenness detects key points within the individual scans leading to good correspondence and accurate registration. It is shown that registration with selected number of points not only improves registration speed but also results in more accurate registration.

In chapter six, we conclude the thesis by looking at the advantages and shortcomings that unevenness brings in improving robot perception. We also suggest ways in which the concept of unevenness can be further extended and applied to other areas of robot perception and navigation.

Chapter 2

Sensors on Robot

For a mobile robot to navigate autonomously, it is not only necessary to sense its surroundings, but it is also important to establish the state of the robot, i.e., its position and orientation with respect to the objects around. For this, a mobile robot is equipped with several sensors that provide information both on the surrounding environments and the robot's internal state when negotiating a terrain. Sensors can be primarily classified into Perception, Motion or Localization sensors. Perception sensors like cameras, stereos and laser scanners help robots in understanding the nature of their surroundings prior to planning motion towards goal. Motion sensors help a robot to traverse a terrain following a selected trajectory. Odometric encoders and rotation sensors present on robot's wheels provide information on the extent of robot movement. Magnetic compass provides direction information with respect to earth's magnetic axis. Vibration sensors help determine surface roughness when the robot is actually moving. Outdoor robots are also usually equipped with a Global Positioning System (GPS) receiver for obtaining location (global) information. Inertial Measurement Units (IMU) are commonly used along with GPS for obtaining robot pose estimate. Measuring inertial changes along different directional and rotational axes, IMUs provide information on the accelerations and rotations experienced by robots. In addition, safety sensors like Radar (Range detection and ranging) and 2D Laser bumper scanners prevent collisions by sensing nearby objects. Battery level indicators sense robot's health. Data from several sensors are fused and processed to obtain information on different robot parameters.

Perception sensors sense the robot's surroundings. Information extracted from these sensors is primarily used by robots to understand the surrounding environment. Robot can then identify static and dynamic objects in its surroundings for planning a safe motion towards its goal. Robot's localization information, i.e. its current location and pose in the environment, is established with respect to the detected robust features. Odometric sensors like wheel and rotary encoders provide relative distances travelled by robot by measuring wheel rotations. The actual trajectory traced by the robot however is different from the trajectory computed using odometry because of slippage of wheels at their points of contact with ground. With time, this position error keeps on increasing, making it unreliable to depend purely on odometry for robot localization. In order to overcome this, it is a common practice to obtain the location information of the robot by using the detected features present in the environment. For example, visual odometry is obtained by analyzing the positions of detected features in sequential camera images. Mars exploration rovers determine the rover's pose [46] using visual odometry. A more comprehensive technique for finding robot pose is SLAM [15], where the robot incrementally builds a map of the environment with its location present in it. A map can contain anything between collections of sparse features to dense 3D point clouds. However, both visual odometry and SLAM perform well with dense data obtained from perception sensors.

Cameras are less expensive and are commonly used perception sensors which capture environments in terms of grey level intensity images. Similarly, infrared or thermal cameras [47] capture thermal variations in the environment; this is particularly suitable for detecting negative obstacles, like ditches, particularly during night [48]. Normal cameras capture a limited view of the environment as they point in a single direction and depend on optical lenses for the Field-of-View (FOV). Omni directional cameras overcome limited FOV by capturing image around the sensor with 360° horizontal FOV. While images from different types of cameras provide continuous and intuitive representations that are suitable for human understanding, lack of depth information causes difficulties for robots in perceiving distance to different objects. Depth information helps robots to understand free spaces and occlusions. There are also no illusions like shadows on road appearing as different objects (ditches). Camera images are also affected by illumination changes depending on time and weather conditions. Stereo cameras [45, 49] provide information on depth by calculating disparities between pair of images captured by twin cameras offset by a small distance. Bernini et al.[50] survey different methods using stereo cameras for real time obstacle detection on ground vehicles. Stereo cameras are limited in FOV and range. Due to calculation of depth based on disparity of a feature between two images, they are not very effective in sensing monotonic environments like deserts or plain terrains devoid of features. Time-of-Flight (TOF) cameras provide distance information based on measured time interval between signal emanating from the sensor and detection of the reflected pulse. While TOF cameras provide quick dense measurements, they are small and have limited FOV, failing to provide reasonable environmental representation.

2.1 Laser Scanners

Laser scanners or Lidars (Light detection and ranging) are a class of sensors commonly used in robotics, which measure distance using lasers. Here the measurement device is one dimensional (1D), measuring the distance in terms of range in the pointed direction. A scanning mechanism helps in digitalizing the surroundings, where the environment is represented using a set of discrete range points. Multiple laser beams are fired in different directions returning a range R for each beam which is proportional to the time interval between a laser pulse release and return to the detector after illuminating the nearest encountered surface. This kind of measurement is called Time-of-Flight (TOF) technique. Other measurement principles include Phase-Shift measurement or Triangulation [51]. A collection of these range points along with the vertical and horizontal pointing angles ϕ and θ can be used to obtain a point cloud $P(x, y, z)^T$ in 3D space $\mathbb{R}^3$ with each point $p(x, y, z)$ and $p \in P$ representing the distance from the scanner to the sampled surface.

2.2 Two Dimensional Laser Scanners

Two-dimensional (2D) laser scanners are most commonly used by the robotics community. A 2D laser scanner contains a rotating mirror to sequentially direct laser pulses at different angles (angular resolutions typically 0.25°,0.5° or 1°) along a 2D plane to obtain range measurements in different directions. Commonly used 2D scanners (Sick LMS291, Hokuyo UTM-30LX) with outdoor robots are shown in Figure 2.1 along with a typical 2D scan in XY plane where each point $p(x, y)$ is spatially represented in two dimensions $\mathbb{R}^2$. 2D laser scans are mostly used with indoor robots operating in structured indoor environments, where representation using 2D maps is sufficient. Considering the complexity of the outdoor environments, and the ability of outdoor robots to navigate terrains using six Degrees-of-Freedom (6DOF), it is essential to perceive the environments in 3D.

Several researchers have come up with innovative mechanisms for obtaining 3D point clouds using 2D laser scanners. Multiple 2D laser scanners pointing in different directions can also be used for creating 3D perception. Thrun et al. [52]

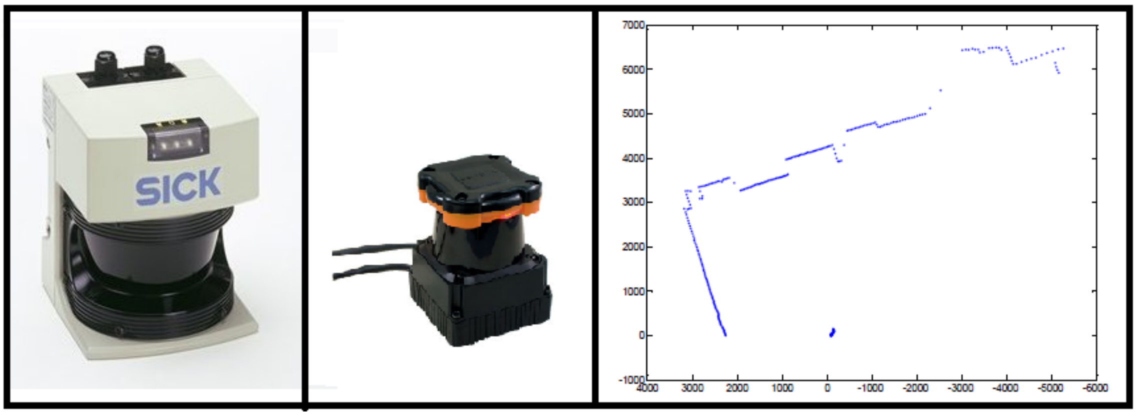


Figure 2.1. Common 2D Laser Scanners: Sick LMS291 and Hokuyo UTM-30LX (Not to scale) along with a typical 2D Plot.

place one scanner horizontally for obstacle detection and another vertically to build 3D scan as and when the robot moves. Howard et al.[53] use a similar arrangement to obtain a dense 3D map of the outdoor environment. Multiple 2D laser scanners are placed on top of Stanley, the DARPA race winning robot [6] to generate a 3D patch in front of robot. Triebel[54] generated 3D scan by placing the robot on a 4DOF manipulator. Bosse and Zlot [55] built a scanning mechanism by placing a compact Hokuyo scanner along with an Inertial Measurement Unit (IMU) on a spring based mechanism which can be either carried by a walking human or be placed on a vehicle so that with vibration the laser scanner points in different directions to generate 3D scan. Thrun et al.[56] mounted a 2D scanner on a helicopter which goes aerial for obtaining a 3D scan. By placing a rotating mirror in front of a 2D scanner [57], the laser beams from the scanners are directed along varying angles to obtain a 3D scan.

Outdoor mobile robots commonly use scanning mechanism with an independent external motor that rotates the whole 2D scanner along an axis [25, 58, 59, 60, 61]. Information of the motor rotation angle along with the range and direction of laser beam of the 2D scanner determines location of points in 3D space. Initial attempts include Surmann et al.[62] using a nodding scanner in the pitch direction. Hähnel and Burgard [63] rotate the sensor in the yaw direction. Wulf and Wagner

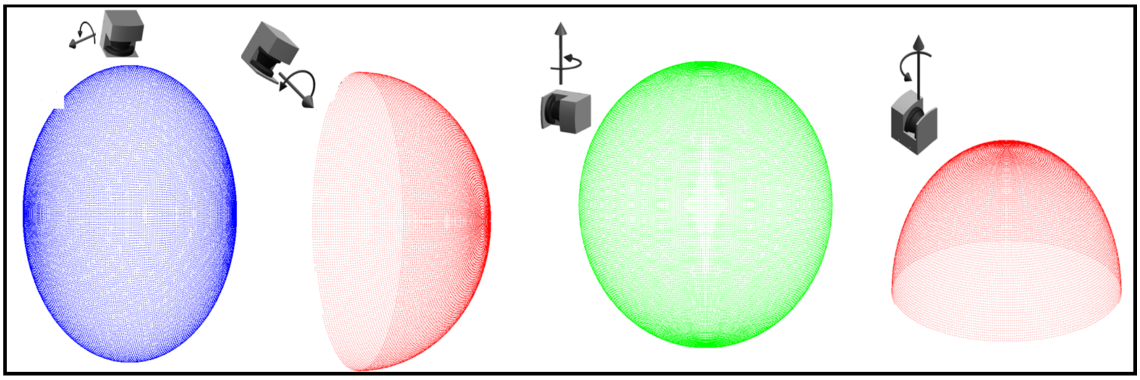


Figure 2.2. Scanning Configurations, starting from Left: Pitch Scan, Roll Scan, Yaw Scan, Upward Yaw Scan.

[64] describe ways for rotating 2D scanners using external motors along with the resulting scan densities that suit different applications. They introduce another rotation scheme called rolling scan which produces a hemispherical dense scan in front of the scanner. Similar rotation is done by pointing the laser scanner upwards called upward yaw scan. The axis of rotations and the resulting densities are replicated in Figure 2.2 by giving a constant range. It can be seen that point densities are more nearer to the axis of rotation and sparse in directions orthogonal to the rotation axis.

Out of these scanning mechanisms, Pitch scan with a nodding mechanism [61] and Yaw scan with a rotating mechanism [25] are commonly used with outdoor mobile robots; some standard products are also available on these lines. The major drawback of these approaches comes from the Stop-Scan-Go nature of scanning which requires few seconds (Typically $10s$) to complete a full 360° scan. This is not desirable with the fast moving robots or in dynamic environments. Triebel [54] place two Sick 2D scanners in back-to-back fashion facing away from each other to rotate in the yaw direction with each scanner covering a 180° view, thus reducing the scan acquisition time to half.

2.3 Three Dimensional Laser Scanners

Three dimensional (3D) laser scanners scan the environment in 3D and produce a point cloud. Earlier, 3D laser scanners (From Riegl, Leica) were mainly terrestrial laser scanners which produced very dense point clouds. These scanners found applications in surveying, architecture and archaeology. High cost, weight and large data storage requirement make them less suitable for autonomous vehicles. Multi-beam rotating laser scanners are compact and are commonly used with autonomous outdoor mobile robots and driverless cars starting with the DARPA Urban challenge in 2007, where the majority of participants used Velodyne HDL-64 laser scanner as the primary perception sensor for detecting obstacles [30]. These are the class of sensors having laser head assemblies that rotates in the yaw direction. A family of Multi-beam laser scanners along with the number of laser beams for each scanner is shown in Figure 2.3. Important specifications of the individual scanners are tabulated in Table 1. A set of laser diodes and detectors (16/32/64) are arranged such that they fire lasers simultaneously at specified pitch angles in a vertical plane. Density of sampling in the radial direction depends on the number of lasers and specifically the pitch angle resolution. The number of range returns equal to the number of different laser beams fired. The laser head assembly continuously rotates in the clockwise direction and fire lasers at regular rotation intervals (Azimuth) before completing a 360° scan. Depending on the sensor model and the user settings, 5 to 20 scans are obtained every second. This rate of rotation dictates the angular resolution in the Azimuth (θ). Range points at increasing pitch angles(ϕ)at a given azimuth angle (θ) will be referred as the radial direction, whereas those in the lateral direction at a given pitch angle (ϕ) will be referred as the transverse direction for the purpose of computing terrain unevenness.

Systematic scanning by the multi-beam laser scanners provides an ordered point cloud where one beam from a laser at a fixed pitch angle (ϕ) samples the

Figure 2.3. Velodyne Laser Scanners: HDL-64, HDL-32, VLP-16 (Number of laser beams shown in Red).

Table 2.1: Important Specifications of Velodyne Laser Scanners.

S.No	Specification	HDL-64E	HDL-32	VLP-16
1	No of Beams	64	32	16
2	Dimensions ($width \times height$)	$203mm \times 284mm$	$86mm \times 145mm$	$104mm \times 145mm$
3	Weight	15Kg	1Kg	0.83Kg
4	Range	$100 - 120m$	$80 - 100m$	$100m$
5	Accuracy	$\pm 2cm$	$\pm 2cm$	$\pm 3cm$
6	Vertical Field-of-View (Bottom to Top)	$26.8°$ $(-24.8° \ to + 2.0°)$	$41.33°$ $(-30.67° \ to + 10.67°)$	$30°$ $(-15° to + 15°)$
7	Pitch Angle Resolution	0.4	$1.33° or 1.34°$	$2°$
8	Azimuth Angle Resolution	$5Hz: 0.08°$ $10Hz: 0.17°$ $20Hz: 0.35°$	$5Hz: 0.08°$ $10Hz: 0.17°$ $20Hz: 0.35°$	$5Hz: 0.1°$ $10Hz: 0.2°$ $20Hz: 0.4°$
9	Data rate	1.3 Million pts/sec	700,000 pts/sec	300,000 pts/sec

environment very closely forming a ring like pattern on a level ground. In the radial direction however the gap between adjacent rings increases with the lasers pointing farther away. The resultant scan provides an ordered point cloud where the information of neighbouring points is implicit, and one does not have to resort to computationally expensive nearest neighbour search. Scan data contains range values along with the laser beam pointing angles, and this information has been used to advantage in this work for developing perception of the robot's surroundings.

Multi-Beam laser scanners commonly used with autonomous robots are the Velodyne HDL-64 and HDL-32 scanners containing a set of 64 and 32 laser diodes/emitters respectively. More recently, a 16 beam Velodyne VLP-16 has been developed which is light weight and less expensive, making it attractive for use in research. In addition, when a 2D laser like the Sick scanner is rotated along the yaw direction in the Stop-Scan-Go fashion, a similar ordered point cloud is obtained with the number of rings equal to the number points in a single 2D scan. Our work mainly uses a compact Velodyne HDL-32 sensor, which is described below in more detail. In addition, a Sick LMS291 2D scanner is rotated along the yaw axis and its range data collected and processed to demonstrate that the presented method works even for such a setup.

2.4 Velodyne HDL-32 Raw Data

The Velodyne HDL-32 laser has a set of 32 laser diodes arranged in a vertical plane. Lasers are fired with pitch angles ϕ between $-30.67° to + 10.67°$. The difference in pitch angles between consecutive laser beams is either $1.33° or 1.34°$ [65]. Negative angles are for lasers pointing towards ground. Positive angles point above the sensor eye level and detect large obstacles like buildings and trees at faraway distances. Velodyne sensors are interfaced through an Ethernet connection. User Datagram Protocol (UDP) packets are parsed to obtain the required data fields. In addition, Velodyne HDL-32 provides data from built-in inertial sensors and an external GPS receiver for global location information. Scanner scans its environment by continuously rotating in a horizontal plane and firing a set of 32 lasers at different azimuth angles (θ). In our case, the scanner rotates at $10Hz$ enabling the sensor to collect $360°$ scans in every $0.1 second$. In one sensor head rotation, the environment is sampled with upto 70000 points. The angular resolution along the azimuth (θ) is about $0.164°$. This results in a FOV of $41.33° \times 360°$, where each laser returns a range R. The resulting scan is a set of points $P(R, \theta, \phi)^T$ representing the environment with each point giving distance to the surface sampled by a laser. The set of points P in 3D space can be represented in Cartesian coordinates $p(x, y, z) \in P$. The x, y and z coordinates are calculated using the following equations for a scanner rotating in the clockwise direction. Rotation angle θ is the clockwise rotation angle starting from the axis left of the robot which is negative X-axis according to our coordinate system (See Figure 2.4).

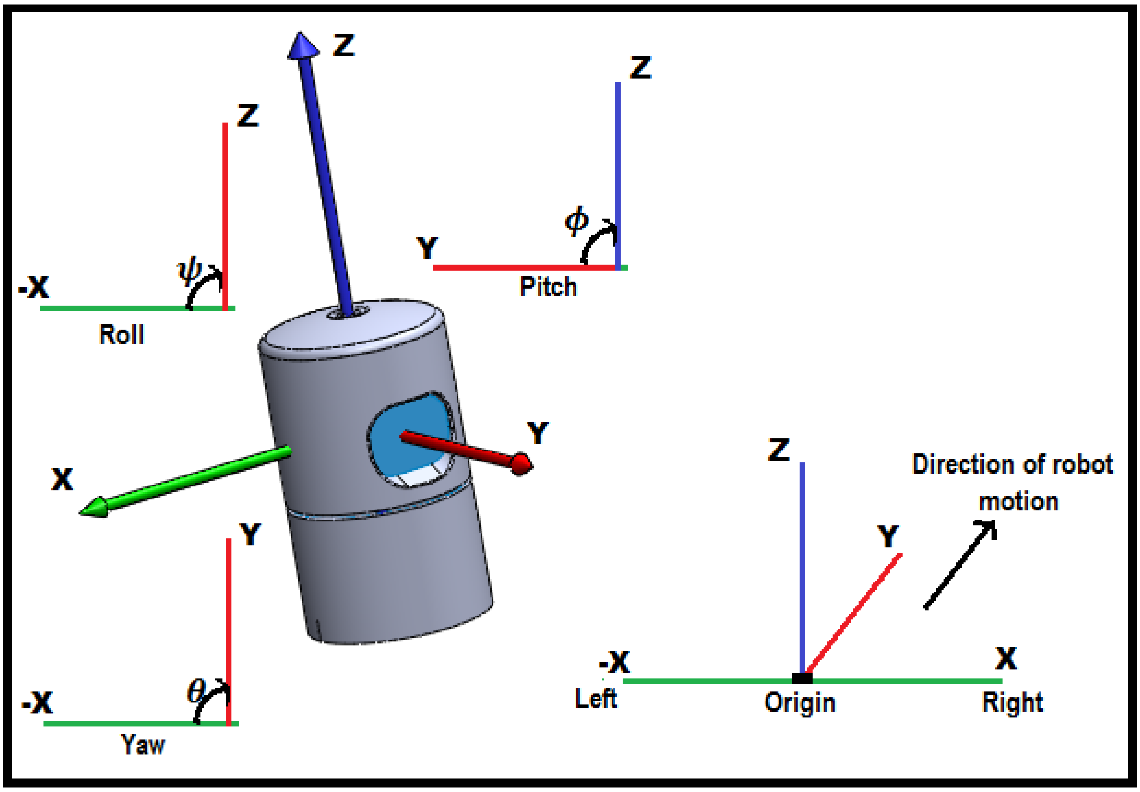


Figure 2.4. Velodyne Coordinate system.

$$
\begin{aligned}
x &= -R\cos\phi\cos\theta \\
y &= -R\cos\phi\sin\theta \\
z &= R\sin\phi
\end{aligned}
\tag{2.1}
$$

Lasers spreading out of a HDL-32 scanner are shown in Figure 2.5 along with a sample scan in Figure 2.6, where the robot is moving towards the north direction. A set of 32 individual lasers are shown in different colours with their

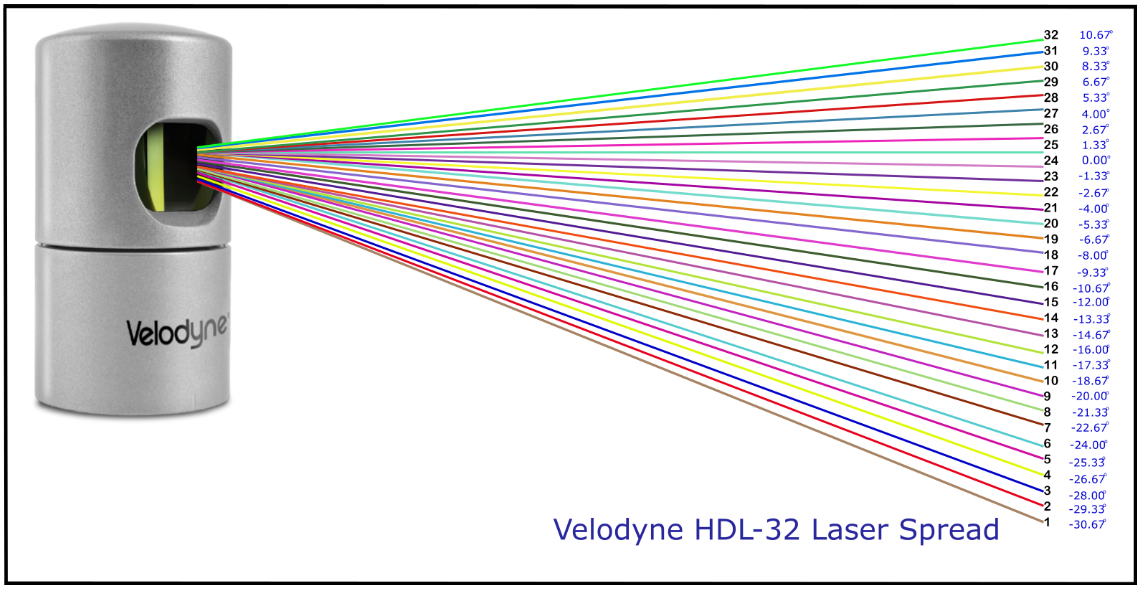


Figure 2.5. Velodyne HDL-32 Laser spread.

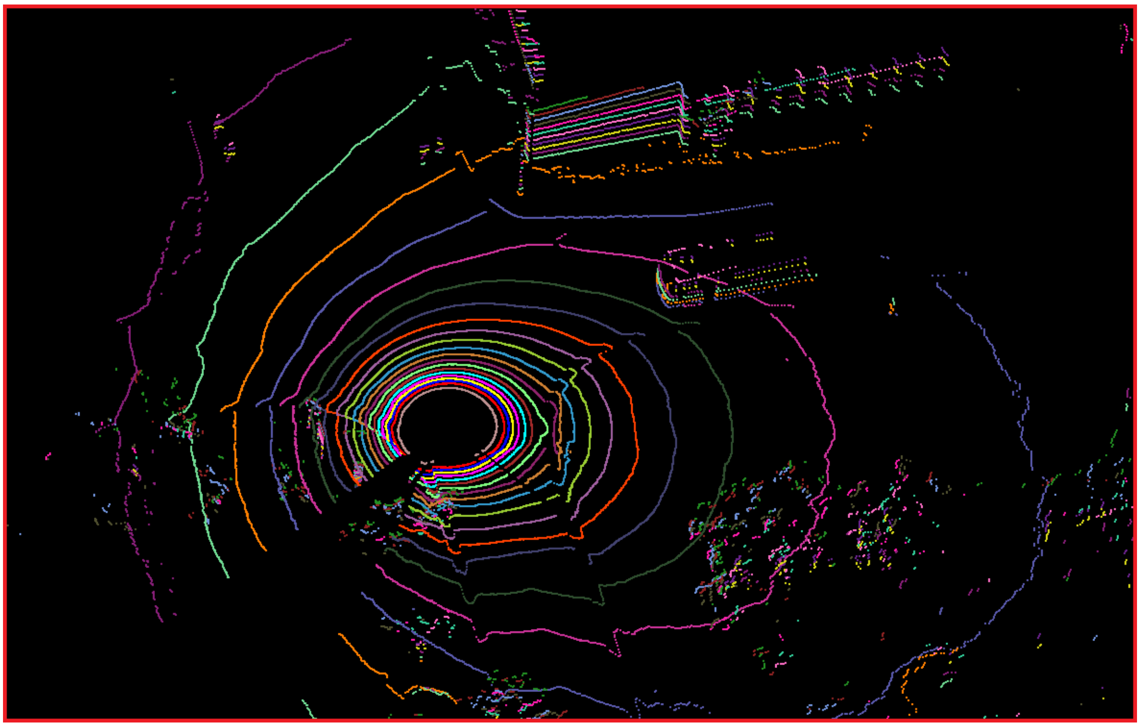


Figure 2.6. Sample Point-cloud with each colour points sampled by one laser beam.

pointing or the pitch angles ϕ_v, where $v = 1$ to 32. In the figure a slight roll experienced by the robot towards left causes the circular rings to compress in that direction. The individual UDP data packets from the scanner are processed by parsing the packet contents of each point p in the point cloud $P(p \in P)$ for Range (R) along with the rotation or azimuth angle (θ) based on the available order of the elevation or pitch angles (ϕ). Additionally, for each point, Velodyne sensors return intensity values (I). While intensity values are also used for developing perception [40], they are often very noisy. They are also non-linear in nature; hence this work did not use intensity values. Apart from data packets, Velodyne scanner also outputs status packets providing information on global position in terms of latitude and longitude obtained from a separate connected GPS receiver along with the universal time for synchronization between different readings. Status packets also provide the inertial measurements along the three axes for pitch roll and yaw. There are three readings each for the rate of rotation and rate of accelerations along the three axes [65].

The resulting data from Velodyne scanners is sparse in the radial direction *i.e.* along the pitch angles. Because of the difference between the consecutive pitch angles being constant (1.33°), as one goes away from the scanner, starting from the inner most laser, the distance between the points sampled by consecutive laser beams increases(Figure 3.1 and Figure 3.2 in Chapter 3). However, in the transverse direction the data is very dense. This is because the angular resolution between the neighbouring points in this direction being very small ($\approx 0.16°$). This data requires specific algorithms for the purpose of extraction of useful information.

Several problems need to be addressed when working with data from 3D Laser scanners. Laser scanners sample surfaces in terms of points. A point in isolation cannot provide any meaningful surface information. Also since the object surfaces are only sampled with a set of points, a complete recovery of the entire geometry of object's surface is difficult. This is particularly so when sampling non continuous surfaces like trees or vegetation. These correspond to violations of the sampling theorem [66].

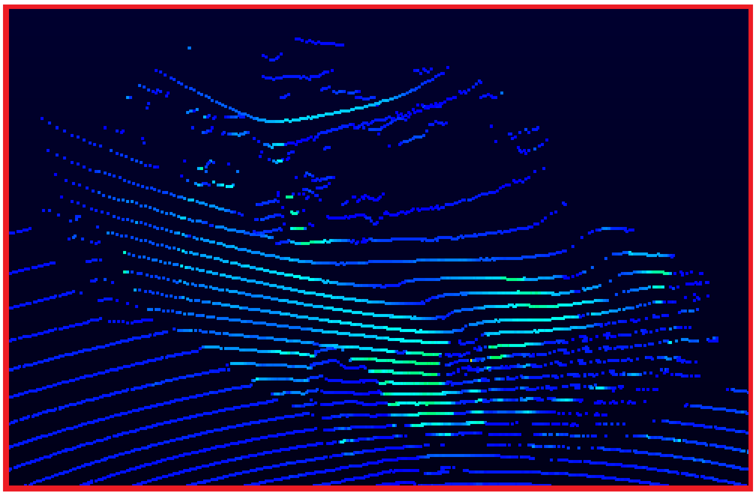


Figure 2.7. A car sampled by laser scanner with intensity values.

To present the challenges of 3D scan data, Figure 2.7 shows a part of laser scan containing a car sampled by different laser points. A subset of points represents an object (car) and the traversable regions (road) under it. An alternative is to build surfaces using neighbouring points. However that also proves difficult given the non-uniform nature of scan along the radial and transverse directions and also because of the fact that different surface patches look alike. It is difficult to establish correspondence between points or surfaces measured at different times from different locations, since exactly the same point is rarely sampled from two locations from a moving robot. It can also be observed that near the surface discontinuities, the points are not too far apart. The range difference between two neighbouring points lying on two different surfaces is also very small to be effectively used for detecting a boundary. For example, in Figure 2.7, it can be observed that the points on the car tyre and the road surfaces are very close to one another, making it difficult to detect tyre edge using distance or range measurements.

Location of the sampled surface point is not only dependent on the laser's range, but also on the pitch (ϕ) and rotation (θ) angles at which the laser beam is fired. It is thus necessary to have accurate values of these angles. Variation of laser pointing angles (intrinsic) from the pre-specified values and placement offsets of the scanner with respect to the vehicle frame (extrinsic) also affect the measurement accuracy of range values. Range measurements are very sensitive to error in rotations; a small error in angle causes a large error in range measurement. Calibration is thus required to overcome these offsets and make corrections to pointing angles. Calibration of the lasers is a research field in itself. While it is easy to calibrate single beam lasers by using hand measurement against the ground truth, this becomes difficult with the multiple beam laser scanners.

Muhamad and Lacroix[67] perform a supervised calibration using a dedicated calibration target based on an optimization process which gives precise optimization parameters for laser calibration starting from an initial estimate. However, optimization is performed by comparison of scanned data with ground truth. This is tedious and becomes difficult with increasing number of lasers. Sheehan et al. [68] perform automatic self-calibration of laser scanners as a task of maximizing the point cloud quality. They use the Renyi Quadratic Entropy to measure the degree of organization in a point cloud. By expressing entropy as a function of unknown system parameters, calibration parameters of the sensor are deduced using an online optimization mechanism. Levinson and Thrun [69] provide a novel fully unsupervised calibration method which also recovers the extrinsic pose of the laser scanner with respect to the vehicle frame. Here no calibration target, labelling or manual measurement is needed. Calibration is done relying on a weak assumption that the points tend to lie on a continuous surface. An energy function is defined to penalize points far away from the surface defined by the points of other sensors. They are able to obtain calibration of all the sensor parameters in a few seconds of running of the vehicle. Manufacturers of the laser scanners perform calibration on individual units at their dedicated calibration facilities before shipping, the knowledge of which is not available to the users, but for the calibration parameters. The present work uses such parameters as are provided by the manufacturer of Velodyne HDL-32.

2.5 Organized Range Data

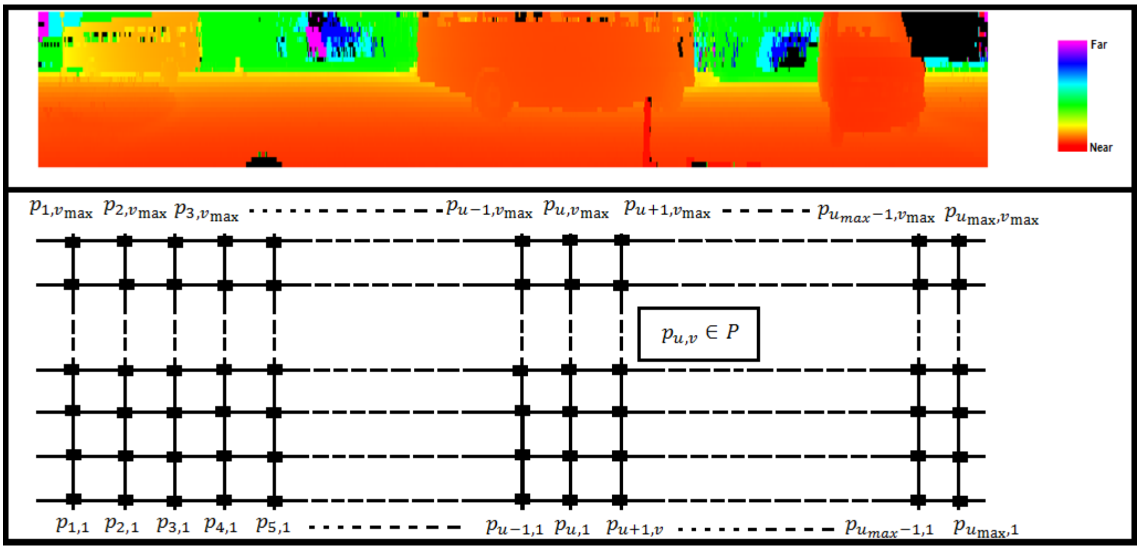


Figure 2.8. Top: Point cloud represented as Range image (Range HSV coded), Bottom: Organization of Range data.

The organized nature of scan from these sensors can be represented using a 2D range image as shown in Figure 2.8. Here the horizontal axis represents the scanner rotation angle (θ) starting from left. Vertical axis represents the pitch angle index v starting with the lowest pointing angle from the bottom. Each point can be indexed for range $R_{u,v}$ by using rotation angle index u as in θ_u and index of elevation angle v as in ϕ_v. A range image can be considered a data matrix indexed for range using u and v.

Figure 2.8 shows a range image obtained from one complete scanner rotation where the range values $R_{u,v}$ are encoded in terms of Hue Saturation Values (HSV) starting with Red for near ranges to Purple for far ranges (Max range for HDL-32 is $80m$). Range image is little stretched along the vertical direction to improve visualization. $R_{u,v}$ is range of point $p_{u,v}$ returned by a laser beam with azimuth angle θ_u and pitch angle ϕ_v. At the centre and left of the image, there are two buses, and to the right there is a utility vehicle (jeep). A vertical pole-like object near the front wheel of the bus is a cylindrical communication antenna present on our robot, which being near is encoded in bright red. The value of max in v_{max} is 32 for the HDL-32. The maximum value max in u_{max} is the number of azimuth angles (θ_u) at which the range readings are taken. Scanner rotating at $10Hz$ frequency results in horizontal max to be between 2100 to 2200 depending on the number of data packets in one rotation. Bottom image Figure 2.8 shows the organized nature of data points with each node $p_{u,v}$ representing a data point connected to its neighbouring points.

Each vertical column contains ranges obtained at one azimuth angle (θ_u). Each row consists of ranges obtained by the laser that fires at pitch angle (ϕ_v) while the sensor rotates through (θ_1 to $\theta_{u_{max}}$). Representation of the range data in this fashion allows the processing algorithms to access neighbouring points directly using indices of points without resorting to computationally expensive nearest neighbour search methods like the k-d trees [70].

2.6 Data conditioning

Raw range data from the laser scanners is conditioned prior to processing by perception algorithms for obstacle detection, segmentation or localization. Data is first pre-processed for reducing noise. Noise in range measurement, in addition to the rough nature of the scanned surfaces makes the use of raw data difficult for direct processing, since we are trying to detect very small discontinuities. One way to reduce noise is by smoothening of data, taking the average of ranges from the neighbouring points. Taking averages however results in smoothening of valid edges, which is not desirable. Bilateral filtering [71] can be applied to reduce noise while preserving the object edges. Techniques for de-noising of data like Weiner deconvolution [72] can also be used. However, these techniques require significant additional processing. As the data from HDL-32 used in our experiments is dense in the transverse direction and sparse in the radial directions, filtering is applied only in the transverse direction, i.e. along the individual rings of a laser scan. A simple median filter [73] with a window size of three points is applied. A narrow window size helped in retaining narrow features such as poles. Median filtering removes points with abnormal deviations from its neighbours.

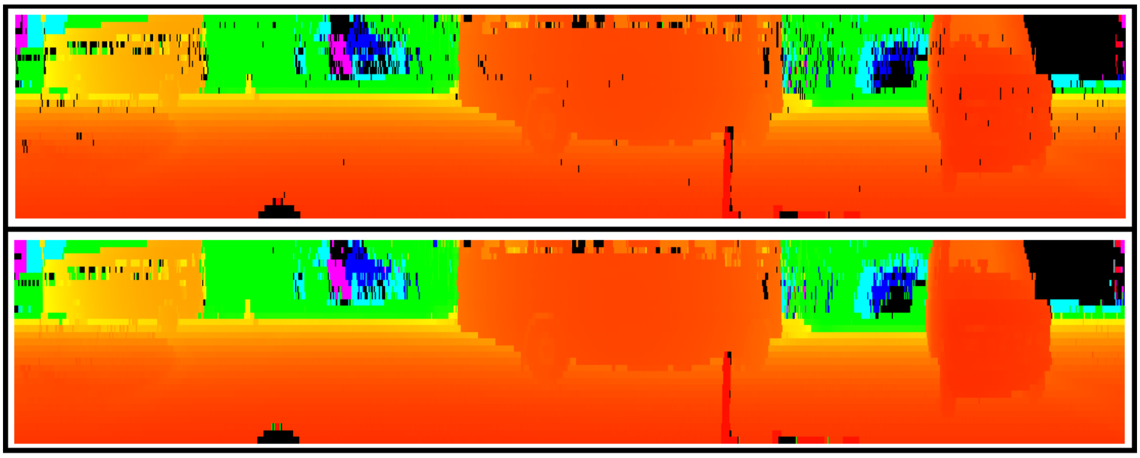


Figure 2.9. Top: Raw data, Bottom: Processed range image after filling missing points.

In addition to noise, raw data contains some points which do not return any range. Specular reflections are the primary reason for such points, where a detector does not detect any reflected beam. Although it is possible to interpolate for the missing data points using range values from their neighbours, we perform a simple look-at-immediate-transverse-neighbours and fill the missing range values. As the data is dense along a ring, and the surface being contiguous, the range values of immediate neighbours are usually similar; we thus fill with the value of least range from the immediate neighbouring points into a point which does not contain any range.

Substituting for null ranges in the raw range image with valid range values, the structure of an organized range image and the continuity of the object surfaces is maintained. In Figure 2.9, the top image shows range image with raw data with several black boxes scattered throughout the image. These are locations in the range matrix with zero values within surfaces that otherwise are supposed to contain some range values. It is observed during experiments that the number of specular reflections is more on a sunny day particularly around noon. The bottom image shows the processed image where most of the void points are given values from their neighbours ensuring smooth object surfaces. Conditioning of points in this fashion will allow unevenness (described later) to be applied on continuous surfaces using neighbouring points. Presence of void points makes it difficult to calculate unevenness, as it relies on range values of neighbouring points. This conditioned data is then processed by perception algorithms to bring out prominent features in the robot's surroundings.

Chapter 3

Unevenness and Traversability

This chapter introduces a novel measure of terrain unevenness that is central to this research work. Unevenness is developed by exploiting the structured nature of range data obtained from the multi-beam rotating laser scanners. Unevenness is shown to be sensitive to discontinuities in the scanned surface. That enables it to bring out clear boundaries between an object and ground and between two overlapping objects. In addition to detecting regular obstacles like trees, buildings and vehicles for the robot to avoid colliding with them, unevenness detects smaller obstacles such as road boundaries and pavement edges. Detection of stones or kerbs is also essential, as otherwise the robot may try to drive over them and hurt or destabilise itself. It is also important to detect negative obstacles like ditches and potholes so that the robot can avoid getting stuck in them.

This chapter defines unevenness (Ω) and then proceeds to derive an analytic expression for it in terms of range data. The resulting unevenness field is analysed in order to set thresholds for detecting obstacles at different ranges. Obstacle detection is faster because it is done as and when the data is received column wise, *i.e.* along the radial direction. Subsequently the traversable region around the robot is obtained as a connectivity graph by connecting the non obstacle ground regions. Unevenness is shown to be robust in detection of obstacles in spite of terrain slopes and sensor tilts caused by the pitch and roll experienced by a moving robot.

3.1 State of the Art

Obstacle detection and traversability are active areas of research in robot navigation. Papadakis[74] classifies methods employed for classifying terrain as a stage preceding motion planning. Primary classification is into Proprioceptive and Exteroceptive sensory data processing methods. Proprioceptive sensors like vibration sensors gauge the nature of surface when the vehicle moves over the terrain [75]. Exteroceptive sensors including Cameras, Time-of-Flight cameras [76], Stereo cameras [45, 49] and Laser scanners perceive the environment from the robot's perspective. This is essential for safe navigation of autonomous vehicles. In some approaches, multiple sensors are used [76, 77, 78]. Because of high accuracy, field-of-view and direct range reading, most methods including ours primarily use Laser scanners. Laser scanners provide a direct measurement of range to the nearest intercepted object.

Data representation is primarily in the form of point clouds, but Occupancy grids [27] are commonly used for spatial representation of the environment. Occupancy grids can be in 2D, or in 3D called voxels [49, 79, 80]. Some approaches [25, 81] build 2D maps from 3D range data. Here substantial information is lost due to projection of data onto 2D grid, making them suitable only to environments with predominantly flat terrains. A 2.5D representation is commonly used as a trade-off between completeness of 3D and the simplicity of 2D representation. It was used successfully [30] in DARPA challenge where the height difference or elevation map was the most common approach in determining obstacles [3, 7, 82] [12]. The height of point from perceived ground gives elevation of the region. This approach faces difficulty in dealing with overhanging objects like tree branches, where a potentially traversable region under a tree is labelled as non-traversable. Overhanging obstacles are dealt with by employing a safe height [3, 31] or by using extended elevation maps [34] where the obstacles above a free space are safely discarded. Points in a grid cell are discretized into different levels based on heights called Multi Level Surface Maps [35] allowing vehicles to traverse on multiple surfaces, like over and under the same bridge.

Polar grids overcome the problem of non-uniform data distribution present in the regular rectangular grids. They provide natural distribution of 3D laser scan with discretization along the rotation angle and returned ranges. Several implementations [31, 32, 83] use polar grids for detecting obstacles. Korchev et al [84] perform real time segmentation using min-map, max-map and difference-map to remove even non-flat ground and then do the 8 connected component analysis and compare performances using rectangular and polar grids. Himmelsbach et al. [32] bin polar segments in radial direction. Each bin has a single representative point reducing the number of data points. Piece-wise line-fitting is done to segment ground and obstacle points. Points are classified as ground when the line segment has slope and height difference smaller than their respective thresholds.

Unevenness detects obstacles at a point level, where each point is classified as either belonging to ground or to an obstacle. Point level detection leads to real time processing of the data points at a sensor rotation angle starting from the inner most point and proceeding outwards. This overcomes the problem of under-segmentation. Chang et al.[85] primarily use slope between last classified ground and the next point for detection of obstacles. For detecting large objects faraway, a height difference threshold of 1m is set. This does not detect smaller obstacles, which unevenness is able to detect. Shneier et al. [78] learn the nearby regions using sensors like stereo or Lidars using a similar approach [85] and then predict the traversable nature of regions beyond the sensor range by using camera images. Steinhauser et al.[86] identify points belonging to road surface by fitting lines to traversable points beginning from the innermost point. Traversable regions are detected beyond the obstacles again by fitting lines. Wulf et al. [25] detect obstacles by finding edges on scans both in the horizontal direction based on range discontinuity and in the vertical direction by locating an obstacle point that is vertically above a ground point. This method however is suited only for predominantly flat terrains. Slopes at times are traversable if they are within the navigational capability of the robot; so a terrain with slope less than a given threshold can be traversable [31, 77, 78, 87]. Morton and Olson [87] use a Height-Length-Density (HLD) terrain classifier which even deals with partial observability for detecting obstacles. Cells are classified using confidence levels.

Segmentation into ground and obstacle points can be model based, where primitive shapes like cylinders [88] are fitted into data. Goron et al. [89] fit planes in dense data regions containing background information like walls and traversable ground regions before segmenting the sparse foreground data clusters. Murarka and Kuipers [49] use stereo data to fit planes for finding ground planes and inclines before connecting them to form a traversable region. Nguyen [31] adapts this method for Velodyne sensor, but using a polar grid containing maximum height values in each cell. After fitting the planes between the adjacent cells using maximum height values, adjacent plane segments are connected into drivable regions. Reina et al. [45] uses an Unevenness Point Descriptor (UPD) to capture terrain irregularity and inclination using normals obtained from point's neighbourhood using stereo data. This works on slopes but is not real time and lacks sharp distinction between ground and obstacles.

Vasudevan et al. [90] propose Gaussian Processes for modelling terrain using sparse data. For sparse data, Douillard et al. [79] improves over Gaussian processes with outlier rejection for ground segmentation, called Gaussian Process Incremental Sample Consensus (GP-INSAC). Following this, Das and Waslander[91] removed ground using GP-INSAC before segmenting non-ground points into clusters that are subsequently used for scan registration. Chen et al. [83] report real-time performance with polar grid processing information using a one dimensional continuous Gaussian Regression in the radial direction and a non-stationary covariance function to determine ground points. This adapts to terrain undulations, making it robust to rough terrains. However, the method shows incorrect results by detecting smaller obstacles such as steps as traversable and uphill roads as obstacles. Santamaria-Navarro et al. [76], apart from detecting obstacles in real time using Time-of-Flight cameras, use a Velodyne Lidar data, and classify the traversable region using a high-level offline classification method. Gaussian processes as a regression tool is used to learn terrain parameters for classifying the terrain.

Graph based approaches using weighted graphs are built based on similarities between the connected nodes. Surface direction, edges, curvatures and smoothness are used as attributes for similarity for construction of graphs. Moosmann et al. [43] perform segmentation using local convexity criteria after building a 2D graph. Convexity is obtained after calculating point normals using four neighbours. Results show the method to detect ground even in non-flat areas. Kuthirummal et al. [59] use graphs to connect traversable cells containing point height histograms for both Lidar and stereo data. The method performs in real time and does not need pitch roll compensation. Guo et al. [92] using graph based approach classify the points into 4 classes of reachable, drivable, obstacle or unknown regions with Markov Random Fields (MRFs) using gradient cues of road geometry. MRFs also find implementations in Anguelov et al. [93] and Wolf et al. [60], where classification obtained from Hidden Markov Models (HMM) is refined using MRFs.

Principal Component Analysis (PCA) on unit regions analyzes spatial distribution of points from the obtained eigenvectors. By applying PCA on individual voxels, Lalonde et al. [80] classify regions into scatter, linear or surface. Surface represents traversable regions, scatter represents obstacles and linear regions have features like wires. Pellenz et al. [81] apply PCA recursively on non-flat regions to overcome data sparsity in a grid and recursively grow traversable region. Sinha and Papadakis [94] assess traversability based on features extracted through PCA by exploring the spatial distribution of the interiors in individual gaps or the orientation distribution of corresponding contour with focus on negative obstacles. PCA is also performed on scaled spin images to obtain point features for further classification in Anguelov et al. [93]. Larson et al. [95] use eigenvectors to get flat regions and their surface normals to compute slope.

In contrast to the methods described above, unevenness identifies the traversable region without using normals, classification, plane fitting, Gaussian Regression, PCA or MRFs, which are computationally expensive or require training for classification. Traversability is obtained by a simple and efficient method on a single frame of sparse laser data primarily by computing an unevenness field using only the ranges and the scan geometry of a rotating laser scanner.

3.2 Obstacle detection in radial direction

The rotating head of Velodyne laser scanner fires a set of lasers (32 for HDL-32) in a vertical plane, which scan the terrain along a radial direction. The way in which the sensor scans the environment was explained in chapter 2. At a given rotation angle (θ) at which lasers are fired, obstacles can be detected along the radial direction starting from the inner most range point and proceeding outwards. This procedure is repeated at each rotation angle (θ) at which the lasers are fired. In terms of range image, obstacle detection is performed column wise starting with the bottom pixel in each column.

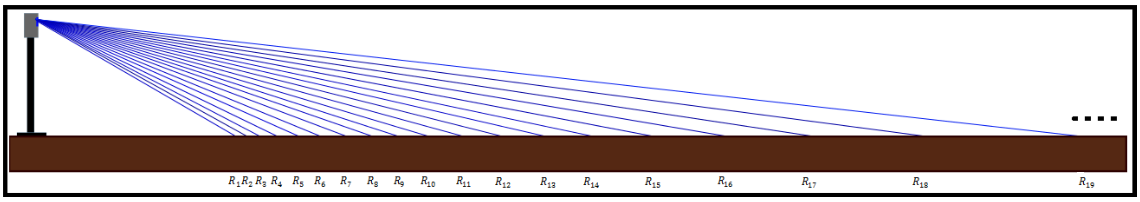


Figure 3.1. Laser points in the radial direction.

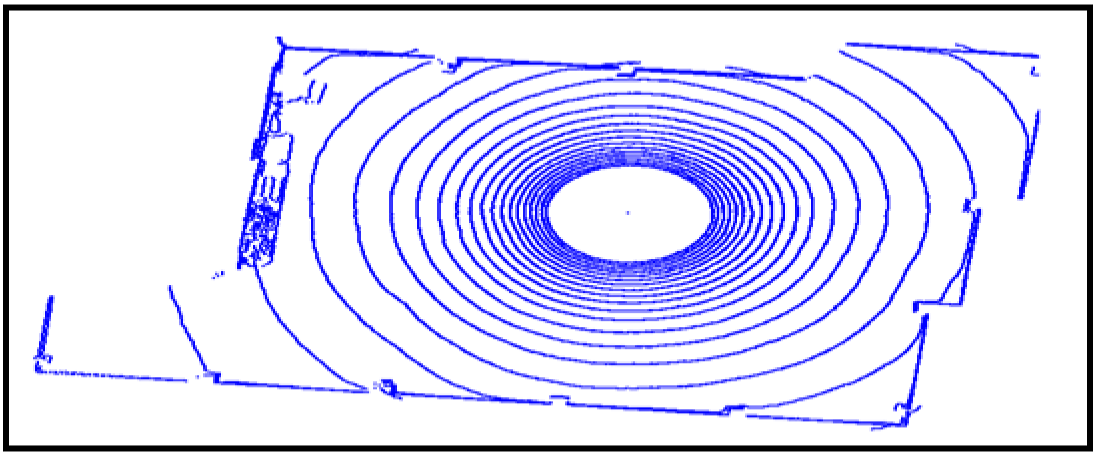


Figure 3.2. Sample laser scan of a terrace.

Figure 3.1 shows a set of laser beams in the radial direction at a rotation angle θ. Only nineteen laser beams are shown, but a total of 24 laser beams sample the ground from an erect sensor. Remaining higher beams are intercepted by either vertical obstacles or return no range when not intercepting any object. The distance interval between the consecutive ground points gradually increases with distance from the sensor. Distance interval is far greater for higher ranges (say between R_{22} and R_{23}). The resulting scan with 360° rotation results in a set of concentric circular ring like patterns on a flat ground. Figure 3.2 shows a sample scan taken over our lab terrace where the terrace floor is sampled by circular rings, but the boundary walls and clutter disturb the circular patterns. Obstacles can be detected by evaluating the following parameters between two consecutive range points [96] in the radial direction. An outer point among the consecutive points can be classified as an obstacle using

1. **Height difference:** When the height difference between the two points is greater than a threshold.

2. **Ring distance:** When the distance between the consecutive points is less than a distance threshold.

3. **Slope:** If the slope intercepted between the consecutive points is greater than a threshold.

4. **Range difference:** When the difference between ranges of two consecutive points is less than an expected range threshold.

Considering the nature of point distribution, height difference between successive points for a vertical obstacle depends on range. The difference at nearby ranges is very small and it typically increases with range. Distance between two points (horizontal) also increases with range and is also affected by the roll and pitch experienced by a moving vehicle resulting in compression and expansion of inter ring distances. Similarly, expansion and compression occurs because of slopes present in the terrain. On the other hand, smaller elevation changes between closer points cause higher slopes at nearby ranges and same obstacle heights result in very small slopes at distant ranges. All these factors cause difficulties in setting thresholds across ranges. The approach using the range difference better adapts to the terrain slopes and tilts experienced by the sensor when the expected range difference is made a function of the inner point's range [3, 33]. This is the idea with which the concept of unevenness is built and developed. Since unevenness is dependent on the difference in ranges and angles between the corresponding beams and not on the absolute pitch angle (ϕ), as we shall describe in the coming sections, unevenness can be used with advantage even with beams that sample higher objects.

Figure 3.3, shows obstacle detection along the radial direction using both the height difference method and unevenness. Height difference method detects

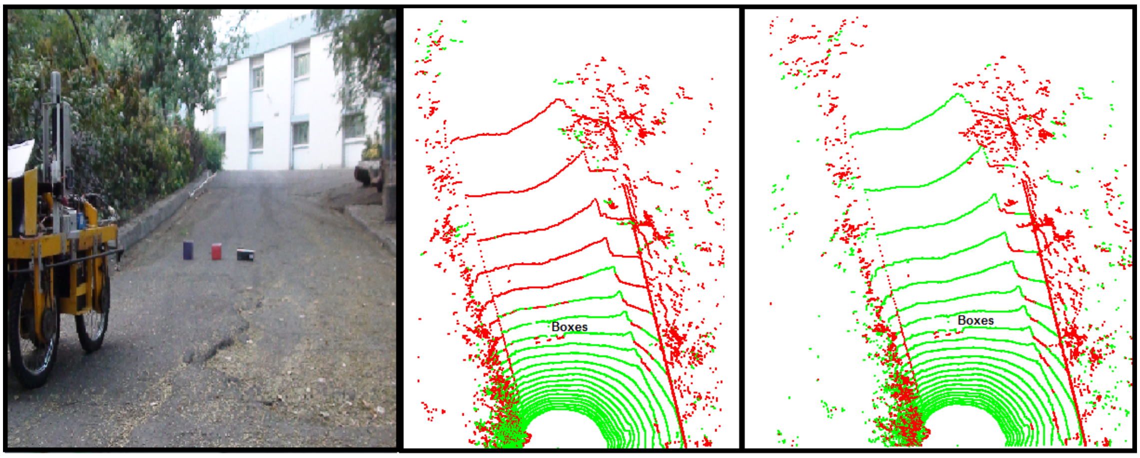


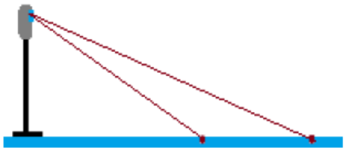
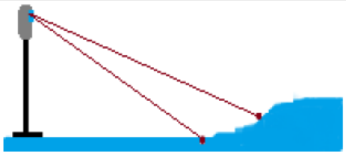
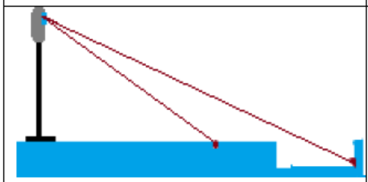
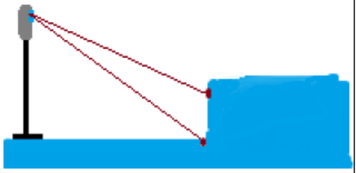
Figure 3.3. Obstacle detection on an upward slope, Left: Environment with robot, Centre: height difference method, Right: Detection using unevenness

large part of traversable slope away from the sensor as obstacle region which is not desirable. Setting a higher height threshold will not detect smaller obstacles like foot path edges. Unevenness being insensitive to smaller slopes detects most of the slope as traversable even when it detects the smaller height obstacle-like boxes $(10cm, 20cm, 30cm)$ and footpath edges as obstacles.

3.3 Unevenness

The range of inner point can be used to calculate the expected range difference between consecutive laser beams on a flat ground. This expected range difference (δR_e) can be computed and compared against the measured range difference (δR_m) that is readily available from sensor readings ($\delta R_m = R_{u,v} - R_{u,v-1}$, where u is the column number along the direction of sensor rotation). Unevenness (Ω) depends on the ratio of measured range difference (δR_m) to the expected range difference (δR_e). We interpret this ratio $(\frac{\delta R_m}{\delta R_e})$ as indicative of evenness of the terrain at the point of measurement. For an even surface, this ratio will be close to unity. Unevenness can now be seen as deviation of this ratio from unity. To capture the traversability of a point, we define unevenness Ω as

Table 3.1: Terrain variations and the respective Unevenness (Ω) values.

Nature of Terrain	**Relation**	**Description**	**Unevenness Ω**
	$\delta R_m = \delta R_e$	$p_{u,v}$ and $p_{u,v-1}$ belong to the same plane. In other words there is continuity in the terrain.	$\Omega = 0$
	$\delta R_m < \delta R_e$	$p_{u,v}$ is a point closer than its normal range to $P_{u,v-1}$ and is an obstacle point with $p_{u,v}$ at a higher elevation	$\Omega < 1$
	$\delta R_m > \delta R_e$	$p_{u,v}$ is a point away than normal from $P_{u,v-1}$ and is a negative obstacle point with $p_{u,v}$ at a lower elevation.	$\Omega < 0$
	$\delta R_m \approx 0$	$p_{u,v}$ and $p_{u,v-1}$ have similar range and$p_{u,v}$ is vertically above $p_{u,v-1}$.	$\Omega = 1$

$$\Omega = 1 - \frac{\delta R_m}{\delta R_e} \tag{3.1}$$

Here, a value of unevenness (Ω) close to zero indicates that the point is on an even surface. On the other hand, values away from zero represent the degree of unevenness of the surface sampled by the point. A positive value indicates a positive obstacle, and a negative value indicates a negative obstacle, i.e., a depression. Values of δR_m and δR_e indicate nature of terrain. The unevenness (Ω) for different terrain possibilities for classifying point $p_{u,v}$ depending on its inner neighbour $p_{u,v-1}$ are listed in Table 3.1.

Unevenness across all the points sampling the robot's surroundings intuitively represents the terrain surrounding a robot. Although range difference between neighbouring laser beams is used earlier in DARPA challenge by Stanford's

Junior [3], the unevenness based approach presented here is novel and original. As detailed later, unevenness (Ω) varies with range and hence needs to be characterized for setting thresholds for the detection of obstacles. Even though we describe the method using a Velodyne HDL-32 Laser scanner, we show later that the method can be adapted to a rotating Sick 2D scanner.

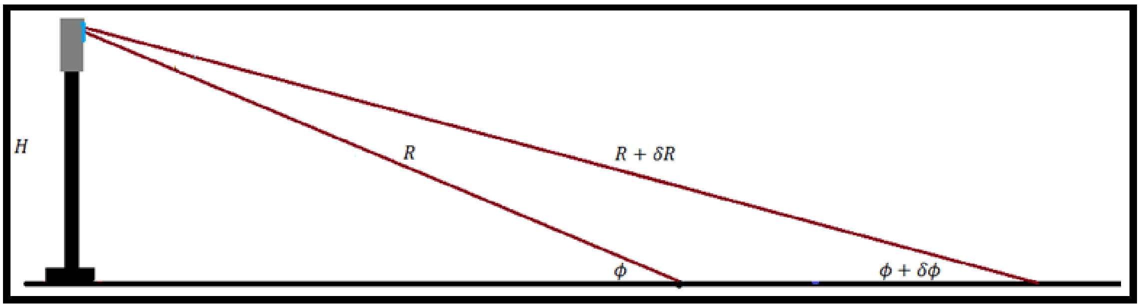


Figure 3.4. Consecutive laser beams in radial direction.

Figure 3.4 shows a laser at pitch angle ϕ and a neighbouring laser at pitch angle $\phi + \delta\phi$, on an even ground. It is possible to relate the difference in ranges δR between the adjacent lasers to the corresponding difference in their pitch angles $\delta\phi$. Our sign convention considers the pitch angles ϕ to be positive as seen from the ground (see Figure 3.4). As a result, $\delta\phi$ is negative as we move to higher beams and longer ranges.

With reference to Figure 3.4, the laser sensor is positioned at height H above ground; the range R of laser beam and its pitch angle ϕ are related as

$$R = \frac{H}{\sin\phi} \tag{3.2}$$

Taking differential on both sides of Equation 3.2, and then substituting $\frac{H}{R}$ in place of $\sin\phi$, a relation between difference in ranges δR and difference in pitch angle $\delta\phi$(negative value)between two adjacent lasers can be obtained.

$$\delta R \approx \frac{-R\sqrt{R^2 - H^2}}{H} * \delta\phi \tag{3.3}$$

Since $\delta\phi$ is negative, the expression results in a positive value for δR. A positive value for $\cos\phi$ is assumed because ϕ lies in the first quadrant ($0° \leq 0 \leq 90°$). This relationship between $\delta\phi$ and δR is only approximate. It is more so at higher ranges, where δR increases in leaps and bounds. However, this relationship holds reasonably well at closer range ($\leq 10m$), and is useful for the purpose of characterising the terrain.

When the measured range difference δR_m between two consecutive ranges at a certain azimuth angle is significantly different (quantified later) from the expected range difference δR_e as obtained from Equation 3.3, the outer range may be considered an obstacle point. Measured range may be affected by sensor tilt or terrain slope. To overcome this uncertainty, it is a common practice [3, 33] to use the measured range R_m at the inner range point in the right hand side of Equation 3.3, and not to use absolute values of ϕ, as in Equation 3.1 to compute R. With that, Equation 3.3 can be rewritten as

$$\delta R_e \approx \frac{-R_m\sqrt{R_m^2 - H^2}}{H} * \delta\phi \tag{3.4}$$

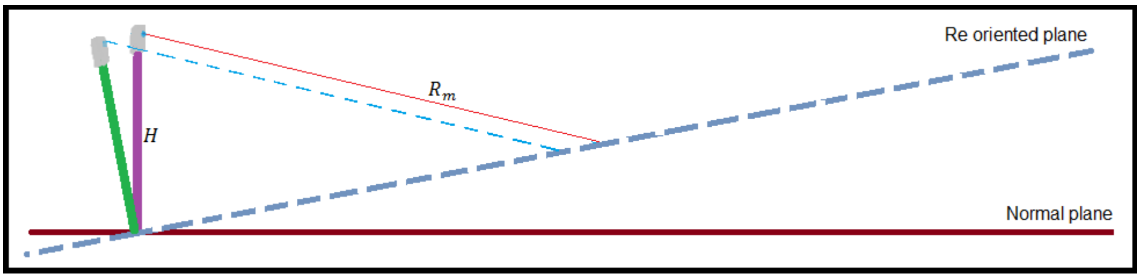


Figure 3.5. Virtual sensor reorientation based on measured range R_m.

This equation forms the basis for detection of obstacles. It is the use of R_m (and not R_e) on the right hand side of Equation 3.4 that makes the resulting value of δR_e relatively insensitive to terrain slopes or small variations in pitch and roll of the vehicle. The use of measured range R_m to compute expected range difference amounts to reorienting the sensor such that the point appears to be on level ground (see Figure 3.5). That is how it expects range difference according to the slope of terrain. The (virtual) sensor reorientation also ensures that angle ϕ for any intercepted point lies in the 1st quadrant ($0°$ to $90°$)for slopes, sensor tilts or even for overhanging obstacles.

3.3.1 Unevenness field

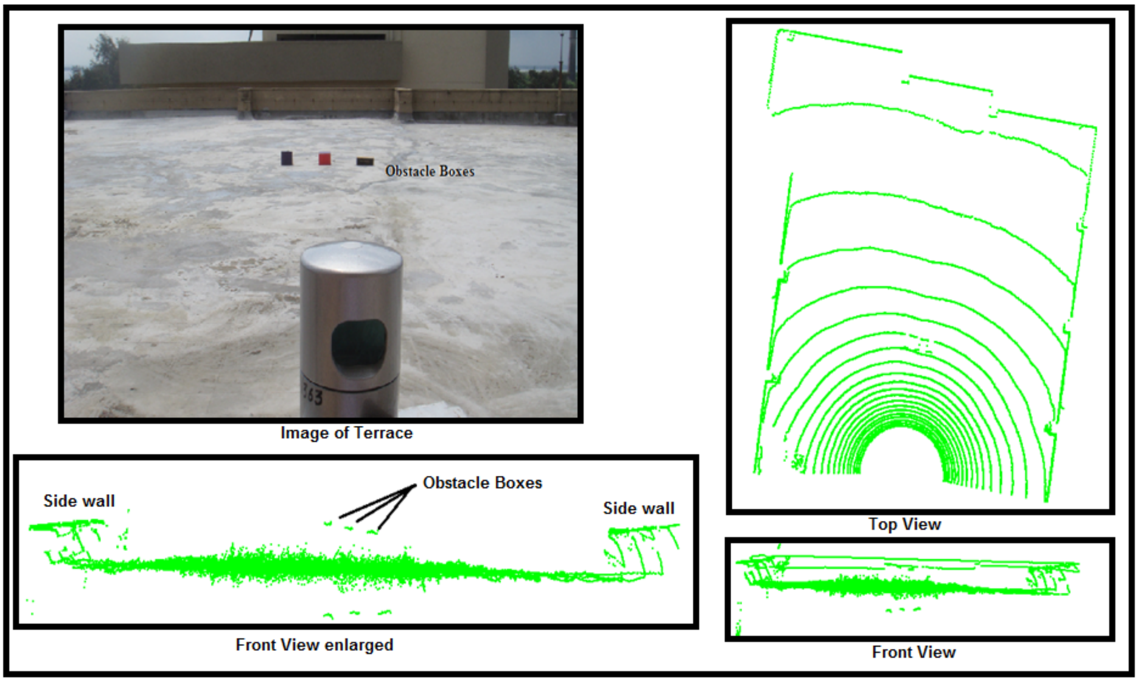


Figure 3.6. Unevenness field: (Clockwise from top left) camera image of terrace, Unevenness (Ω) image of terrace: top view, front view, front view enlarged.

Unevenness computed across all the sampled points in a scan using Equation 3.1 results in an unevenness field that represents the nature of terrain surrounding a robot. This field is visualized in Figure 3.6 for the terrace scan data. Unevenness value, being small, is magnified a 1000 times along the geometric z-axis. Close to sensor, even the flat terrace shows considerable unevenness. This is because of the rough nature of terrace surface, which causes relatively high deviation when the surface is sampled densely close by (radial). The obstacles (boxes) further away are somewhat magnified in terms of unevenness. The walls look smaller in terms of unevenness even though they are quite tall. Thus unevenness, the way it is defined, is very sensitive close to the sensor and the sensitivity reduces progressively as the range increases.

3.4 Modelling and characterizing unevenness

Unevenness Ω is characterized across the range of laser measurements by considering two consecutive laser beams with the inner laser at pitch angle ϕ sampling the ground and the next higher laser with pitch angle $\phi + \delta\phi$ intercepting an obstacle at a height of $-\delta H$ above ground. Negative sign for height change is because of decrease in sensor height with respect to the point of interception of the laser beam.

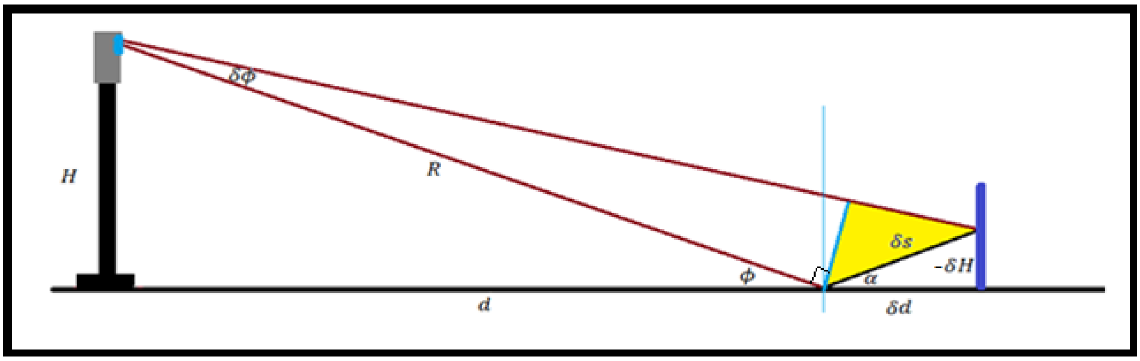


Figure 3.7. Obstacle at the outer ring

The situation, as shown in Figure 3.7, corresponds to a slope α of the terrain between the two beams. With respect to Equation 3.2, on an even ground, we do not expect H to vary with laser beams at different pitch angles ϕ, as all the lasers hit ground at same level. Hence, variation in range R is entirely caused by variation in ϕ. Therefore our expected range difference δR_e is obtained by taking differential on both sides of Equation 3.2 , while treating H as a constant.

$$\delta R_e = -\frac{H}{\sin^2\phi}\cos\phi\delta\phi \tag{3.5}$$

But when the beam at $\phi + \delta\phi$ is intercepted by an obstacle, height H of sensor above the point of interception will also change. Now, allowing the height of sensor H also to vary, the differential of Equation 3.2 becomes

$$\delta R_m = \frac{\delta H}{\sin\phi} - \frac{H}{\sin^2\phi}\cos\phi\delta\phi \tag{3.6}$$

It may be noted that in Equation 3.6, δH is negative for a positive obstacle, so it tends to reduce the range difference, whereas $\delta\phi$ also being negative, tries to add to the range difference. Here we identify range difference in Equation 3.5 as the expected range difference δR_e, as it happens on an even ground. Likewise, the range difference δR_m in Equation 3.6 corresponds to what will actually be obtained in presence of an obstacle, which is why it is identified with measured range difference δR_m.

Using Equation 3.5 and Equation 3.6, unevenness Ω can be expressed as

$$\Omega = 1 - \frac{\delta R_m}{\delta R_e} = \frac{tan\phi}{H}\left(\frac{\delta H}{\delta\phi}\right) \tag{3.7}$$

From the shaded triangle in Figure 3.7

$$cos\left(\frac{\pi}{2} - \phi - \alpha\right) = sin\left(\phi + \alpha\right) = \frac{-R\delta\phi}{\delta s} \tag{3.8}$$

Also from the shaded triangle

$$\sin\alpha = \frac{-\delta H}{\delta s} \tag{3.9}$$

Using Equation 3.8 and Equation 3.9,

$$\frac{\delta H}{\delta \phi} = \frac{R sin\alpha}{sin\,(\phi + \alpha)} \tag{3.10}$$

Now substituting Equation 3.10 for unevenness Ω in Equation 3.7

$$\Omega = \frac{sin\alpha}{cos\phi sin\,(\phi + \alpha)} \tag{3.11}$$

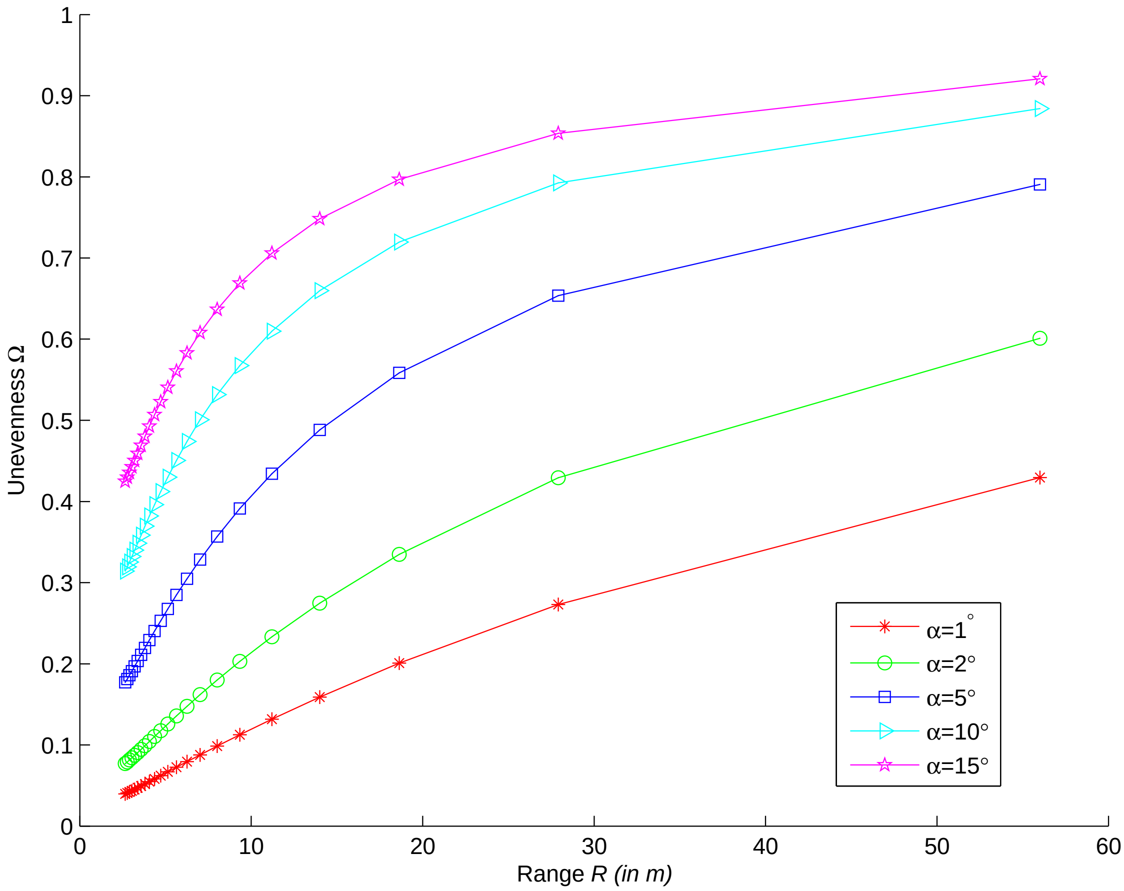


Figure 3.8. Plot of unevenness(Ω) versus Range(R) for different slopes (α).

The unevenness value obtained here is because of slope α between two consecutive points. In Figure 3.8 unevenness (Ω) values for varying ranges R are plotted for different values of slope α. Ranges are obtained for a sensor height H of $1.3m$ using Equation 3.2. From the plot, it can be seen that unevenness is higher for higher values of slopes. In order to detect obstacles, one can think of setting unevenness thresholds based on slopes. However, that proves to be difficult, as even small surmountable obstacles placed close by gives rise to large slopes between neighbouring rings, whereas a small slope at far away distance actually corresponds to a very large obstacle.

Instead of setting unevenness thresholds based on slope between points on neighbouring rings, one can set thresholds based on the inter ring height b(same as $-\delta H$ in Figure 3.7) of obstacle. If d is projection of range R on ground, δd the projection of δs on ground (see Figure 3.7), we can determine the slope α resulting from an obstacle of height b.As the intercepted obstacle height b can be seen to cause a change in sensor's height $-\delta H$, we can substitute b in place of $-\delta H$.

Relating the height of obstacle to its slope

$$\tan\alpha = \frac{b}{\delta d} \tag{3.12}$$

Relating sensor height H with pitch angle ϕ,

$$d = \frac{H}{\tan\phi} \tag{3.13}$$

Differentiating the above equation on both sides

$$\delta d = \frac{\delta H}{\tan\phi} - \frac{H}{\sin^2\phi}\delta\phi \tag{3.14}$$

Substituting for δd in Equation 3.12 and using $b = -\delta H$ as seen from Figure 3.7 we get

$$\tan\alpha = \frac{b}{\frac{-b}{\tan\phi} - \frac{H}{\sin^2\phi}\delta\phi} = \frac{-b\sin\phi}{b\cos\phi + R\delta\phi} \tag{3.15}$$

Expanding Equation 3.11 for unevenness Ω and dividing by $\sin\alpha$

$$\Omega = \frac{1}{\cos^2\phi + \cos\phi\sin\phi\frac{1}{\tan\alpha}} \tag{3.16}$$

Now substituting for $\tan\alpha$ from Equation 3.15

$$\Omega = \frac{-b}{R\cos\phi\delta\phi} \tag{3.17}$$

Substituting for $\cos\phi$, the unevenness Ω can be expressed as

$$\Omega = \frac{-b}{\sqrt{R^2 - H^2}\delta\phi} \tag{3.18}$$

Figure 3.9 plots unevenness Ω values for different values of b at different ranges R. For a given obstacle height b, unevenness Ω decreases with increase in range R. Close to sensor, however, unevenness increases steeply for a given obstacle height.

3.4.1 Setting Thresholds using obstacle interception height

Probability of detection or even interception of small obstacle diminishes with distance. The nature of 3D scan allows a robot to first detect smaller obstacles nearby.

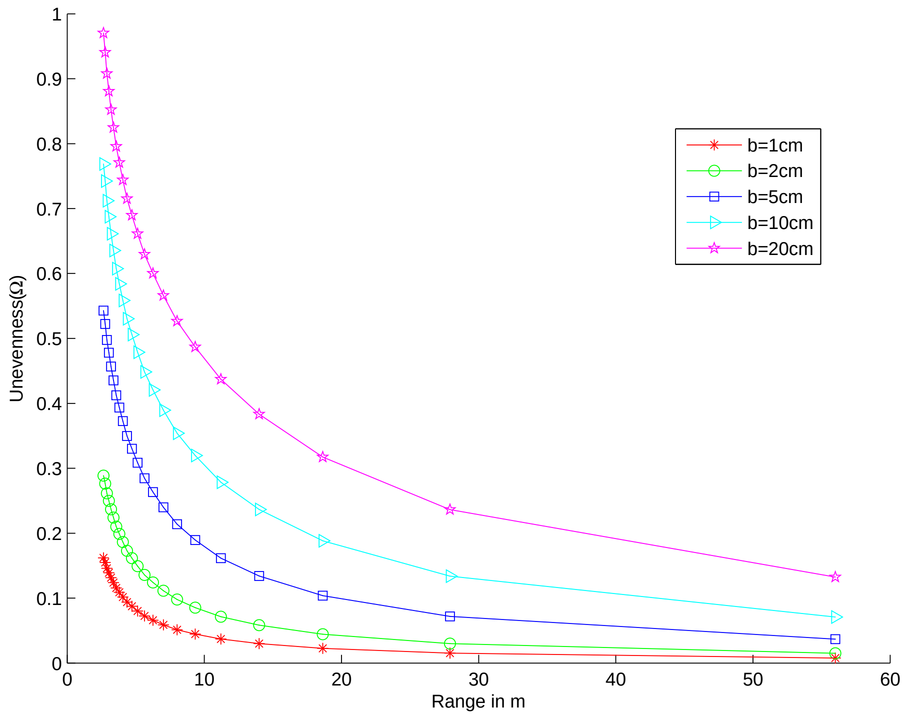


Figure 3.9. Plot of unevenness (Ω) versus Range (R)

Smaller objects faraway are subsequently detected when the robot moves towards them. Bigger size obstacles such as walls, trees, humans or other vehicles are intercepted and detected by lasers even from a distance. For this reason, obstacle threshold in terms of interception height b_{th} is set smaller near the sensor for detecting smaller obstacles like road edges or kerbs. Threshold is then increased linearly with range R using proportionality constant f.

$$b_{th} = a + fR \tag{3.19}$$

This relation is heuristic but worked well for detecting obstacles across the ranges. For positive obstacles, we set value of a^+ (plus sign for positive obstacles; similarly, minus for negative obstacles) at $20mm$. This is the smallest obstacle height at least range. It overcomes high sensitivity of unevenness close to the sensor. We then set obstacle height of $50cm$ at the largest range of $56m$ which is the farthest distance sampled by the laser on ground for an erect sensor height of $1.3m$. Proportionality constant for positive obstacle f^+ is thus obtained as 0.0086. For classifying a point with range R as obstacle, the threshold unevenness(Ω^+_{th}) is computed for the obstacle interception height threshold b_{th} obtained using Equation 3.19. When unevenness (Ω) obtained for the same point using equation Equation 3.1 is more than threshold unevenness(Ω^+_{th}), the point is classified as positive obstacle.

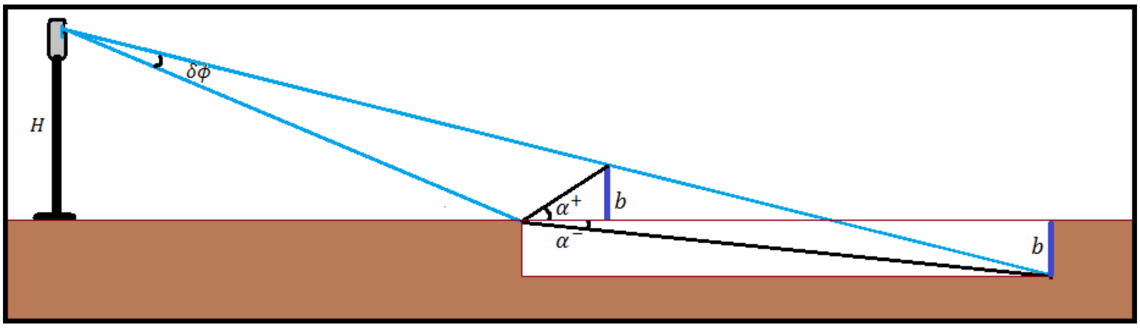


Figure 3.10. Obstacle slopes for obstacle height b in both positive and negative directions

For negative obstacles, the outer point lies below the level of inner point. From Figure 3.10, the negative obstacle subtends a much smaller angle (α^-) compared to positive obstacle angle (α^+) for the same obstacle height (b). Negative obstacle is intercepted at depth b below ground only when it is far from the inner point compared to positive obstacle of same height. This shows the asymmetry between positive and negative obstacles with range. Detection of negative edges is a challenging but important task. While even smaller negative obstacles near robot are easily detectable, they do not scale well with increase in range. Between two adjacent laser beams, the negative obstacle interception height does not increase in the same proportion as that for the positive obstacles. This is well known fact in literature [74, 87, 94] and that's why the negative obstacles are treated separately using absence of data. So we set smaller thresholds for detecting negative obstacles. We start with a minimum obstacle height (read depth) a^- of mere $10mm$ (heuristic) at the shortest range for negative obstacles. However trying to detect a $50cm$ negative obstacle at $56m$ range, we obtain f^- to be 0.00875. Here a point is classified as negative obstacle when its unevenness (Ω) is less than unevenness threshold (Ω_{th}^{-}). A point can thus be classified as non-obstacle when its unevenness lies between the unevenness thresholds in both positive and negative directions ($\Omega_{th}^{-} \leq \Omega \leq \Omega_{th}^{+}$). In order to see the basis for setting different thresholds for detecting positive and negative obstacles, Figure 3.11 plots unevenness curves for different inter-ring obstacle heights (b) against range for both positive and negative obstacles. It shows the asymmetry particularly for ranges near to the sensor. Curves for initial heuristic thresholds heights b_{th}^{+} and b_{th}^{-} are also plotted.

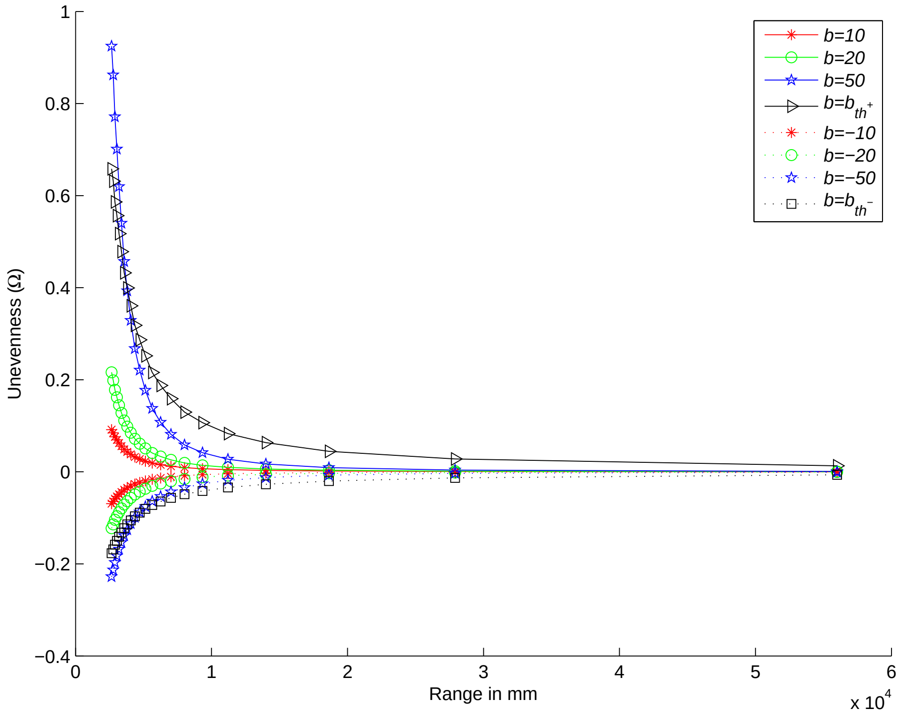


Figure 3.11. Plots for unevenness(Ω) versus Range (R) showing asymmetry for positive and negative obstacles.

3.4.2 Setting threshold using unevenness

When intercepted by a vertical obstacle ($\alpha = 90°$), there is a limiting interception height between the consecutive laser beams, which we call b_{max}, depending on range R, which using Equation 3.15 after substituting for $\alpha = 90°$ evaluates to

$$b_{max} = \frac{R^2 \delta\phi}{\sqrt{R^2 - H^2}} \tag{3.20}$$

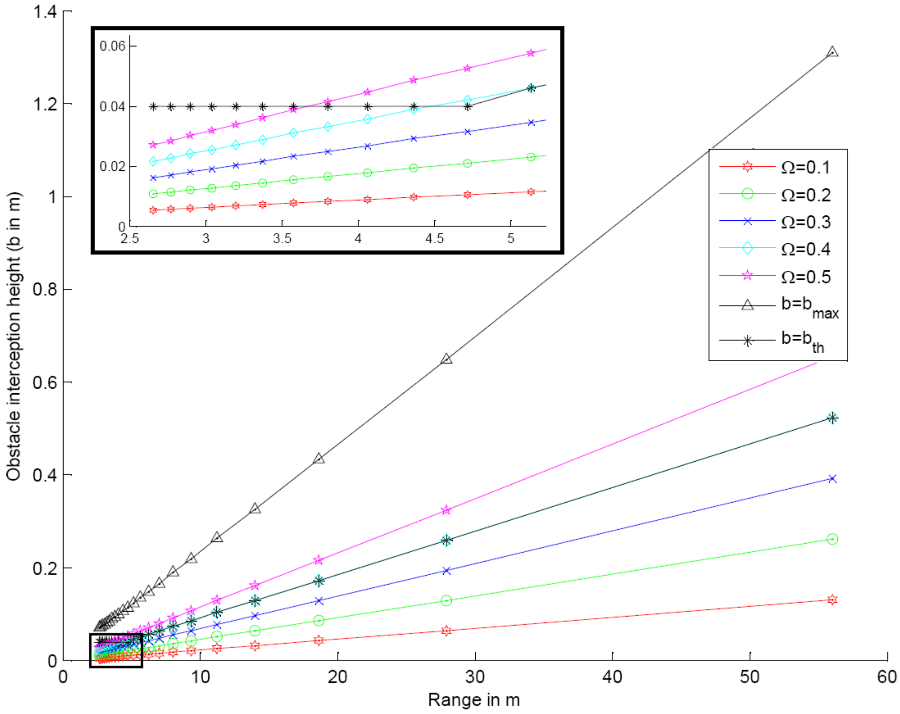


Figure 3.12. Obstacle interception heights (b) for different values of Unevenness (Ω) versus Range (R), Inset: Plots in the near range.

Close to the sensor, b_{max} is small and increases almost linearly with R. This defines and limits the sensitivity of the sensor to obstacles at different ranges. We plot in Figure 3.12, the values of obstacle interception heights (b) against range for different values of unevenness (Ω) using Equation 3.18. We also superimpose b_{max} on the same plot. It is observed that constant unevenness curves correspond roughly to a linear increase in the value of b with range, only slightly deviating close to the sensor (origin). The pattern is similar for the b_{max} curve. That is why it makes sense to set obstacle threshold at a constant level of unevenness. The values of unevenness threshold Ω_{th} can be set according to the robot's ability to negotiate an uneven terrain.

In our experiments with our wheeled mobile robot, we set the unevenness

Algorithm 3.1:

Function: Detect edge in the radial direction, ***radial_edge*** (u, v)

Input: Point with indices u and v

Output: Returns true if $p_{u,v}$ is an obstacle point with respect to $p_{u,v-1}$

Obtain pitch angle difference between radial neighbours

$\delta\phi = \phi_{u,v} - \phi_{u,v-1}$;

Obtain the expected range difference between point $p_{u,v}$ and point $p_{u,v-1}$

$\delta R_e = -\frac{R_{u,v-1}\sqrt{R^2_{u,v-1} - H^2}}{H} * \delta\phi$;

Measure the range difference between point $p_{u,v}$ and point $p_{u,v-1}$

$\delta R_m = R_{u,v} - R_{u,v-1}$;

Obtain the radial unevenness for point $p_{u,v}$

$p_{u,v}.\Omega = 1 - (\frac{\delta R_m}{\delta R_e})$;

IF ($R_{u,v} < 5$) // Points closer than 5m

$p_{u,v}.\Omega_{th^+} = \frac{0.04}{\delta\phi * \sqrt{R^2_{u,v-1} - H^2}}$; // Unevenness for a obstacle of height 4cm

$p_{u,v}.\Omega_{th^-} = \frac{-0.04}{\delta\phi * \sqrt{R^2_{u,v-1} - H^2}}$;

ELSE

$p_{u,v}.\Omega_{th^+} = 0.4$;

$p_{u,v}.\Omega_{th^-} = -0.2$;

END IF

IF ($p_{u,v}.\Omega_{th^-} \leq p_{u,v}.\Omega \leq p_{u,v}.\Omega_{th^+}$)

Return *false*;

ELSE

Return *True* ;

END IF

threshold at 0.4 ($\Omega_{th} = 0.4$). This threshold detects obstacles of $2cm$ at the closest range of $2.6m$ and $50cm$ at the farthest range of $56m$ when Velodyne HDL-32 is at height of $1.3m$ on a level ground. Due to $2cm$ being very small, sensor noise and the rough nature of ground near to sensor often results in detection of spurious obstacle points. In order to overcome this sensitivity, we set $4cm$ as the minimum obstacle height to be classified as an obstacle point. This comes at about $4.5m$ range of the $\Omega = 0.4$ curve. We depict this in the inset of plot in Figure 3.12 which is essentially a subplot for showing curves in the near ranges. The black line with asterisks is the final threshold line for obstacle detection in our experiments. As the obstacle interception height is not measurable, at near ranges ($R < 4.5m$) unevenness threshold Ω_{th} is calculated using Equation 3.18 with the value of b set at $4cm$. Henceforth we shall refer to the threshold unevenness for positive obstacles as Ω^+_{th}.

For negative obstacles, *i.e.*, depressions in the terrain, the same Equation 3.18 holds with the only difference that the value of b is now negative. The resulting unevenness will also be negative. However, unlike positive obstacles, negative obstacles are not so easily visible to the laser scanner because of occlusion by ground or other positive obstacles. So the scanner manages to acquire range data from only the shallow part of a depression. Because of this and the asymmetry as explained earlier with Figure 3.11, we set a lower magnitude of threshold for negative obstacles compared to positive obstacles. We set it at $\Omega_{th}^{-} = -0.2$ for detection of negative obstacles at longer ranges ($R_m > 5m$). At closer distance, a constant height threshold of $b_{th} = -4cm$ is maintained.

The unevenness Ω at a point, obtained using Equation 3.1 from measured ranges is compared against the unevenness thresholds Ω_{th}^{-} and Ω_{th}^{+} at the specified range for detecting obstacles. When the measured unevenness Ω lies outside these two thresholds, the point is classified as an obstacle; otherwise it is a non-obstacle point. In this way, obstacle points are detected in the radial direction from the range data at each azimuth θ. This is shown as function **radial_edge(u,v)** in Algorithm 3.1 .

3.5 Obstacle edge detection in transverse direction

Dense nature of range data along the transverse direction allows for obstacle detection along an individual ring based solely on unevenness that is calculated in the radial direction using the method described in the previous section. For example, a sudden change in unevenness value between two neighbouring points along the ring indicates an obstacle edge. However if an obstacle edge is aligned in the radial direction, **radial_edge** may miss this edge. In order to take care of such situations and also to provide a basis for aggregating traversable region using region growing, we additionally detect obstacles using range difference between neighbouring points along each ring. For this, a Transverse unevenness (Γ) is defined in the same way as radial unevenness (Ω). The estimated arc length δR_e over level ground between two neighbouring points (now transverse) separated by angular interval $\delta\theta$ is $R\delta\theta$. HDL-32 scanner rotating at $10Hz$ has $\delta\theta \approx 0.16°$. The measured arc length δR_m however will be approximately δR (see Figure 3.13), as $R\delta\theta$ is much smaller in comparison. We define transverse unevenness as

$$\Gamma = \frac{\delta R_m}{\delta R_e} \approx \frac{\delta R}{R\delta\theta} \tag{3.21}$$

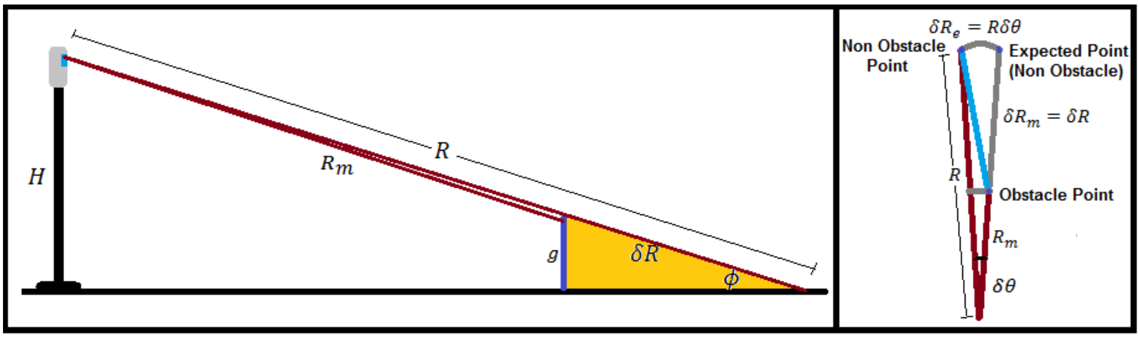


Figure 3.13. Neighbouring lasers in transverse direction, Left: Side view, Right: Illustration with top view.

In Figure 3.13, one laser hits ground with range R, while its immediate neighbour encounters obstacle at height g from perceived ground. Taking ratios of similar sides,

$$\frac{\delta R}{R} = \frac{g}{H} \tag{3.22}$$

Using Equation 3.22, Transverse unevenness threshold Γ_{th} for an edge height g_{th} is rewritten as

$$\Gamma_{th} = \frac{g_{th}}{H\delta\theta} \tag{3.23}$$

Algorithm 3.2:

Function: Detect edge in transverse direction, $\boldsymbol{transverse\_edge}\left((m,n),(m',n)\right)$

Input: Indices of two neighbouring points on a ring, transverse threshold height g_{th}

Output: Returns true if there is an edge between points $p_{m,n}$ and $p_{m',n}$

Calculate transverse unevenness Γ

$\Gamma = \frac{R_{m,n} - R_{m',n}}{R_{m,n} * \delta\theta}$;

Calculate transverse unevenness threshold Γ_{th}

$\Gamma_{Th} = \frac{g_{th}}{H\delta\theta}$;

IF ($|\Gamma| \leq \Gamma_{Th}$)

Return $False$;

ELSE

Return $True$;

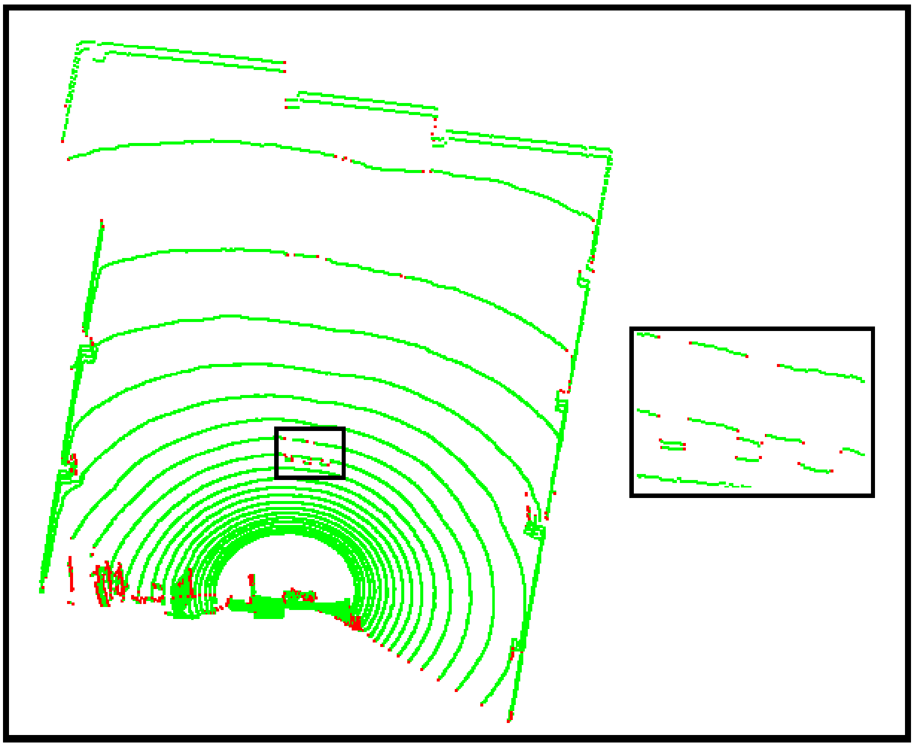


Figure 3.14. Obstacle detection in the transverse direction.

While Equation 3.21 computes transverse unevenness for a point with respect to its neighbour, Equation 3.23 gives threshold for obstacle edge detection given a height threshold g_{th}. For positive obstacles, transverse unevenness is negative, and it is positive for negative obstacles. Unevenness within thresholds ($|\Gamma| \leq \Gamma_{th}$) signifies no edge (function **transverse_edge** in Algorithm 3.2).

Edge detection in the transverse direction is shown in Figure 3.14 on the terrace scan with obstacle boxes by setting threshold $g_{th} = 4cm$. The transverse edge points are shown in red. The edges at boxes are highlighted in inset. The terrace discontinuities are also properly detected as edges. Even at places where the ring has missing data points, the edges are detected.

3.6 Traversability Map

Previous sections describe how individual points are classified into obstacle and non obstacle points. In order to plan a path towards the goal, the robot should obtain the regions that it can actually reach without colliding with obstacles. This is called the traversable region. There can be isolated regions in the environment that are locally traversable but not reachable from the robot's current position. Additionally, some locally planar obstacle surfaces are classified as non-obstacles, like steps in front of a building or planar surfaces (bonnet, roof) of vehicles which are actually not traversable. For determining the entire traversable region, all non-obstacle points starting from the current robot's position are identified and connected into a convenient graph-like data structure. The traversability map so formed in terms of connectivity graph can be used conveniently to plan a path for the robot [49].

One way to obtain the local traversability map is by performing a point level region growing [97] by connecting all the non-obstacle points starting from a traversable point in front of the robot. Neighbouring points in radial and transverse directions are connected until an obstacle edge is detected. The high number of data points (up to 70,000) present in a scan requires $90 - 160ms$ to compute the traversability map (graph) on an Intel i3, 32-bit, $3.2GHz$ processor with $4GB$ of RAM. Although this is quite efficient, it is not real-time considering that the Velodyne supplies a scan every $100ms$. Instead of growing region at the point level, we may grow region at the cell level to improve execution time. It also reduces the number of false (spurious) connections that arise at the point level because of sensor noise and imprecision.

Algorithm 3.3:

Function: Initializing and populating logical polar grid ***Initialize_Grid*** $(p_{u,v} \in P)$

Input: Set of data points P from Velodyne in (R, θ, ϕ) format for θ between 0° and 360°, and ϕ between -30.67° and +10.67° degree; Unevenness threshold Ω_{th}.

Output: Radial grid $c_{u,v} \in C$ of terrain in front of the robot with each cell containing a list of points belonging to it and marked as an obstacle or non-obstacle cell based on obstacle detection criteria.

```
Mark all cells c_{u,v} ∈ C as unoccupied;
u = 1;
FOR θ_u between 0° and 360°
    Read points p_{u,v} , for = 1 to 32 ;
  FOR v = 2 to 32 , do
    Get the indices of the cell C_{col,row} that the point P_{u,v} belongs to:
    col = floor(θ_u ) ; row = v ;        // floor returns a nearest lower integer for a decimal number.
    Add the point P_{u,v} to the list of points contained in cell C_{col,row} :
    list(C_{col,row}) ← P_{u,v};
    Mark cell C_{col,row} as occupied;
    IF (radial_edge(u,v) OR transverse_edge((u,v),(u − 1,v))OR transverse_edge((u,v),(u + 1,v)) )
        Mark the cell C_{col,row} as an obstacle cell;
    ELSEIF (C_{row,col} is NOT obstacle cell)
        Mark the cell C_{col,row} as non-obstacle cell;
    END IF
  END FOR
  u = u + 1 ;
END FOR
Return C;
```

Unlike the traditional polar grid [31, 32, 83, 84] where points are binned in both radial and transverse directions, we bin points only in the transverse direction (Algorithm 3.3). Given the density of Velodyne HDL-32 data, we bin points along each ring with each cell containing points within one degree of azimuth. In the radial direction, the number of cells is equal to the number of lasers (32 cells). A cell $C_{col,row}$ in the grid stores the list of points contained in *col* bin along the azimuth and the laser index $row = v$ contained in it. Based on the obstacle and edge detections in both radial and transverse directions, the cells containing these points are marked as obstacle or non-obstacle. A cell is classified an obstacle cell even when it contains a single obstacle point. All non-obstacle cells starting from a

Algorithm 3.4:
Function: Region growing by connecting traversable cells, ***Region_Grow***(m_0, n_0):
Input: Seed cell c_{m_0,n_0} to start growing traversable region.
Output: Traversable region grown from given seed point.

```
TQ = NULL;          // Initialize a queue TQ to contain indices of points to grow the traversable region
  PUSH (m_0, n_0) into TQ;
DO
  (m, n) = POP from TQ
 FOR i = -1 to 1
  FOR j = -1 to 1
      m' = m + i;
      n' = n + j;
    IF((abs(i) ≠ abs(j)) AND In_Grid(m', n') AND (c_{m',n'} is not checked) AND (c_{m',n'} NOT obstacle cell))
        c_{m',n'} ← traversable;
        c_{m',n'} ← checked;
        PUSH (c_{m',n'}) into TQ;
    END IF
  END FOR
 END FOR
UNTIL TQ = NULL
```

seed cell close to the robot ($C_{90,2}$ in our case) are then joined using region growing similar to [97], but now with cells. It finally provides all cells belonging to the traversable region. Region growing is performed in Algorithm 3.4 using 4 neighbour connect function **Region_Grow**, which connects neighbouring cells in both radial and transverse directions.

All the cells within the given unevenness and range discontinuity thresholds, and not otherwise marked as obstacle cells, are connected to form a traversable region if there is a physical connectivity to the initial seed cell. Once the traversability map is obtained, the points in the point cloud can be displayed according to the traversability attribute of the cells that they belong to. The traversability map, so formed, around the current position of the robot, provides the basis for planning onward motion of the robot towards its goal.

3.7 Results

For validation of methods for detection of obstacles using unevenness, experiments were first carried out placing Velodyne sensor on a stationary stand at a similar height to that when placed on the mobile robot ($1.3m$) and data is taken on the near level terrace of our lab building with the farthest boundary wall being $20m$ from the sensor. Data is then collected with the sensor on the mobile robot and from different locations in our campus with a variety of semi urban environments containing roads, slopes and lawns. The environments also contain moving objects like humans and traffic like buses and cars. As our robot moved at a low speed ($0.2m/s$) during our experiments, we have not made corrections for the vehicle's motion. These corrections can be done as a pre-processing step [86] for determination of exact location of the range points. The results are presented in two subsections; the initial part shows qualitative results for obstacle detection only in the radial direction using proposed method. In the second part, we show the traversability maps built from the data collected from a moving mobile robot. We also present a quantitative comparison of the results with the method by Chang et al. [85] applied to the same data.

3.7.1 Obstacle detection

Unevenness is first characterized by placing three boxes of known heights $10cm, 20cm$ and $30cm$ as obstacles on the terrace floor. Horizontal distances of these boxes are increased in steps of $1m$ from the sensor. This helped in understanding the nature of unevenness with distance in order to set appropriate thresholds for detecting obstacles. In the experiments, by tuning the unevenness threshold (Ω_{th}), the obstacles are detected at all locations when the lasers hit the obstacles.

Figure 3.15 shows the experimental setup for evaluating the obstacle detection method and the resulting scan after obstacle detection (bird's eye view). Results in this section show the non-obstacle and traversable points in green; obstacle points are shown in red. When the obstacles are placed at $5m$, a threshold of $\Omega_{th} = 0.4$ detects the boxes as obstacles, while most of the terrace is detected

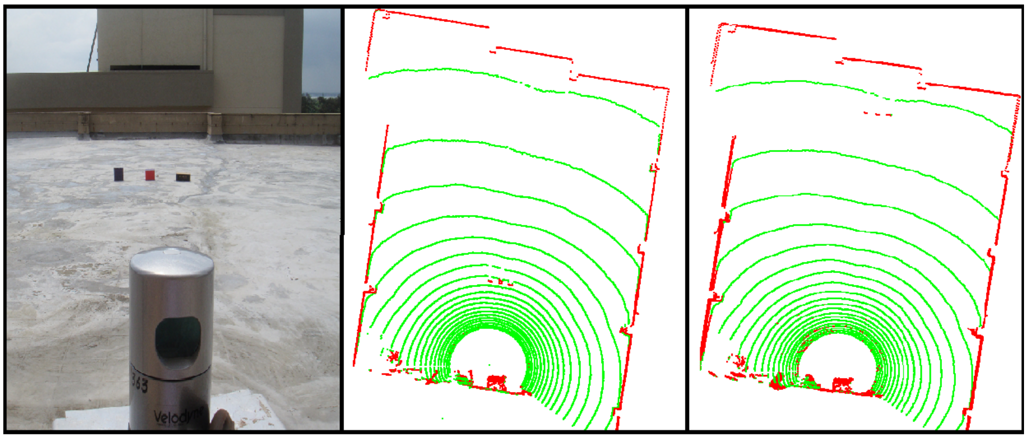


Figure 3.15. Terrace experiment; Left: Velodyne on a stationary stand with boxes as obstacles, Middle: $\Omega_{th} = 0.4$ for boxes placed at $5m$, Right: $\Omega_{th} = 0.2$ for boxes placed at $16m$.

as traversable. Walls surrounding the terrace and clutter are rightly detected as obstacles. When the boxes are placed further away at $16m$, boxes are detected as obstacles only on reducing the threshold to $\Omega_{th} = 0.2$. However, the reduced threshold caused many points near to the sensor to be detected as obstacle points. These spurious points can be seen as red dots on the inner circular rings. This shows the rationale in setting an additional criteria of minimum height threshold ($4cm$) before classifying the point as obstacle apart from the constant unevenness threshold of 0.4 ($\Omega_{th} = 0.4$).

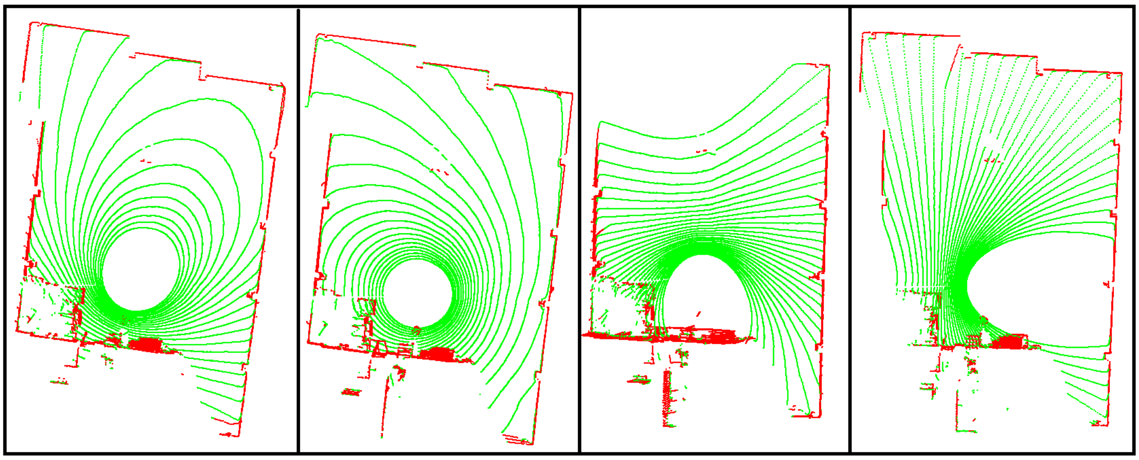


Figure 3.16. Obstacle detection with sensor subjected to tilts in different directions.

Insensitivity of unevenness based approach to overcome sensor tilts is shown by subjecting the sensor to tilts in different directions on the lab terrace with the obstacle boxes. This simulates pitch and roll experienced by a moving vehicle with the compression of rings in the direction of tilt. Figure 3.16 shows detection of boxes as obstacles in spite of the sensor being tilted in different directions. Repeatedly the boxes are clearly separated from the traversable planar region for different directions of sensor tilt. The two right most images show correct classification even for extreme cases where all the 32 lasers intercept the floor. This experiment validates the virtual sensor reorientation model based on the measured range as explained with Figure 3.5 for overcoming terrain slopes in addition to pitch and roll of the vehicle.

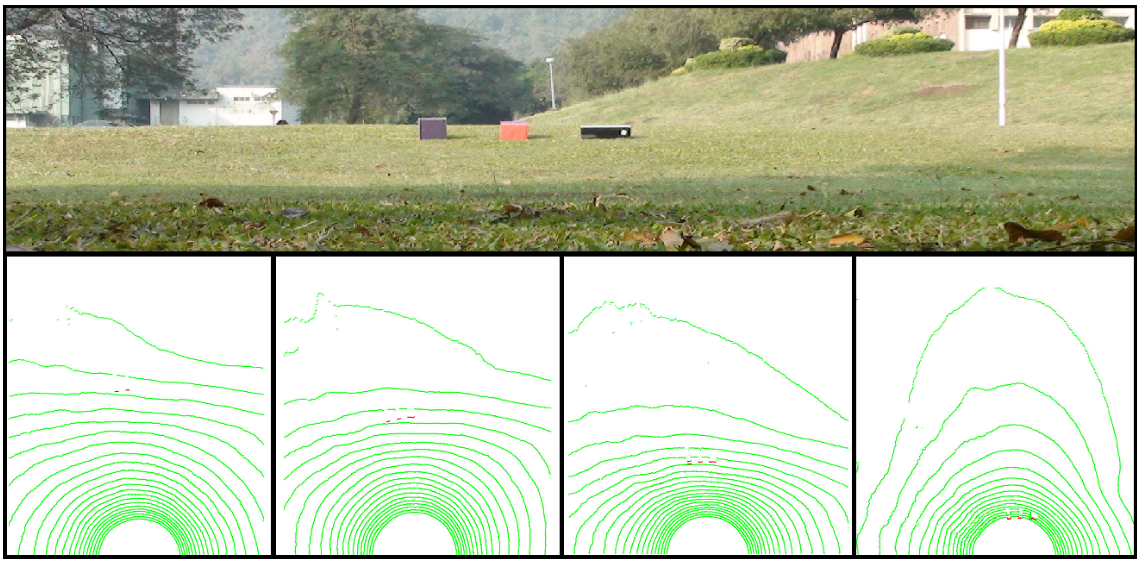


Figure 3.17. Top: Obstacle boxes on a slope in lawn, Bottom: Part of the scans showing obstacles detected when the robot is in motion towards the boxes (Left to Right).

Figure 3.17 shows detection with obstacle boxes present on a slope in a garden lawn. Boxes are repeatedly detected while the slope is shown as traversable from a moving robot from a distance of $20m$ and towards the boxes. Here the threshold is increased to $\Omega_{th} = 0.6$ to accommodate the rough nature of lawn comprising grass blades and leaves. In spite of the shape of the rings at the top getting disturbed (adapting) according to the slope, the terrain is detected as traversable unless there is a steep portion within the slope.

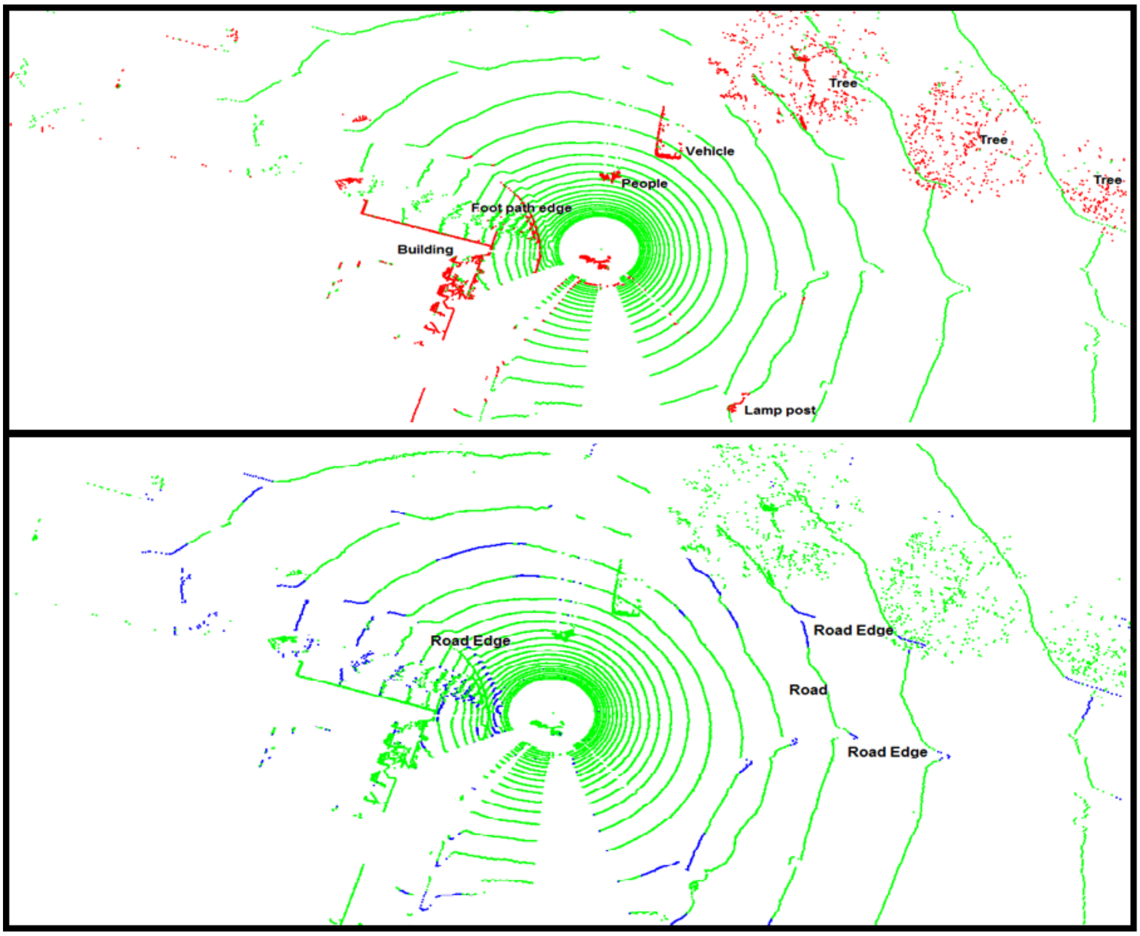


Figure 3.18. Part of a scan showing positive obstacles in Red (Top) and negative edges Blue (Bottom).

In addition to the positive edges, Unevenness also detects the difficult to detect negative obstacles. In Figure 3.18, the bottom image shows road edges detected as negative edges, while the footpath is detected as a positive edge (Top image). The high sensitivity of unevenness allows it to detect small undulations such as road edges. People present in the scan are detected as positive obstacles. A set of positive obstacle edges are seen clearly distinguished from the ground with each ring being a positive edge to the preceding ring.

3.7.2 Traversability

Traversability of a region is obtained by connecting the non-obstacle cells starting from the position of the vehicle. Region growing removes locally planar regions which are not reachable by the robot from its current position. For example, in Figure 3.19, the points sampling the vehicle's bonnet are initially detected as non-obstacles. On region growing, this region is identified as not reachable (points shown in black in the right image). Similarly, planar regions like footpaths can be shown as not reachable or non-traversable regions with region growing when there is no connecting path.

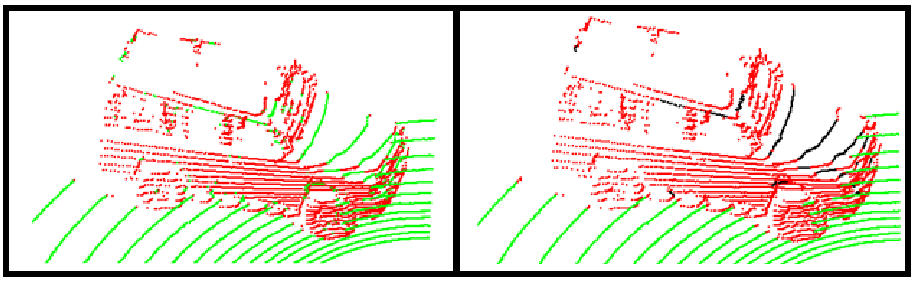


Figure 3.19. Vehicle scan, Left: Without region growing, Right: With Region growing.

First, the images of resulting scans representing different environments are qualitatively presented and evaluated in Figure 3.20 and Figure 3.20. Quantitative results for the same scans are presented in Table 3.2. The variety of environments included flat roads, slopes and lawns. The environments also contained dynamic obstacles like humans and vehicles. The results are compared to the results obtained by a method presented in Algorithm 3.5. For comparison, only the data ahead of the robot is considered where traversability map is built with 180×32 cells.

Algorithm 3.5 describes a method for comparison by Chang et al.[85],which also detects obstacles column wise in a range image starting from the bottom pixel which is near the ground. This method is similar to our method in processing the scan. A point is considered an obstacle point when either the slope from the

Algorithm 3.5:

Obstacle detection using height difference and slope along a column using Chang's method, detects obstacles along the radial direction at a given azimuth with a set of 32 lasers fired at an azimuth.

Input: Point set $q_v(x_v, y_v, z_v)$ belonging to a set of lasers fired on an azimuth angle, where v: 1 *to* 32

Slope threshold β in terms of angle of slope, Height threshold γ.

Output: Ground and obstacle points

```
Ground point q_g(x_g, y_g, z_g) = q_1 ;
FOR v = 2 to 32
  IF slope (z_v-z_g)^2 / ((x_v-x_g)^2+(y_v-y_g)^2+(z_v-z_g)^2) ≥ sin^2 β
    q_v ← obstacle ;
  ELSE | z_v − z_g | ≥ γ
    q_v ← obstacle;
  ELSE
  q_g = q_v ;
  ENDIF
ENDFOR
```

previously detected ground point is greater than the slope threshold or when the height difference to ground is greater than the specified obstacle height threshold. We carry out a quantitative comparison of our method with this method applied on the same environments by setting a height difference threshold γ as $4cm$(same as ours) and the slope threshold β as $25°$ (set quite high to make it more tolerant to slopes). The results reported by Chang et al.[85] however set a height threshold of $1m$, which may be acceptable in secluded outdoor environments for detecting large objects faraway, but for urban environment, such a high threshold fails to detect most obstacles. A smaller threshold on height on the other hand leads to most of the points away from the sensor to be detected as obstacles because of the increasing inter-ring distance between the points. This could probably be overcome by setting a range based threshold. Our method overcomes this problem in spite of sparse data by using a constant threshold based on unevenness and not on either height or slope.

Quantitative results are presented for the same scans (Figure 3.20 and Figure 3.21)based on manually comparing the scans with ground truth. False clas-

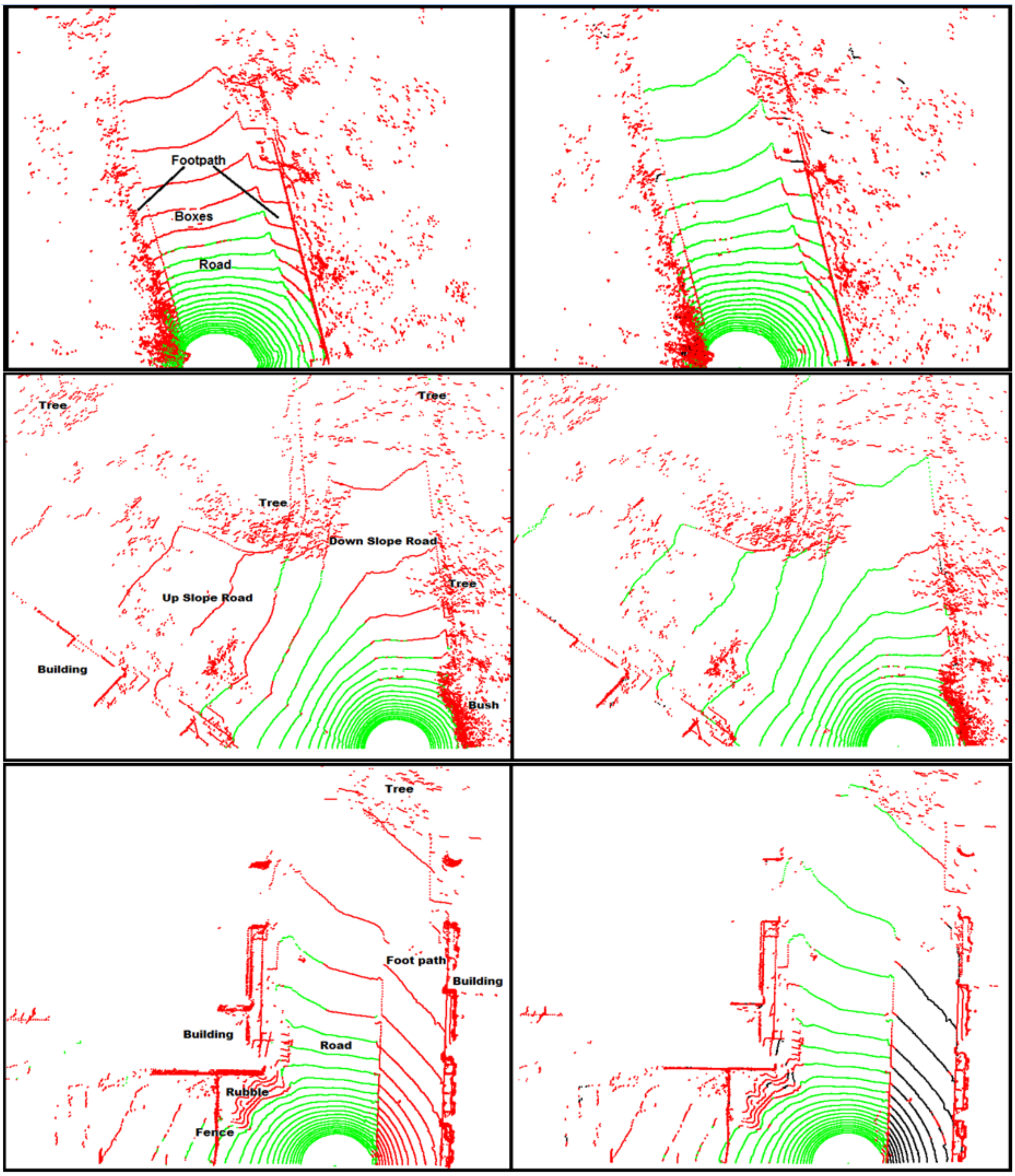


Figure 3.20. Obstacle detection, Left: Chang's method, Right: Proposed method, Top to bottom environments: Upslope, Down slope, Flat road. (Green for traversable, Red for obstacle and Black for not reachable flat regions).

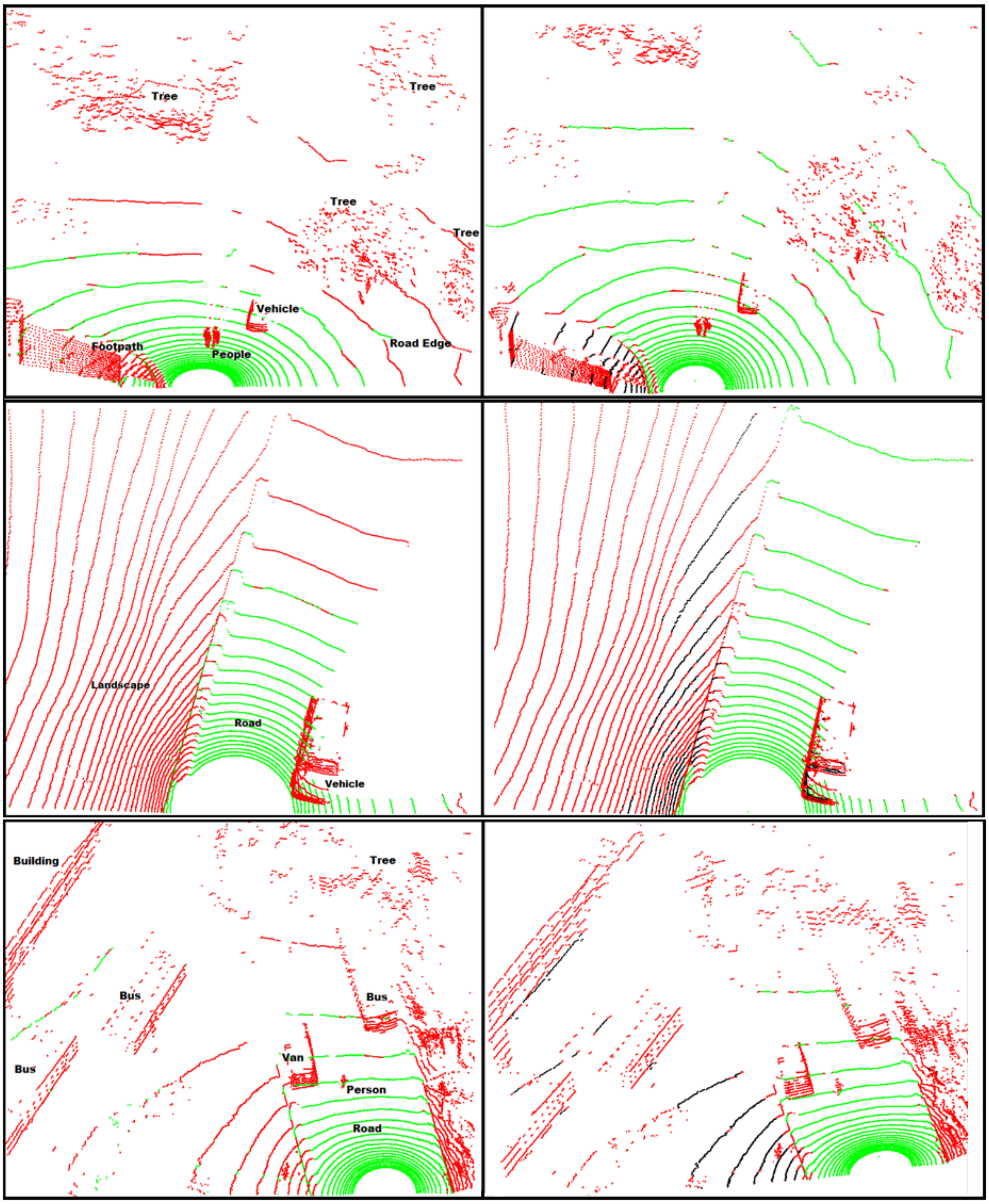


Figure 3.21. Obstacle detection, Left: Chang's method, Right: Proposed method, Top to bottom environments: Road junction, Vehicle descending a slope, parked vehicles (Green for traversable, Red for obstacle and Black for not reachable regions).

Table 3.2: Comparison of false detections, M1: Chang's Method, M2: Proposed Method.

	False Positives(FP)		False Negatives(FN)	
	M1	M2	M1	M2
Scan 1(Up Slope)	37	1	4	9
Scan 2(Down Slope)	66	7	4	0
Scan 3(Flat Road)	20	3	14	6
Scan 4(Road Junction)	31	1	30	70*
Scan 5(Vehicle Descending a slope)	31	0	4	4
Scan 6(Parking)	4	2	26	0

sification of cells can be either because of false positives or false negatives, where the former detects actual traversable region as obstacle and the latter detects an obstacle region as traversable. False negatives are to be avoided at all cost as it may result in the robot misinterpreting obstacle regions as traversable which could harm the robot. For ease of quantitative comparison, instead of measuring ground truth at point level, a rectangular metric grid is built with each cell covering an area of $1m \times 1m$. The points obtained in the polar grid are then superimposed onto the rectangular grid. The false positives and the false negatives are marked by the human operator based on ground truth. A graphical user interface (GUI) is developed for this purpose, where the user can point out cells which are wrongly classified. Even a single wrong point in the cell is punished by classifying the cell as false classification.

From Table 3.2, it can be concluded that the proposed method gives less number of wrong classifications both in terms of False Positives and False Negatives. It performs well particularly when the environments contain slopes while giving comparable results for scans in other environments. The number of false detections is more in scan 1 because of a small region that got connected to the footpath. In scan 4, the number of false positives (indicated by *) is high for both the methods but more for our method; this is because the lawn beyond the road is almost at the same height as the road and the road boundary is not intercepted by any of the lasers. This causes that region to be shown as traversable by both the methods, but because of the smaller height difference threshold most points in method 1 get detected as obstacles.

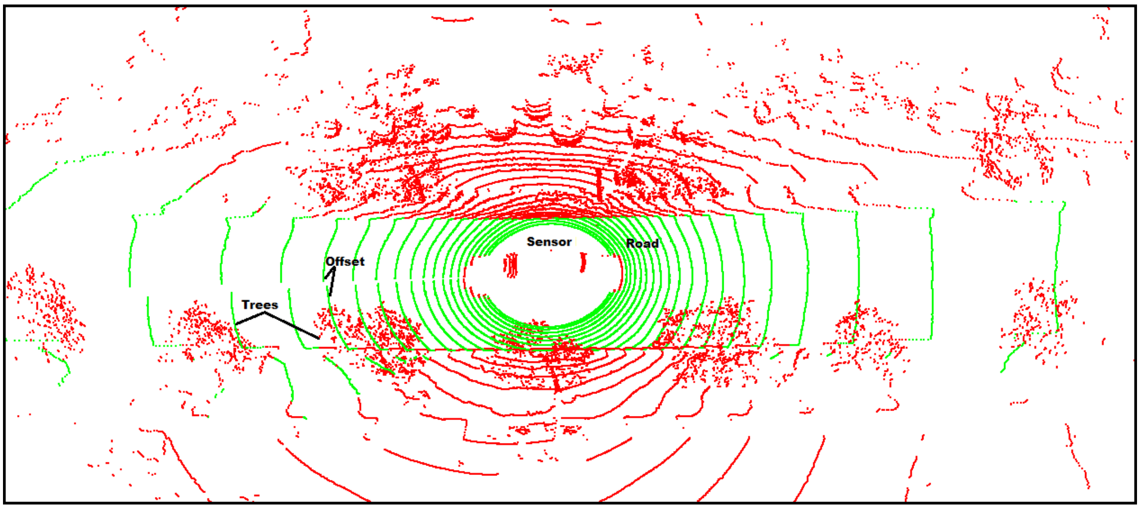


Figure 3.22. Traversability for a fast moving vehicle.

In Figure 3.22, we test the method with data from a fast moving vehicle. This is an externally recorded data, where the sensor height H is arrived at $1.8m$ after trying height thresholds for ground. Here again the road is correctly classified as traversable with regions beyond the road edges including trees as obstacles. Few points falling on the vehicle itself as obstacles can be safely ignored. An offset is observed on individual rings between the start and end of frame data (behind the vehicle to the left). These offsets are to be corrected as in Steinhauser et al.[86]. However, the method classifies the points correctly because the detection of obstacles is on the basis of range data from neighbouring points, which are acquired in quick succession and processed immediately. Detection of obstacle nature of points (Radial and Transverse) and population of 180×32 grid is executed within $15 - 20ms$, while region growing of traversable region took about $5 - 10ms$ on an Intel i3, $32 - bit$, $3.2GHz$ processor with $4GB$ of RAM.

3.7.3 Traversability using Sick Scanner

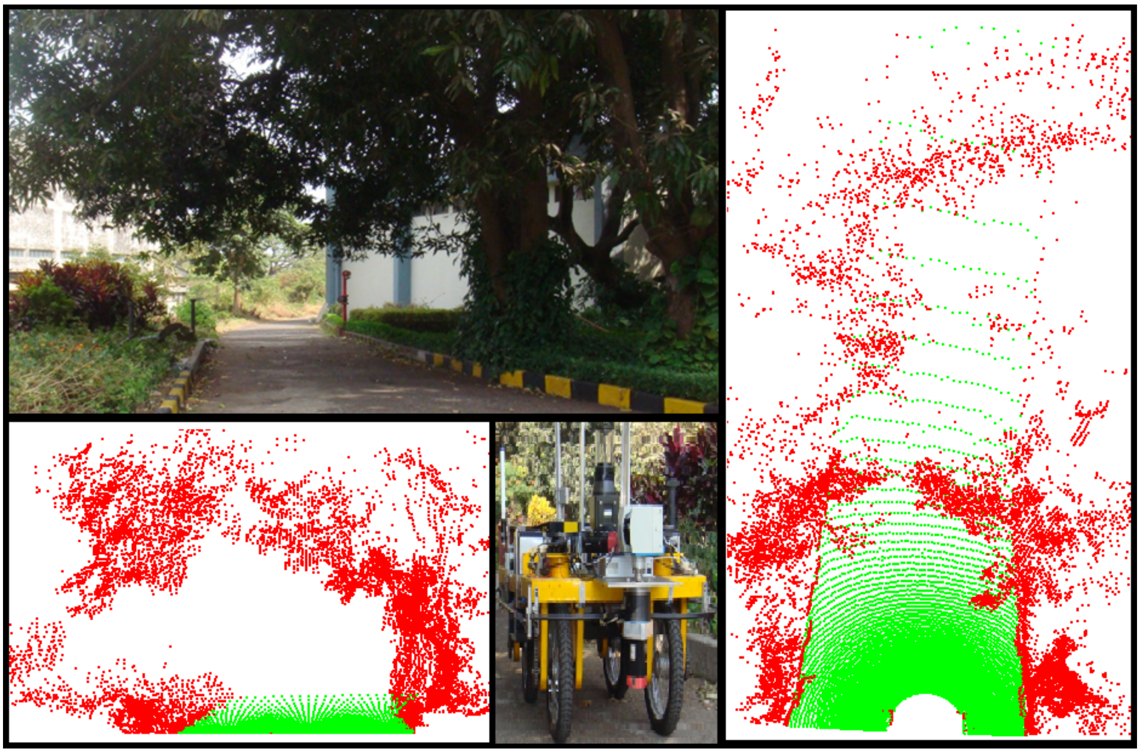


Figure 3.23. Clockwise starting from top left: Environment, Top view of scan, Arrangement of a Rotating Sick Scanner on our Robot, Front view of the scan.

The proposed unevenness method is not limited to Velodyne scanners alone. Experiments are conducted with an outdoor Sick Laser scanner to show the adaptability of the method with other sensors. The Sick scanner is placed at a height of $0.6m$ above ground as shown in Equation 3.23. The scanner is rotated in yaw direction [64] in the stop-scan-go fashion where the scan is taken for each rotation of $1°$ by the motor. Instead of 32 laser readings in the vertical direction for the Velodyne HDL-32, there are 180 readings with ϕ between $-45° and +45°$ ($v = 180$). The pitch angle difference between the consecutive points is $0.5°(\delta\phi)$. Here the difference between pitch angles ($\delta\phi$) being smaller than Velodyne HDL-32, the threshold for detection is set higher at $\Omega_{th} = 0.6$. It can be seen in Equation 3.23 that only the pavement on which the robot is located is properly detected as traversable. The tree tops are shown as obstacles even if they are above the traversable region. The small regions near to the robot in the scan shown in red colour are because of the laser points hitting the plate under the scanner; otherwise there is clear distinction between the traversable and obstacle regions. For clarity, both the front and the top views are shown.

3.8 Conclusion

Conceptually and computationally the measure of unevenness is simple and efficient. Unevenness computed over all sampled points form an unevenness field around the robot. Traversable region is obtained as a connectivity graph in real time on a standard desktop computer. For this purpose, points in the transverse direction are binned into a non-homogeneous (polar) grid where cells grow bigger with distance from the sensor. Results show the effective and efficient detection of both the obstacles and traversable regions.

Unevenness is robust against small sensor tilts during locomotion, and it works well on slopes. This is demonstrated with experimental results against a standard method [85] that uses both height and slope between neighboring range points to detect obstacles. Unevenness out-performs the compared method particularly on and around slopes.

Unevenness is characterized through analysis to arrive at a reasonable policy for setting thresholds for detecting obstacles. Thresholds are influenced primarily by the difference in pitch angle ($\delta\phi$) between neighbouring laser beams and are needed to be adjusted accordingly depending on $\delta\phi$ for different sensors. Thresholds should also be tuned depending on the nature of operating terrain (surfaced road/off-road/lawn etc.), wheel dimension and configuration of the mobile robot. But once set, it works well over all ranges by detecting the regions that should better be avoided during navigation.

Since our mobile robot moved at a low speed of $0.2m/s$, the results are without corrections for motion; the points are classified using the neighbouring ranges. Given the quick nature of scan, this does not affect point classification in any manner. However, motion corrections [86] are needed when the points are mapped into the world coordinate system.

Unevenness is shown to bring out clear boundaries between objects. The obstacles thus detected, including the smallest of features, can be used in the advanced stages of navigation like segmentation, registration and localization which are the subjects of the subsequent chapters.

Chapter 4

Segmentation using Unevenness

In addition to detection of traversable region surrounding a mobile robot, autonomous mobile robots are required to robustly identify different discernible objects that are present in its environment. These individual objects when properly identified can be used as features in the advanced stages of robot navigation. For example, points belonging to important static features can be selected for registration. SLAM algorithms can use these robust features as landmarks. By Detection and Tracking of Moving Objects (DATMO), robots can do motion planning by estimating the movement of fellow traffic participants. In addition, semantic labelling of segmented objects (cars, cyclists, and pedestrians) helps in understanding their generic behaviour. Obstacle detection using unevenness, which depends on the range difference between the neighbouring points, has been described in detail in the previous chapter. Scan is segmented into distinct objects by again using the measured range and the expected range difference between the consecutive points in terms of unevenness (Ω). Using this approach, important discernible features present in the robot's environment are segmented so that they can be used to advantage for robot navigation.

Borrowing the definition of segmentation from the Image processing field

[98], segmentation of a laser scan is considered as partitioning of scan into set of non-overlapping regions whose union is the entire scan. The extent of segmentation however depends on the nature of application. For example, at a coarse level, segmentation is restricted to ground and obstacles, while at a finer level the entire scene are segmented into different objects. An object can be further segmented within itself, like a human figure getting segmented to show individual body parts. For robot navigation, apart from identifying the traversable ground, it is sufficient to segment important individual objects to be used as either landmarks or moving objects. For example, people or vehicles identified as segments are either avoided or tracked. In general, it is sufficient for a robot to segment robust landmarks and moving objects in addition to ground.

Segmentation in general, and for outdoor environments in particular, suffers from both under and over segmentation. In the former, a segment has more points than those sampling an object, and in the later, a single object gets decomposed into multiple segments. Because of its complex nature, segmentation in outdoor environment is subjected to both over and under-segmentation. These add to the difficulties in developing methods for perfect segmentation. Otherwise an Ideal segmentation of point cloud could be defined using the following conditions.

Partition of a scan $\boldsymbol{S}$ into subsets S_i , $i = 1, \ldots, m$, such that

Complete: Scan $\boldsymbol{S} = \bigcup S_i$, $i = 1, \ldots, m$

Disjoint subsets: $S_i \cap S_j = \emptyset, \forall\, i \neq j$

In this work, segmentation is again performed on ordered range data from a Velodyne laser scanner. The cloud, being ordered, encodes the neighbourhood information in the scan, which aids in segmentation without resorting to computationally expensive search for nearest points. Results on segmentation are shown with data collected from our outdoor robot while moving in and around our campus.

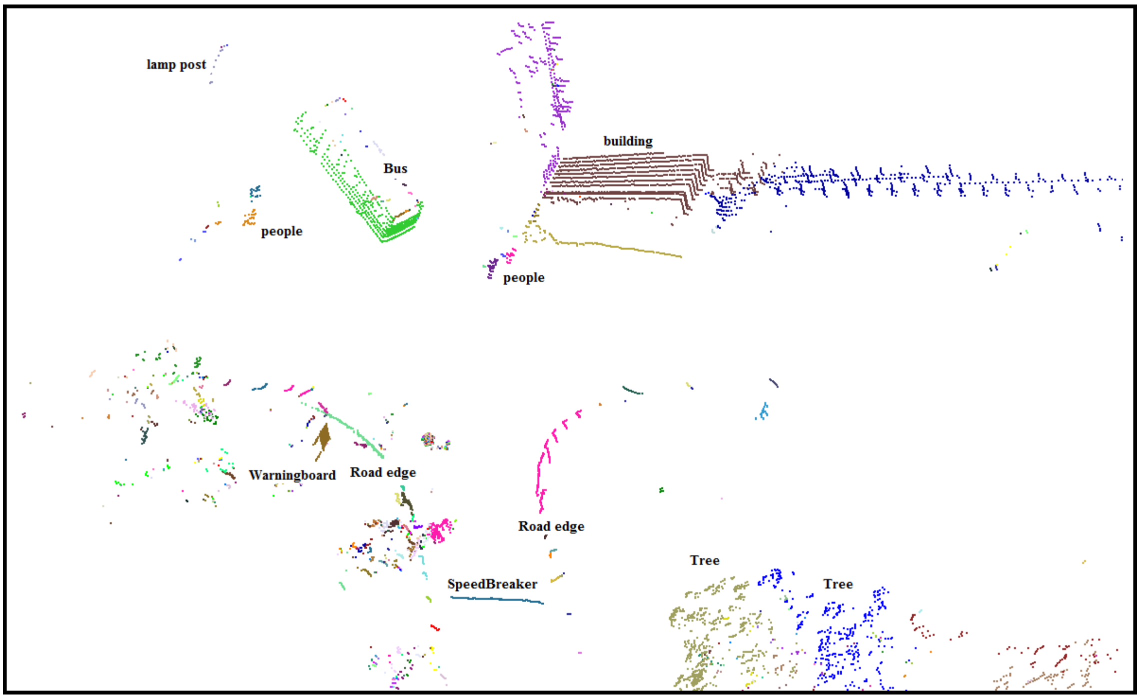


Figure 4.1. Segmentation showing individual features after removing ground

Figure 4.1 shows segmentation of a single scan after removing the ground segment. Even when there is some amount of over-segmentation, several objects and features are distinctly segmented. It can be noted that the speed breaker which in general is traversable is also segmented. This can be used as a feature for navigation.

In the commonly used grid based methods, it is difficult to set the size of grid cells. This sometimes causes close objects to be segmented together. Segmentation using unevenness (Ω) overcomes this problem by operating at a point level, resulting in clear separation between objects. For example, a person walking on ground will be clearly segmented out of ground, instead of few ground points which belong to the cell getting segmented along with the person. As will be shown in the results section, this approach also separates very close objects. Also, the extent of control that can be exercised in segmentation by varying the thresholds based on unevenness (Ω) is presented. Unevenness is assigned as an attribute to each point and region growing is performed to bring out discernible segments present in the robot's environment.

4.1 State of the Art

Segmentation of point clouds has been an active research area in robot navigation and computer vision. A few algorithms dealing with obstacle segmentation are model based and try to fit features, e.g., shapes such as cylinders [88], planes [89] or any other predominant feature likely to be present in the environment. But it is difficult to model the variety of objects encountered in the outdoor environments. Model based algorithms are also computationally expensive and are difficult to implement in real time.

Segmentation algorithms using point clouds can also be classified based on projection of the data into either Grid based or Range based methods. Grid based methods are commonly used where in points in a scan are first discretized into grid cells and then segmented. Grids can be either 2D grids or 3D. 2D representation is less accurate considering the elevation changes in the outdoor terrain as all the points are projected to a plane. This causes all the points vertically above a grid cell to segment together. 3D grids are usually discretized into 3D voxels [99] before clustering them into segments. They consume lot of memory and are difficult to implement in real time.

Several methods dealing with segmentation prefer to remove ground portion in a scan before carrying out the clustering of non-ground points [32, 79, 86, 100]. Ground that acts as a physical link connecting different objects is removed in the first stage of segmentation to facilitate further segmentation of non-ground points. Steinhäuser et al.[86] fit lines in the radial direction of scan for obtaining the traversable ground region and then segments the non-ground points by clustering them using region growing. Himmelsbach et al.[32] improves over his earlier method[100] by segmenting the ground first by fitting lines to the ground points by using a circular grid. Remaining non-ground points are segmented using fast connected component analysis. Further refinement of segments is done by looking at vertical displacement between the points belonging to the same cell. This is claimed to separate vehicles under the trees. The results are reported to be real time.

Doullaird et. al. [99] presents the pipeline for segmentation of point clouds for robotic applications using the voxel grids. After the scan is represented using voxels, the ground region is segmented first before clustering the remaining non-ground voxels into objects. Doullaird et al.[79] further present algorithms for segmentation of both dense and sparse point cloud data. Ground segmentation for dense data is carried out by clustering together the adjacent voxels based on their vertical means and variances. The largest portion found by this method is the ground segment. Non-ground objects are then partitioned using local adjacencies. In addition, dissimilar voxels below the flat voxels are merged into a segment. For example, the flat portion of a car roof is merged with the non-flat voxels below it. In addition to grids, ground can be modelled using either mesh or Gaussian Processes. A terrain mesh is built for separating ground by calculating terrain gradients for dense data or by using an iterative Gaussian Process Incremental Sample Consensus (GP-INSAC) for sparse data.

Grounds sampled by sparse data sensors may require interpolation for bridging gaps in the data. Ground can be detected for sparse data using Gaussian Processes (GP) regression [90], which is an iterative approach to a probabilistic method that can be applied across multiple scans from the sensor for continuous ground surface estimation. Gaussian Process methods separate the ground by considering ground points as inliers and the objects and clutter as outliers. In GP-INSAC[79, 83, 91], deterministic iterations are performed by progressively fitting the model (ground) from a single seed of high quality inliers and not by iterating over randomly selected seeds. Mesh based methods on the other hand are applied on a single scan with the mesh built between the points to model the terrain and the obstacles. Douillard et al. [79] builds a terrain mesh and separates ground by calculating terrain gradients or by using an iterative GP-INSAC for sparse data. Remaining non-ground points are clustered using voxel adjacency. Several methods of segmentation for various types of point clouds (sparse and dense) are presented and compared and reported for near real-time performance.

Korchev et al.[84] performs real time segmentation using min-map, max-map and difference-map and then does the 8-connected component analysis and compares the performance using both the rectangular and the polar grids. However, grid based methods suffer from under-segmentation. Other way of projecting the data is with respect to the rotation of the laser scanner. A range image is thus formed with each cell containing the range with implicit neighbourhood information. Normal intensity image segmentation algorithms from image processing field can thus be adapted to range images. Hoover et al.[101] does an experimental comparison of the methods dealing with the range image segmentation. His method involves fitting of a local surface and then carrying out clustering.

Unlike the Grid methods that are also applied to unordered clouds, there are several methods [43, 86, 93, 102] which work using the point's neighbourhood information. Information on the neighbourhood helps in quickly building the attributes for a point which help towards segmentation and region growing. Normal vectors can be computed [43, 102] for a point without resorting to computationally expensive plane fitting or sum of least squares or RANSAC. Klasing et al.[103] use Radially Bound Nearest Neighbours (RBNN) for clustering segments using Euclidean distance. Real-time performance is achieved because points already attached to clusters are excluded when searching for neighbours. Clusters are merged based on presence of a common point with radial distance within threshold. Klasing et al.[102] further improves RBNN by calculating local normals with the incoming data and clustering using Euclidean and angular distances. Ioannou et al. [42] uses difference between normals obtained from large and small support radius, segmenting even smaller features like kerbs and windows. Implementation uses PCL libraries [104].

It is easier to build graphs with neighbourhood information and carry out graph based segmentation. Moosmann et al.[43] use local convexity as an attribute in combination with vertical structures for obtaining segmentation and reports the performance to be near real time. Other attributes like edges [86], curvature [105, 106], smoothness constraint [107] or surface direction [108] can also be used for building graphs. Golovinskiy and Funkhouser [109] build 3D graph, and by using K-nearest neighbours (KNN) algorithm, segment objects while penalizing weak connections between object and background with min-cut. However, the method requires information on object locations. Angulov et al.[93] performs segmentation using machine learning techniques. Points in the cloud are segmented by supervised learning methods using Markov Random Fields (MRF). As this approach involves training and labelling, this is also not real time.

In contrast to the methods described above, terrain unevenness is a simple measure used for segmentation without resorting to any kind of model fitting. Apart from being simple and efficient, the presented method is robust to terrain slopes and robot's roll and pitch.

4.2 Segmentation

Unevenness, the way it is defined in the previous chapter, captures the smallest of surface discontinuities present in the scan. Considering virtual sensor reorientation (Figure 3.5 in chapter 3), points belonging to a continuous surface have similar unevenness. Change in unevenness indicates surface discontinuity in terms of gap in the terrain or change in surface orientation. Unevenness can thus be used with advantage for segmenting the point cloud into different objects. Given the nature of ordered scan, segmentation can be done using standard region growing, stopped only by detection of edges using difference in unevenness. A näve way of detecting edges using unevenness is by using the difference between unevenness values of the neighbouring points. When the difference is greater than a threshold, it signifies an edge where the region growing must stop.

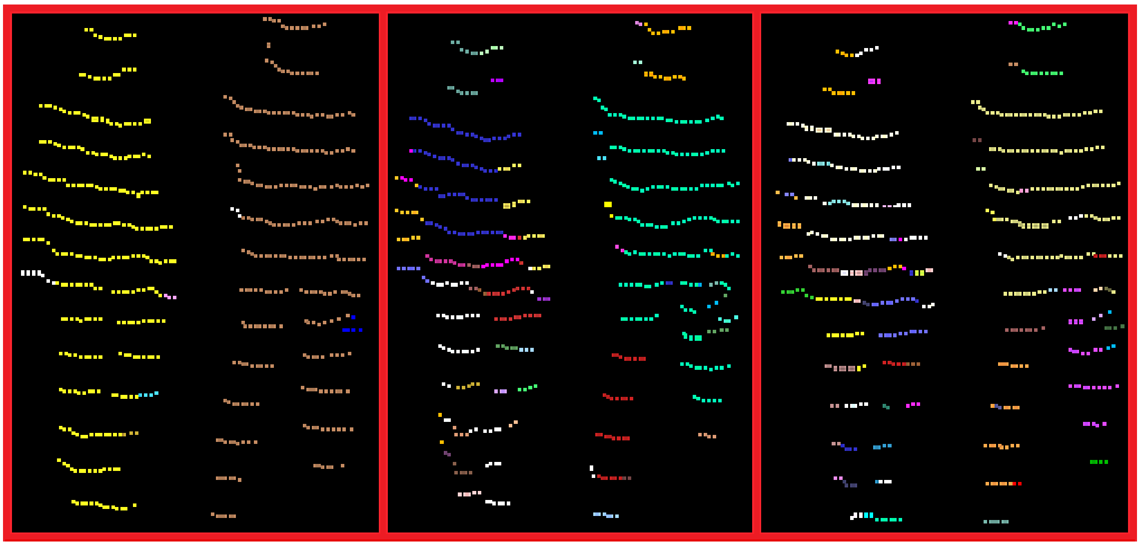


Figure 4.2. Surface segmentation by region growing using difference in unevenness ($\delta\Omega$).

Figure 4.2 shows surface segmentation with region growing using different threshold values of difference in unevenness ($\delta\Omega_{th}$). Surface discontinuity is detected when the difference in unevenness ($\delta\Omega$) between neighbouring points exceed these thresholds. A threshold of $\delta\Omega_{th} = 0.6$ segments two persons walking close by as individual segments. Only few noisy segments are present. On reducing threshold to $\delta\Omega_{th} = 0.06$, different body surfaces (hands, heads, thighs, shin, and heel) appear as different segments. Further reduction $\delta\Omega_{th} = 0.04$ results in over-segmentation bringing out even the irregular cloth surfaces and body curvatures. This shows sensitivity of unevenness in bringing out surface discontinuities. Smaller thresholds also result in over-segmentation of ground which is because of unevenness decreasing with range (analyzed in the previous chapter). Given the robot navigation problem, instead of segmenting surfaces, it is adequate to segment objects as individual segments. The basis of our approach is seen from Figure 4.3, where points either belong to ground with smaller unevenness, or to objects with unevenness close to 1. The overall segmentation approach is detailed in Algorithm 4.1.

Algorithm 4.1:
Steps for segmentation:

1. Acquire a single frame of range data from a rotating laser scanner.
2. Segment *Ground* using unevenness and remove *Ground_Points* from the scan.
3. Segment *Nonground_Points* into individual objects by using region growing with appropriate stopping criteria.
4. Filter out segments which contain fewer numbers of points.
5. Display *Point_Cloud* with important segments.

Using unevenness as point attribute, segmentation is performed in two stages. First, points belonging to traversable ground region are detected and removed before segmenting non-ground points. Points of an object are connected if they have unevenness around 1 using 4-neighbourhood connectivity. Large obstacles in general are vertical with slopes α around 90°. This results in unevenness values close to 1 because of the ranges of two consecutive points being nearly equal. But with irregular obstacles, this unevenness varies around 1. To accommodate for little convexities and concavities present in obstacles, limiting thresholds around 1 are set during region growing. Growth is also restricted with edges detected in the transverse direction using unevenness.

Figure 4.3 shows a representative diagram wherein a robot fires laser beams in a vertical plane onto a tree. Unevenness (Ω) values are indicated at each point. Starting from nearest laser (Point A) and proceeding outwards, points till E are on or near ground having unevenness near 0 because the expected and measured ranges at these points are similar. Point C samples a small bump resulting in a small positive unevenness ($\Omega > 0$). Due to decreased range at C, the measured range at point D, even on ground, exceeds the expected range (D' in inset) resulting in unevenness (Ω)to have a small negative value. Points from F to K sample the

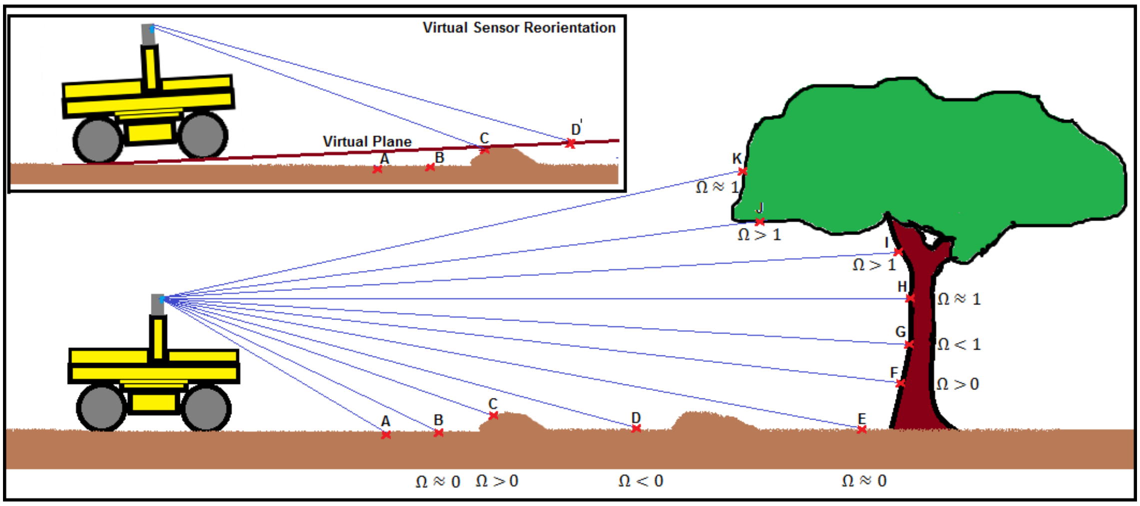


Figure 4.3. Robot and its environment showing unevenness(Ω) values for different points, Inset: Virtual sensor reorientation.

tree with unevenness (Ω) around 1, because ranges of adjacent points on faraway objects are similar. Depending on higher laser being closer or farther from the lower laser, unevenness (Ω) is greater or less than 1. In summary it can be observed that the points sampling the ground have unevenness near to 0 while points sampling the relevant obstacles have unevenness close to 1.

Algorithm 4.2:
Function: Detect the nature of point, ***detect_ground***( )
Input: Point indices u and v.
Output: Classifies $p_{u,v}$ as either *Ground* or *Obstacle* point based on $p_{u,v-1}$.

FOR $u = 1$ *to* $u_{\max}$
  FOR $v = 2$ *to* $v_{\max}$
    Obtain pitch angle difference between radial neighbours
      $\delta\phi = \phi_{u,v} - \phi_{u,v-1}$;
    Obtain the expected range difference between point $p_{u,v}$ and point $p_{u,v-1}$
      $\delta R_e = -\frac{R_{u,v-1}\sqrt{R_{u,v-1}^2 - H^2}}{H} * \delta\phi$;
    Measure the range difference between point $p_{u,v}$ and point $p_{u,v-1}$
      $\delta R_m = R_{u,v} - R_{u,v-1}$;
    Obtain the radial unevenness for point $p_{u,v}$
      $p_{u,v}.\Omega = 1 - (\frac{\delta R_m}{\delta R_e})$;
    IF ($P_{u,v}.\Omega \le 0.4$ *OR* $b \le 4cm$)
      $p_{u,v} \leftarrow$ *Ground;*
    ELSE
      $p_{u,v} \leftarrow$ *Obstacle;*
  END FOR
END FOR

In order to accelerate the entire segmentation, we detect and remove ground in a slightly different fashion without using region growing like in the previous chapter. At each rotation angle, at which the lasers are fired, range points are processed starting from the innermost or closest. Each range point is classified as either ground or non-ground using the unevenness value. However, a minimum height threshold of $4cm$ is maintained for classifying an object as obstacle. Detection of ground points is described as function **detect_ground( )** in Algorithm 4.2.

4.2.1 Segmentation of Non-ground points

Points after removing Ground are further segmented into distinct features or objects using region growing with 4 point neighbourhood. In Figure 4.4, the unevenness (Ω) curves are plotted across pitch angles (ϕ) to understand how unevenness varies with different obstacle slopes (α) using the relationship as in equation Equation 3.11 that is derived in the previous chapter.

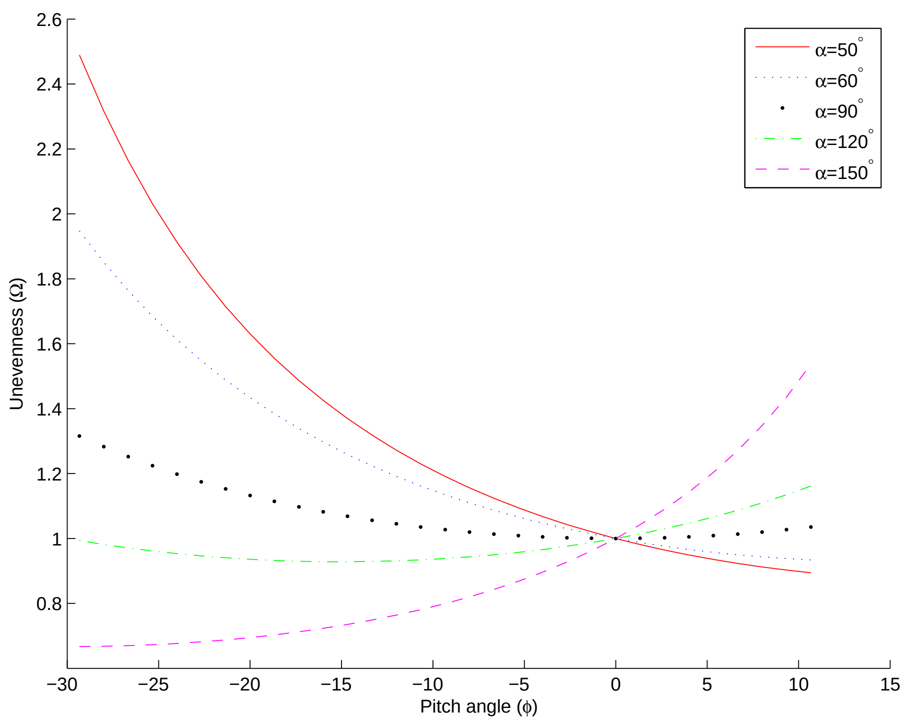


Figure 4.4. Unevenness values for sloped obstacles with pitch angles (ϕ).

This time unevenness is plotted against pitch angles (ϕ) and not range, because obstacles can encounter a laser beam at different ranges. Unevenness (Ω) is seen converging to 1 at 0° pitch angle ($\phi = 0°$) for all slopes. This is the angle at which obstacles are intercepted at sensor's eye level. Around this point, unevenness is around 1 with degree of deviation depending on the amount of slope deviating from 90°. Laser beams around eyelevel ($-10° \leq +10°$)usually are the ones encountering vertical obstacles. This is the region where 90° slope ($\alpha = 90°$)curve is around 1. This provides justification for using unevenness around 1 for growing an object.

Algorithm 4.3:
Function: Region growing by connecting object points, ***Grow_Object***(p_{m_0,n_0})
Input: Seed point for growing obstacle segment p_{m_0,n_0} , Object unevenness limits e_1 and e_2
Output: Points in Obstacle segment L_{obj}

```
L_obj = NULL;        //List of points belonging to object
TQ = NULL;           //Initialize a queue TQ to contain indices of points to grow traversable region
PUSH (m_0,n_0) into TQ;

DO
  (m,n) = POP from TQ
     FOR i = {-1,1}
        m' = m + i;
        n' = n;
        IF ((In_Grid(m',n') AND (p_m',n' is not checked) AND NOT Transverse_Edge((m,n),(m',n'))
              AND (e_1 ≤ p_m',n'.Ω ≤ e_2) AND (abs(R_m,n − R_m',n') ≤ 0.1 * R_m,n))
           L_obj ← p_m',n' ;        //p_m',n' is added to segment L_obj
           p_m',n' ← checked;
           PUSH (p_m',n') into TQ;
        END IF
     END FOR

     FOR j = {-1,1}
        m' = m;
        n' = n + j;
        IF((In_Grid(m',n') AND (p_m',n' is not checked)AND (e_1 ≤ p_m',n'.Ω ≤ e_2) AND (abs(R_m,n − R_m',n') ≤ 0.1 *
              R_m,n))
           L_obj ← p_m',n' ;        //p_m',n' is added to segment L_obj
           p_m',n' ← checked;
           PUSH (p_m',n') into TQ;
        END IF
     END FOR
UNTIL TQ = NULL
```

In summary, segmentation of objects is done by standard region growing [97] with 4-point neighborhood by setting appropriate thresholds so as to accommodate irregular obstacle shapes while growing objects. In [97] similar intensity pixels starting from a seed point or points are connected and assigned a region by looking at immediate neighbours. Here obstacle points are grown based on value of unevenness for a point to be around 1 ($e1 \leq 1 \leq e2$). Here $e1$ and $e2$ are the lower and upper limits for unevenness to restrict region growing. Points on ideal vertical obstacles (walls) have unevenness $\Omega = 1$. In addition to noise, most obstacles are far from being perfectly vertical. Tree trunks, humans and vehicles have curved surfaces causing unevenness to deviate from 1. This can be seen as change in slope from 90° compared to vertical obstacles.

Algorithm 4.4:
Function: Verifies whether the point is in Grid or not, ***In_Grid***(u, v):
Input: Indices of a point $p_{u,v}$
Output: Returns true if the point $p_{u,v}$ is in grid.

IF (($1 \leq u \leq u_{\max}$) AND ($2 \leq v \leq v_{\max}$))
    Return *True*;
ELSE
    Return *False*;

In addition, we restrict region growing in presence of edges in transverse direction. If an edge is detected in the transverse direction (***transverse_ edge***) in the way it is defined for transverse obstacle detection in Chapter 3 (Algorithm 3.2), region growing is stopped in the transverse direction. An edge is considered when the transverse unevenness is greater than the unevenness threshold in that direction ($|\Gamma| \geq \Gamma_{th}$). Although high point density along a ring allows difference in radial unevenness ($\delta\Omega$) alone to capture most edges, ***transverse_ edge*** separates very close objects, for example a person standing near a distant wall where the difference $\delta\Omega$ is very small. In addition to radial and transverse edges, we conservatively halt region growing when the range difference between neighbouring points exceeds 10% of the query point's range. This is to avoid even a single false connection at point level that could cause under-segmentation. Entire non-ground points are segmented using ***Grow_ Object*** function (Algorithm 4.3) by providing unchecked obstacle points as seeds. Function ***In_ Grid*** (Algorithm 4.4) restricts region growing within the range image.

4.3 Results

Segmentation is performed on single scans collected by robot across our campus with scanner at height $H = 1.3m$. Extensive experimentation were performed with limiting threshold values of $e1$ and $e2$ conservatively kept at 0.6 and 1.8. Transverse height threshold (g_{th}) is kept at $40mm$. These thresholds can be tuned to control the degree of segmentation. Our robot mainly moved on roads and green lawns. Apart from regular buildings and trees, environments contained dynamic objects like people and other traffic participants.

4.3.1 Experiments in structured environment

Validation of segmentation algorithms and evaluation of thresholds on unevenness is done on data where the scanner was placed on stationary stand over the lab terrace (structured environment) at $1.3m$ height. Three boxes of heights $10cm, 15cm$ and $20cm$ are placed on floor at different horizontal distances from sensor. Figure 4.5 shows segmentation on the lab terrace data when boxes are at $5m$ distance. Floor in center image, as with all traversable regions in this section, is shown in Green (Ground). Right image shows three distinct segmented boxes after removing ground. Highlighted rectangular portion shows door (Brick Red) and ladder (Blue) leaning on wall. Terrace boundary walls are segmented into multiple segments (Purple, dark Caramel, Yellow) because of transverse edge restriction in region growing.

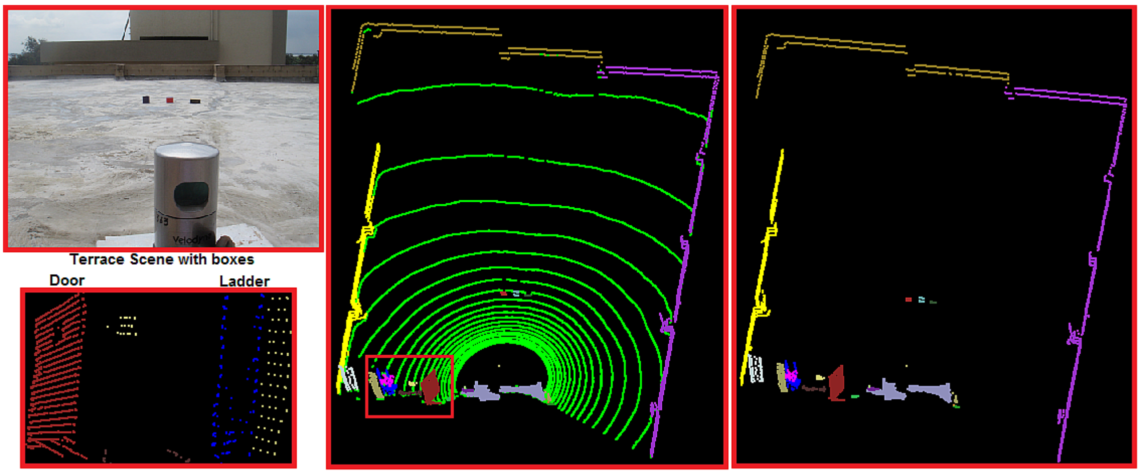


Figure 4.5. Terrace segmentation, Top left: Terrace scene, Centre: Segmented scan, Right: Ground removed, Bottom left: Highlighted Rectangle portion.

Table 4.1: Confusion matrix for terrace data.

True\Inferred	Ground Truth	Ground points	Object points	Error
Ground points	20426	21022	0	2.91%
Object points	32089	596	31493	1.86%

There is clear separation between ground and obstacle points with very few wrong classifications. This is also observed from the confusion matrix (Table 4.1). This gives the actual number of correctly and incorrectly classified points against the ground truth for both the ground and object points. High number of obstacle points classified as ground points is because of local planarity in steps in front of a door. Only 12 object points otherwise are classified as ground points. In this data, no ground point is classified as object point. Errors are calculated as a percentage against ground truth.

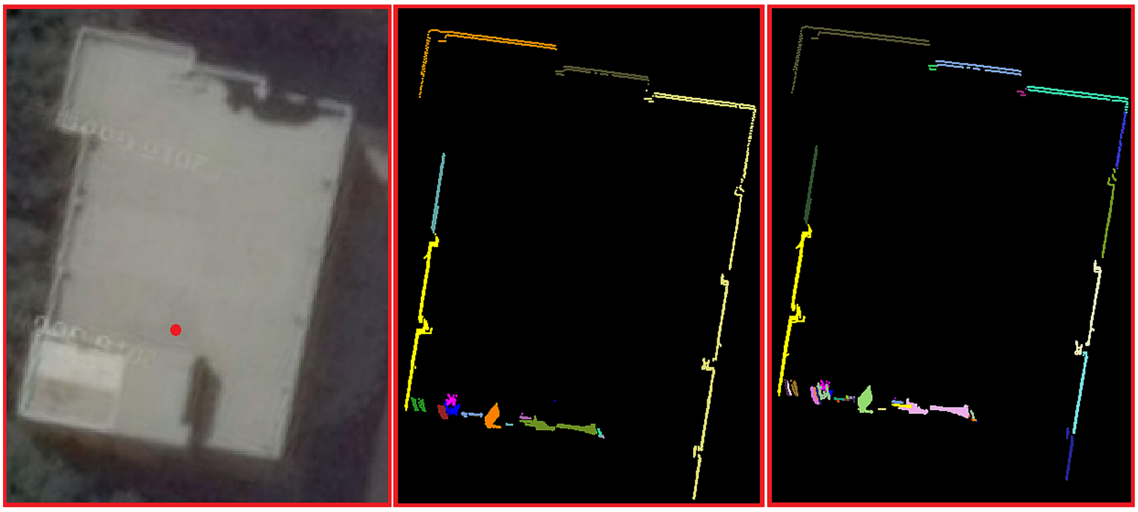


Figure 4.6. Transverse edge detection: Left, lab terrace (Google map) with sensor position (Red dot), Centre, $g_{th} = 40mm$, Right $g_{th} = 10mm$.

The ability of transverse edge detection in differentiating close by objects is demonstrated in Figure 4.6. Here, when the transverse threshold (g_{th}) is reduced from $40mm$ to $10m$, it brings out even small projections (Green and Purple) ahead of walls as distinct segments ($g_{th} = 10mm$).

4.3.2 Unstructured outdoor environments

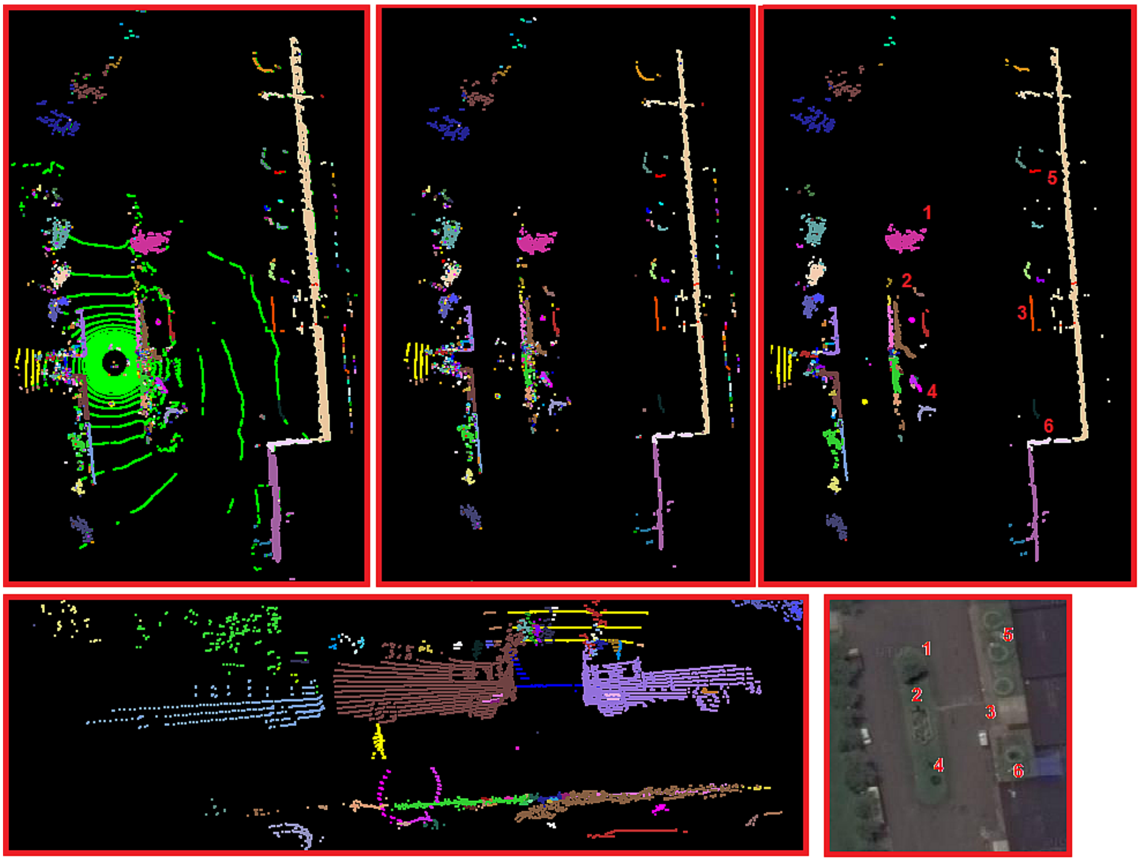


Figure 4.7. Outdoor segmentation, Clockwise from top left: Segmented scan, after removing ground, after removing smaller segments (points ≤ 5), aerial image of scene (google map), and selected side view.

Complexity of outdoor environments and the subjective nature of segmentation make quantitative evaluation against ground truth very difficult. Figure 4.7 shows such segmentation in an outdoor environment along with an aerial map of the region. The results are evaluated qualitatively by observing different segments. The algorithm properly segments the lawn edges, small walls around fountain and circular structures (label 5). However, complex objects, local planarity and occlusion in outdoor environments result in some over-segmentation. Porous tree canopies get over-segmented, even when tree trunks are properly segmented. Segmentation of trunk is not trivial, with Bargoti et al. [110] segmenting trunks in apple orchards using both Lidar and camera. Removing segments containing fewer points improves visualization of important segments without disturbing nature of objects or their positions. Side view shows pedestrian (Yellow) and vehicles closely parked under trees getting separately segmented. Circular ring structure (Magenta, label 4) occludes wall (label 6) from robot's position causing over-segmentation. Most distant trees are well segmented.

4.3.3 Object over-segmentation

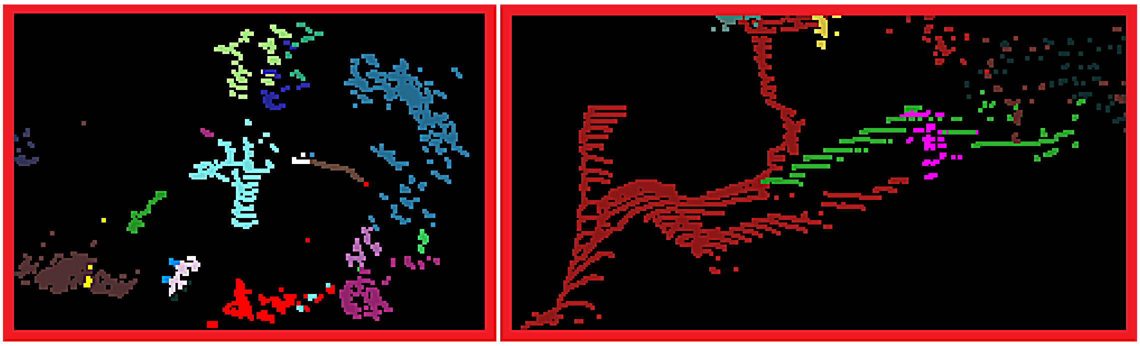


Figure 4.8. Segmentation with trees.

Figure 4.8 shows difficulties when segmenting trees nearby ($\approx 10m$). Data sparsity and porous canopies result in over-segmentation. In left image, while tree trunk (Cyan) is properly segmented, remaining portion segments into different clusters. This is subjective as segments represent clusters separated by some distance. Limited sensor view in vertical direction also contributes to discontinuities. Right image shows another tree with trunk and its branches segmented together (Red) but sub branches (Green) and leaf clusters segmented separately owing to discontinuities in unevenness.

4.3.4 Segmentation on slopes

Proper traversable ground detection even on slopes is the advantage accrued from using unevenness. Figure 4.9 shows two different slopes (approximately 9.9° and

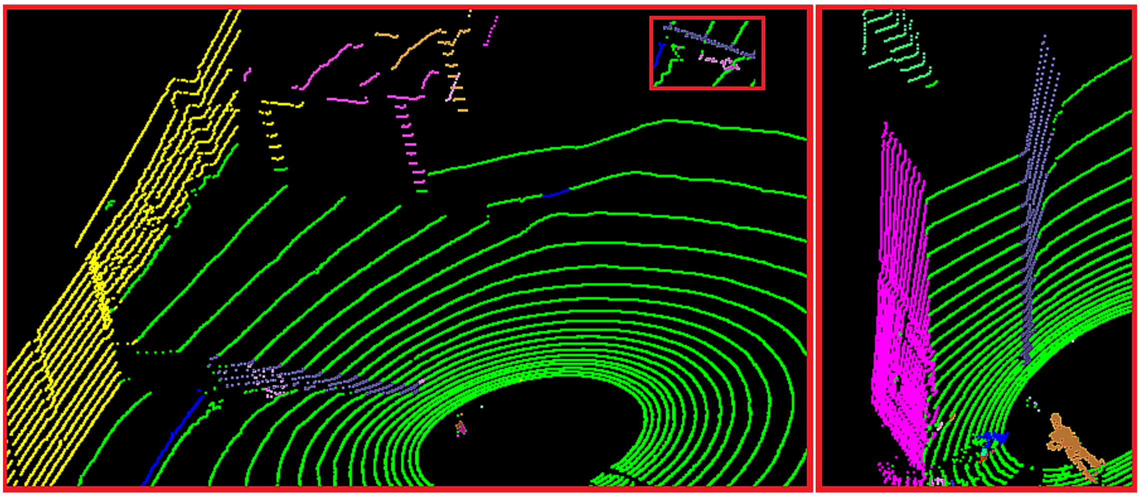


Figure 4.9. Segmentation with ramps, left: First ramp with bicycle in inset, right: Second ramp.

12.9° as traversable. Left image in inset shows bicycle (pink) parked very close to the ramp as different segment. Separating such close objects is difficult with traditional range difference [103] or voxel methods [79].

4.3.5 Density variation and execution speeds

Depending on environment, segmentation at full data density executes in $15-30ms$ on desktop computer with Intel corei3, $3.2GHz$ clock and $4GB$ RAM. This can be considered as realtime performance since the scanner rotates once in $100ms$. It may be noted that region growing is computation intensive and not computation of unevenness. Moosmann et al. [43] reported $600ms$ with reduced density and considers it real time. The algorithms using unevenness are considerably faster. Segmentation can be further accelerated by reducing the scan density. Figure 4.10 shows segmentation of a bicyclist by reducing data density in the transverse direction by interleaving points according to the indicated resolutions. Reduction in densities brings down the execution times but also result in over-segmentation because of loss of neighbourhood connectivity. This brings trade-off between speed and segmentation efficiency. Reasonable segmentation is however achieved with half the density.

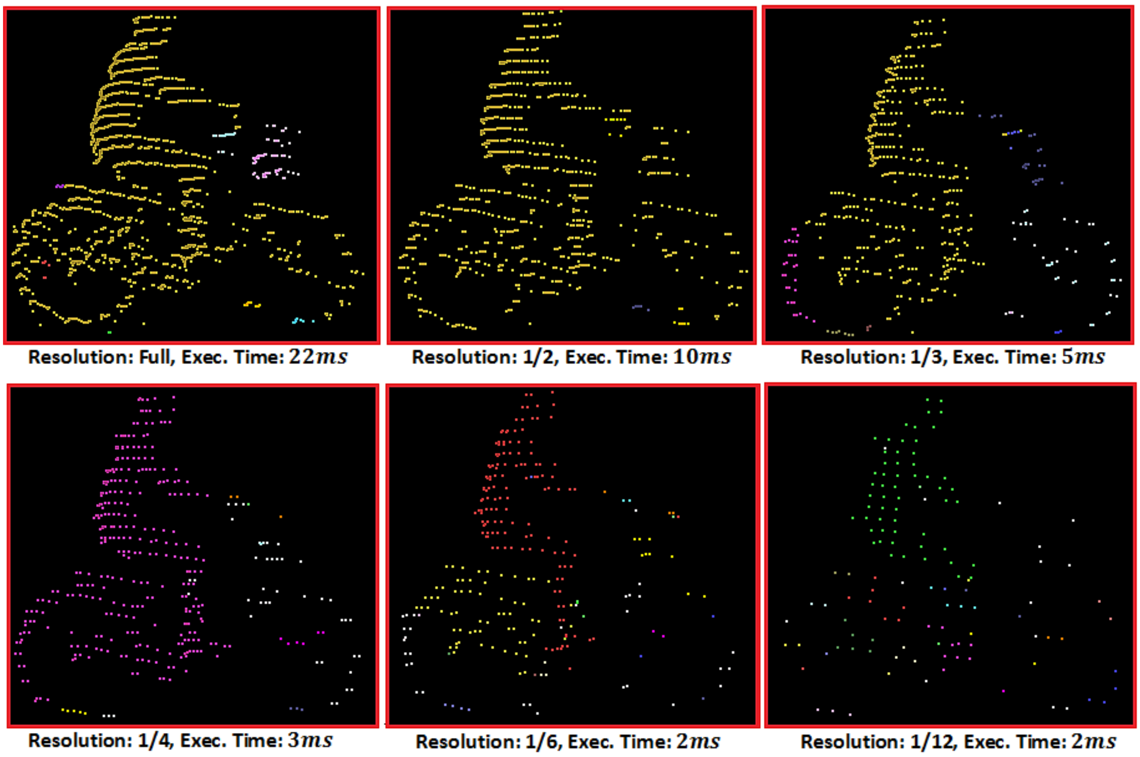


Figure 4.10. Bicyclist segmentation for different data densities along with execution times (whole scan).

4.3.6 Quantification of individual segment results

Quantitative evaluation is done on selected individual segments. Figure 4.11 shows individual features collected across scans. Features are clearly separated from ground and contain only few outliers (different colour points). Quantitative evaluations, performed using Precision, Recall and F-score according to Goutte and Gaussier[111], are tabulated in Table 4.2. Precision is the ratio of number of correctly segmented points to number of segmented points, while recall is ratio of number of correctly segmented points to number of object points. F-score is a test measure considering both precision and recall which accounts for both commission and omission errors in giving an overall quantitative measure for segmentation.

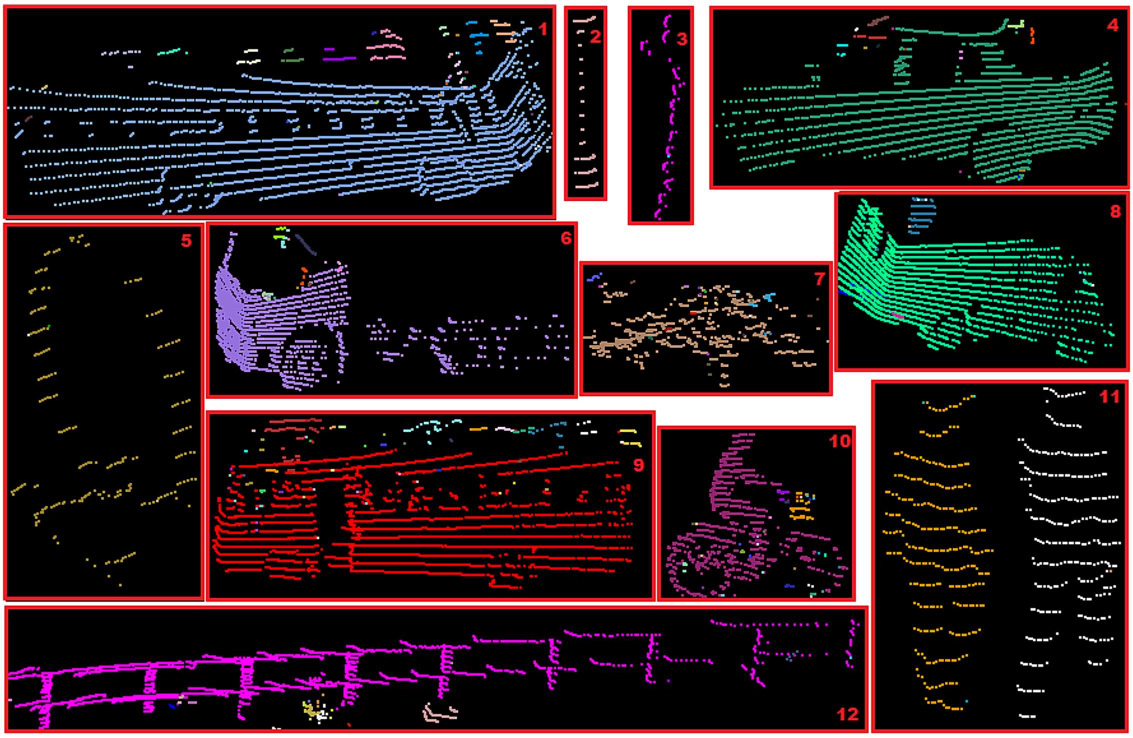


Figure 4.11. Features collected from different scans.

Table 4.2: Measurement tests for features in Figure 4.11

Sno.	Feature	Number of correct points	*precision*	*recall*	$F - score$
1	Bus	3214	1.0	0.9286	0.9630
2	Lamp Post	78	1.0	1.0000	1.0000
3	Fire Hydrant	105	1.0	0.9906	0.9953
4	Car	3244	0.9994	0.9707	0.9848
5	Ring Structure	215	1.0	0.9908	0.9954
6	Volvo Tractor	3206	1.0	0.9587	0.9789
7	Tree	703	1.0	0.9373	0.9677
8	Mail Van	2601	1.0	0.9662	0.9828
9	Minibus	2610	1.0	0.9164	0.9564
10	Cyclist	1431	0.9972	0.9359	0.9656
11	People (2 numbers)	307+298	1.0	0.9934	0.9967
12	Road Railing	2630	0.9985	0.9955	0.9970

$$F - Scrore = 2 * \frac{(precision \times recall)}{(precision + recall)} \tag{4.1}$$

Clear separation of objects from ground gives high precision values. Precision values of Car, Cyclist and Railing are less than 1 as some points fall outside. While precision indicates under-segmentation, recall indicates over-segmentation. Low recall values are obtained for the vehicles when a few lasers pass through windows hitting other surfaces. Similarly, disturbance caused by trees lead to low recall values. Most rigid objects return high values. Overall segmentation efficiency with variety of objects using F-score indicates efficient point level segmentation.

4.4 Conclusion

It is possible to perform an efficient segmentation of relevant objects around a robot at a point level using unevenness in spite of environmental complexity. Unevenness has the ability to capture distinct surfaces in spite of data being radially sparse. Unevenness brings out enough features that can be used by robot for its navigation. It detects surface discontinuities even in presence of terrain slopes or pitching and rolling of robot. Segmentation is carried out using only the ordered range data from the Laser scanner. Objects are grown into segments using point level region growing within the pre-specified unevenness limits. Unlike most methods involving grids or attributes calculated using set of neighbouring points, objects get clearly separated. Method performs segmentation in real time and the execution speed can be further accelerated by reducing density in the transverse direction. Depending on environment and objects, thresholds may need slight adjustments. Segmented features have the potential to be used for robot localization and scan registration which is the subject of our next chapter.

Chapter 5

Scan Registration

A robot equipped with laser scanners can only sample a limited area surrounding it from its current position. This is because of a limited range ($80m$ for HDL32) of the sensor, and also because of occlusions where the scanners are unable to sample regions behind the obstacles. Some of the occluded or out of range regions may be visible to the robot from a different location. That is why it is required to aggregate scans collected by the robot at regular intervals of motion. An ideal approach for aggregating scans is by transformation of current scan to the previous scan when the relative pose difference between the two scans is known. Even when a robot is equipped with wheel encoders, Inertial Measurement Units (IMU) and Global Positioning Systems (GPS), the pose difference between the two scans cannot be accurately and reliably measured. This is because of measurement errors in sensors. Measurement of difference in robot pose between two locations is affected by slippage between wheel and ground. IMUs are sensitive to their exact placement and orientation on robot. GPS is unreliable without clear sky; multipath reflections in presence of buildings and other structures also reduce its accuracy. Instead of trying to measure the relative pose difference between two locations, the popular approach for aggregating scans is by obtaining the rigid transformations between scans taken at different locations by a moving robot by using the scan data itself. Transformation is obtained by matching one scan against the other. This is called registration. Successive registration of scans by obtaining a series of rigid transformations results in tracing the trajectory of the robot.

Registration aggregates multiple scans into a common coordinate frame. Coarse measurements between scans obtained from the odometric sensors are input as initial estimates for the registration algorithms. When a moving robot successfully registers scans starting from its initial position, the robot poses at which the scans are taken can be established. This eventually results in a map with points aggregated from all the scans. This largely overcomes the problem of non-uniform point density of a single scan. Each feature or object in the environment can now be modelled without problems of occlusion. Detected features are useful for Segmentation, Simultaneous Localization and Mapping (SLAM) or Motion planning. Figure 5.1 shows the result of aggregating points by registering five scans from a moving robot. Points from each scan are shown in a different colour with the centre of the circle being around the position of the scanner for that scan.

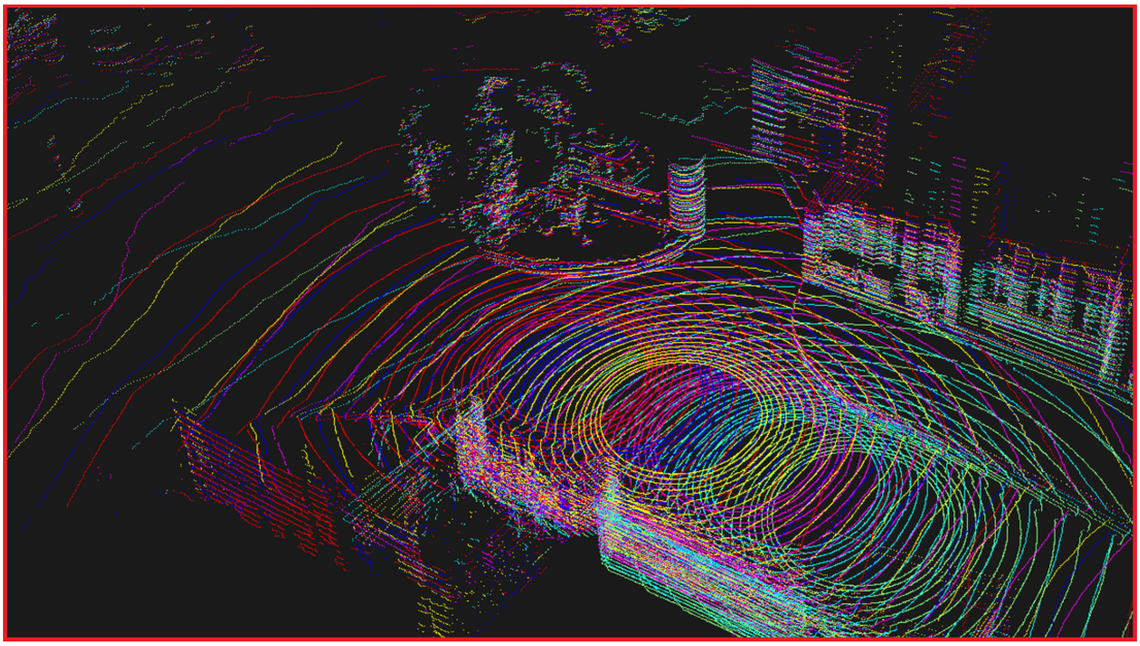


Figure 5.1. Registration of five successive scans from a moving robot.

Registration can primarily be performed using metric (distance) information of points when there is a sufficient overlap between two scans. In addition, other point attributes such as colour and intensity can also be used to establish correspondence between points. The registration problem finally amounts to finding a transformation (rotation and translation) between two scans after establishing (robust) correspondence between the points and matching the scans. Nüchter [51] classifies the scan matching approaches into two categories.

1. **Matching as an optimization Problem:** Here a cost function is used to evaluate the quality of alignment between the scans. Scans are registered by determining the rigid transformation that minimizes the cost function. Rigid transformation includes both rotation and translation components.

2. **Feature based matching:** Extracts few distinguishing features from the scans and uses them as corresponding features for calculating the alignment.

The Iterative Closest Point, or sometimes called the Iterative Corresponding Point (ICP), is a well-established and popular method for scan registration. Besl and McKay [24] is the most cited work on registration, with several variants of ICP published in literature in the last few decades. These along with other popular registration algorithms are described in the next section. The ICP algorithm registers two scans when an initial guess for the relative poses between the scans is provided. The algorithm computes the translation and rotation between the scans such that they match together. This is done by iteratively minimizing the error between the closest or the corresponding points. Rusinkiewicz [112] decomposed the whole registration process into the following stages and compared convergence rates for different ICP variants. An efficient variant is then developed for fast registration.

1. **Selection:** Selection of sufficient number of points in a scan for successful registration.

2. **Matching:** Establishing point correspondences between scans for calculating error metric.

3. **Weighting:** Providing weights between point pairs using strength of correspondence.

4. **Rejection:** Points degrading registration are ignored based on the quality of match.

5. **Error Metric:** Assigning an error metric based on the point pairs.

6. **Minimizing error:** Minimizing the error metric.

It is computationally expensive to generate correspondences for all the points between two scans. Registration for high resolution scans is accelerated by using a subset of points instead of the entire point cloud. Entire point set is then transformed using the rigid transformation obtained from registering the subsets. Two points, being always collinear, bring ambiguity about rotation and are not enough for registration. Theoretically a minimum of three corresponding points from each scan are needed for calculating translation and rotation between scans [113]. Different approaches can be used for subsampling the scans. Simplest and a natural way of sub sampling is selecting a uniform distribution of points [114] with each point getting an equal probability of selection. Because of non-uniform scan density with distance from the scanner, uniform point density is obtained by spatial discretization with each cell containing similar number of points. Other ways include random sampling [115] or selection of points with High intensity gradient [116]. Rusinkiewicz and Levoy's [112] select points such that the distribution of normals among the selected points is as large as possible. This allows selection of points representing small features. The more appropriate subsampling should thus retain points based on the importance of features sampled by them.

Selection of points input to the registration algorithms is an important step for improving the overall execution times. As search for neighbouring point accounts for nearly 90% time in overall registration process, registration is accelerated by either speeding nearest point search or by reducing the number of points. Post detection of unique and distinguishing features in a scan, one can select only points sampling those features for registration ignoring the remaining points during determination of transformation. This reduces the number of searches resulting in speeding up of registration.

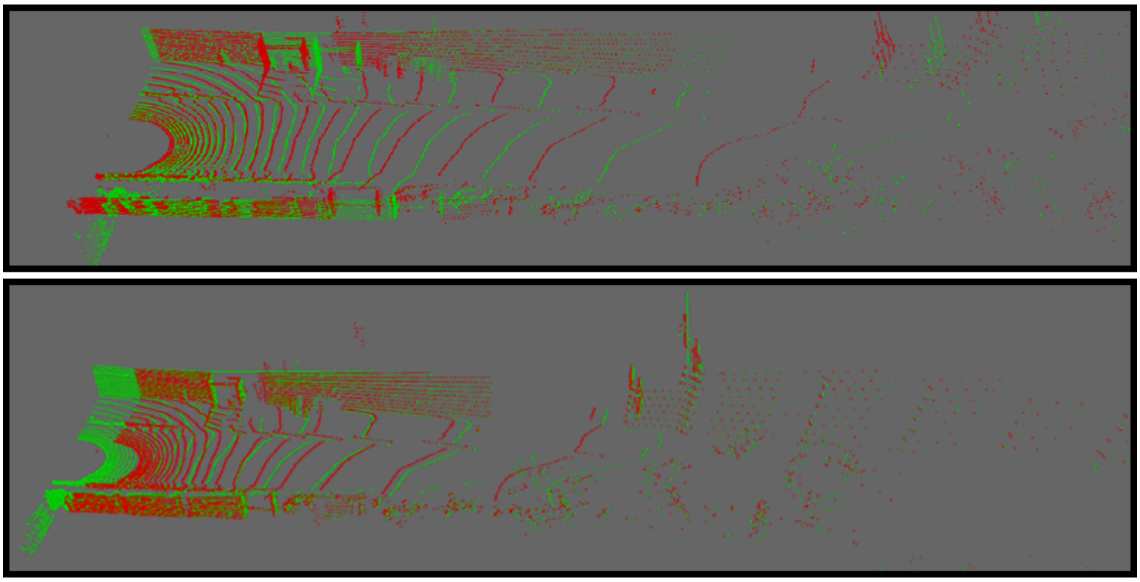


Figure 5.2. ICP registration without initial guess (top), with initial guess (bottom). Here the robot moved 2m between the scans.

The measure of unevenness, introduced earlier in this thesis, leaves feature signatures in scans apart from detection of obstacles and traversable regions. Using unevenness, points representing level surfaces, like the traversable portions or other locally planar regions, can be ignored for scan matching. Removal of ground points reduces the chances of the algorithm to converge to a local minimum. This problem is acute with high density of points on the ground near to the sensor. This region gets sampled as concentric rings with comparatively higher point density in transverse directions. This can be seen in Figure 5.2. During search for closest points for correspondence during the registration process, points on the rings closer to each other in the two scans have the tendency to pair with one another. Due to the huge number of points on the rings, these matches will dominate other genuine matching pairs, whose contribution to the calculation of rigid transformation thus getting diminished. This has a potential to cause the algorithm to converge to a local minimum. The transformation thus obtained will result in incorrect scan registration. Using unevenness, only points that improve registration are selected for finding correspondence. For example, smaller feature signatures like road edges or uneven portions of a road obtained using unevenness are utilized for establishing correspondence and subsequent registration.

5.1 State of the Art

The prominence of registration methods started in the early 1990s with almost simultaneous independent publications of three papers related to registration. The most popular and widely cited among them for the Iterative closest points is by Besl and McKay [24] for registration of scans using point-to-point matching. The ICP relies on finding the rigid transformation with the rotation and translation component that minimizes the sum of Euclidean distances between the model (reference) and the data scans. The corresponding point in the reference scan is the point with the least Euclidean distance to the query point in the data scan to be matched. Chen and Medioni [117] were more specific for aligning of range images by using the point-to-plane variant of ICP, assuming that most of the data is locally planar. Zhang [118] describes ICP for scan registrations using free-form edges and surfaces from the stereo images and adds a robust method for outlier rejection for selecting the points for scan matching. Other approaches include Blas and Levine [119] using the point-to-projection method for matching. Park and Subbarao [120] introduce a point-to-plane registration by combining with the high-speed advantage of the point-to-projection matching for fast and accurate matching.

Other approaches for matching are the Iterative Dual Sampling by Lu and Milios [20] where the point matching is improved by having two sets of correspondence for global scan matching. Censi [121] performs scan matching as a probability distribution approximation problem for global registration of scans. Hähnel and Burgard [63] compute a probability density for each pair of scans and then performs the registration using a greedy hill-climbing search in the likelihood space and show that the probabilistic model gives less estimation error compared to the ICP method. Pulli [122] performs registration of multiple large scans by first aligning the scans pair wise to obtain the constraints before registering large datasets from different views. Rusinkiewicz [112] compares different variants of ICP with focus on convergence speed. Comparisons are carried out for different schemes at different stages of registration such as sampling, finding corresponding points, giving weights and rejection of points. Pomerleau [123] does an extensive study on the different variants of the ICP and relates them into an unique framework and provides a more structured methodology for evaluation of the geometric registrations for robotic applications.

As the density of points becomes sparse away from the scanner, when the robot moves, the density of points at a spatial location will not be the same in different scans, making it difficult for the ICP to find corresponding points in the two scans for all the points. To overcome this, Segal et al.[124] combine the point-to-point and the point-to-plane variants of ICP into a single probabilistic framework to create a kind of plane-to-plane matching called the Generalized-ICP (GICP) wherein the local surface normal of each of the points in the two scans is calculated using the local neighbourhood for matching similar surface structures between the two scans. Use of structural information reduces the effects of improper correspondence because of noise. This improves convergence rate compared to conventional ICP. Pandey et al. [125] extend the GICP by registering the already co-registered points from a camera image and a 3D point cloud from a Laser scanner. The point correspondence between scans is performed by high dimensional feature descriptors such as Scale Invariant Feature Transform (SIFT) [126] or Speeded up Robust Features (SURF) [127] to the 3D points and the final refinement is done by GICP. All variants of ICP require establishing corresponding points using nearest neighbor searches, which are a bottleneck for these algorithms. For finding correspondence of all the points, the complexity of searches will be $\mathrm{O}(n^2)$. Searching the nearest point using a k-d tree [70] is the most common approach used for accelerating the search with complexity reduced to $\mathrm{O}(nlogn)$. Nüchter [61] introduces cached k-d tree with a novel search procedure with a pointer to the predecessor node, accelerating the search by a further 50% compared to the conventional k-d tree search.

Alternative approaches using the Normal Distribution Transform (NDT) are the class of registration algorithms proposed to overcome the problem of correspondence encountered by the ICP variants. The first of these approaches was proposed by Biber and Straßer[128] for 2D environment, where the 2D scan was discretized into cells and each cell is assigned a Normal transform which will locally model the probability of measuring a point. The normal distributions give a piecewise smooth representation of the location with differential probability density. Using this representation, standard numerical optimization methods are applied for registering the scans. Here matching of a point in the data set is done in terms of probability of its occurrence in the reference scan that is represented with normal distributions. Magnusson et al. [39] extended the NDT to 3D scan registration, particularly for mapping underground mines. Here the space is discretized into voxels and each voxel is assigned a Normal distribution. The NDT represents scans as a set of Gaussian distributions that models the surface of the reference scan as a Probability Density Function (PDF). Stoyanov et al.[129] extend NDT further by using a distribution to distribution matching between the scans instead of point-to-distribution matching used in the earlier methods. Both the reference and the data scans are represented as Gaussian distributions. Registration of the scans is performed by minimization of the L2 distance between the distribution sets of the two scans. NDT representation however results in discontinuities at the border of the cells. To overcome this problem, Das and Waslander [91] propose to segment the scan into clusters after removing the ground region and then assign a Normal distribution to each of the clusters and calling the method as Segmented Region Growing NDT (SRG-NDT). With this approach, there is considerable speedup in registration without affecting the accuracy. More recently Kim and Lee[130] proposed a Super Voxel Normal Distribution Transform (SV-NDT), where they consider plane to be the best surface to be modeled without losing information on local surface structures. Super Voxels are generated at the partitioning stage. The SV-NDT method is shown to be more robust compared to the regular NDT methods. In general, the reduction in the number of points also improves the registration speed for other registration algorithms (GICP, NDT) because of the reduced number of computations point wise.

5.2 Registration of scans using ICP

The ICP method proposed by Besl and McKay [24] is the basic method for registration of the scans using point to point matching. The scan taken by the robot in the current pose, called Data scan D, has to be registered against the reference scan taken at the previous pose, called Model scan M. Registration finds a rigid transformation $(\boldsymbol{R}, \boldsymbol{t})$ that minimizes a cost function. Here $\boldsymbol{R}$ is the rotation matrix and $\boldsymbol{t}$ is the translation vector. The cost function is computed as

$$E(\boldsymbol{R}, \boldsymbol{t}) = \sum_{i=1}^{N_m} \sum_{j=1}^{N_d} w_{i,j} \|m_i - (\boldsymbol{R} d_j + \boldsymbol{t})\|^2 \tag{5.1}$$

Here $w_{i,j}$ is the weight of correspondence between points m_i and d_j in model and data scans respectively. Weight $w_{i,j}$ is set 1 if the i^{th} point in the model scan is same as the j^{th} point in data scan. When there is no correspondence, $w_{i,j}$ is set zero.ICP algorithm iteratively minimizes the cost function Equation 5.1 by repeating the following two steps

In each iteration step, the ICP algorithm identifies the closest point for each data point as the corresponding point and determines $(\boldsymbol{R}, \boldsymbol{t})$ that would minimize the cost function $E(\boldsymbol{R}, \boldsymbol{t})$.

Algorithm 5.1:

ICP algorithm

1. ***For*** *$k = 0$ to maxiterations* ***Do***
2. ***For all*** $\boldsymbol{d_j} \in D$ ***Do***
3. Find the closest point within a range d_{max} in the set M for point $\boldsymbol{d_j}$
4. ***End For***
5. *Calculate the transformation $(\boldsymbol{R}, \boldsymbol{t})$ that minimizes the cost function $E(\boldsymbol{R}, \boldsymbol{t})$*
6. *Apply the calculated transformation $(\boldsymbol{R}, \boldsymbol{t})$ to the data set D.*
7. *Compute the difference of the quadratic error, i.e. compute the difference of the value $\|E_{k-1}(\boldsymbol{R}, \boldsymbol{t}) - E_k(\boldsymbol{R}, \boldsymbol{t})\|$ before and after the application of the transformation. If the difference is less than the error threshold ϵ_{th}, terminate*
8. ***End For***

For all the steps in iteration, the error keeps reducing, i.e. $E_{k+1} \leq E_k$. Since the error is bounded, the ICP algorithm is shown to converge to a minimum [24]. In robot applications, two scans do not completely overlap, so a maximum tolerable distance d_{max} is set for a point such that there is no corresponding point beyond this distance. The algorithm can be terminated when the criterion for convergence is met or when it reaches the maximum number of iterations $maxiterations$. During each iteration, the algorithm tries to minimize cost function (Equation 5.1)and tries to maximize the number of corresponding points.

5.2.1 Solution for Rigid transformation

An important step in ICP algorithm is the calculation of rigid transformation $(\boldsymbol{R}, \boldsymbol{t})$ in each iteration step till the algorithm converges. Solutions for finding the transformation include both direct and indirect solutions [51]. The direct solutions, also called as closed form solutions, give the solution in one step without resorting to the iterative processes that need an initial guess to be provided. Examples of the closed solutions include the popular methods by Arun[131] using the Singular Value Decomposition(SVD), and the method by using unit quaternion by Horn[113]. The indirect methods for the solution include the Gauss-Newton, Levenberg-Marquardt or the gradient descent methods. The indirect methods result in higher execution times because of the need to perform several evaluations on Equation 5.1 before arriving at the solution.

The cost function in Equation 5.1 can be reduced to

$$E(\boldsymbol{R}, \boldsymbol{t}) = \frac{1}{N} \sum_{i=1}^{N} w_{i,j} \| m_i - (\boldsymbol{R} d_j + \boldsymbol{t}) \|^2 \tag{5.2}$$

Here,

$$N = \sum_{i=1}^{N_m} \sum_{j=1}^{N_d} sgn w_{i,j} \tag{5.3}$$

The points m_i and d_j are the corresponding points and can be represented as a $tuple(m_i, d_j)$. sgn is the sign function.

The difficulty in minimizing the error function lies in enforcing the orthonormality constraint for the rotation matrix. In most of the algorithms, the computation of rotation matrix $\boldsymbol{R}$ is done first, and then the translation vector $\boldsymbol{t}$ is obtained. For this separation, first the centroids of both the point sets are obtained

$$
\begin{aligned}
c_m &= \frac{1}{N}\sum_{i=1}^{N} m_i \\
c_d &= \frac{1}{N}\sum_{i=1}^{N} d_i
\end{aligned}
\tag{5.4}
$$

Proceeding further, two point sets $M^{'}$ and $D^{'}$ are obtained such that

$$
\begin{aligned}
M^{'} &= \left\{ m_i^{'} = m_i - c_m \right\}_{1 \ldots N} \\
D^{'} &= \left\{ d_i^{'} = d_i - c_d \right\}_{1 \ldots N}
\end{aligned}
\tag{5.5}
$$

Now $E(\boldsymbol{R}, \boldsymbol{t})$ from Equation 5.2 can be written using Equation 5.4 and Equation 5.5 as

$$
E(\boldsymbol{R}, \boldsymbol{t}) = \sum_{i=1}^{N} \| m_i^{'} - \boldsymbol{R} d_i^{'} - (\underbrace{\boldsymbol{t} - c_m + \boldsymbol{R} c_d}_{\tilde{\boldsymbol{t}}}) \|^2 \tag{5.6}
$$

When considering that all the points refer to the translation from the centroids ($\tilde{\boldsymbol{t}}$). For finding only the rotation, this can be considered to be zero [51]. Therefore, the error function is now represented only with the rotation component.

$$
E(\boldsymbol{R}, \boldsymbol{t}) = \sum_{i=1}^{N} \| m_i^{'} - \boldsymbol{R} d_i^{'} \|^2 \tag{5.7}
$$

5.2.2 Computing the Transformation using Singular Value Decomposition (SVD)

The algorithm for finding the rigid transformation by means of least square error was proposed by Arun [131] to find out the rotation ($\boldsymbol{R}$) and translation ($\boldsymbol{t}$). The optimal rotation is obtained by Singular Value Decomposition (SVD) of the correction matrix. Correction matrix is given as

$$\boldsymbol{H} = \sum_{i=1}^{N} m_i'^T d_i' \tag{5.8}$$

For finding the SVD of $\boldsymbol{H}$

$$\boldsymbol{H} = \boldsymbol{U}\boldsymbol{A}\boldsymbol{V}^T \tag{5.9}$$

Here $\boldsymbol{U}$ and $\boldsymbol{V}$ are the orthonormal 3×3 matrices and A is a 3×3 diagonal matrix.

$$\boldsymbol{X} = \boldsymbol{V}\boldsymbol{U}^T \tag{5.10}$$

For finding the validity of the obtained rotation $\boldsymbol{R}$, its determinant needs to be evaluated.

$$\text{if } det(X) = +1 \text{ , then } \boldsymbol{R} = \boldsymbol{X}$$
$$det(X) = -1 \text{ , The algorithm fails}$$

However, for true cases involving point sets, the algorithm does not fail. Once the rotation matrix $\boldsymbol{R}$ is obtained, the translation vector $\boldsymbol{t}$ is obtained using the translation experienced by the centroids

$$\boldsymbol{t} = c_d - \boldsymbol{R}c_m \tag{5.11}$$

This way both Rotation $\boldsymbol{R}$ and Translation $\boldsymbol{t}$ components are obtained. The method is explained in Algorithm 2.

Algorithm 5.2:

Function: Find ***Rigid_Transformation***$(\boldsymbol{R}, \boldsymbol{t})$.

Input: Model set $M = \{m_1, m_2, \dots \dots m_u \dots m_m\}$ and Data set $D = \{d_1, d_2, \dots \dots d_u \dots d_m\}$

Output: Rotation matrix R and Translation vector $\boldsymbol{t}$.

Obtain the centroids for both the datasets:

$$c_m = \frac{1}{m}\sum_{i=1}^{m} m_i$$

$$c_d = \frac{1}{d}\sum_{j=1}^{d} d_j$$

Calculate the variance of the individual points from their respective centroids :

$$m_i' = p_i - c_m$$

$$d_i' = q_i - c_d$$

Obtain a 3×3 covariance matrix:

$$\boldsymbol{H} = \sum_{i=1}^{N} {m_i'}^T d_i'$$

Find Singular Value Decomposition (SVD) of:

$$\boldsymbol{H} = \boldsymbol{U}\boldsymbol{A}\boldsymbol{V}^T$$

Determine the rotation matrix:

$$\boldsymbol{X} = \boldsymbol{V}\boldsymbol{U}^T$$

IF $(det\ (\boldsymbol{X}) = +1)$

$$\boldsymbol{R} = \boldsymbol{X}$$

ELSE IF $(det\ (\boldsymbol{X}) = -1)$

$$\boldsymbol{R} = \widehat{\boldsymbol{X}} = \widehat{\boldsymbol{V}}\boldsymbol{U}^T$$

END IF

Obtain the Translational vector:

$$\boldsymbol{t} = c_d - Rc_m$$

Note: when $det\ (\boldsymbol{X}) = -1$, $\boldsymbol{X}$ would be reflection matrix so column 3 of $\boldsymbol{V}$ is negated to give $\widehat{\boldsymbol{V}}$.

5.3 Detection of Key points using unevenness

Computation of closest point pairs between two scans is the crucial step for registration using ICP algorithm. A naïve implementation for searching nearest points between two scans require $O(|N_D||N_M|)$ evaluation operations, where N_D and N_M are the number of points in the data and the model scans respectively. This result in a complexity of $O(n^2)$ for finding nearest neighbours for all points where for a single point, n nearest neighbor search operations are required by the naïve method. There are several approaches for finding nearest neighbour search [132].State of the art algorithms find the nearest point by dividing the space recursively into subspaces represented by trees. Algorithms like octrees [37] divide space uniformly. Others like Vornoi, Delaunay or k-d tree perform non uniform subdivision of the underlying space. K-d tree, which is a generalized binary search algorithm, is generally used for nearest neighbour search in ICP algorithms. The search complexity is $O(nlogn)$ with k-d trees. Apart from accelerating search procedures, registration speed can be improved by considering only a subset of points in each scan instead of the entire point cloud. The selected points called key points represent important features in a scan that aid in accurate registration while filtering out points that could degrade registration.

In general, scan points can be categorized into ground and non-ground points. As explained earlier in chapter 3, the points sampling the ground from the rotating laser scanners form ring like patterns. Objects present in the environment disturb these patterns. Using the unevenness (Ω) as defined earlier (chapter 3), the points belonging to ground and those belonging to vertical objects are easily determined. Ground points having unevenness less than a given obstacle threshold ($\Omega \leq \Omega_{th}$) can be filtered out. We propose and demonstrate that registration can be performed more efficiently by selecting only the points belonging to important objects, which we call key points, instead of the entire point cloud. Particularly, vertical objects are considered robust features for registration. Instead of explicitly determining the ground points before filtering them and then detecting the vertical obstacles, key points are directly selected using the points whose unevenness values are close to $1 (\Omega \approx 1)$.

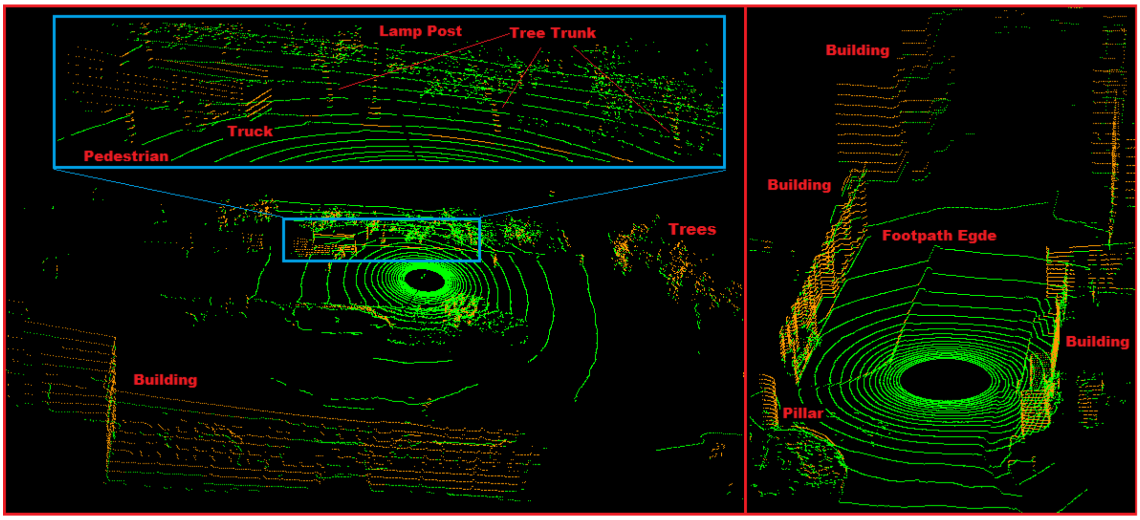


Figure 5.3. Key points in scan (Orange), Left: Semi urban environment 6213 ($\xi = 0.1$), Right: Urban environment 11568 points ($\xi = 0.05$).

Depending on the environment, a key point selection threshold (ξ) is tuned such that it gives enough points in a scan with unevenness close to 1 ($fabs(1-\Omega) \leq \xi$) for registration, i.e. points having unevenness between $1-\xi$ and $1+\xi$. In highly structured urban environments, the value of ξ is kept tight (small) so that points with unevenness in a narrow window around one are selected. In environments not having many notable vertical structures, the value of ξ is relaxed to accommodate points sampling little convex or concave surfaces as key points. For example, points belonging to tree trunk which is not perfectly vertical are selected when the threshold is relaxed. Figure 5.3 shows key points (Orange) within two scans (≈ 55000 points) in different environments. In the urban environment, primarily containing buildings, a tighter threshold ($\xi = 0.05$) detects more number of points (11568) compared to a relaxed threshold ($\xi = 0.1$) in a semi urban environment (6213 points). In structured environments, most points sampling structured building walls have unevenness very close to one. It can be observed that only few footpath edge points are detected as

Table 5.1: Number of key points for different thresholds along with execution times.

	Full Cloud	ξ = 0.5	ξ = 0.05	ξ = 0.005
Number of Key points	54654	19620	11568	3799
Execution Times (s)	2.39	0.62	0.41	0.18

key points. All footpath edge points get added to key point list when the threshold is further relaxed ($\xi = 0.5$); but this increases the number of key points. Threshold can be further reduced to bring down the number of key points to accelerate registration. In a semi-urban environment, a relatively high value of threshold ($\xi = 0.1$) brings out 6213 key points, a majority of which sample the huge building; otherwise the count will be even less. Points sampling the important features like truck, lamp post and pedestrians are detected as key points (inset). This relaxed threshold selects points sampling tree trunks as key points, while rejecting tree tops. However, tree tops far away are selected as key points because range of consecutive points on vertical objects far away is nearly the same. In general, the threshold (ξ) is tuned to obtain sufficient number of points spread over different objects to carry out registration. Even a small robust point set results in accurate registration compared to registration with a higher number of points.

Selection of key points list from a scan using key point threshold (ξ) is detailed in algorithm 3. The number of key points selected using different values of ξ is tabulated in Table 5.1 along with the time taken for registration of two scans from a structured environment using ICP from robot positions separated by $2m$. It can be seen that, correct registration is achieved even with points as low as 4000.

From the table, it can be observed that the execution times are reduced to nearly one-tenth of the execution time for full point cloud. With these reduced execution times and considering the speed at which the Velodyne scans the environment ($10Hz$), every third scan can be registered. Within this time, our robot operating

Algorithm 5.3:

Function: Detection of Key points using unevenness (Ω).

Input: Ordered point cloud data P from Velodyne in (R, θ, ϕ) format for θ between 0° and 360°, and ϕ between -30.67° and +10.67° degree; *keypoint_threshold* ξ.

Output: Set of $KeypointsList$ to be input to the ICP algorithm.

$KeypointsList \leftarrow NULL$
FOR θ_u between 0° and 360°
    Read points $p_{u,v}$, for $v = 1\ to\ 32$;
        FOR $v = 2\ to\ 32$, do
            Obtain the radial unevenness for point $p_{u,v}$: $p_{u,v}.\Omega$;
            IF $((fabs(1 - fabs(P_{u,v}.\Omega)) \leq \xi)$
                $KeypointsList \leftarrow p_{u,v}$
            END IF
        END FOR
  $u = u + 1$;
END FOR

at $2m/s$ would also have moved less than $60cm$. The farthest we have registered the scans with correct alignment in a semi urban environment is $8m$. Although most of the mapping using registration is done offline, this way scans can be considered to have been registered in real time.

5.4 Error quantification with registration using Key points

We demonstrate that not only the speed of execution, but the accuracy of registration is also increased by using key points. Error in registration is determined by comparing against the ground truth. Since it is difficult to obtain the ground truth from a moving robot in absence of prior map, the Velodyne laser scanner placed on a stationary stand was moved on a grid as shown in Figure 5.4. The scans so collected were registered and compared with known displacements of the scanner.

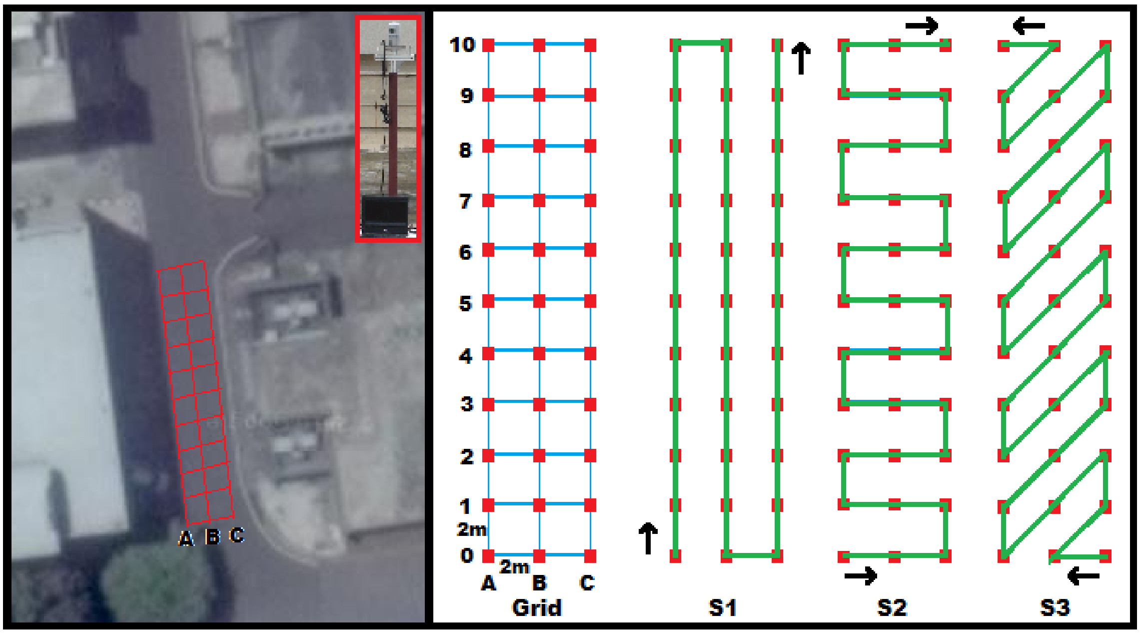


Figure 5.4. (From left): Aerial view with Grid overlay (inset: Velodyne on a stationary stand), Grid with red squares as scan locations, Sequence S1, Sequence S2, Sequence S3, Arrows show directions at start and end.

A set of 33 scans are taken on a near level road in front of our lab building. The scans are taken at locations arranged in a grid like fashion as shown Figure 5.4 where the grid is approximately overlaid on the aerial image of the environment. Grid is constituted with readings along three columns (A,B,C) along the positive X-axis with 11 readings in each column in the positive Y-axis. The distance between neighbouring scan locations is $2m$. Scan locations shown as red square markers in the grid are marked by stretching long strings and by measuring the distances between locations using a measuring tape. After recording the scans, a standard ICP algorithm is applied across the series of scans. Implementation used the Mobile Robot Programming Toolkit (MRPT) library [133] where a k-d tree is used to accelerate search for the nearest points. Registrations are performed without giving any initial guess between the scans.

Instead of comparing the results for a single registration between two scans, a sequence of registrations are performed akin to robot movement starting from one location and covering all the 33 scan locations. Three set of sequences (S1, S2, S3) are considered where the robot starts from the start arrow and moves towards the end arrow by following the path indicated by green lines joining red markers. The three sequences cover straight, lateral and cross movements by the robot. Each scan sequence required 32 registrations, with each registration providing the set of six errors including three errors along X, Y and Z axes and three errors along rotations in the Yaw, Pitch and Roll. Errors are measured by using the distances according to the ground truth and result from the ICP registration. The errors obtained from registrations across the three sequences are aggregated (total 96 registrations) and plotted as error distributions using box plots. A box plot produces a box and whisker plot for each column of errors. The box has lines at the lower quartile, median, and upper quartile values. Whiskers extending from each end of the box represent the most extreme values within 1.5 times the interquartile range. Outliers in data if any with values beyond the ends of the whiskers are displayed with a red "+" sign.

Figure 5.5 shows box plots for the execution speed and the number of key points selected for different values of thresholds (ξ). Compared to the full point size, it can be seen that the number of key points decrease with decrease in (ξ). This is in line with the values given in Table 5.1 for a single registration. The registration time is accelerated to one-tenth compared to full point set and the number of points used for registration (key points) is reduced to one-sixth using the same algorithm. Further acceleration is possible by optimizing the nearest neighbour search, like cached k-d tree [61] with the reduced point set.

Error distributions including both the translational and rotational components compared to the ground truth are again shown using the box plots (see Figure 5.6). Plots are made using the 96 registrations, with each registration containing six different errors. From the error plots, it is observed that the errors are generally decreasing with the reduction in the number of key points by reducing the value of ξ. It can be seen that the median value of all the errors is nearest to zero

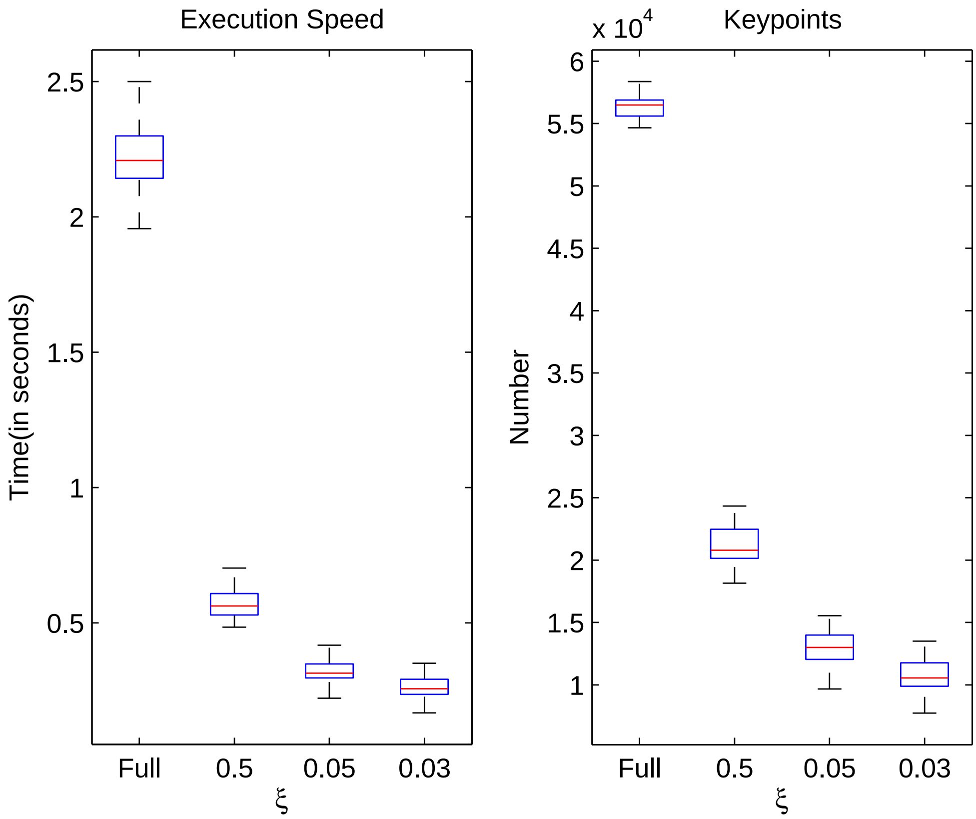


Figure 5.5. Box plots for Execution speeds (left) and Number of Key Points for different Key point thresholds (ξ).

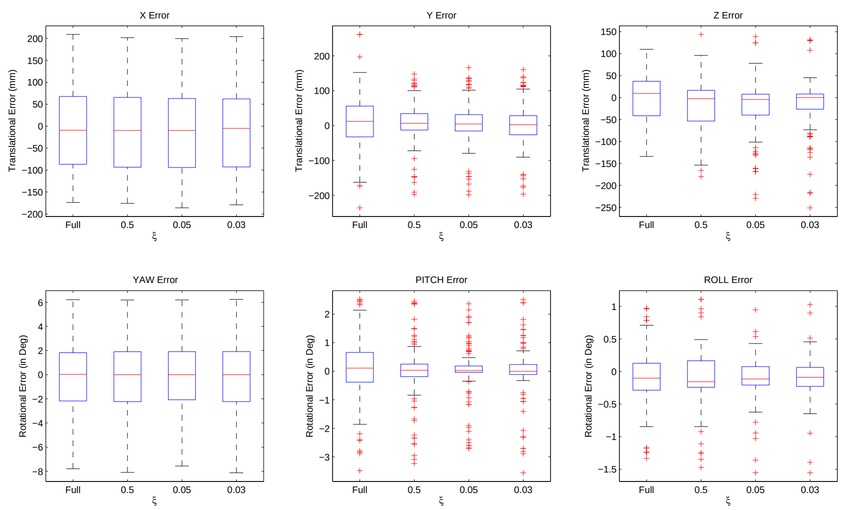


Figure 5.6. Error distributions with full cloud and with different key point thresholds (ξ).

when the number of key points are the least ($\xi = 0.03$) a value at which all the registrations in the sequences were successfully performed. However, the majority of registrations were successful even when ξ is as low as 0.005 with points less than 4000. This validates the fact that by selecting robust key points for registration instead of the entire point cloud, the accuracy of registration is increasing in addition to its speed. It may also be noted that the detection of key points using unevenness (Ω) is simple and direct using only the ordered range data and needs no extra computational effort. It also registers scans accurately without being provided with initial guess.

From the error distributions in the plots, it can be seen that the translational errors (X,Y and Z) are present to an extent of few centimeters and not zero as it should be in the ideal case. Notably X error is extended to $7cm(70mm)$ for the boxes and to $20cm$ for the whiskers. This error is consistent across the plots for different values of ξ and in fact improves for low values of ξ. These small errors are systematic because of the way ground truths are manually marked using the stretched strings for measuring distances and then placing the stationary stand with the scanner. This explains more error in X direction. Similarly, some play between the stand and sensor has led to errors in the Yaw direction (along the scanner rotation). However, this is less than $2°$ in most cases. The Pitch and the Roll errors are however significantly lower ($\leq 0.5°$) even for full clouds. Since the road on which the scans are taken contains some irregularities and is not perfect level surface, there are some outliers in the plots marked by red "+" signs. Outliers are particularly present in the pitch and roll errors which explain surface irregularities causing sensor tilts. Number of outliers appears to be more because of small range in errors. But significantly the overall observation indicates error to be close to zero and to reduce with reduction in ξ. While the Z error also shows overall improvement with decrease in ξ, there are a few scans (see outliers) in which the errors are more with reduced point set compared to full clouds. This is understandable in absence of ground points in the key points, which would have served as a reference in the z-direction. The key points are usually present as horizontal lines; this at times causes points in one horizontal line to incorrectly correspond to points in another horizontal line at a different height in the other scan. The absence of ground points whose presence otherwise could have restrained offset in the Z direction cause these errors. This can be overcome by presenting the ICP algorithm with initial guess particularly in the Z direction, but since the methodology in this work refrained from using initial guesses, the results are reported as it is even for z-coordinate. This error can also be mitigated when the data in the radial direction is dense (example HDL-64 scanner). Another way to overcome z error is by purposefully injecting selected points from the ground into the key points before registration.

5.5 Qualitative Results

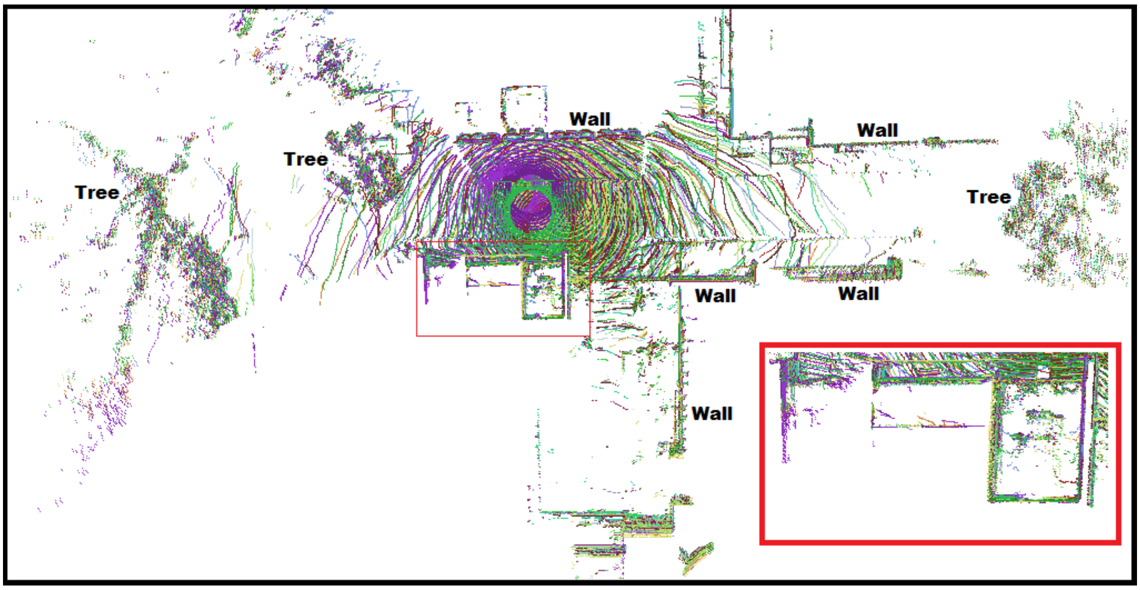


Figure 5.7. Top view of registration result of 15 scans. Inset (Selected portion).

The quality of registration can be evaluated by looking at the crispness of features in the registered scan [91, 134]. Crispness voxelizes point cloud and gives the number of cells containing points. Here crispness is not quantized as we do not compare against other methods, but only evaluate scans visually. Less number of occupied voxels indicates high crispness. When the scans are properly registered, the features in the scan appear very sharp. Improper registration leads to points getting scattered. Figure 5.7 shows result of ICP registration of 15 scans (sequence S2) using only the selected key points for the quantitative evaluation. Each scan is represented by a different colour. In the view from top, one can see that the walls are along straight lines. This shows image crispness. Image in the inset shows a rectangular room with the right wall also sampled as a rectangle. The wall geometry is rectangular because of the scans from both sides of the wall. The walls from the top view are nearly straight but for the perspective view of the image viewer where the walls away from the center appear oblique.

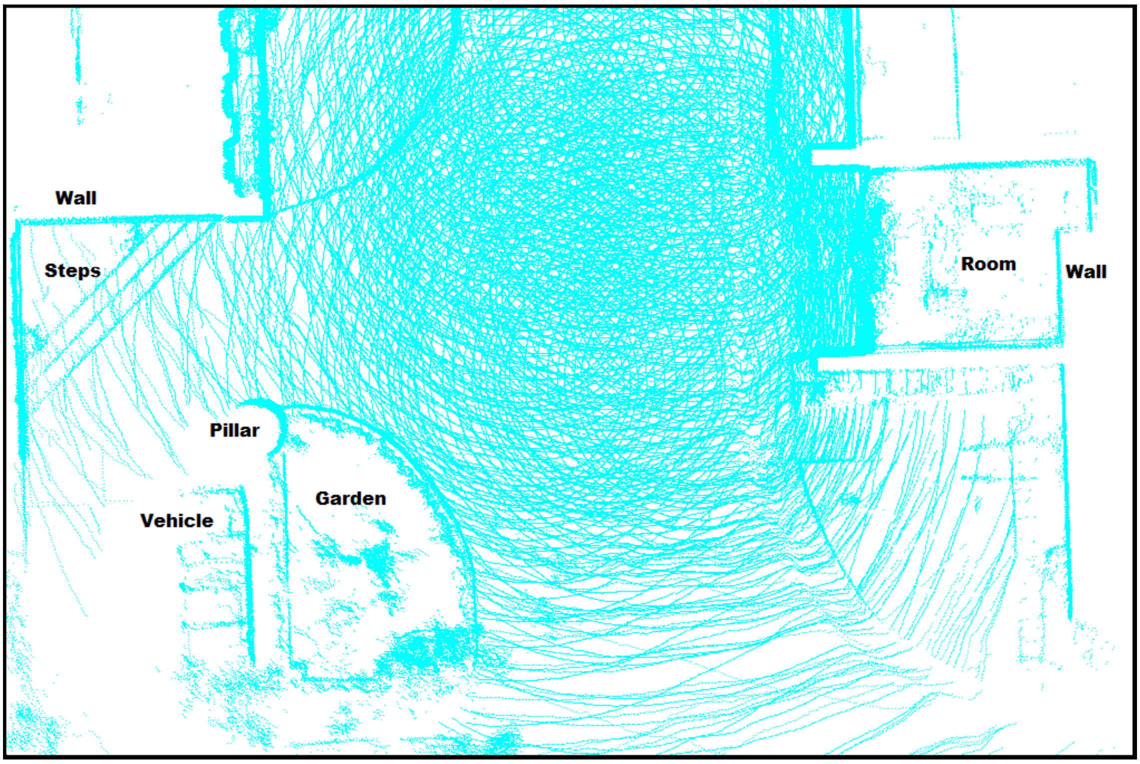


Figure 5.8. Closer top view of registration.

Figure 5.8 shows the top view of the registered scans in a different location. Different objects in the scans are labeled. It can be seen that with registration the density of scan points across the regions become uniform, and individual objects get sampled by more points.

5.6 Conclusion

Registration of scans not only aggregates points from different scans taken from different locations of a moving robot, but also results in robot localization within the environment. The speed of execution however is related quadratically to the number of points contained in each scan. Execution of the registration algorithms can thus be accelerated by selecting a set of key points from a scan that sample robust features instead of full point set. A rigid transformation obtained using only the key point set is applied to transform all the points in the scan against a model scan. In this chapter, unevenness with simple selection criteria is used to select enough number of points for registration. Registration using key points selected using unevenness not only accelerates the registration but also improves the accuracy of registration. Points are selected without explicit ground detection or feature modeling. The success rate of registration is high in spite of the algorithm not being provided with any initial guess of relative poses between the scans. The selection of key points is based on selecting the vertical features in an environment, which form robust features. The key point selection threshold (ξ) is needed to be tuned based on the environment such that it selects enough number of points to perform registration. While the work details about registration using the ICP algorithm in detail, similar results can be replicated using the Generalized ICP (GICP) and the Normal Distribution Transform (NDT).

Chapter 6

Conclusions and Outlook

Developing reliable perception of the surrounding environments in three dimensions is the most important requirement for building autonomous capabilities in an outdoor mobile robot. It is also important for the developed algorithms to execute on-board in real time. This allows the robot to quickly respond to the dynamically changing environments. This work is developed using the Multi-beam rotating laser scanners that are currently used with most autonomous vehicles. A novel measure called unevenness is developed for the purpose of improving robot perception by exploiting the ordered nature in which these sensors scan the environments. Algorithms developed around unevenness contribute to multiple stages of robot navigation in developing perception when no prior maps of the environment are available to the robot.

Inputs to the developed algorithms are the ordered range measurements from the three dimensional (3D) laser scanners. These sensors provide precise range measurements by sampling the surrounding environment in quick time. Sparse data in the radial direction may result in lasers not intercepting smaller obstacles at far away distance. They may only get detected when the robot moves closer to them.

However, reasonable size of objects which are more than 4cm high are detected till 10m range. Given the quick nature of scanning and general operating speed of autonomous mobile robots, this presents no concern for safe navigation. The data, on which the algorithms are applied, have not been corrected for motion of the robot. The algorithms for perception are shown to work without problem as the relative motion between the neighbouring points in a scan is very minute. However, when mapping the points into the world coordinates, the motion should be compensated.

The presented work develops around a notion of unevenness that is computed using the ratio of the measured and the expected range difference between the neighbouring points of an ordered point cloud. Detection of smallest of discontinuities in the direction of evaluation of the scanned points, in spite of terrain slopes or sensor tilts caused by robot motion, is the forte of unevenness. This robustness is achieved because of unevenness using the measured ranges for estimating expected range differences between neighbouring points and not depending on the absolute pitch angles. Calculation of unevenness is simple, and requires very simple preprocessing steps for conditioning raw sensor data for noise reduction using a simple median filter. In order to maintain the order of range data, very few points that return no ranges because of specular reflections are assigned ranges of their nearest neighbours in the transverse direction.

Algorithms using unevenness process the range data at a point level. Unevenness is first used for detection of obstacles around the robot. Ability of unevenness to detect smallest of surface discontinuities allowed detection of clear boundaries between objects and ground and between two nearby objects. This clear separation is advantageous in not only detecting smaller objects but in also using them as features for navigation. Analysing the unevenness field surrounding a robot by assigning unevenness values to individual points, a reasonable policy for setting thresholds on unevenness for our mobile robot is arrived at. Thresholds can be tuned according to the nature of the environment and the ability of the robot. Obstacle edges are also detected along the transverse direction to detect edges that are radially aligned and those missed by checking unevenness only in the radial direction. Considering the point density in the transverse direction, the points within one degree of rotation are binned into a grid cell. Traversable region is then detected as a connectivity graph by connecting all the non-obstacle cells using region growing by starting from an assured ground point in front of the robot.

Unevenness contributes to segmentation of important discernible objects in the individual scans obtained while the robot moves. Objects are segmented by using the standard region growing algorithm using the neighbourhood information present in the ordered point cloud. Growth of a segment is regulated using the unevenness values of the points calculated in the radial direction, as also by the detection of edges using unevenness in the transverse direction. The ability of unevenness in segmenting small surface changes even within an object is demonstrated. It has been assumed that the majority of relevant objects to be segmented are nearly vertical in nature, thus limiting the range of unevenness values for growing objects. Results show that unevenness successfully segments objects which are very close to one another. As in the case of obstacle detection, the thresholds that restrict the growth of an object using region growing can be tuned according to the fineness to which the objects are desired to be segmented. Since the point wise segmentation using unevenness clearly defines the object boundaries, it becomes advantageous to use static objects as landmarks across the scans. Also, because of fast scanning, moving dynamic objects like vehicles and pedestrians are also properly segmented. Dynamic objects recognised as single segments can be tracked for their motion across the scans. This approach for segmentation however suffers when segmenting porous objects like tree tops. Lack of continuity in unevenness values prevents the whole tree canopy to be recognised as a single segment. However, the ability to properly detect the rigid trunk of a tree, which can be used as landmarks instead of the entire tree, has been demonstrated.

Unevenness has been further extended for accelerating scan registration between the successive scans that are collected from different locations from a moving robot. Scan registration is performed by using only the selected number of points instead of the entire point cloud. Points sampling the features that are likely to result in the correct rigid transformation between the two scans are input to the standard registration algorithms while filtering out points mostly belonging to ground that are likely to cause convergence to local minima. Results also show that selection of key points using unevenness not only speeds up registration, but also improves the registration accuracy. Quicker registration makes it easier to quickly aggregate scans from a moving robot to overcome data sparsity. This approach is validated using the experimental results with reasonable ground truth information. Results show proper registration for distances between the scans of up to $8m$ without providing the initial estimate between them. Again, the thresholds on unevenness for selecting the key points can be tuned to adapt to the nature of the operating environments. For an urban environment, a narrow unevenness window around one selects only the points belonging to perfectly vertical objects. Selection window can be relaxed to a certain degree in natural non-urban environments to include points from objects with curved surfaces. Window size is determined primarily to select sufficient number of points to be input to the registration algorithms. Selection of the key points required no explicit detection of the objects or features.

Proposed methods are evaluated using scans taken from a Velodyne HDL-32 laser scanner placed on an outdoor mobile robot and taken to different locations across our campus. Unevenness at different ranges are evaluated for setting thresholds by using data where obstacle boxes of known heights are placed on the lab terrace that has a near level surface. In addition, unevenness has been computed for data collected using a Sick 2D laser scanner using a stop-scan-go method. Here the thresholds are slightly modified to take care of changes in scan angles.

There are some drawbacks with the approaches using unevenness. Primarily the method needs the data to be ordered and to have information on the angles at which the lasers are fired when collecting the data. Unevenness being very sensitive and dependent on the pitch angle difference is needed to be accurately provided with these values using proper calibration techniques. Some sensors or the publicly available data sets may not provide such accurate information. Since unevenness is computed across data obtained from a single laser scan with range and angle information, it is not possible to apply unevenness on data in XYZ format. Unevenness also cannot be used on aggregated point clouds, something which grid based methods are capable of handling. Sparse data from a single scan does not sample some objects, particularly the smaller ones at faraway distances. A surface edge or the boundary between two objects is also sometimes missed because of this. For example, a vehicle kept under a tree could get segmented along with the tree when one point falls on the vehicle and the next on the tree and both are in the same vertical plane. But, in general, all rigid objects are properly segmented. Also, the spacing between the vertical points in the sparse data sometimes leads to registration errors in the Z direction where a wrong horizontal line is selected for correspondence. The thresholds using unevenness depend on the nature of operating environment and the type of vehicle used and is needed to be tuned accordingly and is not constants. Although unevenness is computed for data from fast moving vehicles without motion correction (as the displacement offset between neighbouring points is small due to fast scanning), it will be interesting to see if unevenness is significantly affected for very fast moving vehicles when there is appreciable offset between the neighbouring points. But given all these drawbacks, the method is advantageous to use since the type of the vehicle and the nature of the operating environment is known beforehand. Velodyne lidars are also commonly used with mobile robots, and the angular information of the lasers is also known along with the factory provided calibration angles, which can be further tuned using other standard calibration algorithms. Even when the data is sparse faraway, since the scans are acquired in quick time most significant objects are detected in time to cause any harm to the robot.

Ideally, the unevenness based algorithms need to be compared extensively with existing algorithms on standard datasets available on the Internet. However, the standard datasets generally report point cloud data in cartesian world coordinates. As a result they cannot be used directly to compute unevenness. Unevenness is computed from range data at specified angular intervals in the sensor coordinate system, and that is available directly from a 3D laser scanner like Velodyne. This is the main reason for reporting experimental results in this thesis from range data collected in our campus. Exceptions are datasets made available as examples by Velodyne in required format, which we have analysed in Figure 3.22.

Another point to note is that this thesis primarily introduces the concept of unevenness and demonstrates how it can be used to advantage in different aspects of navigation, viz., obstacle detection, segmentation and scan registration. Because of spreading our studies over these three disparate areas, an extensive study of a single area (comparing with existing methods on some standard data sets) has not been possible. Such detailed studies and comparisons of unevenness based method with existing algorithms for each individual area can possibly be taken up separately as future work.

Despite successful implementation of the proposed algorithms in real time on experimental data, the concept of unevenness has further potential to be exploited for contributing to the field of robot navigation. While this work used the calibrated laser pointing angles provided by the manufacturer, unevenness can intuitively be used for calibration of exact pointing angles from the data taken on a level ground without requiring any external targets. When the pitch angle difference between two consecutive lasers is different from the predefined values, the resultant unevenness value on a level ground deviates from the expected value. Methods can thus be formalized to perform intrinsic calibration of laser beams pointing angles.

Methods using unevenness in this work primarily processed single scans to detect features or key points. Unevenness could be extended for processing multiple scans for Detection and Tracking of Moving Objects (DATMO). This allows detection of dynamic objects to be tracked in the robot's environment. Robot trajectories can be planned according to the motion of traffic participants. More importantly, points belonging to dynamic objects can be filtered out for scan registration. Unevenness can be extended for use in SLAM by detecting robust landmarks in a scan and also by using unevenness signatures in successive scans. It must be noted that unevenness varies with range for a given obstacle size which should be compensated for when checking correspondence across scans. Unevenness can also be exploited to bring out generic characteristics of commonly found objects in outdoor environments. For example, human figures exhibit fixed number of surface variations across their bodies, so do trees and other objects. They can be generically labelled to generate semantic maps using advanced computer science concepts in machine learning. Practically an entire navigation scheme can be developed around the notion of unevenness.

Bibliography

[1] Roland Siegwart, Illah Reza Nourbakhsh, and Davide Scaramuzza. *Introduction to autonomous mobile robots*. MIT press, 2004.

[2] Ingemar Cox and John Leonard. Modeling a dynamic environment using a bayesian multiple hypothesis approach. *Artificial Intelligence*, 66(2):311–344, April 1994. ISSN 0004-3702. doi: 10.1016/0004-3702(94)90029-9.

[3] Michael Montemerlo, Jan Becker, Suhrid Bhat, Hendrik Dahlkamp, Dmitri Dolgov, Scott Ettinger, Dirk Hähnel, Tim Hilden, Gabe Hoffmann, Burkhard Huhnke, Doug Johnston, Stefan Klumpp, Dirk Langer, Anthony Levandowski, Jesse Levinson, Julien Marcil, David Orenstein, Johannes Paefgen, Isaac Penny, Anna Petrovskaya, Mike Pflueger, Ganymed Stanek, David Stavens, Antone Vogt, and Sebastian Thrun. Junior: The stanford entry in the urban challenge. *Journal of Field Robotics*, 25(9):569–597, 2008. ISSN 1556-4967. doi: 10.1002/rob.20258.

[4] Ann Dunkin. Automated guided vehicle systems: An introduction. *Industrial Engineering*, 26:47–53, 1994.

[5] Toshihiro Tsumura. Survey of automated guided vehicle in a Japanese factory. In *1986 IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, pages 1329–1334, Apr 1986. doi: 10.1109/ROBOT.1986.1087602.

[6] Sebastian Thrun, Mike Montemerlo, Hendrik Dahlkamp, David Stavens, An-

drei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, Kenny Lau, Celia Oakley, Mark Palatucci, Vaughan Pratt, Pascal Stang, Sven Strohband, Cedric Dupont, Lars-Erik Jendrossek, Christian Koelen, Charles Markey, Carlo Rummel, Joe van Niekerk, Eric Jensen, Philippe Alessandrini, Gary Bradski, Bob Davies, Scott Ettinger, Adrian Kaehler, Ara Nefian, and Pamela Mahoney. Stanley: The robot that won the darpa grand challenge. *Journal of Field Robotics*, 23(9):661–692, 2006. ISSN 1556-4967. doi: 10.1002/rob.20147.

[7] Chris Urmson, Joshua Anhalt, Drew Bagnell, Christopher Baker, Robert Bittner, M. Clark, John Dolan, Dave Duggins, Tugrul Galatali, Chris Geyer, Michele Gittleman, Sam Harbaugh, Martial Hebert, Thomas Howard, Sascha Kolski, Alonzo Kelly, Maxim Likhachev, Matt McNaughton, Nick Miller, Kevin Peterson, Brian Pilnick, Raj Rajkumar, Paul Rybski, Bryan Salesky, Young-Woo Seo, Sanjiv Singh, Jarrod Snider, Anthony Stentz, William (Red) Whittaker, Ziv Wolkowicki, Jason Ziglar, Hong Bae, Thomas Brown, Daniel Demitrish, Bakhtiar Litkouhi, Jim Nickolaou, Varsha Sadekar, Wende Zhang, Joshua Struble, Michael Taylor, Michael Darms, and Dave Ferguson. Autonomous driving in urban environments: Boss and the urban challenge. *Journal of Field Robotics*, 25(8):425–466, 2008. ISSN 1556-4967. doi: 10.1002/rob.20255.

[8] In So Kweon and Takeo Kanade. Extracting topographic terrain features from elevation maps. *CVGIP: image understanding*, 59(2):171–182, 1994.

[9] Jose-Luis Blanco, Juan-Antonio Fernández-Madrigal, and Javier González. Toward a unified bayesian approach to Hybrid Metric–Topological SLAM. *IEEE Transactions on Robotics*, 24(2):259–270, April 2008. ISSN 1552-3098. doi: 10.1109/TRO.2008.918049.

[10] Nicola Tomatis. *Hybrid, Metric-Topological Representation for Localization*

and Mapping, pages 43–63. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. ISBN 978-3-540-75388-9. doi: 10.1007/978-3-540-75388-9_4.

[11] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. The dynamic window approach to collision avoidance. *IEEE Robotics Automation Magazine*, 4(1): 23–33, Mar 1997. ISSN 1070-9932. doi: 10.1109/100.580977.

[12] Felix von Hundelshausen, Michael Himmelsbach, Falk Hecker, Andre Mueller, and Hans-Joachim Wuensche. Driving with tentacles: Integral structures for sensing and motion. *Journal of Field Robotics*, 25(9):640–673, 2008. ISSN 1556-4967. doi: 10.1002/rob.20256.

[13] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. A probabilistic approach to concurrent mapping and localization for mobile robots. *Machine Learning*, 31(1):29–53, 1998. ISSN 1573-0565. doi: 10.1023/A:1007436523611.

[14] Sebastian Thrun et al. Robotic mapping: A survey. *Exploring artificial intelligence in the new millennium*, 1:1–35, 2002.

[15] Randall Smith and Peter Cheeseman. On the representation and estimation of spatial uncertainty. *The International Journal of Robotics Research*, 5(4): 56–68, 1986. doi: 10.1177/027836498600500404.

[16] Gamini Dissanayake, Paul Newman, Steven Clark, Hugh Durrant-Whyte, and M. Csorba. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Transactions on Robotics and Automation*, 17(3): 229–241, Jun 2001. ISSN 1042-296X. doi: 10.1109/70.938381.

[17] Sebastian Thrun. Particle filters in robotics. In *Proceedings of the Eighteenth Conference on Uncertainty in Artificial Intelligence*, UAI'02, pages 511–518, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc. ISBN 1-55860-897-4.

[18] Sebastian Thrun, Yufeng Liu, Daphne Koller, Andrew Y Ng, Zoubin Ghahramani, and Hugh Durrant-Whyte. Simultaneous localization and mapping with sparse extended information filters. *The International Journal of Robotics Research*, 23(7-8):693–716, 2004.

[19] Michael Montemerlo and Sebastian Thrun. *FastSLAM: A scalable method for the simultaneous localization and mapping problem in robotics*, volume 27. Springer, 2007. doi: 10.1007/978-3-540-46402-0.

[20] Feng Lu and Evangelos Milios. Robot pose estimation in unknown environments by matching 2D range scans. *Journal of Intelligent and Robotic Systems*, 18(3):249–275, 1997. doi: 10.1023/A:1007957421070.

[21] Tim Bailey. *Mobile robot localisation and mapping in extensive outdoor environments*. PhD thesis, The University of Sydney, 2002.

[22] Andreas Nüchter, Kai Lingemann, Joachim Hertzberg, and Hartmut Surmann. 6D SLAM—3D mapping outdoor environments. *Journal of Field Robotics*, 24 (8-9):699–722, 2007. ISSN 1556-4967. doi: 10.1002/rob.20209.

[23] Dorit Borrmann, Jan Elseberg, Kai Lingemann, Andreas Nüchter, and Joachim Hertzberg. Globally consistent 3d mapping with scan matching. *Robotics and Autonomous Systems*, 56(2):130 – 142, 2008. ISSN 0921-8890. doi: http://dx.doi.org/10.1016/j.robot.2007.07.002.

[24] Paul Besl and Neil McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, Feb 1992. ISSN 0162-8828. doi: 10.1109/34.121791.

[25] Oliver Wulf, Christian Brenneke, and Bernardo Wagner. Colored 2d maps for robot navigation with 3d sensor data. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 3, pages 2991–2996, Sendai, Japan, Sept 2004. doi: 10.1109/IROS.2004.1389864.

[26] Hans Moravec and Alberto Elfes. High resolution maps from wide angle sonar. In *1985 IEEE International Conference on Robotics and Automation (IRCA)*, volume 2, pages 116–121, Mar 1985. doi: 10.1109/ROBOT.1985.1087316.

[27] Alberto Elfes. Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57, June 1989. ISSN 0018-9162. doi: 10.1109/2.30720.

[28] Y. Roth-Tabak and Ramesh Jain. Building an environment model using depth information. *Computer*, 22(6):85–90, June 1989. ISSN 0018-9162. doi: 10.1109/2.30724.

[29] Martial Herbert, C. Caillas, Eric Krotkov, In So Kweon, and Takeo Kanade. Terrain mapping for a roving planetary explorer. In *1989 IEEE International Conference on Robotics and Automation (ICRA)*, pages 997–1002, Scottsdale, AZ, May 1989. doi: 10.1109/ROBOT.1989.100111.

[30] Martin Buehler, Karl Iagnemma, and Sanjiv Singh. *The DARPA Urban Challenge: Autonomous Vehicles in City Traffic*. Springer Publishing Company, Incorporated, 1st edition, 2009. ISBN 3642039901, 9783642039904. doi: 10.1007/978-3-642-03991-1.

[31] Chau Nguyen. Constructing Drivability Maps from 3D Laser Range data. Master's thesis.

[32] Michael Himmelsbach, Felix von Hundelshausen, and Hans-Joachim Wuensche. Fast segmentation of 3D point clouds for ground vehicles. In *2010 IEEE Intelligent Vehicles Symposium (IV)*, pages 560–565, SanDiego, CA, June 2010. doi: 10.1109/IVS.2010.5548059.

[33] Dirk Hähnel and Sebastian Thrun. 3D laser-based obstacle detection for autonomous driving. In *Abstract in IEEE Intelligent Robots and Systems IROS*, 2008.

[34] Patrick Pfaff, Rudolph Triebel, and Wolfram Burgard. An efficient extension to elevation maps for outdoor terrain mapping and loop closing. *International Journal of Robotics Research*, 26(2):217–230, February 2007. ISSN 0278-3649. doi: 10.1177/0278364906075165.

[35] Rudolph Triebel, Patrick Pfaff, and Wolfram Burgard. Multi-level surface maps for outdoor terrain mapping and loop closing. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2276–2282, Beijing, China, Oct 2006. doi: 10.1109/IROS.2006.282632.

[36] Armin Hornung, Kai Wurm, Maren Bennewitz, Cyrill Stachniss, and Wolfram Burgard. Octomap: an efficient probabilistic 3d mapping framework based on octrees. *Autonomous Robots*, 34(3):189–206, 2013. ISSN 1573-7527. doi: 10.1007/s10514-012-9321-0.

[37] Donald Meagher. Geometric modeling using octree encoding. *Computer Graphics and Image Processing*, 19(2):129 – 147, 1982. ISSN 0146-664X. doi: 10.1016/0146-664X(82)90104-6.

[38] Jari Saarinen, Henrik Andreasson, Todor Stoyanov, and Achim Lilienthal. 3D normal distributions transform occupancy maps: An efficient representation for mapping in dynamic environments. *The International Journal of Robotics Research*, 32(14):1627–1644, 2013. doi: 10.1177/0278364913499415.

[39] Martin Magnusson, Achim Lilienthal, and Tom Duckett. Scan registration for autonomous mining vehicles using 3D-NDT. *Journal of Field Robotics*, 24(10): 803–827, 2007. ISSN 1556-4967. doi: 10.1002/rob.20204.

[40] Jesse Levinson and Sebastian Thrun. Robust vehicle localization in urban environments using probabilistic maps. In *2010 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4372–4378, Anchorage, May 2010. doi: 10.1109/ROBOT.2010.5509700.

[41] Klass Klasing, Daniel Althoff, Dirk Wollherr, and Martin Buss. Comparison of surface normal estimation methods for range sensing applications. In *2009 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3206–3211, Kobe, Japan, May 2009. doi: 10.1109/ROBOT.2009.5152493.

[42] Yanni Ioannou, Babak Taati, Ronni Harrap, and Michael Greenspan. Difference of normals as a multi-scale operator in unorganized point clouds. In *2012 Second International Conference on 3D Imaging, Modeling, Processing, Visualization Transmission (3DIMPVT)*, pages 501–508, Los Alamitos, CA, USA, Oct 2012. doi: 10.1109/3DIMPVT.2012.12.

[43] Frank Moosmann, Oliver Pink, and Christoph Stiller. Segmentation of 3D lidar data in non-flat urban environments using a local convexity criterion. In *2009 IEEE Intelligent Vehicles Symposium (IV)*, pages 215–220, Xi'an, China, June 2009. doi: 10.1109/IVS.2009.5164280.

[44] Radu Bogdan Rusu, Nico Blodow, Zoltan Csaba Marton, and Michael Beetz. Aligning point cloud views using persistent feature histograms. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3384–3391, Nice, France, Sept 2008. doi: 10.1109/IROS.2008.4650967.

[45] Giulio Reina, Mauro Bellone, Luigi Spedicato, and Nicola Ivan Giannoccaro. 3d traversability awareness for rough terrain mobile robots. *Sensor Review*, 34(2):220–232, 2014. doi: 10.1108/SR-03-2013-644.

[46] Mark Maimone, Yang Cheng, and Larry Matthies. Two years of visual odometry on the mars exploration rovers. *Journal of Field Robotics*, 24(3):169–186, 2007. ISSN 1556-4967. doi: 10.1002/rob.20184.

[47] Arturo Rankin, Andres Huertas, and Larry Matthies. Stereo-vision-based ter-

rain mapping for off-road autonomous navigation. In *SPIE Defense, Security, and Sensing*, volume 7332, Orlando, FL, 2009.

[48] Larry Matthies and Arturo Rankin. Negative obstacle detection by thermal signature. In *2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 1, pages 906–913 vol.1, Las Vegas, USA, Oct 2003. doi: 10.1109/IROS.2003.1250744.

[49] Aniket Murarka and Benjamin Kuipers. A stereo vision based mapping algorithm for detecting inclines, drop-offs, and obstacles for safe local navigation. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1646–1653, St. Louis, USA, Oct 2009. doi: 10.1109/IROS.2009.5354253.

[50] Nicola Bernini, Massimo Bertozzi, Luca Castangia, Marco Patander, and Mario Sabbatelli. Real-time obstacle detection using stereo vision for autonomous ground vehicles: A survey. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 873–878, Qingdao, China, Oct 2014. doi: 10.1109/ITSC.2014.6957799.

[51] Andreas Nüchter. *3D robotic mapping: the simultaneous localization and mapping problem with six degrees of freedom*, volume 52. Springer, 2009. doi: 10.1007/978-3-540-89884-9.

[52] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. A real-time algorithm for mobile robot mapping with applications to multi-robot and 3d mapping. In *2000 IEEE International Conference on Robotics and Automation (ICRA) Millennium Conference*, volume 1, pages 321–328, San Francisco, CA, 2000. doi: 10.1109/ROBOT.2000.844077.

[53] Andrew Howard, Dennis Wolf, and Gaurav Sukhatme. Towards 3D mapping in large urban environments. In *2004 IEEE/RSJ International Conference*

on Intelligent Robots and Systems (IROS), volume 1, pages 419–424, Sendai, Japan, Sept 2004. doi: 10.1109/IROS.2004.1389388.

[54] Rudolph Triebel. *Three-dimensional perception for mobile robots.* PhD thesis, University of Freiburg, 2007.

[55] Micheal Bosse, Robert Zlot, and Paul Flick. Zebedee: Design of a spring-mounted 3-D range sensor with application to mobile mapping. *IEEE Transactions on Robotics*, 28(5):1104–1119, Oct 2012. ISSN 1552-3098. doi: 10.1109/TRO.2012.2200990.

[56] Sebastian Thrun, Mark Diel, and Dirk Hähnel. Scan alignment and 3-D surface modeling with a helicopter platform. In *Field and Service Robotics*, pages 287–297. Springer, 2003. doi: 10.1007/10991459_28.

[57] Julian Ryde and Huosheng Hu. 3D laser range scanner with hemispherical field of view for robot navigation. In *2008 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, pages 891–896, Xian, China, July 2008. doi: 10.1109/AIM.2008.4601779.

[58] Christian Brenneke, Oliver Wulf, and Bernando Wagner. Using 3D laser range data for SLAM in outdoor environments. In *2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 1, pages 188–193, Las Vegas, USA, Oct 2003. doi: 10.1109/IROS.2003.1250626.

[59] Sujit Kuthirummal, Aveek Das, and Supun Samarasekera. A graph traversal based algorithm for obstacle detection using lidar or stereo. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3874–3880, San Francisco, USA, Sept 2011. doi: 10.1109/IROS.2011.6094685.

[60] Dennis Wolf, Gaurav Sukhatme, Dieter Fox, and Wolfram Burgard. Autonomous terrain mapping and classification using hidden markov models.

In *2005 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2026–2031, Barcelona, Spain, April 2005. doi: 10.1109/ROBOT.2005.1570411.

[61] Andreas Nüchter, Kai Lingemann, and Joachim Hertzberg. Cached k-d tree search for ICP algorithms. In *Sixth International Conference on 3-D Digital Imaging and Modeling (3DIM)*, pages 419–426, Washington, DC, USA, Aug 2007. doi: 10.1109/3DIM.2007.15.

[62] Hartmut Surmann, Kai Lingemann, Andreas Nüchter, and Joachim Hertzberg. A 3D laser range finder for autonomous mobile robots. In *32 International Symposium on Robotics(ISR)*, volume 19, pages 153–158, Zurich, Switzerland, 2001.

[63] Dirk Hähnel, Dominik Schulz, and Wolfram Burgard. Map building with mobile robots in populated environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 1, pages 496–501, Lausanne, Switzerland, 2002. doi: 10.1109/IRDS.2002.1041439.

[64] Oliver Wulf and Bernardo Wagner. Fast 3D scanning methods for laser measurement systems. In *International conference on control systems and computer science (CSCS)*, pages 2–5, Bucharest, Romania, 2003.

[65] VelodyneLidar. *Velodyne HDL-32e manual.*

[66] Claude Elwood Shannon. Communication in the presence of noise. *Proceedings of the IRE*, 37(1):10–21, 1949.

[67] Naveed Muhammad and Siman Lacroix. Calibration of a rotating multi-beam lidar. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5648–5653, Toulouse, France, Oct 2010. doi: 10.1109/IROS.2010.5651382.

[68] Mark Sheehan, Alastair Harrison, and Paul Newman. Self-calibration for a 3D laser. *The International Journal of Robotics Research*, 31(5):675–687, 2011.

[69] Jesse Levinson and Sebastian Thrun. Unsupervised calibration for multi-beam lasers. In *Experimental Robotics*, pages 179–193. Springer, 2014. doi: 10.1007/978-3-642-28572-1_13.

[70] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. *Communications of the ACM*, 18(9):509–517, September 1975. ISSN 0001-0782. doi: 10.1145/361002.361007.

[71] Carlo Tomasi and Roberto Manduchi. Bilateral filtering for gray and color images. In *Sixth IEEE International Conference on Computer Vision (ICCV)*, pages 839–846, Bombay, India, Jan 1998. doi: 10.1109/ICCV.1998.710815.

[72] Norbert Wiener. *Extrapolation, interpolation, and smoothing of stationary time series*, volume 2. MIT press Cambridge, 1949.

[73] Rafael Gonzalez and Richard Woods. *Digital Image Processing (3rd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 2006. ISBN 013168728X.

[74] Panagiotis Papadakis. Terrain traversability analysis methods for unmanned ground vehicles: A survey. *Engineering Applications of Artificial Intelligence*, 26(4):1373–1385, April 2013. ISSN 0952-1976. doi: 10.1016/j.engappai.2013.01.006.

[75] Karl Iagnemma, Hassan Shibly, and Steven Dubowsky. On-line terrain parameter estimation for planetary rovers. In *2002 IEEE International Conference on Robotics and Automation (ICRA)*, volume 3, pages 3142–3147, St. Paul, MN, USA, 2002. doi: 10.1109/ROBOT.2002.1013710.

[76] Àngel Santamaria-Navarro, Ernesto Teniente, Martí Morta, and Juan Andrade-Cetto. Terrain classification in complex three-dimensional outdoor

environments. *Journal of Field Robotics*, 32(1):42–60, 2015. ISSN 1556-4967. doi: 10.1002/rob.21521.

[77] Roberto Manduchi, Andres Castano, Ashit Talukder, and Larry Matthies. Obstacle detection and terrain classification for autonomous off-road navigation. *Autonomous Robots*, 18(1):81–102, 2005. ISSN 1573-7527. doi: 10.1023/B:AURO.0000047286.62481.1d.

[78] Michael Shneier, Tommy Chang, Tsai Hong, Will Shackleford, Roger Bostelman, and James Albus. Learning traversability models for autonomous mobile vehicles. *Autonomous Robots*, 24(1):69–86, 2008. ISSN 1573-7527. doi: 10.1007/s10514-007-9063-6.

[79] Bertrand Douillard, James Underwood, Noah Kuntz, Vsevolod Vlaskine, Alastair Quadros, Peter Morton, and Alon Frenkel. On the segmentation of 3d lidar point clouds. In *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2798–2805, Shanghai, China, May 2011. doi: 10.1109/ICRA.2011.5979818.

[80] Jean-François Lalonde, Nicolas Vandapel, Daniel 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. ISSN 1556-4967. doi: 10.1002/rob.20134.

[81] Johannes Pellenz, Frank Neuhaus, Denis Dillenberger, David Gossow, and Dietrich Paulus. Mixed 2D/3D perception for autonomous robots in unstructured environments. In *RoboCup 2010: Robot Soccer World Cup XIV*, pages 303–313. Springer, 2011.

[82] Sören Kammel, Julius Ziegler, Benjamin Pitzer, Moritz Werling, Tobias Gindele, Daniel Jagzent, Joachim Schröder, Michael Thuy, Matthias Goebl, Felix von Hundelshausen, Oliver Pink, Christian Frese, and Christoph Stiller.

Team annieway's autonomous system for the 2007 DARPA urban challenge. *Journal of Field Robotics*, 25(9):615–639, 2008. ISSN 1556-4967. doi: 10.1002/rob.20252.

[83] Tongtong Chen, Bin Dai, Ruili Wang, and Daxue Liu. Gaussian-process-based real-time ground segmentation for autonomous land vehicles. *Journal of Intelligent & Robotic Systems*, 76(3):563–582, 2014. ISSN 1573-0409. doi: 10.1007/s10846-013-9889-4.

[84] Dmitriy Korchev, Shinko Cheng, and Yuri Owechko. On real-time lidar data segmentation and classification. In *International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV)*, page 1, Las Vegas, USA, 2013. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp).

[85] Tommy Chang, Tsai-Hong, Steve Legowik, and Marilyn Abrams. Concealment and obstacle detection for autonomous driving. In *International Association of Science and Technology for Development Robotics and Application*, Santa Barbara, USA,, 1999.

[86] Dominik Steinhauser, Oliver Ruepp, and Darius Burschka. Motion segmentation and scene classification from 3D LIDAR data. In *2008 IEEE Intelligent Vehicles Symposium (IV)*, pages 398–403, Eindhoven, Netherlands, June 2008. doi: 10.1109/IVS.2008.4621281.

[87] Ryan Morton and Edwin Olson. Positive and negative obstacle detection using the HLD classifier. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1579–1584, San Francisco, USA, Sept 2011. doi: 10.1109/IROS.2011.6095142.

[88] Tahir Rabbani and Frank Van Den Heuvel. Efficient hough transform for

automatic detection of cylinders in point clouds. *ISPRS WG III/3, III/4*, 3: 60–65, 2005.

[89] Lucian Goron, Levente Tamas, Istvan Reti, and Gheorghe Lazea. 3D Laser scanning system and 3D segmentation of urban scenes. In *2010 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*, volume 1, pages 1–5, Cluj-Napoca, Romania, May 2010. doi: 10.1109/AQTR.2010.5520872.

[90] Shrihari Vasudevan, Fabio Ramos, Eric Nettleton, and Hugh Durrant-Whyte. Gaussian process modeling of large-scale terrain. *Journal of Field Robotics*, 26(10):812–840, 2009. ISSN 1556-4967. doi: 10.1002/rob.20309.

[91] Arun Das and Steven Waslander. Scan registration using segmented region growing NDT. *The International Journal of Robotics Research*, 33(13):1645–1663, 2014. doi: doi:10.1177/0278364914539404.

[92] Chunzhao Guo, Wataru Sato, Long Han, Seiichi Mita, and David McAllester. Graph-based 2d road representation of 3d point clouds for intelligent vehicles. In *2011 IEEE Intelligent Vehicles Symposium (IV)*, pages 715–721, Baden-Baden, Germany, June 2011. doi: 10.1109/IVS.2011.5940502.

[93] Dragomir Anguelov, Benjamin Taskarf, Vassil Chatalbashev, Daphne Koller, Dinkar Gupta, Geremy Heitz, and Andrew Ng. Discriminative learning of markov random fields for segmentation of 3D scan data. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 2, pages 169–176, San Diego, CA, USA, June 2005. doi: 10.1109/CVPR.2005.133.

[94] Arnab Sinha and Panagiotis Papadakis. Mind the gap: detection and traversability analysis of terrain gaps using LIDAR for safe robot navigation. *Robotica*, 31(07):1085–1101, 2013. doi: 10.1017/S0263574713000349.

[95] Jacoby Larson, Mohan Trivedi, and Michael Bruch. Off-road terrain traversability analysis and hazard avoidance for ugvs. Technical report, DTIC Document, 2011.

[96] Wolfram Burgard. *Techniques for 3D Mapping, Slides from SLAM Summer school 2009*, 2009. URL `http://www.acfr.usyd.edu.au/pdfs/projects/SLAM%20Summer%20School/Wolfram%20Burgard.pdf`.

[97] Rolf Adams and Leanne Bischof. Seeded region growing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(6):641–647, Jun 1994. ISSN 0162-8828. doi: 10.1109/34.295913.

[98] Robert Haralock and Linda Shapiro. *Computer and robot vision.* Addison-Wesley Longman Publishing Co., Inc., 1991.

[99] Bertrand Douillard, James Underwood, Vsevolod Vlaskine, Alastair Quadros, and Sanjiv Singh. A pipeline for the segmentation and classification of 3D point clouds. In *2010 International Symposium on Experimental Robotics (ISER)*, New Delhi and Agra, India, Dec 2010.

[100] Michael Himmelsbach, Thorsten Luettel, and Hans-Joachim Wuensche. Real-time object classification in 3D point clouds using point feature histograms. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 994–1000, St. Louis, MO, Oct 2009. doi: 10.1109/IROS.2009.5354493.

[101] Adam Hoover, Gillian Jean-Baptiste, Xiaoyi Jiang, Patrick Flynn, Horst Bunke, Dmitry Goldgof, Kevin Bowyer, David Eggert, Andrew Fitzgibbon, and Robert Fisher. An experimental comparison of range image segmentation algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(7):673–689, July 1996. ISSN 0162-8828. doi: 10.1109/34.506791.

[102] Klaas Klasing, Dirk Wollherr, and Martin Buss. Realtime segmentation of range data using continuous nearest neighbors. In *2009 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2431–2436, Kobe, Japan, May 2009. doi: 10.1109/ROBOT.2009.5152498.

[103] Klaas Klasing, Dirk Wollherr, and Martin Buss. A clustering method for efficient segmentation of 3D Laser data. In *2008 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4043–4048, Pasadena, CA, May 2008. doi: 10.1109/ROBOT.2008.4543832.

[104] Radu Bogdan Rusu and Steve Cousins. 3D is here: Point cloud library (pcl). In *2011 IEEE International Conference on Robotics and Automation, (ICRA)*, pages 1–4, Shanghai, China, May 2011. doi: 10.1109/ICRA.2011.5980567.

[105] Paul Besl and Ramesh Jain. Segmentation through variable-order surface fitting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10 (2):167–192, Mar 1988. ISSN 0162-8828. doi: 10.1109/34.3881.

[106] Anupama Jagannathan and Eric Miller. Three-dimensional surface mesh segmentation using curvedness-based region growing approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(12):2195–2204, Dec 2007. ISSN 0162-8828. doi: 10.1109/TPAMI.2007.1125.

[107] Tahir Rabbani, Frank Van Den Heuvel, and George Vosselmann. Segmentation of point clouds using smoothness constraint. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36(5): 248–253, 2006.

[108] Luciano Spinello, Rudolph Triebel, and Roland Siegwart. Multimodal detection and tracking of pedestrians in urban environments with explicit ground plane extraction. In *2008 IEEE/RSJ International Conference on Intelligent*

Robots and Systems (IROS), pages 1823–1829, Nice, France, Sept 2008. doi: 10.1109/IROS.2008.4651109.

[109] Aleksey Golovinskiy and Thomas Funkhouser. Min-cut based segmentation of point clouds. In *2009 IEEE 12th International Conference on Computer Vision Workshops (ICCV)*, pages 39–46, Kyoto, Japan, Sept 2009. doi: 10.1109/ICCVW.2009.5457721.

[110] Suchet Bargoti, James Underwood, Juan Nieto, and Salah Sukkarieh. A pipeline for trunk detection in trellis structured apple orchards. *Journal of Field Robotics*, 32(8):1075–1094, 2015. ISSN 1556-4967. doi: 10.1002/rob.21583.

[111] Cyril Goutte and Eric Gaussier. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In *2005 27th European Conference on IR Research, Advances in Information Retrieval (ECIR)*, pages 345–359, Santiago de Compostela, Spain, 2005. doi: 10.1007/978-3-540-31865-1_25.

[112] Szymon Rusinkiewicz and Marc Levoy. Efficient variants of the icp algorithm. In *2001 Third International Conference on 3-D Digital Imaging and Modeling (3DIM)*, pages 145–152, Quebec City, Canada, 2001. IEEE. doi: 10.1109/IM.2001.924423.

[113] Berthold Horn. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America A*, 4(4):629–642, 1987. doi: 10.1364/JOSAA.4.000629.

[114] Greg Turk and Marc Levoy. Zippered polygon meshes from range images. In *21st Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '94, pages 311–318, New York, NY, USA, 1994. ACM. ISBN 0-89791-667-0. doi: 10.1145/192161.192241.

[115] Takeshi Masuda, Katsuhiko Sakaue, and Naokazu Yokoya. Registration and integration of multiple range images for 3-D model construction. In *13th International Conference on Pattern Recognition (ICPR)*, volume 1, pages 879–883 vol.1, Vienna, Aug 1996. doi: 10.1109/ICPR.1996.546150.

[116] Sebastian Weik. Registration of 3-d partial surface models using luminance and depth information. In *Proceedings International Conference on Recent Advances in 3-D Digital Imaging and Modeling*, pages 93–100, Ottawa, Canada, May 1997. doi: 10.1109/IM.1997.603853.

[117] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image Vision Computing*, 10(3):145–155, April 1992. ISSN 0262-8856. doi: 10.1016/0262-8856(92)90066-C.

[118] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, 13(2):119–152, 1994. ISSN 1573-1405. doi: 10.1007/BF01427149.

[119] Gérard Blais and Martin Levine. Registering multiview range data to create 3d computer objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):820–824, Aug 1995. ISSN 0162-8828. doi: 10.1109/34.400574.

[120] Soon-Yong Park and Murali Subbarao. An accurate and fast point-to-plane registration technique. *Pattern Recognition Letters*, 24(16):2967 – 2976, 2003. ISSN 0167-8655. doi: 10.1016/S0167-8655(03)00157-0.

[121] Andrea Censi. Scan matching in a probabilistic framework. In *2006 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2291–2296, Beijing, China, May 2006. doi: 10.1109/ROBOT.2006.1642044.

[122] Kari Pulli. Multiview registration for large data sets. In *1999 Second In-*

ternational Conference on 3-D Digital Imaging and Modeling (3DIM), pages 160–168. IEEE, 1999. doi: 10.1109/IM.1999.805346.

[123] Francois Pomerleau. *Applied registration for robotics. Methodology and tools for ICP-like algorithms.* PhD thesis, ETH Zurich, 2013.

[124] Aleksandr Segal, Dirk Hähnel, and Sebastian Thrun. Generalized-icp. In *Robotics: Science and Systems (RSS)*, Seattle, USA, June 2009. doi: 10.15607/RSS.2009.V.021.

[125] Gaurav Pandey, James McBride, Silvio Savarese, and Ryan Eustice. Toward mutual information based automatic registration of 3d point clouds. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2698–2704, Vilamoura, Portugal, Oct 2012. doi: 10.1109/IROS.2012.6386053.

[126] David Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. ISSN 1573-1405. doi: 10.1023/B:VISI.0000029664.99615.94.

[127] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (SURF). *Computer Vision and Image Understanding*, 110(3): 346 – 359, 2008. ISSN 1077-3142. doi: http://dx.doi.org/10.1016/j.cviu.2007.09.014.

[128] Peter Biber and Wolfgang Straßer. The normal distributions transform: a new approach to laser scan matching. In *2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 3, pages 2743–2748, Las Vegas, USA, Oct 2003. doi: 10.1109/IROS.2003.1249285.

[129] Todor Stoyanov, Martin Magnusson, Henrik Andreasson, and Achim Lilienthal. Fast and accurate scan registration through minimization of the dis-

tance between compact 3d ndt representations. *The International Journal of Robotics Research*, 31(12):1377–1393, 2012.

[130] Ji Kim and Beom Lee. Robust and fast 3-D scan registration using normal distributions transform with supervoxel segmentation. *Robotica*, 34(07):1630–1658, 2016. doi: 10.1017/S0263574714002483.

[131] K.S. Arun, Thomas Huang, and Steven Blostein. Least-squares fitting of two 3-d point sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(5):698–700, Sept 1987. ISSN 0162-8828. doi: 10.1109/TPAMI.1987.4767965.

[132] Hanan Samet. *Foundations of Multidimensional and Metric Data Structures (The Morgan Kaufmann Series in Computer Graphics and Geometric Modeling)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005. ISBN 0123694469.

[133] Mobile Robotics Programming Toolkit. Mobile robotics programming toolkit, 2009. URL `http://www.mrpt.org/`.

[134] Bertrand Douillard, Alastair Quadros, Peter Morton, James P. Underwood, Mark De Deuge, S. Hugosson, M. Hallström, and Tim Bailey. Scan segments matching for pairwise 3d alignment. In *2012 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3033–3040, SaintPaul, MN, May 2012. doi: 10.1109/ICRA.2012.6224788.