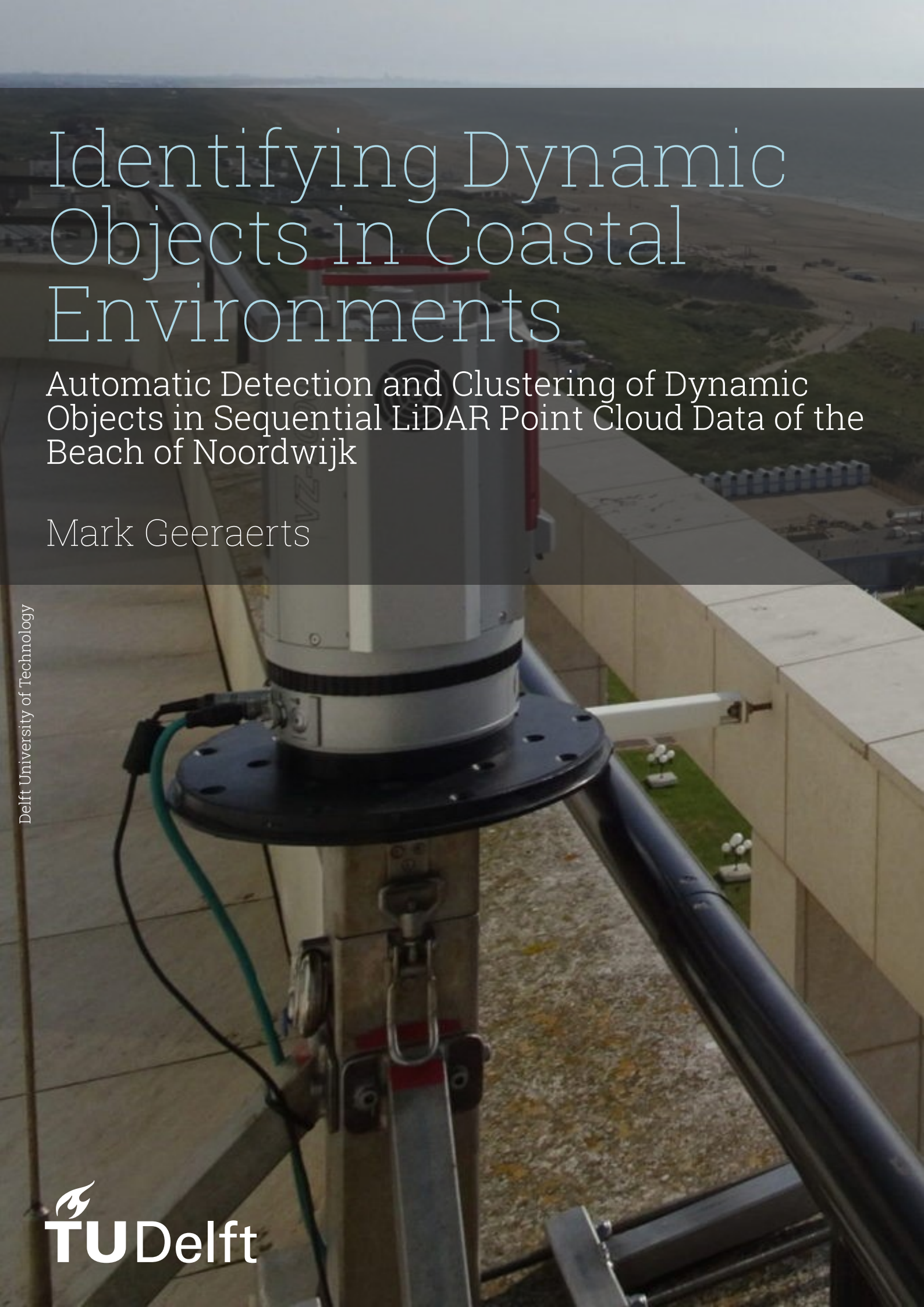
Identifying Dynamic Objects in Coastal Environments

Automatic Detection and Clustering of Dynamic Objects in Sequential LiDAR Point Cloud Data of the Beach of Noordwijk

Mark Geeraerts



Delft University of Technology

TU Delft

Identifying Dynamic Objects in Coastal Environments

Automatic Detection and Clustering of Dynamic Objects in Sequential LiDAR Point Cloud Data of the Beach of Noordwijk

by

Mark Geeraerts

To obtain the degree of Bachelor of Science in Applied Earth Sciences at the Delft University of Technology, to be defended publicly on Tuesday July 8th, 2025 at 1:00 PM.

First Supervisor:	Dr. R.C. Lindenbergh
Second Supervisor:	Ir. D.C. Hulskemper
External Assessor:	Dr. S.E. Vos
Project Duration:	May, 2025 - July, 2025
Faculty:	Faculty of Civil Engineering and Geosciences, TU Delft

Cover: Coastal Laser Scanner on the balcony of a hotel in Noordwijk by CoastScan (n.d.).

TU Delft

Abstract

Identification of dynamic objects in sequential terrestrial Light Detection And Ranging (LiDAR) point cloud data is important for analyzing activity and usage of coastal environments. This research focuses on identifying non-geomorphological dynamic objects, such as people and bulldozers, in sequential terrestrial LiDAR point cloud data acquired by a permanent laser scanner installed in Noordwijk, the Netherlands. A workflow is proposed and demonstrated, consisting of three main components: ground and non-ground separation using a Cloth Simulation Filter, dynamic point detection through Cloud-to-Cloud comparison, and clustering of individual dynamic objects using Density-Based Spatial Clustering of Applications with Noise (DBSCAN). Parameter tuning is performed by evaluating all possible configurations within a defined range and validated against manually identified dynamic objects. For a week-long dataset, the error in the number of detected large dynamic objects is relatively low at 6.9%, whereas the error for small dynamic objects is higher at 23.0%, attributed to their proximity to the ground and to each other. On a point-to-point basis, the optimized configuration results in an average error of 13.5% for large dynamic objects and 33.7% for small dynamic objects with respect to a reference set. A sensitivity analysis using a Monte Carlo simulation with normally distributed parameter variations around the tuned values demonstrates robustness to moderate parameter fluctuations, particularly for larger dynamic objects, which show a standard deviation of 0.07 detected objects, while smaller objects show greater variability with a standard deviation of 0.34 detected objects. The application of the Cloth Simulation Filter adds value by excluding geomorphological processes, contributing to a reduction in error rate of approximately 95% for large objects and 62% for small objects. Overall, the presented workflow offers a robust automated approach for detecting dynamic objects on sandy beaches in LiDAR point cloud data, with demonstrated potential for scalable, long-term monitoring.

Preface

The following thesis is the result of seven weeks of dedicated work, completed as the final step toward obtaining my Bachelor's degree in Applied Earth Sciences at TU Delft. In this research, I explore the possibilities of automatically detecting and clustering dynamic objects in sequential LiDAR point cloud data of the beach of Noordwijk.

This topic was first introduced to me by Roderik Lindenbergh and Daan Hulskemper, and it quickly caught my interest, not only because it involved data from a familiar place, but also because the dataset included bulldozers, which stand out in point cloud data. At first, my aim was simply to detect these vehicles, as they seemed like interesting dynamic objects to identify. However, as my thesis progressed, I began to see the potential to detect a much wider range of objects. While I never managed to find any seals, rumored to be present in the dataset, I did encounter many other dynamic objects: ladders, crates, bicycles, people, and even dogs being walked along the beach. This allowed me to combine technical analysis with creative problem-solving, as different types of objects pose unique challenges for the detection process. Engaging with this LiDAR dataset turned out to be a rewarding experience, and over the course of seven weeks, I learned more than I had anticipated.

I would like to thank the people who made this thesis possible. In particular, I am grateful to Roderik Lindenbergh and Daan Hulskemper for their guidance throughout the project. Whether we were discussing if a detected object might be a cyclist or a horse rider, or addressing technical aspects of the methodology, the conversations were helpful and enjoyable, and their feedback and suggestions helped me get the most out of my thesis. I especially appreciated their approachable attitude, which made me feel welcome within the Geoscience and Remote Sensing department during the duration of my thesis. I would also like to thank Sander Vos for kindly agreeing to act as the external member of my thesis committee, and for being supportive and approachable throughout the process.

Mark Geeraerts
Delft, July 2025

Contents

1

Introduction

Coastal environments, such as sandy beaches along the Dutch coast, are dynamic environments, that have varying purposes such as recreation, wildlife habitat, coastal protection, and commercial activities. Understanding how these beaches are used, by whom, and to what extent, is essential for managing potential conflicts between users. To achieve this understanding, it is necessary to identify dynamic objects on the beach, such as people, and vehicles, that make use of the beach. This research focuses specifically on the beach of Noordwijk, where monitoring campaigns have been conducted using a permanent laser scanning installed at the top of a hotel overlooking the beach (see cover page) (CoastScan, n.d.). Previous research has investigated topics such as geomorphological and topographical change (Kuschnerus et al., 2024), and the development of dunes influenced by beach buildings (Vos et al., 2024). Less attention has been given however to the detection of dynamic objects within the Light Detection And Ranging (LiDAR) point cloud data, despite its importance for understanding beach usage. To address this gap, this research focuses on identifying dynamic objects, which are defined as a non-ground objects that shows change in position between sequential scans. The available data provides an opportunity to explore how dynamic objects can be detected through near-continuous terrestrial laser scanning, which methods are available to do so, and how the parameters driving these methods can be optimized.

1.1. Research Questions

This report raises and aims to answer the research question:

How can dynamic objects in sequential LiDAR point cloud data be automatically identified on sandy beaches?

To answer the research question, several supporting matters will be addressed:

1. How can near-continuous terrestrial LiDAR point cloud data be acquired?
2. What kind of dynamic objects are expected to be identified in LiDAR point cloud data of sandy beaches?
3. What are possible approaches for distinguishing dynamic objects in sequential LiDAR scans?
4. How can suitable parameters be determined for identifying dynamic objects in LiDAR point cloud data?
5. How can the performance of the detection of dynamic objects be evaluated?
6. How sensitive is the performance of dynamic object detection to variations in parameter settings?

1.2. Research Methods

This research will answer the research questions through a literature review and analysis of LiDAR point cloud data of the beach of Noordwijk. Sources are examined concerning near-continuous terrestrial LiDAR data acquisition, point cloud processing algorithms, detection methodologies, and clustering and filtering techniques. The available dataset is analyzed to gain insight into the types of dynamic objects present. The characteristics of the dataset will be described, discussing the variability and quality of the data. Methods are selected to perform dynamic object detection within the point cloud data, which are then applied to the dataset. The selected methods require certain parameters which need to be tuned, and therefore a tuning process is applied. The performance of the tuned parameters will be quantitatively evaluated on a reference set of dynamic objects, and the robustness of the results will be assessed through a sensitivity analysis. Method implementation, parameter tuning, and evaluation are performed in Python, while visualization is carried out in both Python and CloudCompare.

1.3. Report Structure

This report is structured as follows. Chapter 2 examines the scientific background, covering terrestrial LiDAR data acquisition, ground and non-ground separation techniques, dynamic object detection methods, clustering techniques for point cloud data, and nearest neighbor calculations. Chapter 3 describes the data and study area, providing an overview of the research area, outlining the data characteristics, and identifying dynamic objects present in the dataset. Chapter 4 outlines the methodology applied to the dataset. It begins with a workflow overview and justification of the methodological approach, followed by an explanation of the steps taken. These include data preprocessing, the concept and implementation of the Cloth Simulation Filter, Cloud-to-Cloud comparison, and the use of Density-Based Spatial Clustering of Applications with Noise (DBSCAN). Result optimization and evaluation are also discussed. Chapter 5 presents the results, highlighting the outcomes of parameter tuning, performance evaluation, and the findings of the sensitivity analysis. Chapter 6 contains the discussion, which reflects on the research by evaluating the parameter tuning process and performance analysis. It also discusses the selected parameters for various methods and evaluates the impact of non-ground point filtering on the results. Furthermore, it discusses the analysis of one month's worth of data and assesses the scalability and applicability of the proposed workflow. Finally, Chapter 7 will present the conclusion, answering the main and supporting research questions, and providing recommendations for future research.

2

Scientific Background

This chapter provides an overview of the principles and techniques relevant to this research. It begins with a discussion on near-continuous terrestrial LiDAR data acquisition in Section 2.1, covering its fundamental principles, advantages, and limitations. Section 2.2 then explores various methods for dynamic point detection. Ground and non-ground point separation techniques are examined in Section 2.3, followed by an overview of clustering techniques in Section 2.4. Finally, Section 2.5 addresses nearest neighbor calculations.

2.1. Near-Continuous Terrestrial LiDAR Data Acquisition

Light Detection And Ranging (LiDAR) is a laser scanning technique which uses laser pulses to determine distances between a laser scanner and its environment. There are various types of laser scanners, ranging from airborne laser scanning (ALS) to terrestrial laser scanning (TLS) systems. TLS is a ground-based method that is stationary (Vosselman & Maas, 2010). LiDAR scanners use Time-of-Flight (ToF) measurements, which allows the determination of the distance to a scanned object based on the time it takes for a light wave to travel from the scanner, reflect off the object, and return back to the scanner. This one-way-distance d to the object can be expressed as:

$$d = \frac{c \cdot t}{2n} \tag{2.1}$$

where c ($\approx 3.0 \times 10^8$ m/s) is the speed of light in vacuum, t is the travel time from the scanner to a target and back, and n (≈ 1.003) is the correction factor which is equal to refractive index, which depends on various factors, such as temperature, pressure, and humidity (Vosselman & Maas, 2010). Based on the distance obtained through ToF measurements, and the direction in which the laser pulse is sent, the location of the reflecting point can be determined. Sending out laser pulses in various directions will thus result in a 3D point cloud. TLS devices may be subjected to small tilt variations. To account for these variations, a time-dependent alignment matrix can be used to align sequential LiDAR scan data (Vos et al., 2023).

TLS systems produce a point cloud with an uneven point density, as density of the point cloud is a function of distance from the scanner; objects which are located closer to a scanner will be more densely covered than farther away, which may form challenges when processing the LiDAR data (Li et al., 2019). It should be noted that TLS generally provides a high point cloud density in comparison to other techniques such as ALS (Keskin et al., 2024). A limitation of TLS are line-of-sight restrictions. In environments with buildings for instance, line-of-sight restrictions can present a significant challenge, particularly when the research focus extends beyond obstructing structures. Since TLS can only capture what is directly visible, an unobstructed path between the scanner and the target is needed for complete data acquisition. Other factors which may impact the performance of TLS devices are instrumental errors, the laser-surface interaction and surface properties, environmental conditions, and the scan strategy (Muralikrishnan, 2021).

Permanent Laser Scanning (PLS) is a technique that employs a terrestrial laser scanner mounted at a fixed location for an extended period to repeatedly capture 3D scans. Each scan, or epoch, produces a point cloud representing the geometry and reflectivity of the scene at that moment. Over time, these sequential point clouds form a 4D dataset (3D + time) that captures not only changes in the terrain but also the movement of individual dynamic objects within the scene at high temporal frequency (hourly to weekly). This makes PLS suitable for monitoring dynamic processes and dynamic objects in environments such as beaches. However, the large and unstructured nature of the data requires processing methods to extract relevant information from the obtained near-continuous point cloud series (Kuschnerus, 2024).

2.2. Dynamic Point Detection Techniques

In this research, change refers to the significant spatial displacement or appearance of points in sequential LiDAR scans. Change is therefore closely related to dynamic points, which are defined as points that show notable positional differences or appear in one scan but not in the previous one. Kharroubi et al. (2022) has provided a comprehensive review of different methods which are suitable for detecting three dimensional change in point cloud data. These methods can be subdivided into three categories, namely, so-called standard methods, handcrafted machine learning methods and deep learning methods. These methods can be divided into various sub-methods (see Figure 2.1) (Kharroubi et al., 2022).

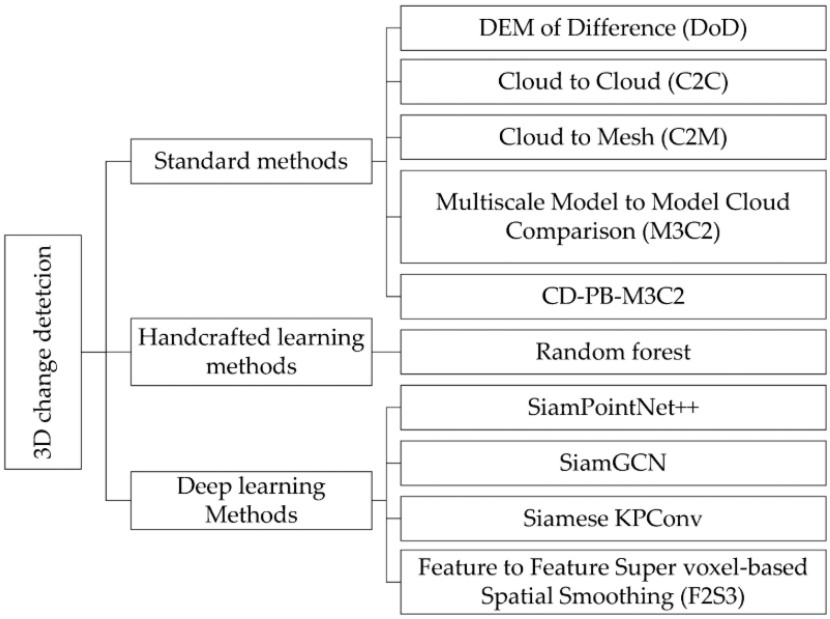


Figure 2.1: Overview on the main methods for 3D change detection, and their sub-methods. Retrieved from Kharroubi et al. (2022).

LiDAR scans made under identical conditions of any object, will not result in identical point clouds, as factors such as measurement noise ensure that the location of points will always vary slightly, even when no change has occurred. (Baltsavias, 1999). Dynamic point identification techniques should therefore be able to differentiate measurement errors, and actual change of point clouds. Standard methods try to determine whether displacement of a point is significant enough to deem it dynamic by calculating distances between points from sequential point cloud measurements, and applying a distance threshold value (Kharroubi et al., 2022). Handcrafted machine learning dynamic detection methods rely on extracting spatially variant features from sequential point clouds, which are used as inputs for a classifier. The classifier is then trained using training data to identify dynamic points, which is applied to the sequential point cloud data for classification (Tran et al., 2018). Deep learning dynamic detection methods use neural networks to learn features from point cloud data, which results in the model being able to recognize patterns without needing to manually extract and input features for training (Guo et al., 2020; Kharroubi et al., 2022). Table 2.1 provides an overview of the strengths and

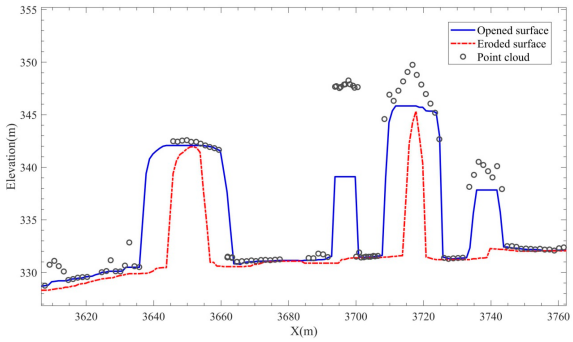
weaknesses of the discussed overarching dynamic object detection techniques.

Table 2.1: Strengths and weaknesses of dynamic object detection techniques. Based on information retrieved from Kharroubi et al. (2022).

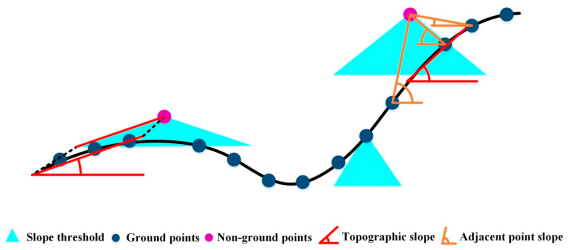
Detection methods	Strengths	Weaknesses
Standard	Simple to use and computationally efficient	Sensitive to noise, complex terrain, and uneven point cloud density
Handcrafted machine learning	Can classify changes with limited data	Quality of classification depends on training data quality
Deep Learning	High accuracy and able to learn complex patterns	Needs large labeled datasets, high computational cost, and difficult to understand

2.3. Ground and non-Ground Separation Techniques

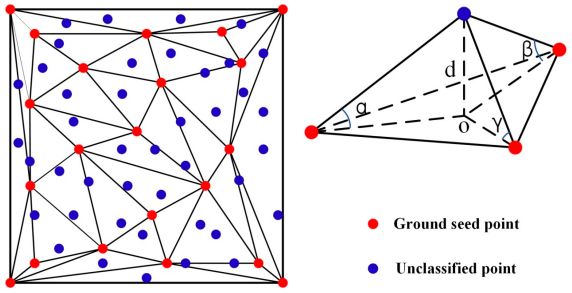
While dynamic point detection techniques are capable of identifying dynamic points, distinguishing these from points associated with dynamic objects is a challenge when solely using these techniques. This is because such techniques are tuned to detect change between sequential scans, but not specifically to differentiate between dynamic points belonging to moving objects and those resulting from geomorphological changes. To distinguish between geomorphological changes and dynamic objects, ground and non-ground points can be separated. Since geomorphological changes typically occur on ground points, while points associated with dynamic objects are usually non-ground, this separation helps isolate points associated with dynamic objects from natural surface changes. Chen et al. (2021) provides an extensive overview of different techniques which may be used for separating ground and non-ground points. A wide variety of techniques exist that can perform this separation of points, which can be classified into five different categories, namely morphology-based, interpolation-based, slope-based, segmentation-based and machine learning-based filtering algorithms (see Figure 2.2) (Chen et al., 2021).



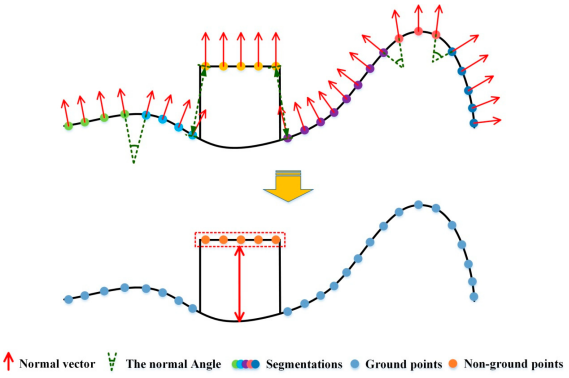
(a) Principle of a morphology-based filtering algorithm.



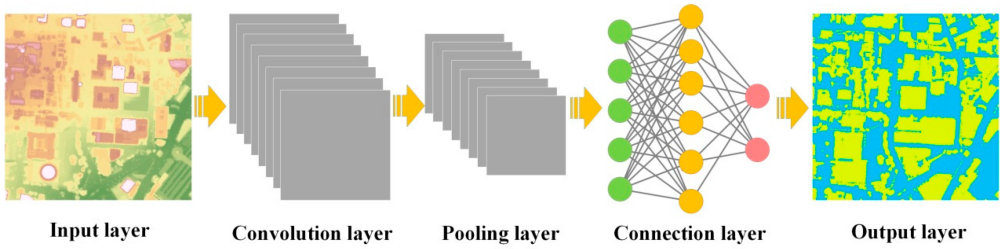
(b) Principle of a slope-based filtering algorithm.



(c) Principle of an interpolation-based filtering algorithm.



(d) Principle of a segmentation-based filtering algorithm.



(e) Principle of a machine learning-based filtering algorithm.

Figure 2.2: Overview of different ground and non-ground filtering algorithms. Retrieved from Chen et al. (2021).

Morphology-based filters use the structural shape of the point cloud to distinguish ground and non-ground points. An example is the use of dilation and erosion operations to process point clouds by expanding and contracting terrain features. Dilation enlarges features in point clouds, while erosion reduces the size. When combined, these opening and closing operations can be used to classify points as ground or non-ground based on the difference relative to the filtered surfaces (see Figure 2.2a) (N. K. Zhang et al., 2003). Slope-based filters make use of variations in the slope to determine whether a point is a ground or non-ground point. If the slope associated with a point exceeds a threshold, then it is classified as a non-ground point (see Figure 2.2b) (Vosselman, 2000). Interpolation-based filters base classification on seed points (ground points) within a moving window, and interpolating these point to construct a reference surface. Points which exceed a certain vertical distance threshold to the surface, are classified as non-ground points (see Figure 2.2c) (J. Zhang & Lin, 2013). Segmentation-based filters divide scans into segments, with each segment being classified as ground or non-ground based on terrain characteristics (see Figure 2.2d) (Yan et al., 2012). Machine learning-based filters make use of neural networks, for instance convolutional neural networks (see Figure 2.2e), to classify point clouds as ground or non-ground points. Machine learning techniques learn to recognize patterns, and based on these patterns are able to make decisions whether a point is classified as a ground or non-ground point (Rizaldy et al., 2018). Table 2.2 provides an overview of the strengths and weaknesses of the discussed techniques.

Table 2.2: Strengths and weaknesses of ground and non-ground point filtering methods. Based on information retrieved from Chen et al. (2021).

Filtering Methods	Strengths	Weaknesses
Morphology-Based	Computationally efficient	Information may be lost due to rasterization (depends on window size)
Slope-Based	Simple and computationally efficient	Slope threshold can cause issues in abrupt terrain (e.g., overhangs)
Interpolation-Based	High filtering accuracy	High computational cost
Segmentation-Based	Maintains the integrity of terrain boundaries	Effectiveness is influenced by segmentation accuracy
Machine Learning-Based	Able to learn complex patterns	Requires large labeled datasets

2.4. Clustering Techniques for Point Cloud Data

Once points associated with a dynamic object have been identified, clustering them can provide added value. This is because grouping related points into a single entity can support the processing and interpretation of the identified dynamic points. Wani (2024) provides an extensive overview of clustering algorithms which are suitable for point clouds. The available clustering techniques can be subdivided in six categories, namely centroid, density-based, graph-based, distribution-based, connectivity, and deep embedded clustering (Wani, 2024). Figure 2.3 shows five of the six overarching clustering methods, applied using a specific technique, and their performance.

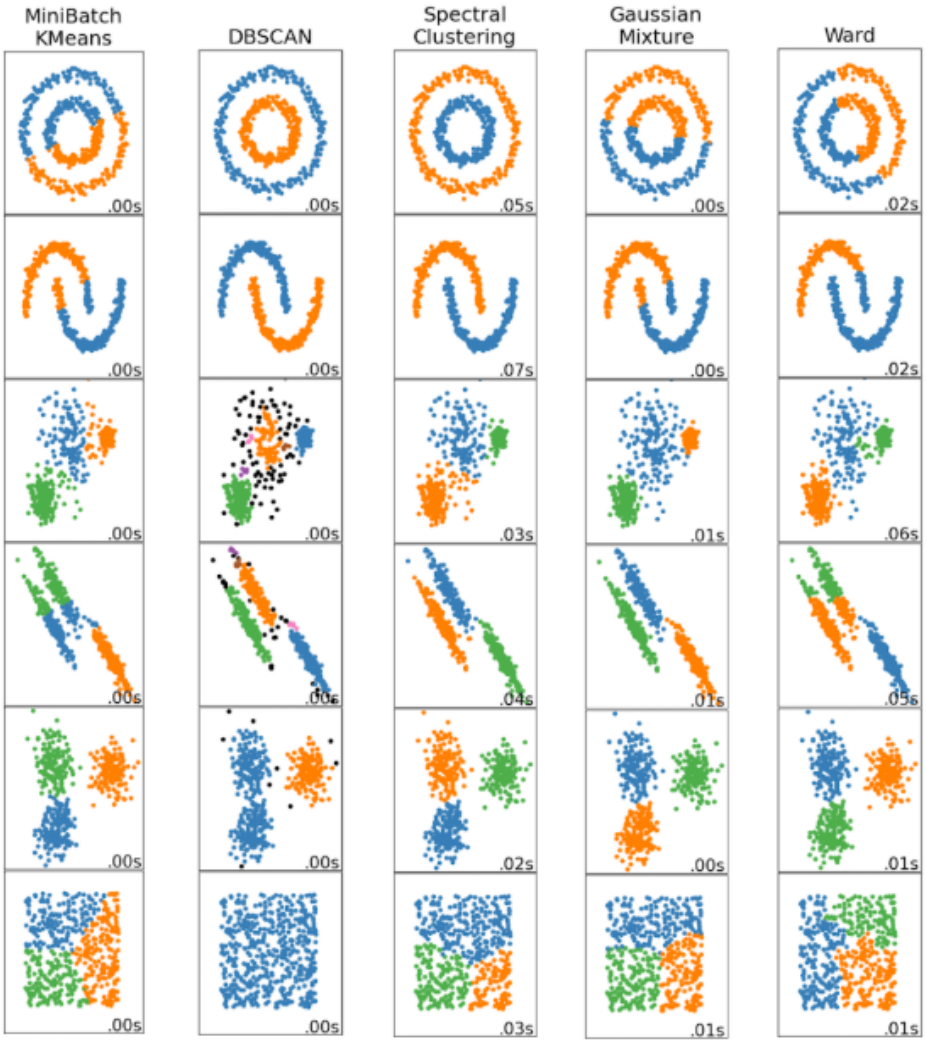


Figure 2.3: From left to right: clustering techniques, and their performance: Centroid clustering (MiniBatch KMeans), Density-based clustering (DBSCAN), Graph-based clustering (Spectral Clustering), Distribution based clustering (Gaussian Mixture), connectivity clustering (Ward). Modified from Scikit-learn (n.d.-a).

Centroid clustering techniques, such as MiniBatch KMeans, group point cloud data by assigning each point to a nearest centroid. The centroid can be defined as the mean or median of a cluster, and is placed such that the variance within a cluster is minimized (Wani, 2024). Density-based clustering techniques, such as DBSCAN, group points by identifying regions of high density separated by areas of low density (see Section 4.5) (Ester et al., 1996). Graph-based clustering, often referred to as spectral clustering, turns point cloud data into a graph where each point is a node, and the connection between nodes represent the similarity of points, for instance the euclidean distance. Points are then clustered based on their similarity compared to the rest of the data (Wani, 2024). Distribution-based clustering techniques, such as Gaussian mixture models, assume that point cloud data is generated from a mixture of distributions. These distributions are each defined by their own mean and covariance matrix, and based on probability, this can be used to assign points to clusters. (Wani, 2024). Connectivity clustering techniques, such as Ward, construct clusters by evaluating the proximity between data points. Each point is initially considered as a separate cluster, and based on distance threshold values, points within a point cloud are grouped to form larger clusters (Jain & Dubes, 1988). Deep embedded clustering makes use of neural networks to shrink point cloud data into simpler forms, and then converts the data into clusters. Deep embedded clustering keeps improving both the way it simplifies the data and the groups it creates, based on the data which is previously provided (Wani, 2024). Table 2.3 provides an overview of the strengths and weaknesses of the discussed clustering techniques.

Table 2.3: Strengths and weaknesses of clustering categories. Based on information retrieved from Wani (2024).

Clustering Methods	Strengths	Weaknesses
Centroid	Efficient, and scales well	Sensitive to noise and outliers, and assumes spherical clusters
Density-based	Handles noise well, and able to handle complex shapes	Struggles with varying densities
Graph-based	Able to cluster complex shapes	High computational cost, and sensitive to noise
Distribution-based	Able to cluster complex shapes	Assumes certain distribution, which may not always be correct
Connectivity	Easy to implement	Struggles with noise, and scalability
Deep Embedded	Able to make complex clustering decisions based on patterns	High computational costs, and needs large datasets to work well

2.5. Nearest Neighbor Calculations

Nearest neighbor calculations are an important component of point cloud processing. Efficient execution of these calculations is therefore of importance. One approach to achieve this, is the implementation of the octree subdivision principle (see Figure 2.4). An octree structures 3D point cloud data in such a way that the point cloud is enclosed by a bounding cube, which is divided into eight equal sub-cubes. Each of these sub-cubes can be divided in the same way, until a maximum 'depth' is reached, or until a cube does not contain any points. This allows nearest neighbor calculations between datasets to be performed efficiently, as instead of comparing every point with all others, the spatial region where the nearest neighbor is likely to reside is quickly identified, which reduces the number of computations, assuming that the clouds are defined in the same reference frame (González-Jorge et al., 2005). Alternatively, a binary division approach can be used, similar to the octree approach, as described by Maneewongvatana and Mount (1999). Various splitting methods can be used, such as the standard split, the midpoint split, the sliding-midpoint split, and the minimum ambiguity split (Maneewongvatana & Mount, 1999). These approaches can be applied in various ways such that the nearest neighbor is determined, with each having their own advantages, based on computational speed and accuracy (González-Jorge et al., 2005).

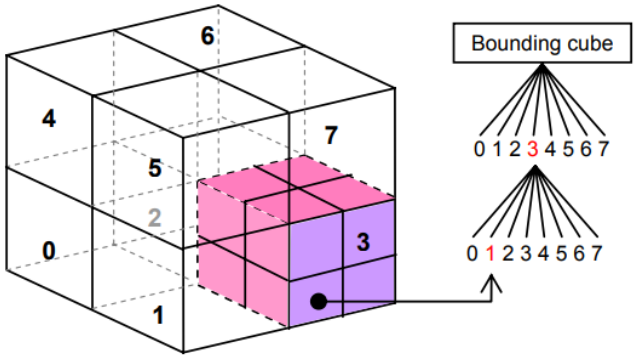


Figure 2.4: Visualization of the octree subdivision principle with bounding box and corresponding decision tree. Retrieved from González-Jorge et al. (2005).

3

Data and Study Area

This chapter introduces the dataset and the context of the study area. It begins with Section 3.1, which provides an overview of the data. Section 3.2 then examines the characteristics of the dataset. Finally, Section 3.3 outlines the small and large dynamic objects within the dataset.

3.1. Data Acquisition and Overview

the sequential LiDAR point cloud data was collected from July 2019 to June 2022 using a Riegl VZ-2000 terrestrial laser scanner, which was located on the rooftop of a hotel at the coast of Noordwijk, spanning an area approximately 1000 meters long, and 350 meters wide. Scans were made hourly, with an angular resolution of $0.03° \times 0.03°$, covering a horizontal range from $30°$ to $130°$, and a vertical range from $78°$ to $281°$. The time required to complete a scan approximately was 7.5 minutes. The correction factor n (see Section 2.1), was adapted based on weather information from KNMI weather stations (Vos et al., 2023). Figure 3.1 shows the exact location of the laser scanner. The research period was set to March 2020, during which a total of 744 scans were acquired. Among these, 11 scans were deemed unsuitable for analysis, and have therefore not been considered in this research.

Figure 3.1: (A) An aerial image showing the location of the laser scanner (red triangle), and the research area covered by the laser scanner in Noordwijk. (B) The laser scanner, positioned on the rooftop of a hotel overlooking the beach. (C) The geographic position of Noordwijk (52.24° N, 4.42° E). Retrieved from Vos et al. (2023).

A scan of the area of interest results in a 3D point cloud of approximately 4 million points depending on scan conditions, and collects the following attributes: X, Y, Z, offset time, echo type, deviation, amplitude and reflectance (see Appendix A for a description of these attributes). The dataset provides for each scan a time-dependent alignment matrix, which is applied to correct for small tilt variations in the laser scanner, and to ensure that sequential scans are aligned with each other. Additionally, pitch (forwards and backward tilt), and roll (side-to-side tilt) during a scan is provided, which may be used to assess the amount of variation in tilt of the laser scanner, and its impact on the measurements (Vos et al., 2023).

3.2. Data Characteristics

In this research, the focus is on identifying dynamic objects located on the beach. The resolution of the data varies due to the angular resolution (see Section 3.1), resulting in a non-uniform point cloud density. Figure 3.2 showcases this uneven distribution, showing that point cloud density decreases with increasing distance from the laser scanner. Effectively, objects located further from the scanner are represented by fewer data points compared to those closer to it. Since this may affect the results when determining whether an object is dynamic, the area of the beach which is considered is limited to a region where the point cloud density is relatively uniform (see Figure 3.2).

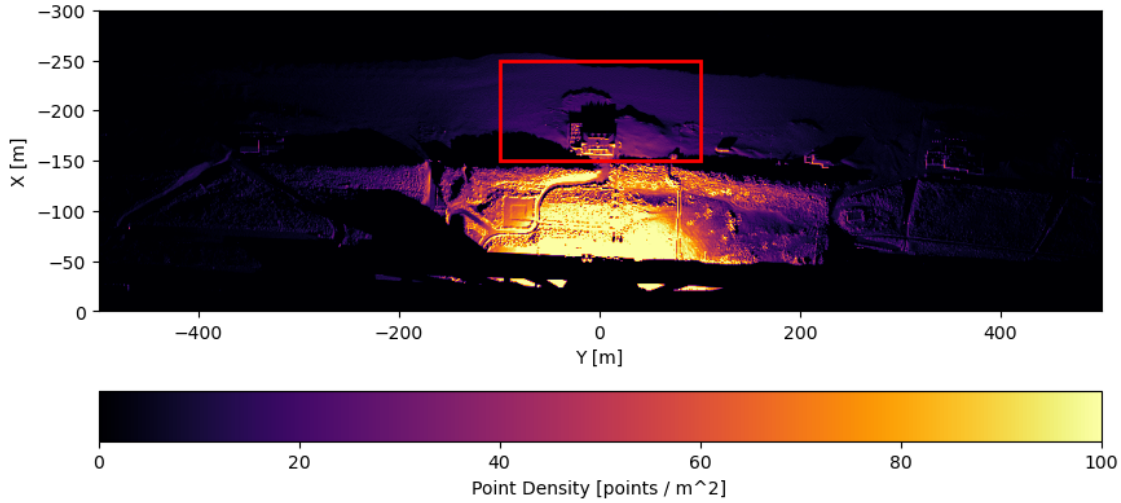


Figure 3.2: Point cloud density of the dataset. Red rectangle indicates the region which is considered in this research. Data from CoastScan scan on March 18th 2020, 00:00.

Since the scanner is positioned on a relatively tall building in the area, it provides a good overview of the region. However, large objects, such as a beach pavilion, obstruct the laser scanner's line of sight, preventing it from capturing points in certain areas. This results in regions where no points are registered, known as occlusions.

Because scans are taken at hourly intervals, it is unlikely that the same dynamic object will appear in sequential scans if it has moved on. As a result, dynamic objects cannot be tracked over time, but their presence can be detected on a per-epoch basis. If a dynamic object moves while the scan is being performed, this motion can distort the point cloud representation, which may lead to warping of its appearance or causing it to appear fragmented.

3.3. Dynamic Objects within the Dataset

In this research, an object is deemed dynamic if it is a non-ground object which shows dynamic behavior relative to a previous epoch. This means that a dynamic object which appears in one epoch, remains static for one or more subsequent epochs, and then disappears, is only classified as dynamic once, namely, at the moment it first appears. Thus, whether an object is deemed dynamic is determined on an epoch-to-epoch basis, rather than by its presence within any single epoch.

A distinction is made in this research when identifying dynamic objects based on their size in the point cloud. Namely, objects consisting of 100 points or more are classified as large objects, while, those with fewer than 100 points are considered small dynamic objects. This distinction allows the performance of the algorithm to be evaluated based on the point cloud size of a dynamic object. Although the selected research area has a relatively uniform point cloud density, the angular resolution of the laser scanner results in fewer points being recorded for identical objects as their distance from the scanner increases. Because of this, an object's classification as large or small is influenced by its position relative to the scanner. Therefore, the number of points representing an object in the point cloud is not characteristic of the object, but rather a basis which can be used to evaluate the results.

Various types of dynamic objects are observed within the research area, ranging from dogs and people, to vehicles and equipment such as cars, excavators, and bulldozers, as well as miscellaneous items like doors, ladders, crates, and barrels. In contrast to large dynamic objects, people and dogs usually have fewer than 100 points, and are thus small dynamic objects. People can typically be recognized by their stick figure appearance, and have an abstract nature due to the limited data points. Figure 3.3 highlights some of the small dynamic objects that are found within the dataset.

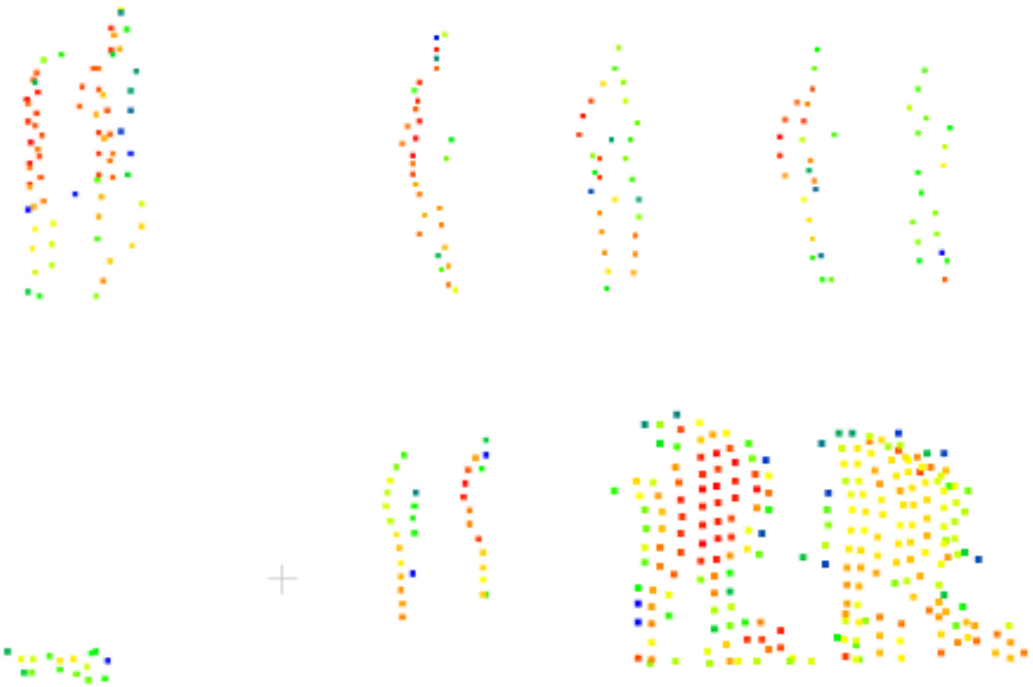


Figure 3.3: Small dynamic objects on the beach of Noordwijk, shown from left to right and top to bottom: two people walking closely together, four people walking spread out, two people with a dog, two potential cyclists. Colors represent point reflectivity. Retrieved from CoastScan scans during March 2020.

Figure 3.4 highlights some of the large dynamic objects which can be observed within the research area. Typically, vehicles and equipment consists of more than 100 points. Some miscellaneous objects (such as ladders, and large crates) may also exceed this number. Appendix B details the size ranges of both small and large dynamic objects, along with the originating scan IDs from which they were extracted.

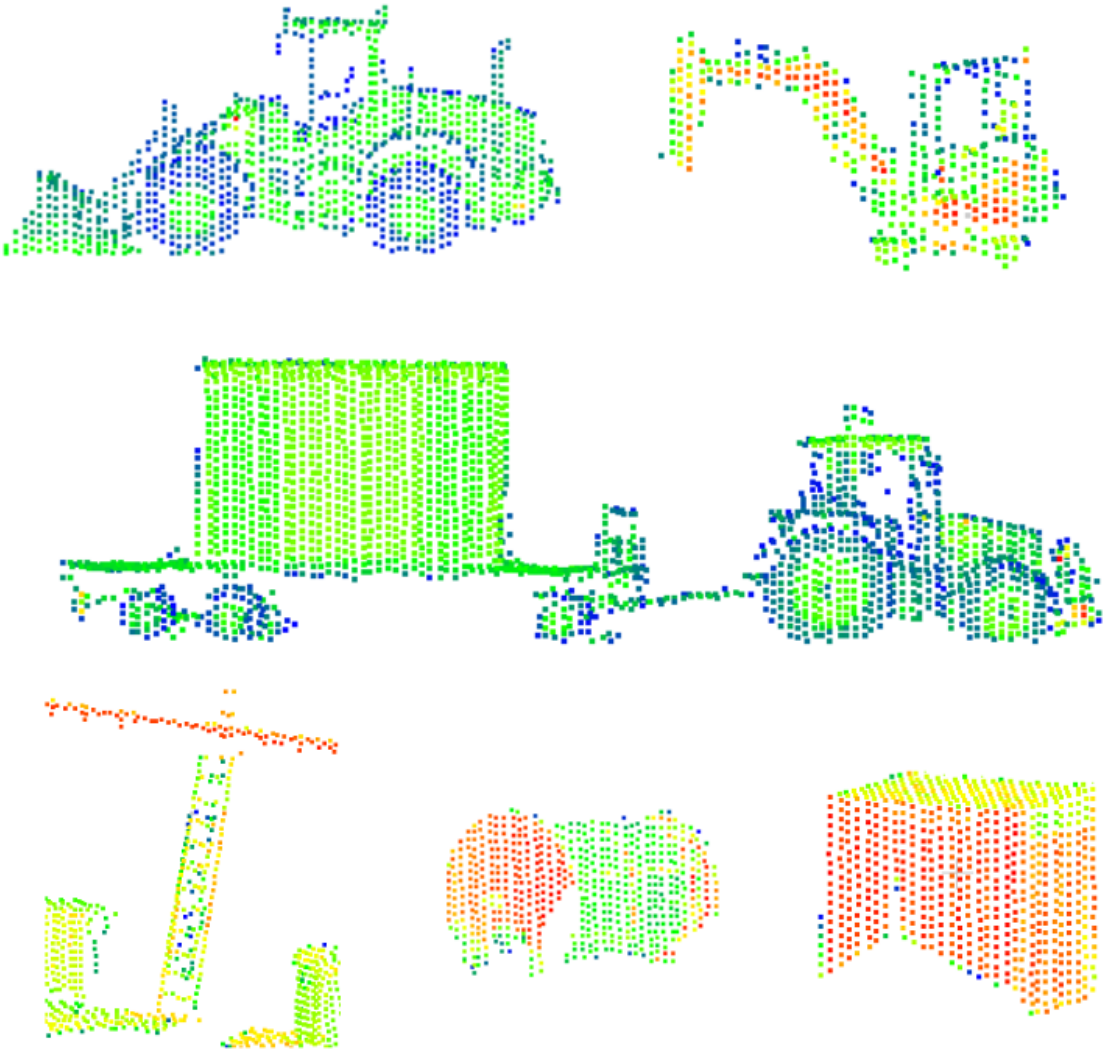


Figure 3.4: Large dynamic objects on the beach of Noordwijk, shown from left to right and top to bottom: bulldozer, excavator, tractor with trailer with cargo, ladder, large barrel and a large crate. Colors represent point reflectivity. Retrieved from CoastScan scans during March 2020.

4

Methodology

This chapter introduces the methodology used to automate the identification of dynamic objects in terrestrial LiDAR point cloud data from the beach of Noordwijk. First, the workflow and the reasoning behind the chosen approach are discussed in Section 4.1. This is followed by a description of the preprocessing steps applied to the dataset in Section 4.2. Then, an overview of the techniques is provided, including the Cloth Simulation Filter, which is used to separate ground from non-ground points, discussed in Section 4.3. Next, the Cloud-to-Cloud comparison method is explained in Section 4.4, which enables the detection of dynamic points by evaluating positional differences between sequential scans. Following this, the DBSCAN algorithm is presented in Section 4.5, which clusters dynamic points into individual objects. Finally, in Section 4.6, a description of the parameter tuning process, the performance evaluation and sensitivity analysis is provided.

4.1. Justification of Methods and Workflow Overview

This section gives an overview on the steps taken to go from a LiDAR point cloud dataset to a dataset which has clustered dynamic objects, and provides a justification of the methods which have been selected to perform this process.

As highlighted in Chapter 2, several methods exist to go from a LiDAR point cloud dataset into one where dynamic objects have been identified. Although multiple approaches are available, a key distinction can be made in terms of dynamic point detection techniques, which form the foundation for detecting dynamic objects. Handcrafted learning methods and deep learning methods, both require training data that must be manually identified and labeled. However, such labeled datasets are not available for this research, and creating one would require extensive manual effort. For this reason, a standard method has been selected to perform the dynamic point detection step, namely Cloud-to-Cloud comparison. This technique is chosen for its simplicity and computational efficiency, which makes it suitable for effective implementation on the dataset.

In contrast to machine learning methods, Cloud-to-Cloud comparison does not directly identify dynamic objects, as it cannot distinguish between geomorphological changes and points associated with dynamic objects. For this reason, ground and non-ground point separation is carried out before the Cloud-to-Cloud comparison step. The method selected for this task is a morphology-based filter, namely the Cloth Simulation Filter. This filter is chosen due to its relatively high computational efficiency, ease of implementation via the Python package CloudCompy, and its relative novelty, which also allows its performance to be evaluated in comparison to the available literature on the method.

Following the detection of dynamic points, a clustering step is applied. This is done to enable the analysis of dynamic objects as separate entities. The clustering technique selected is a density-based clustering algorithm, namely DBSCAN. This method is chosen because of its robustness to noise and its ability to identify clusters of arbitrary shape. Additionally, DBSCAN is relatively efficient with regards to computational performance, and does not require a predefined number of clusters.

Based on the selected methods, the workflow for dynamic object identification will be as follows. First, a preprocessing step is conducted, after which the Cloth Simulation Filter is applied to separate ground and non-ground points. Subsequently, the non-ground points from one scan are compared to the non-ground points from the previous scan using a Cloud-to-Cloud comparison. Finally, the dynamic points identified through this process are clustered using the DBSCAN algorithm. This workflow is visualized in the form of a flowchart, which is shown in Figure 4.1.

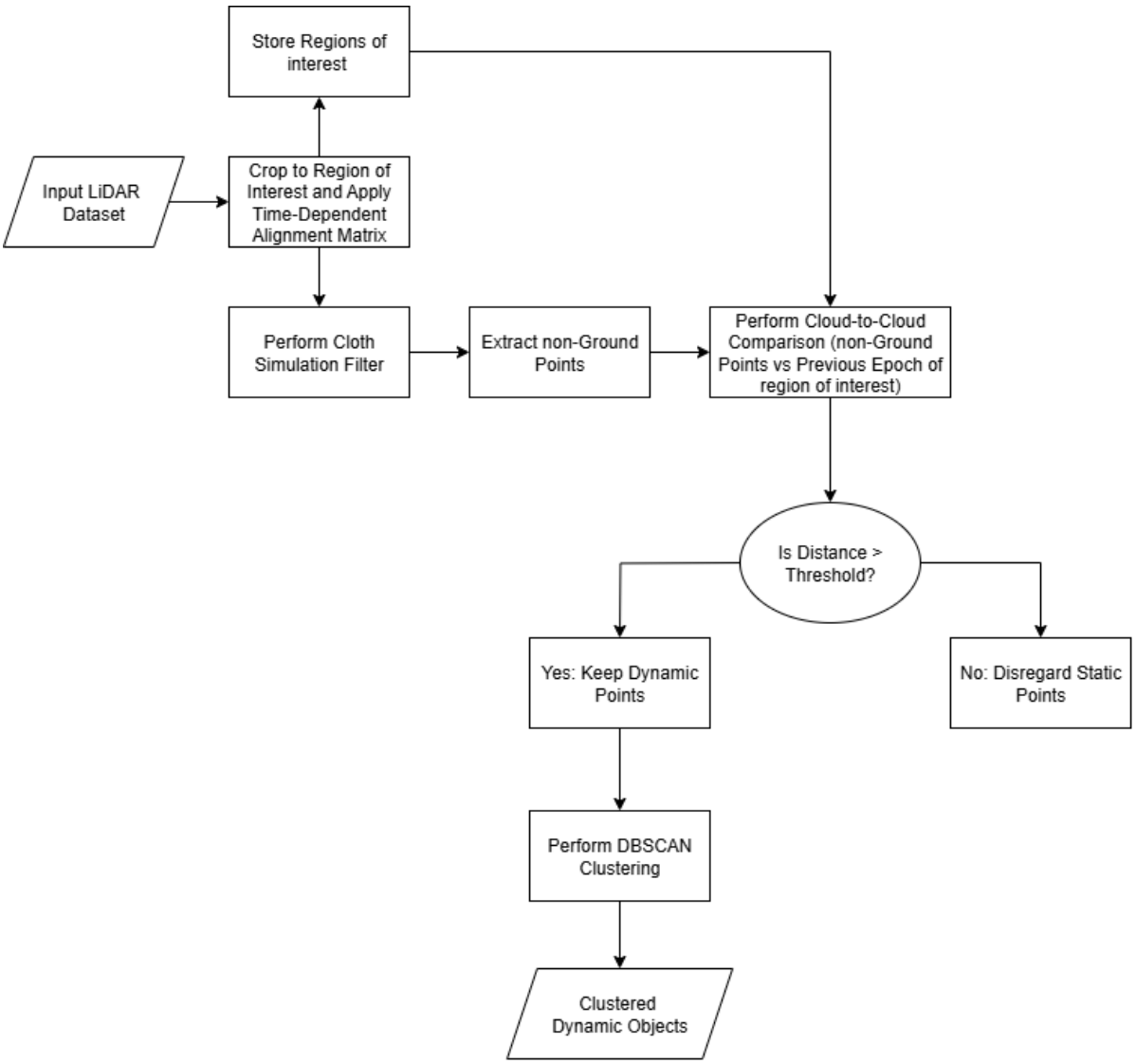


Figure 4.1: Workflow LiDAR dataset to Clustered Dynamic Objects.

4.2. Preprocessing

Preprocessing is performed to prepare three sets of data: a single day (18th March 2020), one week (18–24th March 2020), and one month (March 2020), and is necessary to facilitate the performance of the algorithm set-out in Section 4.1. This evolves applying the time-dependent alignment matrix and cropping the dataset to the selected research area (see Figure 3.2). This is done such that the research area is limited to a region with a relatively uniform point cloud density (see Section 3.2). Additionally, this results in computation time being reduced, while also leaving out data which may negatively effect the parameter tuning process. Figure 4.2 provides a top view of the region of the beach which is selected to perform the workflow on. The beach is a dynamic environment due to natural (tides, waves, etc.,) and anthropogenic activity (bulldozers, beach beds, etc.,), and therefore each scan will be unique.

To evaluate which parameter configuration performs best in identifying dynamic objects (See Section 4.6), two scenarios are explored. Scenario A (see Figure 4.2), includes the beach pavilion, while Scenario B excludes a large portion of it (see Figure 4.3). This is done because data points are occasionally registered inside the beach pavilion, possibly due to doors of windows opening, which results in the LiDAR scanner detecting points inside it. Since it is challenging to determine whether these points represent dynamic objects, both scenarios are considered to assess their impact on the parameter tuning process.

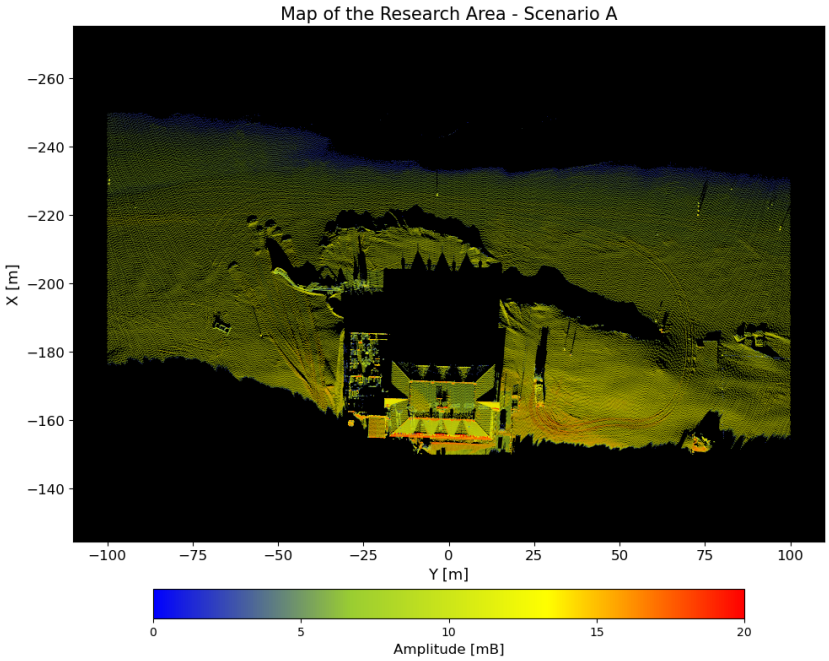


Figure 4.2: Scenario A: Top view of the research area at the beach of Noordwijk. Data from CoastScan scan on March 18th 2020, 00:00.

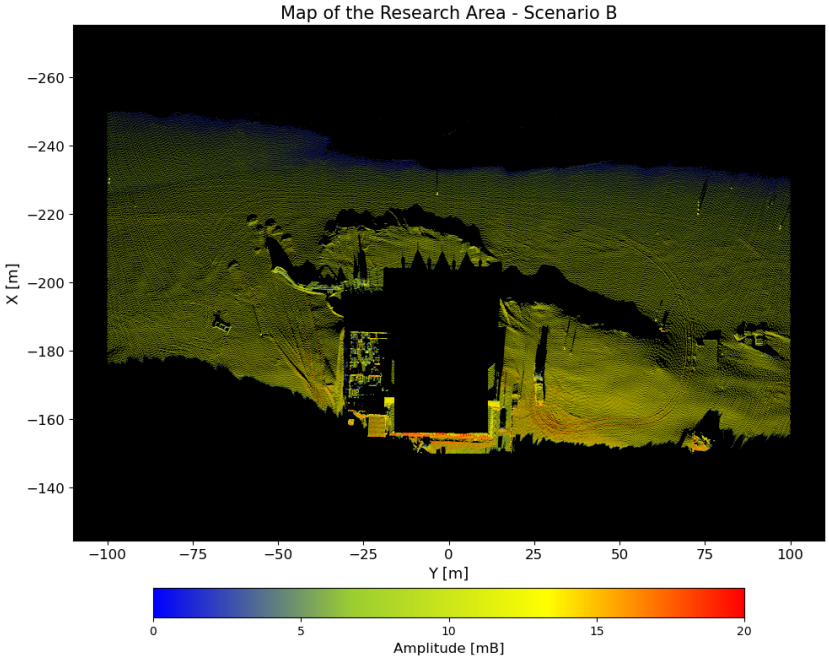


Figure 4.3: Scenario B: Top view of the research area at the beach of Noordwijk excluding a large part of the beach pavilion. Data from CoastScan scan on March 18th 2020, 00:00.

4.3. Separating Ground and non-Ground Points: CSF

This section discusses the theoretical principles underlying the Cloth Simulation Filter (CSF), followed by a discussion as to how the Cloth Simulation Filter is implemented at the beach in Noordwijk. This includes discussing which parameters were selected, and providing justification.

4.3.1. Concept of Cloth Simulation Filter

CSF is based on the principle that when a cloth is laid over a surface, it follows the shape of the surface. A surface typically has non-ground objects, such as vegetation or buildings. Because of this, the cloth rests on these objects, preventing it from reflecting the bare-earth topography, which is the goal of a Digital Terrain Model (DTM). To produce a more accurate DTM, the cloth simulation is applied to an inverted version of the terrain. This allows non-ground objects to be ignored for a large part, which limits the impact on the DTM. The DTM which is created can serve as a boundary to separate ground and non-ground points in LiDAR point cloud data (W. Zhang et al., 2016). Figure 4.4 shows the concept of a Cloth Simulation Filter.

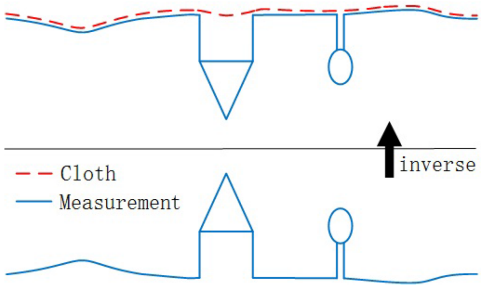


Figure 4.4: Visual representation of the concept of a Cloth Simulation Filter. Retrieved from W. Zhang et al. (2016).

The behavior of the cloth is modeled using a Mass-Spring Model, in which nodes of the grid of the simulated cloth are interconnected by springs, allowing elastic behavior of a cloth to be modeled (Provot, 1995; W. Zhang et al., 2016). Once the cloth is laid over the inverted LiDAR point cloud, its shape can be simulated over time. The position of the nodes of the grid behave in accordance with Newton's second law, which states that the sum of the external (gravitational and collision) forces, and internal (spring) forces that work on a grid node, is equal to the total amount of force of the grid node (W. Zhang et al., 2016).

Several constraints are made when cloth simulation is implemented as a filter for LiDAR point clouds. Movement is constrained to be in the vertical direction, such that this allows the detection of a grid node falling below the terrain by comparing elevation values (W. Zhang et al., 2016). Furthermore, a particle is set to be unmovable once it reaches the ground. Finally, the external and internal forces are processed separately, which is done to simplify the process and achieve high performance (W. Zhang et al., 2016). The main steps in CSF can be observed in Figure 4.5.

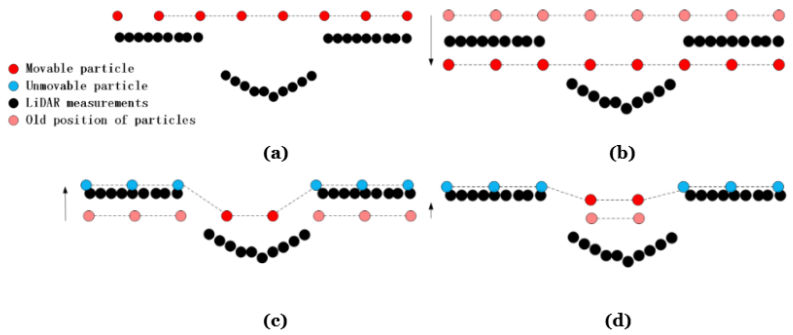


Figure 4.5: Overview of the CSF algorithm: (a) a cloth is positioned above the inverted LiDAR surface; (b) gravity is applied to simulate particle movement; (c) particles that intersect the ground are relocated to the surface and fixed in place; (d) internal cloth forces shift the remaining movable particles. Adapted from W. Zhang et al. (2016).

Table 4.1 summarizes the main parameters involved in CSF and highlights their roles and impacts on the filtering process. This includes whether post-processing occurs, and which threshold value h_{cp} is taken. Other parameters are grid resolution, time step, maximum number of iterations, Maximum height variability, the rigidness of the cloth, and the distance threshold h_{cc}.

Table 4.1: Main parameters of a Cloth Simulation Filter, with a brief description and their impact on the CSF algorithm. Based on information retrieved from W. Zhang et al. (2016)

Parameter	Description	Impact on CSF Algorithm
Grid resolution	Distance between grid nodes in the simulated cloth	Affects the cloth's sensitivity to terrain details. Finer grids capture small variations better but increase computation time.
Time step (dT)	Time interval between simulation steps	Controls the smoothness and stability of the cloth movement. Fixed, as it has been optimized for performance and accuracy.
Maximum number of iterations	Maximum number of times dT is applied	Higher values allow the cloth to settle more accurately on complex terrain if more iterations are needed.
Maximum height variability	Minimum allowed vertical change in the cloth during simulation	Provides a criterion to stop the simulation. Fixed value.
Rigidness (RI)	Controls cloth stiffness by altering displacement (D) of fixed nodes	Impacts how the cloth conforms to terrain. RI=3 (flat): D = ⅞ × VD; RI=2 (sloped): D = ¾ × VD; RI=1 (steep): D = VD.
Post-processing factor	Indicates whether post-processing is applied	Refines the results. Should be applied when RI equals 1 or 2.
Slope fit threshold (h_{cp})	Threshold used in post-processing to determine ground contact	Controls cloth adjustment during refinement. Fixed at 0.3 meters for all datasets.
Distance threshold (h_{cc})	Max distance from a LiDAR point to cloth to be classified as ground	Influences point classification. A smaller threshold results in stricter ground point detection.

Steep slopes might need post-processing to avoid large errors, as a simulated cloth will not fit the points along a steep slope well (see Figure 4.6). Post processing may help smooth the margins of the steep slopes by finding unmovable grid nodes surrounding movable nodes, and comparing the height values with the corresponding point. If the height difference is within a certain threshold h_{cp}, the movable node point is moved to the ground and set as unmovable (W. Zhang et al., 2016).

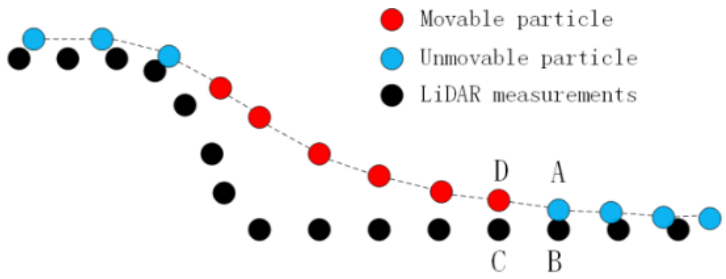


Figure 4.6: The error of a simulated cloth in comparison to LiDAR measurements along steep slopes. Retrieved from W. Zhang et al. (2016).

4.3.2. Implementation of Cloth Simulation Filter

Certain parameters of CSF, as discussed in Section 4.3.1, are prefixed because they are optimized for the algorithm (time step, slope fit threshold, and maximum height variability), and cannot be altered within software (e.g., CloudCompare or the Python package CloudComPy). The other parameters are configurable. W. Zhang et al. (2016) highlights that the algorithm's performance is case-dependent and can vary with different parameter settings. For the grid resolution, a resolution of 2.0 is appropriate for the dataset. Typically, 500 iterations are selected as the maximum number of iterations, but generally the maximum height variation threshold which is built into the program stops the algorithm before it comes into effect. W. Zhang et al. (2016) suggests using a rigidness of 1 for steep terrain, rigidness of 2 for sloped terrain, and rigidness of 3 for flat terrain. The beach can best be described as a sloped terrain, and therefore a rigidness of 2 is taken. As a result, the post-processing factor is enabled. Additionally, the distance threshold is selected to be 0.5, as this largely preserves the integrity of non-ground objects, while successfully separating ground and non-ground points. To perform CSF, the python function CSF of the python package CloudComPy is used, with the configurable parameters and their selected values as described in Table 4.2.

Table 4.2: Selected values for configurable CSF parameters.

Parameter	Selected Value
Grid resolution	2.0
Maximum number of iterations	500
Rigidness (RI)	2
Post-processing factor	Enabled
Distance threshold (h_{cc})	0.5

4.4. Detecting Dynamic Points: C2C

Cloud-to-Cloud comparison (C2C) uses the principle of comparing 3D Euclidean distances of individual points in a point cloud dataset with its nearest neighbor in a sequential dataset (see Section 2.5). A threshold value is applied to nearest neighbors in sequential point cloud data and their associated distances to determine if a point should be classified as dynamic, making the threshold the decisive factor in the classification (Maneewongvatana & Mount, 1999).

C2C is implemented after the separation of ground and non-ground points. This is done primarily to detect dynamic objects while excluding geomorphological changes and occlusions occurring at the surface. Additionally, this separation enhances the performance of C2C and improves the accuracy of dynamic object identification. A binary approach is implemented to determine the nearest neighbor, using the standard split method, meaning that the split takes place at the median (Maneewongvatana & Mount, 1999). This method is selected because it is straightforward to implement using the kDTree Python function from scipy.spatial (Virtanen et al., 2020).

C2C can be performed by comparing either non-ground points to other non-ground points of two consecutive epochs, or by comparing non-ground points to a cropped scan before CSF is performed. Both approaches have their advantages, but comparing non-ground points to a cropped scan from another epoch is chosen as the preferred method, as this reduces noise. The effectiveness of this method, however, depends on the chosen threshold value: if set too high, dynamic objects located close to the ground may be misclassified as non-dynamic because of their proximity to the surface of the consecutive epoch. The exact threshold value applied to the dataset is not predetermined, but is obtained through parameter tuning (see Section 4.6).

4.5. Clustering Dynamic Points: DBSCAN

This section discusses the theoretical principles underlying Density-Based Spatial Clustering of Applications with Noise (DBSCAN), followed by a description as to how DBSCAN is implemented after C2C.

4.5.1. Concept of DBSCAN

DBSCAN makes use of the principle that dynamic objects correspond to dense regions within the point cloud, which are separated from other dynamic objects by areas of lower point density. Whether a point belongs to a dense region depends on whether it is a core point or lies within the neighborhood of one. A core point is defined as a a point which has at least a specified number of neighboring points within a given radius. All points within this radius are considered part of the same cluster. If a point does not meet the core point criteria or is not within the neighborhood of any core point, it is classified as noise (Ester et al., 1996). A flowchart describing the main steps performed in the DBSCAN algorithm is shown in Figure 4.7.

There are two key parameters in the DBSCAN algorithm, one of which being the neighborhood radius threshold value ϵ. This parameter defines the spatial extent within which points are considered neighbors, and determines whether a point is density-reachable. In other words, whether the point lies within the ϵ-neighborhood of a core point. ϵ also influences whether a point is classified as noise or assigned to a cluster. The other key parameter is the minimum number of points, *MinPts*, which defines the minimum number of neighboring points required for a point to be classified as a core point, which sets the density threshold of the algorithm. This determines which points are considered part of a dense region and which are initially classified as noise (Ester et al., 1996).

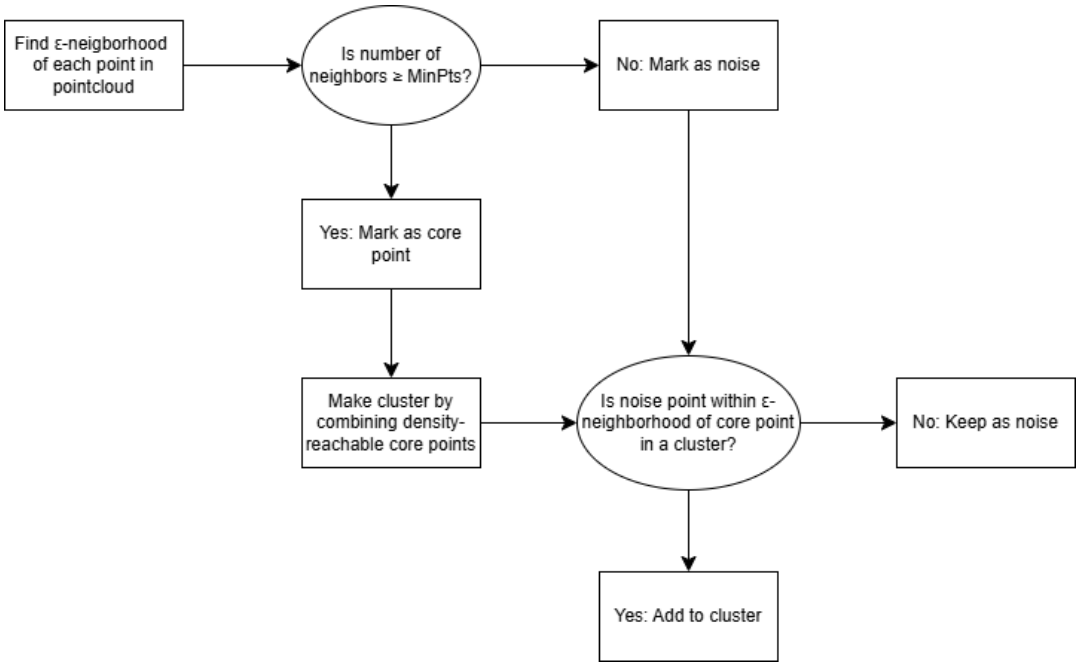


Figure 4.7: Flowchart DBSCAN Algorithm. Based on information retrieved from Ester et al. (1996).

4.5.2. Implementation of DBSCAN

DBSCAN is applied after C2C and CSF. Because of this approach, DBSCAN processes fewer points, making it more efficient. If the C2C comparison and CSF filtering are performed accurately, the amount of noise in the detected dynamic points is expected to be limited. As a result, the points grouped through clustering are more likely to correspond to dynamic objects. To implement DBSCAN, the python function DBSCAN is used from the package sklearn.cluster is used (Scikit-learn, n.d.-b). The function requires limited inputs, with most important parameters being ϵ and *MinPts*. These points are not predetermined, and are selected during parameter tuning (see Section 4.6). The function relies on nearest neighbor calculations, which is done based on 3D euclidean distances.

4.6. Result Optimization and Evaluation

This section describes the parameter tuning process, what the boundary conditions are for the parameter tuning process, and which scenarios are considered. Additionally, the performance evaluation and the sensitivity analysis are discussed.

4.6.1. Parameter Tuning

Parameter tuning is performed to determine which configuration gives the best result. This is done by selecting a reference dataset, and by visually identifying the number of dynamic objects that can be found in the dataset. The data corresponding to the 18th of March 2020 is selected as the reference dataset. This dataset of this day consists of 24 sequential LiDAR scans of the Beach of Noordwijk, without any noteworthy anomalies. The number of dynamic objects is established through visual identification and further refined by evaluating the workflow results using an arbitrary configuration of C2C and DBSCAN parameters, as the optimal configuration for these parameters has not yet been determined. Whether an object is set to be dynamic, is based on whether an object has moved with respect to a consecutive epoch (see Section 3.3). The visually determined number of dynamic objects over the course of the 18th of March is shown in Figure 4.8.

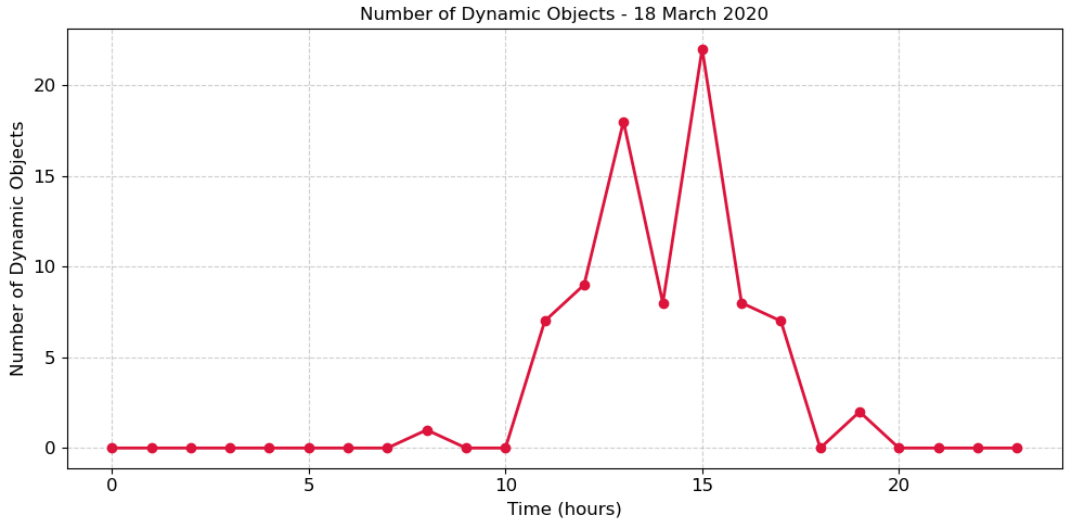


Figure 4.8: Visually determined number of dynamic objects over the course of the 18th of March 2020.

As mentioned in Section 4.2, two scenarios are considered; one scenario includes the beach pavilion when performing the clustering process (Scenario A), the other excludes it (Scenario B). The parameter tuning focuses on the C2C and DBSCAN clustering algorithm. CSF is excluded from the tuning process for two reasons, namely, existing literature provides a clear directive for the configuration of parameters, particularly regarding parameters such as the post-processing factor and rigidness. Additionally, CSF is computationally intensive, and excluding it from the parameter tuning significantly reduces overall processing time. The values used for CSF are highlighted in Table 4.2. The considered ranges and step sizes for C2C and DBSCAN parameters are shown in Table 4.3. The ranges are selected to be

relatively broad, such that a wide range of parameter configurations are considered. *MinPts* is kept relatively small, since many of the dynamic objects consists of a small number of points. Step sizes are selected to determine which configuration works best. Each parameter combination within the ranges shown in Table 4.3 are considered during the parameter tuning process.

Table 4.3: Ranges and step sizes of parameters considered for C2C and DBSCAN.

Parameter	Range	Step size
Distance Threshold (C2C)	[0.1, 1]	0.1
ε (DBSCAN)	[0.1, 1]	0.1
MinPts (DBSCAN)	[5, 15]	1

Outcomes for both scenarios are compared to the reference dataset shown in Figure 4.8. Optimized parameters for C2C and DBSCAN are obtained by identifying the configuration that yields the lowest cumulative deviation from the reference dataset, based on the number of detected clusters.

4.6.2. Performance Evaluation

The goal of the performance evaluation is to assess the effectiveness of the parameter tuning process, and resulting parameter configuration. This is achieved by applying the best-fitting parameters of the best performing scenario to a seven-day evaluation period from March 18 to March 24, 2020, comprising a total of 168 scans. A scenario is considered to have the best performance if it achieves the lowest cumulative deviation with respect to the seven-day reference period.

During the evaluation, a distinction is made between small and large dynamic objects (see Section 3.3), which allows examining whether the algorithm performs more effectively on small dynamic objects (<100 points) or large ones (≥100 points). The number of dynamic objects is established through visual identification, and supplemented by evaluating the results of the workflow using the tuned parameters of the best performing scenario. The visually determined number of small and large dynamic objects throughout the evaluation period are shown in Figure 4.9.

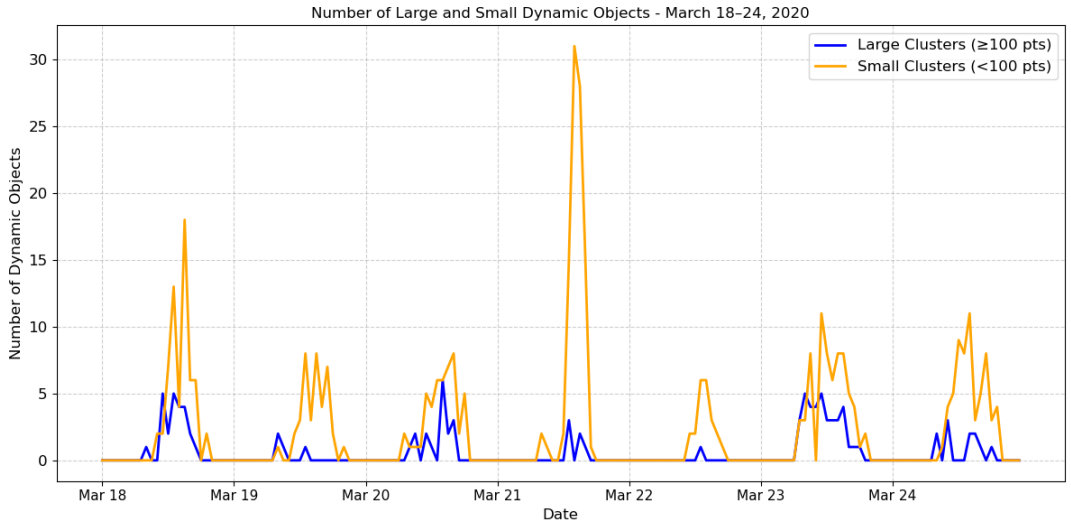


Figure 4.9: Visually determined number of small and large dynamic objects over the course of March 18–24, 2020.

The quality of the detected dynamic objects is evaluated by comparing the true number of points per object to the number identified by the algorithm, using a reference set shown in Figures 3.3 and 3.4. The error is quantified by determining the difference between the true and detected number of points per object. Additionally, an assessment is made whether each dynamic object of the reference set has correctly been clustered as a distinct entity.

4.6.3. Sensitivity Analysis

The sensitivity of the results is assessed based on the parameter configuration of the best performing scenario. This is done by performing the workflow on a seven-day period from March 18 to March 24, 2020, and varying the parameters of C2C, and DBSCAN. CSF is excluded based on the reasons provided in Section 4.6. The variation is obtained by performing a Monte Carlo Simulation. A Monte Carlo Simulation involves random sampling within parameter ranges to assess the sensitivity of the results to parameter fluctuations (Saltelli et al., 2008).

Parameter sampling for the Monte Carlo Simulation is performed using a normal distribution, which requires a mean and standard deviation. The mean values correspond to the optimized parameters from the tuning process for C2C and DBSCAN. The standard deviation for each parameter is set to 10% of the corresponding mean value, which keeps the overall variation of the parameters limited. This is so, because following the parameter tuning process, there should be certainty regarding the parameters. A total of 100 simulation runs is performed to assess the sensitivity of the detected number of small and large dynamic objects to variations around the optimized C2C and DBSCAN parameters. The Monte Carlo simulation complements the parameter tuning by quantifying the robustness of the workflow to small variations in the tuned parameters.

5 Results

This chapter presents the results of the workflow described in Chapter 4. First, the results of the parameter tuning process are presented in Section 5.1, including the results of the two considered scenarios and the performance of the optimal configuration when applied to a seven-day evaluation period. The detection performance is presented for small and large dynamic objects, considering both the error in the number of detected dynamic objects and the detection quality. This is followed by the results of the sensitivity analysis in Section 5.2. Finally, in Section 5.3, the results of applying the workflow to a full month of LiDAR point cloud data is presented.

5.1. Parameter Tuning and Performance Evaluation

Following the parameter tuning process (see Section 4.6), two parameter configurations are obtained based on the considered scenarios. The results of the parameter tuning process corresponding to the scenarios are presented in Table 5.1.

Table 5.1: Results of the parameter tuning process for the considered scenarios.

Scenario	Distance Threshold	ϵ	MinPts	Error (%)
Scenario A	0.8	0.7	11	17.1
Scenario B	0.6	0.7	9	12.9

Following the criteria set for best performing configuration (see Section 4.6), Scenario B performs best, which excludes the beach pavilion. This is so, because it has the lowest cumulative deviation with respect to the reference dataset. Therefore, the corresponding parameters of Scenario B are used to perform the performance evaluation (see Section 4.6.2). Table 5.2 presents the results of the performance evaluation.

Table 5.2: True number of small and large dynamic objects, along with absolute and relative detection errors, during the seven-day evaluation period (March 18–24, 2020)

Dynamic Object Type	True Number	Absolute Error	Error (%)
Small Dynamic Objects	392	90	23.0
Large Dynamic Objects	101	7	6.9

Figure 5.1 presents a graphical overview of the detection performance for small and large dynamic objects, including their respective errors relative to the reference dataset over the evaluation period.

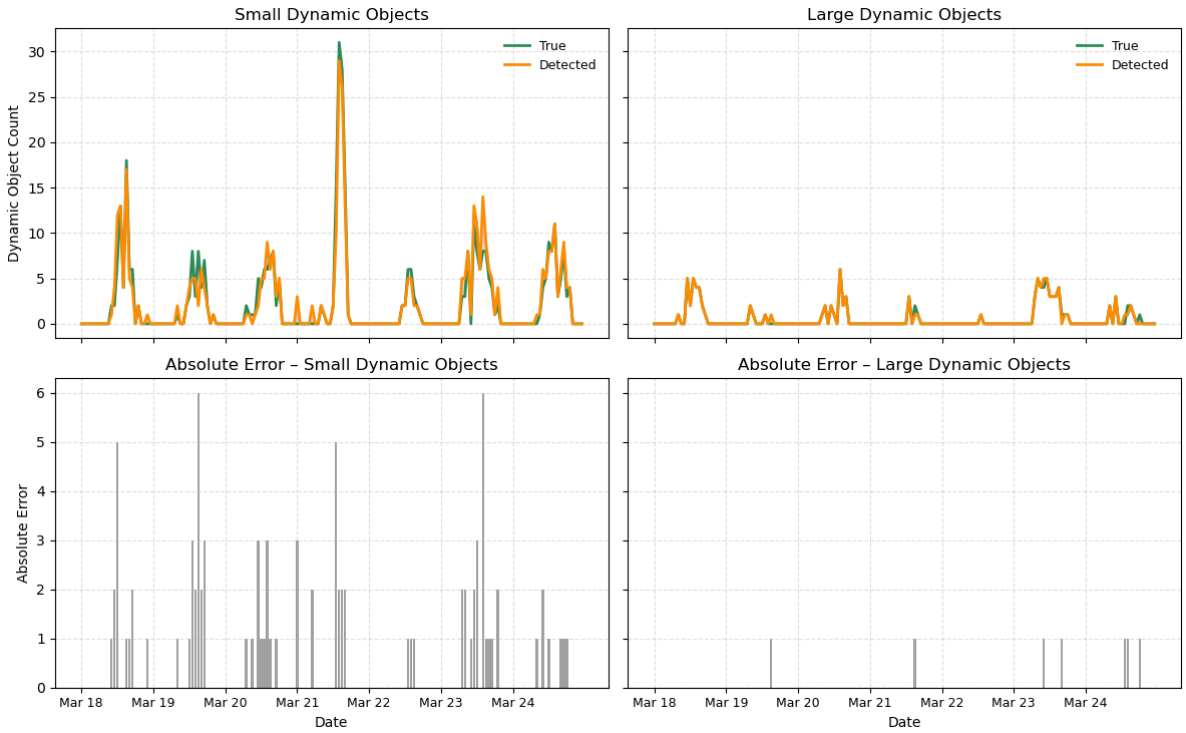


Figure 5.1: Detection performance and absolute error for small and large dynamic objects over the period March 18–24, 2020.

The correctly and incorrectly classified points for small dynamic objects, in comparison to the reference set for small dynamic objects (see Figure 3.3), are shown in Figure 5.2. Correctly classified points are highlighted in green, while misclassified points are shown in red.

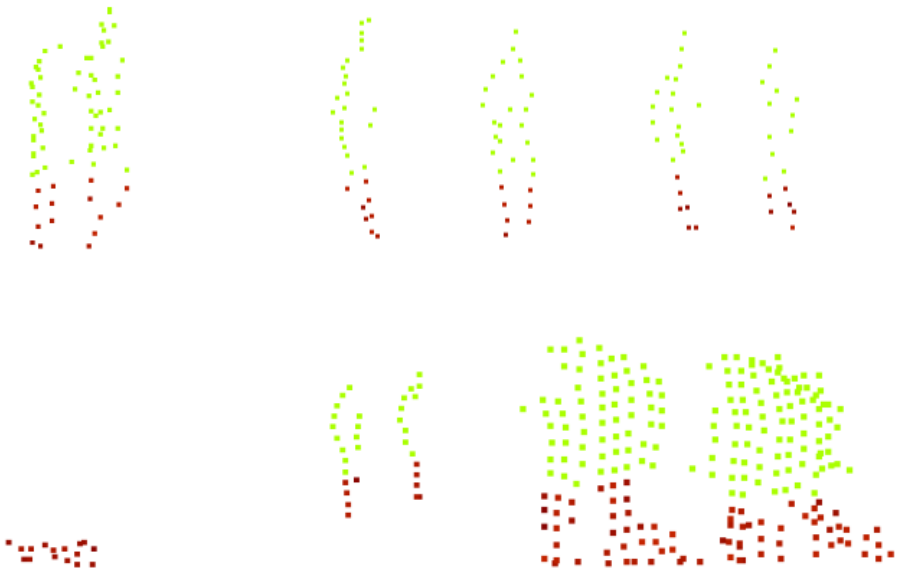


Figure 5.2: Classification of points belonging to small dynamic objects in the reference set. Green indicates correctly classified points, and red indicates missed points. From left to right and top to bottom: two people walking closely together, four people walking spread out, two humans with a dog, and two potential cyclists.

Table 5.3 presents, for each group of small dynamic objects in the reference set, the number of correctly identified points, the number of missed points, the corresponding error percentage, and the number of dynamic objects which are correctly classified (i.e., each visually distinct object was matched to one detected instance). On average, the reference set for large dynamic objects has a 33.7% error in the number of incorrectly detected points. One of the four small dynamic object groups was correctly classified. In the case of the two people walking in close proximity, they were incorrectly grouped into a single cluster, resulting in a misclassification. For the four individuals walking equidistantly, only three dynamic objects were identified instead of four, as two people were mistakenly grouped together. Regarding the humans walking with a dog, the dog was not detected as a dynamic object, as it falls below the C2C distance threshold. The two potential cyclists were correctly classified.

Table 5.3: Detection results for small dynamic objects including total points, correctly detected points, missed points, error percentages, and classification success rate.

Object	Total Pts.	Correct Pts.	Missed Pts.	Error (%)	Classified Correctly
Two People Close	72	57	15	20.83	0/2
Four People Spread Out	101	73	28	27.72	2/4
Dog + Two people	49	23	26	53.06	2/3
Two Cyclists	234	156	78	33.33	2/2

The correctly and incorrectly classified points for large dynamic objects, in comparison to the reference set for large dynamic objects (see Figure 3.4), are shown in Figure 5.3. Correctly classified points are highlighted in green, while misclassified points are shown in red.

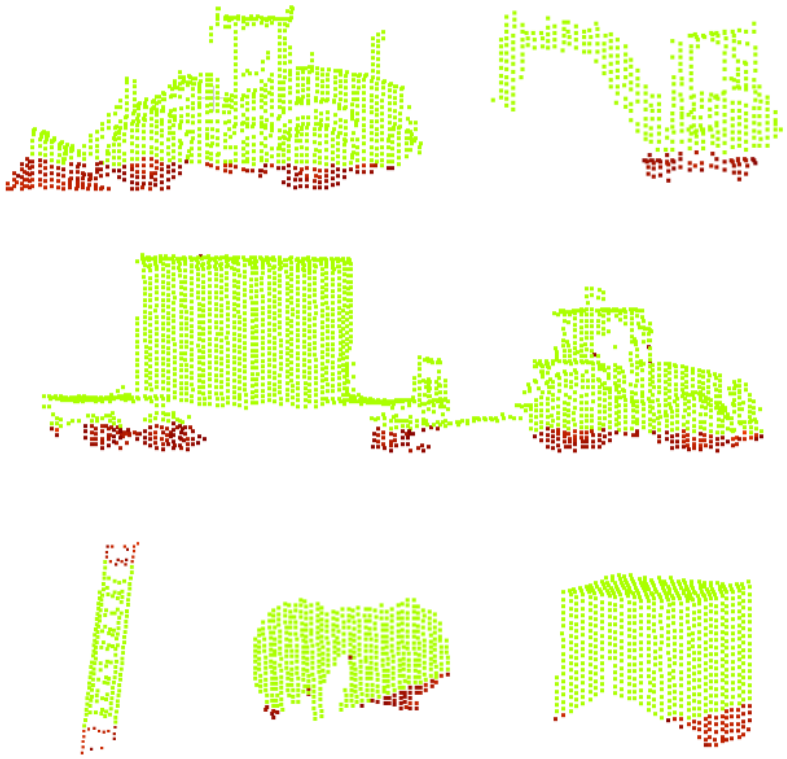


Figure 5.3: Classification of points belonging to large dynamic objects in the reference set. Green indicates correctly classified points, and red indicates missed points. From left to right and top to bottom: bulldozer, excavator, tractor with trailer with cargo, ladder, large barrel and a large crate.

Table 5.4 presents, for each large dynamic object in the reference set, the number of correctly identified points, the number of missed points, the corresponding error percentage, and whether the dynamic object is correctly classified. On average, the reference set for large dynamic objects has a 13.5% error in the number of incorrectly detected points. All large dynamic objects from the reference set were correctly classified.

Table 5.4: Detection results for large dynamic objects including total points, correctly detected points, missed points, error percentages, and classification success rate.

Object	Total Pts.	Correct Pts.	Missed Pts.	Error (%)	Classified Correctly
Bulldozer	1133	925	208	18.36	1/1
Excavator	415	359	56	13.49	1/1
Tractor with Trailer	2254	1904	350	15.53	1/1
Ladder	187	155	32	17.11	1/1
Barrel	594	544	50	8.42	1/1
Crate	722	666	56	7.76	1/1

5.2. Sensitivity Analysis

By performing the Monte Carlo Simulation (see Section 4.6.3), the standard deviation of detected small and large dynamic objects was determined over the period March 18 to 24, 2020. To perform the simulation, the parameters of Scenario B were used (see Table 5.1). Table 5.5 shows the overall standard deviations of the number of detected dynamic objects for both small and large dynamic objects, along with the standard deviations computed for the hours during which at least one dynamic object was detected.

Table 5.5: Standard deviation (number of detected dynamic objects) of small and large dynamic objects in Monte Carlo simulation.

Metric	Small Objects	Large Objects
Overall Std. Dev. [-]	0.34	0.07
Std. Dev. when detected [-]	0.75	0.26

Figures 5.4 and 5.5 illustrate the variability in the number of detected dynamic objects over the period March 18 to 24, 2020, by showing the median ($P50$) and $P10$–$P90$ range of detected small and large dynamic objects, respectively.

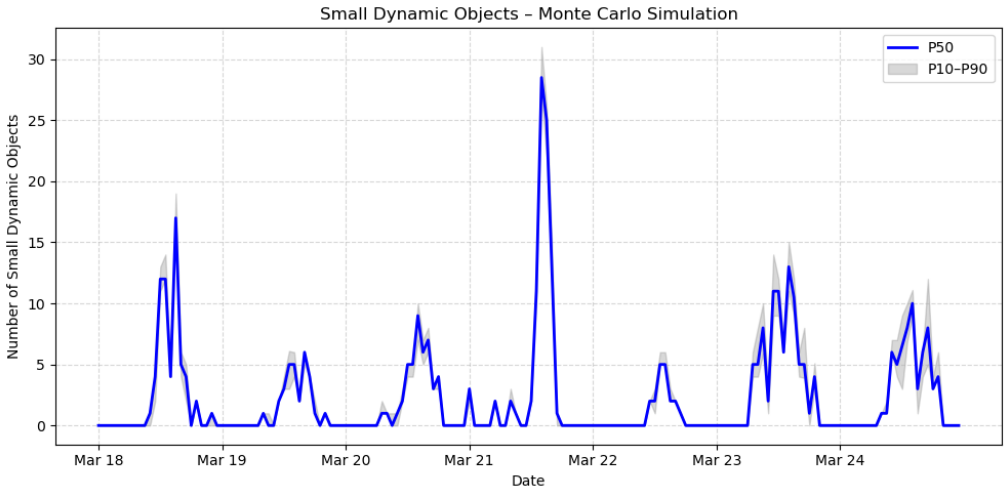


Figure 5.4: Results Monte Carlo Simulation for small dynamic objects over the period March 18 to 24, 2020,

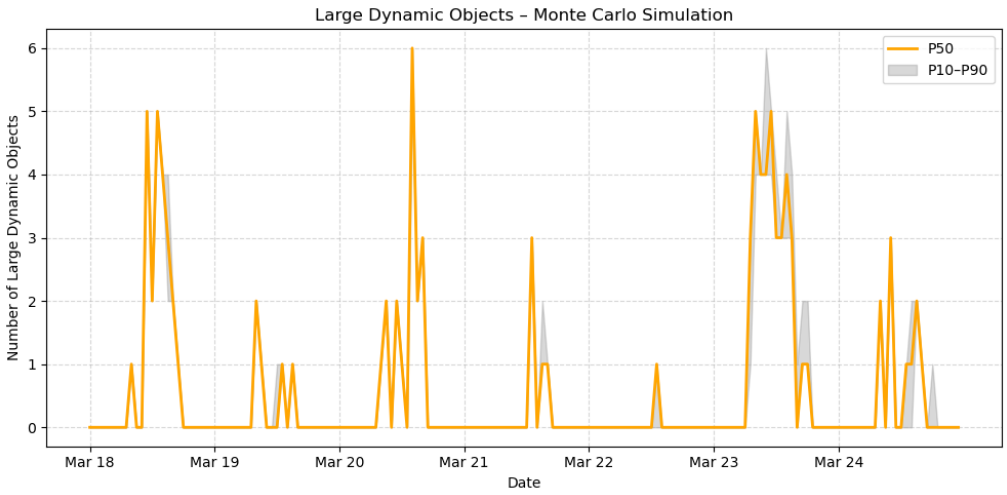


Figure 5.5: Results Monte Carlo Simulation for large dynamic objects over the period March 18 to 24, 2020.

5.3. Analysis of March 2020

The workflow presented in Section 4.1 is applied on a full month of data, namely March 2020, with the parameters corresponding to Scenario B presented in Table 5.1. The number of detected small and large dynamic object per day have been recorded over this period, and are presented in Figure 5.6.

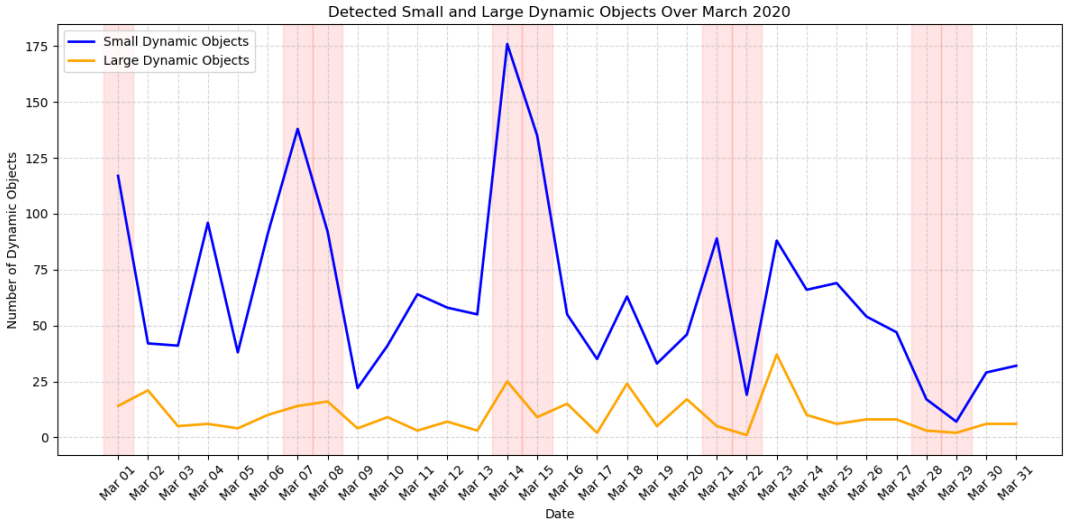


Figure 5.6: Total number of small and large dynamic objects detected over March 2020. Days that fall on a Saturday or Sunday are highlighted in red.

Figures 5.7 and 5.8 present the average number of detected small and large dynamic objects per day of the week ±1 standard deviation. During March 2020, the number of detected small dynamic objects on weekdays (Monday to Friday) has an average standard deviation of 17.7 detected objects, compared to 63.3 detected objects on weekends (Saturday and Sunday). For large dynamic objects, the average standard deviation is 6.8 detected objects on weekdays and 8.4 detected objects on weekends. Table 5.6 presents the standard deviation per day of the week for detected small and large dynamic objects.

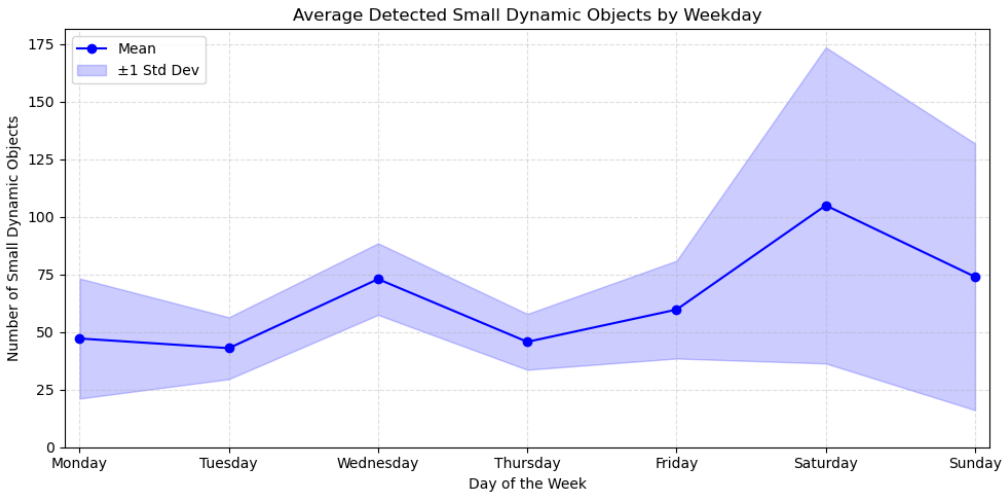


Figure 5.7: Average number of detected small dynamic objects per day of the week, ±1 standard deviation (number of detected dynamic objects).

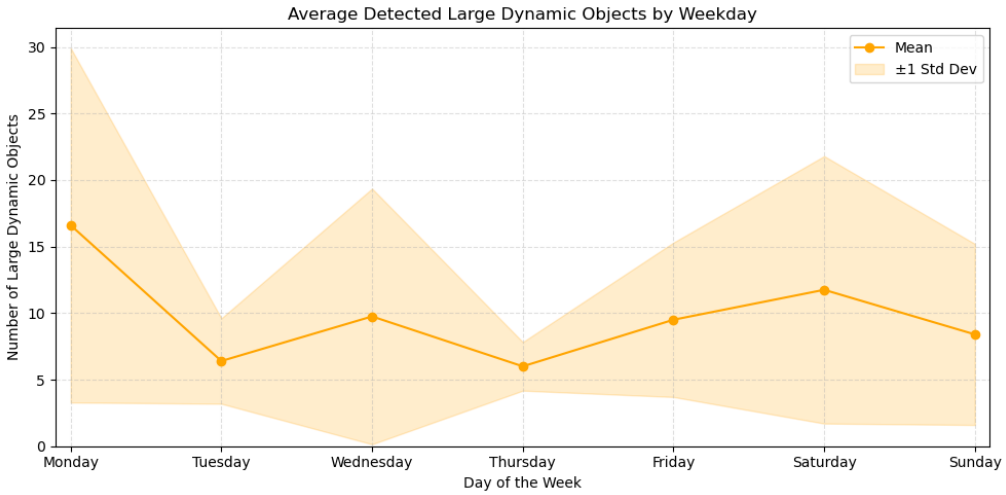


Figure 5.8: Average number of detected large dynamic objects per day of the week, ±1 standard deviation (number of detected dynamic objects).

Table 5.6: Standard deviation (number of detected dynamic objects) of detected small and large dynamic objects by weekday over March 2020.

Day of Week	Small Objects Std. Dev. [-]	Large Objects Std. Dev. [-]
Monday	26.1	13.3
Tuesday	13.4	3.2
Wednesday	15.6	9.6
Thursday	12.1	1.8
Friday	21.2	5.8
Saturday	68.6	10.1
Sunday	57.9	6.8

6

Discussion

This chapter discusses the obtained results and the performance of the workflow. In Section 6.1, the effects of parameters on the accuracy of dynamic object detection are assessed, and the robustness of the selected configuration is discussed. Following this, in Section 6.2, a reflection is provided on specific parameter choices. The limitations and potential improvements in these areas are discussed, alongside their impact on object detection outcomes. Then, in Section 6.3, the impact of non-ground point filtering on the results is discussed. This is followed up by Section 6.4, in which a discussion is held concerning the results obtained of March 2020. Finally, in Section 6.5, the scalability of this research and its applicability in different settings is addressed.

6.1. Evaluation of Parameter Tuning and Performance Analysis

The parameter tuning process showed that excluding points within the pavilion results in the lowest overall classification error. Specifically, the best-performing configuration resulted in a misclassification error of 12.9% for the number of dynamic objects, using a C2C distance threshold of 0.6 m, a DBSCAN ϵ of 0.7 m, and a $MinPts$ value of 9. This also highlights that excluding buildings or other regions prone to noise is beneficial during parameter tuning. Not only does it reduce the overall error, but it also leads to more sensitive parameter settings, which improves the accuracy of dynamic object detection.

The detection errors for large and small dynamic objects should not be interpreted as absolute inaccuracies, but rather as indicative measures of overall detection performance. For example, the computed error for large dynamic objects is 6.9%. However, this does not imply that large dynamic objects were entirely missed. Rather, misclassifications occurred instead of missing these objects. Two main types of errors are observed in the detection of large dynamic objects. First, portions of large dynamic objects partially lie below the C2C distance threshold, which may cause them to lose points during the identification process and drop below 100 points, which leads to their misclassification as small dynamic objects. Second, small dynamic objects located in close proximity can be clustered together, resulting in their incorrect classification as a single large dynamic object. Small dynamic objects however are more likely to be missed. This occurs when they do not have enough points to form a cluster or when they are located too close to the ground, such as the dog shown in Figure 5.2. Moreover, small dynamic objects are also prone to being clustered together if the distance between them falls within the C2C distance threshold, which reduces the number of individually detected dynamic objects, and thus increases the error rate.

In the parameter tuning process, the chosen evaluation criterion was the minimization of the total error in the number of detected dynamic clusters, which is simple and practical to implement, but may not be most effective. Minimizing the error of number of dynamic clusters does not mean directly that the classification error of dynamic objects is minimized. This is due to the misclassification and grouping effects discussed above, as well as the fact that some detected dynamic objects may not correspond to actual moving objects but result from data inaccuracies and changes in the visibility of static objects. These factors negatively impact the tuning process by introducing false positives. An example of a

visibility-related misclassification is shown in Figure 6.1. In this specific case, the misclassification could be the result of a change in the reflectivity of the glass panes installed on the terrace of a beach pavilion. Typically, the glass panes allow a laser pulse to pass through, however for a few hours in the early morning of March 21st 2020 this changed, making the points to appear in the scan data, causing the detection algorithm to mistakenly register parts of these glass panes as newly appeared small dynamic objects. It should be noted that such issues are relatively rare and are exceptions rather than common occurrences. However, they are important to be aware of when analyzing the results of the parameter tuning process.

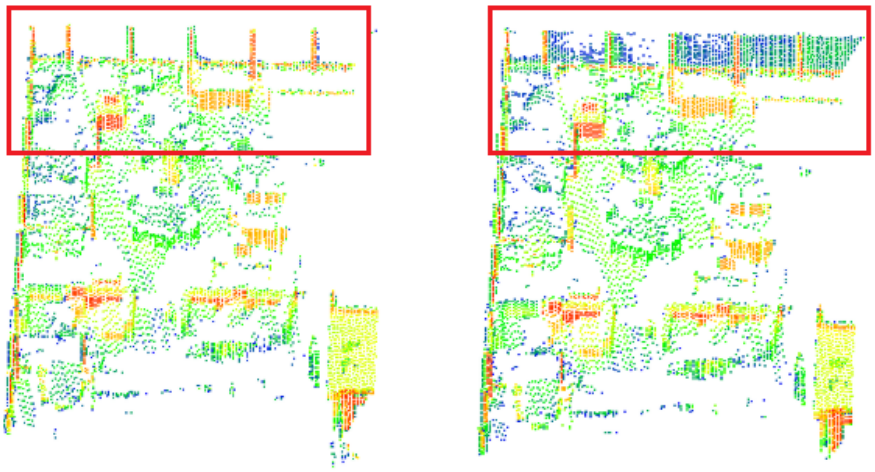


Figure 6.1: Source of a visibility-related misclassification of dynamic objects on the terrace of the beach pavilion. The red box shows the region of error. Respective scans made March 21st 2020 approximately at 00:00 (left) and 01:00 (right). Colors represent point reflectivity, with blue indicating relatively low reflectivity.

Optimization based on the number of dynamic clusters is done due to it being convenient, as it is relatively straightforward to count dynamic objects in the dataset and verify these numbers. An alternative, more robust approach would be to manually segment the dynamic objects in the dataset, and validate whether the points in the identified dynamic clusters overlap with these segmented clusters for a given parameter configuration. This would improve the reliability of quality assessment but would also enable optimization targeted at specific object types by minimizing their respective errors.

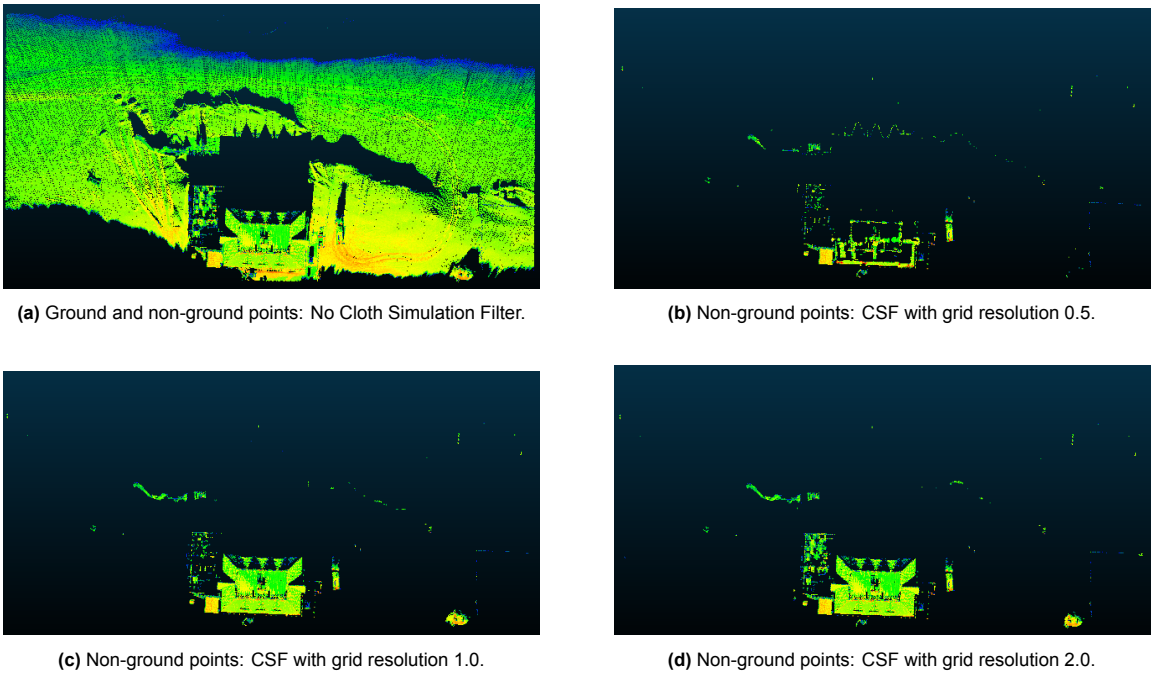
Additionally, optimizing parameters based on a single day of data resulted in a slight increase in average overall error, namely from 12.9% to 17.4%, when evaluating its performance across a full week. This suggests that if the selected day is representative of the broader dataset, the tuned parameters can perform reasonably well over a longer period. However, it remains uncertain whether this configuration would maintain its effectiveness during different times of the year. During summer months for example, increased beach activity may lead to different circumstances, which may need a different parameter configuration for optimal performance.

6.2. Reflection on the Parameter Configuration

The sensitivity analysis performed over a one-week period indicates that the selected parameter configuration is relatively robust. Small variations in parameter values with a normal distribution, namely with a standard deviation of 10%, for both C2C and DBSCAN parameters resulted in limited changes in the final results. Especially the robustness of results for large dynamic objects shows very stable results, which has an overall standard deviation of 0.07. For small dynamic objects this is 0.34. It is important to note that the standard deviation tends to increase when dynamic objects are detected. This shows that the algorithm is more confident under static conditions. Once dynamic objects are identified however, the number of detected objects becomes more sensitive to the parameter configuration, which indicates increased uncertainty in estimating the number of dynamic objects accurately.

6.2.1. CSF Parameters and Building

For the grid resolution of the Cloth Simulation Filter, literature suggested to use 0.5 to get the most accurate representation of the terrain (W. Zhang et al., 2016). However, as shown in Figure 6.2, the performance of the gridsize of 0.5 is not suitable for this dataset, as non-ground points are wrongly classified as ground points, while a grid resolution of 1 and 2 show similar non ground points being selected. Since the grid resolution is closely related to the computational performance of the algorithm, a grid resolution of 2 was eventually selected to increase performance.



(a) Ground and non-ground points: No Cloth Simulation Filter.

(b) Non-ground points: CSF with grid resolution 0.5.

(c) Non-ground points: CSF with grid resolution 1.0.

(d) Non-ground points: CSF with grid resolution 2.0.

Figure 6.2: Results of Cloth Simulation Filter with varying grid resolutions. Data from CoastScan scan on March 18th 2020, 13:00.

Overall, it would be valuable to assess the impact of varying the parameters of the CSF on the parameter tuning and sensitivity analysis. This was not done due to the high computational cost and because the literature provides recommendations for parameter configurations that typically yield accurate results. However, as shown in Figure 6.2, the suggested grid size did not perform best, particularly in the area surrounding the pavilion. This may be due to the roof covering a large surface and being largely flat, causing the algorithm to interpret it as a new stable ground surface, resulting in misclassification. In this case, the pavilion's large, flat roof likely presented a stable surface over which the virtual cloth settled. Because of the fine cloth resolution, the filter may have followed the roof surface too closely, failing to detect it as non-ground. Literature has shown similar effects, where smooth transitions or stable elevated surfaces can mislead the CSF algorithm into classifying non-ground points as ground (W. Zhang et al., 2016; Cai et al., 2023). This shows that the ideal parameter configuration is more ambiguous, and therefore including CSF parameters in the tuning process could add value, especially if following variation in the CSF parameter configuration cause points to be included or excluded from the dataset on an epoch-to-epoch basis.

6.2.2. C2C Distance Parameter

As explained in the methodology, non-ground points are compared to ground points from the previous epoch. As a result, all points within 0.6 meters of the ground are classified as ground points due to C2C distance threshold. This effect is visible in Figures 5.2 and 5.3, where many points are misclassified as ground. While this threshold reduces false positives by avoiding the misclassification of static objects as dynamic, it also causes small dynamic objects (like the dog in Figure 5.2) to go undetected. An alternative approach, discussed in Section 4.4, involves comparing non-ground points across epochs, which would facilitate the detection of smaller dynamic objects located, but also increases the risk of

misclassifying near-ground points, particularly those that are not consistently visible across all scans, as dynamic objects. An example of an object sensitive to this issue is shown in Figure 6.3. Both the poles and ropes of the rope fence are classified as non-ground points, which appear and disappear (partially) in the scan data depending on scan quality, which may lead to incorrectly classifying them as dynamic objects if non-ground points are compared to non-ground points of a consecutive epoch. This could be mitigated by adding an a classification step that identifies and filters out static structures that are occasionally visible. This would allow dynamic objects to be identified more reliably between consecutive epochs.

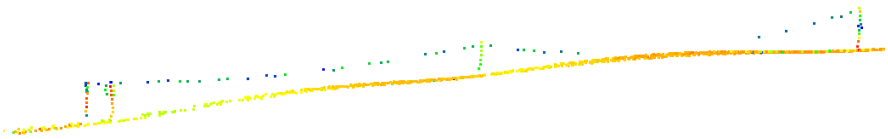
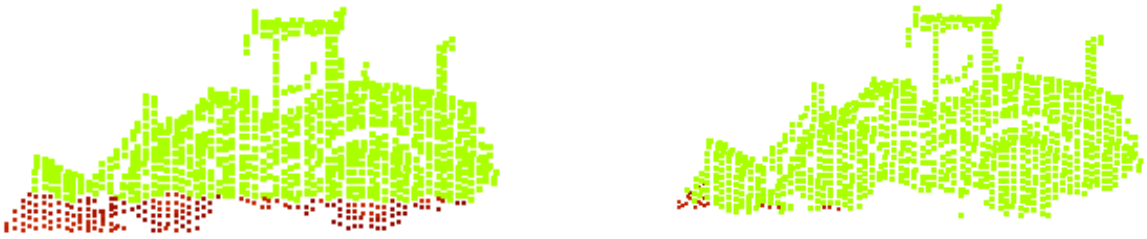


Figure 6.3: Example of a rope fence with poles and ropes which is sensitive to be wrongly classified as a dynamic object when comparing non-ground points of consecutive epochs. Colors represent point reflectivity.

On average, large dynamic objects are clustered more accurately than small dynamic objects when evaluated on a per-point basis, The average error for large objects is 13.5%, compared to 33.7% for small objects of the reference set. This is so, because small objects in the reference set tend to have a greater proportion of their points within the 0.6 m ground threshold. Of the large objects, barrels and crates performed well because they were positioned near occlusions, reducing the number of points classified as static points. The clustering process can be improved upon, by growing identified clusters back down toward the ground by adding points within the C2C threshold that were initially excluded. To exemplify this, the classification error of the bulldozer of the reference set was reduced from 18.36% to 2.12%, as shown in Figure 5.3. This approach can enhance the definition of dynamic objects, but it also introduces additional computational steps and increases the risk of false positives, particularly when dynamic objects are close to one another, and should therefore be implemented with caution.



(a) Bulldozer as performed in current methodology

(b) Bulldozer which has been grown based on C2C distance threshold.

Figure 6.4: Comparison of results with and without growing of clusters to C2C distance.

6.2.3. DBSCAN Parameters

The performance of DBSCAN depends on point cloud density, and all though the research area has a relatively uniform point cloud density (see Section 3.2), it still has density variations. Its fixed ϵ value makes it less effective in low density regions, where small dynamic objects might not be clustered correctly, especially in areas farther from the scanner where point density is lower. A solution to this issue would be to use an adaption on the DBSCAN algorithm, namely to use Hierarchical DBSCAN (HDBSCAN), which doesn't rely on a fixed ϵ and can adapt to local density variations by using a hierarchy of clusters (Campello et al., 2015). This could be especially helpful in making the algorithm more robust when analyzing larger areas with a more varying point cloud density.

Separating nearby dynamic objects is also a challenge of this DBSCAN parameter configuration, partic-

ularly small ones. When objects are less than 0.7 meters apart, they are clustered together. Although lowering ϵ or making it spatially variant might help in some cases, it doesn't solve the issue of ambiguous object boundaries. For example, when two people walk closely side-by-side, it's difficult to define where one ends and the other begins. If such separation is considered essential, deep embedded clustering could offer a solution, as it is capable of learning more complex object boundaries. However, this was not feasible in this research due to the absence of labeled training data.

6.3. Impact of non-ground point filtering on the results

A filter to separate ground from non-ground points was performed to reduce overall noise from ground points, as dynamic points located on the surface are not considered dynamic objects in this research. To test the added value of the CSF, the same detection process is executed both with and without CSF, using identical parameters. Comparing the resulting errors in the number of detected small and large clusters revealed significant differences: when the Cloth Simulation Filter (CSF) is applied, the error is 23.0% for small clusters and 6.9% for large clusters. Without CSF, the errors are 60.5% and 137.3%, respectively. Thus, applying CSF contributed to a reduction in error rate of approximately 62% for small clusters and 95% for large clusters. This suggests that, the algorithm performs significantly better with CSF than without it. This does not come as a surprise, but does provide insights. The clusters exclusively identified by the algorithm without CSF consist of ground points, and can be divided into two categories: ground points that are not dynamic, but are misclassified as dynamic due to occlusions, and ground points that are truly dynamic. Truly dynamic ground points are those affected by factors such as tides, wind, or bulldozers. When analyzing the difference in the observed detection error, these effects can be observed. Within the data, a periodic pattern emerges that appears to correspond closely with the high tide cycles, shown in Figure 6.5, which presents the hourly mean water level with respect to Nederlands Amsterdams Pijl (NAP) at Scheveningen , located approximately 15 km south along the coast (Distance.to, 2025).

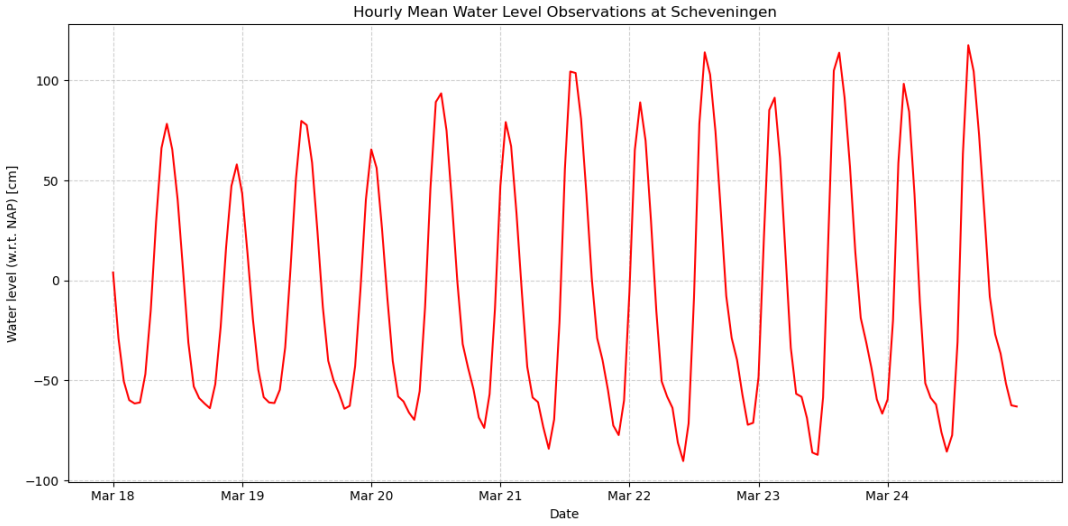


Figure 6.5: Hourly mean water level at Scheveningen from March 18th to March 24th. Data retrieved from Rijkswaterstaat (2025).

Figure 6.6 shows the difference in detected dynamic clusters over time from March 18th to March 24th, with the error of the algorithm applying CSF subtracted from the error of the algorithm without CSF, alongside the moments of highest tide. As mentioned, high tide is closely followed by a peak in the largest error. This delay is expected because dynamic points are only registered by the algorithm when they appear, not when they disappear. Additionally, only high tide is recognized since the selected research area does not include the baseline of the coastline at low tide. What this demonstrates, however, is that the difference between the two methods confirms that the algorithm implementing CSF effectively filters out surface dynamics when separating ground and non-ground points. Additionally, analyzing this difference provides an approach to monitor processes occurring on the beach.

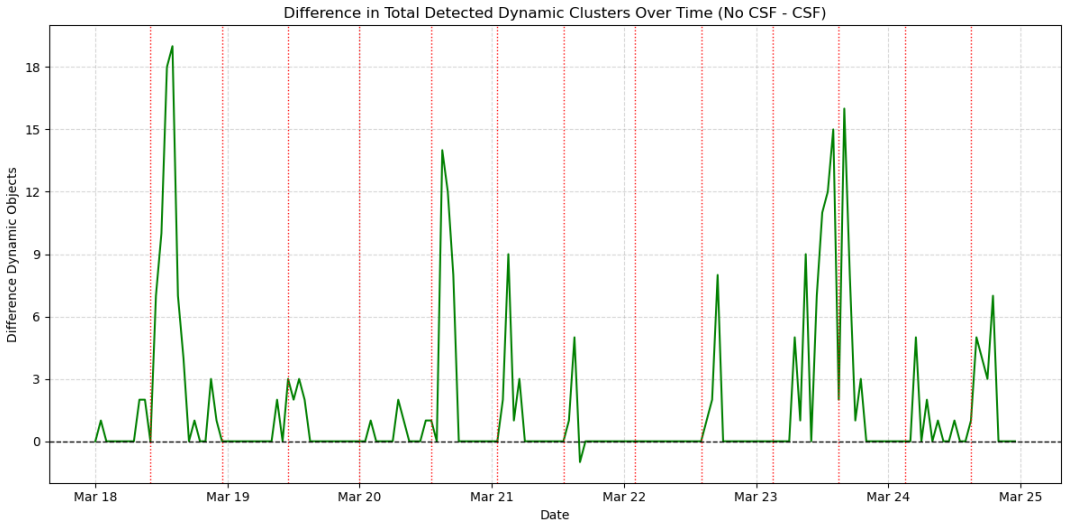


Figure 6.6: Difference in detected number of dynamic objects (No CSF - CSF based algorithm) from March 18th to 24th 2020. Vertical red lines indicate peak of high tide.

6.4. March 2020 Beach Activity and Weather

The results of March 2020 provide insights into patterns of beach usage throughout the week. In particular, a weekly trend can be observed in the number of small dynamic objects (see Figure 5.7). These small dynamic objects can be primarily associated with people visiting the beach for recreational purposes. In contrast, the number of large dynamic objects shows a less distinct trend over the course of the week (see Figure 5.8). This can be attributed to the broader range of activities they represent. Large dynamic objects may include maintenance equipment, such as bulldozers, as well as recreational users, such as cyclists. As such, the activity patterns for large dynamic objects are more balanced. For example, recreational biking may increase on weekends, while beach maintenance requiring heavy machinery is more common during weekdays. As observed, large dynamic objects have a more uniform spread throughout the week compared to small dynamic objects. To be able to draw definitive conclusions about beach usage, it would be useful to classify the large dynamic objects in greater detail, distinguishing between, for example, maintenance vehicles such as bulldozers and recreational cyclists.

Examining small and large dynamic objects more closely shows that the standard deviation for small dynamic objects during weekdays (Monday to Friday) is relatively low at 17.7 detected objects, indicating relatively consistent usage of the beach. Weekends show a higher standard deviation of 63.3 detected objects however, suggesting greater variability in the number of people present. For large dynamic objects, the variability is generally lower, with standard deviations of 6.8 detected objects during weekdays and 8.4 detected objects on weekends, indicating more stable levels of large object activity throughout the week. This could imply that weekday attendance of the beach is more consistent, while weekends are more susceptible to other external factors. To better understand these fluctuations, the number of detected small and large dynamic objects are plotted alongside temperature and precipitation data of Voorschoten, which is the closest KNMI weather station to the beach of Noordwijk, to explore potential correlations (see Figure 6.7) (KNMI, 2025). However, no clear relationship emerges. While precipitation data from early March suggest a decline in beach attendance on rainy days, the limited data range prevents any statistically significant conclusions. Additionally, decline in beach activity is observed around mid-March, despite relatively stable temperature conditions and absence of precipitation during this period. This decline may be attributed to the implementation of COVID-19 restrictions on 15th of March 2020, which may have caused a reduction in the number of people present on the beach during the latter half of the month (Ministerie van Algemene Zaken, 2020). As a result, March 2020 is not a reliable basis for drawing such conclusions, and overall, a longer evaluation period is needed to identify meaningful relationships between weather conditions and beach activity.

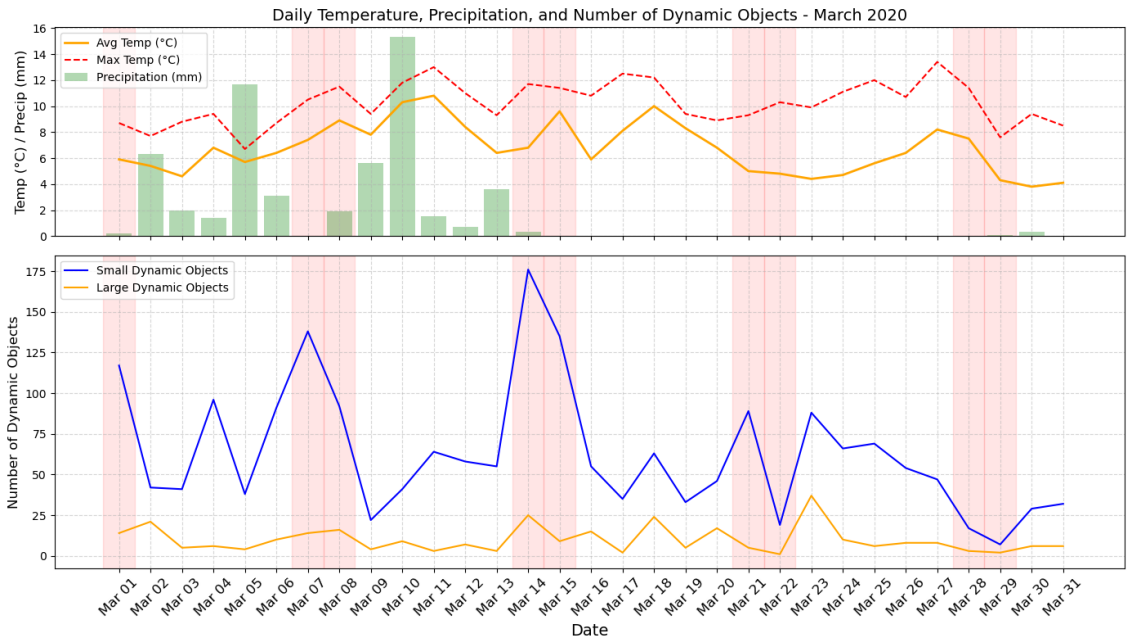


Figure 6.7: Daily (average and maximum) temperature, precipitation, and number of detected small and large dynamic objects over the course of March 2020. Days that fall on a Saturday or Sunday are highlighted in red. Temperature and precipitation data retrieved from KNMI (2025).

6.5. Workflow Scalability and Applicability

The workflow presented in this research was applied to a dataset spanning one month, comprising 733 scans, which excludes 11 scans that were deemed unfit for analysis due to significantly lower point counts. These scans were manually excluded, but for future upscaling, an automated quality control criterion could be introduced, namely, by setting a minimum point count threshold to automatically filter out low-quality scans.

Each cropped scan contained approximately 300,000 points. The overall processing time was around 20 minutes for 733 scans, indicating potential for scaling to a larger number of scans and its applicability to the entire beach area. Scaling the workflow to cover the entire beach would result in increased variability in point cloud density, which may effect the parameter configuration. A potential solution is the use of spatially variant parameterization. However, this would require further investigation into the optimal parameter settings for regions with differing point densities. Additionally, it would be useful to limit excessively high point cloud densities, as these can increase the computational time required to perform the workflow. This algorithm has shown stable performance at densities of approximately 20–30 points per square meter. Thus, down sampling to this point cloud density could be considered to increase performance. Then, post clustering, additional points can be reattached to dynamic objects.

To improve the robustness of dynamic object detection, it may be advantageous to use a nighttime scan from the same day as the reference frame. A nighttime reference does typically not contain dynamic objects, reducing the risk of false negatives caused by objects appearing in the same location, especially on busy days, due to the C2C distance threshold.

This workflow is likely to perform well in settings similar to the beach environment discussed in this research, namely open outdoor spaces with moderate to high point cloud densities and clearly distinguishable dynamic objects. Such environments may include parks, open fields, riverbanks, or tundras, where dynamic objects appear intermittently. However, applying the workflow to more complex environments, such as indoor spaces, environments with complex terrain, and areas with significant line-of-sight restrictions, may result in challenges. It would be valuable to explore how the methodology performs under a broader range of conditions, as this could help identify characteristics under which the approach remains effective for detecting dynamic objects, and where alternative strategies may be needed.

7

Conclusions and Recommendations

This chapter presents the conclusions to the main research question and the supporting research questions. Additionally, it provides recommendations on how the workflow can be improved through future research.

7.1. Conclusions

The purposes of this research is to answer the research question: *How can dynamic objects in LiDAR point cloud data be automatically identified on sandy beaches?* This is addressed by examining how near-continuous terrestrial LiDAR point cloud data can be acquired, identifying the types of dynamic objects present in such data, exploring possible approaches for distinguishing dynamic objects in sequential scans, investigating methods for determining suitable parameter settings, evaluating the performance of dynamic object detection, and assessing the sensitivity of detection performance to variations in those parameters. First, the supporting research questions (RQ1-RQ6) will be addressed, following which, the main research question (MRQ) will be answered.

RQ1: How can near-continuous terrestrial LiDAR point cloud data be acquired?
Near-continuous terrestrial LiDAR point cloud data can be acquired using a permanent laser scanner. Through time-of-flight measurements and the application of environmental corrections, detailed 3D point clouds of coastal environments can be generated. Certain challenges such as line-of-sight limitations, varying point densities, and tilt variations are issues that have to be addressed during data processing.

RQ2: What kind of dynamic objects are expected to be identified in LiDAR point cloud data of sandy beaches?
A permanent laser scanner, the Riegl VZ-2000, conducted hourly scans at Noordwijk from July 2019 to June 2022. The resulting scans contain a range of dynamic objects, from large (e.g., tractors, excavators, crates) to small (e.g., people, dogs). Small and large dynamic objects are differentiated based on the number of points comprising each object, with larger objects represented by a greater number of points in the point cloud. These objects are present within the dataset for a certain time window. The resolution of the scanner and proximity of the objects impact the number of points representing each object, which affects their detectability.

RQ3: What are possible approaches for distinguishing dynamic objects in sequential LiDAR scans?
Distinguishing dynamic objects requires separating them from noise and natural terrain changes. Various approaches exist, from standard change detection methods that compare point-to-point differences, to machine and deep learning approaches that make use of pattern recognition. In this research, Cloud-to-Cloud comparison is selected for its efficiency and simplicity. Ground points are first filtered out using

the Cloth Simulation Filter, and a calibrated distance threshold is then applied to identify spatial deviations that reflect dynamic behavior. Although learning-based methods have a high accuracy and are able to learn complex patterns, the need for extensive labeled data make this less convenient. To improve detection and allow assessment of detected dynamic objects, DBSCAN is applied to group these dynamic points into clusters representing individual moving objects.

RQ4: How can suitable parameters be determined for identifying dynamic objects in LiDAR point cloud data?
Parameter tuning is suitable for select parameters using a reference day which is deemed representative for the data, while visual identification of dynamic objects is used for the optimization process. By analyzing a range of values and combinations for C2C and DBSCAN parameters across different scenarios, namely including or excluding permanent beach structures, an optimal configuration can be obtained by minimizing the cumulative error in the number of clusters that have been detected by the algorithm with respect to the visually determined number of clusters of the reference day. For the Cloth Simulation Filter, literature provides guidelines on which parameter configuration to select.

RQ5: How can the performance of the detection of dynamic objects be evaluated?
Performance can be evaluated by applying the optimized parameters to a week-long dataset from Noordwijk and comparing the detected clusters with visually identified dynamic clusters. In terms of precision for the week-long dataset, errors in the number of detected large dynamic objects are relatively low, with 6.9% error, whereas the error for small dynamic objects is higher, with 23.0% error. This difference in error is largely attributed to small dynamic objects being more frequently clustered together and being located closer to the ground. In an evaluation targeted at a reference set for small and large dynamic objects, large objects are detected with an error of 13.5%, compared to an error of 33.7% for small dynamic objects on a point to point basis. This difference is mainly attributed to their proximity to the ground and lower point counts. Additionally, implementing the Cloth Simulation Filter plays an important role in reducing false positives by filtering out surface changes, resulting in a reduction in error rate of approximately 95% for large dynamic objects and 62% for small dynamic objects.

RQ6: How sensitive is the performance of dynamic object detection to variations in parameter settings?
The sensitivity of the detection performance with respect to parameter settings is evaluated using a Monte Carlo simulation. In this simulation, the tuned parameters are used as the mean values, and fluctuations are introduced using a normal distribution defined by 10% of the mean standard deviation. When applied to a week-long dataset from Noordwijk, this analysis demonstrates that the detection process is relatively robust, with minimal variation in the number of dynamic objects detected. Large dynamic objects show an overall standard deviation of 0.07 detected objects, whereas small dynamic objects are slightly more variable, with a standard deviation of 0.34 detected objects.

MRQ: How can dynamic objects in LiDAR point cloud data be automatically identified on sandy beaches?
As an answer to the main research question, this research shows that by performing preprocessing, applying a Cloth Simulation Filter, performing Cloud-to-Cloud comparison, and clustering objects through DBSCAN, dynamic objects can successfully be automatically identified and separated from the rest of the dataset. Parameter tuning on a representative reference day, validated over a week-long dataset and supported by Monte Carlo sensitivity analysis, shows that the configuration obtained through the tuning process reliably detects large dynamic objects. Smaller dynamic objects are also identified, but with a slightly larger error, due to their proximity to the ground and each other. Additionally, running the workflow on a month-long dataset shows its potential for large-scale implementation. This approach provides a methodology for automatically identifying dynamic objects on sandy beaches in point cloud LiDAR datasets, and can be a useful tool for monitoring activity in coastal and open-space environments.

7.2. Recommendations

The workflow presented in this research has proven suitable for detecting dynamic objects on a sandy beach. However, several limitations and areas for improvement have been identified, leading to recommendations for refining the workflow and highlighting issues that should be addressed in future research.

Adapt Workflow for Small Object Detection
To improve the detection of small dynamic objects, it is recommended to reassess the current implementation of DBSCAN and consider adopting HDBSCAN, which may offer increased sensitivity to smaller clusters. Additionally, focusing on more uniform regions, such as the lower shore face, could lead to more sensitive parameters, which can enhance the detection of small dynamic objects.

Exclude Regions Prone to Noise During Parameter Tuning
When performing the parameter tuning process, it is recommended to use regions prone to noise such as building interiors. This is so, because they are not consistently visible and can lead to poor data quality which may result in classification errors and negatively impact the tuning process.

Improve Dynamic Object Representation
The developed workflow excludes some points associated with dynamic objects. Therefore, it is recommended to grow dynamic objects by adding surrounding points that were excluded, but lie within the Cloud-to-Cloud distance threshold of the object. This increases the completeness of detected dynamic objects. To ensure that only relevant points are included, it is recommended that future research evaluates different growing methods that are able to add points associated with the dynamic object while minimizing the addition of unrelated points, such that the most effective approach can be determined.

Adapt Reference Scan Selection
Dynamic objects in the presented workflow are detected by comparing sequential scans, but the choice of the reference scan impacts detection accuracy. Using the current workflow, dynamic objects may be partially missed during busy periods. To reduce false negatives, it is recommended to use nighttime scans as reference frames for Cloud-to-Cloud comparisons, as nighttime scans typically contain less dynamic activity, and thus improves detection accuracy.

Improve Scalability of Workflow
To ensure robust performance when scaling to analysis of the full three-year dataset, it is recommended to implement filtering of scan data based on the total point count of a point cloud such that low-quality data is excluded. Additionally, it is recommended to investigate the impact of point cloud density on the performance of the workflow and to analyze whether density-based parameter tuning may be useful, such that a larger region with a more variable point cloud density can be examined.

Test Workflow in New Environments
The performance of the presented workflow in settings other than sandy beaches is somewhat uncertain. To verify the robustness of the workflow and assess its broader applicability, it is recommended to test the workflow in other open-space environments as well as in more complex settings, since this may help identify limitations and necessary modifications.

References

Baltsavias, E. (1999). Airborne laser scanning: Basic relations and formulas. *ISPRS Journal of Photogrammetry and Remote Sensing*, *54*(2), 199–214. https://doi.org/https://doi.org/10.1016/S0924-2716(99)00015-5

Cai, S., Yu, S., Hui, Z., & Tang, Z. (2023). Icsf: An improved cloth simulation filtering algorithm for airborne lidar data based on morphological operations. *Forests*, *14*(8). https://doi.org/10.3390/f14081520

Campello, R. J., Moulavi, D., & Sander, J. (2015). Hierarchical density estimates for data clustering, visualization, and outlier detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, *10*(1), 1–51. https://doi.org/10.1145/2733381

Chen, C., Guo, J., Wu, H., Li, Y., & Shi, B. (2021). Performance Comparison of Filtering Algorithms for High-Density Airborne LiDAR Point Clouds over Complex LandScapes. *Remote Sensing*, *13*(14), 2663. https://doi.org/10.3390/rs13142663

CoastScan. (n.d.). *Coastscan: Terrestrial laser scanning data for coastal monitoring* [Accessed: 2025-06-30]. Delft University of Technology. https://coastscan.citg.tudelft.nl/

Distance.to. (2025). Afstand tussen scheveningen en noordwijk [Geraadpleegd op 20 juni 2025]. https://nl.distance.to/Scheveningen/Noordwijk

Ester, M., Kriegel, H.-P., Sander, J., Xu, X., & Institute for Computer Science, University of Munich. (1996). *A density-based algorithm for discovering clusters in large spatial databases with noise*. https://www.aaai.org/Papers/KDD/1996/KDD96-037.pdf

González-Jorge, H., Riveiro, B., Armesto, J., & Arias, P. (2005). Change detection on point cloud data acquired by terrestrial laser scanning. *ISPRS Workshop on Laser Scanning 2005*. https://www.danielgm.net/phd/isprs_laserscanning_2005_dgm.pdf

Guo, Y., Wang, H., Hu, Q., Liu, H., Liu, L., & Bennamoun, M. (2020). Deep learning for 3d point clouds: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *43*(12), 4338–4364. https://doi.org/10.1109/TPAMI.2020.3005434

Jain, A. K., & Dubes, R. C. (1988). *Algorithms for clustering data*. Prentice Hall.

Keskin, S., Kanuk, J., & Erener, A. (2024). Comparative analysis of ALS, TLS and UAV technologies for 3D building modeling and digital twin generation. *aigjournal.com*. https://doi.org/10.5281/zenodo.14555382

Kharroubi, A., Poux, F., Ballouch, Z., Hajji, R., & Billen, R. (2022). Three dimensional change detection using point clouds: A review. *Geomatics*, *2*(4), 457–485. https://doi.org/10.3390/geomatics2040025

KNMI. (2025). Dagwaarden van weerstations. https://daggegevens.knmi.nl/klimatologie/daggegevens

Kuschnerus, M. (2024). *Assessing geomorphologic processes with permanent laser scanning: A case study on the dutch coast* [Dissertation (TU Delft)]. Delft University of Technology. https://doi.org/10.4233/uuid:31b3d8f8-1c0e-4a02-8089-d18034fa8850

Kuschnerus, M., De Vries, S., Antolínez, J. A., Vos, S., & Lindenbergh, R. (2024). Identifying topographic changes at the beach using multiple years of permanent laser scanning. *Coastal Engineering*, *193*, 104594. https://doi.org/10.1016/j.coastaleng.2024.104594

Li, Q., Ma, Y., Anderson, J., Curry, J., & Shan, J. (2019). Towards uniform point density: Evaluation of an adaptive terrestrial laser scanner. *Remote Sensing*, *11*(7). https://doi.org/10.3390/rs11070880

Maneewongvatana, S., & Mount, D. M. (1999). Analysis of approximate nearest neighbor searching with clustered point sets. https://doi.org/10.48550/arXiv.cs/9901013

Ministerie van Algemene Zaken. (2020, November). Aanvullende maatregelen onderwijs, horeca, sport. https://www.rijksoverheid.nl/actueel/nieuws/2020/03/15/aanvullende-maatregelen-onderwijs-horeca-sport

Muralikrishnan, B. (2021). Performance evaluation of terrestrial laser scanners – a review. *Measurement Science & Technology*. https://doi.org/10.1088/1361-6501/abdae3

Provot, X. (1995). Deformation constraints in a mass-spring model to describe rigid cloth behaviour. *Proceedings of Graphics Interface '95*, 147–154.

Rijkswaterstaat. (2025). Astronomisch getij [Accessed: 2025-06-20]. https://waterinfo.rws.nl/publiek/astronomische-getij?parameters=&view=list

Rizaldy, A., Persello, C., Gevaert, C., Elberink, S. O., & Vosselman, G. (2018). Ground and Multi-Class classification of airborne laser scanner point clouds using fully convolutional networks. *Remote Sensing*, *10*(11), 1723. https://doi.org/10.3390/rs10111723

Saltelli, A., Ratto, M., Andres, T., Campolongo, F., Cariboni, J., Gatelli, D., Saisana, M., & Tarantola, S. (2008). *Global sensitivity analysis: The primer*. John Wiley & Sons.

Scikit-learn. (n.d.-a). *Comparing different clustering algorithms on toy datasets* [Accessed: 2025-06-11]. https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html

Scikit-learn. (n.d.-b). *Dbscan — scikit-learn stable documentation* [Accessed: 2025-06-09]. https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html#sklearn.cluster.DBSCAN

Tran, T. H. G., Ressl, C., & Pfeifer, N. (2018). Integrated change detection and classification in urban areas based on airborne laser scanning point clouds. *Sensors*, *18*(2). https://doi.org/10.3390/s18020448

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., & Contributors, S. 1. (2020). *scipy.spatial.cKDTree — scipy v1.10.1 manual* [Accessed: 2025-06-05]. SciPy. https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.cKDTree.html

Vos, S., Kuschnerus, M., Lindenbergh, R., & de Vries, S. (2023, May). A high-resolution 4d spatio-temporal laser scan dataset of the noordwijk beach-dune system, the netherlands: Metadata documentation [Metadata Documentation 04.05.2023].

Vos, S., van IJzendoorn, C., Lindenbergh, R., & de Wulf, A. (2024). Non-uniform dune development in the presence of standalone beach buildings. *Geomorphology*, *466*, 109402. https://doi.org/https://doi.org/10.1016/j.geomorph.2024.109402

Vosselman, G. (2000). Slope based filtering of laser altimetry data. *International Archives of Photogrammetry and Remote Sensing*, *33*, 935–942. https://www.researchgate.net/publication/228719860_Slope_based_filtering_of_laser_altimetry_data

Vosselman, G., & Maas, H.-G. (Eds.). (2010). *Airborne and terrestrial laser scanning*. Whittles Publishing. https://books.google.nl/books?id=J0DNQQAACAAJ

Wani, A. A. (2024). Comprehensive analysis of clustering algorithms: exploring limitations and innovative solutions. *PeerJ Computer Science*, *10*, e2286. https://doi.org/10.7717/peerj-cs.2286

Yan, M., Blaschke, T., Liu, Y., & Wu, L. (2012). An object-based analysis filtering algorithm for airborne laser scanning. *International Journal of Remote Sensing*, *33*(22), 7099–7116. https://doi.org/10.1080/01431161.2012.699694

Zhang, J., & Lin, X. (2013). Filtering airborne lidar data by embedding smoothness-constrained segmentation in progressive tin densification. *ISPRS Journal of Photogrammetry and Remote Sensing*, *81*, 44–59. https://doi.org/10.1016/j.isprsjprs.2013.04.001

Zhang, N. K., Chen, N. S.-C., Whitman, D., Shyu, N. M.-L., Yan, N. J., & Zhang, N. C. (2003). A progressive morphological filter for removing nonground measurements from airborne LIDAR data. *IEEE Transactions on Geoscience and Remote Sensing*, *41*(4), 872–882. https://doi.org/10.1109/tgrs.2003.810682

Zhang, W., Qi, J., Wan, P., Wang, H., Xie, D., Wang, X., & Yan, G. (2016). An easy-to-use airborne lidar data filtering method based on cloth simulation. *Remote Sensing*, *8*(6). https://doi.org/10.3390/rs8060501

A

Description of Point Cloud Data Attributes

Table A.1 summarizes the attributes of the point cloud data collected by the terrestrial laser scanner at the beach of Noordwijk. It includes spatial coordinates, offset time, echo type, deviation, amplitude and reflectance.

Table A.1: Data attributes of the point cloud dataset. Adapted from Vos et al. (2023).

Attribute	Description
X, Y, Z	3D coordinates of laser measurements in local coordinate system (meters), with the origin at the location of the laser scanner. Elevations are given above mean sea level.
Offset time [s]	Relative time of each returned signal within the scan.
Echo type {0,1,2,3}	Position of returned signal within emitted laser pulse: 0 = single, 1 = first, 2 = interior, 3 = last. Multiple echoes occur when laser hits multiple surfaces (e.g., vegetation).
Deviation	Dimensionless estimate of the returned pulse shape after reflection, dependent on surface roughness and structure.
Amplitude [dB]	Amplitude of returned signal at detection threshold, indicating laser return intensity.
Reflectance [dB]	Range-corrected amplitude value provided by TLS manufacturer, representing surface reflectance.

B

Data Specifics of Reference Set

Tables B.1 and B.2 present the dimensions and associated scan identifiers of the dynamic objects from the reference set in the point cloud dataset. Each object's extent in the X, Y, and Z directions (with Z representing height) is listed, along with the specific ID of the CoastScan scan in which it appears.

Table B.1: Size and originating scan ID of small dynamic objects of the reference set.

Object	X [m]	Y [m]	Z [m]	Scan ID
Two People Close	0.76	0.66	1.87	200323_140118
Four People spread out	4.66	0.89	1.76	200318_130049
Two People + Dog	7.28	1.10	1.63	200318_150050
Cyclists	2.22	2.88	1.92	200321_140042

Table B.2: Size and originating scan ID of large dynamic objects of the reference set.

Object	X [m]	Y [m]	Z [m]	Scan ID
Bulldozer	3.08	7.50	3.31	200323_150118.
Excavator	2.08	4.53	2.15	200323_130118
tractor with trailer	4.28	13.86	3.74	200323_100113
Ladder	1.07	0.50	3.23	200323_110113
Barrel	2.28	2.79	2.18	200318_110047
Crate	3.50	3.41	2.69	200323_090113