Master's thesis
Master's Programme in Data Science

Processing 3D point clouds from underground mines

Emma Ilkka

June 26, 2025

Supervisor(s): Dr. Aki Ruotsalainen, Professor Petteri Nurmi, Dr. Agustin Zuñig

Examiner(s): Professor Petteri Nurmi
Dr. Agustin Zuñiga

University of Helsinki
Faculty of Science

P. O. Box 68 (Pietari Kalmin katu 5)

00014 University of Helsinki

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Koulutusohjelma — Utbildningsprogram — Degree programme
Faculty of Science	Master's Programme in Data Science

Tekijä — Författare — Author

Emma Ilkka

Työn nimi — Arbetets titel — Title

Processing 3D point clouds from underground mines

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidantal — Number of pages
Master's thesis	June 26, 2025	48

Tiivistelmä — Referat — Abstract

In the fields of semantic segmentation and surface reconstruction, 3D point clouds collected from underground mines are a less studied subject. Underground mines are unique environments that create challenges for both data collection and subsequent data processing. To provide new insight to this subject, two RandLA-Net models pretrained with large and commonly used datasets, SemanticKITTI and Toronto 3D, are tested for semantic segmentation, and two different surface reconstruction methods, the ball pivoting method and Poisson surface reconstruction method, are implemented on 3D point clouds collected from underground tunnels. The results show that pretrained models trained with data from urban areas could also be useful in semantic segmentation of point clouds collected from underground mines, while the Poisson surface reconstruction method works well for uneven point clouds that have a lot of missing data.

ACM Computing Classification System (CCS):
Computing methodologies → Machine learning → Machine learning approaches

Avainsanat — Nyckelord — Keywords

3D point clouds, 3D mesh, semantic segmentation, surface reconstruction, underground mines

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — Övriga uppgifter — Additional information

Contents

1. Introduction

Underground mining environments can be dangerous places to work. Environmental hazards such as high temperatures, humidity, dust, accumulated water, and harmful airborne gases together with poor lighting, limited network coverage, and the lack of well-established power supply create a challenging working environment for both humans and machines [14]. Therefore, it is important that the data collection phase can be implemented efficiently and safely. Today, lidar sensors are commonly used in different mining environments to map the mining area, calculate excavation volumes, inspect rock slopes, and assess deformations [14]. With vehicle-mounted sensors, human operators may not even need to be in situ, greatly improving the safety of mining work.

In addition to mining environments, lidar-generated point clouds have rapidly become a common source for 3D data that are utilized in urban planning, autonomous driving, forest management, rescue operations, etc. Point clouds have many advantages: they can have an accuracy of millimeters or span areas of multiple kilometers [56]. Sensors can be mounted on hand-held devices, vehicles, or even satellites, making it possible to acquire point data almost anywhere. Lidar-based sensors are invariant to lighting conditions and clouds, so data collection is possible at night or during bad weather conditions. Since sensors have become cheaper, collecting point data has become more accessible and common.

Although a raw 3D point cloud in itself can provide an accurate 3D model of the target, processing the data can make it even more useful. For example, in point cloud segmentation the goal is to identify distinct objects, such as vehicles or vegetation, from the data. In autonomous driving, distinguishing a person from road markings is vital to enable the car to make the right decisions. In an underground mining environment, the separation of man-made objects from the actual tunnel is necessary for an accurate mapping. Multiple deep learning semantic segmentation methods are developed to process point clouds into smaller pieces that offer information to both humans and machines [21].

Although semantic segmentation models are commonly used in autonomous driving etc., their use in underground mining environment is less researched. This may be

because most of the commonly used benchmark datasets either consist of urban outdoor scenes (SemanticKITTI, Toronto 3D, Paris-Lille-3D, Semantic3D etc.) or indoor spaces (S3DIS, ScanNet etc.). Both collecting and labeling point cloud data is a time-consuming work, so using a ready-made dataset with millions of annotated images is a very tempting idea. When used together with a pretrained model, it is possible to move straight to the semantic segmentation part without using much time to data preparations.

Moving those ready-made datasets and pretrained models to a new environment is one of the goals of this thesis. Although utilizing urban outdoor scenes in the semantic segmentation of mining tunnels may sound far-fetched, both environments share similar shapes and objects. Vehicles, machines and humans are commonly seen in both environments, and while poles, fences, and other long and narrow objects are common in urban environments, cables and pipes are common in mines. It is therefore an interesting prospect to see if pretrained models can yield as good results in a new environment.

To bring semantic segmentation into a larger context, the results of semantic segmentation are further used in surface reconstruction. Although surface reconstruction is commonly used to fit scanned data, fill surface holes, and remesh existing models [27], there is no comprehensive research on the strengths and weaknesses of different surface reconstruction methods in the underground mining environment. Combining these two subjects, the use of pretrained models and their use in surface reconstruction, in a less studied environment, will bring some new insights into how to expand the commonly used methods in novel areas.

1.1 Research questions

The goal of this thesis is to answer the following research questions:

- Can pretrained models be a valid choice for semantic segmentation for point clouds collected from underground tunnels?
- What surface reconstruction methods work best for underground tunnels?

1.2 Contributions

Based on the research questions, the contributions of this thesis are following:

- Building a pipeline that allows analyzing and processing 3D point cloud data into 3D mesh

- Researching how pretrained models used for semantic segmentation of urban areas perform in underground mining environment
- Comparing different methods for 3D mesh creation in underground mines and evaluating their strengths and weaknesses

1.3 Outline

This thesis has the following structure:

- In chapter 2, the general properties of 3D point clouds, their segmentation, and use in underground mines are reviewed
- In chapter 3, a pipeline to process point cloud data from underground mines to 3D mesh is introduced
- In chapter 4, the data and the results of the process are presented
- In chapter 5, the strengths and weaknesses of the data and methods are discussed
- In chapter 6, conclusions and summary of the work are provided

2. Point clouds

2.1 Characteristics of point clouds

3D point clouds are unstructured and unordered collections of points in a 3D space usually picturing a surface of some object or area (Figure 2.1). The minimum information that is available from each point is its location in the 3D coordinate space, in relation either to the measuring device or a global coordinate system. Additional information that can be acquired simultaneously with the point cloud collection phase or added afterward includes, for example, colors, normals, and timestamps.

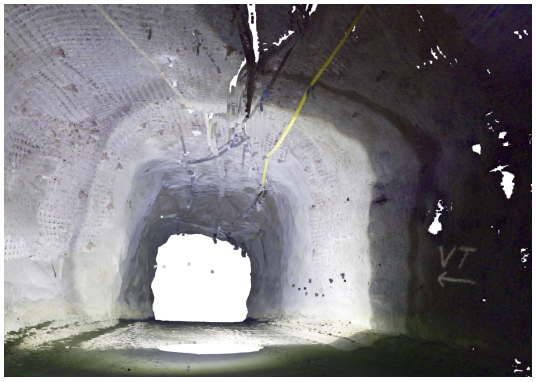
The density of points in point clouds depends on the sensor and the distance between the sensor and the target. Point cloud data collected with airborne or space-borne sensor platforms tend to be sparse (less than 20 points per m^2) due to the long distance. When sensors are moved closer to the target, dense point clouds (more than 100 points per m^2) are available. In dense point clouds, the accuracy is calculated in centimeters or even millimeters. The development of new lidar and RGB-D camera techniques have accelerated the production and availability of large-scale dense point clouds, which in turn has opened up new possibilities for 3D applications, such as 3D (semantic) segmentation.

2.2 Acquiring point cloud data

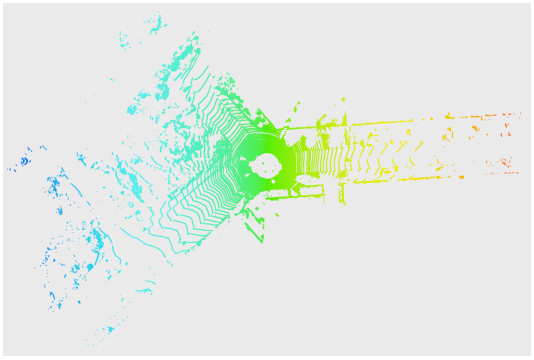
The acquisition methods for point clouds can be divided into four categories: image-derived methods; lidar; red, green, blue-depth (RGB-D) cameras; and synthetic aperture radar (SAR) systems [56]. All of these methods have their strengths and weaknesses (see Table 2.1).

2.2.1 Image-derived methods

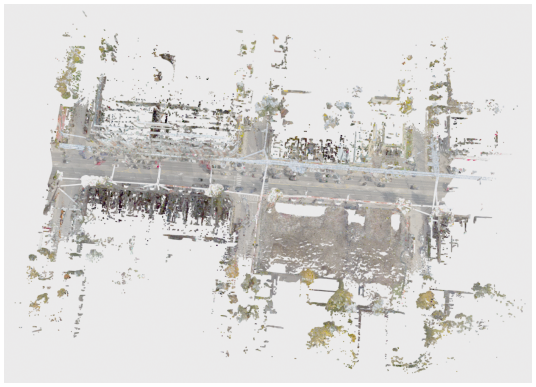
The first method for acquiring point clouds was to generate them indirectly from spectral images. Using a stereotype image derived either through airborne or space-borne systems, unmanned aerial vehicles, or close-range systems, 3D point information is au-



(a) Inside view of a tunnel



(b) Street view from SemanticKITTI dataset



(c) Street view from Toronto 3D dataset

Figure 2.1: Examples of point clouds

tomatically or semi-automatically calculated from images based on different theories. The quality of point clouds depends on the quality and characteristics of images, such as image resolution, lighting conditions, stereo angles, and textures, and requires intense manual work. Methods like dense matching, multi-view stereovision (MVS) and structure from motion (SfM) have made obtaining and processing large-scale point clouds easier, but their quality is not as good as those generated with traditional photogrammetry or lidar [56].

2.2.2 Lidar methods

Lidar systems utilize laser energy to calculate the distance between the sensor and the object. Most of the systems are pulse-based, where a pulse of laser energy is sent towards the target and the distance is calculated based on the traveling time of the pulse. The point density varies greatly depending on the sensors and platforms.

Lidar systems can be built on multiple different platforms. In airborne lidar scanning (ALS) data is collected with airborne platforms. The density of ALS point clouds is typically sparse because of the long distance between the sensor and the target. ALS point clouds resemble the traditional image-derived point clouds, but they are more costly to acquire and do not normally contain spectral information. However, this limitation can be circumvented with multispectral lidar scanning that collects data using different wavelengths.

Terrestrial or static lidar scanning (TLS) utilizes a stationary sensor commonly mounted on a tripod. It is used to scan close or mid-range targets and therefore can produce accurate, high-density 3D point clouds. Because of it's stationary nature, it is only suitable for scanning small areas.

In mobile laser scanning (MLS), the lidar system is attached in a moving vehicle. Currently, MLS is very useful for generating high-quality 3D maps that are needed for autonomous driving. Although its use is usually limited to urban environments and those accessible by roads, unmanned lidar scanning (ULS) takes advantage of drones and other unmanned vehicles. These systems have become popular because they are cheap and flexible. Compared to traditional airborne lidar scanning, ULS can be operated at a lower altitude, allowing greater accuracy [56].

2.2.3 RGB-D methods

An RGB-D camera is used to collect both RGB and depth data. Similarly to lidar, depth information is acquired by computing the distance between the sensor and the target. Similarly to traditional photogrammetry methods, RGB-D cameras do not generate point clouds themselves, but the resulting image can be processed into a

point cloud, since the position of each pixel in 3D space is known relative to the center point of the camera. RGB-D cameras are cheaper than lidar systems, but their use is mostly limited in close-range applications, like indoor environments, and their accuracy depends on the resolution of the camera [56]. The possible poor lighting conditions limit the use of RGB cameras in underground mines.

2.2.4 SAR methods

Synthetic aperture radars are usually mounted on moving platforms, such as aircraft or spacecraft. While the radar moves over a target, the location of the antenna changes, which allows multiple records from different angles to be combined. The result is a 2D or 3D reconstruction of objects. Interferometric SAR is commonly used in remote sensing to generate maps of surface deformations and digital elevation. It has two major techniques, SAR tomography and persistent scatterer interferometry, to create 3D point clouds [56]. SAR tomography has a point density similar to ALS, and with error-correction techniques, it can also reach the same accuracy.

SAR tomography is especially useful for monitoring urban areas and human-made structures. With multiple viewing angles, it is possible to create detailed reconstructions of multiple buildings. Space-borne SAR can also provide temporal information that allows for monitoring changes in building complexes. The negative side is that, with the limited orbit spread and a small number of images, the accuracy diminishes further away from the azimuth. Multiple scattering may also lead to ghost scatterers far away from their actual 3D location.

In the future, multi-source data fusion should grow in popularity. It is already a trend, for example, in remote sensing research and autonomous driving. Autonomous vehicles need to navigate, perceive their surroundings and forecast the actions of other vehicles without help from human driver. This is achieved with the help of lidars, RGB-D cameras and other sensors working together to enable perception of the surroundings, behavior prediction and motion execution [1]. Especially challenging is the perception of the surroundings: urban areas have a lot of moving objects and people that may behave unexpectedly. In addition, perception must be reliable under different weather and lighting conditions, and decisions based on the gathered information must be made in real time. All this requires a seamless cooperation from both sensors and algorithms.

2.2.5 Methods used in mining environments

The commonly used point cloud acquisition methods in mining environments are TLS and MMS (Mobile Mapping Systems), which can combine both lidar and camera sensors on a moving platform [14]. The high density and high accuracy data provided by

Table 2.1: Point cloud acquisition methods

Method	Strengths	Weaknesses
Image-derived	color information, suitable for large areas	quality depends on image characteristics
Lidar	high accuracy, invariant to lighting and weather conditions	can be expensive
RGB-D	color and depth information, cheap	quality depends on image characteristics, close-range
SAR	temporal information, multiple viewing angles	expensive, ghost scatterers

TLS is used to map the mining area, calculate excavation volumes, inspect rock slopes, and assess deformations. Although TLS is mostly suitable for mapping small areas, MMS is a better option for large areas and (harsh) underground environments. MMS sensors are small and fast while providing an accuracy of a few centimeters, which is helpful in hazardous conditions or when minimizing disturbance for mining operations. They can make use of simultaneous localization and mapping (SLAM), which is useful in underground mines, where traditional positioning systems may not be available.

2.3 Preprocessing point cloud data

Real-world point clouds are usually noisy and poor quality for multiple reasons. In underground mines, the environment itself makes measuring difficult due to elevated temperatures, humidity, dust, accumulated water, and harmful airborne gases [14]. Technical challenges arise from the absence of well-established power supply infrastructure, poor lighting, missing coverage of the global positioning system, and the limited wireless network. Moreover, moving vehicles, individuals and other objects may obscure part of the tunnel, leaving gaps and noise in point cloud.

Because all of this, point clouds usually needs some pre-processing before the data can be used in further analysis or visualization. Common methods used are voxelization, outlier removal, and smoothing and normal estimation [31]. In voxelization, a 3D box, or voxel, grid is created over the points, and all points inside a voxel are averaged to create just one point, creating an equal-distance grid of points. Voxelization is a popular method for downsampling, reducing the details of point cloud, but with a small enough voxel size, not too many details are lost.

If a point cloud consists of multiple scannings, multiple points may have almost exactly the same coordinates. Removing points that are too close to each other saves computing power. Moreover, measurement errors can cause point cloud to have erranous points that do not represent any real objects. These points are far apart from other points and can be removed, for example, using statistical analysis on each point's neighborhood and removing the points that do not pass certain criteria. In density-based clustering, points that are close together are grouped together, while points that do not have any neighbors are marked as outliers. DBSCAN [18] is one of the most popular algorithms used for clustering. In short, it computes the number of neighbors each point has in a distance ε and if this number is greater than a set threshold, these points are marked as core points that form a cluster. Other points may be included in the nearby cluster if the distance is less than ε, otherwise they are marked as noise.

In addition to outliers, measurement errors can cause noise, which is prominent especially on smooth surfaces. Smoothing a point cloud helps to remove unwanted noise. In normal estimation, the surface normals are computed for each point. They are vectors that are perpendicular to the tangent plane and provide information, for example, of the curvature of the surface. Normal estimation is necessary for many applications in surface reconstruction, rendering, and virtual reality [57].

2.4 3D point cloud semantic segmentation

In semantic segmentation, a 3D point cloud is divided into separate subsets based on the semantic meaning of different points. It is used to further analyze point clouds and turn unorganized points into understandable objects. The methods can be divided into three different categories: projection-based, discretization-based, point-based and hybrid [21]. Some of the recent semantic segmentation methods are listed in Table 2.2.

2.4.1 Projection-based methods

In projection-based methods, a 3D point cloud is usually projected into 2D images, such as multi-view and spherical images. Lawin et al. [33] were the first to project a 3D point cloud onto 2D images using multiple virtual camera views, while Boulch et al. [9] generated several RGB and depth images using multiple camera positions. What is common to the multi-view segmentation methods, is that they are sensitive to viewpoint selection and occlusion as well as information loss due to the projection step. This last problem can be somewhat alleviated using spherical images instead of multi-view. However, problems with discretization errors and occlusion still persist.

2.4.2 Discretization-based methods

In discretization-based methods, a 3D point cloud is converted into a dense or sparse discrete representation. Dense discretization representations take advantage of voxelization to turn a point cloud into a regular data format, suitable for standard 3D convolutions. Tchapmi et al. [49] join the advantages of neural networks, trilinear interpolation, and fully connected Conditional Random Fields in a semantic segmentation method called SEGCloud, while Dai et al. [16] propose ScanComplete to complete a 3D model from an incomplete 3D scan along with per-voxel semantic labels. Although the 3D shape is preserved, voxelization is still an information loss and a source of discretization artifacts. It is also sensitive to grid resolution, since high resolution requires more memory and computing power, while low resolution introduces a loss of details.

2.4.3 Point-based methods

Point-based methods make direct use of irregular point clouds and can be divided into point-wise MLP methods, point convolution methods, RNN-based methods, and graph-based methods [21]. Since point clouds are orderless and unstructured, they are not suitable for standard 3D convolutions and are therefore converted into regular 3D grids or collection of images [41]. However, this can lead to the loss of details as well as unnecessarily voluminous data. Qi et al. [41] developed PointNet to process raw point cloud data and learn per-point features. These point clouds have three main properties: 1) they are sets of points without specific order, 2) points that are close together form meaningful subsets, meaning there are interactions between points and 3) since the point set is a geometric object, it should be invariant to transformations. Their approach uses a single symmetric function, max-pooling. The network then learns a set of optimization functions and criteria, used to select informative points from the point cloud. Optimal values are aggregated in the final fully connected layers, enabling both shape classification and shape segmentation.

Although PointNet and many later approaches have worked very well on small 3D point clouds, they cannot be directly extended to larger clouds [23]. Some recent approaches like SPG [32], FCPN [43], PCT [11] and RandLA-Net [23] focus on larger point clouds with millions of points. SPG preprocesses points into super graphs before feeding them to neural network, while FCPN and PCT use voxelization and point level networks. RandLA-Net attempts to build a neural architecture that can process large point clouds without any preprocessing.

2.4.4 Hybrid methods

To better make use of all the available data, several methods are developed to learn multi-modal features from 3D point data. These methods usually combine 2D and 3D data in different ways for better results. Dai and Nießner [15] use RGB color information from RGB-D images by first extracting 2D feature maps and then back-projecting them into a 3D space. A 3D convolutional network can therefore learn from both back-projected 2D features and 3D geometric features. Chiang et al. [12] have a similar idea, creating a network that learns 2D textural appearance and 3D structural features in a unified framework. In their work, Jaritz et al. [26] aggregate 2D multi-view image features into 3D point clouds and then use a point-based network to predict 3D semantic labels.

Although the number of different semantic segmentation methods is large, most of them are only tested on commonly used datasets, which limits their use in real-life situations [1]: Many of the datasets are small and clean, while in real life the collected information can be large-scale and noisy; most of the data sets are gathered during good weather conditions, while in real life the weather can be rainy, foggy etc; the pre-collected datasets cannot capture everything, so identifying unknown objects may be a problem. In addition, effectiveness vs. efficiency is an important aspect, for example, in autonomous driving, where an algorithm must be able to process data in real time.

2.4.5 Semantic segmentation in underground mines

Semantic segmentation in underground mines is a difficult task due to the lack of specialized evaluation datasets, varied lighting conditions, and significant disparities in the shapes of detected objects [52]. In addition, many works using semantic segmentation in underground mines focus on solving a specific problem instead of segmenting the entire tunnel. Tong et al. [50] use semantic segmentation to detect underground mine roads, while other articles focus on detecting rocks or gangue from images ([46], [37], [53]). An exception is Wang et al. [52], whose Fast Adaptive Deep Dual-resolution Network (FA-DDRNet) is a real-time semantic segmentation algorithm for underground mines. This algorithm uses 2D images and takes into account the varied lighting conditions that create a challenge in underground mines.

2.5 Surface reconstruction from 3D point clouds

Surface reconstruction is an extensively studied part of computer graphics that is used to fit scanned data, fill surface holes, and remesh existing models [27]. Since the

Table 2.2: Examples of deep learning semantic segmentation methods

Projection-based	2015	MVCNN [47]
	2017	SnapNet [9]
	2017	SnapNet-R [20]
	2018	SqueezeSeg [55]
Discretization-based	2015	VoxNet [38]
	2017	SEGCloud [49]
	2018	PointGrid [34]
	2019	LatticeNet [44]
Point-based	2017	PointNet [41]
	2017	PointNet++ [42]
	2018	PointCNN [36]
	2018	RSNet [24]
	2018	SO-Net [35]
	2019	A-CNN [30]
	2019	DGCNN [54]
	2020	RandLA-Net [23]
Hybrid	2018	3DMV [15]
	2019	UPB [12]
	2019	MVPNet [26]

original scanned point data may be noisy due to scanning errors or difficult scanning conditions, have missing parts because of obstructions, and have nonuniform sampling, multiple different approaches have been researched to provide methods that not only try to reconstruct the surface as closely to the original as possible, but are also able to fill missing data.

2.5.1 Interpolating methods

Interpolating methods typically use combinatorial structures to produce 3D triangle meshes by interpolating all or most of the points. Both Boissonnat [8] and Kolluri et al. [29] propose the use of volume-based Delaunay triangulations, while Edelsbrunner and Mücke [17], Bajaj et al. [5] and Bernardini et al. [7] base their interpolation methods on alpha shapes. The third commonly used approach, Voronoi diagrams, have been proposed, for example, by Amenta et al. [3] and Amenta et al. [4]. Although interpolation as an approach is simple, the resulting 3D mesh may be jagged if the data is noisy [27]. Therefore, some level of post-processing may be necessary.

2.5.2 Approximating methods

An opposite approach to interpolation is to reconstruct an approximation surface directly, using all or most of the points at the same time. In global approach, the implicit function defining the surface is usually based on the sum of radial basis functions (BRFs) ([39], [10], [51]), while in local approach subsets of nearby points are considered separately. These methods include computing a signed distance function ([22], [13]), approximating nearby points by moving least squares ([2], [45]) and subdividing the space with an adaptive octree [40]. Kazhdan et al. [27] take advantage of both global and local approach with their Poisson reconstruction method.

2.5.3 Surface reconstruction in underground mines

Although surface reconstruction itself is an important research subject, its applications in underground mines seem to be more of a side note than a actual research subject. For example, Dabek et al. [14] mention that they used Screened Poisson algorithm [28] in mesh creation, while Grehl et al. [19] used a method that was based on Delaunay tetrahedralization [25]. Neither of these articles gives any reason why the aforementioned methods were chosen. Outside of the mining environment, there are some tunnel reconstruction methods that are applied to circular or ellipsoid-shaped shield tunnels (for example [59]). In general, there is no consistent research on how different surface reconstruction methods fare in an underground mining environment.

3. Building a pipeline to process point cloud data from underground mines

With the challenges of data acquisition and missing data, the lack of evaluation datasets and the fact that very few semantic segmentation or surface reconstruction methods are actually tested in underground mining environment, processing point clouds from underground mines is a nontrivial task. Existing preprocessing techniques, such as voxelization and normal estimations, should work fine with tunnel data, but selecting a correct semantic segmentation or feature extraction method is more difficult. Therefore, instead of guessing for the best method, a couple of commonly used semantic segmentation datasets together with a semantic segmentation model that has reached good results with different datasets are selected for feature extraction. For surface reconstruction, two different methods with opposing approaches are chosen. This will be a novel approach and will provide some insight into the processing of point clouds from underground mines.

3.1 Overview of the pipeline process

The process starting with raw 3D point cloud and ending in 3D mesh is divided into three steps:

1) Preprocessing step: Raw point cloud data is run through some preprocessing before being used for further analyzing. The methods used include voxelization and computing surface normals.

2) Feature extraction step: Objects that are not part of the tunnel (pipes, cables, vehicles, persons etc.) are extracted from point clouds before the final step. The methods used include clustering and semantic segmentation.

3) Surface reconstruction step: After point clouds are preprocessed and unnecessary

objects are removed, a 3D mesh is constructed. The methods used include the ball pivoting method and the Poisson surface reconstruction method.

3.2 Preprocessing the tunnel data

The preprocessing phase of point cloud analysis usually includes voxelization, outlier removal, and possibly smoothing and normal estimation. A voxel can be considered as a 3D counterpart of a 2D pixel, basically a cube. The process of voxelization of a point cloud involves four general steps [58]. First, the bounding box covering the entire point cloud is calculated. Then, the space defined by the bounding box is divided into regular 3D cuboids. In the third step, the point cloud is segmented into smaller pieces by these 3D cuboids. And finally, each point is represented by one of the cuboids. Although from a geometric standpoint a voxel has six faces, eight vertices and twelve edges, only the center point or corner points are used to represent a voxel.

This method of compressing multiple points into one has many benefits [58]. A voxel inherits the attributes of its points, for example, color, intensities, and semantics, and the cubic shape means that calculating the number of points inside a voxel can be used to calculate local densities. Approximating points inside a voxel by plane models or surfaces helps to reduce outliers and noise. Since voxels are indexed during voxelization, the spatial topology is preserved in a 3D grid or tree structure, which makes finding the neighboring voxels simple. And finally, the size of a voxel can be determined beforehand, making it easy to store data at different resolutions.

3.3 Extracting tunnel features

While deep learning segmentation methods are the current state-of-the-art in feature extraction, more traditional methods, such as clustering, are still worth using. A simpler algorithm can work as well as a more complicated one. Therefore, clustering is included in the feature extraction part as a baseline for feature extraction.

3.3.1 Extracting features with clustering method

Although clustering is usually used for preprocessing and cleaning data, it could also be used to cluster data in preprocessed point clouds. In theory, since tunnel walls are usually smooth without clear protrusions, different machines, cables, railings, etc. form clear entities that could be separated from the tunnel walls. DBSCAN [18] is a well-tested and popular algorithm used in clustering and was therefore considered an optimal choice.

DBSCAN is a density-based clustering algorithm that is designed to find arbitrary-shaped clusters from large datasets. The basic idea is to classify points as core points, border points, or noise. Core points need to satisfy two conditions: they have minimum number of neighboring points and all those points are within minimum distance. If a point q is within a minimum distance from a core point p, it is considered directly density-reachable. If a point q is not within the minimum distance from point p, but there is a chain of points $p_1...p_n$, where $p_1 = p$ and $p_n = q$ and p_{i+1} is directly density-reachable from p_i, q is considered density-reachable from p. If a point does not satisfy the conditions of a core point, but is density-reachable, it is classified as a border point. Two border points may not be density-reachable from each other, but they may still belong in the same cluster through density-connectivity. Formally, point p is density-connected to point q if there is a point o from which points p and q can be density-reached. Therefore, a group of points form a cluster if 1) any of the points can be density-reached from another point and 2) all points are density-connected to each other. Points that do not satisfy these conditions do not belong in a cluster and are classified as noise.

3.3.2 Semantic segmentation with RandLA-Net

Although clustering is used as a baseline for feature extraction, one of the goals of this thesis is to find out if a better result could be reached with machine learning algorithms and pretrained models. Since mining tunnels can be very long and collected point clouds could contain a lot of points, the model should be able to process large-scale point clouds efficiently. One such model is RandLA-Net that has both Tensorflow and PyTorch implementations in Open3D library. It has also reached good results in multiple semantic segmentation tests [21], and there are preset weights available for different datasets.

RandLA-Net utilizes random sampling to reduce point density together with local feature aggregation that retains prominent features. Although random sampling (RS) is a faster and more efficient way to sample points compared to techniques like farthest point sampling (FPS), generator-based sampling (GS), and continuous relaxation based sampling (CRS), it may lose useful point features due to its randomness. RandLA-Net solves this problem with a local feature aggregation module. The module has three neural units: 1) local spatial encoding, 2) attentive pooling, and 3) dilated residual block. In short, a local spatial encoding unit and an attentive pooling unit work together to find and aggregate local geometric patterns and features of K nearest points and turn them into an informative feature vector. Multiple local spatial encoding units and attentive pooling units with a skip connection are then stacked as a dilated residual

block. The idea of stacking is to increase the receptive field for each point, so that geometric details are better preserved through downsampling. In the end, RandLA-Net is implemented simply by stacking multiple local feature aggregation modules and random sampling layers.

3.3.3 Using pretrained models in semantic segmentation

Most of the available datasets and pretrained models are tailored for self-driving vehicles and focus on outdoor images. To test their usability in underground mines, two of such datasets were selected: SemanticKITTI and Toronto 3D.

SemanticKITTI [6] is a large dataset with 4549 million annotated points and 28 annotated classes, based on the odometry dataset of the KITTI Vision benchmark. Point clouds are generated with a commonly used automotive lidar and show inner city traffic, residential areas, highway scenes and countryside roads. Points are classified in 28 classes and 7 root categories that are ground, structure, vehicle, nature, human, object and outlier. The most frequent classes are road, sidewalk, building, vegetation and terrain, while motorcyclist has the smallest number of annotated points.

Toronto 3D [48] is a large-scale point cloud covering a 1 km street view in Toronto, Canada. It consists of around 78.3 million annotated points and 8 annotated classes. The data is collected with a vehicle-mounted MLS system, so that the dataset covers the whole range of the sensor approximately 100 meters away from the road centerline. The points were annotated either road, road marking, natural, building, utility line, pole, car, fence or unclassified.

Toronto 3D has a variation in point density, since it uses points from the whole measurement range of the lidar without trimming and utilizes repeated scans from point cloud collection. The point density in center areas can be more than 10 times higher than in far away areas due to the repeated scans when vehicle was stopped in intersections during the data collection.

3.4 Reconstructing point cloud surface into 3D mesh

The final phase of the pipeline is to reconstruct the surface of the processed point cloud. Most of the 3D surface reconstruction methods are tested on small objects and how well they could reconstruct the outer surface of those objects. In the case of underground mines, it is the inner surface that needs to be reconstructed. Moreover, there are areas that don't have any surface, like the ends of the tunnels and possible ventilation shafts, which makes this a nontrivial task. Two different types of methods are considered: the

ball pivoting method [7] and the Poisson surface reconstruction method [27].

The ball pivoting method reconstructs the surface by interpolating a given point cloud. The idea is very simple: starting from a seed triangle, a ball with user-given radius moves across the surface of a point cloud. When it comes into contact with three points, a triangle is formed. The ball then pivots around the edge, still keeping contact to the edge points, until it finds a new point and forms a new triangle. When all reachable edges are found, a new seed triangle is located and the process continues until all points are considered. Whether or not a triangle is formed depends on the radius of the ball: If the ball is too large, it will touch more than three points simultaneously, and no triangle will be formed. On the other hand, if the ball is too small, it will touch less than three points and will "drop through the hole". In the case of missing data, holes larger than the radius of the ball may be present. However, the user can use multiple balls of different radii to handle uneven surfaces. If the data is noisy, with outliers and extra points, surface normals are used to determine if a point is accepted or not.

The Poisson surface reconstruction method, as the name suggests, approaches surface reconstruction as a spatial Poisson problem. The goal is to find a scalar function, called indicator function, that approximates a vector field defined by oriented points as closely as possible. Since the gradient of the indicator function forms a vector field that is equal to the inward surface normals for points near the surface and zero otherwise, both vector fields are closely linked. By introducing a divergence operator, this problem could be transformed into a standard Poisson problem of computing a scalar function whose Laplacian (divergence of gradient) is equal to the divergence of the original vector field and then reducing them from each other. Contrary to the ball pivoting method, all points are considered at the same time, which allows the Poisson reconstruction method to robustly approximate noisy data and create smooth surfaces.

4. Testing the pipeline

With the pipeline to process point clouds from underground mines described in Chapter 3, the next step is to test it with suitable data. In this chapter, both the available point clouds and the results of semantic segmentation and surface reconstruction are presented.

4.1 Available data

Four raw 3D point cloud files were provided by a mining company. Each of the point clouds depicts a part of the mining tunnel (Figure 4.1). The first point cloud (pc 1) represents a part of the tunnel that forks at one end (Table 4.1). The second one (pc 2) is from a straight tunnel. The third point cloud (pc 3) pictures a slightly curving tunnel, while the fourth point cloud (pc 4) is from a clearly curving tunnel. All tunnels have relatively smooth walls that are covered with concrete, with pc 1 also having some grid texture on the wall. There were no other obstructions in the tunnels except pipes and cables hanging from the ceilings.

The point clouds have been collected with an RTC360 laser scanner standing on a tripod that covers 360 degrees around the scanner. The scanning was done with medium accuracy, meaning that the distance between two points is around 6 millimeters 10 meters away. Each of the point clouds has around 10 million points, with all of the points also having an RBG color value. They were originally stored in .e57 format but converted to .ply format.

Table 4.1: The data

point cloud	n. of points	description
pc 1	9 118 161	forking tunnel
pc 2	9 655 212	straight tunnel
pc 3	9 767 733	slightly curving tunnel
pc 4	9 754 013	curving tunnel

(a) Top view of pc 1

(b) Inside view of pc 1

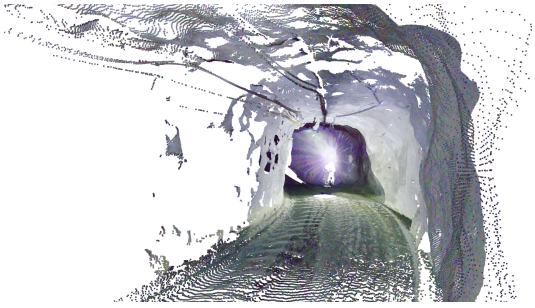
(c) Top view of pc 2

(d) Inside view of pc 2

(e) Top view of pc 3

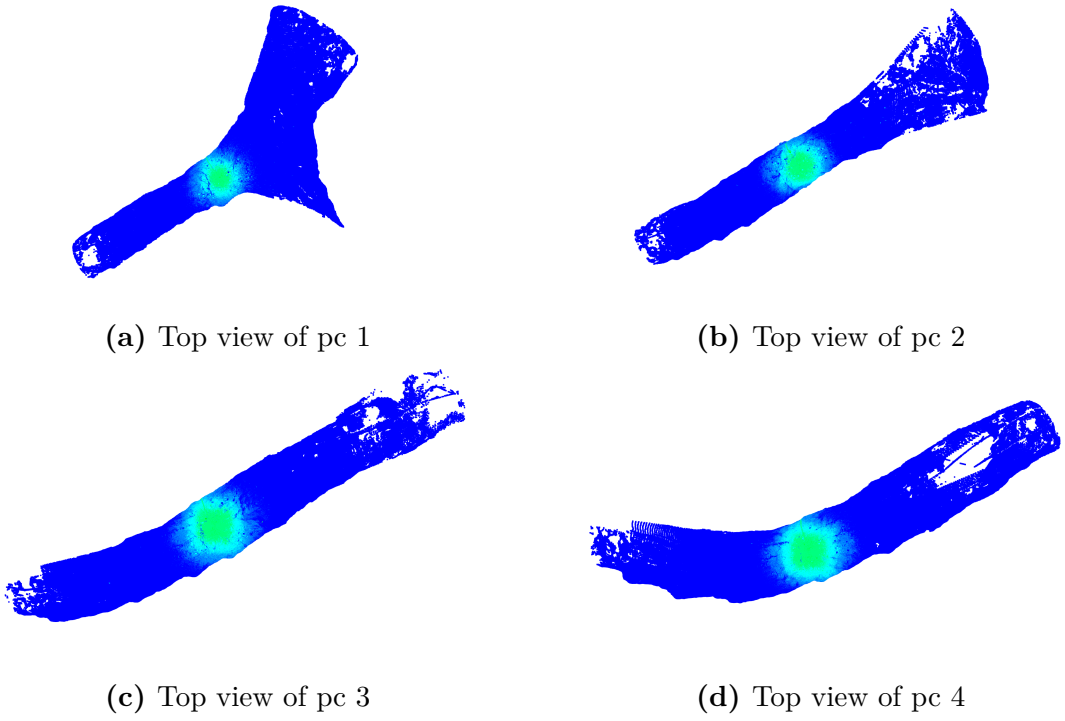
(f) Inside view of pc 3

(g) Top view of pc 4



(h) Inside view of pc 4

Figure 4.1: The four point clouds



(a) Top view of pc 1

(b) Top view of pc 2

(c) Top view of pc 3

(d) Top view of pc 4

Figure 4.2: The distance of each point to its nearest neighbor. Each quantile (color) has the same number of points.

4.2 Characteristics of the data

Since every one of the four point clouds was collected with one scan, the density of points is highest near the scanner and decreases further away (Figure 4.2). In addition, there are smaller and larger holes in the point clouds, where the scanner has not been able to collect data due to uneven wall surfaces. There are also very large holes in the shape of the person standing next to the scanner and one hole beneath the scanner. Because of changing densities and large holes, these point clouds make a very uneven surface. Reconstructing the missing data was not part of the methods used in this thesis.

It should also be noted that there was no ground-truth available to evaluate the results of semantic segmentation. There were no annotated versions of these point clouds, which limited the numerical estimations.

4.3 Reducing the number of points with voxelization

Since the provided point clouds were quite large, consisting of around 10 million points each, voxelization was used to reduce their size. Voxelization was performed using the

Open3D library [60]. From the different voxelization sizes tested, in the end sizes 0.01, 0.02 and 0.05 were selected. In all these cases, the details were preserved well and there was a noticeable difference only at voxel size 0.05 (Figure 4.3), although the number of points was significantly reduced at all voxelization levels (Table 4.2). In addition to reducing the number of points, voxelization also transformed the point clouds into equal-distance grids. The number of points in original point clouds was also slightly reduced when duplicate points were removed.

Table 4.2: The effect of voxelization on number of points

voxelization size	n. of points	% of original points
original	9 998 695	100
0.01	2 661 602	27
0.02	1 028 235	10
0.05	251 047	3

4.4 Clustering and semantic segmentation methods in feature extraction

The original point cloud and three downsampled versions of each of the four point clouds were used in feature extraction. Although the differences in the results between the different point clouds were small, the differences were larger between different levels of voxelization and the original point clouds.

4.4.1 Clustering

For the original point cloud (Figures 4.4a, b, c and d), DBSCAN could identify parts of a few objects (pipes, cables etc.) in the center of the point cloud, but most of the parts of the point cloud were identified as noise (black areas). This result was somewhat expected, since DBSCAN does not work well in the case of point clouds with uneven densities. Finding an optimal epsilon and minimum number of points was challenging, since the density of points decreased farther away from the center.

With voxelized point clouds (Figures 4.4e,f, g, h and Figure 4.5) smaller number of points were identified as noise. Although in this way more points could be preserved, it also meant that fewer points that were part of cables and pipes were classified as their own entities.

Overall, the number of clusters stayed very high, which seemed to be due to the large number of holes in the point clouds. Although the high number of clusters might

(a) original image

(b) original image

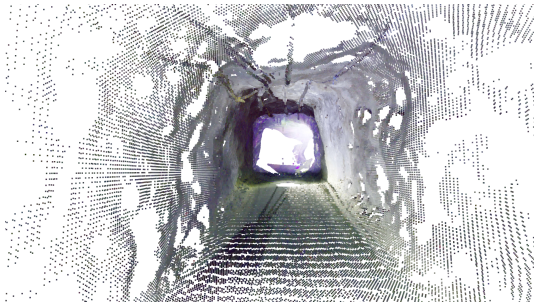
(c) voxelization with size 0.01

(d) voxelization with size 0.01

(e) voxelization with size 0.02

(f) voxelization with size 0.02

(g) voxelization with size 0.05



(h) voxelization with size 0.05

Figure 4.3: Different level of voxelization

not be a problem in itself, it might become a problem if those clusters do not form clear entities. As can be seen in Figures 4.5d and 4.5f, in the best case some parts of the pipes were classified as noise, while other parts were classified as part of larger clusters. In other words, man-made objects could not be separated from tunnels with this clustering method.

Table 4.3: The number of DBSCAN clusters

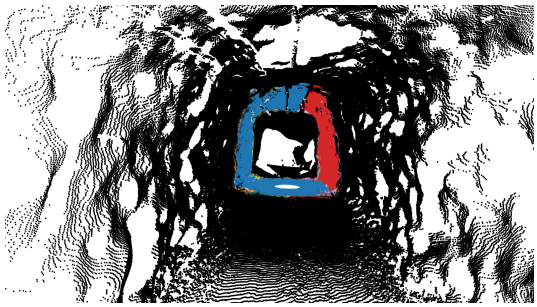
voxel size	epsilon	min. n. of points	n. of clusters
original	0.01	5	15814
	0.01	10	10660
	0.01	15	8670
0.01	5	5	2361
	10	5	259
	5	10	1342
	5	20	562
	5	50	429
0.02	5	5	315
	5	10	354
	5	20	285
	4	10	535
	3	10	1032
0.05	2	5	622
	2	10	581
	2	15	1048
	3	15	149

4.4.2 Semantic segmentation methods

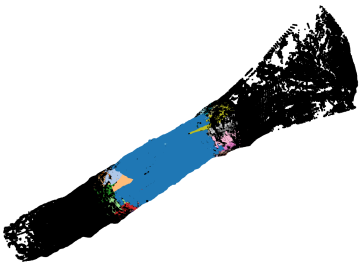
Two pretrained RandLA-Net models were tested. The first model used SemanticKITTI as training data, while the second model used Toronto 3D. The preset weights for both models were downloaded from https://github.com/isl-org/Open3D-ML/blob/main/model_zoo.md.

For all four point clouds, the original point cloud and three different voxelized point clouds with voxel sizes 0.01, 0.02 and 0.05 were classified with pretrained RandLA-Net model. In all cases, all points in the original point clouds were annotated in one class, "person" (Figure 4.6a), while points in total 12 voxelized point clouds were annotated in all cases as either "fence", "vegetation", "terrain" or "pole" (Figures 4.6c,

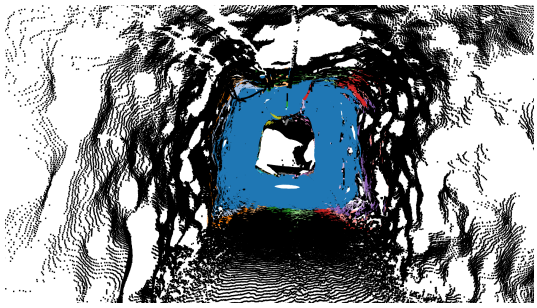
(a) Top view of original pc with epsilon 0.01 and minimum points 15



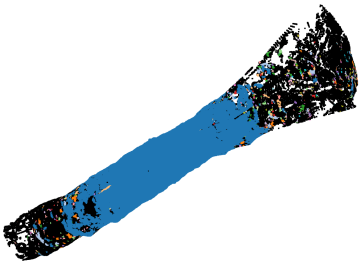
(b) Inside view of original pc with epsilon 0.01 and minimum points 15



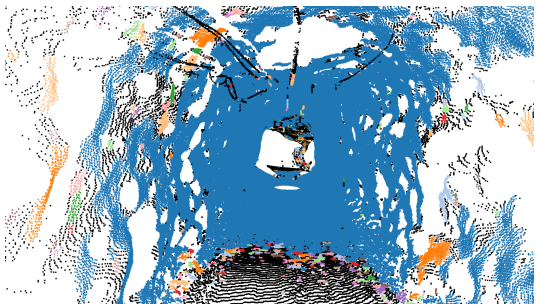
(c) Top view of original pc with epsilon 0.02 and minimum points 10



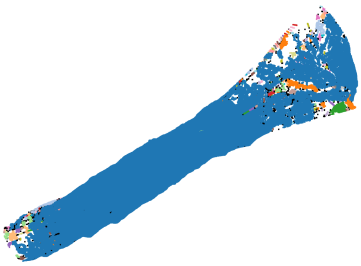
(d) Inside view of original pc with epsilon 0.02 and minimum points 10



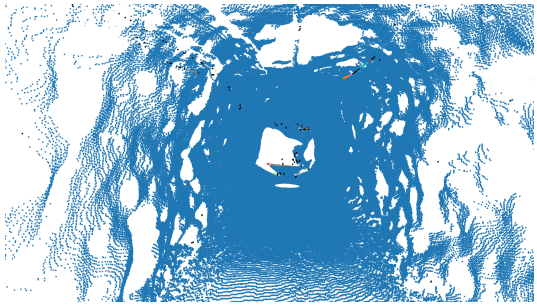
(e) Top view of pc with voxel size 0.01, epsilon 5 and minimum points 10



(f) Inside view of pc with voxel size 0.01, epsilon 5 and minimum points 10

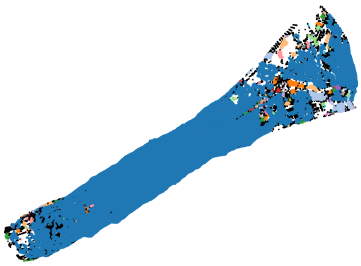


(g) Top view of pc with voxel size 0.01, epsilon 10 and minimum points 5

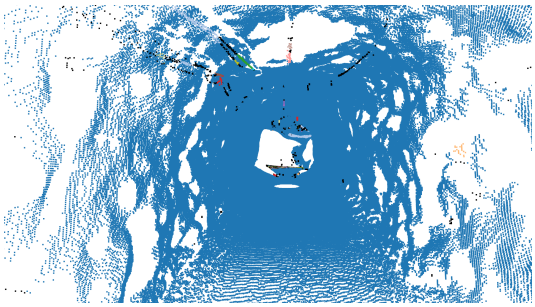


(h) Inside view of pc with voxel size 0.01, epsilon 10 and minimum points 5

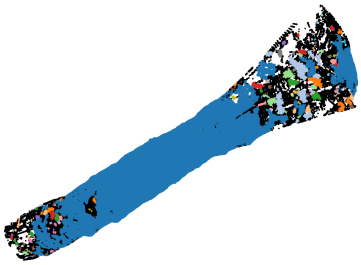
Figure 4.4: Clustering results for the original point cloud and the version with voxel size 0.01



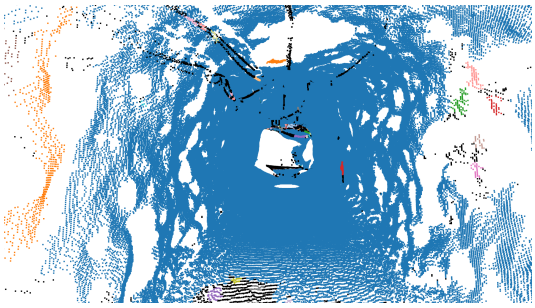
(a) Top view of pc with voxel size 0.02, epsilon 5 and minimum points 10



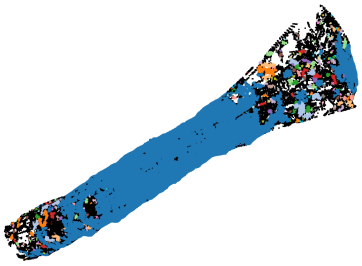
(b) Inside view of pc with voxel size 0.02, epsilon 5 and minimum points 10



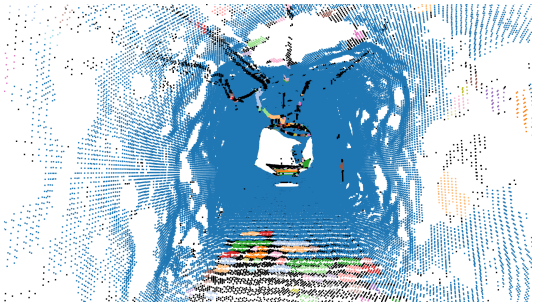
(c) Top view of pc with voxel size 0.02, epsilon 5 and minimum points 20



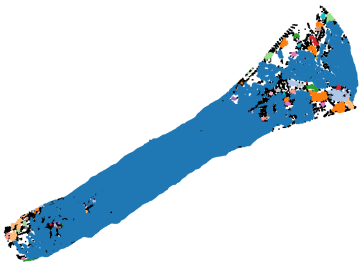
(d) Inside view of pc with voxel size 0.02, epsilon 5 and minimum points 20



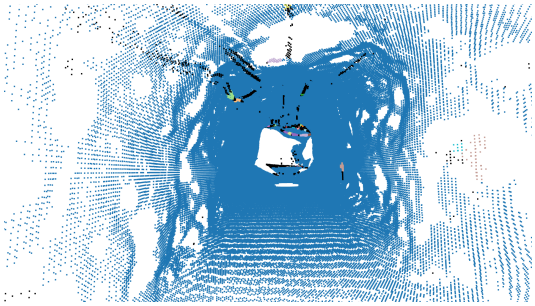
(e) Top view of pc with voxel size 0.05, epsilon 2 and minimum points 10



(f) Inside view of pc with voxel size 0.05, epsilon 2 and minimum points 10



(g) Top view of pc with voxel size 0.05, epsilon 3 and minimum points 15



(h) Top view of pc with voxel size 0.05, epsilon 3 and minimum points 15

Figure 4.5: Clustering results for the point clouds with voxel sizes 0.02 and 0.05

4.6e and 4.6g). Vegetation was the dominant class in all voxelized point clouds, while few parts of the floor and walls were classified as "pole". Only a few individual points were classified as either "fence" or "terrain". In none of the cases cables were classified separately from the ceilings.

When all 16 point clouds were classified with RandLA-Net model pretrained with Toronto 3D dataset, points were annotated as "unclassified", "road markings" and "natural", and in some cases also as "building" and/or "car" (Figure 4.7). In particular, the walls of the tunnels were mostly classified as "natural", while the ceilings were classified in most parts as "road markings", and the floors were partly classified as "unclassified". Only a few points were classified as "building" or "car". The model was unable to identify pipes and cables, but the ceilings, where most cables were located, were classified separately from other parts of the tunnel.

4.4.3 Comparison between different methods

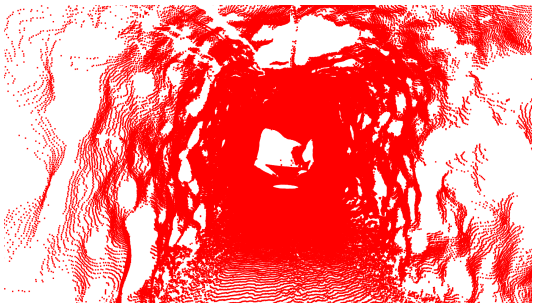
Both clustering and machine learning methods were able to extract some features from point clouds, although the results were very different. The clustering method only looks for clusters, while machine learning methods use semantic segmentation to actually categorize points in different classes. The results of clustering depend on the selected values of epsilon and minimum number of points, while the results of machine learning algorithms depend on both the selected model and the data that are used to train it.

In clustering, in addition to the selected values of epsilon and minimum number of points, voxelization played a large role in the results. Using the original point cloud with changing density resulted in much larger numbers of both clusters and number of points classified as noise. Although cables and pipes are indeed separate structures from tunnel walls, the changing density and numerous holes in the point clouds made it very difficult to separate these structures from other clusters. Voxelization helped negate the problem with density, but it could not remove the problem with holes. Therefore, clustering was not a reliable way to extract features from point clouds.

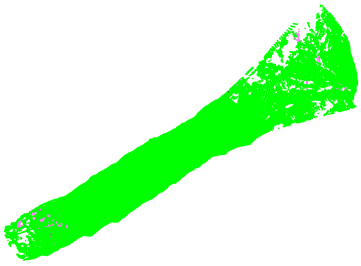
In semantic segmentation, different pretrained models gave very different results. Since both SemanticKITTI and Toronto 3D datasets are generated by lidar attached to a moving vehicle and mostly contain outdoor objects, their classes are not compatible with objects found in underground mining settings. However, some of the objects found outdoors have shapes similar to those found in mines. For example, cables and pipes form straight objects that can resemble poles or utility lines, both classes found in Toronto 3D set.

Although the model trained with SemanticKITTI failed to distinguish different

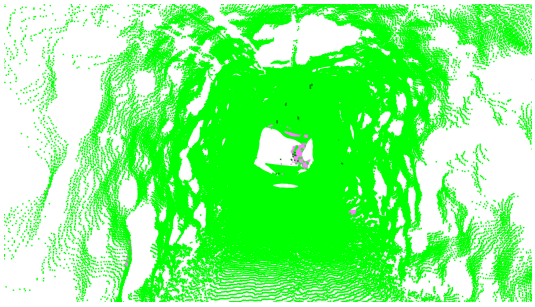
(a) Top view of pc 2



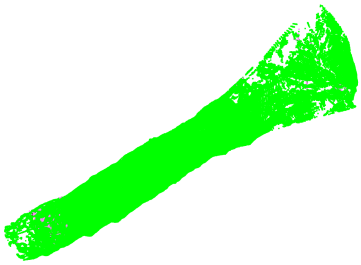
(b) Inside view of pc 2



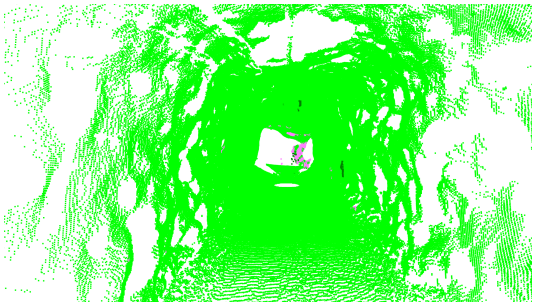
(c) Top view of pc 2 with voxelization 0.01



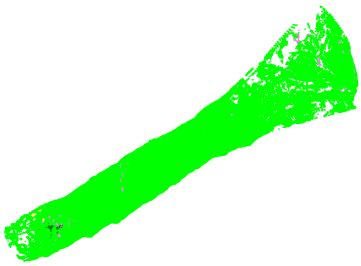
(d) Inside view of pc 2 with voxelization 0.01



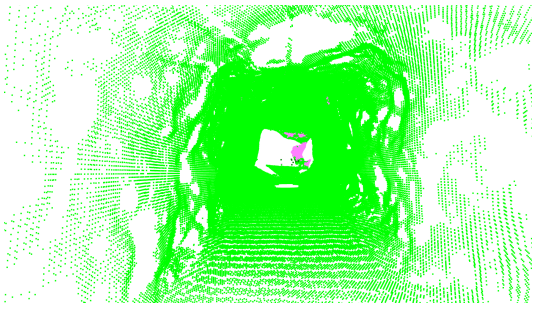
(e) Top view of pc 2 with voxelization 0.02



(f) Inside view of pc 2 with voxelization 0.02

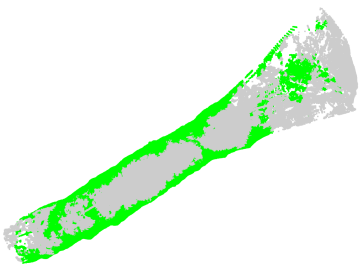


(g) Top view of pc 2 with voxelization 0.05

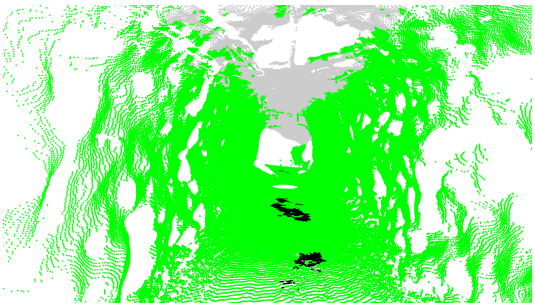


(h) Inside view of pc 2 with voxelization 0.05

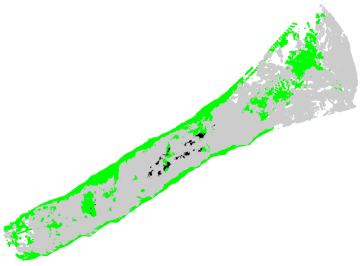
Figure 4.6: Model pretrained with SemanticKITTI dataset. Red = "person", bright green = "vegetation", light purple = "pole", yellow = "fence" and dark green = "terrain".



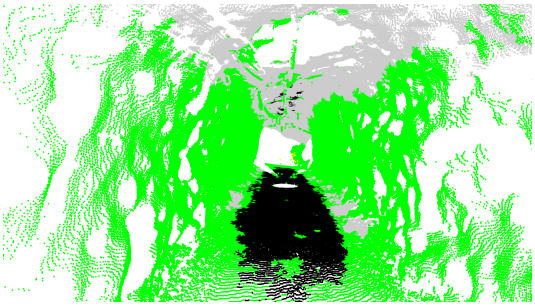
(a) Top view of pc 2



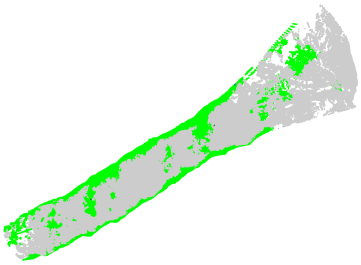
(b) Inside view of pc 2



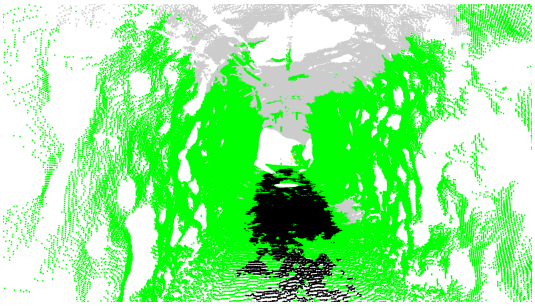
(c) Top view of pc 2 with voxelization 0.01



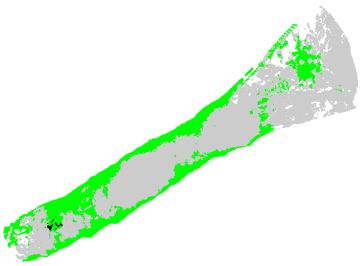
(d) Inside view of pc 2 with voxelization 0.01



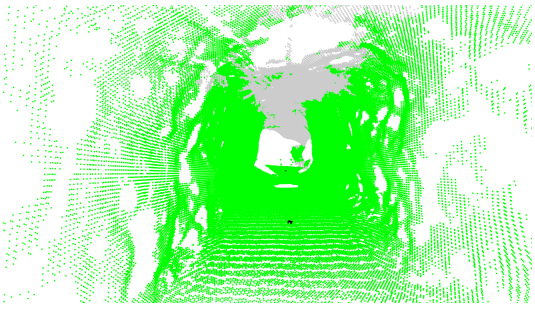
(e) Top view of pc 2 with voxelization 0.02



(f) Inside view of pc 2 with voxelization 0.02



(g) Top view of pc 2 with voxelization 0.05



(h) Inside view of pc 2 with voxelization 0.05

Figure 4.7: Model pretrained with Toronto3D dataset. Green = "natural", gray = "road markings", black = "unclassified", yellow = "building" and blue = "car".

shapes from point clouds, Toronto 3D did a bit better. The fact that most of the points were classified as "vegetation" or "natural" may be due to holes in the point clouds. While artificial objects (buildings, vehicles, etc.) form even and angular shapes, vegetation usually has a less distinguishable shape. It is also interesting that the ceiling of a tunnel was classified as "road marking" by a model trained with Toronto 3D. Road markings are basically straight lines on an even surface, and the same could be said for pipes in the ceiling.

4.5 Ball pivoting and Poisson methods in surface reconstruction

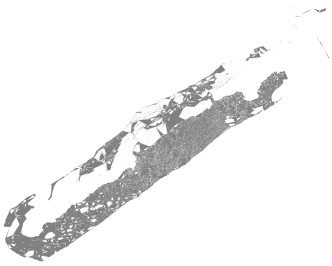
From the original four point clouds, versions with voxel sizes 0.01, 0.02 and 0.05, so in total 12 point clouds, were tested with surface reconstruction. Before actual surface reconstruction, extra objects were removed from the point clouds. Since all objects (cables and pipes) were located in the ceilings of the tunnels, removing those basically meant removing the whole ceiling. Based on the results of semantic segmentation (see Section 4.4.2), the areas labeled as "road markings" were removed. After that, the surface normals were computed and oriented for all resulting point clouds. Open3D library was used for computing surface normals and implementing both surface reconstruction methods.

4.5.1 Ball pivoting method

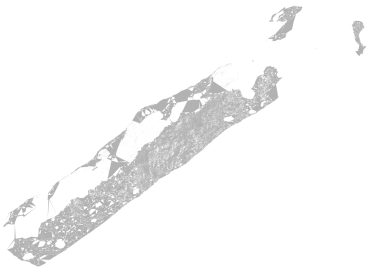
In the ball pivoting method, multiple balls of different sizes were used to interpolate the surface. The selected balls were 0.1, 0.5, 1, 3, 6, 10, 20 and 50 in size. Even with multiple balls, this method was unable to interpolate across all points, but both small and larger holes were left in the surface (Figure 4.8). Moreover, the result was quite jagged, which is more of a visual problem. Although some of the missing data was reconstructed with this method, in the case of larger holes, only part of the hole was reconstructed. The results were similar for all point clouds and in all voxelization levels. The degree of reconstructed ceiling area depended on the results of semantic segmentation and how much of the ceiling was removed.

4.5.2 Poisson surface reconstruction method

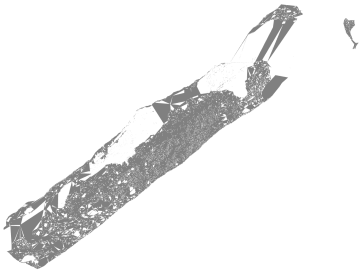
Poisson surface reconstruction was performed using depth value 11. This method created a smooth mesh that covered almost the entire tunnel area (Figure 4.9b). Although most of the parts of the surface with missing data were reconstructed, the mesh also



(a) Side view of the mesh generated from pc with voxel size 0.01



(b) Side view of the mesh generated from pc with voxel size 0.02



(c) Side view of the mesh generated from pc with voxel size 0.05

(d) Detail of the mesh generated from pc with voxel size 0.01

Figure 4.8: The results of the Ball pivoting surface reconstruction

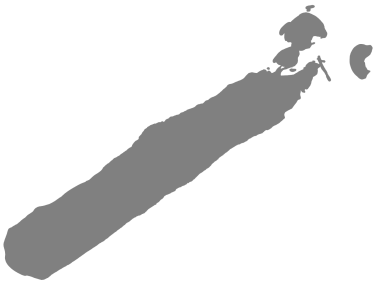
covered parts of the tunnel that should have been left open (such as the ends of the tunnel) and created some extra "bumps". These areas had a low density of points and are marked blue in Figure 4.9c. By removing these parts, the tunnel resembled more of the original point cloud (Figure 4.9d).

Figure 4.10 offers a better comparison between the original point clouds and generated meshes. The vertices of the meshes were transformed back into points to see how they fit the points from the point clouds used in mesh generation. The dark blue parts indicate the areas where the mesh has reconstructed some of the missing data. Depending on how large areas of the surface were removed, some larger holes were still left in the meshes.

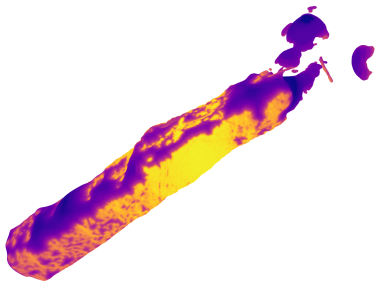
4.5.3 Comparison between different methods

The ball pivoting method was able to reconstruct the surface in general, but the result was jagged and inconsistent. Compared to the result of the Poisson method, the ball pivoting method left more holes behind. It is also notable that many of those holes were actually quite small, while the Poisson method only left larger holes. Since the performance of the ball pivoting method depends on the size of the balls used in interpolation, selecting the ball(s) of the right size should provide a clean result.

(a) Side view of pc 2 without ceiling



(b) Side view of pc 2 mesh before post-postprocessing



(c) Side view of pc 2 density mesh

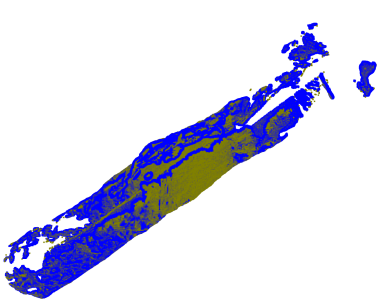
(d) Side view of pc 2 after post-processing

Figure 4.9: The process of generating a 3D mesh by Poisson surface reconstruction method.

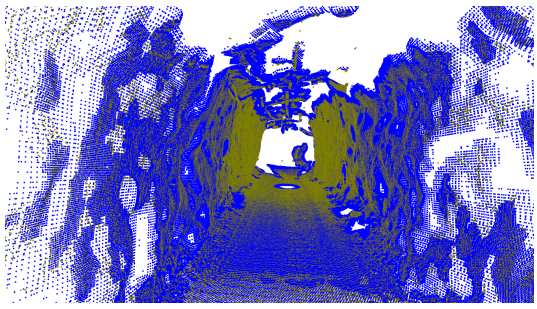
However, this method was implemented with multiple different balls, but the results did not really improve. All tested point clouds were voxelized into equal-distance grids, so changing densities should not have been a problem. Not taking into account the missing data, it is possible that the points created a too uneven surface for the method to be able to fully interpolate.

Unlike the ball pivoting method, the Poisson reconstruction method created a very smooth and even surface. Although it could reconstruct the areas of missing data, it also added extra surfaces in areas of low point density. Removing these extra surfaces required post-processing that also removed some of these reconstructed surfaces. The choice of how much to remove greatly affects the result. If a point cloud contains a lot of missing data and areas of low point density, it seems inevitable that not all missing data could be reconstructed. However, this is also a question of priorities. Sometimes extra surfaces are not a problem and post-processing might not be necessary.

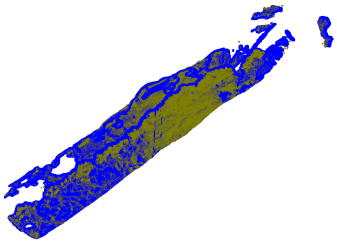
Of the two methods, the Poisson surface reconstruction method seems to work better on point clouds collected from underground tunnels, not just because of the missing data but also because of the uneven nature of the point clouds. Even though the Poisson method may require some post-processing, it provides a more consistent surface reconstruction. If the ball pivoting method is used, some post-processing, such as smoothing, could improve the result.



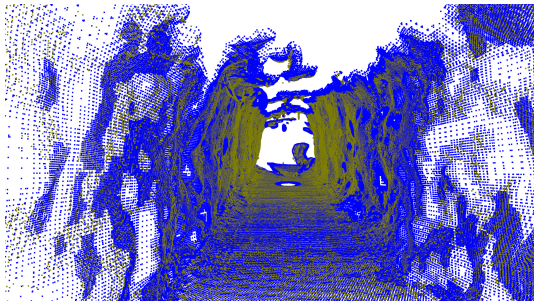
(a) Side view of pc with voxel size 0.01 and generated mesh



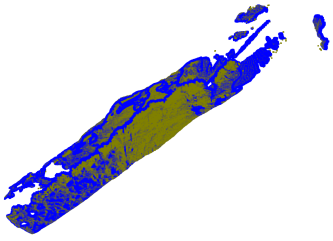
(b) Inside view of pc with voxel size 0.01 and generated mesh



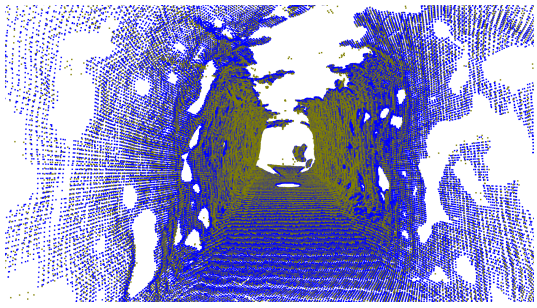
(c) Side view of pc with voxel size 0.02 and generated mesh



(d) Inside view of pc with voxel size 0.02 and generated mesh



(e) Side view of pc with voxel size 0.05 and generated mesh



(f) Inside view of pc with voxel size 0.05 and generated mesh

Figure 4.10: The comparison of point clouds and generated meshes. Yellow = point cloud, blue = generated mesh.

4.6 Summary of the results

Proximity-based clustering was used as a baseline for feature extraction, and on the basis of the results, it is safe to say that it does not yield good results with the kind of data used in this work. When DBSCAN was developed, one of the goals was to identify clusters of arbitrary shape and size [18]. Although this would usually be a strength, in this case it turned out to be a liability. When point clouds contain a lot of missing data, the algorithm can not reliably identify which parts in the point clouds are actual objects and what is noise.

The results of the semantic segmentation were mixed. Although it would have been theoretically possible that semantic segmentation could have been able to identify familiar shapes from the data, in practice it did not happen. It is possible that there was too much missing data for the models to accurately identify smaller entities, such as pipes and cables, or that the shapes of the point clouds diverged too much from the point clouds used in the datasets. However, this does not mean that the results were useless. Although the smaller entities escaped the classification, the model trained with Toronto 3D dataset was able to identify larger entities, such as floors, walls and ceilings. Therefore, the final answer to the first research question "Can pretrained models be a valid choice for semantic segmentation for point clouds collected from underground tunnels?" would therefore be yes, depending on the objective.

The results of the surface reconstruction were similarly mixed. The ball pivoting method could adequately reconstruct the existing surface, but the result was jagged and inconsistent, most probably due to the uneven nature of the point clouds. Even though all of the tested point clouds were voxelized into equal-distance grids, the undulating surface would still be a challenge for the balls. On the other hand, the Poisson method reconstructed the existing surface smoothly and consistently. In the case of missing data, the ball pivoting method was able to reconstruct some parts, while the Poisson method reconstructed most of the parts. To answer the second research question "What surface reconstruction methods work best for underground tunnels?", the answer would be the Poisson method.

5. Discussion

Based on the results from Chapter 4, both the pipeline used to process point clouds from underground mines and the data collected from underground mines had several strengths and weaknesses. These are discussed next.

5.1 Building the pipeline

3D point clouds can be processed and analyzed in multiple ways depending on the objective of the analysis. In this thesis, the goal was to produce a 3D mesh that is cleaned of extra features. While the steps of this pipeline (preprocessing, feature extraction and surface reconstruction) are common steps of point cloud processing that worked well in this case, the pipeline itself should not be understood as a general pipeline that works in all situations. Depending on the data, steps like smoothing, skeletonization or upsampling may be necessary for analyzing process and should therefore be included in the pipeline. There is no one pipeline that fits all situations or even most of the situations.

5.2 The effect of changing density and missing data

Collecting point cloud data in underground mining environment is a challenging task. Both environmental hazards and technical limitations affect the quality of the data collected. It is the rule rather than the exception that some data will be missing due to obscuring machines, people, and other objects. On the other hand, while some parts are completely missing point data, in some parts the point density is much higher than in others. Although some of these challenges could be mitigated with extensive preprocessing, it mostly comes down to carefully selecting and testing appropriate algorithms. The effect of changing density and missing data was especially obvious when clustering was tested as a feature extraction method. On the other hand, the RandLA-Net model trained with the Toronto 3D dataset was invariant to changing point densities because the Toronto 3D dataset contains point clouds with changing

densities.

5.3 Using existing datasets and pretrained models

Large datasets, with millions to even billions of annotated points are more and more popular nowadays. This has made machine learning model training easier than ever and saves a lot of time. At the same time, machine learning models trained with these datasets are commonly available. With the limited data available in this thesis, the use of both an existing dataset and a pretrained model formed a convenient possibility to skip the time-consuming training part and to test how these models would work in an underground mining environment, which is a totally different environment than where the dataset data was collected. Both SemanticKITTI and Toronto 3D datasets are collected using a moving vehicle and consist of street views and urban areas. While urban areas and underground mining environment may seem to be very far from each other, they have also similarities: both may include vehicles and persons, and while urban areas have poles, fences and other straight objects, underground mining environment has pipes and cables. Although pretrained models were unable to identify individual objects, the model trained with Toronto 3D was able to identify the area where most of the objects were located. This suggests that the use of pretrained models in underground mining environment could be a valid option even with datasets containing objects from different environments.

5.4 Reconstructing the surface

In this work, two different methods, Poisson and ball pivoting, were tested for surface reconstruction of underground tunnels. Although adequate results could be reached with both methods, there were differences in those results, especially how holes in the data were handled. Poisson method creates surface between all points, no matter how far they are from each other, while ball pivoting uses the size of the ball(s) to determine which points should be connected to each other. If there is a lot of missing data, careful selection of parameters becomes crucial. Although parts of missing data in walls and floor should be patched, it is not reasonable to patch the end of tunnels.

5.5 The lack of ground truth

The performance of semantic segmentation is usually evaluated with metrics like (mean) Intersection-over-Union (IoU) or Overall Accuracy (OA). These numbers show

how well a model can annotate points in correct classes. Currently, there are no large, readily available and annotated datasets for underground mines. Wang et al. [52] prepared an annotated dataset for their work, but at the time of writing, it is not freely available and consists of 2D images instead of 3D point clouds. Although building that kind of dataset was outside the scope of this work, hopefully in the future annotated point clouds containing other than outdoor street views or indoor scenes will become more common.

6. Conclusions and summary

This thesis covers the process of preprocessing, feature extraction and 3D mesh creation using data collected from underground mining environment. The properties, acquisition methods and segmentation methods of 3D point clouds are presented, as well as the use of 3D point clouds in underground mining environment. A pipeline to process point cloud data is explained, and together with it several relevant methods, algorithms, and datasets are highlighted. The possibilities of pretrained models in the context of semantic segmentation of underground tunnel point clouds are explored. Finally, two different surface reconstruction methods and their suitability for 3D mesh creation based on underground tunnel point clouds are compared.

6.1 Semantic segmentation and pretrained models

Two RandLA-Net models trained with large point cloud datasets containing urban outdoor scenes were used to classify four point clouds collected from underground tunnels. The ability of those models to correctly classify the points varied based on both the model and the level of voxelization. The first model was trained using the SemanticKITTI dataset, and when tested with raw point cloud data with uneven point density, the model was unable to extract any features, e.g. all points were classified in the same category. The model was then tested with the same point clouds preprocessed with three different levels of voxelization. Although the points were classified into four different categories, in reality, the model was unable to extract any actual features. Based on this work, a model trained with SemanticKITTI does not seem to be a viable way to classify points in point clouds with uneven point density and large areas of missing data.

The second model was trained using the Toronto 3D dataset. Both raw and voxelated point clouds provided similar results. Similarly to the first one, this model was also unable to extract actual features, but instead larger entities, basically the floor, the walls and the ceiling, were classified separately. Although this was not as good a result as was hoped, in this case separating the ceiling from the rest of the tunnel proved to be enough, since all the man-made objects were located in the ceiling.

Removing the whole ceiling thus removed extra objects from the point clouds.

In summary, while both models failed in semantic segmentation, pretrained models can still be useful outside their original context. It might not always be necessary to separately classify every object if a coarser classification will work as well. Therefore, more research on the use of pretrained models in different environments would be recommended.

6.2 Reconstructing the surface of a tunnel

Two different surface reconstruction methods were used to create 3D meshes based on four point clouds collected from underground tunnels. The first method used ball pivoting that interpolates the points into a 3D triangle mesh. It was tested with point clouds of three different levels of voxelization, and in all cases the result was somewhat jagged and inconsistent.

The second method used a Poisson algorithm to create a smooth 3D mesh. This method could handle different point densities and missing data well, but also created extra surfaces that had to be removed afterwards.

In summary, the two methods tested perform best in opposite situations. While the ball pivoting method struggles to create an even surface without holes, the Poisson surface reconstruction method creates an even surface without holes, even in those cases where all holes should not be removed. Therefore, the ball pivoting method works best with point clouds that have an even point density and no missing data, whereas the Poisson surface reconstruction method provides good results even with uneven point density and missing data, but needs some post-processing.

Bibliography

[1] R. Abbasi, A. K. Bashir, A. Rehman, and Y. Ge. 3D lidar point cloud segmentation for automated driving. *IEEE Intelligent Transportation Systems Magazine*, 17(1):8–29, 2025.

[2] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva. Point set surfaces. In *Proceedings Visualization, 2001. VIS'01.*, pages 21–29, 2001.

[3] N. Amenta, M. Bern, and M. Kamvysselis. A new voronoi-based surface reconstruction algorithm. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques (SIGGRAPH)*, pages 415–421, 1998.

[4] N. Amenta, S. Choi, and R. K. Kolluri. The power crust, unions of balls, and the medial axis transform. *Computational Geometry*, 19(2-3):127–153, 2001.

[5] C. L. Bajaj, F. Bernardini, and G. Xu. Automatic reconstruction of surfaces and scalar fields from 3D scans. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques (SIGGRAPH)*, pages 109–118, 1995.

[6] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall. SemanticKITTI: A dataset for semantic scene understanding of LiDAR sequences. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[7] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):349–359, 1999.

[8] J.-D. Boissonnat. Geometric structures for three dimensional shape representation. *ACM Transactions on Graphics (TOG)*, 3(4):266–286, 1984.

[9] A. Boulch, B. L. Saux, and N. Audebert. Unstructured point cloud semantic labeling using deep segmentation networks. In *Eurographics Workshop on 3D Object Retrieval*, pages 17–24, 2017.

[10] J. C. Carr, R. K. Beatson, J. B. Cherrie, T. J. Mitchell, W. R. Fright, B. C. McCallum, and T. R. Evans. Reconstruction and representation of 3D objects with radial basis functions. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques (SIGGRAPH)*, pages 67–76, 2001.

[11] S. Chen, S. Niu, T. Lan, and B. Liu. PCT: large-scale 3D point cloud representations via graph inception networks with applications to autonomous driving. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 4395–4399, 2019.

[12] H.-Y. Chiang, Y.-L. Lin, Y.-C. Liu, and W. H. Hsu. A unified point-based framework for 3D segmentation. In *2019 International Conference on 3D Vision (3DV)*, pages 155–163, 2019.

[13] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques (SIGGRAPH)*, pages 303–312, 1996.

[14] P. Dabek, J. Wodecki, P. Kuwaja, A. Wróblewski, A. Macek, and R. Zimroz. 3D point cloud regularization method for uniform mesh generation of mining excavations. *ISPRS Journal of Photogrammetry and Remote Sensing 218*, pages 324–343, 2024.

[15] A. Dai and M. Nießner. 3DMV: Joint 3D-multi-view prediction for 3D semantic scene segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 452–468, 2018.

[16] A. Dai, D. Ritchie, M. Bokeloh, S. Reed, J. Sturm, and M. Nießner. ScanComplete: Large-scale scene completion and semantic segmentation for 3D scans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[17] H. Edelsbrunner and E. P. Mücke. Three-dimensional alpha shapes. *ACM Transactions On Graphics (TOG)*, 3(1):43–72, 1994.

[18] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density based algorithm for discovering clusters in large spatial database with noise. *KDD-96 Proceedings*, pages 226–231, 1996.

[19] S. Grehl, M. Sastuba, M. Donner, M. Ferber, F. Schreiter, H. Mischo, and B. Jung. Towards virtualization of underground mines using mobile robots - from 3D scans to virtual mines. In *MPES 2015 - Smart Inoovation in Mining*, 2015.

[20] J. Guerry, A. Boulch, B. L. Saux, J. Moras, A. Plyer, and D. Filliat. Snapnet-r: Consistent 3D multi-view semantic labeling for robotics. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 669–678, 2017.

[21] Y. Guo, H. Wang, Q. Hu, H. Liu, L. Liu, and M. Bennamoun. Deep learning for 3D point clouds: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(12):4338–4364, 2021.

[22] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. Surface reconstruction from unorganized points. In *Proceedings of the 19th annual conference on computer graphics and interactive techniques (SIGGRAPH)*, pages 71–78, 1992.

[23] Q. Hu, B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham. RandLA-Net: efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11108–11117, 2020.

[24] Q. Huang, W. Wang, and U. Neumann. Recurrent slice networks for 3D segmentation of point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2626–2635, 2018.

[25] M. Jancosek and T. Pajdla. Multi-view reconstruction preserving weakly-supported surfaces. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3121–3128, 2011.

[26] M. Jaritz, J. Gu, and H. Su. Multi-view PointNet for 3D scene understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019.

[27] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *Eurographics Symposium on Geometry Processing*, 2006.

[28] M. Kazhdan and H. Hoppe. Screened Poisson surface reconstruction. *ACM Transactions on Graphics*, 32(3), 2013.

[29] R. Kolluri, J. R. Shewchuk, and J. F. O'Brien. Spectral surface reconstruction from noisy point clouds. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing (SPG)*, pages 11–21, 2004.

[30] A. Komarichev, Z. Zhong, and J. Hua. A-CNN: Annularly convolutional neural networks on point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7421–7430, 2019.

[31] T. Kot, P. Novak, and J. Babjak. Visualization of point clouds built from 3D scanning in coal mines. In *17th International Carpathian Control Conference (ICCC)*, pages 372–377, 2016.

[32] L. Landrieu and M. Simonovsky. Large-scale point cloud semantic segmentation with superpoint graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4558–4567, 2018.

[33] F. J. Lawin, M. Danelljan, P. Tosteberg, G. Bhat, F. S. Khan, and M. Felsberg. Deep projective 3D semantic segmentation. In *International Conference on Computer Analysis of Images and Patterns*, pages 95–107, 2017.

[34] T. Le and Y. Duan. PointGrid: A deep network for 3D shape understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9204–9214, 2018.

[35] J. Li, B. M. Chen, and G. H. Lee. SO-Net: Self-organizing network for point cloud analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9397–9406, 2018.

[36] Y. Li, R. Bu, M. Sun, W. Wu, X. Di, and B. Chen. PointCNN: Convolution on X-transformed points. In *32nd Conference on Neural Information Processing Systems (NeurIPS 2018)*, 2018.

[37] Z. Lv, W. Wang, K. Zhang, W. Li, J. Feng, and Z. Xu. A synchronous detection-segmentation method for oversized gangue on a coal preparation plant based on multi-task learning. *Minerals Engineering*, 187, 2022.

[38] D. Maturana and S. Scherer. VoxNet: A 3D convolutional neural network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 922–928, 2015.

[39] S. Muraki. Volumetric shape description of range data using "blobby model". In *Proceedings of the 18th annual conference on Computer graphics and interactive techniques (SIGGRAPH)*, pages 227–235, 1991.

[40] Y. Ohtake, A. Belyaev, M. Alexa, G. Turk, and H.-P. Seidel. Multi-level partition of unity implicits. In *Acm Siggraph 2005 Courses*, pages 173–es. Association for Computing Machinery, 2005.

[41] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Point Net: Deep learning on point sets for 3D classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 652–660, 2017.

[42] C. R. Qi, L. Yi, H. Su, and L. Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *31st Conference on Neural Information Processing Systems (NIPS 2017)*, 2017.

[43] D. Rethage, J. Wald, J. Sturm, N. Navab, and F. Tombari. Fully-convolutional point networks for large-scale point clouds. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 596–611, 2018.

[44] R. A. Rosu, P. Schütt, J. Quenzel, and S. Behnke. LatticeNet: Fast point cloud segmentation using permutohedral lattices. *arXiv:1912.05905*, 2019.

[45] C. Shen, J. F. O'Brien, and J. R. Shewchuk. Interpolating and approximating implicit surfaces from polygon soup. In *ACM SIGGRAPH 2004 Papers*, pages 896–904. Association for Computing Machinery, 2004.

[46] L. Si, Z. Wang, X. Xiong, and C. Tan. Coal-rock recognition method of fully-mechanized coal mining face based on improved U-net network model. *Journal of China Coal Society*, 46:578–589, 2021.

[47] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller. Multi-view convolutional neural networks for 3D shape recognition. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 945–953, 2015.

[48] W. Tan, N. Qin, L. Ma, Y. Li, J. Du, G. Cai, K. Yang, and J. Li. Toronto-3D: A large-scale mobile lidar dataset for semantic segmentation of urban roadways. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 202–203, 2020.

[49] L. Tchapmi, C. Choy, I. Armeni, J. Gwak, and S. Savarese. SEGCloud: Semantic segmentation of 3D point clouds. In *2017 International Conference on 3D Vision (3DV)*, pages 537–547, 2017.

[50] Z. Tong, W. Zhang, , and X. Zhang. Underground mine road detection using deep learning technique. *Applied Sciences*, 13(7), 2023.

[51] G. Turk and J. F. O'brien. Modelling with implicit surfaces that interpolate. *ACM Transactions on Graphics (TOG)*, 21(4):855–873, 2002.

[52] J. Wang, D. Li, Q. Long, Z. Zhao, X. Gao, J. Chen, and K. Yang. Real-time semantic segmentation for underground mine tunnel. *Engineering Applications of Artificial Intelligence*, 133, 2024.

[53] X. Wang, Y. Guo, S. Wang, G. Cheng, X. Wang, and L. He. Rapid detection of incomplete coal and gangue based on improved pspnet. *Measurement*, 201, 2022.

[54] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon. Dynamic graph CNN for learning on point clouds. *ACM Transactions on Graphics (TOG)*, 38(5):1–12, 2019.

[55] B. Wu, A. Wan, X. Yue, and K. Keutzer. SqueezeSeg: Convolutional neural nets with recurrent CRF for real-time road-object segmentation from 3D LiDAR point cloud. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1887–1893, 2018.

[56] Y. Xie, J. Tian, and X. X. Zhu. Linking points with labels in 3D: A review of point cloud semantic segmentation. *IEEE Geoscience and remote sensing magazine*, 8(4):38–59, 2020.

[57] H. Xiu, X. Liu, W. Wang, K.-S. Kim, and M. Matsuoka. MSECNet: Accurate and robust normal estimation for 3D point clouds by multi-scale edge conditioning. *arXiv:2308.02237*, 2023.

[58] Y. Xu, X. Tong, and U. Stilla. Voxel-based representation of 3D point clouds: Methods, applications, and its potential use in the construction industry. *Automation in Construction*, 126, 2021.

[59] C. Yi, D. Lu, Q. Xie, S. Liu, H. Li, M. Wei, and J. Wang. Hierarchical tunnel modeling from 3D raw LiDAR point cloud. *Computer-Aided Design*, 114:143–154, 2019.

[60] Q.-Y. Zhou, J. Park, and V. Koltun. Open3D: A modern library for 3D data processing. *arXiv:1801.09847*, 2018.