Chair of Computational Modeling and Simulation
Department of Civil, Geo and Environmental Engineering
Technical University of Munich

Efficient Vertical Object Detection in Large High-Quality Point Clouds of Construction Sites

Scientific work to obtain the degree

Master of Science (M.Sc.)

at the Department of Civil, Geo and Environmental Engineering of the Technical University of Munich.

Supervised by Prof. Dr.-Ing. André Borrmann
Alexander Braun M.Sc.
Dipl.-Math.(FH) Heiko Bauer
Florian Noichl M.Sc.
Chair of Computational Modeling and Simulation

Submitted by Miguel Arturo Vega Torres [redacted]

Submitted on 15. October 2020

Abstract

Nowadays, even when adherence to project schedule and budget are the most critical performance metrics among project owners, still 53% and 66% of typical construction projects exhibit schedule delays and cost overruns, respectively. Intending to contribute to a more efficient construction progress monitoring, this thesis proposes a method to detect some of the most common temporary objects classes in a laser scanner point cloud of a construction site. These objects can provide a precise estimate of the current construction progress. For this purpose, this thesis focuses on the detection of cranes, scaffolds, and formwork.

The method involves computer vision and machine learning techniques to detect vertical instances of the selected object classes. The proposed workflow begins with the automatic downsampling and rotation of the point cloud. Subsequently, the target objects are detected using a combination of several techniques: image processing over vertical projections, finding patterns in 3D detected contours and performing checks over specifically generated vertical cross-sections. A deep learning algorithm was leveraged to classify these cross-sections for the purpose of formwork detection.

The method was applied on three real point clouds of construction sites to assess its accuracy. The results reveal that the method achieves average rates above 88% for precision and recall. Moreover, the technique also achieved outstanding computational time performance. This demonstrates the capability of the method to support the automatic segmentation of point clouds of construction sites. Further development can be done to increase the precision and automation of the technique.

Zusammenfassung

Heutzutage weisen 53% der typischen Bauprojekte Verzögerungen und sogar 66% Budgetüberschreitungen auf, obwohl die Einhaltung des Projektplans und des Budgets die wichtigsten Leistungsmetriken für die Bauherren sind. Mit dem Ziel, zu einer effizienteren überwachung des Baufortschritts beizutragen, wird in dieser Arbeit eine Methode vorgeschlagen, mit der einige der häufigsten temporären Objekttypen in einer Laserscanner-Punktwolke einer Baustelle erkannt werden können. Die Erkennung dieser Objekte ermöglicht eine genaue Abschätzung des Baufortschritts. Hierfür konzentriert sich diese Arbeit auf die Erkennung von Kränen, Gerüsten und Schalungselementen.
Das Verfahren umfasst Methoden aus Computer Vision und Machine Learning, um vertikale Instanzen der ausgewählten Objektklassen zu erkennen. Der vorgeschlagene Workflow beginnt mit einem automatischen Downsampling und der Rotation der Punktwolke. Im Anschluss werden die Zielobjekte mithilfe einer Kombination verschiedener Techniken erfasst. Sie umfassen unter anderem die Bildverarbeitung über vertikale Projektionen, Auffinden von Mustern in den in 3D erkannten Konturen sowie überprüfungen von gezielt hierfür erzeugten vertikalen Querschnitten. Zur Klassifizierung der Querschnitte zum Zweck der Schalungserkennung wurde ein Deep Learning Algorithmus genutzt.
Um die Genauigkeit der Methode bewerten zu können, wurde diese auf drei realen Punktwolken von Baustellen angewendet. Die Ergebnisse zeigen, dass die Methode Durchschnittsraten über 88% für Präzision und Rückruf erreicht. Darüber hinaus erzielt das Verfahren eine hervorragende Rechenzeitleistung. Dies zeigt die Fähigkeit des Verfahrens, die automatische Segmentierung von Punktwolken von Baustellen zu unterstützen. Die Methode kann weiterentwickelt werden, um die Präzision und Automatisierung zu erhöhen.

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms

DL	Deep Learning
DNN	Deep Neural Network
FLTK	Fast Light Tool Kit
GUI	Graphical User Interface
LSSHOT	Linear Straight Signature Histogram of Orientations
MEP	Mechanical Electrical and Plumbing
ML	Machine Learning
MLS	Mobile Laser Scanner
MVS	Multi-View Stereo
NFoV	normal field of view
PCA	Principal Component Analysis
PCL	Point Cloud Library
PRG	Precedence Relationship Graph
ROIs	Regions of Interest
SfM	Structure-from-Motion
SLAM	Simultaneous localization and mapping
TLS	Terrestrial Laser Scanner
UAV	Unmanned Aerial Vehicle
VG	Voxel Grid

Chapter 1

Introduction

1.1 Motivation

Nowadays, inefficiencies, such as cost and time overruns, are still a regular occurrence in the construction industry. According to Mace and Jones (2016) 53% and 66% of typical construction projects record schedule delays and cost overruns, respectively.

The Klynveld Peat Marwick Goerdeler International Cooperative (KPMG), in its Global Construction Survey, (Armstrong & Gilge, 2017) revealed that adherence to the project schedule is not only the most essential performance measure in construction industry contracts but also the central issue in the execution of the projects.

After a careful analysis of the most recent research, reports and internal anecdotal observations from more than 100 construction projects over the past 10 years Lin and Golparvar-Fard (2020), identified the key issues that contribute the most to the lack of productivity in the construction industry. Among these issues, the main three are 1) discrepancy among owners, contractors and subcontractors regarding how projects are progressing; 2) the absence of systematic reporting of performance to project teams causes unresolved issues to pile up quickly; and 3) missed linkage to physical progress - the individuals concerned in project planning or revising short and long-term plans are commonly not present on construction sites.

One of the main root-causes of these issues is the fact that in the construction industry, the monitoring process is still mostly performed manually. This practice is expensive, labour-intensive and not comprehensive. On large construction sites, the call for exhaustive and accurate monitoring techniques rapidly increases since the area becomes too large to be manually monitored.

Many approaches have emerged to address this problem (please see Chapter 3 for more detailed information), perhaps one of the most promising approaches is the one implemented by Braun et al., introduced in 2016 and most newly developed in 2020, in which, among other steps, a 4D building information model is compared with a point cloud of a construction site, allowing the tracking of progress. This is possible because in such a model, all construction elements, besides having 3D geometry, are linked with time information, enabling a report of the planned state of construction at any given time.

One of the preeminent challenges with this approach is the presence of temporary construction elements in the point cloud. These temporary elements are usually not present in the building information model, and even worse, may occlude large portions of the permanent structures in the point cloud. This makes a reliable comparison with the 3D geometry of

the model more challenging. These temporary elements are mainly: scaffolds, formwork, cranes, reinforcement and machinery.

In an effort to overcome this challenge, the goal of this thesis is the detection of cranes, scaffolds, and formwork in laser scanner point clouds of construction sites. Besides the fact that these objects are prevalent on a construction site, detecting them is useful for the following reasons:

Cranes

Knowing how many operational cranes there are along with their exact position and height, not only gives a rough idea about the state of the construction progress but also enables the verification of compliance with safety regulations, like the distance from the crane to the building or to other cranes. Furthermore, this information could be useful to make retroactive improvements regarding the type of crane and the location where a crane should be deployed, which is a critical factor that directly influences the productivity on a construction site (Simons, 2016; Stromberger, 2012).

Scaffold

Detecting scaffolding components is not only useful to track the progress of the construction site but also to perform precise safety regulations checks regarding the minimum requirements that scaffold should fulfil, this could be done implementing corroborated methods such as those introduced by Wang (2019). This last step is crucial because, as Wang identified, falling from scaffolds is one of the leading causes of fatal accidents on construction sites.

Formwork

Formwork is always needed when the concrete in-situ construction technique is used, which in concrete construction is currently the most common construction method. Identifying the location of the formwork gives crucial information about the exact current state of construction progress. A placed formwork does not exclusively represent a building element that is currently under construction, it also indirectly gives key information about other completed tasks in the construction site. For example, it can give information about the previous construction of a concrete slab or the completion of previous tasks to build a foundation, such as excavating footings and compacting the soil on which the formwork is placed. After the detection of formwork elements and thanks to the low ranging error of laser scanner point clouds, the quality of the construction can also be evaluated. Beyond the correct position of the formwork element itself, the precise location of all elements present in formwork drawings, like openings and important elements, can automatically be verified. Moreover, an automated dimensional quality assessment can also be performed, in which the compliance with the structural plans can be ensured, before pouring concrete;

this can be done using validated methods applied to laser scanner point clouds such as those proposed by M.-K. Kim et al. (2020).

In addition to the previously mentioned reasons, the detection of the selected object classes could also assist the management of these elements in the construction site.

1.2 Research objectives

The main goal of this thesis is the development and testing of an algorithm capable of performing automatic detection of vertical instances of the selected three object classes (see Section 1.1) in high-quality point clouds of construction sites.

The research aims to answer the following question:

How is an automatic recognition of the selected three object classes in high-quality point clouds of construction sites possible?

Additionally, the following research sub-questions will be answered as well:

Object segmentation

- Is it possible to implement a safe and efficient prefiltering of the tentative permanent structures of a building to improve computational speed?
- To what extent can the verticality, horizontality, and parallelism of the objects on a construction site be exploited for the recognition of the three target object classes?

Validation

- How good is the proposed method compared to existing methods?
- To what extent can the proposed method assist the monitoring of the construction process?

Further research

- What are the bigger challenges in object recognition at different stages of the construction?
- Is it possible to achieve perfect recognition? What is needed to accomplish it?
- What semantic improvements in the point cloud are necessary to obtain greater automation in the scope of construction progress monitoring?

1.3 Research scope

The following points define the limits of this thesis.

- The research focus is the detection of the selected three specific object classes. Other classes, for example, walls, floors, stairs, ceiling, and columns, might be partially automatically isolated in a preprocessing step. However, their accurate segmentation lies outside the scope of this research.

- The method will only seek vertical instances of the selected object classes, slanted or horizontally positioned exemplars will not be detected. In this specific case, this seems to be a reasonable condition taking into account that on a construction site, objects of the selected classes are almost always in a vertical position, at least when they are being used.

1.4 Reading guide

This thesis is structured in the following chapters:

- Chapter 2 introduces the theoretical background needed to understand essential concepts in this thesis. It contains the theory of Octree, KD-Tree, as well as the typical geometry of the selected target objects. This information gives fundamental support that will be useful for the successful detection of the objects in a point cloud.

- Chapter 3 gives an overview of current approaches of construction progress monitoring and point cloud segmentation. The limitations of these methods are discussed.

- Chapter 4 explains the proposed approach in this research. It illustrates the workflow of the implemented vertical object detection method.

- Chapter 5 reports results, analysis and validations of the proposed method. On top of that, computational performance analyses are presented.

- In Chapter 6 all the research questions are answered, the limitations of the proposed workflow are explained and discussed, and finally the main contributions of this thesis and ideas for future work are presented.

Chapter 2

Theoretical Basics

2.1 Space-partitioning data structures

This section introduces the theory behind the two data structures used in this thesis: Octree and Kd-trees. An additional explanation of why these data structures are used is also provided.

2.1.1 Octree

An octree structure is perhaps the simplest way to partition a point cloud. Like pixels in an image, voxels are basic rasterized units of the three-dimensional space, which represent a location on a regular cubic grid.

A point cloud can be converted into a voxel-grid, and the object geometries can be approximated with a given spatial resolution, defined as the voxel size, sometimes also called leaf size.

An octree-structure divides the space recursively into eight equal subspaces until the given minimum voxel size is reached. Finally, each sub-space (or voxel) can be found along the octree. A general representation of the voxel-structure is shown in Figure 2.1.

In this thesis, an octree structure is used in the downsampling process, in which the number of points is reduced, and a minimum distance between two points or, equivalently, the maximum number of points in a particular volume is ensured. While the Point Cloud Library (PCL) (Rusu & Cousins, 2011) allows the straightforward implementation of a Voxel Grid (VG) filter to downsample a point cloud, if the point cloud is too large and the leaf size is too small, this method will throw an exception. This is because there might be insufficient memory to record all voxels. Therefore, one first partition in an octree structure is necessary. Once the point cloud is organized into an octree with a voxel size of 5 meters, the voxel grid method with 5 millimetres leaf-size is applied in every voxel of the octree. This method not only allows a uniform and fast downsampling of large point clouds but also preserves the colour information of the original point cloud.

Since the VG method approximates the point cloud with the centroid in every voxel, it might not accurately represent the underlying surface in cases where there is a lot of noise in the data or the leaf size is large and the objects present curved surfaces. Nonetheless, considering the low amount of noise in terrestrial laser scanner (TLS) point clouds and the fact that this thesis focuses on detecting objects that are larger than 3 centimetres (considering the diameter of a scaffold upright as the smallest dimension) a leaf-size of

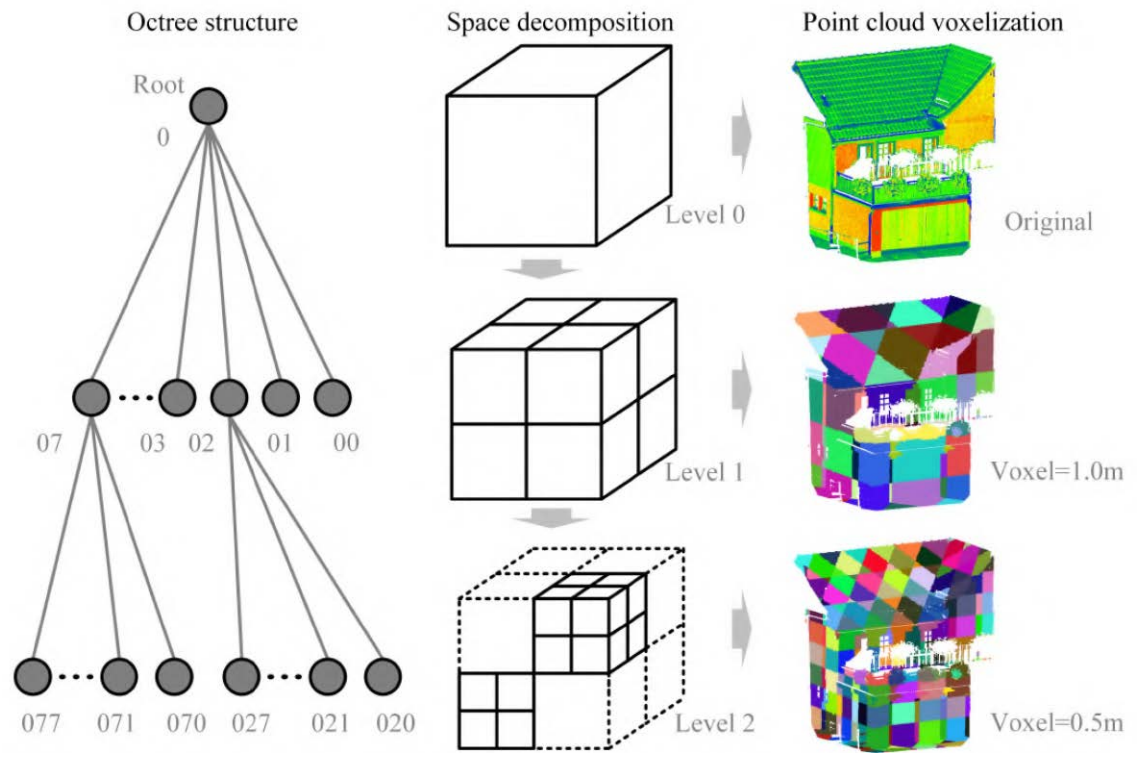


Figure 2.1: Representation of the octree-structure (Xu, 2019).

5 millimetres still preserves all the necessary features to properly represent the original geometry of the target objects.

The comprehensive review of the different algorithms to filter point clouds elaborated by X.-F. Han et al. (2017), showed that whereas the VG method is not the best to remove the noise, it is the most computationally efficient while performing a downsample task.

The VG method also plays a critical role in this thesis since it will ensure a minimum distance between every two points. This characteristic will be exploited to isolate vertical objects of a certain minimum height in the point cloud (see Chapter 4). This would not be possible if the downsampling method redistributes the points leaving non-uniform densities, as in the case of the optimum dataset method introduced by Blaszczak-Bak (2016). This method tries to preserve more points where there are complex surfaces and less where there are uncomplicated areas. According to Suchocki and Błaszczak-Bąk (2019) the optimum dataset method presents a suitable solution for the reduction of datasets from diagnostic measurements of building objects using TLS, since it preserves more points in sections of the point cloud where cracks, crevices or cavities are present. Similarly, Eickeler and Borrmann (2018) introduce a method to reduce the size of large point clouds, preserving geometrical features.

In cases where a lot of noise is present, like in point clouds captured with RGB-D cameras, other methods like Growing Neural Gas (Orts-Escolano et al., 2015), Edge Aware Resample or L0 minimization (X.-F. Han et al., 2017) yield better results than the VG method. Moreover, methods based on Graph Laplacian Regularization (J. Zeng et al., 2020) or Deep

Learning (Rakotosaona et al., 2019) represent novel alternatives to remove outliers and reduce noise in unordered point clouds.

2.1.2 KD-Tree

KD-Tree, first introduced by Bentley (1975), is a data structure devised for arranging points in a K-Dimensional space. The idea is that a binary tree is used to partition the dataset with samples sorted to the leaves of the tree. This offers a fast kNN query time of $O(k \log(n))$, rather than $O(n)$ as in the Grid case (Vermeulen et al., 2017).

The Kd-Tree always splits the data along hyperplanes (lines in the 2D case) that are perpendicular to the axes. The partition is typically at the median value for the selected feature. In a 2D case, the algorithm starts finding a middle point along the horizontal axis and then divides the space into two halves perpendicular to this axis at this middle point. For each subdivision, the central point along the vertical axis is found. Subsequently, the subdivisions are divided into halves at that middle point with a line perpendicular to the vertical axis. The steps above are repeated until the partition space can not be further subdivided.

Figure 2.2 is an example of the subdivisions of a 2D space. The corresponding binary tree is on the right side of the plot.

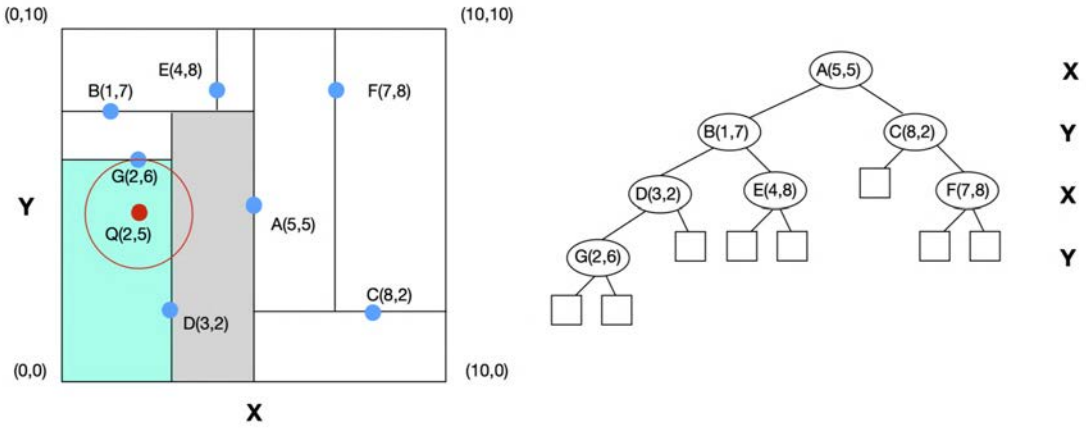


Figure 2.2: The Kd-Tree partitioning of the 2D feature space shown on the left corresponds to the tree on the right (Cunningham & Delany, 2020).

To query the nearest neighbours of Q with a Kd-Tree, the algorithm will locate it in the cyan region (hypercube in higher dimensions) represented by G in the binary tree. While the nearest neighbour could also be in the grey area, the distance between G and Q already gives an upper bound on the distance to this neighbour, as shown with the red circle around Q. Except for the grey region, the rest of the tree can be excluded from consideration. This capability to exclude large parts of the data is the reason why it yields the $O(k \log(n))$ performance (Cunningham & Delany, 2020).

In this thesis, the usage of an optimized header-only library for building Kd-Tree named nanoflann (Blanco & Rai, 2014) is considered the most appropriate, taking into account that Vermeulen et al. (2017) demonstrated that the application of this library for kNN search, outperforms other state-of-the-art methods.

2.2 Geometry of the target objects

This section summarizes necessary specifications about the usual geometry of the target objects, which are of crucial importance to detect these objects in the point cloud. Additional justification for the selection of certain types of target objects is also given.

2.2.1 Cranes

Some of the most common types of cranes in the construction industry are the crawler crane, self-erecting crane, telescopic crane, and tower crane (Simons, 2016). A summary of the advantages and disadvantages of every type is given in Norman-Spencer (2012). This thesis focuses on tower cranes because they are the most commonly used in the construction of tall buildings (Böttcher and Neuenhagen, 1997, p. 58).

The main components of a tower crane are the base, mast, slewing unit, operating cabin, jib, and counter-jib. The mast rests on the base of the crane and gives the crane its designed height. On top of the mast, the slewing unit with gears, motor, and rotation system is connected. At the front of the slewing unit, the jib is attached and at the back the counter-jib with weights. Figure 2.3 shows the main components of a tower crane.

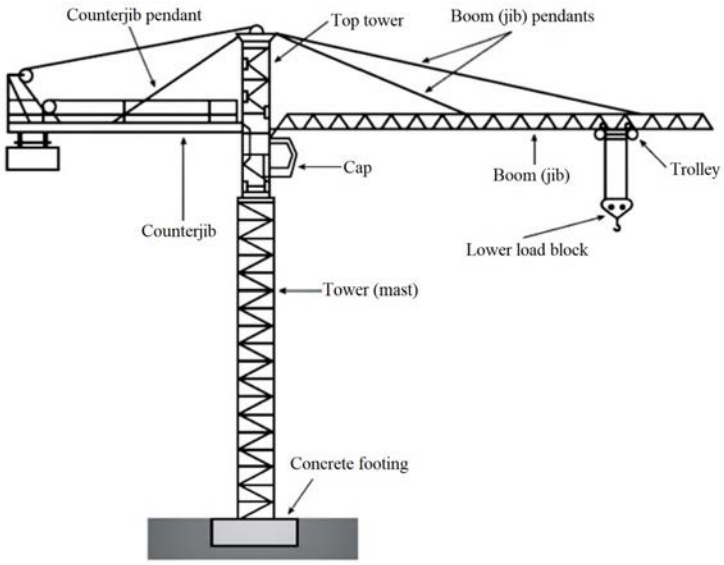


Figure 2.3: Tower crane components.

To detect the cranes in a point cloud, the geometry of the mast will play a significant role. Therefore it is explained in more detail in the following section.

2.2.1.1 Mast geometry

The mast is generally made of individual steel trussed sections that are connected together. The number of sections will determine the overall height of the crane.

While a section of a mast is always a square, its breadth can vary between 1.2 m to 2.5 m depending on the crane's type (see Figure 2.4).

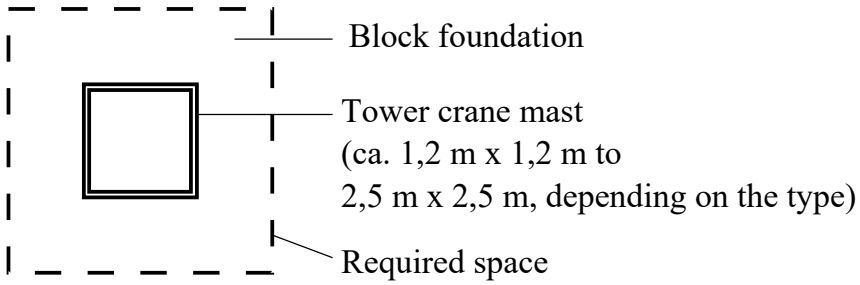


Figure 2.4: Top view of a tower crane mast with dimensions (Schach and Otto, 2017, p. 28).

The geometry of every section can differ depending on the type of crane and the manufacturer. There are even cases when a single crane is composed of different section types; some of the most common are shown in Figure 2.5.

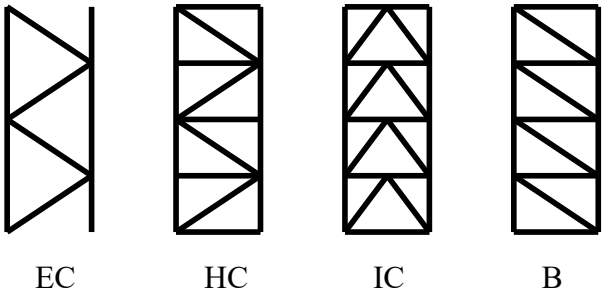


Figure 2.5: Different types of tower crane mast.

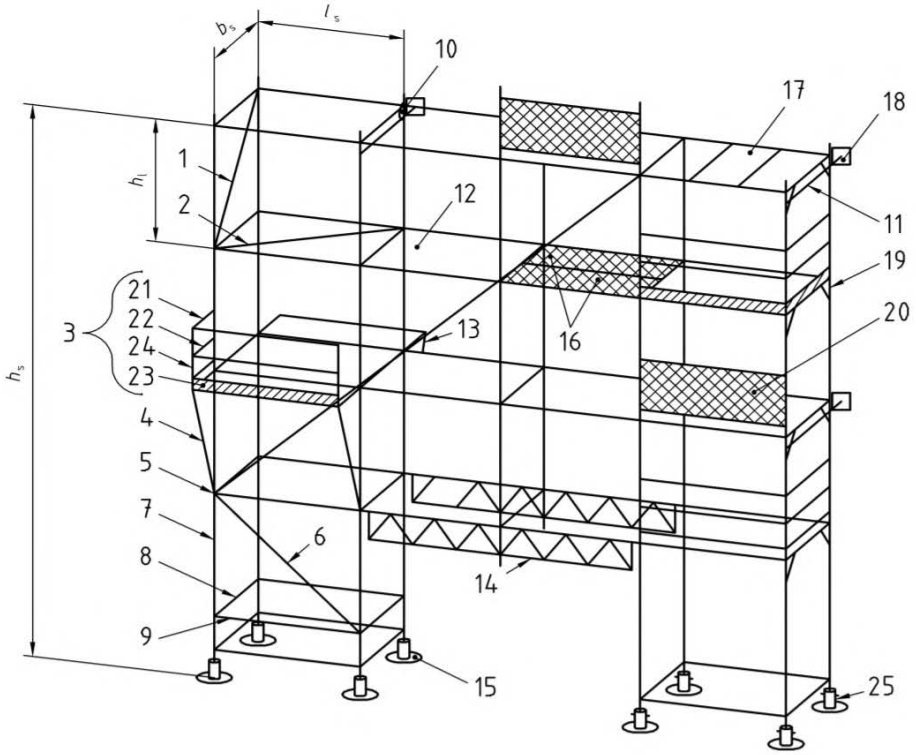
The exact geometry of every section can vary drastically depending on the manufacturer, the type of crane, and its size. There are also unique section designs, mostly for the base mast section. For example, for cranes that can climb inside a building; they need a particular configuration of space to be able to install a hydraulic mechanism inside them.

For the purpose of this thesis and to keep the method reliable and straightforward, only cranes with a horizontal line in the mast will be detected, i.e. cranes with a mast section type EC (according to Figure 2.5) will not be recognized. Moreover, to allow the detection of self-erecting cranes that usually have a smaller mast width in comparison with tower cranes, a minimum mast width of 1 m instead of 1.2 m will be used.

2.2.2 Scaffold

Opposite to sections of a tower cranes mast, scaffold elements consist of different smaller pieces that are usually manually assembled on the construction site. These are mainly: uprights (or standards), guard-rails, toe-boards, and work platforms. Additionally, there are also special sections of the scaffold system with diagonal braces, stairs, or additional accessories that enable the scaffold to adapt to different needs, for example, bridges or

extensions to make the scaffold wider. This thesis will focus on detecting faced scaffold elements. The main parts of a faced scaffold are shown in Figure 2.6.



Key

h_s	Scaffold height	11	Tie member (3.23)
b_s	Scaffold bay width, centre to centre of standards	12	Platform (3.15)
l_s	Scaffold bay length, centre to centre of standards	13	Bracket (-)
h_l	Scaffold lift height	14	Bridging ledger (-)
		15	Base plate (3.3)
1	Bracing in vertical plane ((transverse diagonal) (3.6)	16	Platform unit (3.16)
		17	Horizontal frame (-)
2	Bracing in horizontal plane (3.5)	18	Anchorage (3.1)
3	Side protection (3.19)	19	Vertical frame (-)
4	Bracket brace (-)	20	Fencing structure (5.5.5)
5	Node (3.13)	21	Principal guardrail (5.5.2)
6	Bracing in vertical plane (longitudinal diagonal) (3.6)	22	Intermediate guardrail (5.5.3)
		23	Toeboard (5.5.4)
7	Standard (3.21)	24	Post (-)
8	Transom (3.24)	25	Base jack (3.2)
9	Ledger (3.10)		
10	Coupler(3.8)		

Figure 2.6: Faced scaffold components in accordance with EN 12811-1.

Depending on the manufacturer, the exact geometry of a scaffold can vary, but some minimum distances are established by standardized norms. Following DIN EN 12 811-1, the minimum scaffold bay width is 0.6 m, and while there could be scaffold bay width of more than 2.4 m, in this thesis only scaffold with a maximum width of 1.2 m will be considered. This consideration is based on the fact that cost-effective scaffold systems are mainly made in the width classes W06 and W09 (Schach and Otto, 2017, p. 240), which have a width between the selected range (0.6 m to 1.20 m) in accordance with the table 1 of DIN EN 12 811-1. Similarly, the scaffold bay length could vary between 1.5 m to 3 m in line with DIN 4420-4. Figure 2.7 present the main components of a scaffold, together with its standardized minimum and maximum dimensions.

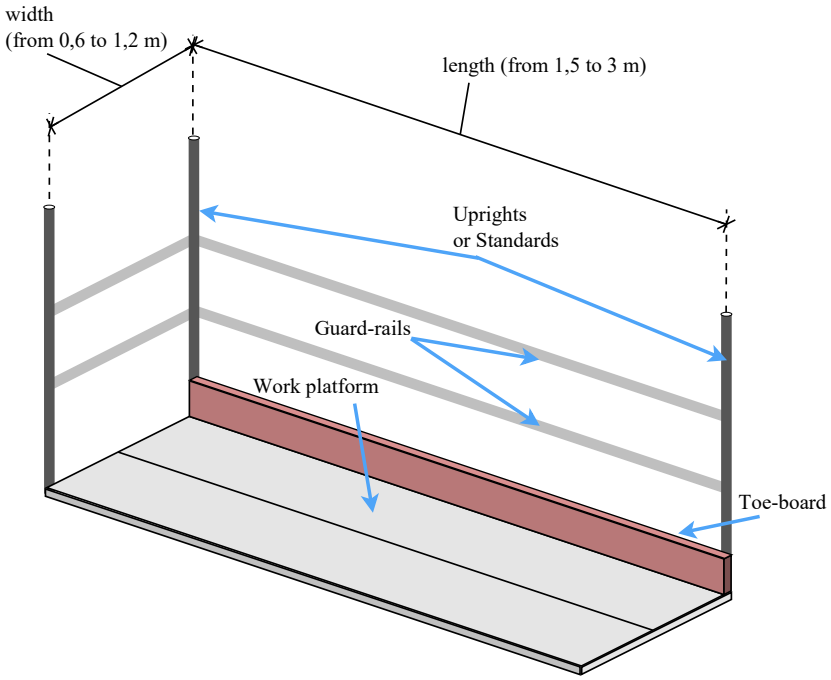


Figure 2.7: Scaffold main components and dimension ranges.

2.2.3 Formwork

Among the many types of formwork, the most common are wall, column, and slab formwork. Similar to scaffold elements, there could be specialized types of formwork, and they could also have additional accessories, for example, a working platform. This thesis will concentrate on wall formwork. Figure 2.8 shows a render of wall formwork elements as they would be placed on a construction site.

Whereas the exact geometry of a formwork element depends on the manufacturer, the basic idea of vertical and horizontal wailing beams in front of an interior wall panel always remains constant. Since the task is to detect whether there is a formwork or not and not its exact position or type, it is considered sufficient to work with the styles that a single manufacturer provides. In this case, wall formwork elements of the manufacturer PERI will be used. These are shown in Figure 2.9.

Figure 2.8: PERI "DOMINO" Wall formwork render (PERI, 2018).

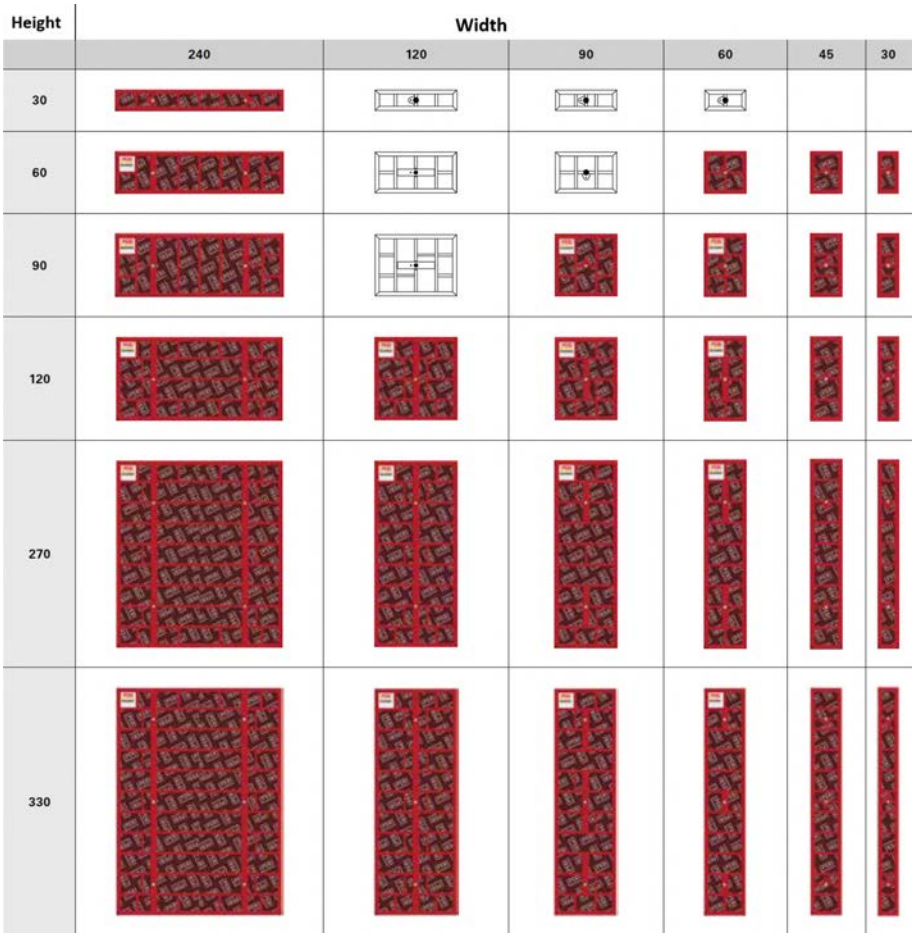


Figure 2.9: PERI Wall formwork dimensions (PERI, 2018).

Chapter 3

Related Work

3.1 Automated construction monitoring

In the past few years, there has been a lot of research on the topic of construction progress monitoring. Based on previous research, there are now two main approaches to capture the as-built status of a construction site. While one is purely based on 2D images, the other is based on point clouds. The basic idea is to compare photos as well as the point clouds against a 4D building information model to monitor the progress on site. This is possible because, as explained in the motivation (see Section 1.1), such a model contains not only the 3D geometric information of the building elements but also the time information when they should be built.

The main advantage of 2D images is that they can be captured with low-cost cameras and can be used to quickly document the status of the construction site. Therefore, there have been many investigations in this area, e.g., Kropp et al. (2018) compares registered 2D videos with a 4D building information model to perform indoor progress estimation and delay prediction in construction sites. The method can identify, for instance, whether the drywall is already installed or whether it is already painted or not, as well as the presence of particular building elements, such as radiators. This method is powerful but is highly dependent on the accuracy of the image-to-model registration process. Since their images do not contain depth information, a reliable comparison between the captured image and the 4D building information model is very challenging.

If there are sufficient images captured from different points of view, a 3D point cloud reconstruction is possible using Structure-from-Motion (SfM) methods (Wu, 2013). Once a point cloud (as-built) is generated, it can be compared against the geometry of the 4D building Information Model (as-planned) to track the progress of construction.

3.1.1 Image-based point clouds

Golparvar-Fard et al. (2011, 2015) uses a SfM process to create photogrammetric point clouds from unstructured images. For the comparison of as-planned and as-built geometry, the scene is discretised into a voxel grid. Subsequently, probabilistic and supervised machine learning approaches were used to define the construction progress, also enabling to take into account occlusions.

Braun et al. introduced a method in which instead of using a voxel grid, the deviations between a point cloud and the building model are measured directly and verify with a

scoring function (2016, 2019). More recently, Braun et al. (2020) presented a robust method to overcome the challenge of temporary elements present in a point cloud of a construction site. This challenge is illustrated in Figure 3.1, in which the presence of transient objects in the point cloud does not allow a straightforward geometric comparison with the building information model. To detect the presence of these temporary elements, they first compare the 4D building information model with a photogrammetric point cloud. Then, a Precedence Relationship Graph (PRG) allows them to make assumptions concerning regions that are not visible due to occlusions, and hence not directly detectable with the first comparison. Subsequently, to detect visible objects that were not recognised in the geometric comparison with the point cloud, three strategies where developed: The first is a distance threshold, to consider the possibility that formwork elements with a thickness of around 0.2 m are present in the point cloud; the second is colour-based object detection, which basically exploits the fact that formwork elements have different colour distribution in contrast with concrete elements; the third and final strategy is based on deep learning techniques; here a convolutional neural network was trained to detect formwork, scaffolding, columns, and walls. To summarise, the introduced methods originate a much more robust object detection process in comparison with a purely geometry-based approach, achieving higher detection accuracy in a range between 80% to 90%.

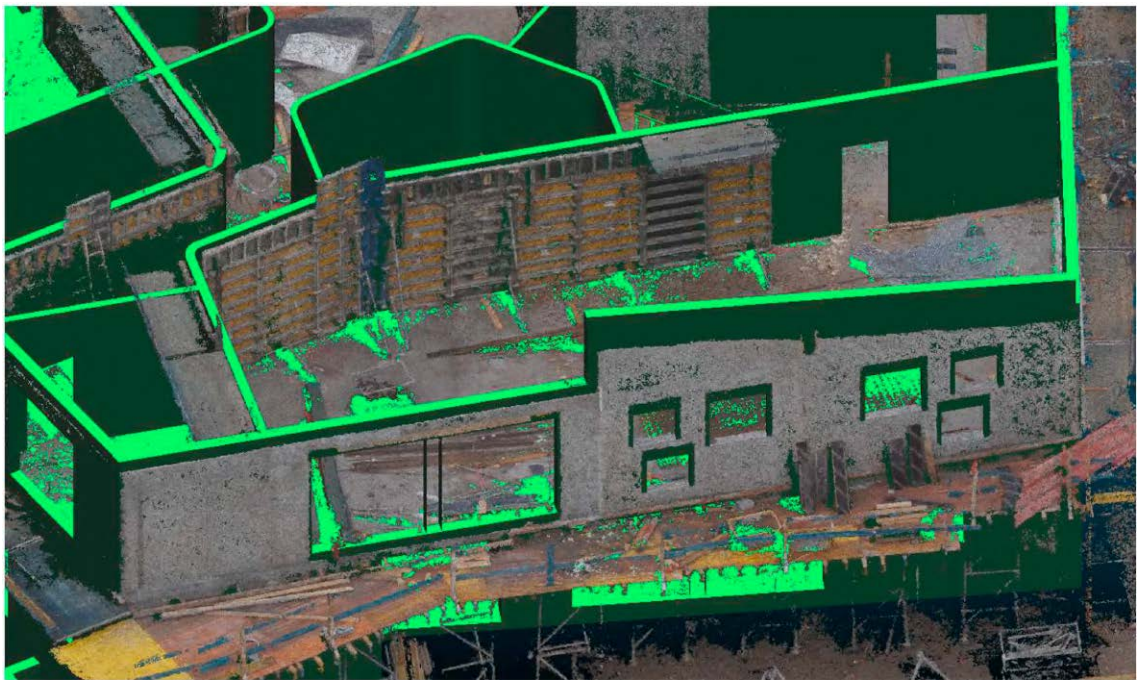


Figure 3.1: Comparison of a photogrammetric point cloud with a building information model. Temporary structures, such as formwork, that are not present in the model hinder the possibility of making a simple 3D geometric as-built vs. as-planned comparison (Braun et al., 2020).

For indoor mapping, Amer and Golparvar-Fard (2018) proposed a method based on SfM, Multi-View Stereo (MVS), and Simultaneous localization and mapping (SLAM) algorithms in which walkthrough videos are used to locally reconstructed workspaces in 3D without using markers. This allows project teams to capture images of every workspace at a

high frequency (e.g., daily) and continuously produce dense 3D models, enabling to track continuous changes in indoor construction scenes.

Laser scanning represents another alternative to generate point clouds. Whereas image-based methods provide multiple viewpoints and therefore fewer occlusions, laser scanners offer more accurate and denser point clouds of the objects of interest, which allows a more precise and reliable geometric comparison.

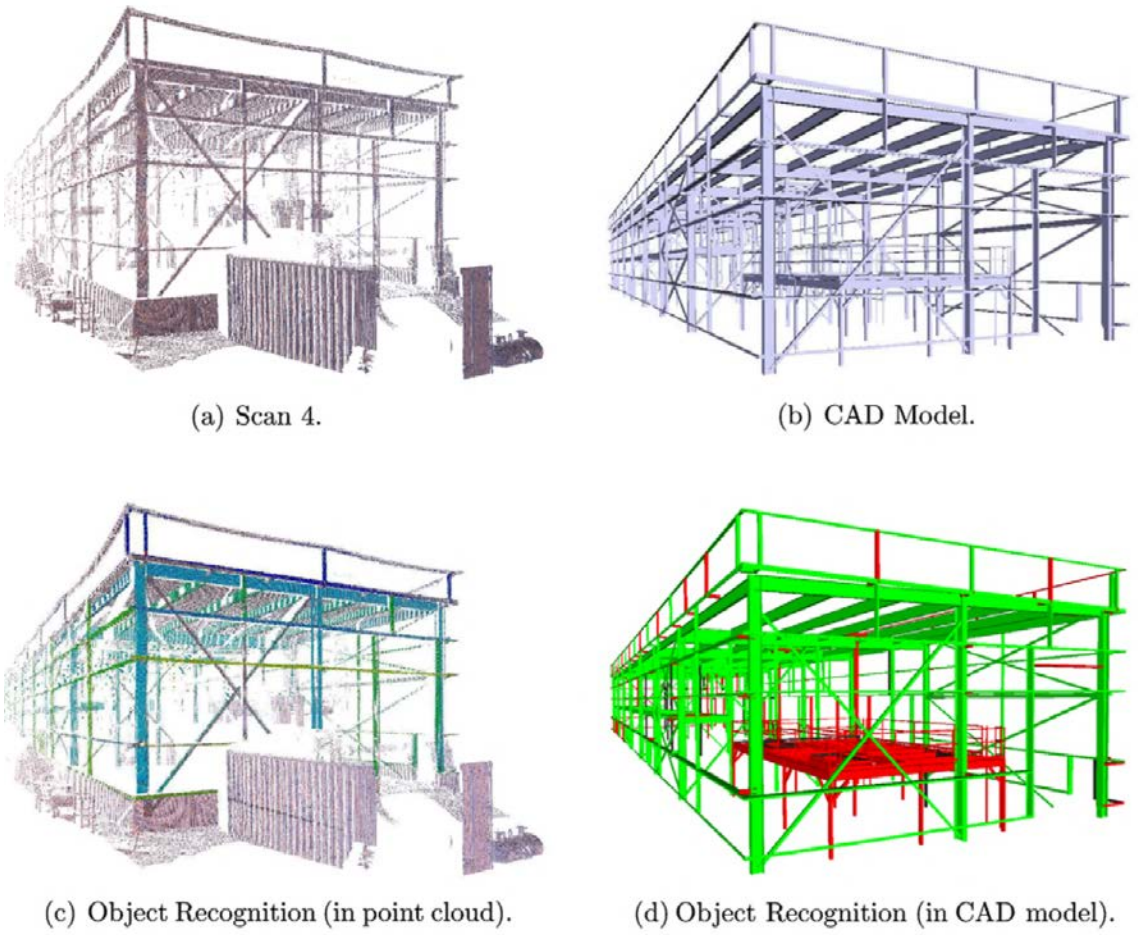
3.1.2 Laser scanner point clouds

Bosché and Haas (2008) presented one of the first approaches concerning construction progress tracking with laser scanner point clouds. After manually referencing the 3D CAD model in the laser scanner's coordinate frame, the method transforms the model into a point cloud. Then, it compares this generated point cloud with the laser scanner point cloud with a *range point matching metric*, which is used to define a proximity threshold between the as-planned and the as-built points. Later Bosché (2010, 2012) proposed another approach for large scale as-planned vs as-built comparison, in which the generated point clouds are co-registered with the model using a semi-automated coarse registration approach and a complementary ICP-based refined registration algorithm. Results of this approach are shown in Figure 3.2. Turkan et al. (2012, 2013) further developed this system for progress estimation using a 4D building information model and the earned value analysis. Then Turkan et al. (2014) use distance thresholds and a *negative space volume technique* to detect temporary and secondary objects in a point cloud of a construction site after the comparison with a 3D/4D model.

C. Kim et al. (2013) describe an additional process to review construction status and determine if the identified component states are mutually consistent and subsequently to modify inaccurate reports caused by incomplete 3D point cloud data set. Based on this work, Son et al. (2017) proposed a method to automatically update the schedule through the use of project-management software given a point cloud and a 4D building information model.

Bosché et al. (2015) propose a Scan-vs-BIM object detection workflow to monitor the built status of Mechanical Electrical and Plumbing (MEP) works, computing the "percentage built as planned" metric. They integrated the Hough transform to detect circular cross-sections and a comparison with a 4D BIM to be able to handle out-of-plane deviations and pipe occlusions. While the method worked well, it was developed to detect only objects with a known cylindrical geometric shape.

More recently, K. Han et al. (2018) introduce a colour-based detection method to determinate the state of the construction of elements in a point cloud of a construction site. After a geometrical comparison of a point cloud with a 4D building information model, the average value of the colour of the points that belong to the BIM element is compared with colour values from a large dataset of patches of different materials in a construction site. This



(a) Scan 4.

(b) CAD Model.

(c) Object Recognition (in point cloud).

(d) Object Recognition (in CAD model).

Figure 3.2: Performance of the approach proposed by Bosché (2010) for automated recognition of 3D CAD model objects in large construction site laser scans: (a) Laser scanner point cloud; (b) the 3D CAD model after registration with laser scanner point cloud; (c) object recognition results displayed in the point cloud. Each point cloud recognised as corresponding to a CAD model object is displayed with a unique colour. Grey points are those that have not been matched to any of the CAD objects; (d) object recognition results displayed in the CAD model, where objects coloured in green are those recognised in the point cloud.

allows them to determine the material of every building element. The underlying dataset is constituted of various materials such as asphalt, cement, formwork, etc.

While most of the previously presented methods heavily rely on the assumption of the existence of a 4D building information model with accurate geometric and time information to be able to detect building elements, the 3D object detection problem should still be solvable without a building information model. Without this assumption, this problem becomes more challenging.

3.2 Point cloud segmentation

This Section presents an overview of studies that attempt to segment the point cloud without having a building information model.

Whereas there has not been much research on the detection of exactly the selected object classes using point clouds, there has been plenty of research on the reconstruction of a building information model from point clouds (Fichtner, 2016; Maalek et al., 2019; Macher et al., 2017). For example, in a very recent study, Nikoohemat et al. (2020) propose a workflow for indoor 3D reconstruction from point clouds taken from a Mobile Laser Scanner (MLS), in which the permanent structures (such as walls, floors and ceilings, and stairs) and clutter are differentiated based on a surface region growing algorithm, a posterior adjacency graph, and heuristic rules. An overview of the method is presented in Figure 3.3. While this approach allows the identification of slanted walls, ceilings, and ramps, it comes with the computational cost of needing to perform a surface growing algorithm on the entire point cloud. Their method also utilise the trajectory of the MLS to find the different building levels, this makes part of their approach not suitable for point clouds of multi-storey buildings captured with a TLS, due to the evident absence of this trajectory.

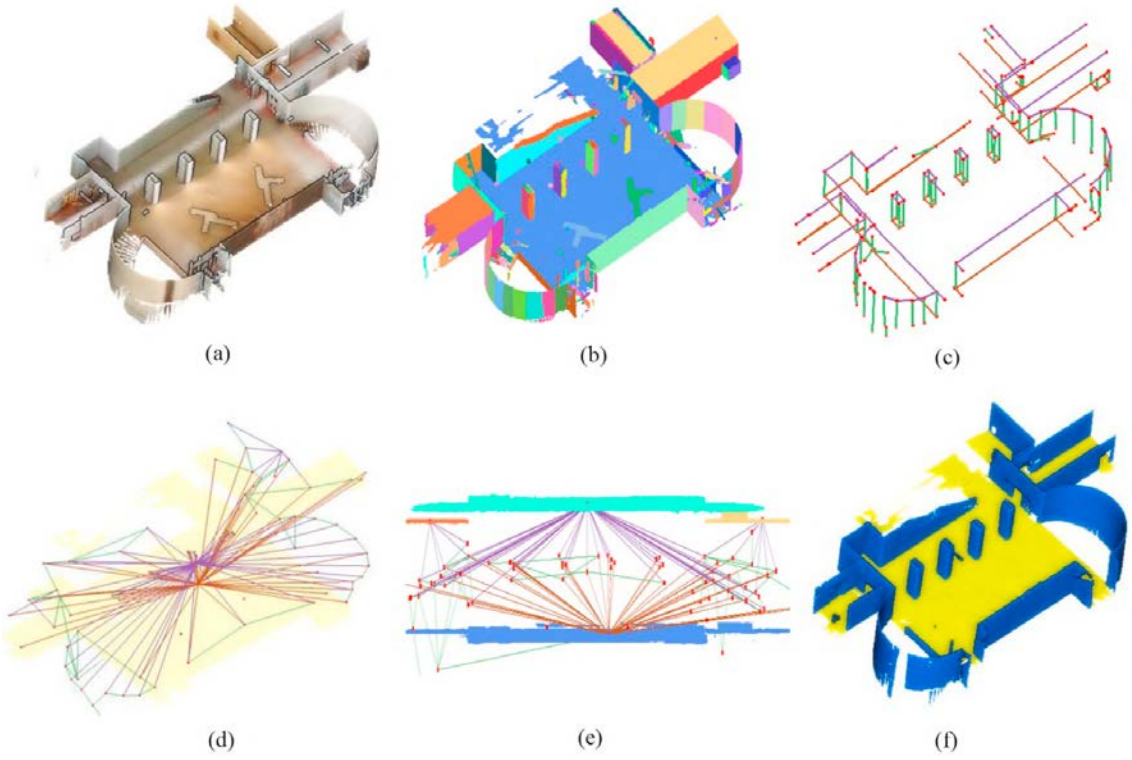


Figure 3.3: The process of identifying a permanent structures proposed by Nikoohemat et al. (2020). (a) Point clouds, (b) points segmented using the surface growing algorithm, (c) intersection between adjacent segments, (d) and (e) the adjacency graph where edges are colored by three classes (wall, floor and ceiling), (f) detected walls (blue) and floor (yellow).

Deep learning has proved its superiority in many real-life applications. Unfortunately and since the point clouds don't have a sequential order, like images or sound, Deep Neural Network (DNN) cannot be trained on point clouds right away. As a picture is organised in pixels, a point cloud has to be divided into chunks or voxels of fixed size to be used to train a DNN. Depending on their size, voxels can become very memory expensive because to capture in-depth features a high resolution is needed. Lamentably, in 3D space, the

memory consumption rises by the power of three. Nonetheless, there has been a lot of research in this area.

For example, Breu (2019) investigated the application of a DNN to detect objects in a point cloud of a construction site. More specifically Breu used a custom implementation of PointNet (Qi et al., 2016) in PyTorch[1] to extract features from synthetic random situated mesh objects, such as containers, houses, excavators, etc. The proposed workflow cut the generated point cloud into equal slices and samples points from the surface of the mesh objects and then preprocesses them to make them trainable by PointNet. While the method achieved good results in artificially generated data, in real data, it showed a low performance. As the author explains, this approach has two main drawbacks: One is the lack of appropriate real training data, and the second is the limitation of PointNet to process a maximum of 2500 points per input. The latter suggests that this approach is suitable for detecting only small objects in the point cloud, considering that scenes, where large objects are located, could perfectly have millions of points.

Another related study was done by Sun (2020). She leverages a trade-off method that combines the 3D distance information with the 2D detection results from YOLOv3 to generate frustum proposals. Then uses Frustum PointNets to implement 3D instance segmentation and 3D bounding box estimation on the point clouds. The 2D object detection is based on a panorama image rather than a normal field of view (NFoV) images. Therefore it is necessary to map the panorama to NFoV images and projecting the detected results back to the panorama. This approach allows a very fast computational performance for 3D object detection. However, it has the following three main drawbacks: First, the results of the 3D detection heavily depend on the 2D detection performance, which depends on colour information and image-background conditions. Second, the 3D detector can merely detect one object in each frustum. Since the implemented 3D segmentation networks only segment one object from each frustum, the occluded objects are regarded as the background if one was already detected. Third, the target object has to be present in at least two frustums with a large difference in its view angles. Otherwise, the generated 3D box will be biased, since the intersecting space still cannot be obtained and the background elements can not be eliminated.

In another recent research, S. Zeng et al. (2020) use deep point features extracted with a DNN to perform building element retrieval in point clouds of construction sites and historical buildings. The method is based on the idea of metric learning, it can classify building elements with complex geometrical shape characteristics without requiring pre-built CAD/BIM models, but just a user exemplar selection from the laser-scanned point cloud itself. According to the authors, this method can extend to complex building environments and generalise better to unknown objects, including temporary structures and equipment. The DNN architecture used to compute the features is shown in Figure 3.4. Their method demonstrates to achieve high precision. However, it has the following three drawbacks: First, it is not rotational-invariant, i.e. it retrieves only elements that are a rigid translation of the selected exemplar; second, extracting the deep features is very

[1] More specifically, he used the implementation from fxia22: https://github.com/fxia22/pointnet.pytorch

computationally expensive; and third, it is a semi-automated approach that additionally requires some domain knowledge by the user to select the correct building element exemplar with appropriate boundaries.

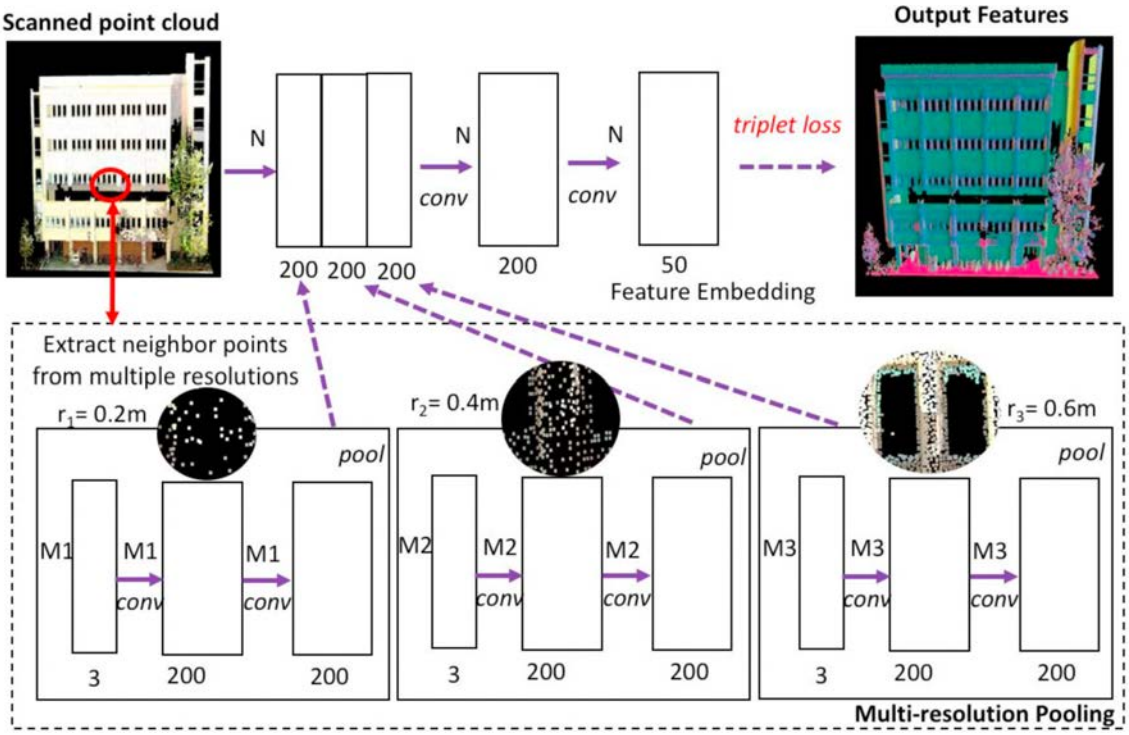


Figure 3.4: Deep neural network architecture used to compute point-level features in the method proposed by S. Zeng et al. (2020).

Xu et al. (2018) propose a method to detect scaffolds in photogrammetric point clouds of construction sites. Firstly, and after a preprocessing, they automatically isolate the building facades together with scaffolds using an orthogonal projection and a local maximum threshold approach. Then they use a histogram in the vertical axis, to find the different horizontal structures of the scaffolds. Afterwards, a plane fitting algorithm based on RANSAC is applied, with the constrain of just fitting horizontal and vertical planes. Subsequently, the parallelism between the facade and the scaffolds is exploited to separate these two objects. In the next step, they used a proposed feature descriptor for point clouds, called Linear Straight Signature Histogram of Orientations (LSSHOT), which is based on Principal Component Analysis (PCA). The robustness of their descriptor, allows them to identify the different building parts of scaffolds from low-quality point clouds.

Additionally, a random forest algorithm was trained to classify the output of their feature descriptor. Finally, Xu et al. also proposed a method to reconstruct the small patches of the classified points, and so endow them with regular parametric representations, or in other words, building a CAD-model of the recognised parts of the scaffold out of the segmented point cloud. This methodology is illustrated in Figure 3.5. This seems to be a compelling method, although it assumes that the scaffold elements have a width of 0.8 m and are always next to a wall, which is not necessarily always the case.

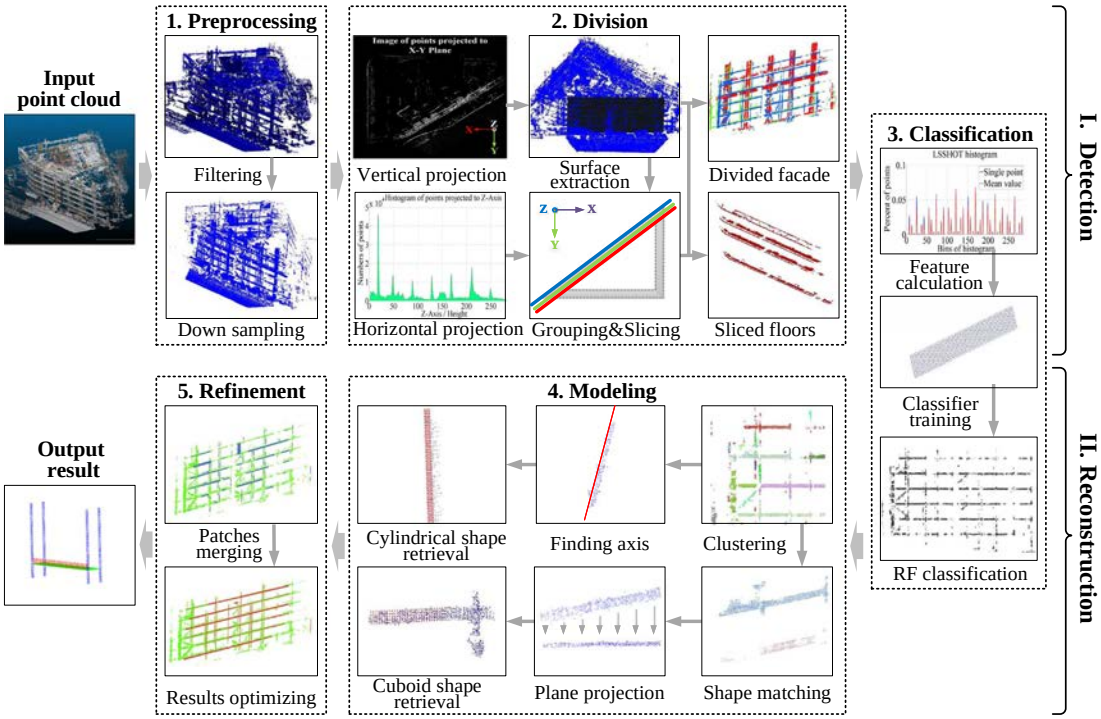


Figure 3.5: Proposed methodology for the detection and reconstruction of scaffolds by Xu et al. (2018).

More recently, Wang (2019) proposed a workflow to detect scaffolds. However, instead of using photogrammetric point clouds, they used laser scanner point clouds. The workflow was devised to enable automatic safety regulation checks over scaffold elements. Their method extracts horizontal and vertical slices of previously manually cut small sections of the point clouds where scaffold elements are present. Subsequently, minimising the root-square-error, circles are fitted in every horizontal section. Once a circle is detected, its centre is vertically projected in a grey-scale image. Later a threshold on this image reveals the position of uprights, and thus also of the scaffold element. Then with a similar strategy and using plane and line fitting techniques all the other parts of the scaffold (work platforms, toe-boards, and guard-rails) are successfully detected, and their position is validated against the respective safety regulations. Figure 3.6 shows some results after the application of these techniques. This method seems to be very robust, but as the authors mentioned there is still room for improvement; its main three drawbacks are: First, whereas the process does not assume the presence of a wall next to the scaffold for its detection, as in Xu et al. (2018), it supposes that its uprights are perfectly vertical, which is not in every case true. Second, the method was developed to detect scaffolds in previously small manually clipped sections of the point cloud, which makes the task much more manageable as working with the entire point cloud of a construction site. The third and final drawback is that slicing a point cloud every 0.05 m, generating a vertical projection of every slide and on the top of that performing a circle fitting in every projection, is very computationally expensive and maybe even not feasible in large point clouds.

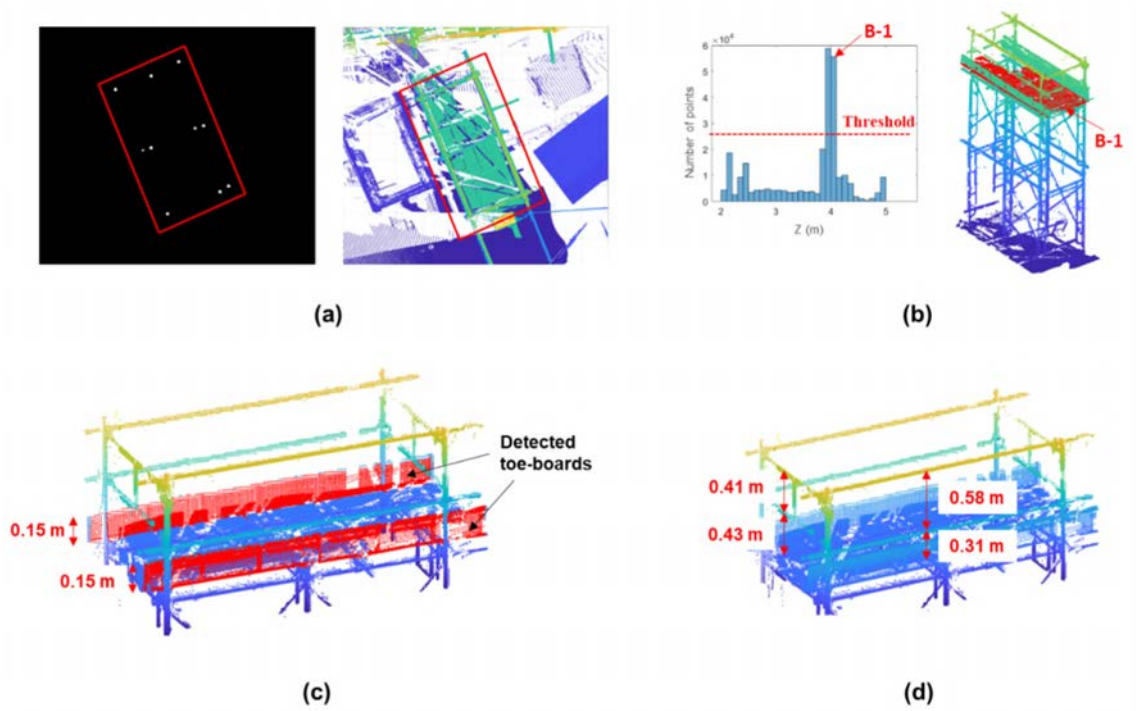


Figure 3.6: Results after applying the technique proposed by Wang (2019): (a) the detected area of the scaffold shown as a red polygonal area, (b) detected work platform from the point cloud data shown as red points, (c) detected toe-boards and their heights, and (d) detected guard-rails and their locations.

For a more comprehensive literature review about applications of 3D point cloud data in the construction industry featured in publications between 2004 and 2018, the reader is referred to Wang and Kim (2019).

3.3 Conclusions related work

There has been a lot of improvements in automatic construction progress monitoring in the past 10 years. While some methods are based on Images, from which point clouds could be generated, others are based on laser scanner point clouds. Moreover, most of the techniques presented in that section rely on the assumption of the existence of the 4D building information model. With such a model and a point cloud, an as-built vs as-planned comparison is possible, allowing the automatic monitoring of the progress. However, the automatic tracking is hindered due to the presence of temporary building elements which should still be detectable without having such a model.

There has not been much previous research on the detection of exactly the selected objects using point clouds. Most of the related work focuses on the reconstruction of a building information model from point clouds (Fichtner, 2016; Maalek et al., 2019; Macher et al., 2017; Nikoohemat et al., 2020).

While deep learning approaches for point cloud segmentation seem to be very promising, aside from requiring many labelled data, they still have some additional critical limitations.

One limitation is the maximum number of points that can be processed at the same time, making the method not suitable to detect large objects. Another main drawback is, e.g. the non-rotational invariant constraint, which restricts the practice to only be able to find items with known XYZ-orientation.

More promising methods like the ones proposed by Xu et al. (2018) or Wang (2019), take advantage of the verticality of the objects to detect scaffold elements, as well as in-depth knowledge of the underlying geometry of the objects, like dimensions of the uprights or possible bay width distances. While still having some drawbacks, these methods showed promising results for the specific case of scaffold detection in point clouds of construction sites. Further work can be done to detect scaffolds more efficiently, as well as to recognise additional objects, such as cranes and formwork elements.

Chapter 4

Methodology

4.1 Overview

The workflow of the proposed crane, scaffold, and formwork detection is illustrated in Figure 4.1. The first step is a preprocessing of the raw laser scanner point cloud, in which a downsampling step based on an Octree-voxel grid method is applied, followed by a rotation of the point cloud, that will align it to the building axes.

The second step is the detection of cranes, in which different techniques are used. To recognize cranes, first, the Regions of Interest (ROIs) that may contain cranes are separated from the rest of the point cloud using image processing techniques over a vertical projection of the point cloud. A second step involves 3D vertical contour detection and merging. Later, an algorithm will search a pattern characteristic of a tower crane in these vertical lines, which will reveal the possible positions of the cranes. Subsequently, the final location of cranes is determined by applying checks over vertical cross-sections projections.

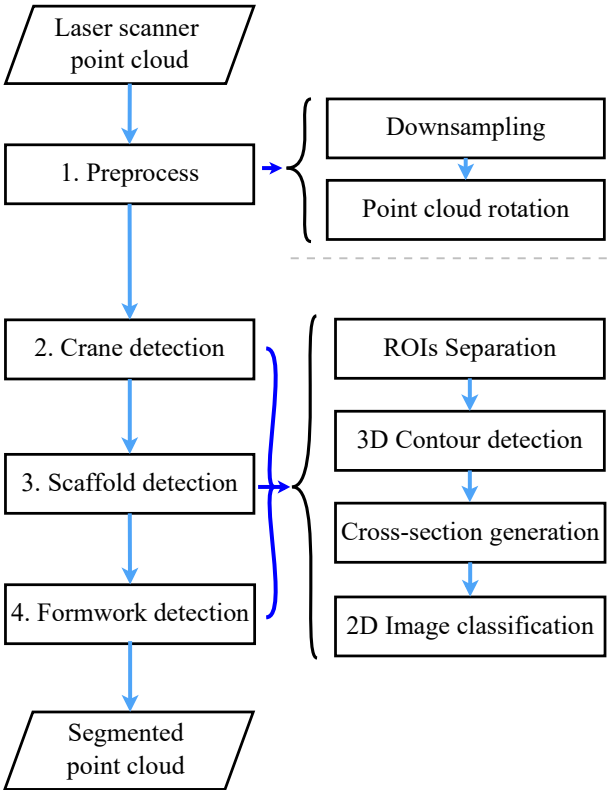


Figure 4.1: Workflow overview.

The third step is the detection of scaffolds. Here the procedure is very similar to the detection of cranes. However, the pattern which will be searched in vertical lines is specific to the standardized dimensions of scaffold elements.

As the last step, formwork elements are detected. Here once again, the ROIs that might contain formwork elements are prefiltered, vertical cross-sections projections are generated, and a Machine Learning (ML) algorithm is used to determine the presence of formwork elements. In the following sections, these four steps are explained in more detail.

4.2 Preprocessing

4.2.1 Downsampling

Filtering or downsampling the point cloud is vital for two reasons: First, it will allow the method to take advantage of the fact that the point cloud has a uniform density, and second, it is the first step that will reduce the computational cost as the number of points is reduced almost by half.

As already mentioned in Section 2.1.1, while working with massive point clouds and small leaf sizes, the VG method of PCL can not be implemented right away. This is because there is a maximum limit to the number of voxels the computer can handle before running out of memory. For example, with a voxel-grid leaf size (VG_{ls}) of 5 mm the method will throw an exception if the point cloud is broader than 6.45 m in every dimension (i.e. if ΔX, ΔY and ΔZ are all bigger than 6.45 m). On the other hand, if VG_{ls} is, e.g. set 8 times larger than 5 mm, to 0.04 m (like done in Xu et al. (2018)) the maximum point cloud size to be able to implement the method is 51.6 m in every dimension or, equivalently, 137.388 m^3.

Since in this thesis, it is essential to have a very dense point cloud, another filtering method has to be implemented that allows working with large point clouds and a VG_{ls} of 5 mm. To solve this problem, and as already introduced in Chapter 2, the point cloud is first organized into an octree with a resolution of 5 m. Once the octree is created, the voxel grid method with a leaf size of 5 mm can now be applied in every leaf voxel of the octree. This is equivalent to dividing the point cloud in occupied 3D squared boxes of 5 m size and then treating every peace of the point cloud as a separated point cloud. While it is also possible to use an octree resolution of 5 mm directly, experiments showed that with this resolution, the process does not only require more time but also does not filter the point cloud correctly, leaving unwanted points between the boundaries of the octree leaves.

The selected method not only allows a uniform and fast downsampling of large point clouds but also preserves the color of the original point cloud. Additionally, since VG_{ls} is small enough, all the necessary geometric features are also conserved. Figure 4.2 shows a large and a small section of the point cloud together with their downsampled versions.

4.2.2 Point cloud rotation

This step aims to rotate the point cloud so that it is aligned with the principal axes of the building. While the data does not necessarily need to follow the Manhattan-World assumption [1], the fact that most of the buildings or at least large sections of them are design based on a rectangular grid will be exploited here. To be able to take advantage of this fact, it is necessary to rotate the point cloud so that the main walls are parallel to

[1] This assumption states that indoor scenes were built on a cartesian grid which led to regularities in the image edge gradient statistics.

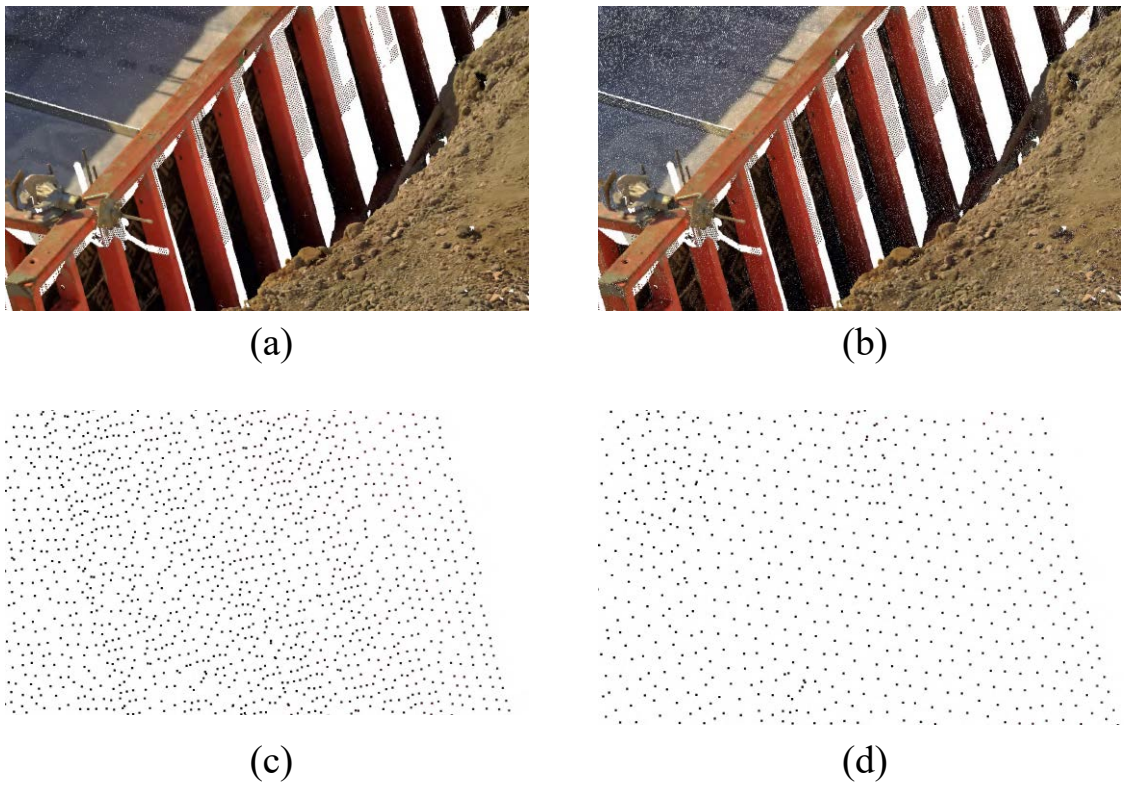


Figure 4.2: Point cloud down sampling with a leaf size (VG_{ls}) of 5 mm: (a) original point cloud section with a formwork element; (b) downsampled point cloud with a leaf size of 5 mm; (c) smaller section of the point cloud; (d) downsampled version of (c).

either the x- and the y-axis. Figure 4.4 shows a top viewpoint cloud which is not aligned with the structural building axes.

The automated method to find building axes proposed here consist of three steps: 1) Walls ROIs Separation, 2) 2D Line detection and 3) determination of the final angle of rotation.

It is imperative to mention that before applying this method, the point cloud has to be divided into different sections, each section with the points for each building floor. This is a requirement for the process to be able to filter objects by its minimum height. Figure 4.4 illustrates the first level of the building, this is the only step that is still done manually, the rest is a fully automated process.

4.2.2.1 Walls ROIs separation

In a construction site, many vertical objects are present, however, usually, only large load-bearing walls are correctly aligned to the structural axes of the building. Hence, and in an effort to accomplish a reliable method, firstly, these large walls are separated from the rest of the point cloud.

This separation is mainly done by applying thresholds on a vertical projection of the point cloud. When trying to detect vertical objects in a point cloud, generating vertical projections seems to be a very suitable approach. This is because it will not only reduce

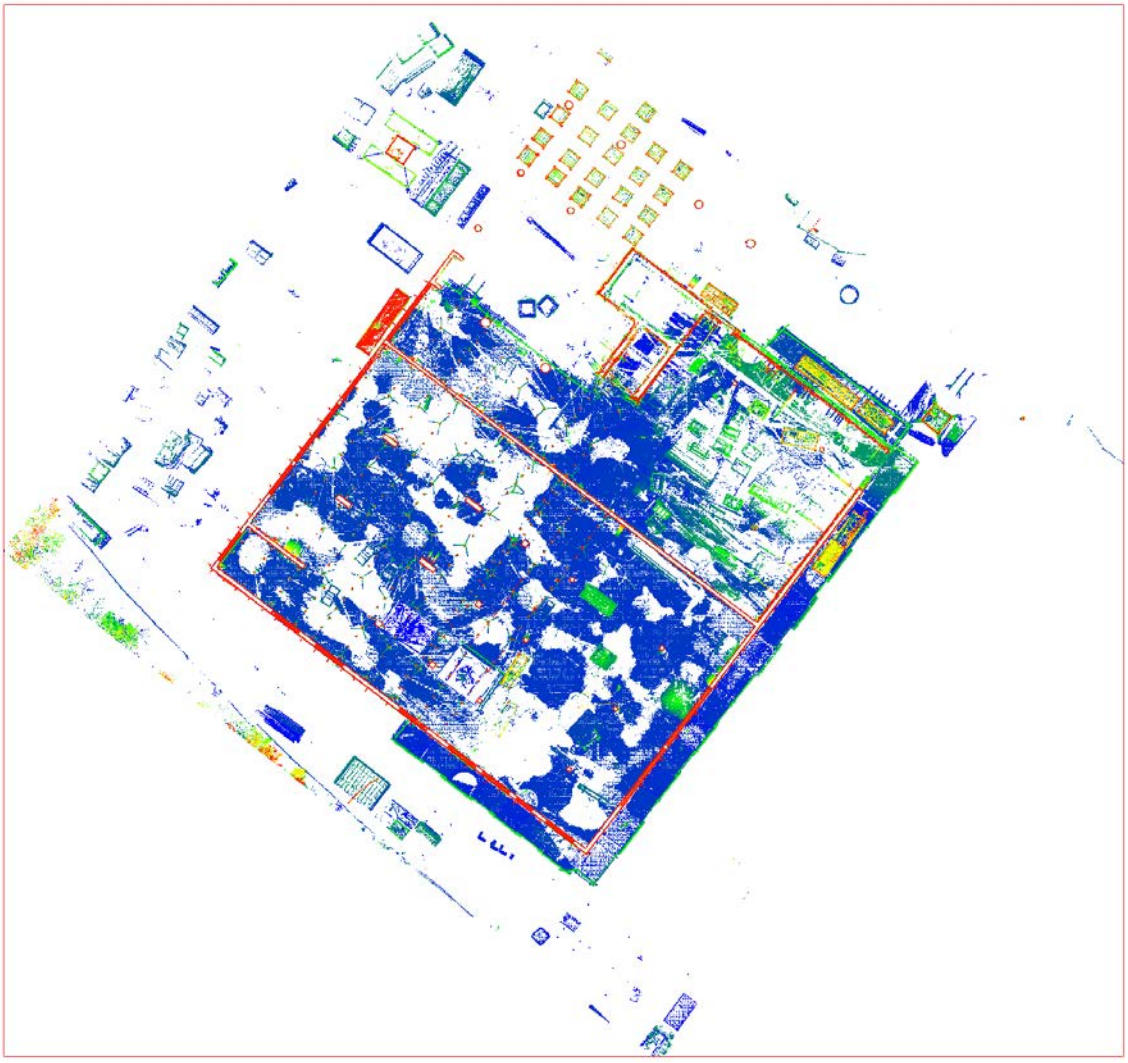


Figure 4.3: Top view of a not aligned point cloud with the structural building axes. Test dataset: Nr. 2.

the amount of data to be processed but also because 2.5D images (i.e. 2D images with height information) can be processed much faster than 3D points.

As the point cloud was already downsampled, it is known that the minimum distance between two points is 5 mm. Considering the presence of occlusions in the point cloud, and the possible presence of formwork covering the walls, it is assumed that vertical walls may have at least 1.2 m (h_{min}) of height, which is around half of the height of an average wall. Since the points are spaced every 5 mm, a projection of the points between two consecutive building floors in the XY plane in a grayscale accumulation image (I_{acc}), which stores the number of points projected on each pixel, allows the differentiation of the objects by its height.

To make this point clear, consider a vertical line of 1 m length. If the line is formed by points every 5 mm, it implies that the line is actually a column of 200 points. If these points are projected in the XY plane in a grey-scale accumulation image, they will be represented

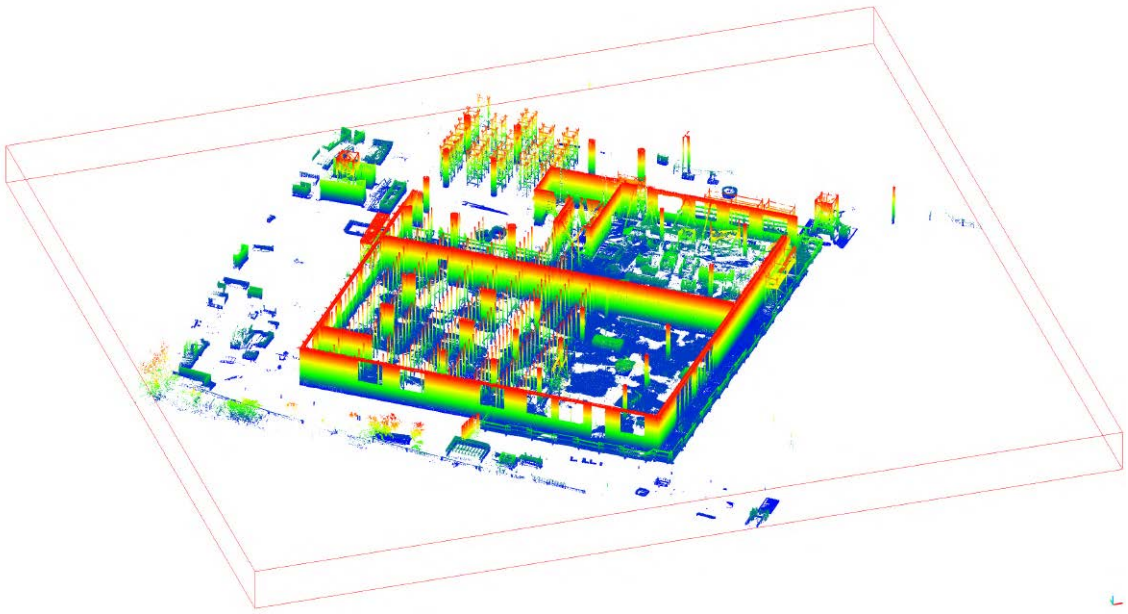


Figure 4.4: Clipped first floor of the point cloud. Test dataset: Nr. 2.

as a pixel with value 200. In this way, it is possible to separate objects of different heights using a vertical projection, as long as they have a vertical non-occluded surface.

Formally, the threshold operation that transforms the image I_{acc} into a binary image B, where only vertical elements higher than h_{min} are present, can be defined as follows.

$$B = \text{Th}\left(I_{\text{acc}}\right) = \{(r, c) \in I_{\text{acc}} \mid I_{\text{acc}}(r, c) > h_{\text{min}}\}$$

Where r and c are the corresponding row and column indices of the pixels in the image.

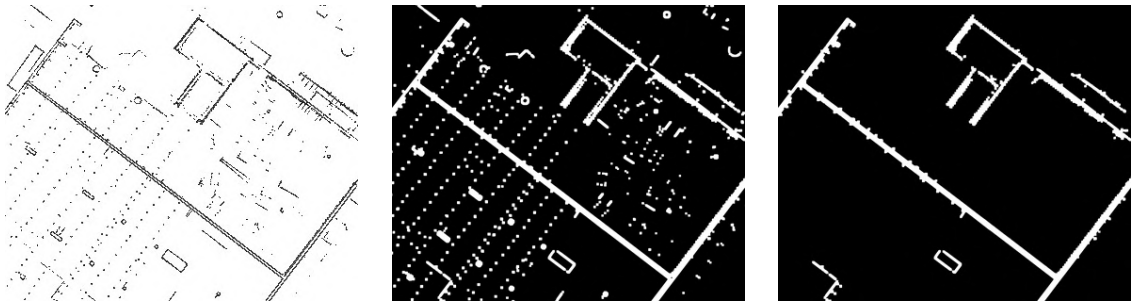
Subsequently, the objects can also be differentiated by their size, using the blobs or connected components produced in the vertical projection. Before this, 10 iterations of a dilation morphological operation with a *structural element* (S) with rectangular shape of size 10 x 10 (S_{R10}), will join small blobs that are close to each other and which may conform bigger objects (as shown in Figure 4.5b); later the blobs can be separated by its number of white pixels. Large objects will then be represented by blobs with large areas or equivalently, with a large number of pixels. The relationship between the number of pixels and the real area depends on the *grid side length* used while creating the vertical projection. In this thesis the *grid side length* has a value of 5 mm, this means that one pixel in the image represents 5 mm or, equivalently, a horizontal line of 1 m length will be represented by 200 pixels. Consequently, 1 m^2 will be represented by a region containing 40.000 pixels.

Let D be the image resulting from applying the dilation operation denoted by $\oplus$ with a structural element S to the binary image B ($D = B \oplus S$) and let $C_1, C_2, \ldots, C_n$ be the connected components on the binary image D ($C_i \cap C_j = \varnothing \mid i \neq j$). Then, the ROIs of the walls (W_{regions}) are the connected components C_i of the image D which have an area larger than A_{min}:

$$W_{\text{regions}} = \{C_i \in D \mid A(C_i) > A_{\min}\}$$

Where A is the count of the pixels in the corresponding blob $A(C_i) = \sum_{(r,c)\in C_i} 1$ divided by 40.000, which is the number of pixels in one square meter.

Since all the walls in a room will be connected, after the dilation step, they will be represented by a single blob in the vertical projection. Assuming that the minimum length of all walls in a room is 5 m and its width is 0.3 m, it means that the walls in a room will be represented by blobs with a minimum area of A_{min} =1.5 m^2. Figure 4.5c shows the final wall ROIs, which are the result of filtering by size the blobs in a dilated vertical projection after passing a height threshold.



(a) (b) (c)

Figure 4.5: Wall ROIs in a vertical projection: (a) original vertical projection (for better visibility, the inverted binary version is shown here); (b) binary image after threshold and dilation; (c) final Wall ROIs (W_{regions}) after separation by blob size. Test dataset: Nr. 2.

4.2.2.2 2D Line detection

Once the ROIs of large walls are isolated in W_{regions}, this image is used as a mask to filter the original vertical projection. Then, using the probabilistic Hough transform algorithm, 2D lines are fitted in this filtered vertical projection. For better line detection accuracy, the angular resolution is set to $\pi/(180 \cdot 100)$, allowing the algorithm to search for lines with an angle precision of up to two decimal places.

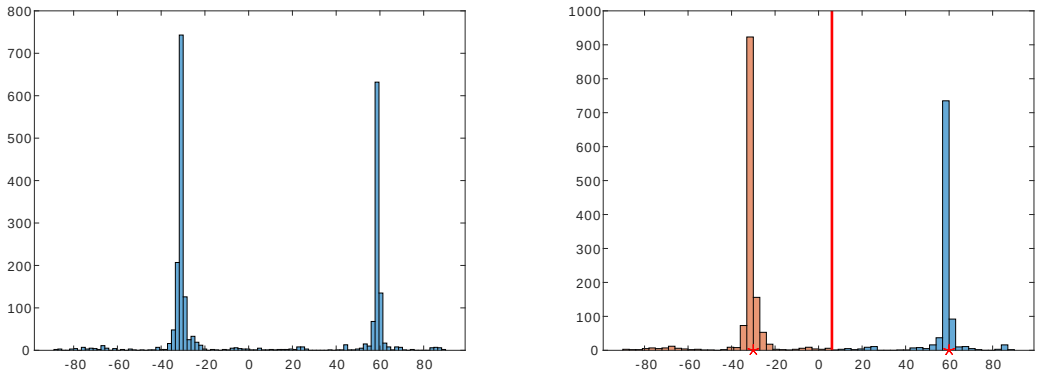
The minimum line length and the maximum space allowed between lines were set at 0.5 m and 1 m, respectively. 0.5 m is considered an appropriate value for the minimum line length, since the smallest lines can be part of smaller objects that are not aligned with the structural building's axes and additionally, only in sporadic cases, a wall is shorter than 0.5 m. On the other hand, the maximum space allowed between lines will merge smaller segments that lie on the same wall. Figure 4.6 presents the regions of the original vertical projection where only walls and large objects are present, together with its 2D detected lines.

(a) (b)

Figure 4.6: 2D line detection in filtered vertical projection: (a) inverted filtered vertical projection after applying W_{regions} as a mask; (b) 2D detected lines in (a). Test dataset: Nr. 2.

4.2.2.3 Angle of rotation

Finally, the angle of rotation is determined using the *k-means* algorithm over the slopes of the previously detected 2D lines. This method assumes that there are not many walls that are not orthogonal to the building axes. Under this assumption, the majority of the detected 2D lines will have either of the two angles that correspond to the searched angle of rotation. Therefore, it is possible to apply *k-means* to find out the values of the 2 centres of the two groups of line-angles in a one-dimensional histogram. Figure 4.7a shows a one-dimensional histogram of the azimuth angles of the detected 2D lines and Figure 4.7b shows the resulted two groups and centres after applying *k-means* a maximum of 10 iterations or until an error of less than 0.1 is achieved. In this case, the two found angles are -31.3° and 61.3°, taking either of them as the angle of rotation concerning the Z-axis will align the point cloud with the building axes.



(a) (b)

Figure 4.7: Final angle of rotation with *k-means*: (a) one-dimensional histogram of the azimuth angles of the detected 2D lines; (b) the resulted two groups and centers after applying *k-means*.

Since the two building axes should be orthogonal, the difference between the two centres has to be around 90°, if it is not the case the algorithm will throw an exception. This is just to confirm that the correct construction axes were found.

Once the point cloud is downsampled and aligned with the structural building axes, the next step is the detection of the target objects.

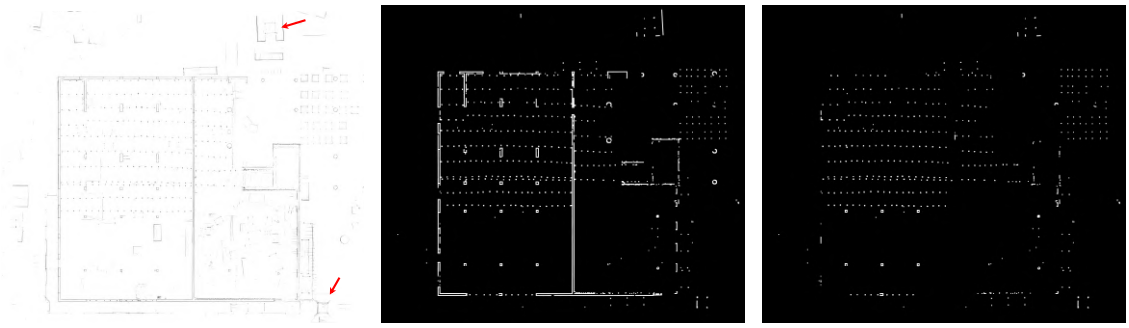
4.3 Crane detection

This step aims to detect the cranes present in the scan data. It consists of three sub-steps, which are 1) Cranes ROIs Separation, 2) 3D contour detection and 3) individual crane recognition. Details of each step are described as follows.

4.3.1 Cranes ROIs separation

This step aims to efficiently filter out points that have more likelihood to belong to a crane from the rest of the point cloud. Similarly, as done to filter the regions with walls to rotate the point cloud, this section will take advantage of the verticality of the tower cranes.

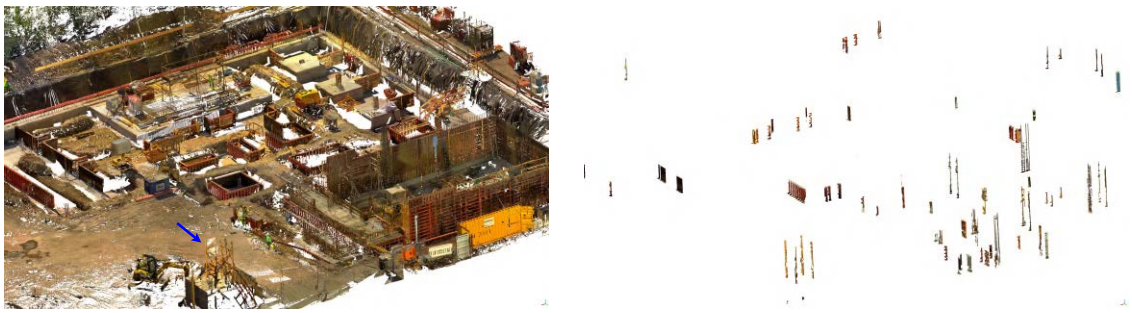
Analogously as performed in Section 4.2.2.1, this step applies a threshold on a dilated vertical projection to filter objects taller than $h_{min} = 0.7$ m and with an area in a vertical projection between $A_{min} = 0.0075$ m^2 to $A_{max} = 0.3$ m^2. A_{min} was selected assuming that a steal angle of the tower mast has a minimum side length of 0.10 m (or a bit less because of occlusions) and A_{max} thinking about the possibility that a side of the crane (of maximum 3 m long) would be in a single blob. This might be caused by the presence of other objects like lamps, posters or banners that are hanging on a side of the tower crane mast. Here the dilation operation is performed only 3 times with a *rectangular structural element* of size 10 x 10 (S_{R10}). Figure 4.8 illustrates the filtering process of the crane ROIs with a vertical projection.



(a) (b) (c)

Figure 4.8: Crane ROIs in a vertical projection: (a) original vertical projection with arrows indicating the location of the cranes (for better visibility the inverted binary version is shown here); (b) binary image after threshold and dilation; (c) final crane ROIs after separation by blob size. Test dataset: Nr. 2.

At this point, it is essential to clarify that the goal of this step is not to filter all the points that belong to cranes, but at least those that belong to the characteristic four vertical steel angles that constitute the mast of a tower crane. Later, these four vertical angles will be searched among all the other thin and tall objects that will pass this filter. In Figure 4.9b, all the elements that pass the filter are shown.



(a) (b)

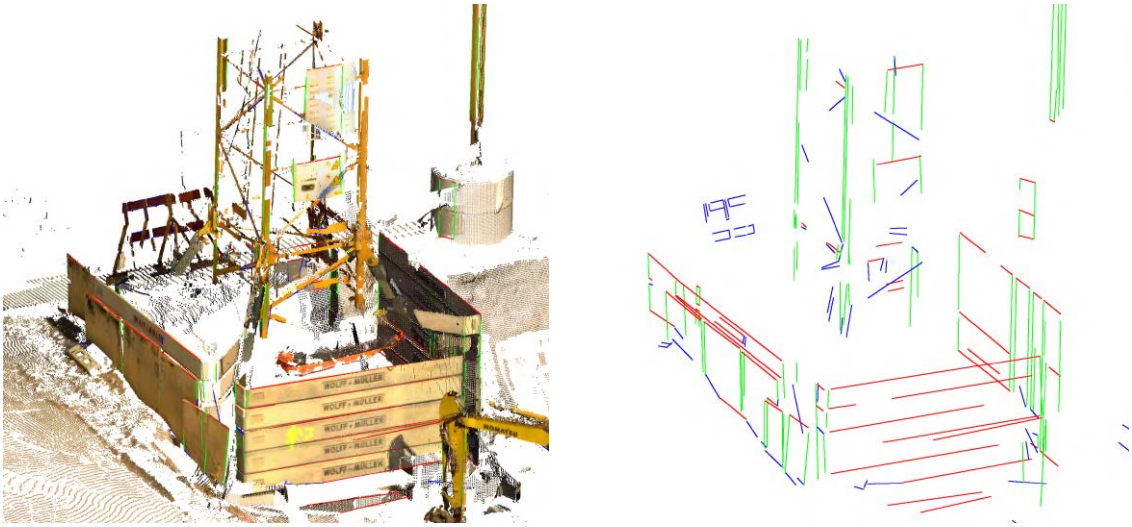
Figure 4.9: Crane ROIs in point cloud: (a) original point cloud with a blue arrow indicating the location of the crane; (b) Crane ROIs, notice the presence of other thin and tall objects additional to the cranes. Test dataset: Nr. 1.

4.3.2 3D contour detection

The 3D line detection is performed using the code provided by Lu et al. (2019). They implemented a fast 3D line detection algorithm that demonstrated to overcome other state-of-the-art methods. As they carefully explain in their paper, the process is divided into three main steps: First, the point cloud is segmented in regions based on the previous calculation of the Principal Component Analysis (PCA) information of every point; second, 3D planes are fitted in every region and lines are detected over a 2D projection of this planes which are then projected back to the 3D space; and finally, in a post-processing step, the detected 3D lines are passed through an outlier removal and a horizontal merging process. For a more in detail explanation of the complete procedure, the reader is referred to Lu et al. (2019).

The implementation of the algorithm of Lu et al. (2019) plays a crucial role in the proposed object recognition method, not only because it allows translating from unorganized points to 3D lines that delineate the objects, but also because it is fast. Therefore its implementation in large point clouds is very convenient. Figure 4.10 illustrates the 3D line detection results in a point cloud with the lower section of a tower crane mast.

Once the lines are detected, they are divided according to their inclination into three groups: vertical, horizontal and diagonal lines. A line is considered vertical if its angle concerning the Z-axis is smaller than 5°. Similarly, a line is deemed to be horizontal if its angle concerning the Z-axis is something between 85° to 95°. The rest of the lines are classified as diagonal. Figure 4.11a presents the detected 3D contours in the crane ROIs point cloud with different colours depending on their inclination.



(a) (b)

Figure 4.10: Detected 3D contours in a point cloud with a crane: (a) original clipped point cloud of a crane with only points of the first building level; (b) 3D detected contours in (a). Test dataset: Nr. 1.

Subsequently, the vertical lines are merged. This step is essential because sometimes, due to point cloud occlusions, the algorithm of Lu et al. will accurately detect small vertical segments that actually are part of a single large vertical object (see for example the crane detected lines in Figure 4.10b or in Figure 4.11a). Since the goal here is to detect relatively large vertical objects, it seems sensible to merge the small pieces of vertical lines into a single line if they are close enough. To this end, the vertical lines are treated as 2D points on a horizontal plane, then these points are grouped taking into account that the maximum distance between two points in a group must not exceed 20 cm. The lines in a group are merged, taking the total average of the X and Y coordinates and the minimum and the maximum Z values of the lines in the group. Once the vertical lines are merged, they are also filtered by their length, leaving behind the lines that are less than 1.5 m long. Figure 4.11b shows the resulted merged vertical lines after filtering by their size.

(a) (b)

Figure 4.11: Detected 3D contours in the crane ROIs point cloud: (a) in cyan, red and magenta are the vertical, horizontal and diagonal originally detected 3D contours, respectively; (b) merged 3D vertical contours in crane ROIs point cloud. Test dataset: Nr. 1.

4.3.3 Individual crane recognition

Now that the vertical lines are detected, the pattern that characterized a crane will be searched in these vertical lines. As explained in Section 2.2.1, the mast of tower cranes always has a characteristic square section, with a lateral size between 1 m (1.2 m of tower cranes and 1 m for self-erecting cranes) and 2.5 m. Therefore, the main goal of this step is to find four vertical lines, which follow this geometric distribution, typical of a Cuboid. To find the groups of vertical lines with this geometry, the algorithm proposed here (see Algorithms 1, 2 and 3) will first search for pairs of vertical lines with a distance in between in the range from 0.8 m to 2.7 m (± 0.2 m of the original range). Once a pair of lines with this characteristic is found, the algorithm checks if there exists some overlap in their height, i.e. if both lines are in similar height ranges. The exact overlap check is detailed in Algorithm 2.

As illustrated in Figure 4.12, there are four possible regions where the other two steel angles could be present. The location of these regions is known since cranes mast have a square horizontal cross-section (as shown in Figure 2.4). The other two steel angles must be parallel on either of the two sides of the horizontal line that connects the two originally found possible steel angles. As shown in Algorithm 1, the method will use the distance between the two original lines to determine the centres of these four regions. A tolerance of 0.2 m around these four centres will cover all the possible areas where the other two steel angles could be present. Notice in Algorithm 3 that also the cases when only three lines were found are saved (in P_1). However, experiments on the tested dataset showed that checking only the groups of four lines is sufficient.

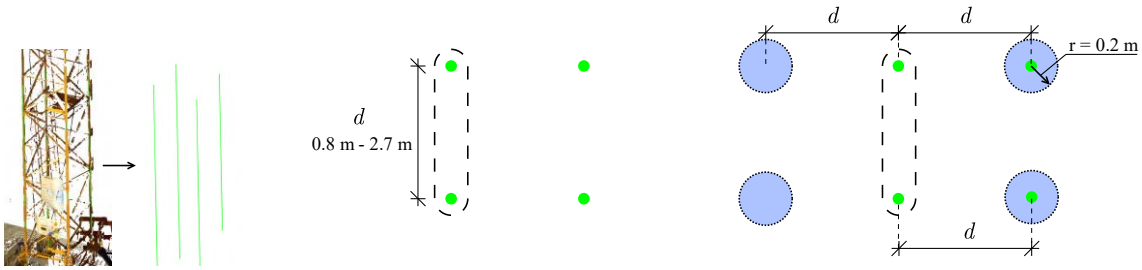


Figure 4.12: Determination of the location of the possible regions with cranes: (a) original 3D vertical lines of a tower crane mast; (b) top view of the vertical lines, a pair of vertical lines with a distance in between in the range from 0.8 m to 2.7 m is indicated with a dashed line; (c) the other two steel angles could be in the four blue regions, which are 0.2 m radius circles located at a distance d (distance between the first two found lines) on each side in the perpendicular direction to the horizontal line connecting the original pair of lines found.

After detecting the groups of four lines that reveal the possible location of the cranes, the next step is to extract the points that belong to the likely crane mast faces. For this purpose, the algorithm will check the presence of points located between every two continuous vertical lines of every detected group. This will filter out groups of lines that

Algorithm 1: Find pattern in vertical lines

Input : A vector with the merged vertical lines $M = \{L_0, L_1, ..., L_n\}$
Output : Vector of vectors of line indices $P \leftarrow \varnothing$ revealing possible crane locations

```
for ∀(L_i, L_j) ∈ M : i < j do
    p_i ← L_i^0, p_j ← L_j^0
    d ← ‖p_i − p_j‖
    u ← (p_j − p_i)/d
    u^⊥ ← (−u_y, u_x)
    if 0.8 < d < 2.7 and overlap(L_i, L_j) then
        C ← {p_i + du^⊥, p_j + du^⊥, p_i − du^⊥, p_j − du^⊥}
        R ← ∅
        for ∀c ∈ C do
            for ∀L_k ∈ M : i < k and k ≠ j do
                p_k ← L_k^0
                t ← ‖p_k − c‖
                if t < 0.2 and overlap(L_i, L_k) then
                    R ← R ∪ k
                else
                    R ← R ∪ 0
                end
            end
        end
        saveLineIndices(i, j, R^0, R^1)
        saveLineIndices(i, j, R^2, R^3)
    end
end
```

Algorithm 2: Determine if there is overlapping between vertical lines

```
bool ovelap(L_i, L_j)
    Input  : two vertical lines L_i, L_j
    Output : true if there is a vertical overlap between the two lines
    Zmax_i ← max(L_i^0.z, L_i^1.z), Zmin_i ← min(L_i^0.z, L_i^1.z)
    Zmax_j ← max(L_j^0.z, L_j^1.z), Zmin_j ← min(L_j^0.z, L_j^1.z)
    if Zmin_i < Zmax_j < Zmax_i or Zmin_i < Zmin_j < Zmax_i then
        return true
    else
        return false
    end
```

Algorithm 3: Save line indices

```
void saveLineIndices(i, j, k, m)
    Input  : indices i, j, k, m of lines
    Output : saves not repeated line indices if found
    if k > 0 and m > 0 and {j, i, k, m} ∉ P_0 then
        P_0 ← P_0 ∪ {j, i, k, m}
    else if k == 0 and m > 0 and {j, i, m} ∉ P_1 then
        P_1 ← P_1 ∪ {j, i, m}
    else if k > 0 and m == 0 and {j, i, k} ∉ P_1 then
        P_1 ← P_1 ∪ {j, i, k}
    end
```

although having the same underlying geometric distribution, do not represent cranes in the point cloud. Figure 5.9 illustrates some of these cases.

After the previously described occupancy check, a vertical 2D cross-section will be created. Every cross-section represents one face of the possible tower crane mast and will then be used to classify the groups of lines as a crane or as non-crane. The algorithm will check the existence of a horizontal line in every one of the four faces of the mast of the tower crane. As shown in Figure 4.13, all the vertical cross-sections of a crane have at least one horizontal line that is at least 80% long of the total width of the image.

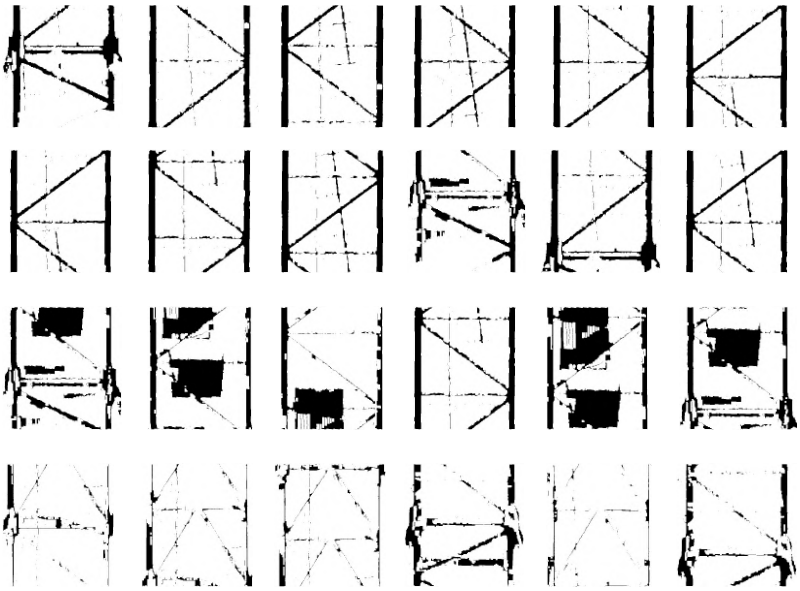
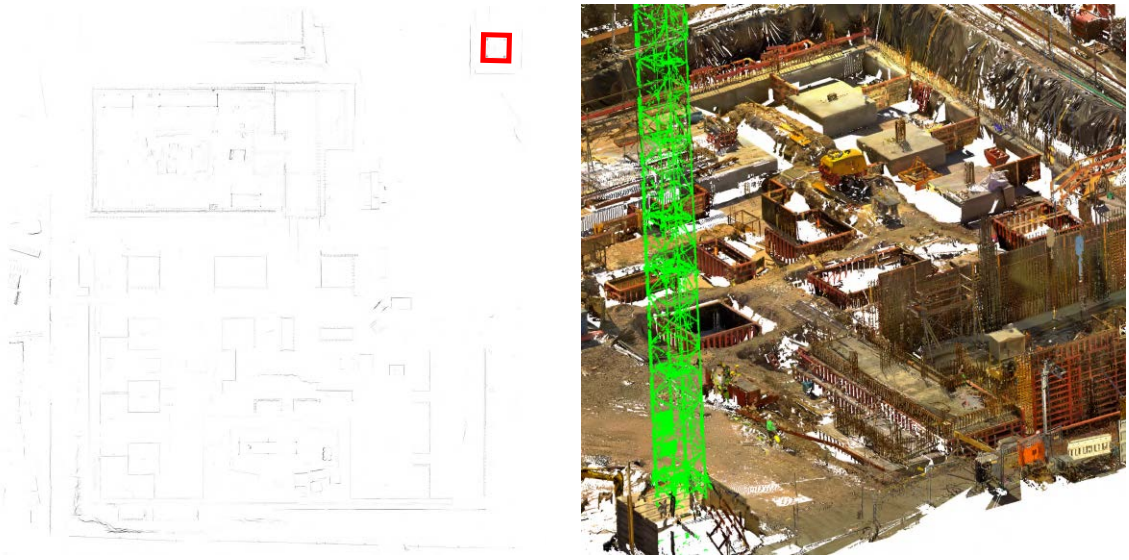


Figure 4.13: Small set of automatically generated crane vertical cross-sections

Once the four cross-sections passed this occupancy and horizontal line checks. The corresponding group of four lines should contain the possible crane points. Therefore, the algorithm will fit the smallest possible 2D rectangle, in which these four lines lie, and then all the edges of the fitted rectangle are extended outwards by a distance of 0.2 m. This is to include a larger area in case that crane components are beyond the locations of the founded lines of the tower crane mast. Subsequently, the height of every possible crane will be check to be at least 10 m below the maximum height of all the cranes point clouds. After passing this total height threshold, and as shown in the rectangle in Figure 4.14a, the region for the crane is found. Later, as shown in Figure 4.14b, the point cloud data within the rectangle is extracted. Finally, and similarly as performed in Turkan et al. (2014), once an object is detected, the corresponding matched points are removed from the point cloud, in which the next object will be searched.

4.4 Scaffold detection

The scaffold detection process follows very similar steps as crane detection, with two main differences: First, the threshold values of the ROIs separation step are different. Second,



(a) (b)

Figure 4.14: Final detected crane: (a) vertical projection with crane location indicated with a red rectangle; (b) 3D detected crane points, which are inside the region indicated in (a). Test dataset: Nr. 1.

the method for detecting the pattern on vertical lines must also be adjusted to detect not only square but also rectangular patterns characteristic of a scaffold.

4.4.1 Scaffold ROIs separation

Considering that scaffolds could be more occluded in comparison with cranes, the height threshold for scaffold ROIs h_{min} was set to 0.2 m. This step will not only filter out ground floor points, but also some short objects.

Subsequently, a dilation operation is performed 6 times with an *elliptical structuring element* of size 5 x 5 (S_{E5}). Here, the use of an *elliptical structuring element* is considered more appropriate in place of the default *rectangular* one, since the uprights have cylindrical sections. Note that the size of the structuring element is also half the size used for the crane detection, this is because the scaffold uprights are smaller than the steel angles of the crane.

The minimum blob size A_{min} was set to 0.002 m^2, while the maximum A_{max} to 0.075 m^2. This is considering that an average diameter of an upright of 0.05 m, its vertical projected area will be of 0.002 m^2. A_{max} was set considering the possible presence of other vertical objects close to the uprights, which after dilation will be merged.

Similar as with the cranes, after separating the ROIs, vertical 3D contours are detected, merged and filtered by a minimum length of 0.4 m.

4.4.2 Individual scaffold recognition

Something unique about scaffolds is that they do not always follow a squared pattern, as happens to be the case with the cranes. Nonetheless, the uprights of scaffolds typically follow a rectangular design, with dimensions that vary by manufacturer. As explained in Section 2.2.2, there are some usual ranges for the scaffold bay length and width. Moreover, as confirmed by Wang (2019), and because all scaffolds must have vertical uprights touching the ground, it is reliable to recognize scaffolds by finding the uprights.

Since the range for the bay length is greater than that of the bay width, the proposed algorithm will first search for pairs of vertical lines with a distance in between in the range from 1.5 m to 3 m (bay length range). Similarly, as with the cranes, the algorithm checks if there exists some overlap in their height (see Algorithm 2).

Analogously to the crane case and since scaffolds are rectangular, the four possible locations of the other two uprights are known. The other two uprights must be parallel on either of the two sides of the horizontal line that connects the two originally found possible uprights. Given that the bay width could vary between 0.6 m to 1.20 m, taking a distance of 0.9 m with a tolerance of 0.3 m around these four points, will cover all the possible regions where the other two uprights could be present. These potential regions with their correspondent tolerance are shown in Figure 4.15.

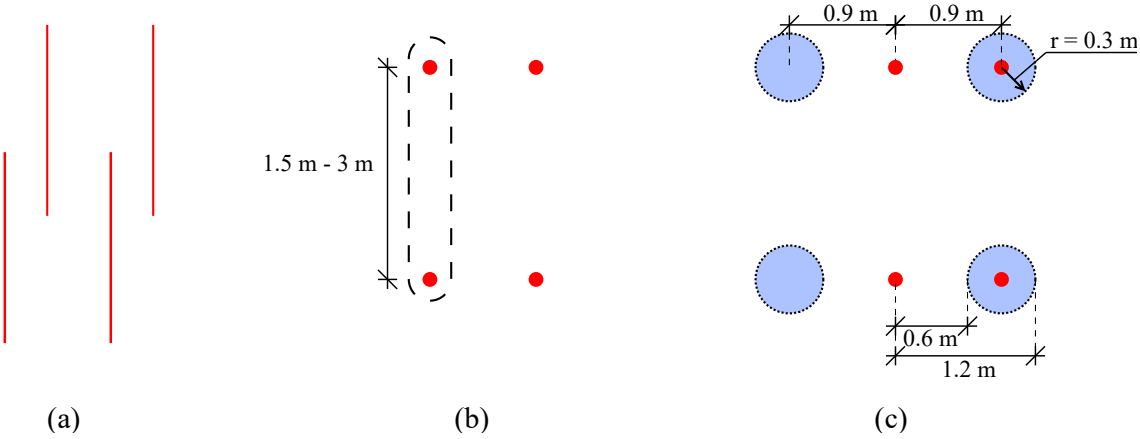


Figure 4.15: Determination of the location of the possible regions with uprights: (a) original 3D vertical lines of a scaffold element; (b) from an aerial perspective a pair of vertical lines with a distance in between in the range from 1.5 m to 3 m was found; (c) the other two uprights could be in the four blue regions, which are 0.3 m radius circles located at 0.9 m on each side in the perpendicular direction to the horizontal line connecting the previously pair of lines found.

Once the groups of four lines that reveal the possible location of the scaffolds are detected, the following steps to detect scaffolds are very similar as with the cranes. An occupancy check between every two continuous vertical lines of every group is performed, flowed by a horizontal line check over cross-sections. Here, at least three sides must have a horizontal line of at least 90% of the total width of the generated cross-section. Finally, scaffold points will be extracted, and the remainder of the point cloud will be used to search Formwork elements.

4.5 Formwork detection

The formwork detection procedure differs from the other two presented detection processes in two aspects: First, while the threshold values are very similar as for wall ROIs separation, once the ROIs with formwork are separated from the whole point cloud, they are then filtered in blobs that are aligned to the X and Y-axes. Second, in every aligned blob-point cloud, vertical cross-sections will be generated and classified with a Deep Learning (DL) algorithm, revealing the location of the formwork elements.

4.5.1 Formwork ROIs separation

Since formwork (especially foundation formwork) could be much shorter in comparison with walls, the height threshold for formwork ROIs h_{min} was set to 0.075 m. This step will basically filter out ground floor points or points that have less than 15 neighbours with the same or very similar X and Y coordinates.

Subsequently, a dilation operation is performed 6 times with an S_{R10} *rectangular structuring element*, and the minimum blob size A_{min} was set to 0.25 m^2. The reasons for these values are the same as those explained in Section 4.2.2.1 to separate the wall ROIs. However, taking into account the smaller size of formwork in comparison with walls, especially foundation formwork, A_{min} is smaller in comparison with the one used for wall separation.

4.5.2 Individual formwork recognition

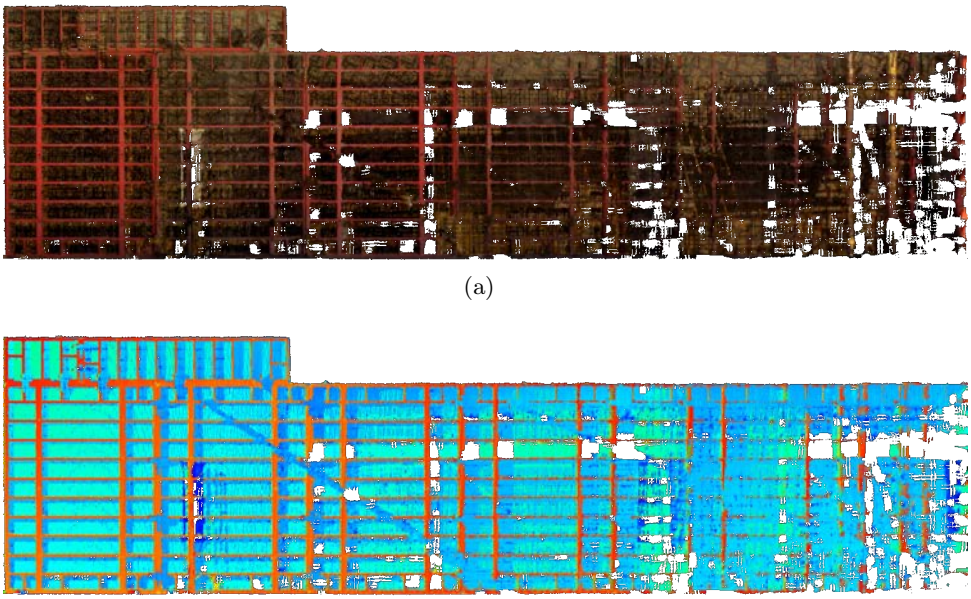
Here, opposite as with the cranes and scaffolds, the 3D line detection is not implemented right away over the formwork ROIs. This is because, since the point cloud is already rotated, it is possible to filter out formwork elements that are aligned to the building axes more efficiently.

To detect formwork elements that follow the Manhattan-Word assumption, the blobs of the formwork ROIs are further subdivided in blobs that are aligned to the X and Y-axes. These axes correspond to the building axes as the point cloud was already rotated. These blobs are shown in Figure 4.16.

(a) (b)

Figure 4.16: Horizontal and vertical formwork blobs: (a) vertical formwork blobs; (b) horizontal formwork blobs in dataset Nr. 3.

Subsequently, vertical cross-sections or facade view projections will be generated. To find the right location where these cross-sections must be created, 2D lines are detected in a vertical projection of the point cloud in every blob. For horizontal blobs, the algorithm search for the lowest and the highest horizontal lines. If the difference between them is larger than 11 cm (the standard width of formwork), then there might be a formwork element. To finally identify which blobs are formwork or not, two vertical cross-sections will be generated for every blob. One from the top, where the highest horizontal line is located. And another from the bottom where the lowest horizontal line is located. The blobs in vertical regions were previously rotated 90° to be able to treat them as horizontal ones. Over the generated cross-sections, a DL algorithm will allow the final formwork detection. Something unique about the cross-section for formwork is the fact that they will also contain depth information, this enables the DL algorithm to take into consideration not only the exterior rips of a formwork but also its interior plane surface. Figure 4.17 illustrates the difference between a cross-section with real colour an another with depth information.



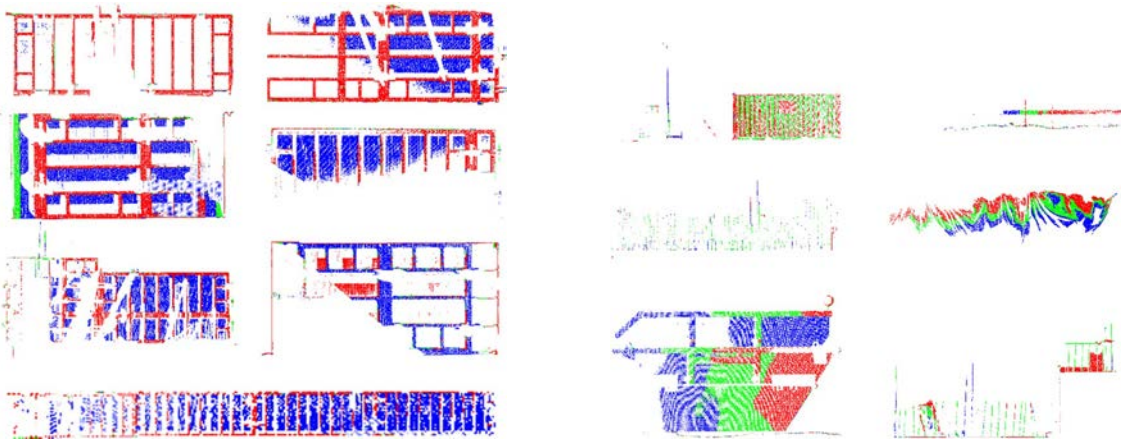
(a)

(b)

Figure 4.17: Formwork vertical cross-section: (a) with the real color; (b) with depth information.

The PyTorch C++ frontend was used to train and test the used DL algorithm. The neural network used consists of 5 convolutional layers with max-pooling and ReLU activation, and 3 fully connected layers. 244 images were used to train the model, these were generated with dataset Nr. 1, and a dataset augmentation step. Figure 4.18 shows some of these images. The data set was augmented with mirror 180° flips over the x, y and both axes, generating 3 new images form every original one. The dataset was divided into 60% training set and 40% testing set. Two Dropouts were used to prevent overfitting, one located after the convolutional layers and the other after the first fully connected layer. After 90 epochs, the algorithm achieved a maximum accuracy of 93,3% over the testing set, demonstrating to be suitable for this binary classification task. The following are the parameters used

to train the neural network: image size = 238, train batch size: 8, test batch size: 200, optimizer: Stochastic gradient descent, learning rate: 0.001 with a momentum of 0.5, loss function: negative log-likelihood.



(a) (b)

Figure 4.18: Set of formwork cross-section: (a) formwork; (b) bogus.

4.6 Parameter summary

Table 4.1 summarizes the different parameters used to filter the objects in the point cloud. Here h_{min} refers to the minimum height in meters that the object needs to have to be filtered out in the accumulated grey-scale image. S is the *structural element* and its corresponding size used in the dilation step. D_i is the number of iterations of the dilation operation. A_{min} and A_{max} are the minimum and maximum areas of the blobs to filter the objects after the dilation. Finally, l_{min} refers to the minimum length of the merged vertical lines which applies only for crane and scaffold detection.

Table 4.1: Parameter Summary

Parameter	**Cranes**	**Scaffolds**	**Formworks**	**Walls**
h_{min} (m)	0.7	0.2	0.075	1.2
S	R10x10	E5x5	R10x10	R10x10
D_i	3	6	6	5
A_{min} (m^2)	0.0075	0.002	0.25	1.5
A_{max} (m^2)	0.3	0.075	MAX	MAX
l_{min} (m)	1.5	0.4	N/A	N/A

Chapter 5

Results and Analysis

In this chapter, the results of applying the proposed methods on real datasets are presented and analysed. Firstly, the datasets, hardware and software used are discussed in Section 5.1. Then the validation results after applying the proposed method are shown as well as a detailed analysis of the relevant cases. Finally, the computational time of the process is given and analysed.

5.1 Datasets and Tools

5.1.1 Point cloud datasets

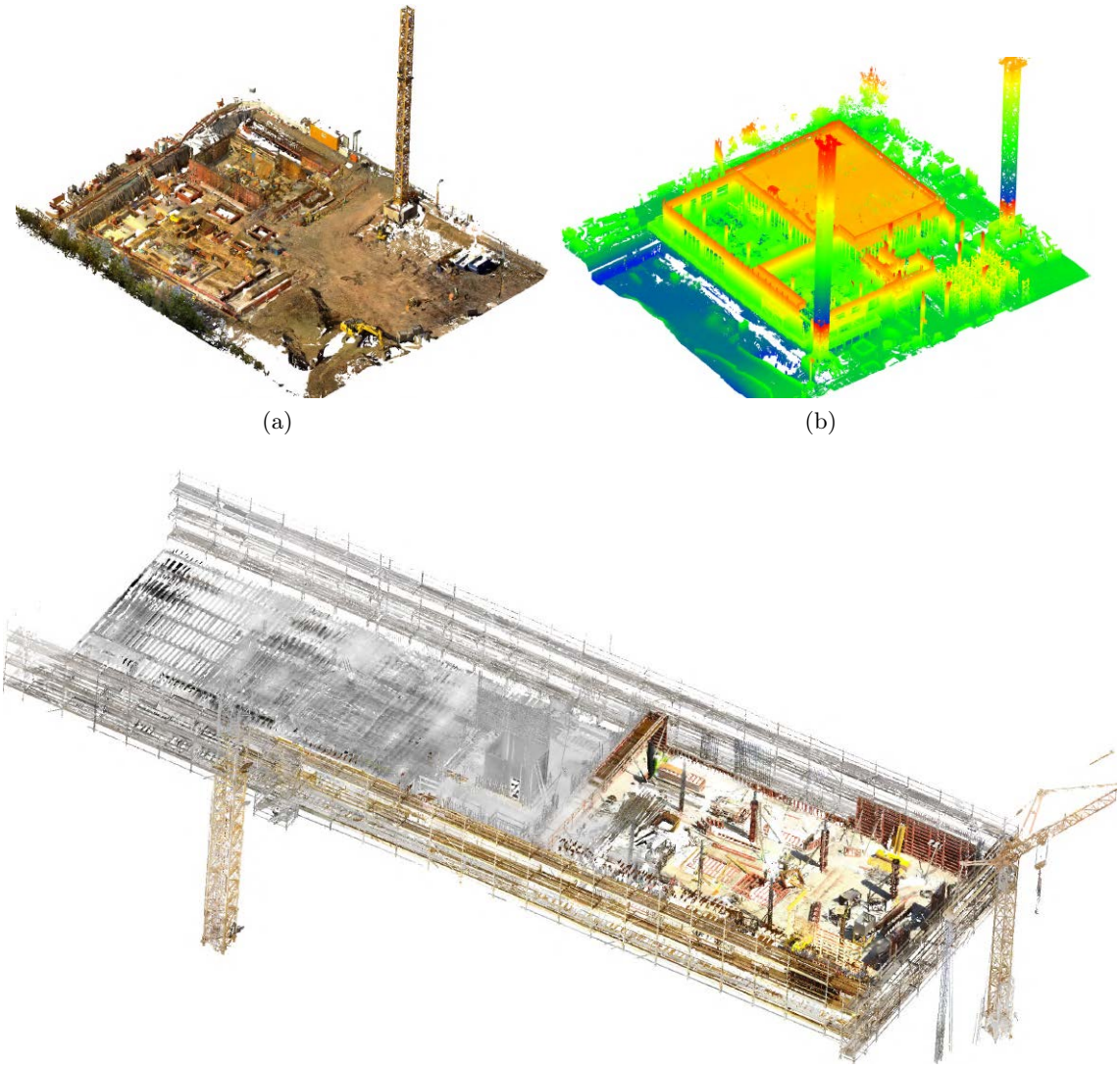
The performance of the proposed method was validated on four different laser scanner point clouds obtained from a construction site in Germany under various phases of construction. The point cloud data were acquired using a terrestrial laser scanner at several scanning locations over the entire construction site. Table 5.1 enumerates the different datasets, providing additional information about the month they were obtained, their aligned dimensions, the volume and the area they cover, and the number of points they contain. Figure 5.1 illustrates the datasets.

Table 5.1: Point cloud Datasets

Nr.	Acquisition month	Δx, Δy, Δz [m]	Volume [m^3]	Area [m^2]	Nr. of points
1	Sep. 2019	71, 58, 46	189.428	4.118	127.121.272
2	Nov. 2019	53, 60, 46	146.280	3.180	223.272.813
3	Apr. 2020	39, 78, 25	76.050	3.042	67.213.140

It is worth mentioning that dataset Nr. 3 is a smaller, manually selected region of a vast dataset. Since the original dataset has a size of more than 20 GB (in their compressed zip format), it will require a lot of computational time to work with it directly. However, the selected region is still quite large, having more than 50 million points.

Table 5.2 shows the number of points of the datasets after the downsampling step and clipping the level of interest. The minimum and maximum Z-values of the level of interest were inserted manually, to be able to cut it. In the case of a multi-storey building, for better results, these values should include neither the ceiling nor the floor of the respective level. However, if they contain only the floor, the method should still work correctly, as long as there are no objects of the story directly below.



(a) (b)

(c)

Figure 5.1: Point cloud datasets: (a) dataset Nr. 1; (b) dataset Nr. 2 (as it is originally colorless, it is shown here with height ramp colors); (c) dataset Nr. 3. (note here the lack of color at the back part of the point cloud)

Note that only the points that are in the clipped level of interest will be used to detect the location of the target objects. The automatically found angle of rotation of the point clouds are -31.3°, -53.4° and 0.2° for the respective datasets, number 1, 2, and 3.

5.1.2 Hardware

LASER SCANNER

All the point clouds used throughout this thesis were captured with a FARO Laser Scanner Focus S 350 Plus. This is a high-speed 3D terrestrial laser scanner for long-range applications. It is compact, lightweight (4.2 kg including battery) and device for indoor and outdoor environments to deliver detailed measurement and documentation with a ranging error of

Table 5.2: Point cloud Datasets

Point cloud	Number of points for each dataset		
	Nr. 1	Nr. 2	Nr. 3
Original	127.121.272	223.272.813	67.213.140
After Downsampling	64.182.855	187.654.059	62.049.852
After clipping level of interest	64.065.562	51.716.679	62.047.390

±1mm[1], a dual-axis compensator (levelling each scan with an accuracy of 19 arcsec) and a height sensor (height relative to a fixed point via an electronic barometer). For more information about this laser scanner, the reader is referred to the official Technical sheet[2] of FARO Focus Laser Scanners.

COMPUTER

The algorithms in this thesis were all developed and tested on a laptop with a 64-bit Windows 10 operating system. The specifications of the used laptop are stated as follows:

Name and brand	Dell XPS 15 9560
Processor	Intel(R) Core(TM) i7-7700HQ CPU @ 2.80 GHz
RAM	32 GB
GPU	GeForce GTX 1050

[1] According to the technical sheet the ranging error is defined as a systematic measurement error at around 10 m and 25 m.

[2] Available in https://insights.faro.com/long-range-laser-scanners/techsheet-faro-focus-laser-scanners

5.1.3 Software

The programming language C++ is used in the implementation of all the steps of the methodology. The following are the additional dependencies of the code:

- OpenCV 4.2[3] is used intensively for image processing, clustering and 2D line detection.
- PCL[4] is a powerful library to deal with point cloud processes, here is used to read and write point clouds in PLY and PCD binary format and to downsample the point cloud.
- OpenMP is used in some functions of the code to perform parallel computations.
- The 3D line detection is performed using the open-source code provided by Lu et al. (2019)[5] which in turn uses nanoflann[6] for building KD-Tree and find the nearest neighbours of every point in the point cloud.
- The PyTorch C++ Frontend[7] is used to train and load a neural network and perform inference over generated vertical cross-section projections.
- A plug-in application was developed for the FARO software SCENE, which allows the easy usage of the proposed method in this software. This plug-in requires the SCENE API App Developer Package version 2.0.21 or posterior[8].
- Fast Light Tool Kit (FLTK)[9] was used to create the Graphical User Interface (GUI) of this plug-in.

[3] https://github.com/opencv/opencv
[4] https://github.com/PointCloudLibrary/pcl
[5] https://github.com/xiaohulugo/3DLineDetection
[6] https://github.com/jlblancoc/nanoflann
[7] https://pytorch.org/cppdocs/frontend.html
[8] https://developer.faro.com/md_pages_guides_plugin_app.html
[9] https://github.com/fltk/fltk

5.2 Validation results

Figure 5.2 presents the segmentation results of the three data sets, here the cranes are visualised in green, the scaffolds in blue, and the formwork elements in red. Table 5.3 shows the validation results for every dataset, giving the precision, recall, and F_1-score for every target object, those were calculated based on the number of points on the respective segmented point cloud, and with the following formulas:

$$\text{Precision} = \frac{\text{TruePositives}}{\text{TruePositives} + \text{FalsePositives}}$$

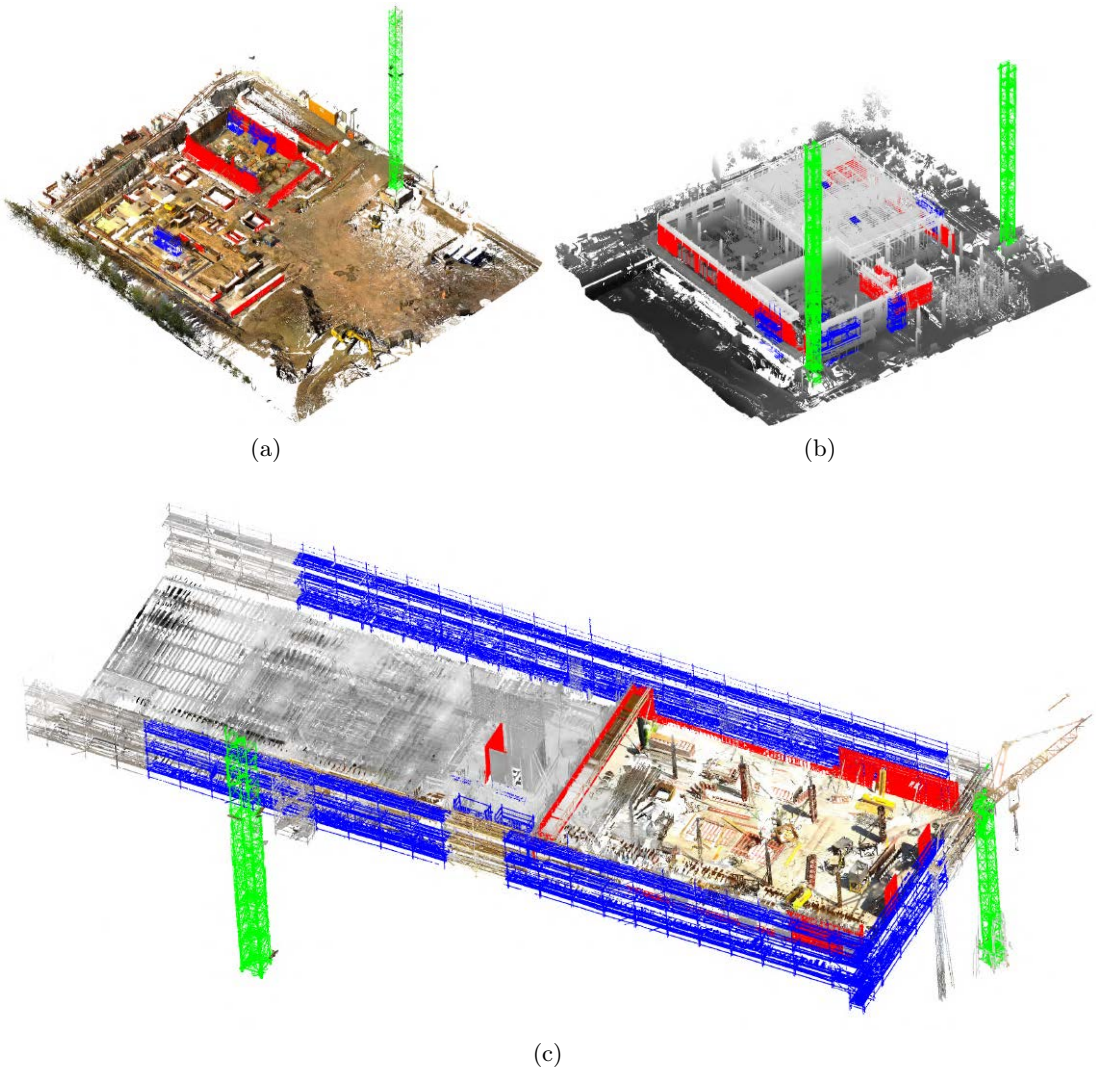
$$\text{Recall} = \frac{\text{TruePositives}}{\text{TruePositives} + \text{FalseNegatives}}$$

$$F_1 \text{ score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

It is worth mentioning that these validation results were calculated based on points, mainly because it is challenging to count separate instances of formwork elements. As later shown in figure Figure 5.8, the formwork elements that are over a single wall would be counted as a single instance. However, in this section of the wall, there are not only formwork elements. Detecting these additional points as formwork reduces the precision of the method. Such small details could not be perceived with only counting instances. Therefore using the points is a more reliable measure.

Table 5.3: Validation Results

Dataset Nr.	Object	Precision	Recall	F_1-Score
1	Crane Mast	100,0%	100,0%	100,0%
	Scaffold	100,0%	100,0%	100,0%
	Formwork	85,1%	68,1%	75,7%
2	Crane Mast	100,0%	100,0%	100,0%
	Scaffold	89,1%	95,1%	92,0%
	Formwork	36,4%	90,3%	51,9%
3	Crane Mast	100,0%	100,0%	100,0%
	Scaffold	100,0%	82,6%	90,5%
	Formwork	85,1%	100,0%	91,9%
	Overall	88,4%	92,9%	89,1%



(a) (b)

(c)

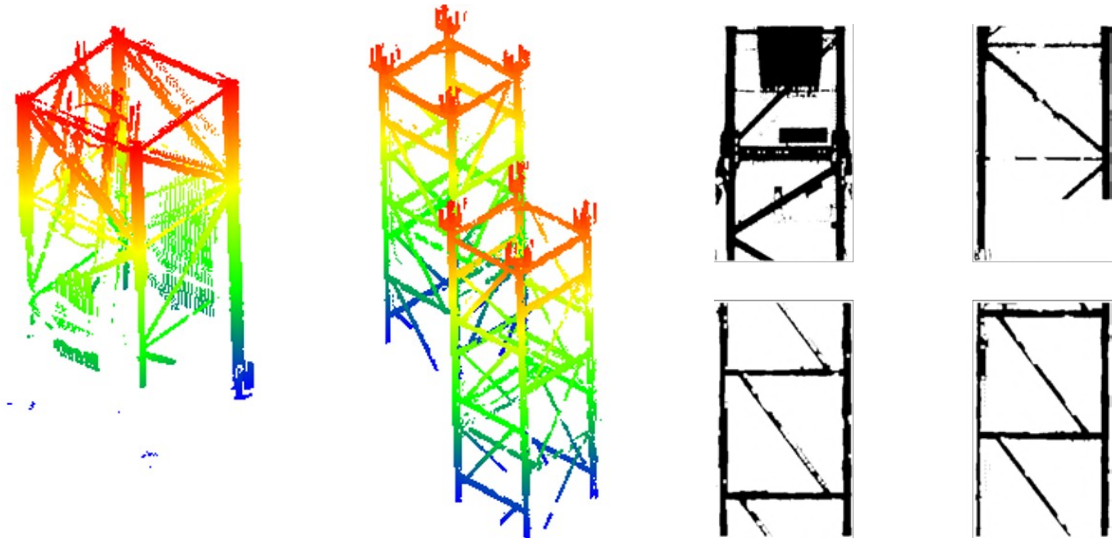
Figure 5.2: Automatically segmented Point clouds: (a) dataset Nr. 1; (b) dataset Nr. 2 (as it is originally colorless, it is shown here with height ramp gray-scale colors); (c) dataset Nr. 3. In green detected cranes, in blue detected scaffolds, and in red detected formwork elements.

5.3 Analysis

5.3.1 Crane Detection

One compelling finding while detecting cranes was that shoring elements might have very similar geometric features as cranes. Therefore it is not enough to differentiate them only by their cross-sections, as shown in Figure 5.3. A differentiation with their total height, as explained in the Chapter 4 offers a quick and effective solution for this issue. However, this solution implies the manual deletion of the jib and of the hook of the crane. Otherwise, these points could be above the regions where shoring are present and will yield to a wrong total height calculation.

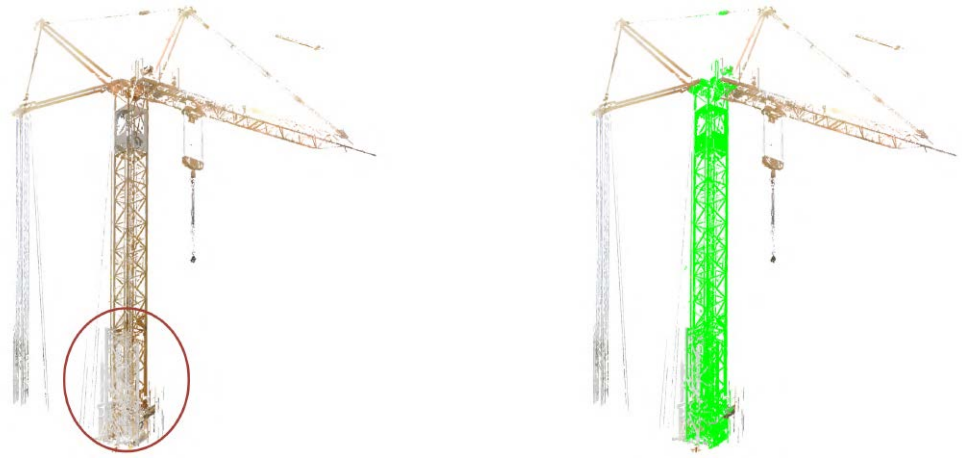
Another worth mentioning case is shown in Figure 5.4. Here we are dealing with a self-erecting crane that was moving during the point cloud acquisition. Since self-erecting cranes



(a) (b)

Figure 5.3: The geometric similarity between cross-sections of shoring and cranes: (a) the 3D point cloud of a section of one mast of a tower crane next to two shoring elements ;(b) cross-sections: the first two are cranes and the following two are shoring

move not only the jib but also the mast; the point cloud of the mast of the crane is surrounded by the noise produced by scans of the same crane in another position. Nonetheless, the crane was not moving all the time. Therefore there are still some characteristic features that allowed its correct detection.



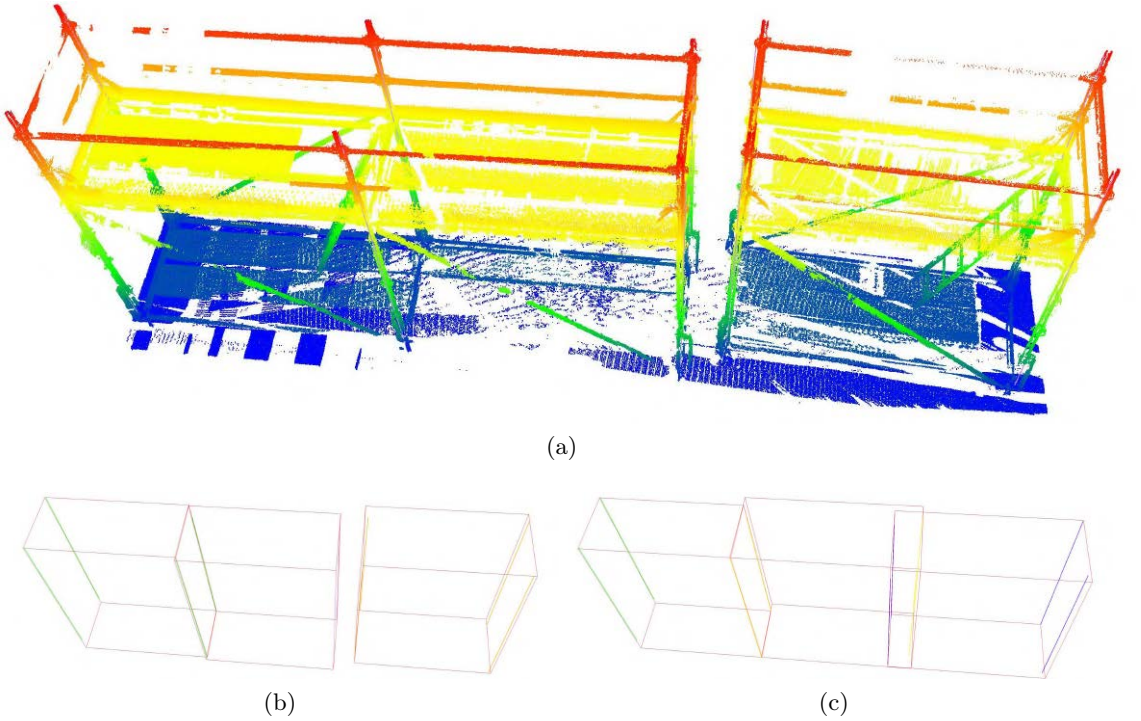
(a) (b)

Figure 5.4: Possible issue while detecting self-erecting cranes: (a) the self-erecting crane was moving during the scan acquisition, therefore there is a lot of noise in the point cloud ;(b) however, some important geometric features were still present in the point cloud, which allowed the detection of the crane.

5.3.2 Scaffold Detection

Whereas cranes are always separated enough from each other, scaffold elements are usually very close or even join together to build large scaffold systems. This fact produces interesting,

unexpected results. Figure 5.5 illustrates the case were two scaffold elements are close enough to each other to cause small incorrect detections. Instead of only detecting two scaffold instances, the algorithm will retrieve four. Two are the correct ones, and the other two are the overlapped scaffolds, as shown in Figure 5.5b. This is because even when this set of uprights do not build a scaffold, they are within the distances ranges established inFigure 4.15 and also passed the cross-sections checks.



(a)

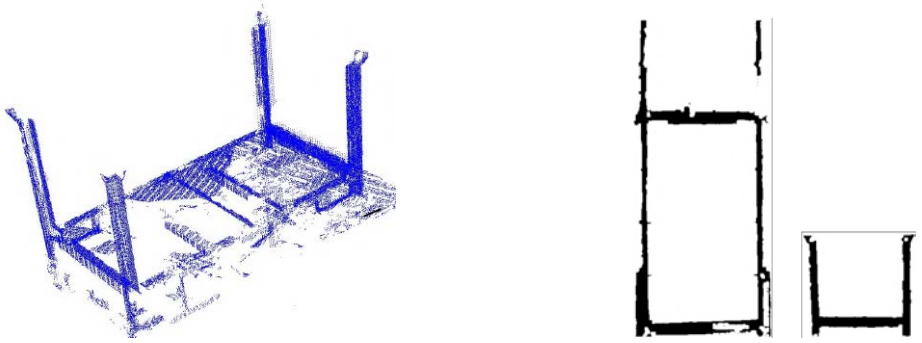
(b) (c)

Figure 5.5: Overlapped wrong detected scaffolds: (a) detected Scaffold point cloud; (b)correctly detected scaffold instances; (c) Overlapped wrong detections.

The reason why the method was not 100% precise in the dataset Nr. 2 is because the object illustrated in Figure 5.6 was wrongly detected as a scaffolding element. The item in question is a stacking pallet or container for props. There are two reasons for this false detection. First, the distances between the four vertical steel angles of this pallet are within the ranges established for scaffold detection. Second, as shown in Figure 5.6b, the cross-sections of this object have a clear horizontal line, allowing the item to pass the cross-section check for scaffolds.

A possible way to avoid this sort of false positives could be to consider also the circularity of the uprights as a requirement to detect scaffolds. Another alternative could be checking the presence of at least two horizontal lines in the cross-section. However, these two ideas would require a scaffold scanned from different locations and almost free of occlusions.

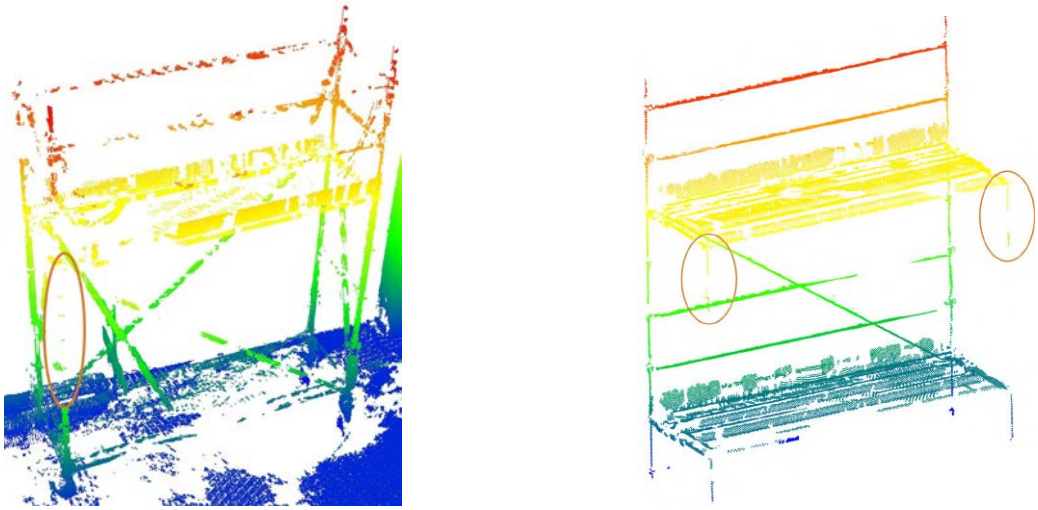
Occlusions are indeed the reason why the method did not always have perfect recall. Figure 5.7 presents the two cases where the scaffolds were not detected. Figure 5.7a shows the single not detected scaffold element present in the dataset Nr. 2. As can be visualised, the presence of horizontal occlusions, especially in one of its uprights hinder the detection of this object. In Figure 5.7b shows some of the not detected scaffolds in the dataset Nr. 3.



(a) (b)

Figure 5.6: False positive scaffold: (a) 3D view of the stacking pallet for props; (b) cross-sections of the short sides of a scaffold (left) and a stacking pallet (right). Note that both have the characteristic horizontal line. However, the pallet has only one line, while scaffolds may have at least two; as long as it was not occluded in the scan.

Here the problem is that the interior uprights (close to the facade of the building) were not scanned good enough. Without enough points of these uprights, the method will never be able to detect scaffolds, even if the other parts of the scaffolds are entirely present in the point cloud.



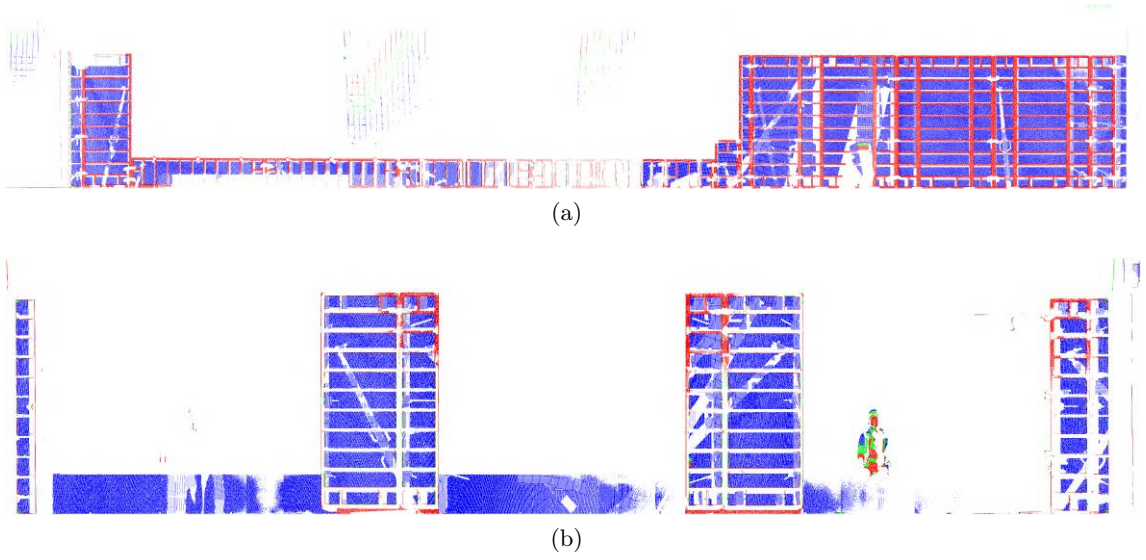
(a) (b)

Figure 5.7: True negative scaffolds: (a) non-detected scaffold in dataset Nr. 2; (b) one instances of the non-detected scaffold in dataset Nr. 3.

A possible solution for the case shown in Figure 5.7a might be to analyse also all the possible triplets of vertical lines that have the pattern of a scaffold. As shown in 3, these triplets of lines are also saved. However, this might not only be more computationally expensive but also might manifest the detection of more false-negative scaffold objects, which in turn would reduce the precision of the algorithm. Despite this, this should still be tested.

5.3.3 Formwork Detection

Regarding formwork detection, there are still three main challenges to overcome. First, is the presence of other objects next to formwork elements and in the same wall in the point cloud. Figure 5.8 illustrates this issue. Here not only formwork but also vertical reinforcement is present. However, with the proposed method, the whole section is classified as formwork. To overcome this issue, it has to be treated as an object segmentation problem and not as an image classification problem. In an object segmentation problem, instead of classifying the complete image, the location of the different objects in the image is detected. To this end would be necessary to label the exact location of the different target objects over several generated cross-sections and subsequently train a model to perform object segmentation in 2D images.



(a)

(b)

Figure 5.8: Formwork detection issue: (a) the whole section will be classified as formwork, even when in the image there is also reinforcement; (b) besides formwork elements, there are also window parapets.

The second challenge in formwork detection is the detection of formwork elements that do not follow the Manhattan -World assumption. To address this issue, one could analyse the regions where several 3D horizontal lines are present, generate a cross-section there, and classify it as well as done with the Manhattan-Wold formwork elements. The third and final challenge while detecting formwork is again the presence of occlusions. Whereas high wall formwork usually did not present many occlusions, foundation formwork was sometimes heavily occluded. Therefore, the method performed much better in dataset Nr.3 in comparison with dataset Nr.1, because in dataset Nr. 1 the foundation formwork was very occluded. This should be solvable scanning the construction site from more scanning locations.

5.4 Computational time analysis

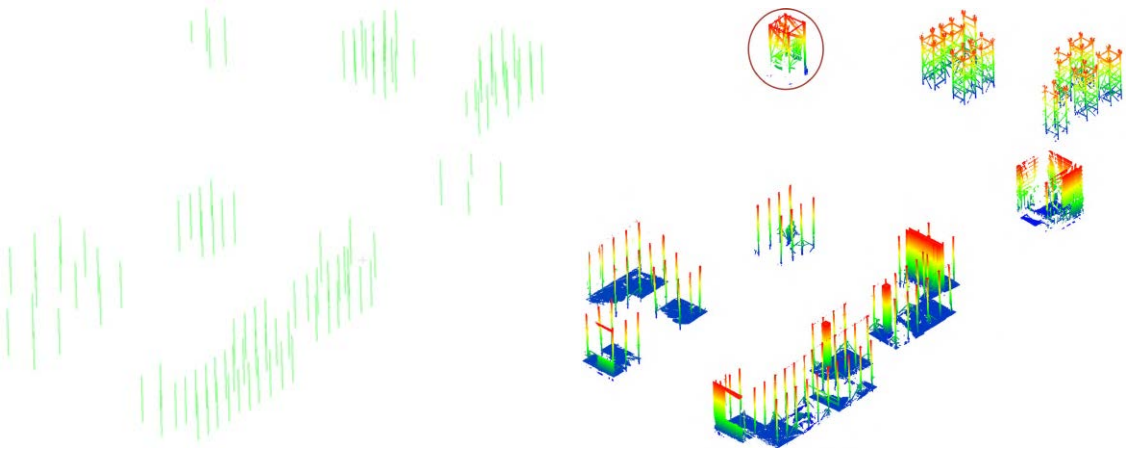
Table 5.4 presents the times in seconds for each of the steps in the methodology for each dataset.

Table 5.4: Computational time

Step	Sub-step	Computational time for each dataset [s]		
		Nr. 1	Nr. 2	Nr. 3
Preprocessing	Downsampling	35	63	14
	Clipping level of interst	7	16	6
	ROIs Separation	5	4	2
	Finding building axes	9	9	4
	Point cloud rotation	2	5	1
	Vertical projection	9	6	7
	Total Preprocessing	**67**	**103**	**34**
Crane detection	ROIs Separation	5	6	3
	3D Line detection	12	77	11
	Find pattern in lines	1	1	0
	Cross-section checks	8	*230*	29
	Height check	2	4	4
	Crane points extraction	23	63	48
	Total Crane detection	**51**	**381**	**95**
Scaffold detection	ROIs Separation	8	8	5
	3D Line detection	9	105	17
	Find pattern in lines	1	1	0
	Cross-section checks	81	*1939*	621
	Scaffold points extraction	69	192	83
	Total Scaffold detection	**168**	**2245**	**726**
Formwork detection	ROIs Separation	3	9	4
	Horizontal and Vertical ROIs	11	10	5
	Blob Point cloud classification	64	88	40
	Formwork points extraction	75	41	23
	Total Formwork detection	**153**	**148**	**72**
Total time	in seconds	439	2877	927
	in minutes	7,3	48,0	15,5

One of the main issues regarding computational time was manifested while detecting cranes and scaffolds. The problem is the presence of props that support slap formwork. This is because they are not only thin and tall as the steel angles of the mast of a tower crane, but also they can perfectly have a cuboid geometric distribution. This is illustrated in Figure 5.9.

Here more than 30 instances of possible cranes where detected, only after searching for the pattern in vertical lines. The next step implies the cross-section generation for each of the likely cranes, to perform the occupancy check and the horizontal line check. The need to perform these checks in such a large number of possible cranes is very computationally expensive. However, this is necessary to properly detect the cranes in the point cloud.



(a) (b)

Figure 5.9: Detected groups of vertical elements for cranes only using the vertical lines: (a) only detected vertical lines; (b) the corresponding point cloud inside the regions delimitated from the groups of vertical lines. Note that even when there is only a single crane, the code detected more than 30 elements with the same pattern in vertical lines as in a crane. Most of the detected elements are props (in the front of the image (b)) and shoring (in the back).

One possible solution could be the separation of indoor and outdoor spaces. Once the areas are separated, the crane and scaffold search can be performed only over the point cloud located in outdoor areas. However, this would prevent the detection of scaffold elements in indoor spaces. Another possible solution could be to search for a diagonal foot in the surroundings of every detected vertical element. Wherever this diagonal foot is present, then there is a prop for slap formwork and not a scaffold or crane element. However, not all the props have this diagonal foot, and only separating some of them might not totally solve the problem. Another, perhaps more reliable solution might be to detect the presence of ceiling about the vertical elements, neither scaffold nor cranes would be touching a roof above them. Despite that, it might be even more computationally expensive to detect and differentiate ceilings of work platforms, in comparison to only checking the presence of horizontal lines in generated cross-sections.

After manually removing the regions with props in the dataset Nr. 2. Crane detection and scaffold detection take, 68 s and 437 s, instead of 381 s and 2245 s. Needing a total time of 608 s, 1637 s (almost 30 minutes) less than with the presence of props in the point cloud.

Chapter 6

Conclusions and Further Development

This thesis aimed to design a workflow that takes a 3D laser scanner point cloud of a construction site and efficiently detects defined target objects, giving semantic meaning to the points in the point clouds. The target objects are cranes, scaffolds, and formwork.

To draw the conclusions, the research questions defined in Section 1.2 are answered. Subsequently, the choice of the methods used will be discussed. Finally, the limitations and ideas for future work will be presented.

6.1 Conclusions

To draw conclusions for this research, first the sub-questions defined in Section 1.2 are answered separately. After this, the main research question is answered.

Object segmentation

1. ***Is it possible to implement a safe and efficient prefiltering of the tentative permanent structures of a building to improve computational speed?***

   The achieved computational performance would not have been possible without a prefiltering of the point cloud. Since a laser scanner point cloud of a large construction site could have millions of points, a fast and feasible solution should try to reduce the number of points before they are processed, leaving only the ones that represent the objects of interest. To this end, a method that distinguishes regions of interest ROIs in the point cloud is first applied in the thesis. The process is based on the intrinsic geometric features of the target objects. For example, to identify possible regions with cranes or uprights of a scaffold, the method will search high and thin vertical items. On the contrary, to query the areas where formwork could be present, the process explores lower and broader regions. This search can be done over a grey-scale accumulation image of the point cloud, as explained in Chapter 4.

2. ***To what extent can the verticality, horizontality, and parallelism of the objects on a construction site be exploited for the recognition of the three target object classes?***

   Indeed, the verticality of the target objects is the most critical feature that allows its fast and accurate segmentation with the proposed workflow. The horizontality also plays an essential role in multi-storey buildings since the point cloud has to be divided into vertical sections depending on the different floor levels. Subsequently, at every

level, the usual parallelism and orthogonality of the walls enable a fast formwork search once the point cloud is rotated according to the main building axes. Evidently, the Manhattan-World assumption does not hold for every building, therefore searching walls that are not aligned to the main building axes is still necessary. However, for structural and architectural simplicity, the majority of the building floor plans follow a rectangular grid for the most part. Therefore, it seems reasonable to first take advantage of this fact before looking for walls in other directions.

Validation

1. ***How good is the proposed method compared to existing methods?***

   In comparison with the proposed method, Xu et al. (2018) have two restrictions: First, the scaffold must be next to a facade; Second, it must have a bay width of 0.8 m. Considering more possible scaffold dimensions makes the proposed method in this thesis more robust. However, it will give lower performance than Xu et al. (2018) in low-quality point clouds.

   In comparison with Wang (2019), the method is not restricted to clipping the point cloud to specific regions where the scaffold is present. Since they do not provide computational time, it is not possible to compare the computational performance reliably. Nonetheless, as their method generates horizontal slices every 0.05 m and fits circles in each of them, such a technique would undoubtedly take more time than an approach that simply projects all the points in 2.5D grey-scale accumulated images and filter the objects. However, Wang (2019) provides additional steps to detect the parts of every scaffold system and check compliance with the regulations. Such procedures are not provided in this thesis.

   A final comparison of the proposed method with S. Zeng et al. (2020) in terms of computational times lends useful insights/insightful results. According to S. Zeng et al. (2020), their process requires around 60 seconds to extract the features of a point cloud with 400,000 points. Considering the linear relationship of the time concerning the number of points, one could estimate that for a point cloud with more than 50 Million points (as the ones used in this thesis), their method would require around 125 minutes to only extract the features. This is two times more than the time that the proposed method requires to detect the three target objects in the more critical case, i.e. Dataset Nr. 2. However, the method proposed by S. Zeng et al. (2020) would be more appropriate to recognize objects with more complex geometries, which have other important geometric features different than the ones considered in this thesis.

2. ***To what extent can the proposed method assist the monitoring of the construction process?***

   As explained in the motivation (see Section 1.1), detecting the location of the formwork gives crucial information about the exact current state of the construction progress. If, for example, in the last level of a construction site, all walls are covered with formwork elements, only detecting them would give a clear statement about the actual status of

the construction site. Since usually, this is not the case, also recognizing already built structures like walls, floor, and columns would be necessary. Despite that, the best way to automatically track the planned schedule is with a 4D building information model. Once the temporary target objects are detected in the point cloud, it might be easier to automatically detect the rest of the objects. This automatic detection can be done with a reliable geometrical comparison of the rest of the point cloud with the building information model as performed by Braun and Borrmann (2019).

It is worth mentioning that to achieve a good overview of the construction progress, it is only necessary to scan the last superficial levels of the construction site together with its surroundings. This is possible because the presence of internal parts and lower levels could be inferred with a PRG as explained in Braun et al. (2020). This would not only reduce the amount of data to be processed but will also facilitate its acquisition, avoiding the need to scan every single level. However, if the goal is to document every available space of the construction site, this procedure is inevitable.

Further research

1. ***What are the bigger challenges in object recognition at different stages of the construction?***

   Three big challenges were identified while trying to detect the objects in the point cloud. First, while searching for cranes and scaffolds, the presence of props that support slab formwork force the method to generate many cross-sections. This significantly reduces computational time performance. One solution might be to firstly separate indoor and outdoor spaces and perform scaffold detection only in outdoor spaces. However, this would not allow detecting scaffold elements that are inside the building. Second, the fact that a single wall does not necessarily have formwork in its whole extension hinders a precise detection of this object. A possible solution would require the implementation of a model capable of performing object segmentation (instead of a simple image classification) over the generated cross-sections, which in turn might be useful to detect vertically placed reinforcement. Third, the presence of occlusions (more precisely the horizontal ones) makes it almost impossible to detect the objects with this method. For example, a scaffolding element with one occluded upright will not be detected, even when the rest of its parts are present in the point cloud. This case was illustrated in Figure 5.7. The best way to solve this issue is by avoiding occlusions while scanning possibly by using not only one but different scanning techniques. For example, a TLS might be very appropriate to scan the exterior part of the building, and since it is accurate even in large distances, only a few scans would be required. An MLS might be used to detect the last, more superficial levels of a multi-storey construction site, and finally, a Unmanned Aerial Vehicle (UAV) laser scanning system that scans everything from a top view could include surfaces that were not possible to scan with the other technologies due to accessibility issues.

2. ***Is it possible to achieve perfect recognition? What is needed to accomplish it?***

Perhaps the most important and most obvious factor in achieving a perfect recognition is that the object has to be scanned from different positions avoiding leaving occluded sections. During the development of this thesis, it was found that for the cranes, it is usually not a problem to have scans from different positions, since their surroundings are typically large free areas. However, two factors were identified that could hinder its recognition: First, the presence of many objects hanging on the surface of the mast of the tower crane (as lamps, posters, and banners). For example, if the selected vertical section of the crane has four banners in its four sides, the method will not detect that crane, since the prefiltering step will not allow passing this object. Nonetheless, this is not usually the case if the section is close to the base of the crane since banners are generally in a higher position where they are more visible. Second, in the cases of self-erecting cranes, if the crane was in movement during the acquisition of the scans, the resulted point cloud might be full of unwanted points that recorded the different positions of the crane mast, making almost impossible to detect the crane.

For scaffold recognition two factors were identified as well: the first factor is similar to the first one found for cranes, if scaffolds are covered by a very dense safety screen (as usual on many construction sites), the scaffolds might not be detected. This is because the safety screen might be very close to the uprights and if they are projected in a vertical projection, they will be prefiltered as wall elements since they will not be anymore thin and small. The second factor that hinders scaffold detection is the fact that they are usually only scanned from one side. Since the scaffolds are usually very close to the facade of the building, mainly only outdoor scans contribute to the acquisition of the scaffold point cloud, and as explained in a previous item, only with one occluded upright, the scaffold will not be detected.

The problem of the presence of a safety screen was also identified by Wang (2019). Perhaps their method would be more appropriate in these cases since it is based on the circularity of the uprights. Although, still the problem might be to have not occluded scanned scaffolds. This may be solvable with a TLS with several scan locations over the scaffold elements. The recently released FARO Trek offers a possible solution for navigating and scanning such environments while maintaining high accuracy. Other methods to access and scan narrow spaces are, for example, UAV-borne laser scanning systems or an MLS, which usually have, however, less accuracy than a TLS.

While wall formwork elements were usually not heavily occluded, foundation formwork was much more occluded, creating more challenges to detect them. The usage of FARO Trek would certainly overcome this issue since the scans would be done from a lower level in comparison with a traditional TLS. Additionally, to achieve better formwork recognition performance, the geometry of every possible formwork element could be incorporated. This would allow detecting the location of formwork elements that are partially occluded vertically and therefore, not totally present in the point cloud.

3. ***What semantic improvements in the point cloud are necessary to obtain greater automation in the scope of construction progress monitoring?***

To completely automate the proposed method, an additional workflow should be implemented to automatically separate the stories of a building. This method should be robust enough to detect slight changes in the height (caused by small stairs or ramps) in a single storey in large buildings, to successfully separate them.

Furthermore, an improvement in the detection of formwork elements is needed, together with the recognition of vertical and horizontal placed reinforcement. As mentioned in the motivation (see Section 1.1) detecting the location of the formwork not only gives crucial information about the exact current state of construction progress but also could be useful to evaluate the quality of the construction. To this aim an integration with a detailed 4D building information model is necessary. Besides allowing to check compliance with the schedule, an integration with a 4D building information model will allow identifying and verifying the presence of openings and essential building elements. Subsequently, and as done by M.-K. Kim et al. (2020), an automated dimensional quality assessment can also be performed, to ensure compliance with the structural plans, once the reinforcement is recognized in the point cloud.

As mentioned before, separating outdoor-indoor spaces would allow identifying faster scaffolds and cranes, and additionally, it would be an essential step to detect slab formwork.

Once all the temporary objects and the permanent structures have been detected (in the best case after a geometrical comparison with a 4D building information model), it might be possible to quantify the construction progress. For example, retrieve information about how many m^3 of concrete have been used as well as the area of slab, walls and number of columns built, along with all the work that was required to carry them out as m^2 of formwork and reinforcement placed. All this would not only support the monitoring of the progress but also would contribute to a systematic, transparent and seamless work on the construction site.

In the following, a summarized answer to the main research question is given.

How is an automatic recognition of the selected three object classes in high-quality point clouds of construction sites possible?

As a final conclusion, one could argue that as long as there is a way to infer basic geometrical constrains on the target objects, it might be possible to achieve very impressive performance on a 3D object detection problem. This not only in terms of accuracy but also in computational performance. In this thesis, the vertical orientation of the target objects, as well as its minimum height and other geometrical features, played an essential role in being able to detect them in the point cloud. Similar to genetic algorithms, the successful implementation of such a method requires careful engineering of the representation of the objects, in this case, it means a precise knowledge of the geometry of the target objects. Evidently, such a method would not be applicable to all objects (e.g. deformable objects).

Nonetheless, the process is not limited to just some given examples or non-rotational invariant constraints.

Furthermore, using 2D and 2.5D projections allows the implementation of a very efficient method to filter and subsequently detect objects in the point cloud. This enables the application of the technique on very large point clouds without compromising its accuracy. Finally, the implementation of a deep learning algorithm to classify 2.5D projections of vertical cross-sections of the point cloud proved to be very accurate for formwork classification.

6.2 Discussion

As a FARO requirement, the proposed method should not depend on color information. Therefore, the proposed approach uses only geometric information to detect the target objects in the point cloud of the construction site. However, using colour information would allow the implementation of additional methods, such as object segmentation on 2D colour images of construction sites with previously trained ML models like the one made recently available by Nath and Behzadan (2020), and re-projection over the point cloud as done by Qi et al. (2017). This could improve the detection speed. Nonetheless, the fact that the proposed method is based solely on geometric information increases the robustness of the method, allowing its implementation also in cases where colour information is not available.

The method also requires a high-quality point cloud with a resolution of around 5 millimetres or less. Small adjustments can make the algorithm usable in point clouds with higher resolutions, perhaps up to 2 cm (higher resolutions might hinder scaffold detection mostly). However, working with low-quality point clouds, such as those generated from photogrammetric methods, would require major changes. One of them would be the implementation of another method to detect vertical 3D contours in point clouds. The LSSHOT point descriptor developed by Xu et al. (2018) could be a suitable way to achieve such a method, as it was originally developed for low-quality point clouds.

Some empirical parameters were established in the proposed workflow. However, they are all general enough to allow recognition of a large variety of target objects. For example, there are some small constrains in the minimum height and dimension of the vertical projected areas of the objects that allow its fast separation of the rest of the point cloud. Similarly, the algorithms to find the vertical lines of cranes and scaffolds were not only designed within the standards distance ranges given by the respective regulations, but also with some tolerances allowing the method to include small variations of these distances. Such variations could arise in the data due to unforeseen circumstances, as is not unusual in real point clouds.

6.3 Contributions

This thesis adds some new insights into the research on construction site point cloud segmentation. More precisely, this thesis introduces the following contributions:

- A method to quickly and uniformly downsample large point clouds maintaining a low resolution.
- The applications of morphological operations over 2.5D accumulated images, produced from the projection of the point cloud, proved to be a very suitable and efficient way to filter vertical objects in the point cloud.
- It was demonstrated that it is possible to reliably and quickly find the orthogonal main axes of a building, using 2D detected lines in a prefiltered vertical projection of the point cloud.
- A detailed definition of the geometry of the target objects as they are defined by the corresponding regulations or manufacturers was provided.
- A dedicated algorithm to find specific patterns in 3D vertical detected contours proved to be useful to detect vertical objects, as long as minimal constraints on their geometry can be defined.
- The generation of 2D projections from vertical cross-sections allowed the development of a highly accurate vertical object detection method.
- The implementation of a deep learning algorithm to classified 2.5D projections of vertical cross-sections showed promising results to accurately detect formwork elements in the point cloud.

6.4 Limitations and recommendations for further development

In this thesis, it has been proven that is is possible to efficiently detect cranes, scaffolding, and formwork elements in large 3D laser scanner point clouds of construction sites. However, the method still has some shortcomings, and will therefore not work on every point cloud. Moreover, the method could be extended to detect more objects and enrich the point cloud with more semantic information. For these reasons, a summary of recommendations for future research is provided.

- A workflow could be developed to separate the stories of a building automatically. This method should be robust enough to detect slight changes in the height in a single story in large buildings caused by small stairs or ramps, to successfully separate the stories.

- One of the big challenges was the presence of props that support slab formwork and have similar geometric distribution as uprights of scaffolds or steel angles of cranes. To overcome this challenge, one possible solution might imply separating outdoor and indoor spaces. This would allow faster identification of scaffolds and cranes in outdoor spaces since props that support slab formwork are only present in indoor spaces. However, this would not allow detecting scaffold elements that are inside of the building. Additionally, the separation of spaces would be an important first step to detect slab formwork.

- More precise detection of formwork elements could be achieved with 2D object segmentation techniques. To this aim, an appropriate dataset of labelled images has to be generated, and a model has to be trained. Besides that, the recognition of formwork elements that do not follow the Manhattan-word assumption is still missing. This could be possibly done by applying the 3D contour detector in the rest of the unclassified point cloud after detecting the Manhattan-word formworks. Locations, where a certain number of horizontal contours are detected, could be further analyzed similarly as it is done with the other formwork elements.

- After detecting formwork elements, the recognition of vertical and horizontal placed reinforcement would complete the main set of not permanently-visible objects that determine the current state of the construction progress.

- To achieve a fully automated construction monitoring, it is required the integration with a detailed 4D building information model containing the geometry and time information of the permanent structures, as done by Braun et al. (2020). This would also enable the identification and verification of the presence of openings and important building elements in the right location on the construction site.

- Subsequently, and as done by M.-K. Kim et al. (2020), an automated dimensional quality assessment can also be performed, to ensure compliance with the structural plans.

- Safety regulations can be also be verified in cranes and scaffold elements, for the latter Wang (2019) already proposed a method that requires the detection of every single element of the scaffolds, such as guard-rails, toe-boars, and working platforms.

- A major, constrain with Terrestrial Laser Scanner (TLS) is the low mobility and flexibility of setting up the scanner at several locations, which is crucial to avoid occlusions. Although some new techniques like the FARO Trek might contribute to overcoming this challenge, it is necessary to investigate also other point cloud data acquisition techniques with higher flexibility and mobility to achieve less occluded point clouds. Other techniques include photogrammetry with images taken from cameras mounted on a UAV (as Braun et al. (2020) proved to be a suitable method for construction progress monitoring), UAV-borne laser scanning system, and Mobile Laser Scanner (MLS).

- The methods developed in this thesis could be speed-up with the parallelization of some functions, for example, the generation of the 2.5D accumulated images, as well as the 2D line detection. OpenCV offers not only CPU but also GPU parallel computations that could make a significant performance improvement.

- Finally, it would be useful to generate a navigation graph of a construction site. This with the aim of ensuring that there are always safe and efficient paths for mobility on the construction site. However, this would require detecting ladders and all possible paths in a scaffold system, which are usually not easily visible. Therefore, this is still a very challenging task.

References

Amer, F., & Golparvar-Fard, M. (2018). Decentralized visual 3d mapping of scattered work locations for high-frequency tracking of indoor construction activities. In *Construction research congress 2018*. https://doi.org/10.1061/9780784481264.048

Armstrong, G., & Gilge, C. (2017). Global construction survey: Make it, or break it–reimagining governance, people and technology in the construction industry.

Bentley, J. L. (1975). Multidimensional binary search trees used for associative searching. *Communications of the ACM*, *18*(9), 509–517. https://doi.org/10.1145/361002.361007

Blanco, J. L., & Rai, P. K. (2014). Nanoflann: A c++ header-only fork of flann, a library for nearest neighbor (nn) wih kd-trees.

Blaszczak-Bak, W. (2016). New optimum dataset method in LiDAR processing. *Acta Geodynamica et Geomaterialia*, 381–388. https://doi.org/10.13168/agg.2016.0020

Bosché, F., & Haas, C. (2008). Automated retrieval of 3d CAD model objects in construction range images. *Automation in Construction*, *17*(4), 499–512. https://doi.org/10.1016/j.autcon.2007.09.001

Bosché, F. (2010). Automated recognition of 3d CAD model objects in laser scans and calculation of as-built dimensions for dimensional compliance control in construction. *Advanced Engineering Informatics*, *24*(1), 107–118. https://doi.org/10.1016/j.aei.2009.08.006

Bosché, F. (2012). Plane-based registration of construction laser scans with 3d/4d building models. *Advanced Engineering Informatics*, *26*(1), 90–102. https://doi.org/10.1016/j.aei.2011.08.009

Bosché, F., Ahmed, M., Turkan, Y., Haas, C. T., & Haas, R. (2015). The value of integrating scan-to-BIM and scan-vs-BIM techniques for construction monitoring using laser scanning and BIM: The case of cylindrical MEP components. *Automation in Construction*, *49*, 201–213. https://doi.org/10.1016/j.autcon.2014.05.014

Böttcher, P. D., & Neuenhagen, H. (1997). *Baustelleneinrichtung: Betriebliche organisation, geräte, kosten, checklisten*. Bau-Verlag.

Braun, A., Tuttas, S., Stilla, U., Borrmann, A., & Center, L. O. (2016). Incorporating knowledge on construction methods into automated progress monitoring techniques. https://publications.cms.bgu.tum.de/2016BraunEGICE.pdf

Braun, A., & Borrmann, A. (2019). Combining inverse photogrammetry and BIM for automated labeling of construction site images for machine learning. *Automation in Construction*, *106*, 102879. https://doi.org/10.1016/j.autcon.2019.102879

Braun, A., Tuttas, S., Borrmann, A., & Stilla, U. (2018). Bim-based progress monitoring. In A. Borrmann, M. König, C. Koch, & J. Beetz (Eds.), *Building information modeling*. Springer. https://link.springer.com/chapter/10.1007/978-3-319-92862-3_28

Braun, A., Tuttas, S., Borrmann, A., & Stilla, U. (2020). Improving progress monitoring by fusing point clouds, semantic data and computer vision. *Automation in Construction*, *116*, 103210. https://doi.org/10.1016/j.autcon.2020.103210

Breu, T. (2019). *Point cloud classification as a support for bim-based progress tracking* (Master's thesis). Technische Universität München. https://publications.cms.bgu.tum.de/theses/2019_breu_braun.pdf

Cunningham, P., & Delany, S. J. (2020). K-nearest neighbour classifiers: 2nd edition (with python examples)arXiv 2004.04523v2.

Eickeler, F., & Borrmann, A. (2018). Adaptive feature-conserving compression for large scale point clouds. In *Eg-ice*. https://publications.cms.bgu.tum.de/2019_Adaptive_feature-conserving_compression_for_large_scale_point_clouds_EG-ICE2019.pdf

Fichtner, F. (2016). *Semantic enrichment of a point cloud based on an octree for multi-storey pathfinding* (Master's thesis). TU Delft. https://doi.org/https://doi.org/10.1111/tgis.12308

Golparvar-Fard, M., Pena-Mora, F., & Savarese, S. (2011). Monitoring changes of 3d building elements from unordered photo collections. In *2011 IEEE international conference on computer vision workshops (ICCV workshops)*. https://doi.org/10.1109/iccvw.2011.6130250

Golparvar-Fard, M., Peña-Mora, F., & Savarese, S. (2015). Automated progress monitoring using unordered daily construction photographs and IFC-based building information models. *Journal of Computing in Civil Engineering*, *29*(1), 04014025. https://doi.org/10.1061/(asce)cp.1943-5487.0000205

Han, K., Degol, J., & Golparvar-Fard, M. (2018). Geometry- and appearance-based reasoning of construction progress monitoring. *Journal of Construction Engineering and Management*, *144*(2), 04017110. https://doi.org/10.1061/(asce)co.1943-7862.0001428

Han, X.-F., Jin, J. S., Wang, M.-J., Jiang, W., Gao, L., & Xiao, L. (2017). A review of algorithms for filtering the 3d point cloud. *Signal Processing: Image Communication*, *57*, 103–112. https://doi.org/10.1016/j.image.2017.05.009

Kim, C., Son, H., & Kim, C. (2013). Automated construction progress measurement using a 4d building information model and 3d data. *Automation in Construction*, *31*, 75–82. https://doi.org/10.1016/j.autcon.2012.11.041

Kim, M.-K., Thedja, J. P. P., & Wang, Q. (2020). Automated dimensional quality assessment for formwork and rebar of reinforced concrete components using 3d point cloud data. *Automation in Construction*, *112*, 103077. https://doi.org/10.1016/j.autcon.2020.103077

Kropp, C., Koch, C., & König, M. (2018). Interior construction state recognition with 4d BIM registered image sequences. *Automation in Construction*, *86*, 11–32. https://doi.org/10.1016/j.autcon.2017.10.027

Lin, J. J., & Golparvar-Fard, M. (2020). Construction progress monitoring using cyber-physical systems. In *Cyber-physical systems in the built environment* (pp. 63–87). Springer. https://www.springer.com/gp/book/9783030415594

Lu, X., Liu, Y., & Li, K. (2019). Fast 3d line segment detection from unorganized point cloud. *arXiv preprint arXiv:1901.02532*. arXiv:1901.02532

Maalek, R., Lichti, D. D., & Ruwanpura, J. Y. (2019). Automatic recognition of common structural elements from point clouds for automated progress monitoring and dimensional quality control in reinforced concrete construction. *Remote Sensing*, *11*(9), 1102. https://doi.org/10.3390/rs11091102

Mace, B., & Jones, S. (2016). *How satisfied, really satisfied, are owners?*

Macher, H., Landes, T., & Grussenmeyer, P. (2017). From point clouds to building information models: 3d semi-automatic reconstruction of indoors of existing buildings. *Applied Sciences*, *7*(10), 1030. https://doi.org/10.3390/app7101030

Nath, N. D., & Behzadan, A. H. (2020). Deep convolutional networks for construction object detection under different visual conditions. *Frontiers in Built Environment*, *6*. https://doi.org/10.3389/fbuil.2020.00097

Nikoohemat, S., Diakité, A. A., Zlatanova, S., & Vosselman, G. (2020). Indoor 3d reconstruction from point clouds for optimal routing in complex buildings to support disaster management. *Automation in Construction*, *113*, 103109. https://doi.org/https://doi.org/10.1016/j.autcon.2020.103109

Norman-Spencer. (2012). *Types of cranes*. http://www.norman-spencer.com/wp-content/uploads/2018/04/Types_of_Cranes.pdf

Orts-Escolano, S., Garcia-Rodriguez, J., Morell, V., Cazorla, M., Perez, J. A. S., & Garcia-Garcia, A. (2015). 3d surface reconstruction of noisy point clouds using growing neural gas: 3d object/scene reconstruction. *Neural Processing Letters*, *43*(2), 401–423. https://doi.org/10.1007/s11063-015-9421-x

PERI. (2018). Dominodie kompaktere wandschalungfür vielfältige einsätze im hoch- und tiefbau. https://www.peri.de/produkte/schalungssysteme/wandschalungen/domino-wandschalung.html

Qi, C. R., Liu, W., Wu, C., Su, H., & Guibas, L. J. (2017). Frustum pointnets for 3d object detection from rgb-d dataarXiv 1711.08488v2.

Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2016). Pointnet: Deep learning on point sets for 3d classification and segmentationarXiv 1612.00593v2.

Rakotosaona, M.-J., Barbera, V. L., Guerrero, P., Mitra, N. J., & Ovsjanikov, M. (2019). PointCleanNet : Learning to denoise and remove outliers from dense point clouds. *Computer Graphics Forum*, *39*(1), 185–203. https://doi.org/10.1111/cgf.13753

Rusu, R. B., & Cousins, S. (2011). 3d is here: Point cloud library (PCL). In *2011 IEEE international conference on robotics and automation*. https://doi.org/10.1109/icra.2011.5980567

Schach, R., & Otto, J. (2017). *Baustelleneinrichtung*. Springer Fachmedien Wiesbaden. https://doi.org/10.1007/978-3-658-16066-1

Simons, R. J. (2016). *The optimization of crane deployment by using bim: Development of a tool, based on knowledge and experience within the field of residential construction projects* (Master's thesis). Eindhoven University of Technology. https://research.tue.nl/en/studentTheses/the-optimization-of-crane-deployment-by-using-bim

Son, H., Kim, C., & Cho, Y. K. (2017). Automated schedule updates using as-built data and a 4d building information model. *Journal of Management in Engineering*, *33*(4), 04017012. https://doi.org/10.1061/(asce)me.1943-5479.0000528

Stromberger, D. (2012). *Einfluss des kranstandortest auf die produktivitäat in baubetrieb.pdf* (Master's thesis). TU Graz. https://diglib.tugraz.at/download.php?id=576a76751e16e&location=browse

Suchocki, C., & Błaszczak-Bąk, W. (2019). Down-sampling of point clouds for the technical diagnostics of buildings and structures. *Geosciences*, *9*(2), 70. https://doi.org/10.3390/geosciences9020070

Sun, J. (2020). *3d point cloud object detection based on 2d objectdetection* (Master's thesis). University of Mannheim.

Turkan, Y., Bosche, F., Haas, C. T., & Haas, R. (2012). Automated progress tracking using 4d schedule and 3d sensing technologies. *Automation in Construction*, *22*, 414–421. https://doi.org/10.1016/j.autcon.2011.10.003

Turkan, Y., Bosché, F., Haas, C. T., & Haas, R. (2013). Toward automated earned value tracking using 3d imaging tools. *Journal of Construction Engineering and Management*, *139*(4), 423–433. https://doi.org/10.1061/(asce)co.1943-7862.0000629

Turkan, Y., Bosché, F., Haas, C. T., & Haas, R. (2014). Tracking of secondary and temporary objects in structural concrete work. *Construction Innovation*, *14*(2), 145–167. https://doi.org/10.1108/ci-12-2012-0063

Vermeulen, J. L., Hillebrand, A., & Geraerts, R. (2017). A comparative study of k-nearest neighbour techniques in crowd simulation. *Computer Animation and Virtual Worlds*, *28*(3-4), e1775. https://doi.org/10.1002/cav.1775

Wang, Q. (2019). Automatic checks from 3d point cloud data for safety regulation compliance for scaffold work platforms. *Automation in Construction*, *104*, 38–51. https://doi.org/10.1016/j.autcon.2019.04.008

Wang, Q., & Kim, M.-K. (2019). Applications of 3d point cloud data in the construction industry: A fifteen-year review from 2004 to 2018. *Advanced Engineering Informatics*, *39*, 306–319. https://doi.org/10.1016/j.aei.2019.02.007

Wu, C. (2013). Towards linear-time incremental structure from motion. In *2013 international conference on 3d vision*. https://doi.org/10.1109/3dv.2013.25

Xu, Y. (2019). *Reconstruction of building objects from point clouds of built environment and construction sites* (Doctoral dissertation). Technische Universität München. https://www.pf.bgu.tum.de/pub/2019/xu_phd19_dis.pdf

Xu, Y., Tuttas, S., Hoegner, L., & Stilla, U. (2018). Reconstruction of scaffolds from a photogrammetric point cloud of construction sites using a novel 3d local feature descriptor. *Automation in Construction*, *85*, 76–95. https://doi.org/10.1016/j.autcon.2017.09.014

Zeng, J., Cheung, G., Ng, M., Pang, J., & Yang, C. (2020). 3d point cloud denoising using graph laplacian regularization of a low dimensional manifold model. *IEEE Transactions on Image Processing*, *29*, 3474–3489. https://doi.org/10.1109/tip.2019.2961429

Zeng, S., Chen, J., & Cho, Y. K. (2020). User exemplar-based building element retrieval from raw point clouds using deep point-level features. *Automation in Construction*, *114*, 103159. https://doi.org/https://doi.org/10.1016/j.autcon.2020.103159

Chapter 7

Declaration

I hereby affirm that I have independently written the thesis submitted by me and have not used any sources or aids other than those indicated.

Location, Date, Signature