Automated Generation of Geometric Digital Twins of Existing Reinforced Concrete Bridges

Ruodan Lu

Emmanuel College

Department of Engineering
University of Cambridge

This dissertation is submitted for the degree of

Doctor of Philosophy

September 2018

To Liang and my parents.

Declaration

This dissertation is the result of my own work and includes nothing, which is the outcome of work done in collaboration except where specifically indicated in the text. It has not been previously submitted, in part or whole, to any university of institution for any degree, diploma, or other qualification.

In accordance with the Department of Engineering guidelines, this thesis is does not exceed 65,000 words, and it contains less than 150 figures.

Signed:

Date: 07/02/2019

Ruodan Lu
Cambridge

Abstract

The cost and effort of modelling existing bridges from point clouds currently outweighs the perceived benefits of the resulting model. The time required for generating a geometric Bridge Information Model, a holistic data model which has recently become known as a "Digital Twin", of an existing bridge from Point Cloud Data is roughly ten times greater than laser scanning it. There is a pressing need to automate this process. This is particularly true for the highway infrastructure sector because Bridge Digital Twin Generation is an efficient means for documenting bridge condition data. Based on a two-year inspection cycle, there is a need for at least 315,000 bridge inspections per annum across the United States and the United Kingdom. This explains why there is a huge market demand for less labour-intensive bridge documentation techniques that can efficiently boost bridge management productivity.

Previous research has achieved the automatic generation of surface primitives combined with rule-based classification to create labelled cuboids and cylinders from point clouds. While existing methods work well in synthetic datasets or simplified cases, they encounter huge challenges when dealing with real-world bridge point clouds, which are often unevenly distributed and suffer from occlusions. In addition, real bridge topology is much more complicated than idealized cases. Real bridge geometries are defined with curved horizontal alignments, and varying vertical elevations and cross-sections. These characteristics increase the modelling difficulties, which is why none of the existing methods can handle reliably.

The **objective** of this PhD research is to devise, implement, and benchmark a novel framework that can reasonably generate labelled geometric object models of constructed bridges comprising concrete elements in an established data format (i.e. Industry Foundation Classes). This objective is achieved by answering the following **research questions**: (1) how to effectively detect reinforced concrete bridge components in Point Cloud Data? And (2) how to effectively fit 3D solid models in the format of Industry Foundation Classes to the detected point clusters?

The proposed framework employs bridge engineering knowledge that mimics the intelligence of human modellers to detect and model reinforced concrete bridge objects in point clouds. This framework directly extracts structural bridge components and then models them without generating low-level shape primitives. Experimental results suggest that the proposed framework can perform quickly and reliably with complex and incomplete real-world bridge point clouds featuring occlusions and unevenly distributed points. The results of experiments on ten real-world bridge point clouds indicate that the framework achieves an overall micro-average detection F1-score of 98.4%, an average modelling accuracy of $\overline{\mathrm{C2C}}_{\mathrm{Auto}}$ 7.05 cm, and the average modelling time of merely 37.8 seconds. Compared to the laborious and time-consuming manual practice, the proposed framework can realize a direct time-savings of 95.8%. This is the first framework of its kind to achieve such high and reliable performance of geometric digital twin generation of existing bridges.

Contributions. This PhD research provides the unprecedented ability to rapidly model geometric bridge concrete elements, based on quantitative measurements. This is a huge leap over the current practice of Bridge Digital Twin Generation, which performs this operation manually. The presented research activities will create the foundations for generating meaningful digital twins of existing bridges that can be used over the whole lifecycle of a bridge. As a result, the knowledge created in this PhD research will enable the future development of novel, automated applications for real-time condition assessment and retrofit engineering.

Keywords: Digital Twin, Bridge, Point Cloud Data, IFC

Acknowledgements

First and foremost, I would like to express my earnest appreciation to my supervisor Doctor Ioannis Brilakis for all his contributions of guidance to make my Ph.D. experience stimulating. At many stages in the course of this research project, I benefited from his advice, particularly so when sharing of his exceptional scientific experience as well as raising the research visibility. I am very grateful to Professor Campbell Middleton. His careful review and advice contributed enormously to the production of my research paper. Thanks also go to Professor Rafael Sacks for his support on my work throughout the SeeBridge project. I would also like to thank Professor Carl Rasmussen for his help during my first year at Cambridge.

I acknowledge EPSRC, EU Infravation SeeBridge project, and Trimble Research Fund for sponsoring my PhD.

I consider myself very fortunate indeed to have had the opportunity to pursue this research work toward Ph.D. at University of Cambridge. Thanks for all the bright and brilliant people surround me who spur me to excel. Special thanks are due to my lab mates (Stephania, Marianna, Ioannis, Bella, Eirini, Steve, Philipp, Eva, Maria, Sevde, and Mahendrini) for their discussions and practical help. In particular, I would like to thank Bella Nguyen for sharing her experience and offering help on public presentation. To Philipp Hüthwohl, I want to thank him for sharing his research materials and collaborating with me on SeeBridge project as well as many award applications. Thanks go to Steve Vick for helping me with grammar problems and reviewing my paper in the early stage of my PhD.

I express my gratitude to my Chinese friends. Rui, Liang, and Jingwen, thank you for being my lovely roommates during my last year at Cambridge. Forever Norwich 8! Qianchen, Zhonglu, Jingtao, Cong, Zengle and many others, thank you very much for your precious friendship and accompany.

I submit my deepest thanks to my mother and father, who always remain at my back to support me, for their unconditional love.

Last, I would like to deliver my best respects to Professor Liang Guo. It would have impossible to complete this PhD without his support. He is not only my beloved life partner but also my mentor, my strongest and most loyal career partner.

Journal publications from data presented in thesis

Lu, R., Brilakis, I., & Middleton, C. (2018). Detection of Structural Components in Point Clouds of Existing RC Bridges. In Journal of Computer-Aided Civil and Infrastructure Engineering, 0(0). 2018. (Impact factor: 5.475)

Conference publications from data presented in thesis

Lu, R., Brilakis, I., & Middleton, C. (2018). Detection of Key Components of Existing Bridge in Point Cloud Datasets. 17th International Conference on Computing in Civil and Building Engineering, 5-7 June 2018, Tampere, Finland.

Lu, R. & Brilakis, I. (2018). Recursive Segmentation for as-is Bridge Information Modelling. In 2nd Lean Construction (LC3) conference. 4 – 12 July 2017. Heraklion, Crete, Greece

Awards obtained during PhD candidature

[1] IET Award (out of 350 entries) – Category: Information Technology | 2018
The Institution of Engineering and Technology (IET), United Kingdom

[2] 1st Prize (out of 54 teams) - 11th ICCEM (Innovation Competition in Construction Engineering and Management) | 2018
Tsinghua University, China

[3] 1st Prize – The First British-Chinese High-level Talent Business Plan Competition – Category: Electronic Information and Artificial Intelligence (the only female winner, out of 150 entries) | 2018
Chinese Students and Scholars Association (CSSA), United Kingdom

[4] FIATECH CETI AWARD 2017 – Category: Outstanding Student Research Project (the only winner in this category) | 2018
Construction Industry Institute, United States

[5] 1st Prize – 2018 Young Innovator and Entrepreneur Competition | 2018
European-Chinese Association & European Times Culture Media Group, France

[6] Best Poster – RISE Award – Category: Design, Innovation and Creativity | 2018
Leeds Beckett University, SEEDS Conference, United Kingdom

[7] 4th Prize – Tongji Innovation Award | 2018
Tongji University, China-France Students Associations, France

[8] 1st Prize – LC3 Construction Innovation Competition | 2017
Heraklion, Greece

Contents

List of Tables

List of Figures

List of Abbreviations and Acronyms

AASHTO	Association of State Highway and Transportation Officials
AEC	Architecture, Engineering, and Construction
ASCE	American Society of Civil Engineers
BMS	Bridge Management Systems
BrIM	Bridge Information Modelling
BIM	Building Information Modelling
CAD	Computer Aided Design
CSG	Constructive Solid Geometry
DoT	Department of Transportation
DMRB	Design Manual for Roads and Bridges
DT	Digital Twin
FHWA	Federal Highway Administration
FM	Facility Management
gDT	Geometric Digital Twin
HA	Highways Act
IFC	Industry Foundation Classes
LOD	Level of Detail
LS	Laser Scanning
MVD	Model View Definition
NAO	National Audit Office
PCD	Point Cloud Data
RC	Reinforced Concrete
SVD	Singular Value Decomposition
TLS	Terrestrial Laser Scanning
UML	Unified Modelling Language
US	United States
UK	United Kingdom
XML	Extensible Markup Language

1. INTRODUCTION

The main objective of this research is to automate the generation of geometric digital twins of existing reinforced concrete bridges as stated in the thesis title. The author starts first by breaking down the title into its parts and defining each one in detail, to ensure that the reader has a precise understanding of the main objective.

The author defines a **Digital Twin (DT)** to be a digital replica of a real-world asset. The asset could be a tunnel, a building, a bridge, or any other man-made asset of the built environment. A DT differs from traditional computer-aided design (CAD), nor does it serve as merely an Internet of Things (IoT) solution. It could be much more than either. A DT is based on massive, cumulative, real-time, real-world data measurements across an array of dimensions (Buckley & Logan, 2017), and consequent use of a digital model across the entire lifecycle of an infrastructure. The model comprises both three-dimensional (3D) geometry of the infrastructure components as well as a comprehensive set of semantic information, including material, functions, and relationships between the components. The author uses the adjective **Geometric (gDT)** to highlight that this thesis mainly focuses on the geometric representation of the DT. This includes the semantic meanings and shapes of the components constituting the asset. However, information such as their texture, material, and damages are not included. **Generation** refers to the process of producing a gDT from raw Point Cloud Data (PCD). The term **Automated** is used to explain that this research aims to drastically reduce the manual labour needed to generate a gDT when compared to the existing manual methods. Note that this differs from “automatic”, which means no manual labour necessary. Some human assistance will still be required in the author’s case. **Existing** refers to bridges that are already built. This differs from generating the gDT of an asset in the design phase or during construction. **Reinforced Concrete (RC)** Bridges refers to the typical slab and beam & slab bridges where both slabs, beams, and other components are made of RC. They are commonly found in highway over- or under-passes and small river crossings. This thesis only focuses on these bridge types as they constitute the vast majority of bridges in the road network. The author also only focuses on the most important and highly detectable part of the bridges (specifically elements deck slab, pier, pier cap, and girder), as modelling the underground, underwater or less detectable components will require substantially different data collection solutions that are beyond the scope of this work.

The author provides additional clarity of the main objective of this thesis by defining the input and output of the framework, the technical objectives and research questions, and the contributions achieved. The **input** of the proposed framework is a PCD of an existing RC bridge. The **output** is its gDT in a specific data format, i.e. the open construction industry data standard, Industry Foundation Classes (IFC). It is a data exchange format which makes it possible to transport infrastructure data between software platforms. It ensures a uniform, unequivocal description of the geometric and semantic information so that they are clear in their meanings. The technical **objectives** of this research are (1) automate the detection of the most important above ground bridge RC components in PCD, and (2) automate the fitting of IFC objects to the detected point clusters of bridge RC components. These two objectives are achieved by answering the following main **research questions**: (1) how to efficiently and reliably cluster and classify a bridge PCD into labelled point-clusters corresponding to the structural RC components of the bridge, and (2) how to effectively extract the geometric features of each point-cluster and model them to their corresponding IFC standards. **Main contributions.** This research is the first of its kind to achieve robust detection performance for four key component types (i.e. deck slab, pier, pier cap, and girder) in real-world RC bridge PCD and to cost-effectively generate their corresponding IFC objects with high modelling accuracy. The proposed framework has the potential to be extended to adapt to other bridge types by adding additional technical layers. It provides foundations for other researchers to build upon to integrate the framework in bridge management systems (BMS) currently used in practice so that BMS databases could be enriched with information that is more accurate, more detailed, and more accessible. The following section provides a clear explanation of the immense value of DT for bridges as justification for the need to devise methods that generate the DT's basic form: the gDT.

1.1. Bridge Digital Twin: What and Why?

Bridges are fundamental to a nation’s transport infrastructure network. According to section 41 of the Highways Act, 1980 (HA, 1980), the highway authority has a duty to manage and maintain the majority of bridges at public expense, unless it can prove that someone else is responsible. Therefore, it is crucial that bridge management minimizes disruption, risk and consequential costs to road users and makes economic and efficient use of resources (FHWA, 2012b).

Unfortunately, bridge collapses keep happening and cause casualties (Figure 1.1). Bridge failures are rarely an instantaneous event, although they happen in a moment. The sequence of events that sometimes lead to a bridge collapse often begins with improper design, construction, and use. Every time a bridge collapse occurs, we are reawakened to the fact that our nation’s most iconic structures need constant vigilance in the form of maintenance – the motivation of this PhD research.

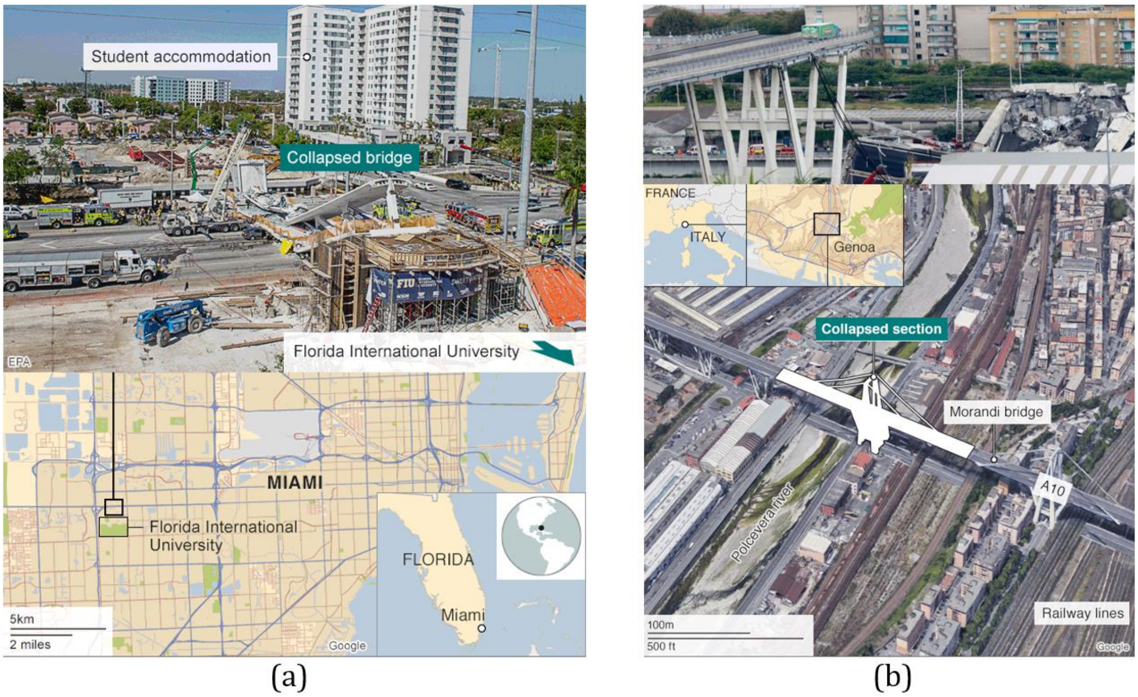


Figure 1.1 (a) American footbridge collapse near Florida International University on 15 March, 2018, at least 6 dead (Image: BBC News, 2018b); (b) Italian bridge collapses in Genoa on 15 August, 2018, at least 43 dead (Image: BBC News, 2018a)

The United States (US) spends roughly $12.8 billion to address deteriorating bridge conditions every year (ASCE, 2013), while the United Kingdom (UK) spends £4 billion annually for maintaining the road network (NAO, 2018) (Figure 1.2). The reasons behind these massive costs are partly because highway bridge owners face a major challenge with collecting, structuring, and managing the data needed for rapid repair, maintenance, and retrofit of their bridges. The data available in Bridge Management Systems (BMS) does not meet the standard of information needed for sound decision-making (ASCE, 2017).

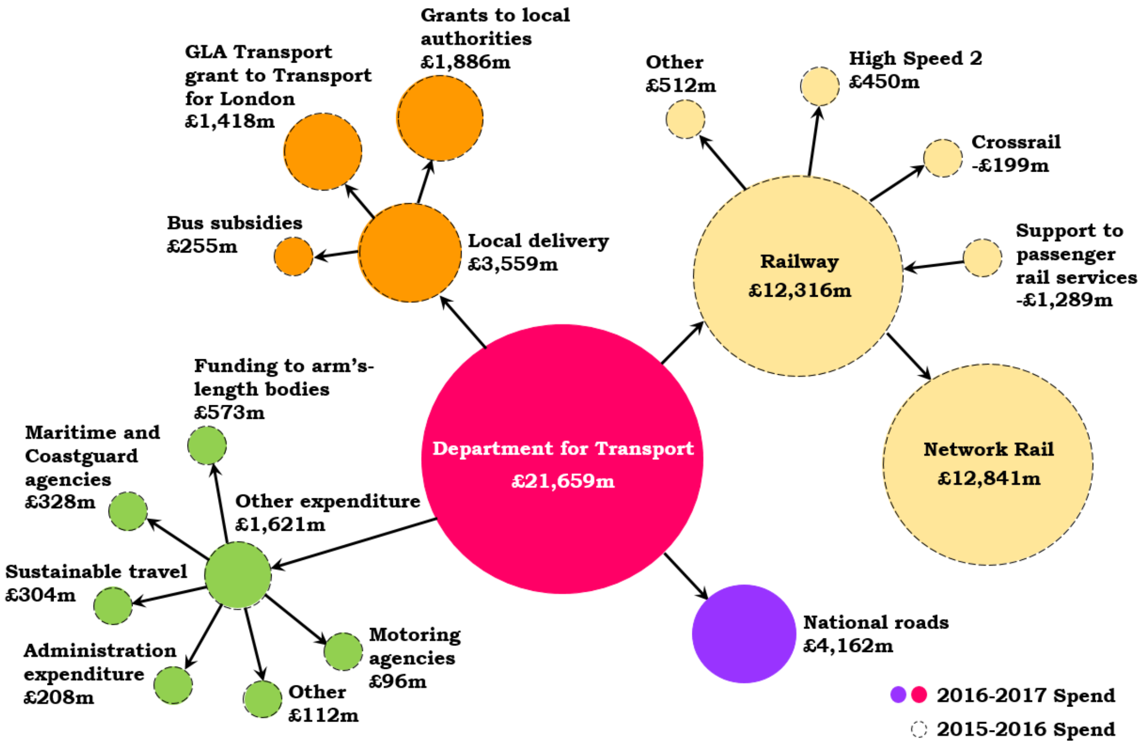


Figure 1.2 Where the UK Department for Transport spends its money (NAO, 2018)

The DT is not a new concept. A DT of existing infrastructure, such as an existing bridge, can continuously learn and update itself from multiple data sources to represent the near real-time[1] status and working condition (HM Government, 2013). A bridge DT provides a digital representation of a real bridge and forms an optimal basis for computational applications. The actual content

[1] In this context, near real-time means using sensor data to reflect the up-to-date status of an asset during each inspection by automated data processing. The status includes but not limited to the as-is geometries, damage evolution (concrete spalling, degradation, crack, corrosion of embedded rebar, and etc. from last inspection), and maintenance/operational history of the inspected asset.

of a bridge DT depends on the use case and the project phase it is applied in. Typical DT use cases include but not limited to visualization, drawing generation, progress monitoring, and project management. In addition, a DT is maintained throughout the whole life-cycle of a bridge and can be accessed at any time (Parrott & Lane, 2017). Hence, it provides infrastructure owners with an early insight into the potential risks induced by climatic events, heavy vehicle loads or aging (Koch et al., 2014). This repository of data and models of existing bridges can be integrated into a decision-making support tool which enables engineers and decision makers to understand, learn, and reason, so that improved decisions can be made.

The greatest value of using DTs is that they are projected to save substantial costs for highway authorities and bridge owners by automating the inspection process and enabling accurate condition assessments and timely maintenance decisions. Industry optimistically believes that the wider adoption of DTs will unlock 15—25% savings to the global infrastructure market by 2025 (Barbosa et al., 2017; Gerbert et al., 2016). The following paragraph identifies the importance of regular inspections for the large number of existing bridges in the US and the UK.

There are 614,387 bridges in the US (ASCE, 2017). Over 9.1% of all bridges are rated as structurally deficient and 13.6% as functionally obsolete (ASCE, 2017). The cost of eliminating these and new deficiencies equates to an overall investment of $123 billion (ASCE, 2017). As this money is not expected to be available, federal and state agencies have developed bridge inspection and rating tools aimed at prioritizing bridge rehabilitation projects in order to maximize the cost and benefit ratio of their investments. Likewise, in the UK, Network Rail and other bridge owners such as local authorities, London Underground, Transport for London, etc. manage more than 30,000 bridges on the UK's motorways and major A-roads (Network Rail, 2015). This means, based on a two-year inspection cycle (the general inspection interval), there is a need for at least 315,000 bridge inspections per annum across the US and the UK. Inspections, generally provide the most up-to-date and comprehensive data on the condition of bridges and as such are a key input for maintenance planning. The following texts briefly review current status of bridge inspections in the US and the UK. This explains why highway asset owners need DTs to conduct the future inspection work.

In the US, the Department of Transportation (DoT) of each state conducts bridge inspections and evaluates the bridge condition following the National Bridge Inspection Standards (FHWA, 2012) and the AASHTO (American Association of State Highway and Transportation Officials) Manual (AASHTO, 2011). In the UK, bridge owners undertake bridge inspections following the Inspection Manual for Highway Structures (2007). Before going onto a bridge, certified inspectors must prepare sketches and note templates for references throughout the inspection (Highways England, 2018a). Bridge inspection today, is a largely labour-intensive manual process. Visual inspection is still the most common form of condition monitoring used by bridge owners worldwide despite Structure Health Monitoring systems growing in popularity in the world of bridge engineering (Webb et al., 2014). During inspection, actual bridge conditions are recorded by observing visible damage and defects that lie on primary bridge components, such as slabs, exterior girders, and piers. The resulting physical condition information of structural elements from visual assessment is then entered into a BMS such as the US's AASHTOWare Bridge Management Software (AASHTOWare, 2018), formerly PONTIS, or the UK's National Structures Database (NATS) (Flaig & Lark, 2000), to rate the bridge condition.

Different rating systems are used to rate the bridge condition. For example, in the US, local DoT uses the system of National Bridge Inventory (NBI) General Condition Ratings (GCRs) to describe the existing in-place bridge, compared to the as-completed condition, in order to categorize the necessary bridge maintenance action (Figure 1.3). A conservative condition rating will result in unnecessary actions, such as costly bridge strengthening or repairs.

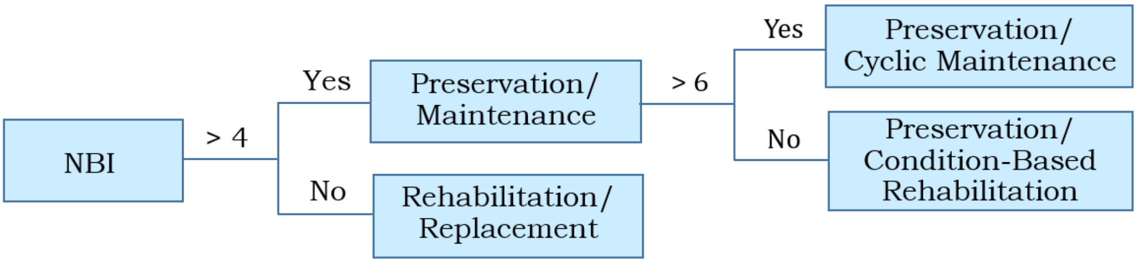

NBI
> 4
Yes
No
Preservation/ Maintenance
Rehabilitation/ Replacement
> 6
Yes
No
Preservation/ Cyclic Maintenance
Preservation/ Condition-Based Rehabilitation

Figure 1.3 Bridge action categories depend on the condition rating (FHWA, 2018)

Likewise, in the UK, asset owners use the Code of Practice (UK Roads Liaison Group, 2013) to guide on how inspection results and other structural performance data should be used to inform the maintenance planning process. However, both rating systems are qualitative in nature. The data currently recorded in these BMS does not support meaningful structural modelling information for rating or permitting purposes (Rashidi et al., 2016). This is particularly true for Reinforced Concrete (RC) bridges, where the location and type of cracking and material deterioration can have a crucial influence on the expected performance. AASHTOWare, NATS, and other BMSs are geared primarily to make system-wide prioritization decisions based on high-level comparisons of condition data (U.S. DoT, 2015; Vassou, 2010). They do not assess the actual condition of a particular bridge component and of a particular location of the component. Having a gDT would be quite useful for this purpose as texture and damage information can then be properly integrated with the bridge geometry on a component-level. So, how far are we now on creating such gDTs? In the following section, the author provides a review on the current state of implementation of DT in the Architecture, Engineering, and Construction (AEC) and Facility Management (FM) sectors.

1.2. Bridge Digital Twin: Where are we now?

Fundamental to effective management is an inspection regime that provides timely, accurate, and appropriately detailed information of asset condition and performance. The need for innovative inspection solutions has led to numerous academic and industrial efforts toward Bridge Information Modelling (Hüthwohl et al., 2018; Sacks et al., 2018), i.e. the technology for modelling the bridge DT, which is conventionally termed **BrIM** (and BIM for modelling building DT) (Eastman at al., 2011). Hereafter, the author uses "**bridge DT generation**" instead of BrIM/BIM to keep the terminology consistency in this thesis.

Globally, bridge DT generation is a relatively small but growing activity compared to its building counterpart. Figure 1.4 shows that the percentage of using DT on over half of the infrastructure projects in the US and Europe has doubled or even tripled from 2013 to 2017 (Buckley & Logan, 2017; Lee et al., 2014).

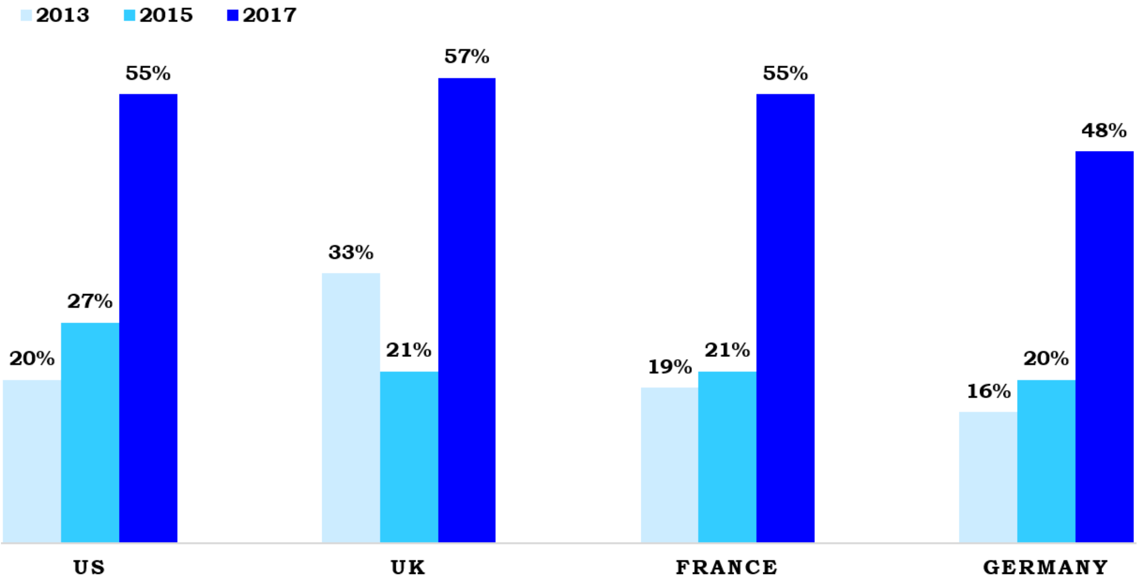


Figure 1.4 Use of DT on 50% of infrastructure projects by country (Buckley & Logan, 2017)

The core of bridge DT generation is to use the gDTs and a common data environment to support a reliable basis for decision making during the life-cycle of a bridge (Eastman at al., 2011). According to Koch et al. (2014), three types of DT can be defined in the life-cycle of a bridge:

- **“As-designed”** DT, produced by the design team and containing detailed information generated by the subcontractors and suppliers. This model contains the performance requirements and other data necessary to define the bridge’s intended function.
- **“As-built”** DT, produced by the general contractor and reflecting the state of the bridge at the time of its completion.
- **“As-is”** DT, produced by the bridge management agency through surveys of the bridge at regular time intervals. This model is ready for use in maintenance and operations.

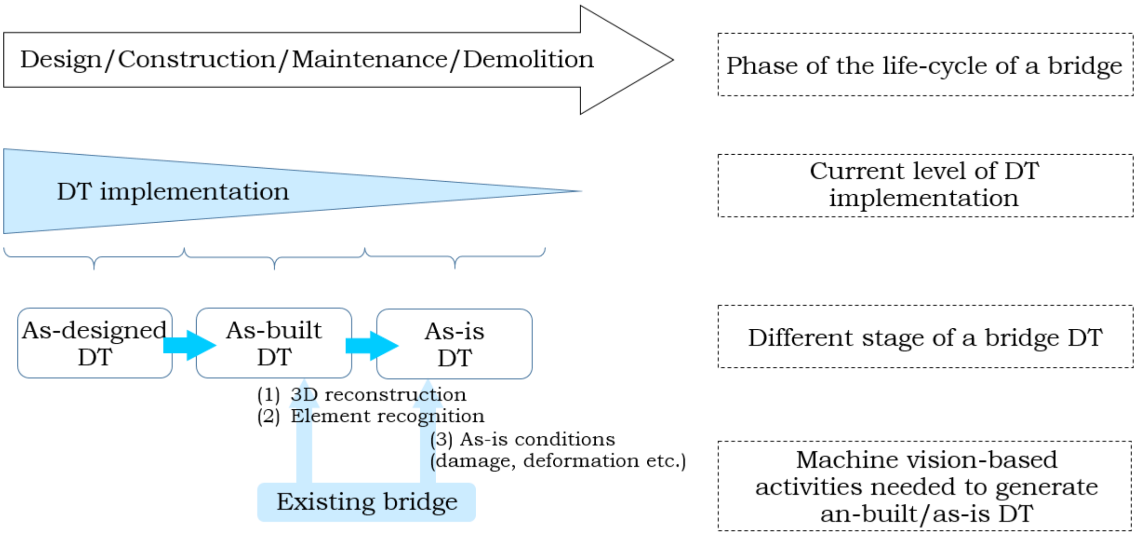

Design/Construction/Maintenance/Demolition
Phase of the life-cycle of a bridge
DT implementation
Current level of DT implementation
As-designed DT
As-built DT
As-is DT
Different stage of a bridge DT
(1) 3D reconstruction
(2) Element recognition
(3) As-is conditions (damage, deformation etc.)
Existing bridge
Machine vision-based activities needed to generate an-built/as-is DT

Figure 1.5 Life-cycle of a bridge and three types of DT implementation (adapted from Koch et al., 2014)

As shown in Figure 1.5, the use of a bridge DT is greatest during the design stage (as-designed), while little use is made in the closeout stage (as-built DTs), and almost absent in the maintenance stage (as-is). This finding is statistically in line with that of Buckley & Logan (2017) (Table 1.1). Almost no as-is bridge DTs are generated, so no expected value is realized (0% US, 2% UK, 1% France, and 0% Germany).

	US	UK	France	Germany
Before Design Begins				
Preplanning (US)/Brief (UK, France, Germany)	7%	0%	4%	2%
Predesign (US)/Concept (UK, France, Germany)	15%	22%	10%	19%
During Design				
Design Development (US)/Developed Design (UK, France, Germany)	36%	49%	49%	44%
Construction Documentation (US Only)	11%	-	-	-
Bidding/Construction/Installation				
Bid Letting (US)	1%	-	-	-
Production (UK, France, Germany)		13%	20%	22%
Construction (US)/Installation (UK, France, Germany)	28%	7%	3%	13%
Post-Construction				
Project Closeout (US)/As Built (UK, France, Germany)	0%	7%	12%	0%
Maintenance (US)/Use (UK, France, Germany)	0%	2%	1%	0%

Table 1.1 Project Stage at Which DT Provides the Greatest Value (According to Engineers and Contractors) (Buckley & Logan, 2017)

The reports forecast that the wider adoption of DTs will unlock 15—25% savings to the global infrastructure market by 2025 (Barbosa et al., 2017; Gerbert et al., 2016). The World Economic Forum (2016) claims that it is the technology-led change most likely to deliver the highest impact to the construction industry.

Yet, today, bridge owners do not generate DTs for existing bridges because they perceive that the cost of doing so outweighs their benefits (West & Blackburn, 2017). Hereafter, the "**DT**" specifically **refers to** the "**as-is DT**", generated for existing infrastructure, except as otherwise noted. In the following section, the author reviews the current practice of DT generation. This explains why the DT implementation is so limited.

1.3. Current Practice of Digital Twin Generation

A DT is the digital representation which serves as a link between a real-world asset and a DT itself. DT can be continuously updated using data collected from the sensors. The data is first collected and used to constitute a twin in an unstructured data format of the real-world asset, such as a “point cloud” of an asset. It is a low-level digital representation which does not contain any meaningful information. The low-level twin is then converted into a high-level digital representation, namely the DT, through a “Twinning” phase, which aims to structuralize the unordered raw data. In the following text, the author discusses first the current state of data collection, i.e. PCD collection, for generating a gDT, and then discusses the current practice of how to convert PCD into a DT.

There are numerous sensors on the market capable of collecting real-time data, such as mobile phones, digital cameras, videos, fibre optic sensors, acoustic emission sensors, electrochemical fatigue sensors, and so on. Among these sensors, laser scanners, have already been widely implemented for faster and better data collection (Laing et al., 2014; Tang, 2009). Laser scanning (LS) is a technology where an object’s surface is sampled using a line of laser light. Time-of-flight (or pulse-based) scanners are the most common type of laser scanner for civil engineering projects because of their long effective maximum range up to 1000 m and data collection rates of up to 50,000 points per second (Pfeifer & Briese, 2007). It is a way to capture an object’s exact size and shape into the computer world as a digital representation, which is called a “point cloud” – a collection of XYZ co-ordinates in a 3D co-ordinate system. It may also include additional information, such as colour and reflectivity values. The following text consists of two parts: In Section 1.3.1, the author presents current state on collecting PCD for the purpose of bridge inspection. In Section 1.3.2, the author reviews the state of practice of DT generation from PCD using the cutting-edge software. This demonstrates the practice of DT generation is a daunting task, despite LS has been widely adopted, which, in turn, explains why the DT implementation is limited.

1.3.1. TLS-based inspection

The use of terrestrial laser scanning (TLS) is gaining increasing interest because it is non-destructive, non-contact, highly accurate, rapid and can operate over a long range (Riveiro et al., 2016). Previous studies have demonstrated how PCD can be used as the input for construction progress monitoring (Bosché, 2012; Bosché et al., 2015; Bosché et al., 2009; Kim et al., 2013), structural health monitoring (Park et al., 2007; Park et al., 2015), and construction dimension surveying (Anil et al., 2011; Anil et al., 2013; Tang, 2009).

TLS is also able to drastically reduce the on-site bridge inspection time compared to the conventional manual bridge inspection approach mentioned earlier. Instead of spending days or weeks, inspectors need only a matter of hours to scan a typical highway or steel bridge. The average time spent on bridge surveying field work is discussed in (Tang et al., 2007) and (Foltz, 2000). Table 1.2 shows a comparison of bridge inspection data collection times between using the traditional visual method and the TLS-based method. As the table shows, visual bridge inspection is a time-consuming process and varies from case to case based on the complexity of the bridge and the inspection type. Note that, it only compares the time spent on the spatial geometric data collection. The actual damage detection and assessment remains manual. One of objectives (minor) of this research is also to report the average time of the PCD collection for typical highway RC bridges. The author elaborates on the data collection activities as well as the post-processing stage later in this thesis as expected in the hourglass model. The detailed activities as well as the statistics of the collected data are presented in Chapter 6 Section 6.2.

		Manual		**TLS**	
Case study	Bridge type	# of inspectors	Time (h)	# of inspectors	Time (h)
(Foltz, 2000)	RS	4	112	2	3.5
(Tang et al., 2007)	HC	3	8	1	<2
(Gyetvai et al., 2018)	RS	-	-	-	1

RS: River Steel, HC: Highway Concrete

Table 1.2 Time spent on manual & TLS-based bridge inspection

TLS reduces survey labour hours and inspection cost. Lane closures are often unavoidable when using traditional manual inspection methods and accumulate additional costs. In the UK, there are more than 18,000 full or partial lane closures lasting a total of over 20,000 hours. This costs £1 billion annually (Department for Transport, 2011). In Australia, €11,000 to €24,000 per day is the cost reported for the closure of two to four lanes, for secondary and primary roads respectively (West & Brereton, 2013). The negative impacts resulting from traffic interruption can be minimized to a large extent if TLS is employed. The cost-benefit of TLS for data collection of highway facilities is investigated in (Yen et al., 2014), where the authors studied the cases of the state departments of transportation in Washington (WSDoT) and California (Caltrans) and claimed that the current variable costs to them range only from $100 and $150 per structure. Highway asset owners have gradually adopted TLS as the equipment and service costs of TLS continue to decrease (Randall, 2013). However, as discussed earlier, the adoption of DT is currently quite limited. Many asset owners find themselves data rich but information poor (Koch et al., 2014). This is because TLS generates millions of unstructured points where useful information is difficult to be extracted. A solution that can generate a gDT is urgently needed in order to structuralize the PCD such that one can derive useful higher-level information from the structured data, i.e. the gDT. To this end, the following section presents the current state of practice of the gDT generation from PCD using the cutting-edge software solution.

1.3.2. From PCD-to-DT

Although there are already many capable LS hardware solutions on the market for the efficient collection of accurate bridge geometry data in the form of point clouds, the adoption of DT is very limited for existing bridges. This is mainly because manual DT generation of even a seemingly simple structure from point clouds is a daunting task, even for a skilled modeller. More than two thirds of the human effort needed for generating a DT is dominated by geometric modelling from PCD. The time required to manually create a DT from PCD using cutting edge modelling software tends to be ten times greater than that required to obtain the original point cloud (Trimble, 2017).

Generating a DT from PCD is time-intensive and costly. The total cost of DT generation can be broken down to fixed and variable costs. Fixed costs refer to the fees for DT modelling software licenses, hardware required for using the software, and training for inexperienced modellers. Training is necessary because generating a DT is a very domain-specific task; it is not realistic to assume that even proficient CAD professionals will be competent enough to manipulate modelling software without any specialized training (McNell et al., 2011). Variable costs are the fees spent on each individual modelling project, which are usually represented by the total modelling hours of and the corresponding hourly labour cost. Assuming the fixed costs and the hourly labour cost are constant, the total cost of gDT generation is then determined by the total modelling hours. This means cost savings will be achieved if we can reduce the total modelling time by automated solutions. In the following text, the author first defines **(1)** the end-user requirements (EURs) of DT generation. The author then **(2)** provides a brief review of existing software packages to see how far they have achieved in terms of degree of automation on DT generation, according to the EURs. The author finally **(3)** identifies what is the most time-consuming step in the whole process of bridge DT generation from the PCD by investigating the current modelling practice.

End-user requirements (EURs)

The end-users of bridge DTs are the inspectors, structural engineers, and the decision makers. The end-user requirements (EURs) define the information that will be required by the end-users from both their own internal team and from suppliers for the development of the project and for the operation of the completed built asset. The EURs should clearly articulate the information requirements for each supplier and describe the expected information deliverables in terms of documents, model files, and structured information. However, the nature of the EURs depends on the complexity of the project, the experience, and the requirements of the end-users. Experienced end-users may develop very detailed EURs, whilst others may only set out high-level requirements, and some basic rules, leaving the supplier to propose how those requirements will be met. Based on the interviews with several inspection agencies (e.g. Amey) and engineering consulting companies (e.g. Kedmor Engineers Ltd), the author deduces the following EURs of a bridge DT. Broadly, EURs should include:

- **EUR 1:** Structural-component-level digital representation. A DT of a sensed bridge contains the main bridge component types (e.g. pier, beam, deck, bearing, etc.). Elements that are occluded or that are too small or invisible to be discerned due to insufficient scan resolution are not provided.

- **EUR 2:** Component explicit geometry representation and property sets. A bridge DT is prepared for the use as an "Inspection Digital Model" in a BMS. The full geometry should represent as-is conditions of the sensed bridge.
- **EUR 3:** Component taxonomy. The components making up a bridge DT should not only be modelled, but also identified and labelled by their element types.
- **EUR 4:** Component implicit information such as spatial semantics of attributes and structural relationships, material, cost, schedule, etc. A bridge DT should be sufficiently semantically meaningful to provide most of the information needed for decision-making concerning the repair, retrofit or build of a bridge.
- **EUR 5**: Component damage information and the association to the bridge parts. Damage type (structural crack, non-structural crack, spalling, scaling, efflorescence, carrion, and others), location, and orientation should be exactly identified and embedded into the DT along with the texture/image data for each visible element.

A DT should also be exchanged in between various project participants who use different platforms, so

- **EUR 6:** All EURs should be presented in a platform neutral data format, such as IFC.

Next, the author reviews current software to check whether these EURs are satisfied. Major vendors such as Autodesk, Bentley, Trimble, AVEVA and ClearEdge3D, etc. provide the most advanced PCD-to-DT modelling software solutions. Wang et al. (2015) and Agapaki & Brilakis (2018) provided thorough reviews of the pros and cons of current DT modelling commercial software. These software packages are able to automate to a large extent the DT generation process, however, they are still far from being fully automatic. For example, existing software packages can automatically extract the maximum amount of planar features, up to 90% pipes in a plant point cloud, and specific standard shapes like valves and flanges from industry catalogues (ClearEdge3D, 2017) followed by fitting built-in models to them, though a few clicks and manual adjustment may be required. This means that ClearEdge3D has realized a certain degree of automation on the DT generation as the EUR 1 & EUR 2 have been partially automated. However, ClearEdge3D is tailored for building and industrial environments. The spec-driven component library of ClearEdge3D can only recognize and fit point cloud subparts with standardised shapes such as rectangular walls, pipes, valves, flanges, and steel beams, etc. based on an industry specification table or a custom user-defined table (ClearEdge3D, 2017). For other commercial applications, none of them can automate any one of the EURs. Modellers must first manually segment a PCD into subparts, and then manually fit 3D shapes to the subparts (EUR 1 & EUR 2). This demands a significant amount of attention when extracting the target objects. Modellers need to repeatedly rotate the point cloud to various views and try to select regions of interest using clipping polygons. Then, fitting accurate 3D shapes to the segmented point clusters is challenging. Most software applications provide built-in shape libraries from which a few predefined and generic construction component primitives can be found (Figure 1.6). However, the set of allowable primitives is limited (Wang et al., 2015). Next, to meet the EURs, modellers need to enrich other explicit and implicit information such as the component's taxonomy (EUR 3), the connectivity and aggregation (EUR 4), and the defects (EUR 5). Then, all EURs need to be exported in IFC format (EUR 6). However, only EUR 3 and EUR 6 can be done manually, other EURs, i.e. EUR 4 and 5, are yet unsupported in current software. In the following, the author investigates the current DT generation practice for bridges, which identifies the challenges as well as the most laborious work during the whole modelling process.

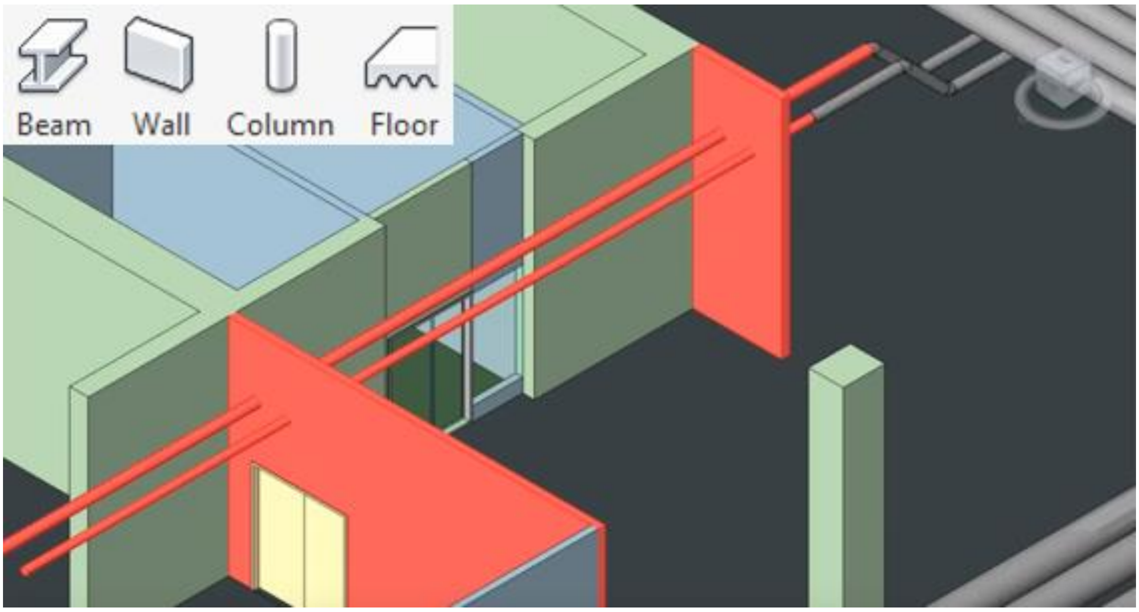


Figure 1.6 Revit fits cuboids and cylinders to indoor elements using the built-in library

Given that RC bridge components usually have arbitrary shapes, containing skews or imperfections, and cannot be simply fitted using those idealized predefined shapes, modellers must manually create an accurate solid form to fit each point cluster as none of the existing software package can do this automatically. In the following text, the author elaborates the entire workflow of the DT generation of a typical RC bridge from its PCD using CloudCompare 2.6.2 (2017) and Autodesk Revit 2016 (2016). CloudCompare is used for segmenting the point cloud into point clusters making up a bridge. Revit is used for importing the point clusters, customizing and fitting 3D shapes to them, and finally exporting the Revit modelling project into an IFC file. Many software solutions provide segmentation functionality, such as Trimble RealWorks (2018). The author randomly chooses CloudCompare as it can read multiple data formats. Revit provides excellent flexibilities that allow users to design a shape in a more freeform manner. Geometry in Revit’s Family consists of solid and void forms. Solid forms represent the actual physical parts of the family and void forms are used to carve away portions of the solid forms. For example, one can create a solid form box, and then use a void form to cut a hole in it like a doughnut. Both solid and void forms come in five varieties. These include Extrusion, Blend, Revolve, Sweep and Swept Blend (Figure 1.7). One can build complex forms using a combination of the solid and void forms available in the Family editor as noted. However, managing a complex form in a single Family object can become cumbersome. In many cases, it makes sense to break a complex object into discrete parts and build the parts as separate Family objects. Given the powerful customization capability that Revit's Family provides, the author, therefore, chooses Revit to generate the bridge DT.

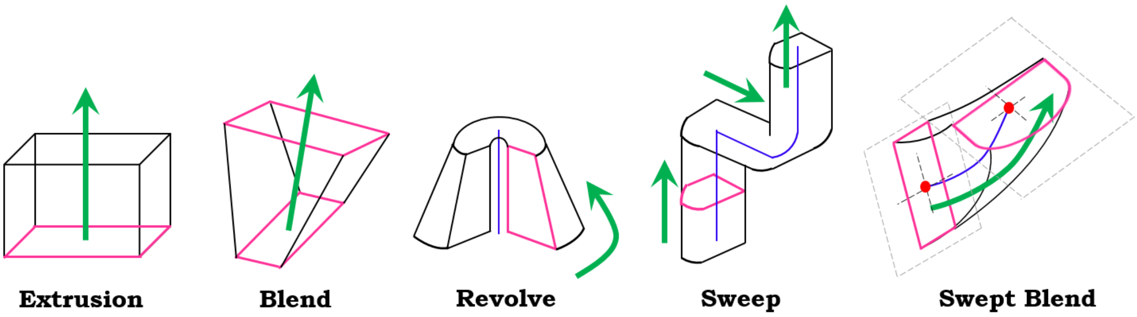


Figure 1.7 Forms available in Revit Family editor

Figure 1.8 illustrates the workflow of the manual bridge DT generation from the registered PCD as well as the expected EUR achieved for each step. The whole process consists of 8 steps. The detailed description of each step can be found in Appendix A.

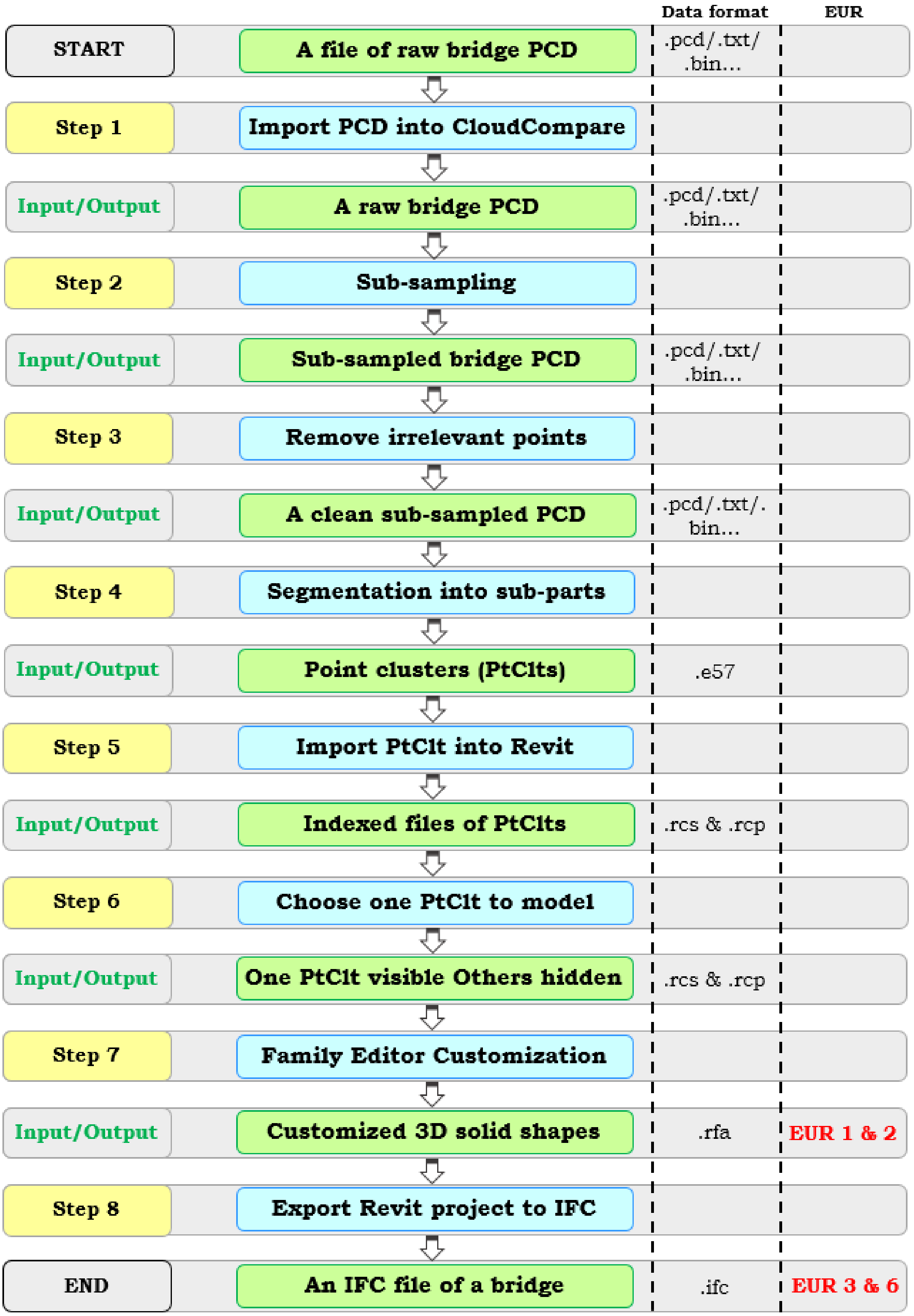


Figure 1.8 Workflow of the manual bridge gDT generation from PCD

As demonstrated in Appendix A, a modeller can only manually produce a bridge gDT with components' labels using current software. However, the resulting gDT is not an actual bridge DT as EURs 4 and 5 are missing. In addition, this gDT modelling process is laborious, containing many repetitive processes. Step 7 is the most time-consuming step, with 95% of the total modelling time spent on customizing shapes and fitting them to the point-clusters. The overall average gDT generation time of a typical RC highway-bridge PCD will be reported in Chapter 6 as part of the ground truth preparation work in this thesis. The following text presents several observations from the DT manual modelling practice. This supports the argument that Step 7 is the most time-consuming step. This also demonstrates human reasoning ability a modeller has in modelling.

Figure 1.9 shows an example of pier modelling using Revit. The pier is in an unusual form and cannot be fitted using a cuboid. A modeller needs to customize the shape. With the help of the Family editor, the customization can be done by outlining the cross-section and extruding along its extruded direction. Likewise, Figure 1.10 shows another example of deck slab modelling. The slab is skewed in the horizontal as well as vertical direction and varies in elevation. These characteristics make the manual modelling quite challenging. A modeller needs to segment the entire slab into sub-segments and then fit each sub-segment with a customized Revit Family object. This way, the topology of the deck slab is approximated using multiple sub-segments, which are relatively easier to model.

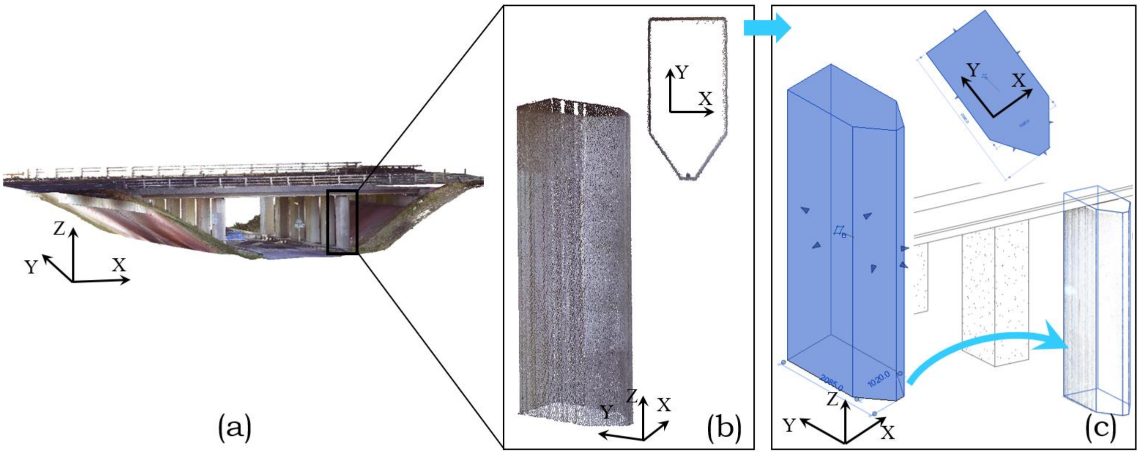


Figure 1.9 Example of pier modelling: (a) point cloud; (b) segmentation; (c) insert point cloud into Revit and fit the point cluster with an unusual shape through customized modelling

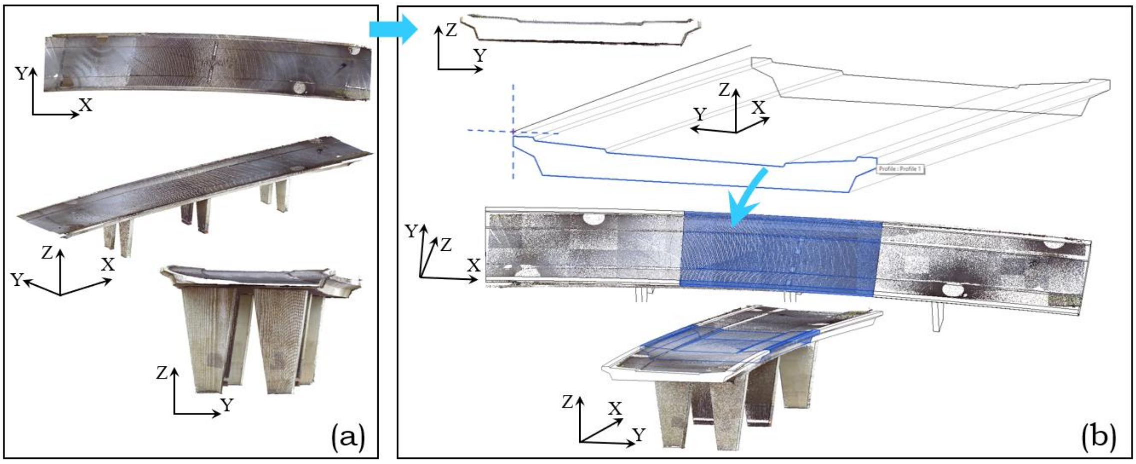


Figure 1.10 Example of deck slab modelling: (a) point cloud; (b) segmentation, insert point cloud into the software, and shape customization of the slab segment through customized modelling

It is also worth noting that occlusions (Figure 1.11) and varying point density (Figure 1.10 (b)) normally do not seriously affect the manual modelling performance as a modeller can infer missing or ambiguous geometries based on the neighbouring points and his or her perception of the object being modelled. For example, the web points of girders underneath the deck may be almost missing (Figure 1.11 (a)). A modeller can still infer the girder profile using the captured flange points and the estimated girder height. Another example, is the bottom part of piers located in the middle of two highway lanes, which are often partially occluded due to the in-situ highway guardrails (Figure 1.11 (b) (c)). Yet, modellers can easily understand the continuity and produce an entire pier model.

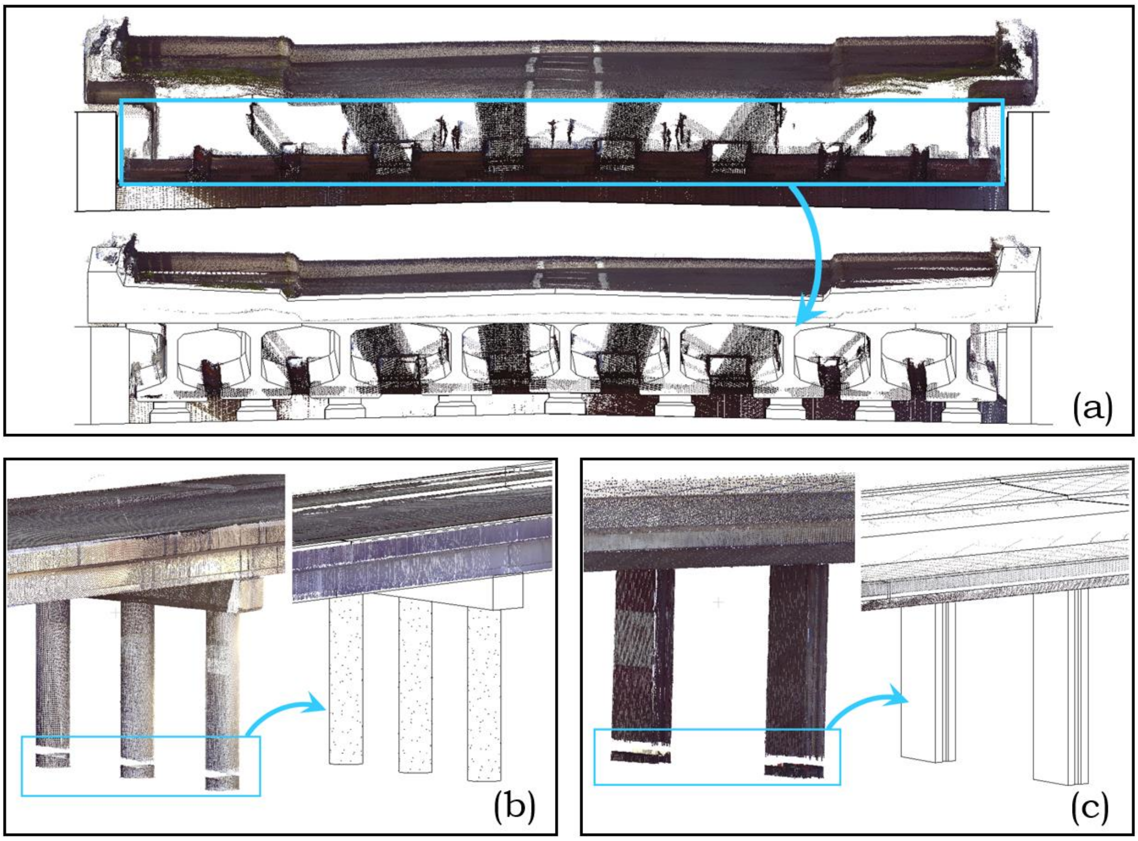


Figure 1.11 Examples of modelling with presence of occlusions in resulting point clouds: (a) fitting shapes to girders; (b) (c) fitting shapes to piers

Last, the author summarizes the "bottlenecks" of current software packages in modelling an actual bridge DT as following:

1) Existing software packages can semi-automatically extract standardized shapes in PCD. However, they cannot automatically extract non-canonical shapes, which are frequently present in RC bridges. Manual shape customization is laborious and time-consuming.
2) EUR 2 is manually achieved. In addition, the presence of occlusions and sparse data slows down the workflow and adds hours of adjustments. However, it is worth noting that human reasoning can easily identify not only the shapes of bridge components but can also deduce the functional properties and fill in missing information by interpreting the spatial and topological relationships between the components.
3) EURs 1, 3, and 6 are manually achieved and EURs 4 and 5 are unavailable within existing applications. The generated 3D models from existing software packages do not carry any implicit information (metadata). The metadata is

necessary to produce a "meaningful" DT, which can be used to support the condition rating, including but not limited to the semantic meaning of elements, element materials, relationship, defects, schedule, cost, and maintenance history. These attributes must be further added.

4) There is no single software that can offer a one-stop DT generation solution. Modellers have to shuttle intermediate results in different formats back and forth between different software packages during the modelling process, giving rise to the possibility of information loss.

This thesis intends to meet the EURs 1, 2, 3, and 6, i.e. the EURs required to generate a gDT with component-level semantic labels. The author states in the following section the research design, writing structure, overview summary of upcoming sections, and the expected contributions of this thesis.

1.4. Research Design, Writing Structure

This thesis starts by introducing the background information of industry, which ensures readers to understand the context of this PhD research. In order to formulate research objectives, research questions, and highlight the originality of this research, the author provides a literature review. It covers and links existing works related to the author's research, and identify the remaining knowledge gaps. The author then describes the developed methods, data collection, experiments, and results by breaking down all the details of how they take place from beginning to end to demonstrate that the knowledge gaps have been filled and this thesis makes contributions.

Specifically, the overall structure of this thesis follows the **hourglass model** (Schulte, 2003). The author starts off broad with an introduction (Chapter 1) of a social and practical issue. The introduction reviews the state of practice to derive the end user requirements and the technical limitations of practice. The background (Chapter 2) reviews the state of research that focuses on the limitations identified to derive the remaining knowledge gaps, technical objectives, and several specific research questions. As its narrowest scope, the hourglass model then focuses on presenting the hypothesized framework (Chapter 3, 4, and 5) followed by its implementation, experiments and results (Chapter 6) to test its performance. The conclusions (Chapter 7) then broaden up again, by discussing and interpreting the results, presenting knowledge contributions in the narrowest sense, then linking the results to the literature to explore broader contributions to practice and the society as a whole.

The following expected contributions to knowledge will be achieved if the derived research questions are well addressed:

- **Exp. contribution 1:** It will be the first method of its kind to robustly and reliable detect RC bridge components in real-world PCD.
- **Exp. contribution 2:** It will be the first method of its kind to cost-effectively generate gDTs for existing RC bridges in IFC-infra format.
- **Exp. contribution 3:** The generated bridge gDTs will be validated through a quantitative measurement.

2. State of Research

The use of existing software packages in the DT modelling process is still human dependent to a great extent. Extensive manual effort is required for practitioners to extract component point clusters from PCD followed by manually fitting them with accurate 3D solid shapes. Modelling the world around us is a longstanding goal in the field of Computer Graphics, Computer Vision. Central to this objective is a means of acquiring a gDT of a real-world physical object. Much research effort has been devoted to automating the modelling process.

The author divides the existing research methods into two parts according to the EURs of the gDT generation: **(1)** object detection in PCD (EURs 1 and 3) (Section 2.1); and **(2)** 3D solid model fitting to point clusters (EURs 1, 2 and 6) (Section 2.2). Partial automation of the PCD-to-gDT process has been achieved with the help of visual pattern recognition concepts and geometric modelling techniques in recent years. A detailed review of these concepts is presented in the following.

2.1. Object Detection in PCD

Object detection aims to locate a desired object in a scene (Uijlings et al., 2013). The author defines "detection" in this context as the combination of **clustering** (from a point cloud to point clusters) and **classification** (labelling the point clusters). Current methods of point cloud clustering generally follow a "bottom-up" approach (Section 2.1.1), which goes from points to surfaces or patches followed by semantic labelling to derive objects. Most point cloud classification methods follow a "top-down" approach (Section 2.1.2), which employs human visual perception such as relationships and contexts to detect specific instances embedded in PCD or to infer the semantics of components in a geometric model.

Real PCD is imperfect data with many problems, such as occlusions and varying point density. There are two kinds of **occlusions**: visual occlusions and physical occlusions. Visual occlusions refer to occlusions where one or more objects are visually blocking the laser sensor from being able to see another object (Hsiao & Hebert, 2012). This is a simple form of occlusion that can be ameliorated by taking multiple scans from different standpoints. Physical occlusions refer to occlusions that cannot be removed by moving the scanner or orbiting the target scanned object in any direction. This form of occlusions is attributed to on-site constraints, such as unmovable obstacles, furniture, trees and so on. Hereafter, the author uses "occlusions" to refer to the physical occlusions. **Varying point density** is caused when the distribution of the points sampling on the scanned surface (sampling density or point density) is non-uniform. This often occurs because the distance from the target object to the scanner position can vary significantly, as well because of an object's geometry. These characteristics of real PCD render the sampled surfaces of many regions with few or no measurements.

The author reviews both existing bottom-up and top-down detection methods and investigates how far they have solved the above-mentioned challenges in the following texts. The author also explores what specific limitations these methods have and why these methods cannot be directly applied to the problem of detecting bridge components in real PCD.

2.1.1. Bottom-up detection

The bottom-up approach pieces together low-level primitive features like points to generate complex systems at successively higher levels until a top-level system is formed (Borenstein & Ullman, 2008). The higher-level features are typically the surface normal (Klasing et al., 2009; Sampath, 2010), meshes (Marton et al., 2009), surface planes/patches (Zhang et al., 2015), non-uniform B-Spline surfaces (NURBS) (Dimitrov & Golparvar-Fard, 2015; Dimitrov et al., 2016), and voxels (Vo et al., 2015).

A detailed survey of detection methods was investigated by Pătrăucean et al. (2015), Douillard et al. (2011), and Tang et al. (2010). A large body of literature has been devoted to generating surface-based primitives, especially planar surfaces (Pătrăucean et al., 2015). This is due to the high frequency of planar elements encountered in building models (Zhang et al., 2015) and to the reduced complexity of the problem (Xiong et al., 2013). Three main methods arise from the literature: RANdom Sample Consensus, Region Growing, and the Hough-Transform paradigm.

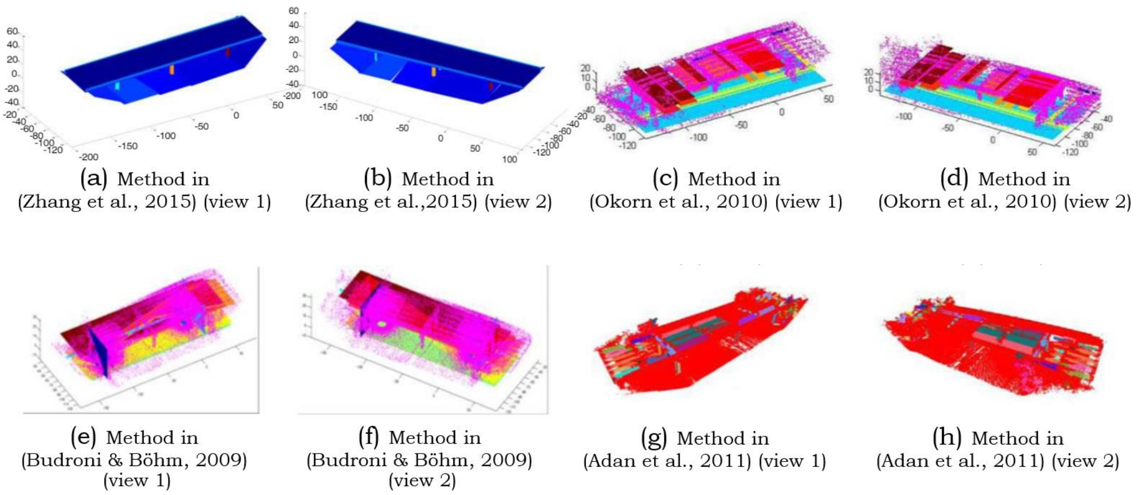
RANdom Sample Consensus (RANSAC) is one of the most well-known algorithms for point cloud clustering, especially for detecting planar surfaces in PCD. RANSAC has become a fundamental tool in computer vision since 1981, the year it was initially introduced by Fischler and Bolles (1981). The RANSAC algorithm is essentially composed of two steps that are iteratively repeated. In the first step, hypothesis shapes are generated by random selection of a minimal subset of points followed by estimation of the fitting model parameter of the sample subset. In the second step, RANSAC iteratively checks the remaining points to determine whether they are consistent with the model instantiated by the estimated model parameters obtained from the first step. The shape model that possesses the largest percentage of inliers (that fit the instantiated model within an error threshold) is extracted. For instance, to extract planes, three points are randomly selected from the PCD. Each three-point subset forms a plane hypothesis. Then, in the remaining PCD, points that are consistent with the plane hypothesis (i.e., within a distance criterion) are iteratively examined. A new best plane hypothesis is formed when more points in the remaining points are found to be consistent with the plane hypothesis. Specifically, the best planes are determined using the root mean square error (RMSE):

$$RMSE = \sqrt{\sum_{i=1}^{N}\left(\frac{|ax_i+by_i+cz_i+d|}{\sqrt{a^2+b^2+c^2}}\right)^2 \cdot \frac{1}{N}}, \quad \text{(Eq. 2.1)}$$

where (x, y, z) and N are the Cartesian coordinates of a point constituting the plane and its point count, and (a, b, c, d) is the parameter set which defines a 3D plane (i.e., $ax_i + by_i + cz_i + d = 0$).

A number of plane extraction studies are based on the RANSAC algorithm. For example, Tarsha-Kurdi et al. (2007) proposed an extended RANSAC algorithm to extract roof planes in low density PCD with different complexities. Jung et al. (2014), Arikan et al. (2013), Bosché (2012), and Nüchter & Hertzberg (2008) used RANSAC to detect planar surfaces such as walls, floors, ceilings, etc. in building PCD. Whilst the RANSAC algorithm is proven to be effective in the presence of noise and outliers (Tarsha-Kurdi et al., 2007), it has some limitations. First, since RANSAC is used to determine different planes from a single grouping, it thus suffers from spurious-planes (i.e., planes overlapping multiple reference planes or a plane snatching points from its neighbouring planes) which are frequently produced, especially around the boundaries (Jung et al., 2014; Yan et al., 2012). Second, RANSAC requires prior knowledge about the data. This means that the selection of a fixed number of shape hypotheses implies that a prior estimate of the inlier ratio is available. This is often not the case in practice. For example, Schnabel et al. (2007) detected five basic shapes (planes, spheres, cylinders, cones, and tori) with RANSAC using random sampling of minimal sets in a point cloud. Yet, given the computationally-expensive nature of this algorithm, it is unrealistic to use it to detect complex geometries. Hence, RANSAC-based methods tend to perform well in relatively simplified scenarios and with synthetic data but are not ready to tackle real bridge components whose as-weathered and as-damaged shapes further increase the as-designed complexity.

Recently, Zhang et al. (2015) presented a novel method to automatically detect planar patches from large-scale noisy bridge PCD. Their method begins with a recovery of the linear dependence relationships amongst the PCD by solving a group-sparsity inducing optimization problem. The recovered linear dependence is then used to cluster the points, followed by parameter extraction for each clustered group of points via Singular Value Decomposition (SVD), MLESAC (Torr & Zisserman, 2000), and the α-shape boundary detection algorithm. Zhang et al. evaluated their method on a bridge point cloud. The experimental results (detection rate 78%) indicated that their method outperforms existing baseline methods (Figure 2.1). Unfortunately, it cannot detect pier patches when the point densities of those regions are low.



(a) Method in (Zhang et al., 2015) (view 1)

(b) Method in (Zhang et al.,2015) (view 2)

(c) Method in (Okorn et al., 2010) (view 1)

(d) Method in (Okorn et al., 2010) (view 2)

(e) Method in (Budroni & Böhm, 2009) (view 1)

(f) Method in (Budroni & Böhm, 2009) (view 2)

(g) Method in (Adan et al., 2011) (view 1)

(h) Method in (Adan et al., 2011) (view 2)

Figure 2.1 Extracted planar patches of a bridge point cloud using different bottom-up methods (Zhang et al., 2015)

Region Growing (RG) is also a widely used scheme for point cloud clustering. The RG method was first introduced in the computer vision literature by Besl & Jain (1988) for grouping pixels with similar properties in 2D images. The extended implementation of the RG method to 3D PCD aims to merge or join similar neighbouring points and to deliver a set of point clusters belonging to the same piecewise-smooth surface (Hähnel et al., 2003). Similar to RANSAC, RG consists of two main steps. The first step starts with a set of small iteratively merged areas by arbitrarily choosing initial seeds. The second step adds in neighbouring points based on similarity of the surface normal (Belton & Lichti, 2006; Dorninger & Pfeifer, 2008; Kawashima et al., 2014; Macher et al., 2017), curvature (Hobi & Ginzler, 2012), or co-planarity (Xiong & Huber, 2010) until an edge is reached when a non-surface point is detected or the distance from the seed point exceeds a threshold. Normally, the optimal plane is detected when the sum of the square distances to the points $\{p_i\}$ in the current region is minimized. The normal of the detected plane is given by the eigenvector corresponding to the smallest eigenvalue of the 3×3 covariance matrix:

$$C = \sum_{i=1}^{n}(p_i - m)^T \cdot (p_i - m), \qquad \text{(Eq. 2.2)}$$

where

$$m = \frac{1}{n}\sum_{i=1}^{n} p_i \qquad \text{(Eq. 2.3)}$$

is the centre of the points $\{p_i\}$. The minimum eigenvalue corresponds to the sum of the squares distances between the plane and the points $\{p_i\}$.

Xiao et al. (2013) proposed two complementary plane segmentation algorithms for two different environments: a sub-window-based RG algorithm for structured environments (ordered PCD), and a hybrid RG algorithm for unstructured environments (unordered PCD). The sub-window-based RG method aims to use a relatively larger growth unit in the region growing procedure through sub-windows. Both algorithms outperform traditional point-based RG methods in terms of computational time (0.408 s for a 3×3 sub-window size and 0.198 s for a 4×4 sub-window size vs. 0.864 s for point-based RG). However, no other quantitative evaluations are provided in the work. Likewise, Xiong et al. (2013) suggested an RG-based algorithm to extract planar patches from a voxelized version of building PCD. This method is similar to the RG algorithm proposed by (Rabbani et al., 2006), which used the Total Least Square algorithm to fit a plane to the local neighbourhood of points within a radius criterion (10 times the voxel size), thereby providing a surface normal estimation. The average detection rates of planar patches (clutter, wall, ceiling, and floor) were 89.5% precision and 91% recall.

Walsh et al. (2013) presented an RG algorithm to detect both planar and curved surfaces in PCD. The authors used the total number of clusters formed by the investigated point and its neighbouring points to determine whether a point should be absorbed to the segment being grown. The authors assumed that a point belonging to a flat surface should only have one cluster. A region grows until all boundaries are surrounded by points that have more than one cluster. Similarly, a point belonging to a curved surface should have more than five clusters. A region grows until all points with more than five clusters are merged. If the cluster number is between one and five, the point is then classified as an edge point. However, as shown in their experiment, their method cannot properly detect the edge/boundary between a pier cap and a pier in a bridge point cloud (Walsh et al., 2013). The segmentation was finally achieved after manually choosing key points and manually merging the smaller surfaces around the boundaries.

Likewise, Dimitrov & Golparvar-fard (2015) suggested an upgraded RG method through which the seed is found adaptively. This method can deal with curved surfaces with a large range of surface roughness. It excels when the input PCD does not suffer from substantive occlusions and generates the smallest Hamming distance. Their method has a very low rate of false positives yet suffers from some limitations. First, it over-segments objects when non-trivial occlusions are present. Second, the generated surfaces require merging techniques to further produce higher-level point clusters which can be used to model 3D objects. Third, the runtime of the algorithm (6 hours for a subsampled point cloud of roughly 1 million points) is not efficient enough compared to manual operation.

The persistent occlusion problem in real PCD was addressed by Xiong et al. (2013) through a learning-paradigm that can detect occluded planar surfaces and estimate their shapes in building PCD. However, their method cannot be applied to bridge settings because the occluded surfaces in a bridge point cloud do not follow a specific pattern as in a building point cloud. Their algorithm detects rectangular-shaped openings, such as windows and doorways, assuming there are many identical openings on a wall and that the rectangle is the predominant shape in most buildings. Similarly, Laefer & Truong-Hong (2017) developed a kernel-density-estimation-based method for modelling steel members by simulating several possible occlusions. In contrast, the occluded regions do not have such a repeated pattern in bridge PCD. Most of the occlusions are due to on-site vegetation and long-distance scanning. Thus, occlusions in the context of bridges are in arbitrary locations and shapes, which cannot be tackled by the learning method nor by the simulation method.

In general, RG-based methods have been proven to be efficient at object detection in PCD. However, they suffer from occlusion effects, and also have the boundary weakness, which is due to the inaccurate estimation of normals or curvatures of points near region boundaries. These limitations give rise to issues such as over-/under-segmentation, which often requires a certain amount of manual adjustment (Díaz-Vilariño et al., 2015).

Hough-Transform (HT) is another a commonly used point cloud clustering method based on a feature extraction technique. It was initially invented for machine analysis of bubble chamber photographs (Hough, 1959). The HT, which is universally used in the computer vision and image processing community, was introduced by Duda & Hart (1972) after the related patent by Hough (1962). The HT was then popularized by Ballard (1981) and widely used for shape detection in 2D and 3D data. A detailed survey of HT-based methods can be found in (Mukhopadhyay & Chaudhuri, 2015). The major use of HT is in 2D where the number of parameters is typically small. For example, Okorn et al. (2010), and Adan & Huber (2011) proposed effective HT methods to detect walls in building PCD with clutter. The method first detects floors and ceilings through histograms of the height data. It then projects the remaining points in 2D followed by detecting walls from their point density histogram through an HT line detector, detecting the peaks in a 2D configuration space. Budroni & Böhm (2009) also used HT to classify horizontal and vertical walls in cluttered PCD after space partitioning. The methods proposed by Xiong et al. (2013) and Díaz-Vilariño et al. (2015) are similar. Both use HT to detect the strong horizontal and vertical lines in range image for building opening boundary detection. These methods work efficiently for detection of in indoor planar elements. However, HT becomes computationally prohibitive when the number of dimension increases (Rabbani et al., 2006), which increases the number of parameters. HT-based methods still work as a voting scheme for 3D data. Given a type of parameterized geometric primitive, the HT maps every point in the dataset to a manifold in the parameter space. This parameter space manifold contains many cells acting as accumulators, in which each point casts votes. The HT-based object detection method again consists of two major steps. The first step maps data points into a parameter space for voting. The second step searches for the cells with the maximum number of votes. Taking plane detection as an example, $\varphi, \theta,$ and ρ form the 3D Hough space (φ, θ, ρ) (Figure 2.2). Each point then votes for all sets of parameters (φ, θ, ρ) that define a plane in $\mathbb{R}^3$ on which it may lie. This means, given a point p_i in Cartesian coordinates, all planes on which p_i lies are found by identifying all possible sets of $\varphi, \theta,$ and ρ that satisfy:

$$p_x \cdot cos\theta \cdot sin\varphi + p_y \cdot sin\varphi \cdot sin\theta + p_z \cdot cos\varphi = \rho \qquad \text{(Eq. 2.4)}$$

where ρ is the distance of point p_i to the origin. The intersection of the three curves in Hough space corresponds to the polar coordinates defining the plane spanned by the three points. The cells with the highest values represent the most prominent plane.

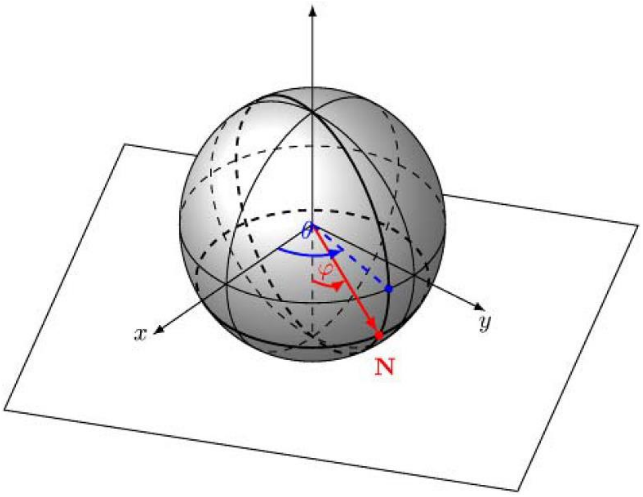


Figure 2.2 Normal vector described by polar coordinates (Borrmann et al., 2011)

Numerous studies based on HT focused on detecting planes in PCD. For example, Vosselman et al. (2004) and Vosselman (2009) employed HT-based methods to detect 3D roof planes in PCD. Likewise, Borrmann et al. (2011) suggested a Hough space accumulator structure for the detection of 3D planar structures. However, Tarsha-Kurdi et al. (2007) reported that plane detection rates using HT are lower than those using RANSAC.

3D planes still contain a moderate number of parameters. The Hough parameter space for plane detection is 3D, whereas for cylinder detection, the HT requires a 5D Hough parameter space. It is thus not computationally feasible to use HT directly for higher dimensional detection problems. Rabbani (2006) suggested a two-stage approach to detect cylinders in PCD to reduce the computational complexity as well as the number of dimension. This method transforms the 5D Hough space cylinder detection problem into a 2D and a 3D

Hough space problem. First, it uses Gaussian sphere of the PCD as its input to find strong hypothesis for cylinder orientation. It then performs 3D HT for a neighbouring direction found in first step, resulting in estimation of the position and radius (or diameter). Patil et al. (2017) modified Rabbani's method through an area-based adaptive method to estimate the cylinder orientation. Similarly, Ahmed et al. (2014) employed HT to detect straight cylindrical pipes through thin slices resampled from PCD. Tombari & Stefano (2010) proposed a Hough voting method which aims to accumulate evidence for the presence of the object being sought. In this method, an object is detected if it accumulates enough feature votes at a given position in 3D space. The position of the object is computed using a set of correspondences between the 3D model and the current scene.

In general, HT is a powerful tool for detecting simple geometric objects in noisy and cluttered PCD. However, HT is sensitive to parameter dimensions and cannot be applied in practice to shapes characterized by too many parameters, since this would cause a sparse, high-dimensional accumulator leading to poor performance and high memory requirements (Hassanein et al., 2015). This constraint impedes the use of HT in the detection of real bridge objects, which often contain skews and imperfections, and cannot be described using generic shapes with limited parameters.

The author concludes from the above that the abovementioned three main surface-based bottom-up methods (RANSAC, RG, and HT) are generally reliable for the detection of generic 2D and 3D object shapes in the presence of noise and outliers, but their high computational requirements render these methods less effective for detecting complex objects like real bridge components, which usually contain complex geometries. Aside from these computational intensive methods, another computationally more efficient method has drawn attention in the literature. The following section reviews this method.

Octree-Based (OB) methods have been proposed in the recent literature to tackle the issue of computational complexity and reduce the original point cloud size. Unlike prior work, OB methods do not seek to find properties for each individual point (e.g. surface normal, curvature and so on). Rather, they are based on a voxel data structure, processing cubes of data at a time.

The octree-based method relies on space subdivision techniques, which divide up the whole of the object space into regular or cubic voxels and in some way label each voxel according to object occupancy (Meagher, 1982). Traditional space subdivision techniques are costly in term of memory consumption. A way to reduce the memory cost is to impose a data structure on the basic voxel labelling scheme. Octree, is a popular method for organizing voxel data in a hierarchical data structure (Figure 2.3).

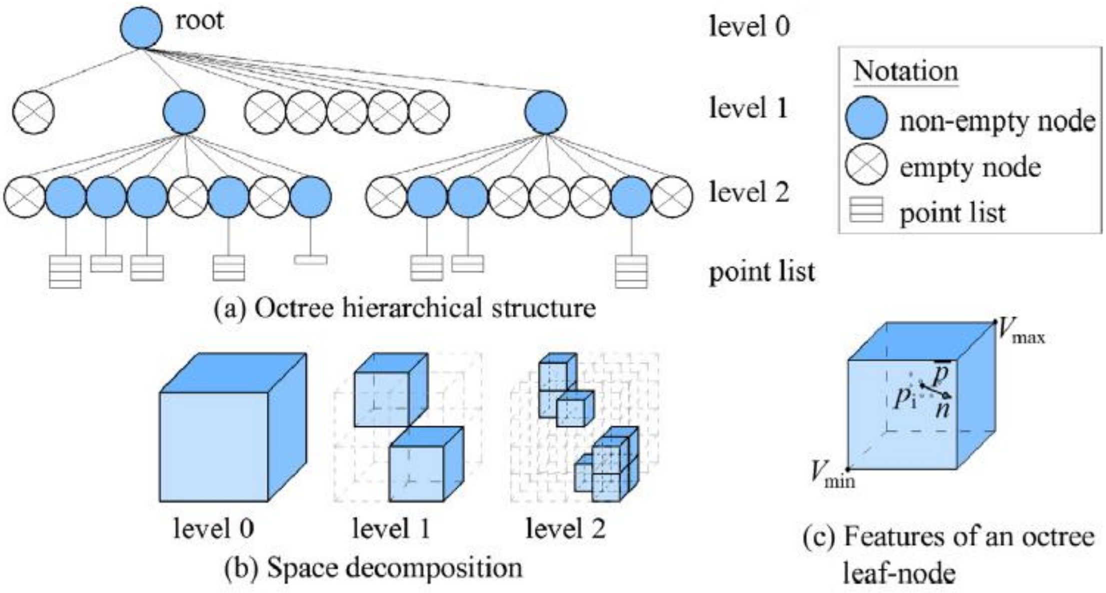


Figure 2.3 Octree representation (Vo et al., 2015)

Su et al. (2016) presented an OB segmentation method designed for piping systems (pipes, vessels, and walls). This method uses graph theory and a set of connectivity criteria to split and merge voxels into larger connected components. This bottom-up merging is performed recursively from the deepest tree level upward. Truong-Hong et al. (2013) introduced a technique to automatically extract building façade features in PCD for computational modelling. The authors proposed a Façade Angle algorithm to detect the rectangular shape of the opening through the combination of angle criterion, voxelization, and a Flying Voxel method (Truong-Hong, et al., 2012). Xu et al. (2018) suggested an OB probabilistic segmentation model for construction sites. The authors also partitioned the scene into voxels. However, the segmentation accuracy of this method is quite sensitive to the voxel size. This problem was discussed by Vo et al. (2015), who proposed an octree RG-based algorithm for surface patch segmentation in urban environments (Figure 2.4). Their method can semi-automatically adjust the voxel size using an adaptive octree, through which the processing time of surface segmentation is 40 times faster than the conventional point-based method suggested in (Rabbani et al., 2006). However, this method faces the difficulty of patch generation for low point density regions.

In general, voxel-based clustering is more computationally efficient than point-based clustering. Yet, voxel size determination remains largely a user-defined task. Its value depends on the point density as well as the desired level of detail (LOD) of output clusters.

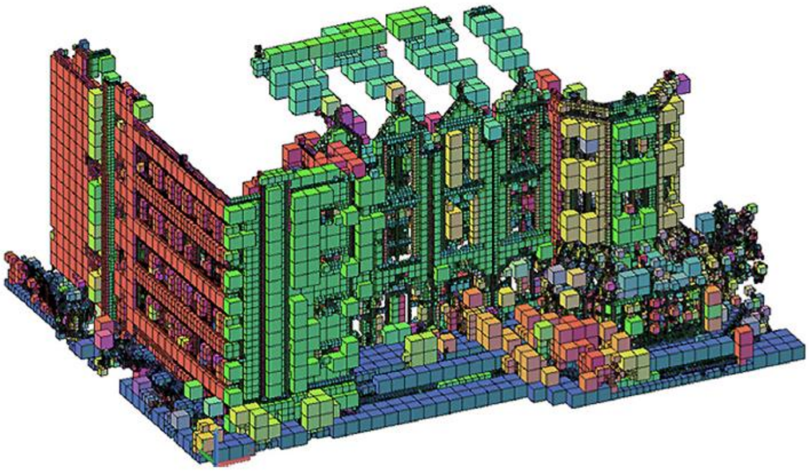


Figure 2.4 An adaptive octree (voxels are encoded in colours by normal vectors) (Vo et al., 2015)

2.1.2. Top-down detection

The author contends that bottom-up detection schemes are rarely suitable for point cloud classification. Classification through low-level primitives is insufficient since local surfaces, patches or voxels can be labelled as such, but it is difficult to determine whether they belong to the same instance. For example, a planar surface does not possess enough discriminating power to distinguish a wall from a cabinet (Rusu et al., 2009). The intervention of high-level information is required to overcome such challenges (Pinheiro et al., 2016).

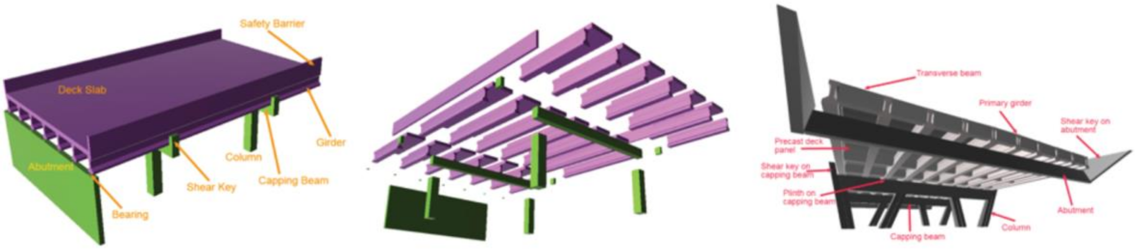
Research in neuroscience shows that, when we glance at a real-world scene, we can immediately understand and make highly accurate inferences based on our rich knowledge and expectations for that specific scene (Engel et al., 2001). The influences of the information beyond the scene are known as top-down influences (Corbetta & Shulman, 2002). The top-down approach is typically a heuristic approach for object detection. It begins with a broad-picture view and then is broken into compositional sub-problems that are easier to solve (Kokkinos et al., 2006). It usually combines a set of engineering criteria and classifies objects in PCD that meet the criteria. Prior studies show that knowledge-based classification methods are robust, as domain-specific information such as known parameters (e.g. diameter) or constraints (e.g. direction) (Ahmed et al., 2014), known object instances (Dore & Murphy, 2014; Pu & Vosselman, 2009), and topological relationships (Koppula et al., 2011), are invariant to factors such as pose and appearance.

A pioneering study using a top-down modelling approach was REFAB (Reverse Engineering FeAture-Based) (Thompson et al., 1999), which uses geometric constraints such as parallelism, concentricity, perpendicularity and symmetry) to convert points to mechanical solid models. Similarly, the method presented in (Su et al., 2016) uses a set of connectivity criteria such as proximity, orientation, and curvature to merge and label industrial components (pipe, vessels, and walls) across voxels. Ahmed et al. (2014) used the pre-knowledge of the diameters and the approximate number of pipes in the scene to facilitate detection. Similarly, Son et al. (2013) proposed a knowledge-based method for detecting industrial plant objects based on the known surface curvature and size of the pipelines. Perez-Gallardo et al. (2017) suggested a semantic model-based system to detect four object classes (pipe, plane, elbows, and valves) in an industrial scene using topological information. Dore & Murphy (2014) suggested a semi-automatic approach to generate façades gDTs of existing historic buildings. This method develops a shape library containing parametric classical architectural objects whose geometric parameters such as shape or size can be altered. After the façade model is automatically generated, the initial position and size of the façade elements are estimated followed by user manual editing of the object sizes. Likewise, Laefer & Truong-Hong (2017) leveraged the steel standard library to identify and match the cross-sections of steel frames in PCD. Nüchter & Hertzberg (2008) used a relationship reasoning network for semantic mapping. Their network uses a set of pair-wise relationship rules such as parallel, equal height, above, under, and orthogonal to coarsely classify the major planes (wall, ceiling, floor and door) in an indoor PCD. Wang et al. (2015) employed a rule-based building envelop component classification algorithm to categorize each boundary surface-based point cluster into its corresponding category.

Recent research also relies on existing as-designed documents to inform the top-down modelling strategy, which can simplify point cloud clustering and classification tasks (Liu et al., 2012). This is because the information gleaned from prior data can serve as guidance that shifts the focus from object detection to matching between a point cloud and existing models (Bosché, 2010; Yue et al., 2006). Belsky et al. (Belsky et al., 2014) encapsulated domain expert knowledge in the form of rule sets in order to infer and enrich semantics for a building geometric model.

However, the methods developed in the above-mentioned studies are tailored for buildings, indoor environments, and industrial objects. They are not invariant to their use case and not tailored for use in bridge settings, as the geometric properties of bridge components are quite different than in connectivity and appearance to objects in other types of infrastructure. What is more, there are few as-built or as-is models for existing bridges so that little prior knowledge of the embedded objects in a point cloud is available.

Recently, some studies have started to employ top-down strategies to detect bridge components in PCD. For instance, Riveiro et al. (2016) used specific topological constraints to segment masonry bridge PCD through point normal clustering. However, their algorithm is based on histograms that largely depend on data quality. This means that the method may not generate informative histogram features if the point density of some piers or walls is much lower than that of the other components. Thus, it is difficult to generalize this algorithm to large RC bridges, as real PCD usually suffer from occlusions and non-uniformly-distributed points. Like the previously mentioned work (Nüchter & Hertzberg, 2008), Ma et al. (2017) leveraged relationship knowledge and shape features to classify bridge 3D solid objects in a gDT (Figure 2.5). First, the input of this method needs to be a bridge gDT (not a bridge point cloud) without any semantic meaning. Second, the method assumes that the bridge gDT is developed in a grid system, in ideal geometries and that the pairwise relationship between two 3D solid objects is well defined. These assumptions are quite restrictive and make the method less feasible for real cases, as bridges usually possess various curved horizontal and vertical alignments and cross-sections.



(a) A synthetic bridge model

(b) A real RC bridge model

Figure 2.5 Bridge object classification using pairwise spatial relationships (Ma et al., 2018)

2.1.3. Other detection methods

Data-driven, learning-based methods have been widely used to predict unknown instance labels based on training feature sets and manually added labels that facilitate supervised learning. Xiong et al. (2013) proposed a probabilistic graphical model to label the extracted planar surfaces of buildings. Armeni et al. (2016) used a Support Vector Machine detector through a sliding window to identify whether there was an object class of interest in a predefined box in space in a building PCD. Zhang et al. (2014) used a synthetic training set to train a multi-class Adaboost decision tree classifier, which can assign bridge component labels to surface primitives. As in the previous cases, this method was designed for simplified bridge designs that do not contain skews, irregularities or complex objects, which is often the case for real bridge components.

Numerous volumetric Convolutional Neural Network (CNN) and Deep Learning frameworks have been proposed by transforming points into voxel grids. For example, Maturana & Scherer (2015) proposed an occupancy grid representation with a supervised 3D CNN called *VoxNet* to classify objects from the volumetric data. Qi et al. (2016) suggested a volumetric and multi-view CNN for object classification in PCD. Likewise, Tatarchenko et al., (2017) proposed a deep convolutional architecture which can generate 3D outputs by using an octree representation. Instead of transforming a point cloud into regular 3D voxel grids, Qi et al. (2017) introduced a novel deep neural network called *PointNet*, which can directly consume points. It is worth noting that complex neural networks such as CNNs and Deep Learning frameworks have millions of parameters. Complex models have been proven to have better accuracy, but they may able to memorize the data and run into an overfitting problem if a massive amount of data is absent (Lecun et al., 2015). The author therefore contends that the major restrictions to applying these data-driven machine learning schemes to bridge component detection tasks include: (1) the lack of a sufficient number of labelled large-scale real bridge PCD to train a good model, and (2) the high computing burden. These methods usually require a substantial down-sampling task before they can be used even in high performance computing systems (e.g. via Google's TensorFlow).

2.2. Model Fitting to Point Cluster

The process of fitting a 3D solid model to a labelled point cluster aims to transform a point cluster of unordered spatial points into a structured, information-rich 3D representation (Watt, 2000). In order to generate a meaningful DT of a real-world asset, the 3D representation should be in an object-oriented data format and contain a range of attributes such as geometries, materials, defects, and so on. This research only focuses on the geometric modelling, i.e. only the shape and size are taken into account to describe a gDT.

In the AEC/FM sectors, the widely used data exchange format of product models is Industry Foundation Classes (IFC). However, IFC is too large and complicated to be described in this thesis in full. The author first provides a detailed review of the existing model fitting techniques in Section 2.2.1 and then, an introduction of IFC and IFC geometric representation are given in Section 2.2.2 and Section 2.2.3, respectively.

2.2.1. Model fitting techniques

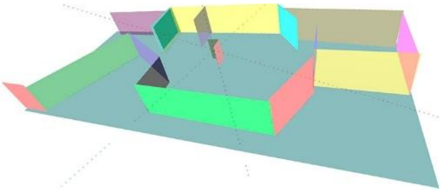
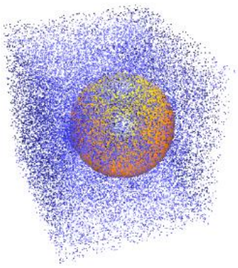
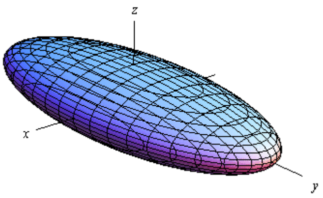
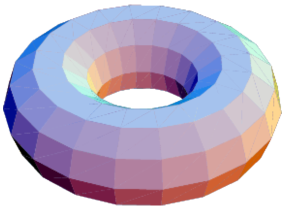
Model fitting aims to fit a 3D model to a point cluster. In other words, model fitting is modelling: the process of leveraging computer graphic techniques to form the 3D shape of a point cluster. The point cluster herein is the output of the previous object detection step, which is the subpart of a point cloud. The 3D object is approximate in the sense that it describes the geometry or the shape of a point cluster to produce its digital 3D representation to an acceptable quality based on the specific required level of detail.

Digital representations of objects have been intensively studied and remain very much an unsolved problem in computer graphics (Akenine-Möller et al., 2018). This is because there is no universal solution to describe an object. Different representational methods have their advantages and disadvantages. How to choose a representation totally depends on (1) the nature of the object being modelled, (2) the particular modelling technique that we choose to use, and (3) the application scenario where we bring the object to life. In the context of bridge gDT generation, for the ideal is a representation that is cost-effective and can describe the geometric property of a bridge component as well as possible so that the produced model can be used by end-users in different downstream applications.

Existing shape representation methods can be categorized into four groups: Implicit Representation, Boundary Representation, Constructive Solid Geometry, and Swept Solid Representation. The author reviews each of these in the following text.

2.2.1.1. Implicit Representation

One solid modelling approach is based on the representation of 3D shapes using mathematical formulations, i.e. implicit functions. Using a single real-valued function of three variables in computer-aided geometric modelling, an arbitrary constructive solid can be defined as $f(x,y,z) \geq 0$ (Rvachev, 1963, 1974, 1982) and its surface as $f(x,y,z) = 0$ (Ricci, 1973). Common implicit surface definitions include, but are not limited to, the following (Table 2.1):

Shape	Equation	Example
Plane	$ax + by + cz = d$ (Eq. 2.5)	 (Limberger & Oliveira, 2015)
Sphere	$x^2 + y^2 + z^2 = r^2$ (Eq. 2.6)	 (Schnabel et al., 2007)
Ellipsoid	$(\frac{x}{r_x})^2 + (\frac{y}{r_y})^2 + (\frac{z}{r_z})^2 = 1$ (Eq. 2.7)	 (Weisstein, 2018a)
Torus	$(\sqrt{x^2 + y^2} - R)^2 + z^2 = r^2$ (Eq. 2.8)	 (Weisstein, 2018d)

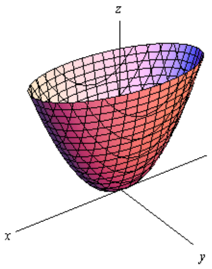
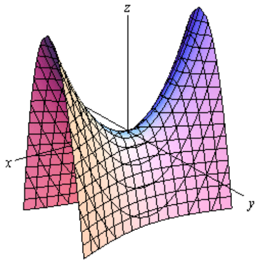
Elliptic paraboloid	$z = \frac{x^2}{a^2} + \frac{y^2}{b^2}$ (Eq. 2.9)	 (Weisstein, 2018b)
Hyperbolic paraboloid	$z = \frac{x^2}{b^2} - \frac{y^2}{a^2}$ (Eq. 2.10)	 (Weisstein, 2018c)

Table 2.1 Common implicit surfaces

Apart from the examples presented in the above table, Gerardo-Castro et al. (2014; 2013) also leveraged Gaussian Process (Rasmussen & Williams, 2004) Implicit Surface to reconstruct a surface using only a small subset of available points.

Implicit surfaces have difficulty with describing the sharp features such as edges and vertices, although they are efficient at checking whether a point lies inside, outside, or on the surface (Song & Jüttler, 2009). Given that only a very limited number of primitives can be represented exactly by algebraic formulations, implicit functions are of limited usefulness when modelling real-world 3D objects such as bridge components, as they do not take idealized shapes. In addition, real bridge components contain defects that further reduce the effectiveness of these implicit representations. There is a trade-off between the accuracy of the representation and the bulk of information used for shapes that cannot be represented by mathematical formulations. The author elaborates three other basic modelling types: Boundary Representation (B-Rep), Constructive Solid Geometry (CSG), and Swept Solid Representation (SSR), in the following sections. The review mainly focuses on geometric modelling using PCD in the DT-related literature.

2.2.1.2. Boundary Representation

Boundary Representation (B-Rep) is a method for describing shapes using their limits, i.e. boundary surfaces. Thus, B-Rep can be considered an extension to the wireframe model used in (Xiong et al., 2013). The model represented using B-Rep is an explicit representation, as the object is represented by a complicated data structure giving information about each of the vertices, edges, and loops and how they are joined together to form the object (Figure 2.6). The geometry of a vertex is given by its (x, y, z) coordinates. The edges are straight or curved lines. A face is represented by some description of its surface (algebraic or parametric forms can be used). For example, a flat quadrilateral is made up of four vertices joined by four straight lines or a bi-cubic parametric patch (Watt, 2000). A curvilinear quadrilateral is made up of four vertices joined by four cubic curves (Dimitrov et al., 2016). Kwon et al. (2004) introduced a rapid and accurate (precision error < 5%) local spatial modelling algorithm to fit sparse PCD to planes, cuboids, and cylinders in B-Rep, assuming that a construction site consists of these primitives. Valero et al. (2012) developed a method to yield B-Rep models for indoor planar objects (walls, ceilings and floors) with high modelling precision (between 0.8 cm and 1.88 cm for vertical and horizontal edges) (Figure 2.7 (a)). Based on this work, Valero et al. (2016) then upgraded their method to detect more objects, such as tables, chairs and wardrobes, in an indoor environment through a technology called radio frequency identification (RFID) (Figure 2.7 (b)).

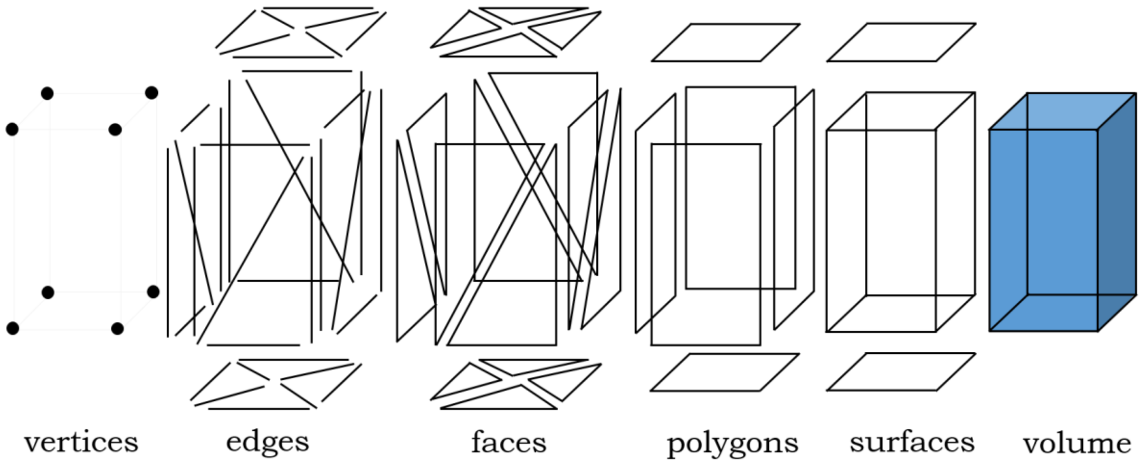


Figure 2.6 B-Rep object consists of different types of element

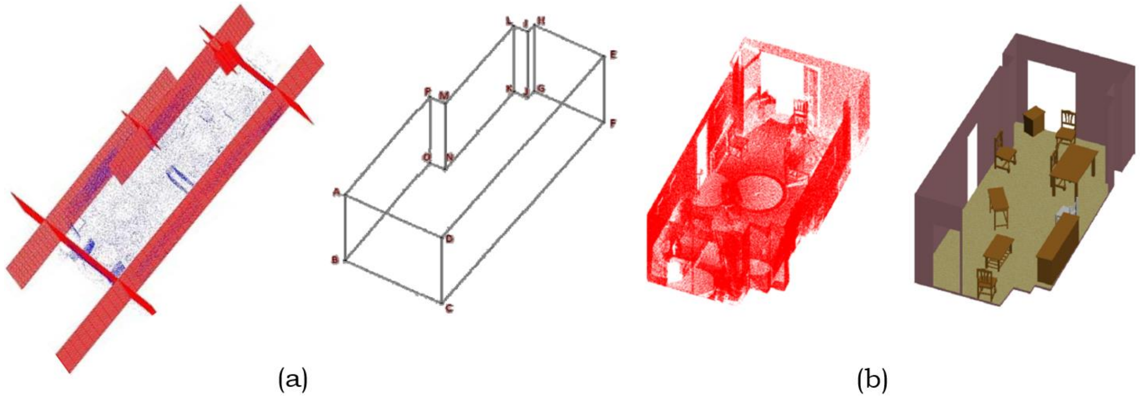


Figure 2.7 (a) Plane fitting to the walls and plane intersections (left), and labelled vertices of the room (Valero et al., 2012); (b) point cloud (left) and the produced 3D model (Valero et al., 2016)

Both Tessellated Surface Representation (TSR) and Polygon/Mesh Representation (PR/MR) can be considered as B-Rep types. A final model of TSR or PR/MR, is represented as a collection of connected surface elements. Oesau et al. (2014) leveraged a graph-cut formulation and HT to reconstruct a synthetic building PCD into a mesh-based model (Figure 2.8). It is possible to store the surface triangles as IFC objects using the coordinates of their vertices, although in this work the authors did not discuss the conversion from the mesh model into a component-level resolution gDT. However, representing an object using polygon facets is a low-level machine representation. It is inconvenient for practitioners to use directly. Further processing is necessary in order to acquire a more advanced representation of gDT such that component-level attribute information can be added. It is worth noting that PR/MR is capable of modelling arbitrary geometry. Representing an object using polygonal facets or polygon mesh is actually the most popular representation in computer graphics (Vince, 2013). It is capable of modelling subtly shaped objects such as the human head (Jeong et al., 2002) with a net of patches or mesh. However, there are certain difficulties with polygon mesh models.

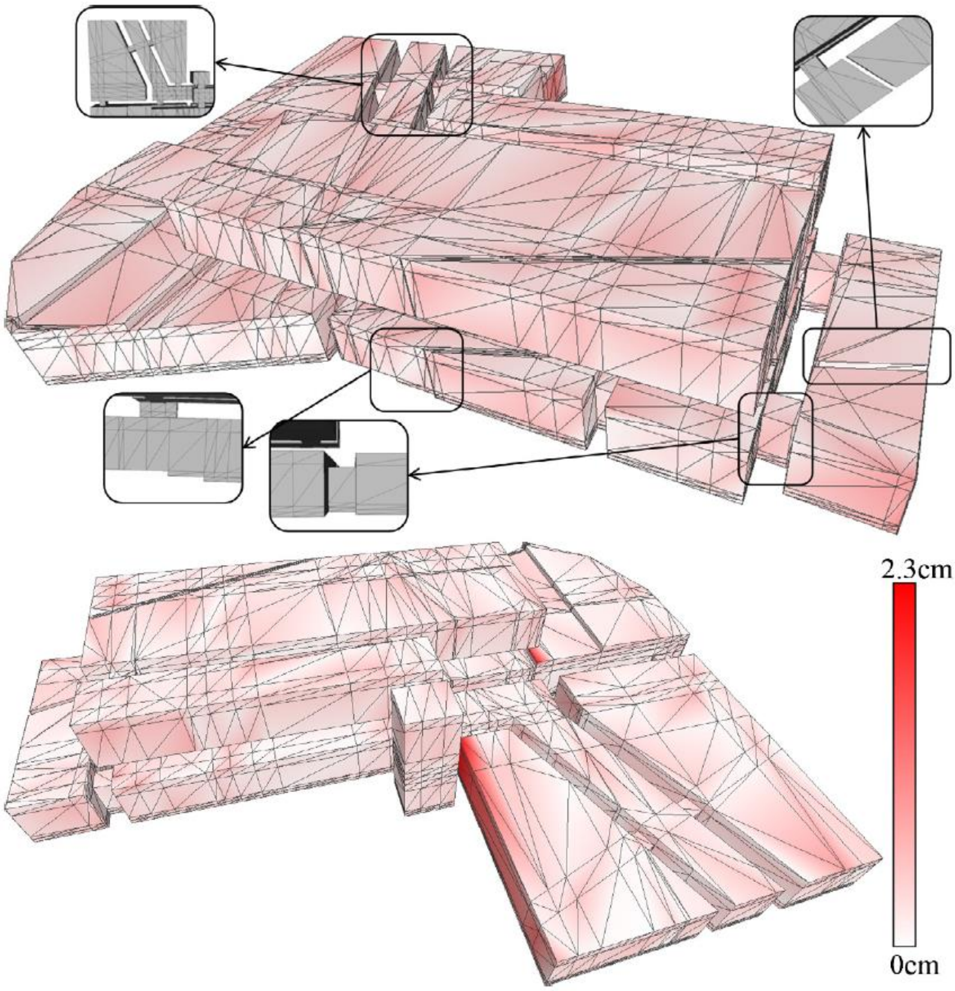


Figure 2.8 A triangle mesh model reconstruction from a synthetic point cloud dataset. Two views of the same reconstruction are shown, coloured by the Hausdorff distance from the result to the ground truth (Oesau et al., 2014).

Foremost among these difficulties is the **(1) level of detail (LOD)**, which depends on the number of polygon facets used, and is arbitrarily accurate (Hoppe et al., 1992). A minor edge displacement may destroy the whole visual representation of a continuous surface with a low-resolution polygon mesh model. However, the complexity is considerably higher for a higher-resolution model (Luximon et al., 2012). High resolution affects the rendering time and model creation cost, and the results can be unduly complex. For instance, it is possible to use thousands of polygon facets to represent the natural surface of a bridge component, and then store them as tessellation models. However, is this really necessary? When modellers manually digitize real bridge-components using modelling software packages, they normally work with an application that does not demand subtle undulations of the object surface (Chapter 1, Section

1.3.2). In addition, PR/MR is still a very unstructured data structure. It is more convenient for end-users of the resulting gDTs such as engineers, inspectors, and decision-makers, to work with a more smoothed solid model consisted of structured data, which can facilitate manipulation, comparison, analysis, edition, and information enrichment. It is difficult to achieve these goals with mesh-based models. An option is to reduce the polygon resolution without degrading the rendered presentation. But by how much?

This question tends to be related to the input points used for creating the mesh, because the size of individual polygons depends on the local spatial curvature (Eck et al., 1995). When the curvature is high, the rate of polygon generation needs to be increased. That is to say more polygons per unit area of the surface occur when the curvature twists rapidly, and vice versa. Point Cloud Subsampling techniques are investigated in (Zhang & Tang, 2015) and (Chen et al., 2017). Zhang & Tang (2015) proposed a visual complexity analysis algorithm to detect discontinuous locations where detailed laser scanning is required. Chen et al. (2017) examined a PCD compression method that enables automatic compression of a point cloud with varying subsampling rates according to the geometric complexities of parts of the data, and a user-specified compression ratio. Complicated segments are assigned higher compression ratios to retain more data, and vice versa. Generating a customized resolution polygonal model through a targeted LS process and a smart PCD compression method could be a very promising alternative, but further steps are still necessary to render higher-order component-level information (EURs 1 and 2).

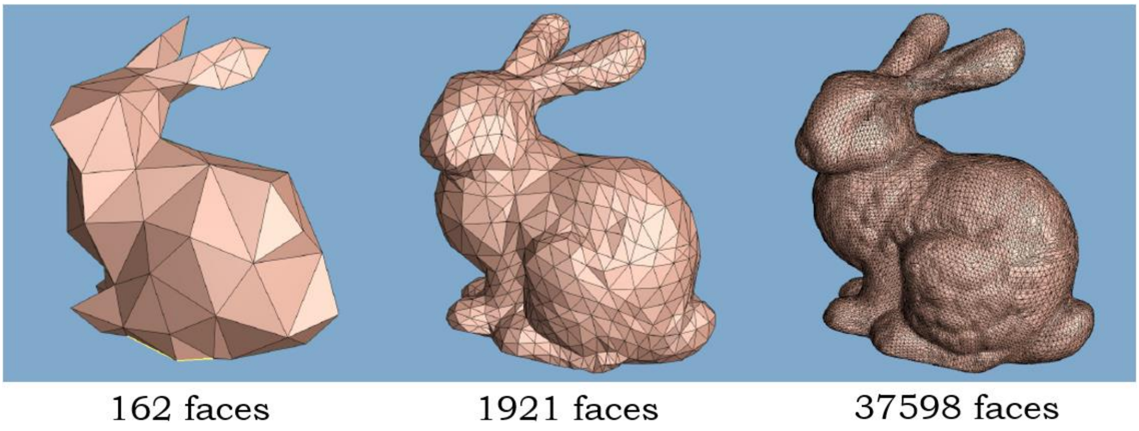


Figure 2.9 Multiresolution of meshes (Eck et al., 1995)

Secondly, polygon mesh models based on point clouds suffer from issues such as **(2) missing data points, i.e. occlusions**. Occlusions commonly occur in the scanning process due to limited line-of-sight of the laser sensor, where several portions of the scanned object are not sampled. A large occluded region is hardly smoothed (Carr et al., 2003). As a result, PR/MP does not guarantee that a group of polygons facets can form a closed mesh model.

Thirdly, mesh representation is a surface representation, offering **(3) no sense of volume** (Oesau et al., 2014). This restricts the volumetric and global property analysis (EUR 2). For example, it is more difficult to extract geometric properties such as the radius of a cylindrical column on a mesh representation. This limitation renders model comparisons difficult for practitioners between two points in time. Component level comparison is hard to achieve (EURs 1 and 2) for both model calculations and model editions without a volumetric model. In summary, the author contends that the use of polygon mesh models is not the most suitable way digitize the bridge component geometry.

2.2.1.3. Constructive Solid Geometry

Constructive Solid Geometry (CSG) is a high-level volumetric representation that works both as a shape representation and a record of how an object was built up (Deng et al., 2016). The final shape can be represented as the combination of a set of elementary solid primitives, which follow a certain "logic". The primitives can be cuboids, cylinders, spheres, cones, and so on. When building a model, these primitives are created and positioned, then combined using Boolean set operators such as union, subtract, intersect and so on (Figure 2.10).

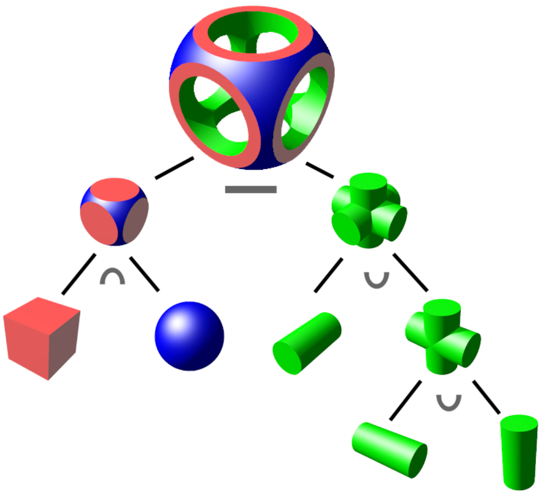


Figure 2.10 A CSG tree, where leaves represent primitives and nodes represent operations. The nodes are labelled ∩ for intersection, ∪ for union, and − for subtraction. (Image: Wikipedia, 2018)

For simple primitives, both the methods proposed by Rabbani (2006) and Patil et al. (2017) can be used for modelling a cylindrical piping system. The random sampling method of Schnabel et al. (2007) can be used to automatically model objects composed of five basic shapes (planes, spheres, cylinders, cones, and tori). This method can obtain a representation solely consisting of shape proxies. Walsh et al. (2013) developed a shape library containing generic description of objects (e.g. cuboid, cylinder) to fit point clusters using surface fitting in a least squares sense. The reliability of this method subjects to real data is unclear as a quantitative assessment of the fitting performance is not given in the work. Rusu et al. (2008) proposed a model fitting module to fit kitchen objects (e.g. cupboards and appliances) using 3D cuboids (Figure 2.11).

Their algorithm replaces segmented planar surfaces with polygons that are then fitted with 3D rectangular parallelepipeds by projecting the vertical polygons onto the walls and satisfying a size criterion.

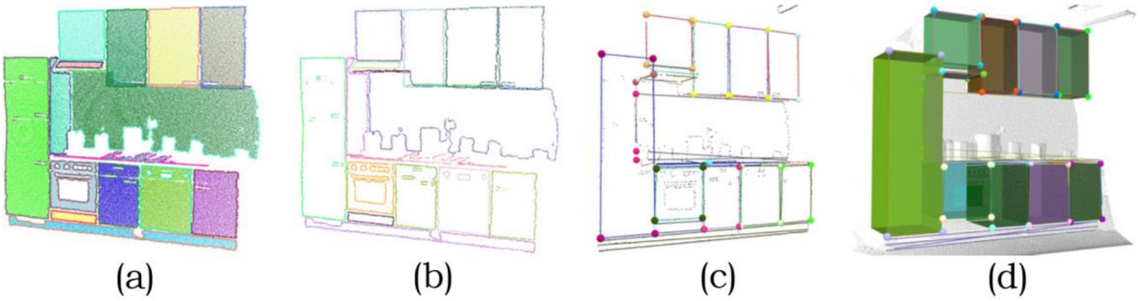


Figure 2.11 (a) Segmented planar regions; (b) Boundary points; (c) Best fitted lines to each region boundary; (d) Cuboid fitting (Rusu et al., 2008)

Similarly, Xiao and Furukawa (2012) introduced an algorithm called "inverse CSG" to reconstruct large-scale indoor environments with a CSG representation consisting of volumetric primitives by imposing regularization constraints (Figure 2.12). This method uses only cuboids as volumetric primitives, assuming that they are the most common shapes found in indoor walls. Further extension of geometric primitive types is necessary as construction elements may contain other shapes.

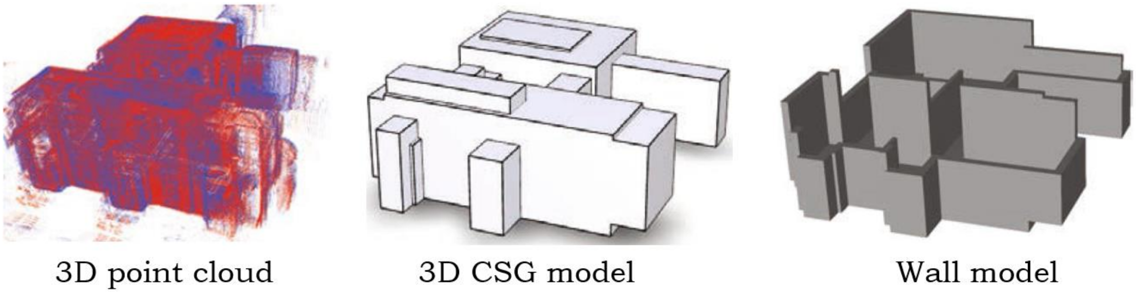


Figure 2.12 CSG representation of a museum (adapted from Xiao & Furukawa, 2014)

Zhang et al. (2014) designed a multi-class Adaboost decision tree classifier from surface primitive features to classify both infrastructure components (pier, beam, deck etc.) and 3D shape entities labels (cuboid, cylinder, sheet etc.). The final outputs were 3D solid objects in IFC format (Figure 2.13), although the authors did not provide any details about the IFC entities. Additionally, this method is tailored for idealized or simplified topology designs that do not consider the real geometries of bridge components. For example, a real sloped slab with varying vertical elevation cannot be simply modelled by a single sheet.

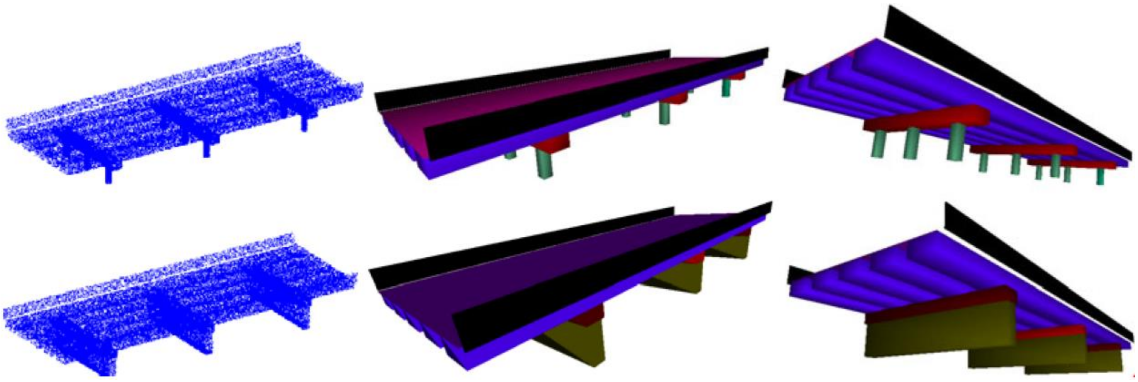


Figure 2.13 Fitted IFC entities in synthetic bridge PCD (Zhang et al., 2014)

The power of Boolean operations of CSG representation is demonstrated in Figure 2.10. However, CSG does suffer from serious drawbacks. With the Boolean operation alone, the range of shapes to be modelled is severely restricted and it is difficult to construct unusual shapes. The high-level nature of CSG representation imposes a high burden on the pre-design for irregular objects. As a result, CSG may be less suitable for representing real bridge components, which are more complex than simple primitives, such as cuboids and cylinders. In addition, a detailed modification to one face of the primitive cannot be easily implemented by using set operations.

2.2.1.4. Swept Solid Representation

Swept Solid Representation (SSR) or Extrusion is a representation of model which creates a 3D solid shape by sweeping a 2D profile that is completely enclosed by a contour line along a specific path in the third dimension (Figure 2.14). The enclosing contour would be a combination of contiguous lines, arcs, and polylines. The sweeping line could be a straight line perpendicular to the contour surface, or it could be a curve in 3D space. As the solid extruded volume is swept, the contour surface could remain fixed or it could be scaled (Figure 2.14). This representation provides geometric information about the profile, beginning and end points, between which the 2D contour surface is extruded. Budroni and Böhm (2010) suggested a plane-sweep-based detection method to extrude planar elements (such as walls) in indoor environments (Figure 2.15).

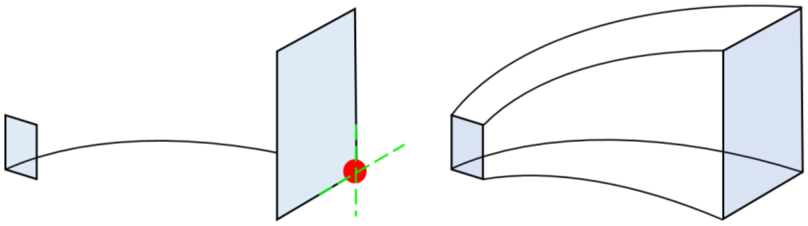


Figure 2.14 Swept solid extruded between scaled 2D contour surfaces

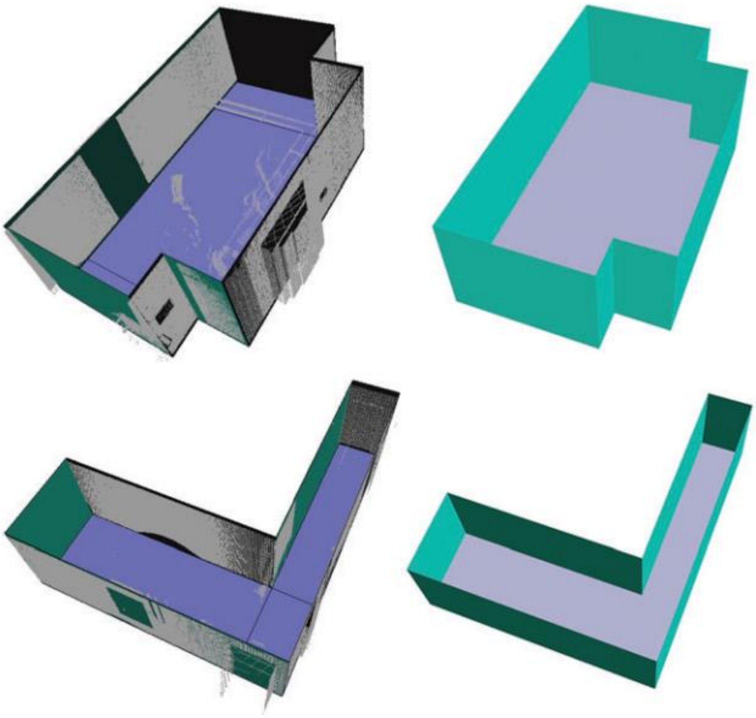


Figure 2.15 Plane sweeping detection of indoor walls (Budroni & Boehm, 2010)

Likewise, Ochmann et al. (2016) presented an automatic approach for reconstructing parametric 3D building models from indoor PCD. Their output model enables high-level editing operation, meaning that users can easily operate wall removal or room reshaping through a user-friendly interface (Figure 2.16). This property is especially useful for design, renovation or retrofit projects, but is not suitable for modelling bridge gDT, where many non-planar elements exist. Laefer & Truong-Hong (2017) introduced a kernel-density-estimated-based method to identify the cross-sections of steel beams in PCD. A cross-section is matched to one real steel type from a steel standard library and is extruded along the alignment of the cross-section (Figure 2.17). However, this method does not take the as-damaged shape into account and produces the steel beams in perfect condition.

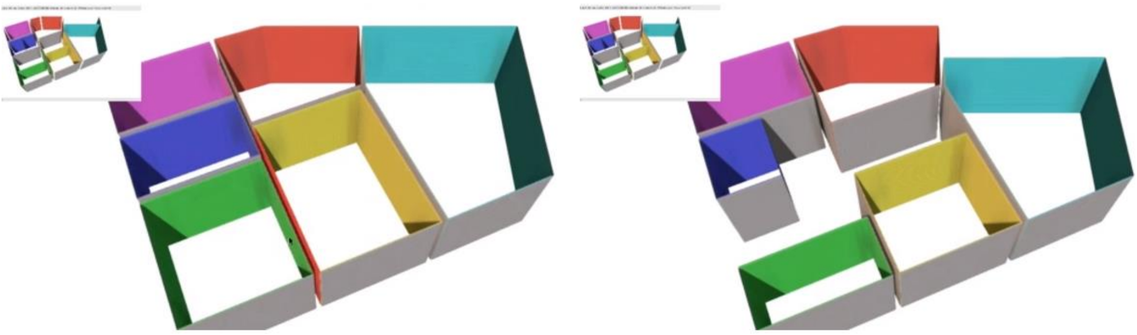


Figure 2.16 A parametric indoor model (left); manipulation of wall removal, reshaping and displacement (Ochmann et al., 2016).
Video can be accessed from:
https://www.sciencedirect.com/science/article/pii/S0097849315001119

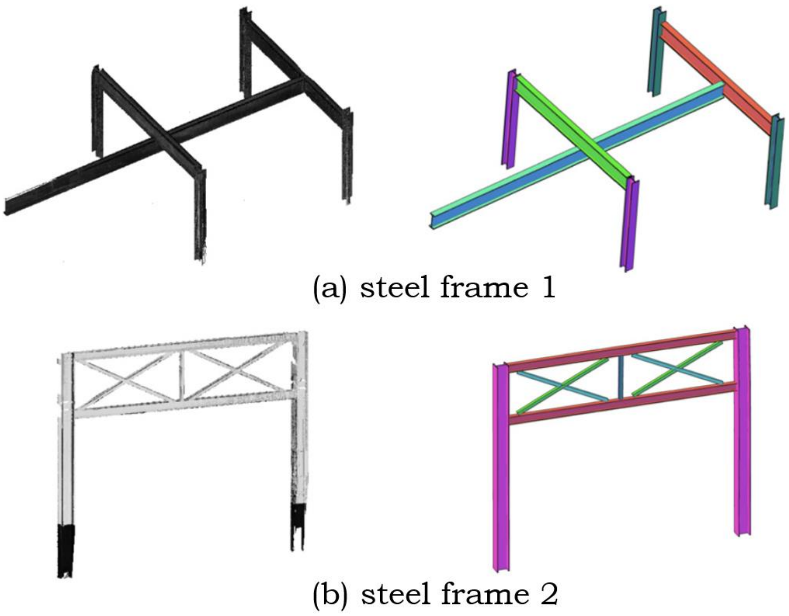


Figure 2.17 3D as-is BIM model generation of two steel frames (Laefer & Truong-Hong, 2017)

The author therefore conclude that the sweeping method has been studied in several recent studies. However, its application is limited to indoor planar and industrial standardized elements. Its implementation has not yet been explored in bridge gDT generation. In addition, none of the existing methods of the PCD-to-gDT process have explained explicitly how to represent the generated model in IFC format (Zhang et al., 2014). In the next section, the author introduces IFC and Model View Definition in the context of bridge modelling.

2.2.2. IFC and Model View Definition

The author stated earlier in this thesis that bridge DT generation is conventionally termed as BrIM (Chapter 1, Section 1.2), an advanced modelling approach that is based on the broader definition of the bridge geometric model – “objects” that make up the physical bridge asset. Bridge DT is a holistic digital representation of physical and functional characteristics of the bridge, which provides a shared knowledge resource for information to support a reliable basis for decisions during its life-cycle (Eastman et al., 2011; Fanning et al., 2014). In order to support its use in the bridge industry, all associated life-cycle information of a bridge should be represented in a neutral, readily understandable, computer-interpretable form that remains sufficient as well as consistent when stored and exchanged. Using such a standard for representing bridge information in a digital format which can be rapidly adopted by existing software tools with minimal ambiguity will offer the opportunity for use in digital project delivery, 3D visualization, automated machine control, network-level study, and more, as a routine part of project development and asset management (Venugopal et al., 2012).

IFC is an object-based file format with a data model developed by buildingSMART (2018c). It is a data representation standard and file format used to define construction-related CAD graphical data as 3D real-world objects (similar to the .dxf format of AutoCAD). Today, IFC is supported by about 150 software applications worldwide to enable better work flow and interoperability in the AEC/FM industry (buildingSMART, 2018).

IFC provides a set of definitions for all object element types encountered in the AEC/FM sectors and a text-based structure for storing those definitions in a data file, based on an open data exchange standard, i.e. the IFC schema. As a result, the content of an IFC file heavily depends on the modelling technique adopted by the modellers, and on the export capabilities of the authoring tool of modelling software packages (Belsky, 2015). IFC requires a hierarchical data structure to make the model exchange to be meaningful for importing tools. It defines three basic components for modelling constructions: objects, relationships, and properties. An object is an abstract super-type entity, *IfcObject,* structured in an order hierarchy. An instance of the entity is used to represent a real-world object. The concept of relationships is the objectified relationship, *IfcRelationship*, relating different objects to each other, and the property definition, *IfcPropertyDefinition*, is the generalization of all characteristics and context information that can be added to an object.

In the infrastructure sector, the data exchange solution for bridges is lagging far behind that for buildings, despite of the DT adoption for infrastructure projects having increased in the last few years (Buckley & Logan, 2017) (Chapter 1, Section 1.2). Much effort has been devoted to the development of IFC-Bridge related extensions. Yabuki et al. (2006) and Lebegue et al. (2012) are two examples that focus on the IFC standard extension for bridges. Lee & Kim (2011) introduced an IFC extension plan targeting integrated roads, bridges, and tunnels, in which they focused on the enrichment of spatial elements. Ji et al. (2013) introduced an extension to the IFC-Bridge format, providing a means of interchanging parametric bridge models. They describe in detail the necessary entities introduced to define parameters and capture dimensional and geometric constraints. Likewise, Amann et al. (2015) suggested a generalized alignment model that can be extended with cross sections to describe a road body. This model can be further used for other product data models of linear infrastructure contractions, such as tunnels and bridges.

However, the above-mentioned bridge-related IFC extensions have not yet been fully integrated into the official release of the IFC schema (Figure 2.18). This is mainly because complete standardization of DT generation is a large multi-year, national, and international effort. The planning and execution of such an undertaking can vary widely in scope, cost and temporal effectiveness (Ismail et al., 2016). For example, adopting the Precast/Pre-stressed Concrete Institute Model View Definition took more than half a decade to fully execute (Belsky et al., 2014; Panushev et al., 2010). The following paragraph explains what the Model View Definition is and why we need it.

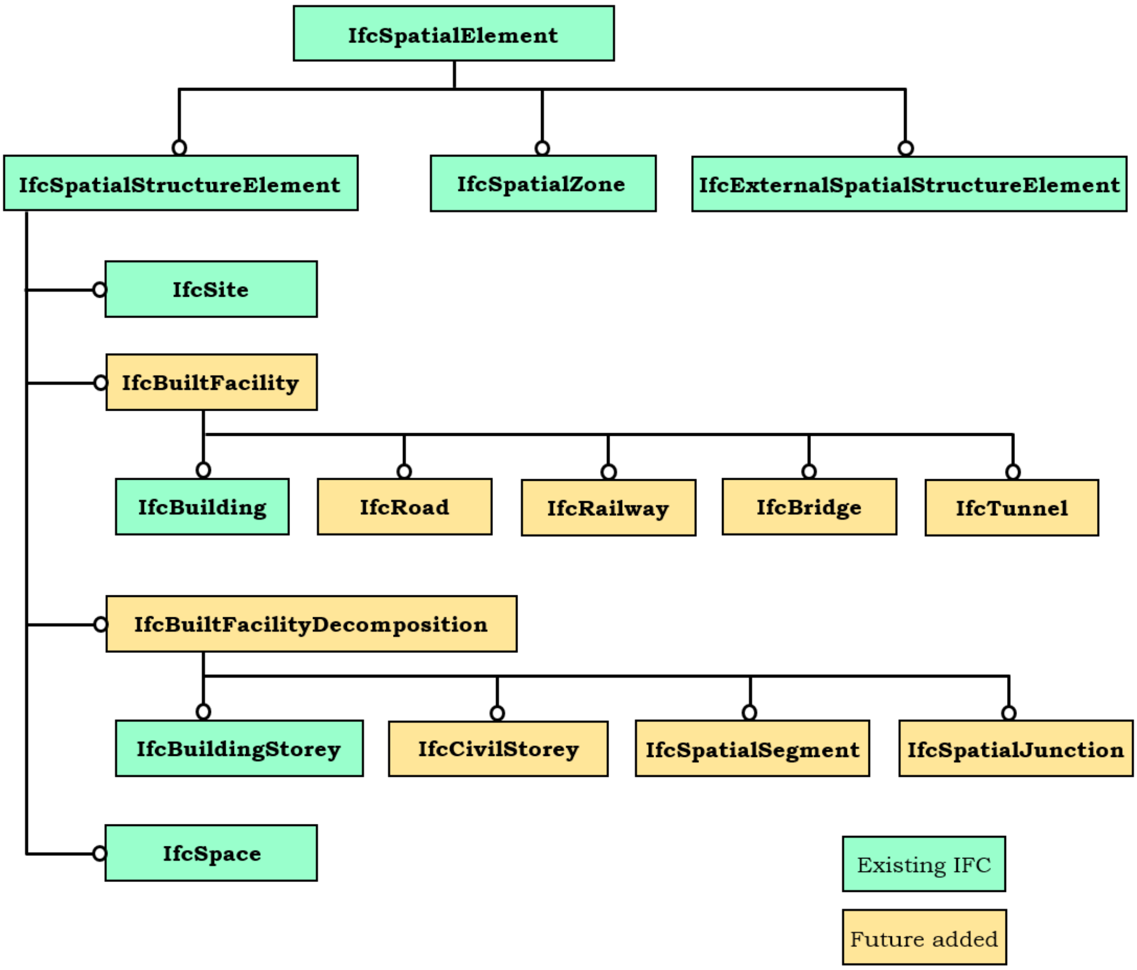


Figure 2.18 Proposed changes to the spatial structure to support infrastructure (Borrmann et al., 2017)

IFC is very large and defines a detailed schema of roughly 800 data types for representing building objects, their relationships, and associated lifecycle information (e.g. tasks, resources, systems, requirements, and project participants). This huge size may deter usage by newer vendors and render data exchange unreliable. This is due to inconsistencies in the assumptions that different implementers of exchange functions make about how information should be interpreted (Sacks et al., 2010). Venugopal et al. (2012) listed four different ways to represent a floor slab entity in a parking garage using IFC schema, depending upon the context and the LOD required. This example demonstrates the richness of IFC, its redundancy, and the inconsistency. To circumvent the inconsistency problem, a layer of specificity is needed to select and specify the appropriate information entities and their attributes from a schema for a particular context (Eastman et al., 2011). Specific uses of IFC have been narrowed to smaller subsets using a fraction of the data definitions, called Model View Definitions (MVD), which has also been done for bridges as part of this effort. MVD describes the subset of a data schema that is required to exchange the data required in specific data exchange scenarios.

The SeeBridge research team compiled an Information Delivery Manual (IDM) (Sacks et al., 2018) to ensure that the final bridge DT would be sufficiently semantically meaningful to provide most of the information needed for subsequent bridge repair, retrofit and rebuild work. Based on this IDM, a MVD (Sacks et al., 2018) was then delivered that defines the information concepts needed, and suggests a binding to the IFC4 Add2 standard for exchange of bridge DTs. The SeeBridge IDM and MVD approach is defined in a buildingSMART International Standard (2018) and has been used in different DT generation interoperability research projects (Sacks et al., 2010; Venugopal et al., 2012).

2.2.3. IFC Geometric Representation

The range of possible information to be exchanged in the AEC sector is huge. The IFC coverage increases along with the user and developers' needs. There is no universal definition of what information a bridge DT must provide as the proper information heavily depends on its application scenario. The fundamental feature of DTs is the 3D geometry, without which many DT applications do not exist. However, the 3D geometry on its own is not sufficient to provide a meaningful DT, which also needs to convey semantics. This means that all objects constituting a DT possess a meaning, i.e. they are instances of object types such as a Pier, Girder, or Deck Slab and so on. Thus, bridge DTs are models comprise both the 3D geometry of the bridge components as well as a comprehensive set of semantic information, including materials, functions, and relationships between components. This section focuses on principles involved in representing IFC geometry and the most important geometry representations.

According to Borrmann et al. (2018), an object in a DT is initially described as a semantic identity and can then be linked with one or more geometric representations. This ability allows objects in a DT to use application-specific geometric representations. This also provides flexibility to link one or more geometric representations with a semantic object.

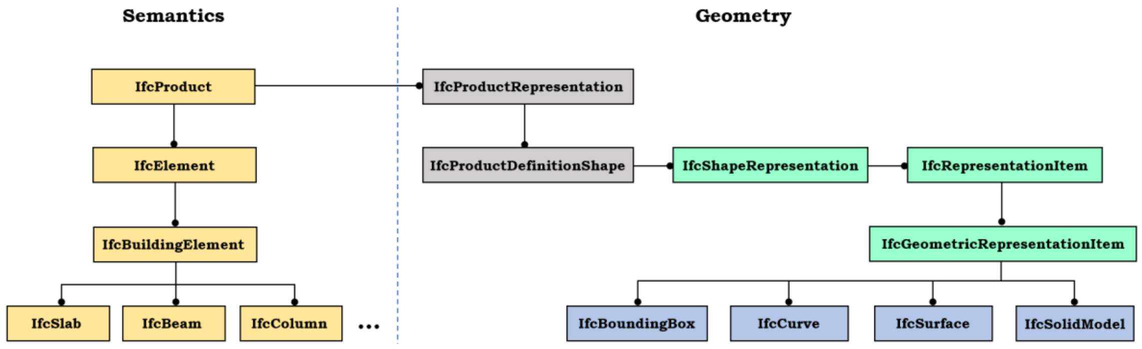


Figure 2.19 Semantic structure and geometric description in IFC data model (adapted from Borrmann et al., 2018)

As shown in Figure 2.19, all geometry representations in IFC data model can be grouped into four classes: 1) Bounding Boxes; 2) Curves; 3) Surface models; and 4) Solid models. The following texts describe each in turn.

Bounding Boxes can be represented using *IfcBoundingBox*. Bounding Boxes are highly simplified geometric representation for 3D objects that are usually used as placeholders. *IfcBoundingBox* is defined by a placement corner point and dimensions of the three sides as a cuboid.

Then, to model line objects, *IfcCurve* and its subclasses *IfcBoundedCurve*, *IfcLine*, and *IfcConic* can be used. To model sophisticated and complex geometries, freeform curved edges (i.e. splines) and curved surfaces are required. Specifically, a freeform 3D curve is mathematically described as parametric curves, meaning that the x, y, z coordinates are functions tracing a 3D curve at common parameters.

Next, surface models are used to represent composite surfaces comprised of sub-surfaces. They can be curved surfaces (e.g. NURBS) or flat surfaces (e.g. mesh). The author mentioned earlier in Section 2.2.1.2, both TSR (tessellation) and PR/MR (polygon or mesh) can be regarded as B-Rep types. TSR is a very simple geometric representation that can be interpreted by almost all visualization software applications. *IfcTriangulatedFaceSet* can be used to represent the tessellated surfaces, i.e. polygons with an arbitrary number of edges, or triangular mesh. TSR cannot represent curved surfaces ideally but approximate them into triangular facets. In this case, the curved surface can be described using a finer mesh size, if accuracy is a concern. Specifically, *IfcBSplineSurface* can be used for representing curved surfaces, such as NURBS surfaces (buildingSMART, 2016a).

One classic way to generate 3D objects as solid models is through CSG approach. *IfcCsgPrimitive3D* and its subclasses such as *IfcBlock*, *IfcRightCircularCylinder*, *IfcSphere*, and so on can be used. The combination operations can be performed using *IfcBooleanResult*. However, as earlier mentioned, the use of CSG is very limited due to the fact that the use of primitives is very restrictive. By contrast, SSR (Swept Solid Representation or Extrusion) is widely used for creating 3D objects in IFC. Possible representations include but not limited to the classes summarized in the following. Detailed descriptions can be found in (Borrmann et al., 2018). In general, *IfcSweptAreaSolid* and its subclasses *IfcExtrudedAreaSolid*, *IfcRevolvedAreaSolid*, *IfcFixedReferenceSweptAreaSolid*, and *IfcSurfaceCurveSwptAreaSolid* can be used to present extruded solids. A closed profile (i.e. cross-section) *IfcArbitraryClosedProfileDef*, which is the most common subclass of *IfcProfileDef*, is necessary for this representation. For example, when using *IfcExtrudedAreaSolid*, the ExtrudedDirection is defined so that *IfcArbitraryClosedProfileDef* can be extruded along the direction. When using *IfcRevolvedAreaSolid*, both ExtrudedDirection and axis are defined so that *IfcArbitraryClosedProfileDef* can rotate around the axis up to a given angle. Then, *IfcFixedReferenceSweptAreaSolid* allows the extrusion to be done along any curve in space through the attribute Directrix. That is to say, the profile is extruded along a specific axis defined by the attribute FixedReference. By contrast, *IfcSectionedSpine* and *IfcSweptDiskSolid* are two representations working in a different but similar way.

Section 2.1 reviews techniques related to objection detection in point clouds, Section 2.2 reviews modelling techniques. Specifically, Section 2.2.2 and Section 2.2.3 provide introductions of IFC, MVD, and geometric representation in IFC standards. This thesis does not focus on the development of any IFC-Bridge extension, but rather on geometric modelling from point clouds using IFC. To this end, one research question needs to be answered is how to geometrically describe bridge objects using existing IFC standards.

The author summarizes the gaps in knowledge, the objectives, and the research questions of this PhD thesis in the following text after reviewing current techniques and standardization for generating DTs.

Area	***Surfaces***	***Labelled Surfaces***	***Shapes Without Labels***	***Labelled (Volumetric) Clusters***	***Labelled 3D Model***	***Enriched IFC Model***
Bridge		(Riveiro et al., 2016) (Zhang et al., 2015)	(Walsh et al., 2013)	🟦	🟦 (Zhang et al., 2014)	*(Hüthwohl et al., 2018) *(Ma et al., 2017)
Building	(Dimitrov et al., 2016) (Vo et al., 2015) (Jung et al., 2014) (Klasing et al., 2009) (Marton et al., 2009)	(Xu et al., 2018) (Wang et al., 2015) (Díaz-Vilariño et al., 2015) (Xiong et al., 2013) (Adan & Huber, 2011) (Koppula et al., 2011) (Okorn et al., 2010) (Budroni & Boehm, 2009) (Rusu et al., 2008)	(Dimitrov & Golparvar-Fard, 2015) (Xiao & Furukawa, 2014) (Rusu et al., 2009) (Schnabel et al., 2007)	(Qi et al., 2017) (Armeni et al., 2016) (Su et al., 2016)	(Macher et al., 2017) (Ochmann et al., 2016) (Valero et al., 2016) (Valero et al., 2012)	*(Belsky et al., 2014)
Industry				(Patil et al., 2017) (Su et al., 2016) (Son et al., 2015) (Ahmed et al., 2014) (Son et al., 2013) (Rabbani, 2006)	(Laefer & Truong-Hong, 2017)	

IFC Maturity

* Input: gDT

🟦 Objectives of this thesis

Table 2.2 State of research on DT generation from PCD in terms of IFC Maturity

2.3. Gaps in Knowledge

The author summarizes in Table 2.2 the state of research on DT generation with regard to the application domain and the maturity of the resulting models. Most current studies focus on generating gDTs from PCD for buildings and industrial facilities. Fewer studies focus on the bridge gDT generation. Among existing object detection methods in DT-related work, a large body of methods concentrates on low-level primitive generation (Dimitrov et al., 2016), i.e. surface clustering from points. Far fewer methods can directly generate labelled point clusters. Existing methods either focus on generating building or industry elements, which basically take generic shapes, such as cuboids (Macher et al., 2017), cylinders (Patil et al., 2017) or standardized steel beams (Laefer & Truong-Hong, 2017). These methods cannot be directly applied to detecting bridge components in PCD because real bridge elements are hardly in those idealized shapes. Every real bridge component contains skews and irregularities. In addition, real PCD are noisy and imperfect, suffering from occlusions and sparseness. These factors render methods designed for synthetic data or simplified scenarios ineffective. The few exiting bridge-DT-related studies that work well have restrictive constraints – they either take an idealized bridge solid model as input to infer the semantic meaning of components (Ma et al., 2017) or they train a classifier with generic shapes or test against a synthetic bridge point cloud (Zhang et al., 2014).

Research is still in the early stage of gDT creation for existing infrastructure. Among existing object fitting work, almost all methods are used for generating building or industrial elements, such as walls, ceilings, floors, and standardized industrial elements. These objects are simply represented as extruded planes elements, such as cuboids, or extruded steel beams. The gDT generation for existing RC bridges is almost missing in the literature.

In short, the problem of detecting bridge objects in the form of labelled point clusters from real bridge PCD has yet to be solved. **Gap 1**: No method can directly produce labelled bridge point clusters making up a bridge point cloud.

Likewise, the problem of fitting 3D solid models in IFC format to real bridge point clusters has yet to be addressed. **Gap 2**: No method can reconstruct labelled real bridge point clusters into 3D IFC objects.

A third problem yet to be solved effectively is that of semantic enrichment (i.e. life-cycle information enrichment) for bridge gDTs (Hüthwohl et al., 2018). **Gap 3**: No method can enrich a bridge gDT with all the bridge's life-cycle information.

In addition, the problem of evaluating the quality and degree of automation of a generated gDT compared to its PCD has yet to be studied in depth. **Gap 4:** No standardized metric is available for the quantitative evaluation of a gDT.

Last, but not least, the problem of standardizing the definition of the "level of detail" (LOD) for a gDT has yet to be solved. The use of LOD in current studies on DT generation is quite vague and perfunctory. High-level industry standard specifications are urgently needed to enable quality assessment of generated objects in the gDTs according to their corresponding LOD. **Gap 5:** No standardized specifications are available to clearly define the LOD of (as-is) gDT generation and to gauge the quality of the resulting gDTs.

2.4. Objectives, Research Questions & Hypothesis

This thesis aims to fill the above-mentioned first two gaps (Gaps 1 and 2). It also aims to provide a quantitative measurement for assessing the quality of the resulting bridge gDT (Gap 4). The main objectives of this research are

- **Objective 1:** Automatically detect four types of bridge structural component in PCD (Chapter 4), and
- **Objective 2:** Automatically fit 3D solid models in IFC format to the four types of bridge point cluster (Chapter 5).

The minor objective of this research is:

- **Objective 3**: Leverage reasonable 3D model assessment metrics to assess the generated bridge gDTs.

These objectives are realized respectively by answering the following research questions:

- **Research question 1:** How to detect objects in imperfect RC bridge PCD where occlusions and non-uniformly distributed points exist, in the form of labelled point clusters making up a bridge?
- **Research question 2:** How to extract and use the geometric features to reconstruct the labelled point clusters in arbitrary shapes of real bridge PCD, into 3D solid models in IFC format? And
- **Research question 3**: How to evaluate the spatial modelling accuracy of a bridge gDT reconstructed from a point cloud?

The **hypothesis** of this PhD thesis is that an automated top-down framework can detect structural component types in RC bridge PCD with high detection rate and generate their corresponding geometric 3D solid models using current IFC standards with high modelling accuracy as well as with much less time and effort than the current practice. This hypothesis will be tested with a PCD collection of ten highway RC bridges in the UK.

3. Proposed Framework

This chapter presents the hypothesized framework which was sponsored by the EU Infravation SeeBridge project under Grant No. 31109806.0007 (SeeBridge, 2014). SeeBridge aims to provide a step change in the way asset owners inspect existing bridges in virtual 3D space by using PCD and high-resolution images to automatically generate semantically enriched bridge models, namely meaningful DTs of real bridges. To this end, SeeBridge creates an automated system through various research activities, i.e. research work packages WPs. Details of the main WPs can be found in Appendix B. This thesis presents work conducted under WPs 2, 3, and 4. All work included in this thesis is done by the author independently without any collaborative content.

3.1. Introduction

This PhD thesis intends to deliver and validate an automated framework to generate gDTs of existing RC bridges from PCD. This automated framework mimics the human modelling workflow and the intelligence of modellers' in detecting and modelling bridge components in PCD. To this end, the proposed framework consists of two major parts, corresponding to four EURs (Chapter 1, Section 1.3.2). Specifically, **(1) Object Detection** (Chapter 4) aims to meet EUR 1, (partially) EUR 2, and EUR 3, and **(2) IFC object fitting** (Chapter 5) aims to meet EUR 2 and EUR 6.

The author proposes a novel top-down framework which exploits bridge topology knowledge as guidance to directly extract labelled point clusters corresponding to bridge components without generating surface primitives, and then to efficiently reconstruct these labelled point clusters into 3D IFC components.

Real-world bridges are neither perfectly straight nor flat. Bridge geometries are defined by horizontal straight or curved alignments, vertical elevations, and varying cross-sections. The author proposes a slicing-based algorithm to tackle these difficulties. This algorithm is repeatedly used throughout the whole solution until all the components are detected and modelled. The proposed slicing algorithm can deal with the skew complexity of real bridge geometries and can quickly select a set of candidate locations for target objects. The global topology of a bridge can also be well approximated using multiple slices.

3.2. Framework Scope

The scope of the thesis is focused on the compilation of the bridge gDTs and on reducing the cost, duration and the effort required to acquire them. This scope includes the following several aspects.

Data source. This thesis assumes that neither the "as-designed" or the "as-built" bridge gDT are available. Given the current state of affairs, in which thousands of bridges are in service that do not have DTs but must be inspected nevertheless, this thesis scope considers the need to compile the gDT of an existing bridge based on the TLS information alone.

Bridge type. Figure 3.1 illustrates both **applicable and non-applicable bridge topology types** of the proposed automated framework. This thesis focuses on typical RC slab and beam-slab bridges. This is because in the UK, on motorways and major A-roads, 73% of existing highway bridges and 86% of planned future bridges are RC slab and beam-slab bridges (Kim et al., 2016). These two types of bridges reflect the fundamental bridge-design rules that allow for the natural introduction of geometric constraints, describing the relationships that should hold between various components of a bridge. The framework is also applicable for tied arch bridges if their above-deck parts (tied arch) are removed in advance. Similarly, the framework is partially applicable for suspension bridges and cable-stayed bridges if the ropes and cables are removed in advance. However, the deck of these types of bridges is usually composed of steel elements. This requires different methodology to tackle with. In addition, the proposed framework is not applicable for arch bridges, cantilever bridges, and truss bridges.

Key components. The key components of RC slab bridges include slabs, piers, and pier caps, and for RC beam-slab bridges there are slabs, girders, piers, and pier caps (Table 3.1). Thus, the framework deals with four very important and highly detectable structural components of slab and beam-slab bridges: slabs, piers, pier caps, and girders, in line with Kedar's (2016) first element identification evaluation category, i.e. Category 1, which takes into account the importance and the detectability of each element (Appendix C). For each bridge type, the elements are divided into four importance classes and three detectability classes (Table 3.2).

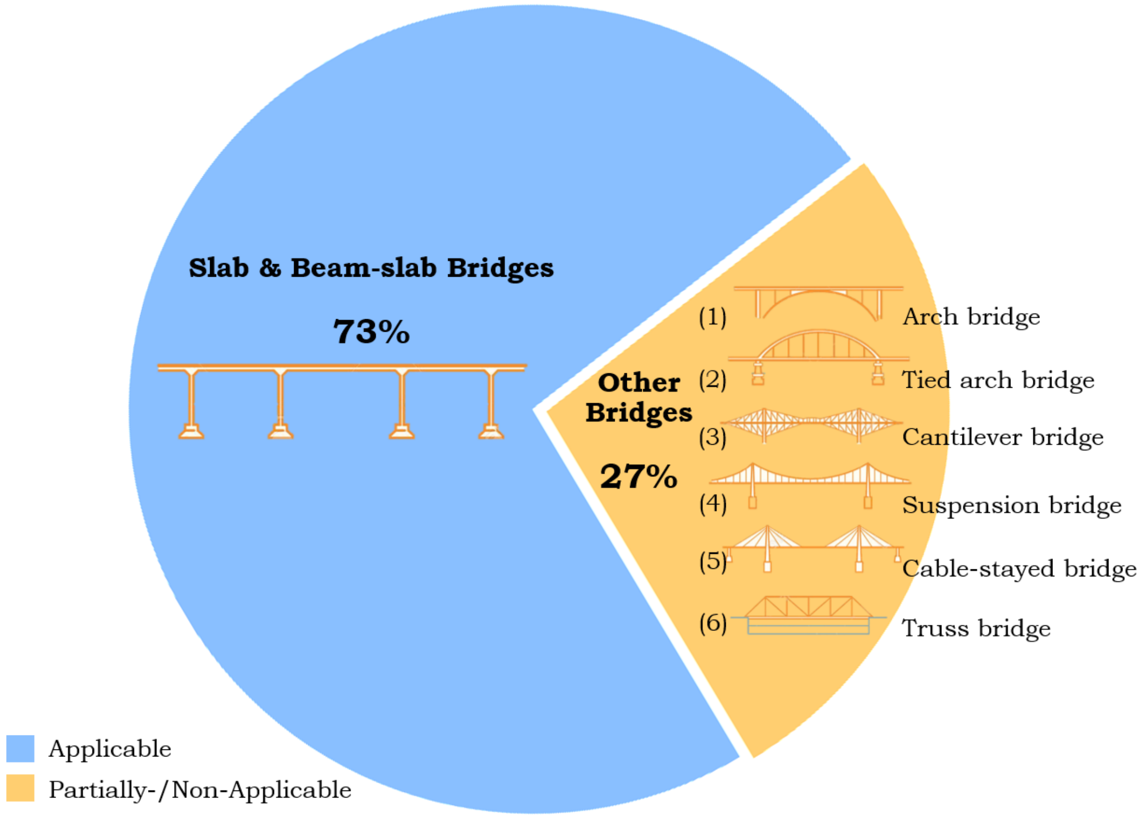


Figure 3.1 Applicable and Partially-/Non-Applicable bridge topology types of the proposed framework

Bridge Type	Key Components	
	Superstructure	Substructure
Slab	deck	pier, pier cap
Beam-Slab	deck, beam/girder	pier, pier cap

Table 3.1 Key components for bridge type (Kim et al., 2016)

		Element importance				
		Very High	High	Medium	Low	-
Element Detectability	Detectable	Category 1	Category 2	Category 3	Category 3	Category 4
	Partially detectable	Category 2	Category 3	Category 3	Category 4	Category 4
	Non-detectable	Category 5	Category 5	Category 5	Category 5	Category 5

Table 3.2 Element identification evaluation categories (Kedar, 2016)

Semantics. This thesis only focuses on identifying and enriching the component-level semantics of four types of component in the slab and beam-slab bridges and digitizing their geometric shapes and attributes in IFC format (EURs 1, 2, 3, and 6). The enrichment of other semantic information (EURs 4 and 5) such as material, defects, additions of relationships (aggregations and connections), and maintenance history and so on are beyond the scope of this research.

3.3. Framework outline

The author illustrates the developed top-down framework of this thesis in Figure 3.2. The whole framework consists of two major processes: **Process 1**, detection of four bridge component types in real PCD, and **Process 2**, run-time model fitting to the component point clusters according to the current IFC standards. Chapter 4 and 5 describe these two processes in detail, respectively.

The whole framework starts with a raw registered PCD of an existing RC bridge (data format: points in .pcd or .txt). Irrelevant points such as vegetation, trees, on-site traffic, etc. are then manually removed. The cropped bridge PCD contains the four types of structural component to be modelled (data format: points in .pcd or .txt). The author then automatically aligns the cropped bridge PCD such that its centre axis, i.e. horizontal alignment, is roughly parallel to the X-axis of the global coordinate system. Thus, the output of this step is a roughly aligned bridge PCD (data format: points in .pcd or .txt).

Next, the author proposes a four-step object detection method (Process 1) through which the structural components underlying in an RC bridge PCD are detected. The final outputs of this process are four structural bridge components, namely slab, pier caps, piers, and girders, in the form of labelled point clusters (data format: points in .pcd or .txt). Chapter 4 elaborates this process, where the author answers **research question 1**, i.e., how to detect RC bridge components in imperfect PCD where occlusions and unevenly distributed points exist, in the form of labelled point clusters?

Then, the author suggests manual refinement to remove the noise maintained from Process 1 followed by proposing an efficient IFC object fitting method (Process 2) through which the four types of point clusters can be directly modelled into 3D IFC objects in different resolution. The final outputs of this process are two IFC files of a bridge, corresponding to two different levels of detail: LOD 200 and LOD 250—300, data format: 3D objects in .ifc. Chapter 5 elaborates this process, where the author answers **research questions 2 and 3**, i.e., how to extract and use the geometric features to describe the labelled point clusters in arbitrary shapes of real bridge PCD, into 3D solid models in IFC format? And how to evaluate the accuracy of the resulting bridge model (gDT)?

So far, there is no method that can reliably detect RC bridge structural objects embedded in real PCD featuring defects. The expected **contribution of Process 1** is that it is the first method of its kind to achieve robust and reliable performance for detecting four types of RC bridge component in the form of labelled point clusters in real-world PCD. In addition, there is no method that can fit real bridge point clusters with 3D solid models in IFC format. The **contribution of Process 2** is that it is the first method of its kind to efficiently generate a highly accurate gDT in IFC format using labelled point clusters making up an existing RC bridge.

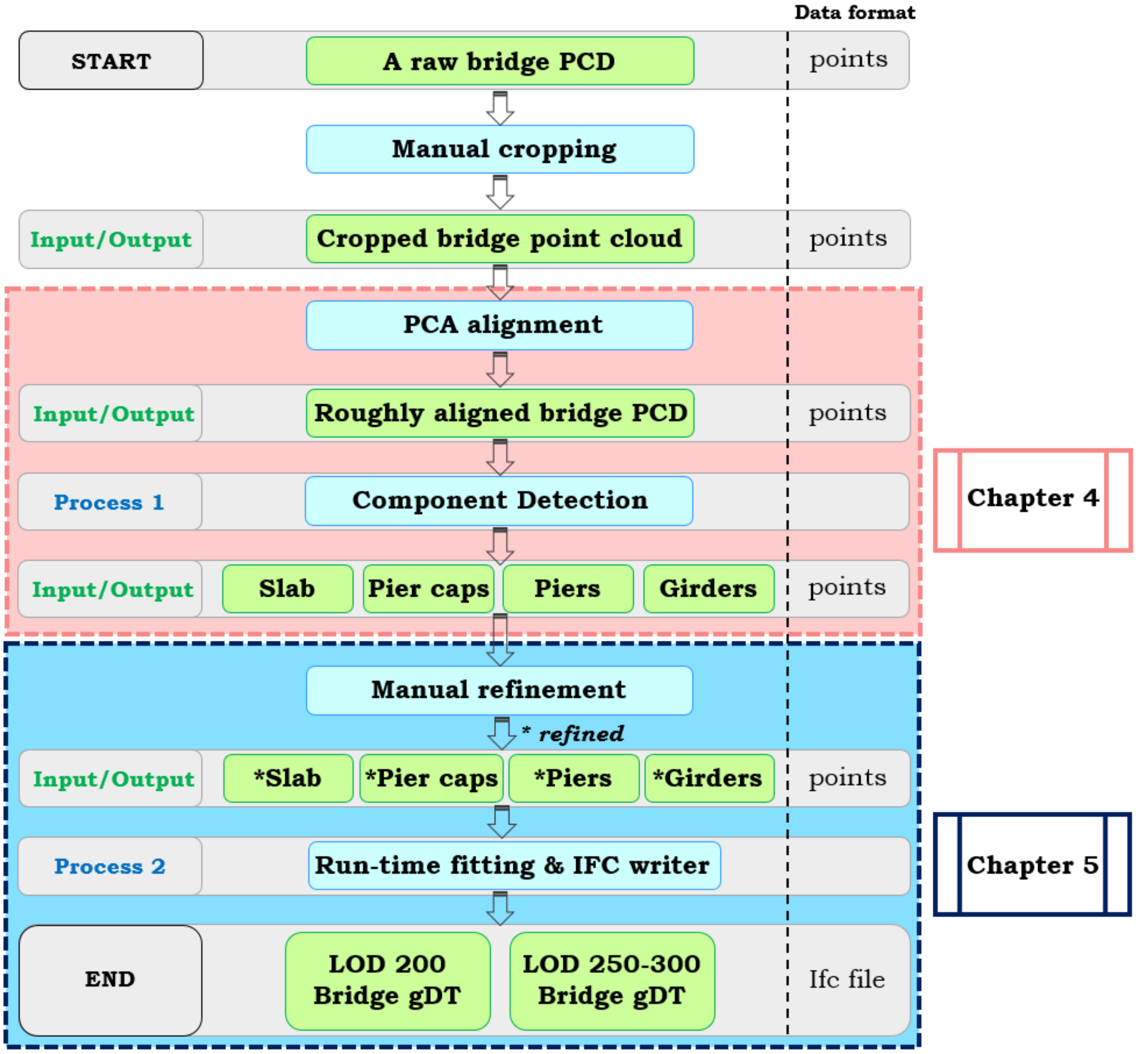


Figure 3.2 The input/output workflow of the proposed framework in this thesis

3.4. Assumptions

A list of assumptions is made so that the goal of this PhD thesis does not become impossibly large to complete. As claimed earlier in the framework scope, this thesis takes the TLS data alone as data source. Thus, one big assumption is that

A0. The registration quality of the PCD is high enough to conduct any post-processing.

This means the author does not take the registration error into account in this thesis. This assumption is confirmed in the experiments. In the following texts, the author lists other specific assumptions for methods presented in Chapter 4 and 5, respectively.

- **Assumptions of Process 1**

According to national standards (Highways England, 2018c), the author assumes the proposed object detection method is feasible in the context of RC bridge gDT generation under the following conditions, which are also confirmed in the experiments.

A1. Pier assembly and deck assembly can be separated using the ratio ρ_1.

A2. Pier area and deck assembly can be separated using the ratio ρ_2.

A3a. A pier assembly does not contain a pier cap if a single pier area is detected in the pier assembly and the width of the pier is almost the same as that of the pier assembly.

A3b. Surface normals are distinct features that can be used to distinguish a pier cap from a pier.

A3c. Pier cap parts can be extracted from the deck assembly using the ratio $\rho_{3b} = \frac{\rho_1}{\rho_2}$.

A4. The density histograms along the best view can be used to segment the girders in the deck assembly segment.

A1 & A2 are validated experimentally in Section 6.2.4.1 while A3a, A3b, A3c, and A4 are validated in Section 6.2.4.2. This thesis also assumes that an RC bridge satisfies the following conditions:

A5. The piers are vertical or quasi-vertical.

A6. On-site clutter and irrelevant points are properly removed manually.

- **Assumptions of Process 2**

The author assumes the proposed IFC object fitting method is feasible in the context of RC bridge gDT generation under the following conditions:

A7. The input used in the proposed method presented in this chapter are ideal results produced from the previous step (object detection), so that abnormal noisy points and outliers have been manually removed.

A8. The author assumes the bridges investigated in this research do not have major deformations or collapse. Moderate deformations are allowed. This is to say, major departures of components from their as-built shapes do not exist. This assumption is used to fit a curve to the bridge deck slab (see A9).

A9. The horizontal alignment of the bridge deck slab is assumed to follow a parabola curve.

A10. The slab segments are assumed to be straight along the tangent direction of the horizontal alignment.

A11. The cross-section of components to be extruded for gDT generation is considered to be constant along its extrusion direction.

A12. The pier cap height is assumed not significant so that the extruded direction of a pier cap is considered to be vertical and the inclination of a pier cap along the Z-axis is not taken into account.

A13. Girders are placed along the alignment and elevation of each span.

A14. Girders are straight in each span so that they can be represented through extrusion. Revolved girders are not studied in this research.

A15. Girders are standard precast concrete bridge beams so that the template matching method is feasible.

4. Bridge Component Detection in PCD

The **objective** of this chapter is to provide an automated method to detect four types of bridge structural component in PCD. This objective is achieved by answering the **research question**: How to effectively cluster and classify a real-world RC bridge PCD featuring defects into labelled point-clusters corresponding to the four types of structural components constituting a bridge?

This chapter was published in the Journal of Computer-Aided Civil and Infrastructure Engineering under the title “Detection of Structural Components in Point Clouds of Existing RC Bridges” (Impact factor: 5.475). This paper was co-authored with Dr Ioannis Brilakis and Prof Campbell R. Middleton. The author’s contributions include: conceptualization, data collection, methodology, software, validation, writing the manuscript, and editing. Dr Ioannis Brilakis’s contributions include: funding acquision, supervision, discussion, and reviews. Prof Campbell R. Middleton’s contributions include: discussion and reviews.

4.1. Introduction

In this chapter, the author aims to (1) segment an RC bridge PCD into mutually disjointed point clusters corresponding to the four types of component making up the bridge, and (2) assign correct semantic labels to these point clusters. To this end, the author proposes a novel top-down method. The novelty of this method lies in the fact that it directly extracts the key components of RC bridges in the form of labelled point clusters without generating low-level shape primitives.

The work presented in this chapter is deeply rooted in the area of artificial intelligence, especially computer vision and computer graphics. Most existing studies follow a bottom-up clustering approach. Using correlations, these methods can piece together low-level 3D points, generate new hypotheses, and then link them together to form higher level elements.

However, the literature review (Chapter 2, Section 2.1) reveals that a simple bottom-up clustering method that generates low-level primitives has not, to date, managed to solve the above-mentioned research question regarding detecting objects in real bridge PCD. Challenges remain unsolved, for example existing methods cannot robustly handle data defects such as occlusions and the varying sampling point density existing in real PCD. In addition, none of the existing methods can efficiently detect bridge components in PCD with irregular underlying geometries. Low data quality and high data uncertainty impede the combination of basic elements from being converted to more complex models.

Note that object detection not only needs to cluster elements, it also needs to label them, and thus assign elements with real meaning. This task is defined in this thesis as “classification”. As discussed earlier in Chapter 2, object classification through low-level primitives is hard to achieve. The intervention of high-level class-specific information is necessary to classify objects in a meaningful manner.

By contrast, the top-down approach is faster than its bottom-up counterpart. It can process a large amount of data, subdividing a given problem into a series of sub-problems that are supposedly easier to solve. The top-down approach is also based on domain-knowledge and relies on a symbolic description of the simplified world. Note that the top-down approach, whose goal is to use a set of pre-defined recognition criteria or rules to mimic human intelligence, suffers from some inherent limitations. Inevitably, a combinatorial explosion of the number of rules occurs due to the complexity of the environment and it is impossible to predict all situations (especially unknown ones) that will be encountered by an autonomous entity. Yet, the limitations of the top-down approach are irrelevant in the context of slab and beam-slab RC bridges. This is because the level of variance of these types of bridges is relatively low compared to other real-world objects. Bridge structural components are distinct 3D solid objects, and the taxonomy of bridge components for any given context is finite and well defined. Hence, this chapter intends to provide a novel top-down object detection method that can detect four types of bridge component embedded in real PCD.

4.2. Proposed Method

4.2.1. Overview

Human expertise can facilitate the search for a specific object in a scene because our guesses for the embedded objects are best when we know what to expect in a point cloud. Hence, the chapter investigates a brand-new top-down object detection method which directly extracts key components of RC bridges underlying in the PCD without generating low-level shape primitives. The general thrust behind the proposed top-down detection approach is to use the fundamental bridge design rules such as bridge topological constraints, given the low level of variance of the topological layout of RC slab and beam-slab bridges.

The **hypothesis** of this chapter is that the top-down bridge-component detection method has a high detection rate and there is no significant difference in detection performance for different RC bridges. In addition, the detection time is less compared to that of the current practice. This hypothesis will be tested with a PCD collection of ten highway RC bridges in the UK (Chapter 6).

In this chapter, the author uses the point cloud of the Nine Wells bridge to demonstrate the development of the suggested method. The bridge was constructed to the south west of Addenbrooke's Hospital in Cambridge in 2009 to carry a new road over a railway line. It comprises three spans of approximately 30 m in length, each constructed from 12 precast concrete "SY beams" with an in-situ concrete deck slab (Graham, 2014). The bridge also contains a group of 4 cylindrical piers with a pier cap on top, a wall-like pier, and two wall-like vertical abutments as supports (Figure 4.1).

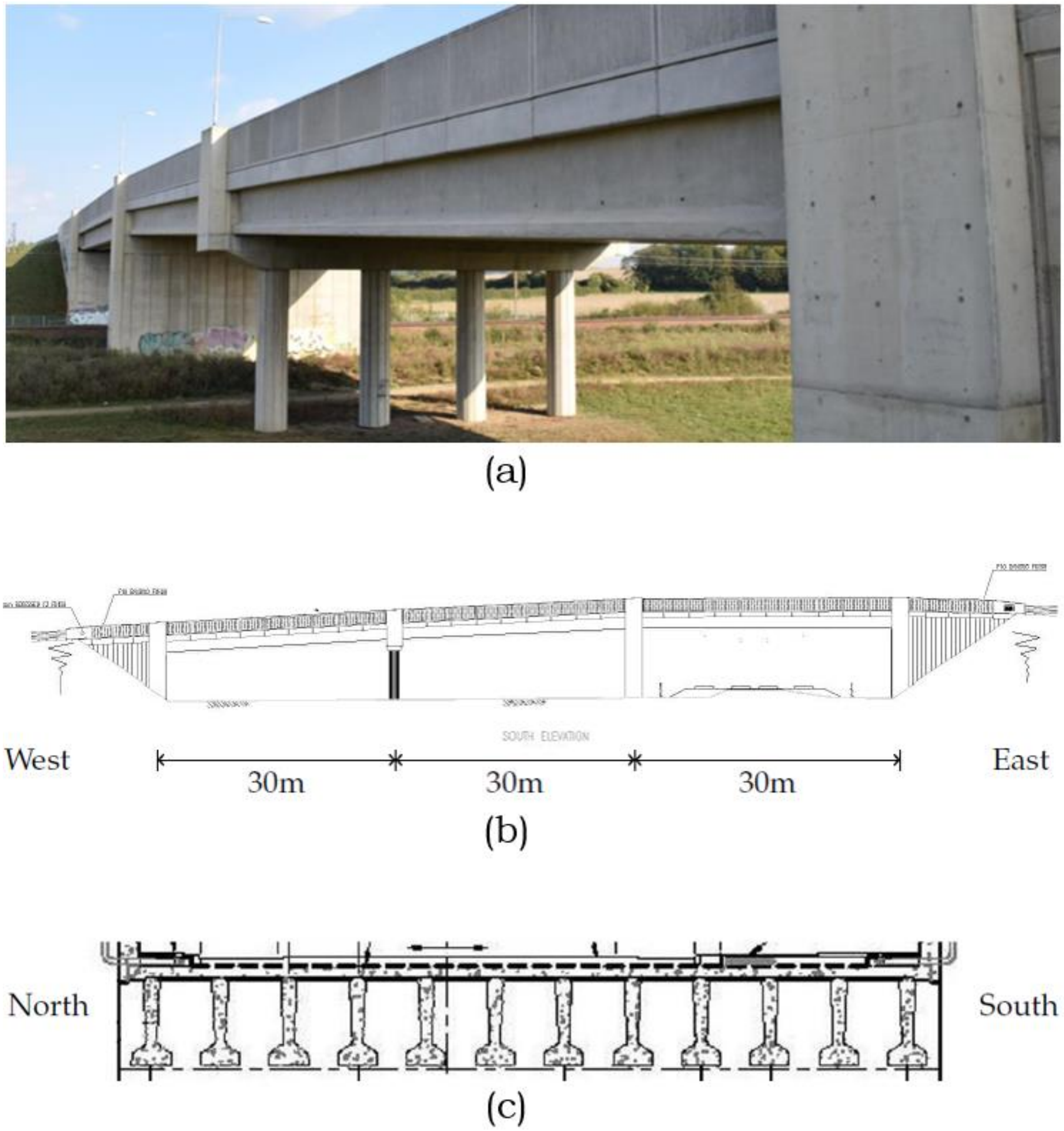


Figure 4.1 (a) Northwest view; (b) Side view; and (c) SY beams of the Nine Wells Bridge (Graham, 2014)

This method bypasses the stage of surface generation altogether and directly obtains segmented and labelled point clusters. It breaks down a large bridge point cloud into sub-datasets through a recursive slicing algorithm. That is, the method chops the point cloud by means of a 'virtual parallel scalpel' with a specified equal thickness. This algorithm is repeatedly used with sub-datasets until target objects are found and all small detection problems are solved. The key insight behind this method is to formulate the geometric feature search to explore shortcuts so that the target objects can be quickly located in the point cloud.

The workflow of the proposed method in this chapter is illustrated in Figure 4.2. Dashed frames refer to ambiguous components that may or may not exist in a bridge point cloud. Acronyms are used to present the inputs, intermediate outputs and final outputs of each step. For example, $\{B|A\}$ represents that B is a subset of A. A is derived from a previous step and is the superset of B. $\{B|A\}$ may also represent that B is a property set of A (e.g., $\{x_i|A\}$ means the x values of A). More precisely, 'deck assembly' refers to the areas which contain slab and girders (if they exist), 'pier assembly' refers to the areas which contain a transverse strip of slab, pier caps (if they exist) and piers, and 'pier areas' refers to the subsets of the pier assembly which contain a small part of the slab strips, part of the pier cap, and an individual pier.

The first two steps in the proposed detection method are recursive. The first step segments a whole aligned bridge point cloud (D_N) into the pier assembly (denoted α_M) and deck assembly (denoted ${\alpha_M}^C$). The second and third steps detect pier areas (denoted β_{mp}) and pier caps (denoted P_C) in the pier assembly and deck assembly. The last step detects girders (denoted $girders$) and slab (denoted $slab$) in a merged deck assembly. Note that pier caps and girders may not exist in some bridge point clouds.

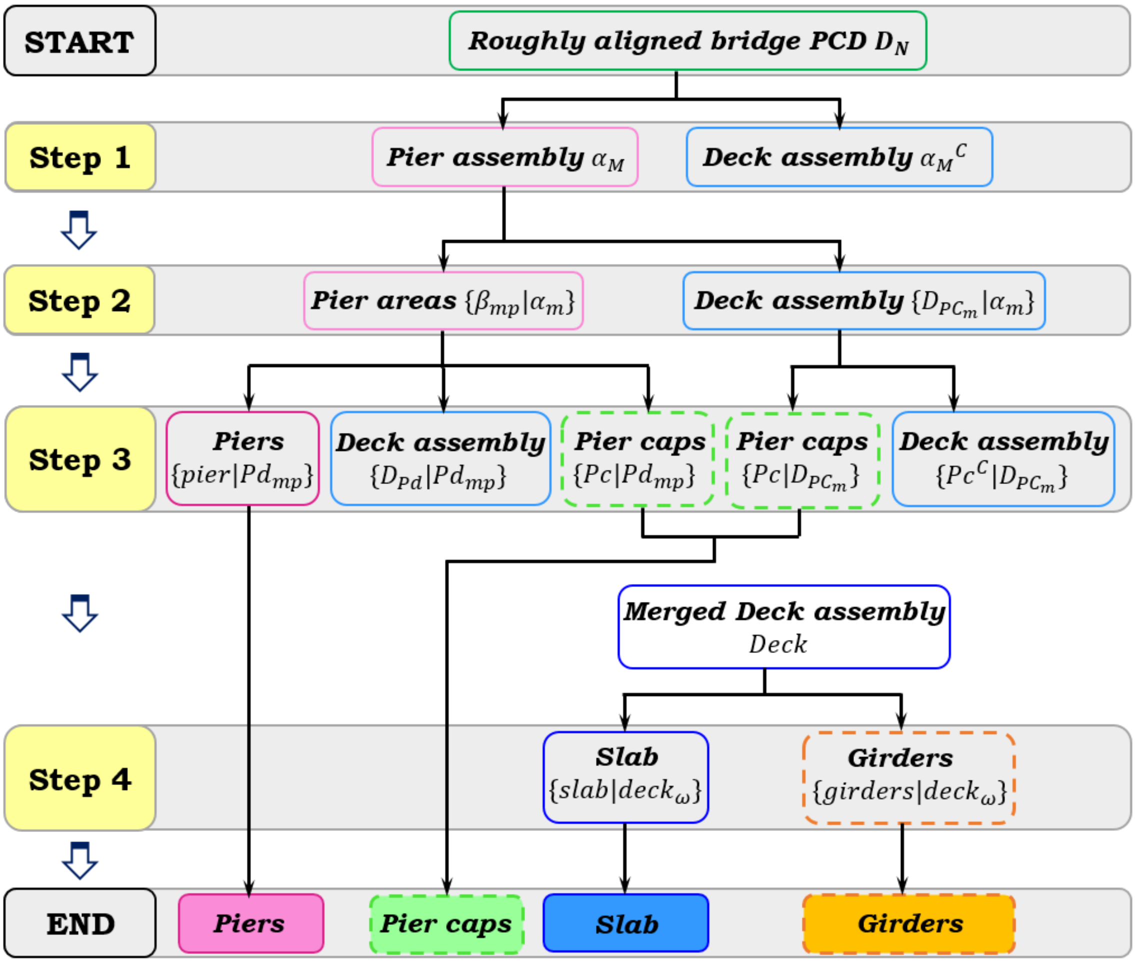

START
Roughly aligned bridge PCD D_N
Step 1
Pier assembly α_M
Deck assembly $\alpha_M{}^C$
Step 2
Pier areas $\{\beta_{mp}|\alpha_m\}$
Deck assembly $\{D_{PC_m}|\alpha_m\}$
Step 3
Piers $\{pier|Pd_{mp}\}$
Deck assembly $\{D_{Pd}|Pd_{mp}\}$
Pier caps $\{Pc|Pd_{mp}\}$
Pier caps $\{Pc|D_{PC_m}\}$
Deck assembly $\{Pc^C|D_{PC_m}\}$
Merged Deck assembly
Deck
Step 4
Slab $\{slab|deck_\omega\}$
Girders $\{girders|deck_\omega\}$
END
Piers
Pier caps
Slab
Girders

Figure 4.2 Workflow of the proposed object detection method

4.2.2. Step 1 – Pier assembly and deck assembly detection

A bridge point cloud is given at an arbitrary position and orientation. The pose of a bridge should be normalized in advance as all features to be extracted in further steps lie in a canonical coordinate frame. Principal Component Analysis (PCA) is leveraged to align a bridge such that the horizontal alignment of the bridge is positioned roughly parallel to the global X-axis (Figure 4.3). Note that the alignment is not perfect because PCA provides a rough estimate (Liu & Ramani, 2009) and also because the bridge itself contains a certain degree of skewness. Specifically, the direction of the principal axis is defined as three eigenvectors of the covariance matrix of a bridge point set. Given a bridge point cloud of N points $q_i = (x_i, y_i, z_i)^T$, $i = 1,2, \dots, N$, the covariance matrix C is defined as:

$$C = \frac{1}{N}\sum_{i=1}^{N}\{(q_i - q_{centre})(q_i - q_{centre})^T\}, \qquad \text{(Eq. 4.1)}$$

where the centre of points $q_{centre} = (\mu_x,\ \mu_y,\ \mu_z)^T = \frac{1}{N}\sum_{i=1}^{N} q_i$. Let u_1, u_2, u_3 and v_1, v_2, v_3 be eigenvectors and eigenvalues of the covariance matrix C, respectively and $v_1 \geq v_2 \geq v_3$. C can be factorized as:

$$C = U\Sigma V^T, \qquad \text{(Eq. 4.2)}$$

where $U = [u_1, u_2, u_3]$ and $\Sigma = \text{diag}(v_1, v_2, v_3)$. The rotation matrix R can be estimated from:

$$R = U_m {U_d}^{-1}, \qquad \text{(Eq. 4.3)}$$

where U_m and U_d are eigenvectors of the model data and the original data, respectively.

Approximate alignment at an early stage makes it possible to inject strong 3D priors in canonical normalized space and to reformulate features employed for recursive segmentation in the following steps. Therefore, the input of Step 1 is a roughly aligned bridge point cloud $D_N = \{p_i: i = 1,2,\dots,N\}$, where each point is defined as $p_i = (x_i, y_i, z_i)^T$.

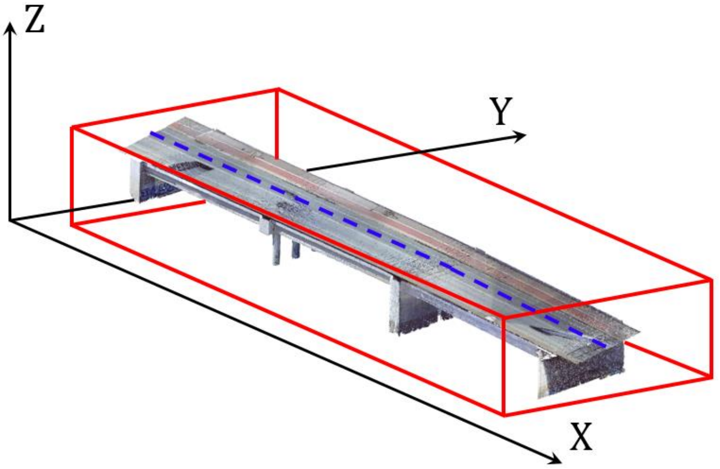


Figure 4.3 Roughly aligned bridge point cloud using PCA

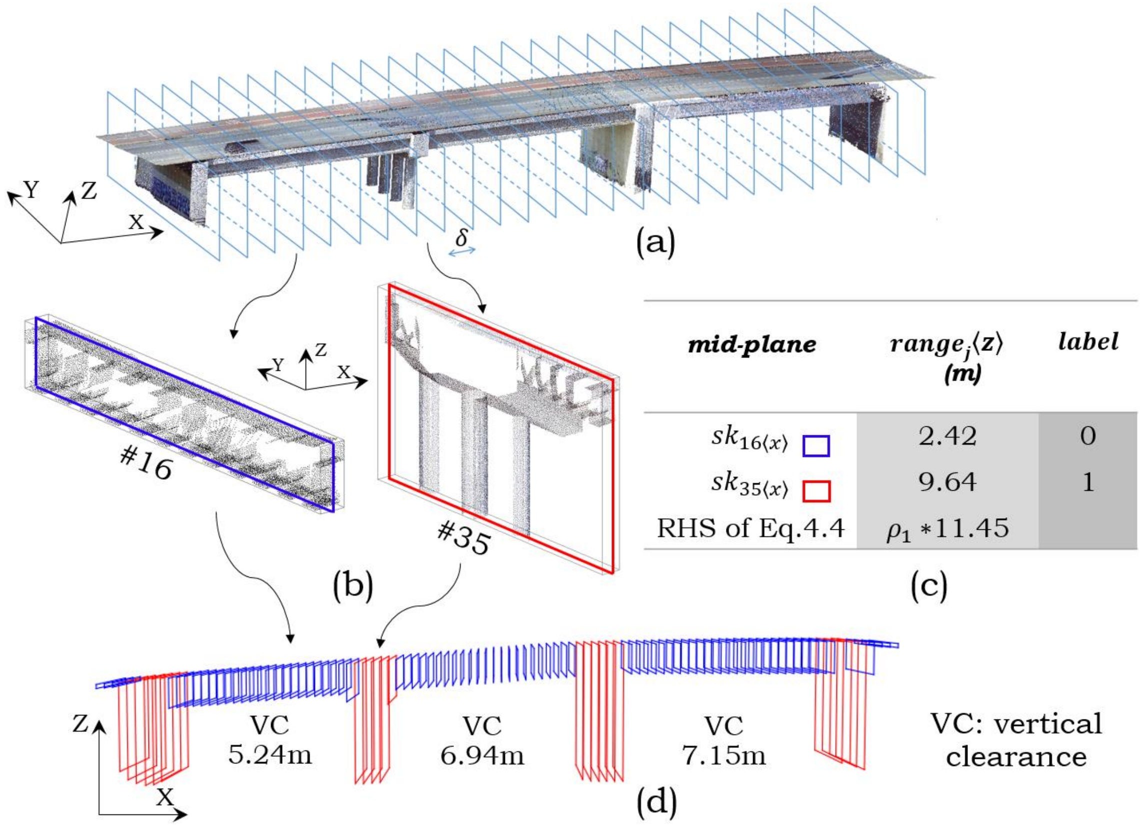
Step 1 aims to classify all the bridge points, D_N, into two groups: pier assembly group $\alpha_M = \{\alpha_1, \alpha_2 \dots, \alpha_m\}$, where m is the number of the pier assembly and deck assembly group ${\alpha_M}^C$. To this end, the method chops D_N into multiple slices along the X-axis (Figure 4.4 (a)). Let J be the number of slices, then the method obtains slices $S_X = \{S_{j\langle x\rangle}: j = 1,2,\dots,J\}$, where $\langle x\rangle$ refers to the axis of slicing. Define $D_j = \{p_{ji}\} = \{p_i | S_{j\langle x\rangle}\}$ to be the point set in slice $S_{j\langle x\rangle}$ so that $\sum_{j=1}^{J} \sum_{i=1}^{D_j} p_{ji} = D_N$, where p_{ji} is the ith point in the jth slice. Note that slicing might lead to an empty slice where local points are missing or a slice with just one single point. In this case, the geometric features cannot be computed. This slicing method can prevent such situations from happening. By assuming the slice thicknesses δ to be constant, the initialized number of slices J is proportional to the length of the bridge ($J \propto |\max\{x_i|D_N\} - \min\{x_i|D_N\}|$). Here, δ is set to be 0.5 m. When the “virtual scalpel” encounters an empty or a single-point slice, the method will infer the geometric feature from the nearest “sound” slice: if $|D_j| = \emptyset$ or $|D_j| = 1$, then $S_{j\langle x\rangle} \cong S_{j-\varphi\langle x\rangle}$, where φ is the count of slices between $S_{j\langle x\rangle}$ and the closest non-empty and non-single-point slice. This approximation is not perfect but provides immunity to locally incomplete data.

A feature detector is then needed that can distinguish the pier assembly from the deck assembly. Each slice $S_{j\langle x\rangle}$ is bounded by a 3D axis-aligned-bounding-box and a 2D skeleton $sk_{j\langle x\rangle}$ is drawn for each slice using the mid-

plane of its bounding-box (Figure 4.4 (b)). According to bridge engineering knowledge, piers support the deck against gravity, so they should start from the ground. Therefore, the height of a pier assembly slice should be much larger than that of a deck assembly slice (Figure 4.4 (b)). Thus, in each slice $S_{j\langle x\rangle}$, the method extracts the geometric feature $range_{j\langle z\rangle}$ which is the height of $S_{j\langle x\rangle}$. It then classifies $S_{j\langle x\rangle}$ as a pier assembly slice if Eq. 1 is satisfied; otherwise, $S_{j\langle x\rangle}$ is considered to be a deck assembly slice:
if

$$range_{j\langle z\rangle} > \rho_1 |\max\{z_i | D_N\} - \min\{z_i | D_N\}|, \qquad \text{(Eq. 4.4)}$$

then,
$S_{j\langle x\rangle} \leftarrow$ pier assembly,
where ρ_1 is a discrimination parameter that refers to the thickness ratio of the deck assembly relative to the height of the bridge, which should not be affected by the varying elevation (Figure 4.4 (d)). This assumption (see A1, Chapter 3, Section 3.4) will be experimentally justified in Chapter 6, Section 6.2.4.1. Adjacent slices with the same assembly property are merged into a cluster. Finally, the pier assembly α_M and deck assembly ${\alpha_M}^C$ (Figure 4.5 (a)) are acquired.



mid-plane	***range$_j$⟨z⟩ (m)***	***label***
$sk_{16\langle x\rangle}$ □	2.42	0
$sk_{35\langle x\rangle}$ □	9.64	1
RHS of Eq.4.4	$\rho_1 * 11.45$	

Figure 4.4 (a) Slicing along X-axis; (b) Deck assembly slice (blue) and pier assembly slice (red); (c) Comparison between deck assembly slice and pier assembly slice; (d) Mid-planes $\{sk_{j\langle x\rangle}\}$

4.2.3. Step 2 – Pier area detection in pier assembly

The inputs of Step 2 are the pier assemblies α_M output from Step 1. The output(s) are deck assemblies $D_{PC_M} = \{D_{PC_m} | \alpha_m\}$ and pier area(s) $\beta_{MP} = \{\beta_{mp} | \alpha_m\}$, where α_M = $D_{PC_M} \cup \beta_{MP}$ (Figure 4.5). Each pier assembly α_m can be considered to be a smaller scale bridge point cloud so that Step 2 follows the same procedure as Step 1, except that the slicing is performed along the Y-axis of α_m to obtain slices $S_Y = \{S_{j\langle y\rangle} | \alpha_m\}$. Again, the value of $range_{j\langle z\rangle}$ for each slice $S_{j\langle y\rangle}$ is extracted. The method classifies $S_{j\langle y\rangle}$ as a pier area slice if the Eq. 2 is satisfied; otherwise, $S_{j\langle y\rangle}$ is considered to be a deck assembly slice:
if

$$range_{j\langle z\rangle} > \rho_2 |\max\{z_i | \alpha_m\} - \min\{z_i | \alpha_m\}|, \quad \text{(Eq. 4.5)}$$

then,
$S_{j\langle y\rangle} \leftarrow$ pier area,
ρ_2 is another discrimination parameter that is used to separate the pier area from the rest of α_m. For a pier assembly without pier cap, $\rho_2 = \rho_1$; otherwise, $\rho_2 > \rho_1$. This assumption (see A2, Chapter 3, Section 3.4) will be experimentally justified in Chapter 6, Section 6.2.4.1.

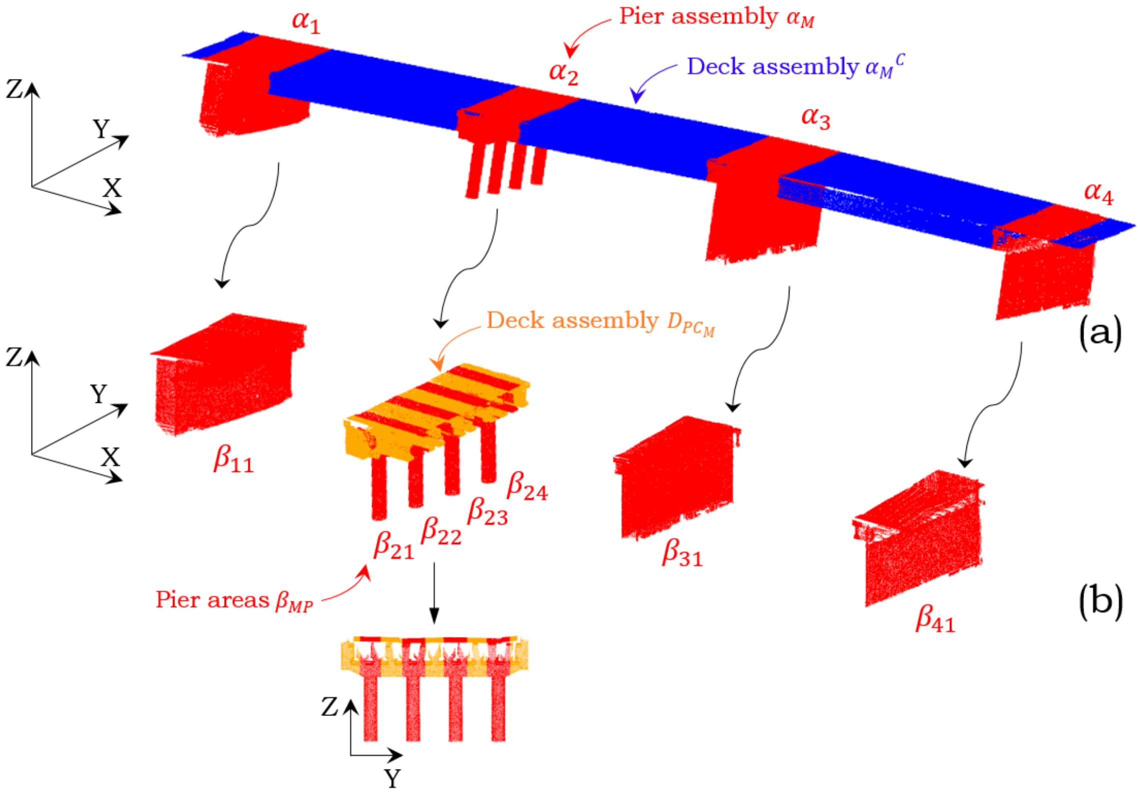


Figure 4.5 (a) Deck assembly $\alpha_M{}^C$ (blue) and pier assemblies α_M (red); (b) deck assembly D_{PC_M} (orange) and pier areas β_{MP} (red)

4.2.4. Step 3 – Pier cap detection

The workflow of this step is illustrated in Figure 4.6. This step attempts to detect pier caps (Figure 4.7) using surface normals through triangulation in the upper part of the pier area. Triangulation can be computed efficiently for a relatively small region without noticeably affecting either computation time or memory overhead.

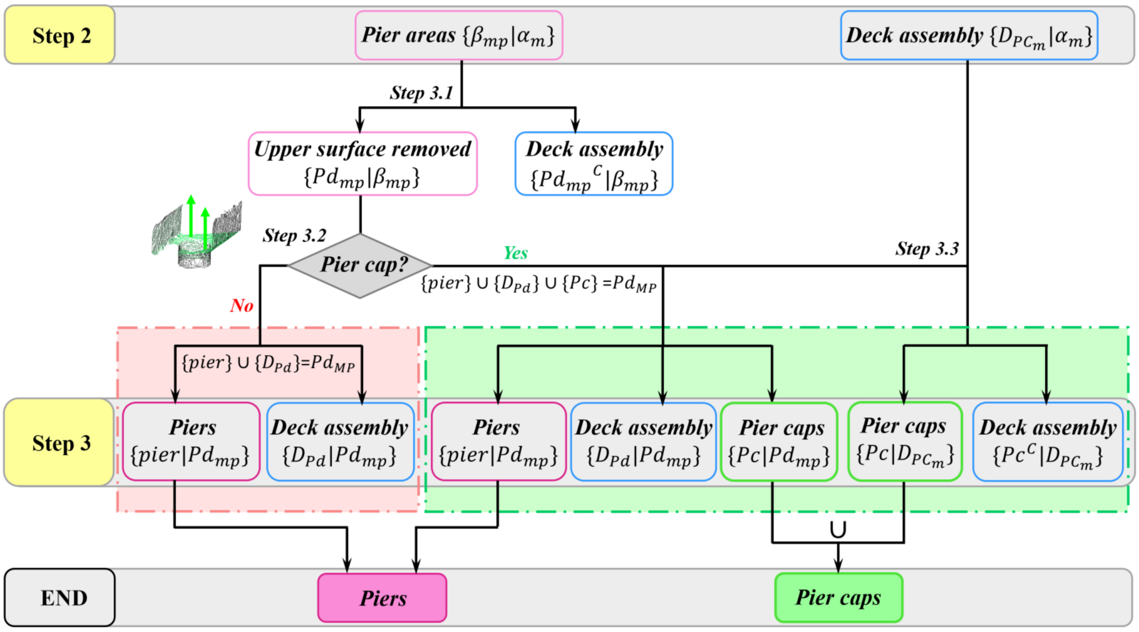

Step 2
Pier areas $\{\beta_{mp}|\alpha_m\}$
Deck assembly $\{D_{PC_m}|\alpha_m\}$
Step 3.1
Upper surface removed $\{Pd_{mp}|\beta_{mp}\}$
Deck assembly $\{Pd_{mp}{}^{C}|\beta_{mp}\}$
Step 3.2
Pier cap?
Yes
Step 3.3
$\{pier\} \cup \{D_{Pd}\} \cup \{Pc\} = Pd_{MP}$
No
$\{pier\} \cup \{D_{Pd}\} = Pd_{MP}$
Step 3
Piers $\{pier|Pd_{mp}\}$
Deck assembly $\{D_{Pd}|Pd_{mp}\}$
Piers $\{pier|Pd_{mp}\}$
Deck assembly $\{D_{Pd}|Pd_{mp}\}$
Pier caps $\{Pc|Pd_{mp}\}$
Pier caps $\{Pc|D_{PC_m}\}$
Deck assembly $\{Pc^{C}|D_{PC_m}\}$
$\cup$
END
Piers
Pier caps

Figure 4.6 Input/output flowchart of Step 3

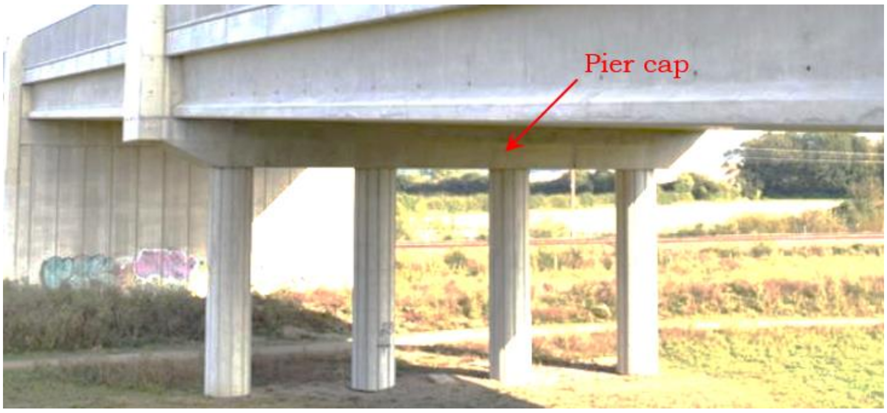


Figure 4.7 Location of a pier cap in a bridge

4.2.4.1. Step 3.1 – Remove upper deck slab surface

In this step, the method aims to remove the upper slab surface points from the pier area(s) $\{\beta_{mp}\}$ output from Step 2. The reason to remove the upper deck slab surface is that a large number of deck surface points is collected during the scanning process. These points are uninformative to detect pier caps and can be removed in advance. The void space between the upper and bottom slab surfaces is used as a discriminator. This blank part is consistent since the laser scanner can project laser beams only onto an object external surface. According to the Design Manual for Roads and Bridges (Highways England, 2018c), the general transverse maximum gradient is defined to be 5% (1/20) so that the lower bound of the upper slab points is λ_{min}=5%$W_{\beta_{mp}}$, where $W_{\beta_{mp}}$ is the width of β_{mp} (Figure 4.8) and the upper bound is $\lambda_{max} = \rho_1 H_{\beta_{mp}}$, where $H_{\beta_{mp}}$ is the height of β_{mp}. Define $\Delta\lambda$ to be the range where upper slab surface points are located. There should be 5%$W_{\beta_{mp}} < \Delta\lambda < \rho_{3a} H_{\beta_{mp}} < \rho_1 H_{\beta_{mp}}$, where ρ_{3a} is the slab thickness ratio estimation (Chapter 6, Section 6.2.4). The points in $\Delta\lambda$ are then removed and the remaining points in pier area(s) are denoted as $\{Pd_{mp}\}$ (Figure 4.8 upright).

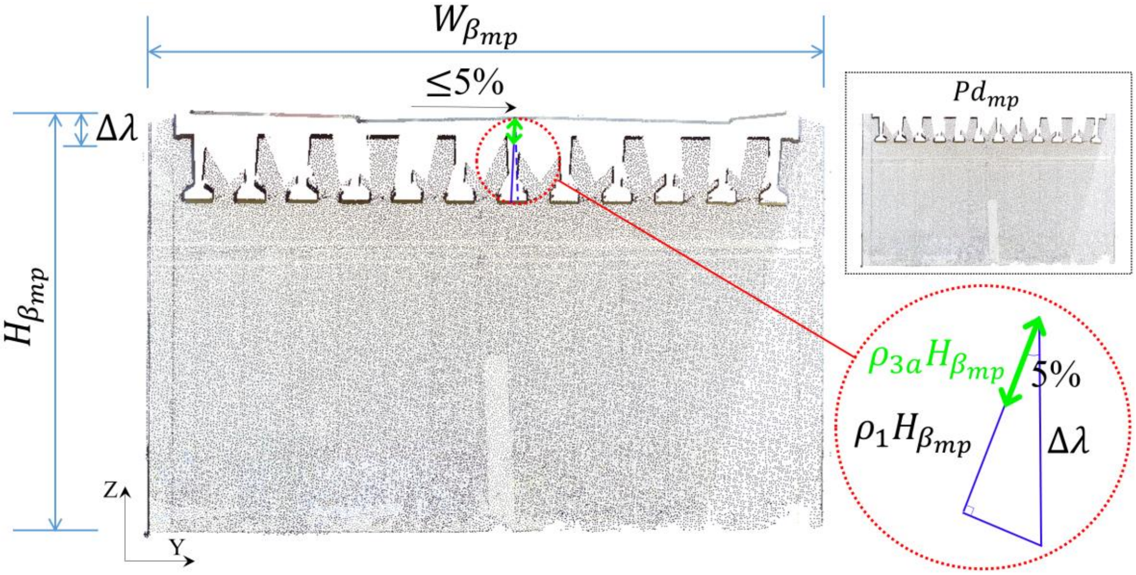


Figure 4.8 Upper slab surface range $\Delta\lambda$ in β_{mp}

4.2.4.2. Step 3.2 – Pier cap detection at top of piers

This step aims to detect pier caps $\{Pc|Pd_{mp}\}$. The input is the output of the refined pier areas $\{Pd_{mp}\}$ from Step 3.1. Pier caps lie underneath the slab, playing an important role in distributing concentrated loads from the superstructure evenly over the area of the piers. For each pier assembly α_m:

Scenario 1: a single pier area is detected in the pier assembly to which it belongs (Eq. 4.6), and the pier area extends almost the full width of the pier assembly (Eq. 4.7):

$$\text{if}\begin{cases}\beta_M = 1 & \text{(Eq. 4.6)}\\ \text{and} & \text{(Eq. 4.7)}\\ W_{\beta_{mp}} \cong W_{\alpha_m}\end{cases}$$

where $\beta_M = \{\beta_{m1}, \beta_{m2} \dots \beta_{mp}\}$ (Figure 4.8 (b)). Then, the method considers α_m to be a wall-type-pier assembly as the single wall-type-pier supports the whole deck assembly above. As a result, there is no pier cap (see A3a, Chapter 3, Section 3.4).

Scenario 2: for a pier assembly with a cap-and-column pier (i.e. $\beta_M > 1$) or, for a single detected pier area β_{mp}, but $W_{\beta_p} \ll W_{\alpha_m}$. In these two cases, the pier assembly α_m may or may not have a pier cap (e.g., multiple columns without caps). This scenario is more complex and requires further detection.

Given that pier caps are located on the top of the pier, the upper part of Pd_{mp} (the top ρ_2) is used to detect the pier cap (Figure 4.9 (a)). Denote this part as $upper_{Pd_{mp}}$, which contains the deck assembly's bottom surface, the pier cap (if it exists) and a small part of the pier (Figure 4.9 (b)). A margin part (0.5 times the length of Pd_{mp} along the Y-axis) is also temporally added on both sides of the upper part of pier. Then, the method generates the mesh for $upper_{Pd_{mp}}$ with the margins and computes the normal of each triangular surface. The margin parts ensure enough normals to be captured. Estimating a normal per given triangle is completed by the cross product of two vectors on this triangle. Define $\{\vec{n}_t\,(n_t^x, n_t^y, n_t^z): t = 1,2,\dots,T\}$ to be the normal vectors of the triangles. The normal indicates the surface orientation. If a cluster of surface normals is revealed in $upper_{Pd_{mp}}$ whose orientations are quasi-parallel to the Z-axis (i.e. downward- or upward-oriented normal), where

$$-90° < \arctan(\theta_t) < -85° \\ \text{or} \\ 85° < \arctan(\theta_t) < 90°, \text{where } \theta_t = \frac{n_t^z}{\sqrt{{n_t^x}^2 + {n_t^y}^2}}, \qquad \text{(Eq. 4.8)}$$

and if those normals are found around the level $\rho_1(\max\{z_i|\beta_{mp}\} - \min\{z_i|\beta_{mp}\})$ (Figure 4.9 (c) red), then the points (feature points) constituting these surfaces together with the points in $upper_{Pd_{mp}}$ that lie above the feature points are classified as the deck assembly. Otherwise, the pier cap feature points are detected if a cluster of downward- or upward-oriented normals is found around the level $\rho_2(\max\{z_i|\beta_{mp}\} - \min\{z_i|\beta_{mp}\})$ (see A3b, Chapter 3, Section 3.4) (Figure 4.9 (c) green). The method iterates through $\{Pd_{mp}\}$ using the same procedure and the pier caps $\{Pc|Pd_{mp}\}$ and the piers $\{pier|Pd_{mp}\}$ are acquired.

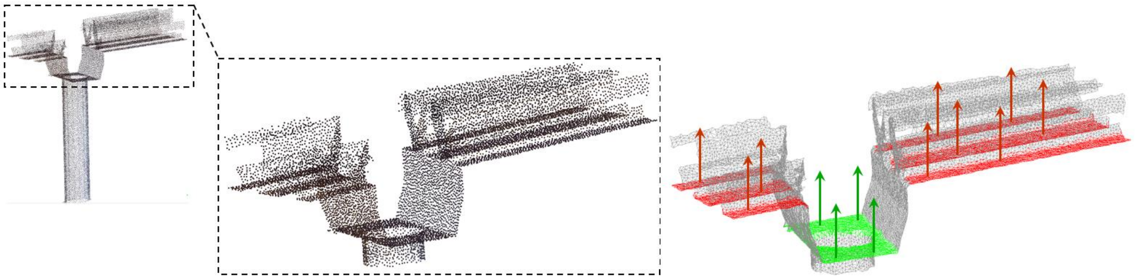

(a) Pd_{mp} with margins (b) $upper_{Pd_{mp}}$ with margins (c) Normal detection

Figure 4.9 Pier cap detection using surface normals

4.2.4.3. Step 3.3 – Pier cap extraction from deck assembly

The detected pier caps in Step 3.2 imply that they should also be present in D_{PC_M}, which is the deck assembly output from Step 2. The pier cap parts from D_{PC_M} are extracted in the following way, where $D_{PC_M} = \{D_{PC_m}\}$ (Figure 4.10 (a)).

First, the points of D_{PC_m} are projected onto the YZ-plane followed by generating density histograms along the Y-axis through which the number of points is tallied within multi-equal-width bins. The width of bin is determined using the square-root choice:

$$bin_w = \frac{|\max\{y_i|Cluster\} - \min\{y_i|Cluster\}|}{\lceil\sqrt{n}\rceil}, \qquad \text{(Eq. 4.9)}$$

where $Cluster$ represents D_{PC_m} and n is its point count. Then, the bins are clustered using the gaps between them (Figure 4.10 (b)), so that D_{PC_m} is segmented (Figure 4.10 (c)). Denote the segments as $\{\gamma_{m(p+1)}\}$ and then a slicing procedure along the X-axis of $\{\gamma_{m(p+1)}\}$ is performed. For $\gamma_{m(p+1)}$, the pier cap area is detected if:

$$range_{j\langle z\rangle} > \rho_{3b}\left|\max\{z_i|\gamma_{m(p+1)}\} - \min\{z_i|\gamma_{m(p+1)}\}\right|, \qquad \text{(Eq. 4.10)}$$

where $\rho_{3b} = \frac{\rho_1}{\rho_2}$ with ρ_1 and ρ_2 being determined in Step 1 and 2, respectively (see A3c, Chapter 3, Section 3.4) (Figure 4.10 (d)). Next, the procedure is similar to Step 3.1 and Step 3.2. The extracted pier cap area is considered to be a smaller scale of β_{mp}. The upper slab surface points are removed and classified as the deck assembly, followed by triangulation to detect and classify the deck assembly's bottom surface points. The pier cap parts $\{Pc|D_{PC_m}\}$ are finally acquired (Figure 4.10 (f)). In the end, both pier cap parts output from Step 3.2 and Step 3.3 are merged (Figure 4.11).

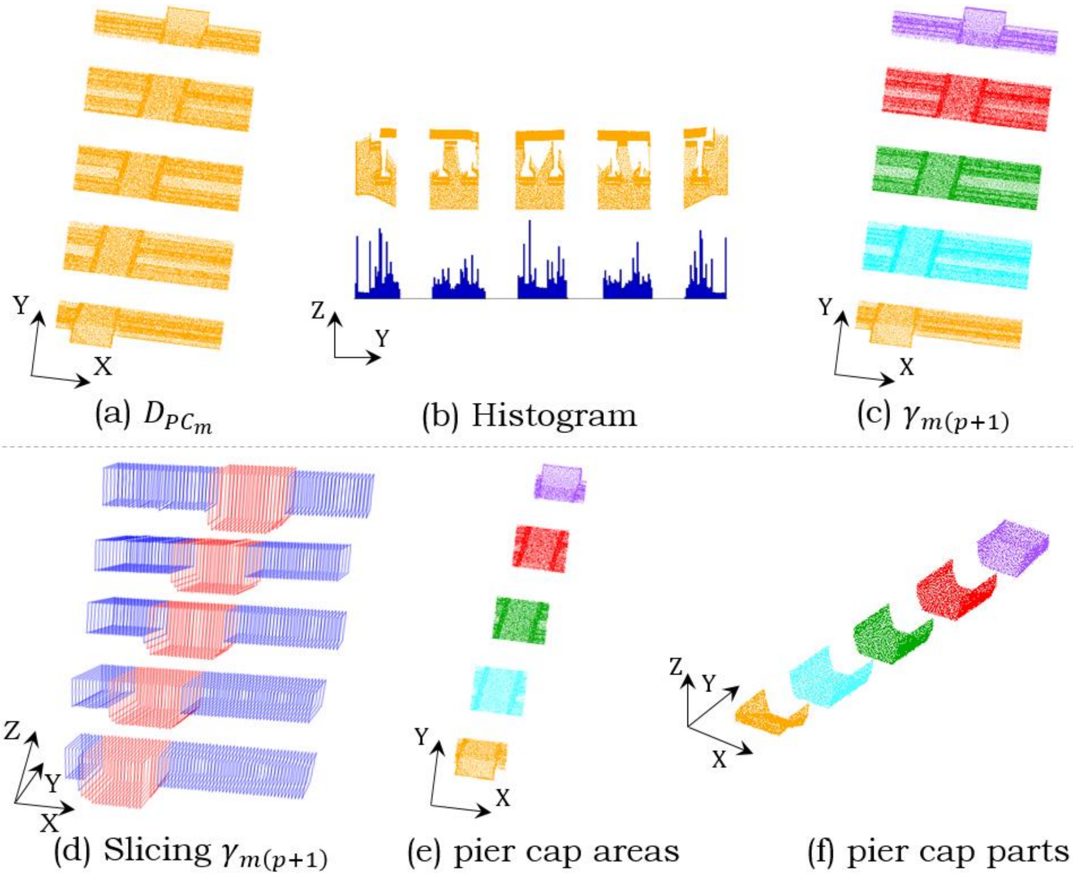


Figure 4.10 Extract pier cap from D_{PC_M}

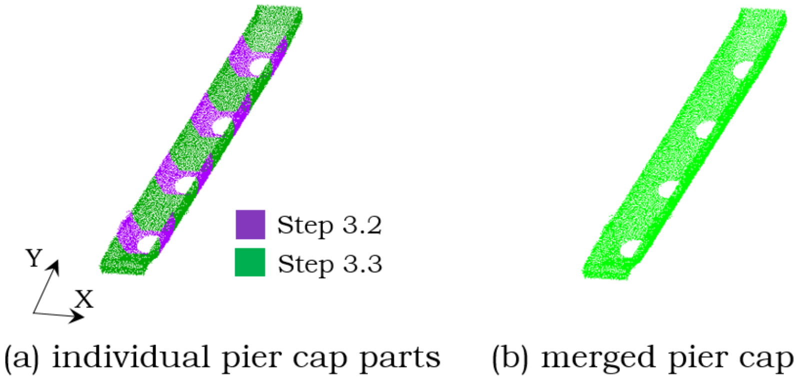


Figure 4.11 Merge pier cap parts

4.2.5. Step 4 – Girder detection

Step 4 aims to detect girders in the deck assembly. This is achieved in two sub-steps: Step 4.1, segment the deck assembly into several segments $\{deck_\omega\}$ and Step 4.2, detect girders in each segment $\{girders|deck_\omega\}$.

4.2.5.1. Step 4.1 – Segment the deck assembly into several segments

To begin with, the method conducts a merging process to build up a whole deck assembly cluster, which is composed of the slab and girders (if they exist). This involves piecing together all point clusters classified as deck assembly in the previous steps (Figure 4.12).

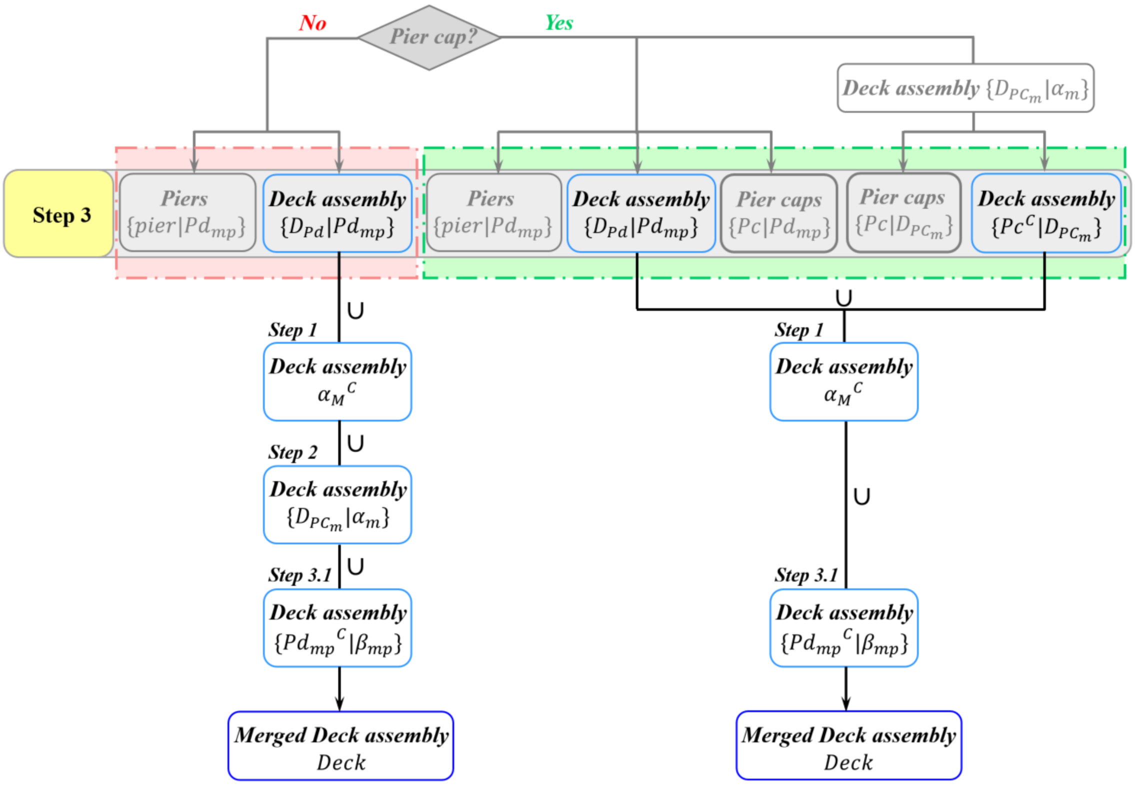

No
Pier cap?
Yes
Deck assembly $\{D_{PC_m}|\alpha_m\}$
Step 3
Piers $\{pier|Pd_{mp}\}$
Deck assembly $\{D_{Pd}|Pd_{mp}\}$
Piers $\{pier|Pd_{mp}\}$
Deck assembly $\{D_{Pd}|Pd_{mp}\}$
Pier caps $\{Pc|Pd_{mp}\}$
Pier caps $\{Pc|D_{PC_m}\}$
Deck assembly $\{Pc^C|D_{PC_m}\}$
∪
∪
Step 1
Deck assembly $\alpha_M{}^C$
Step 1
Deck assembly $\alpha_M{}^C$
Step 2
∪
Deck assembly $\{D_{PC_m}|\alpha_m\}$
∪
Step 3.1
∪
Deck assembly $\{Pd_{mp}{}^C|\beta_{mp}\}$
Step 3.1
Deck assembly $\{Pd_{mp}{}^C|\beta_{mp}\}$
Merged Deck assembly
Deck
Merged Deck assembly
Deck

Figure 4.12 Merging to acquire the deck assembly

For a beam-slab bridge, the length of the girder (beam) depends on the span, which is the distance between the two intermediate supports (Wai-Fah & Lian, 2014). The merged deck assembly needs to be split into several segments in order to find the appropriate length of the span. The best cutting planes are not necessarily orthogonal to the horizontal alignment of the bridge (along the central axis of the deck slab), but rather depend on the orientation of the expansion joints. This is because two adjacent deck assembly segments must be interconnected by the expansion joints as per the Highway Agency's BD 33 design code (Highways England, 1994). The choice of joint depends on many factors, including imposed loadings, anticipated movement, temperature range, deck shortening and deck rotation. Pier clusters and pier caps are then oriented based on the joints. Hence, the problem of finding the best cutting planes is transformed into determination of the orientation of the pier clusters or pier caps (Figure 4.13). A 3D oriented-bounding-box (OBB) is employed to capture the orientation property. OBB is the tightest oriented box depicting a given 3D point set. Specifically, SVD is used to acquire the new basis of the 3D point set in a sense of PCA, i.e. U (Eq. 4.3). Next, the method finds the axis-aligned-bounding-box (AABB) of the point set by projecting the points to the new coordinates system, followed by mapping the AABB back to the original coordinates system using the inverse matrix of U (i.e. U^{-1}). In this way, the OBB vertices are found in the original coordinates system.

All bridges have piers, but not all have pier caps. Hence, a pier cluster is used to create an OBB. Only its bottom-half is used in order to avoid the retained boundary points from the pier cap as those points might lead to a shifted OBB.

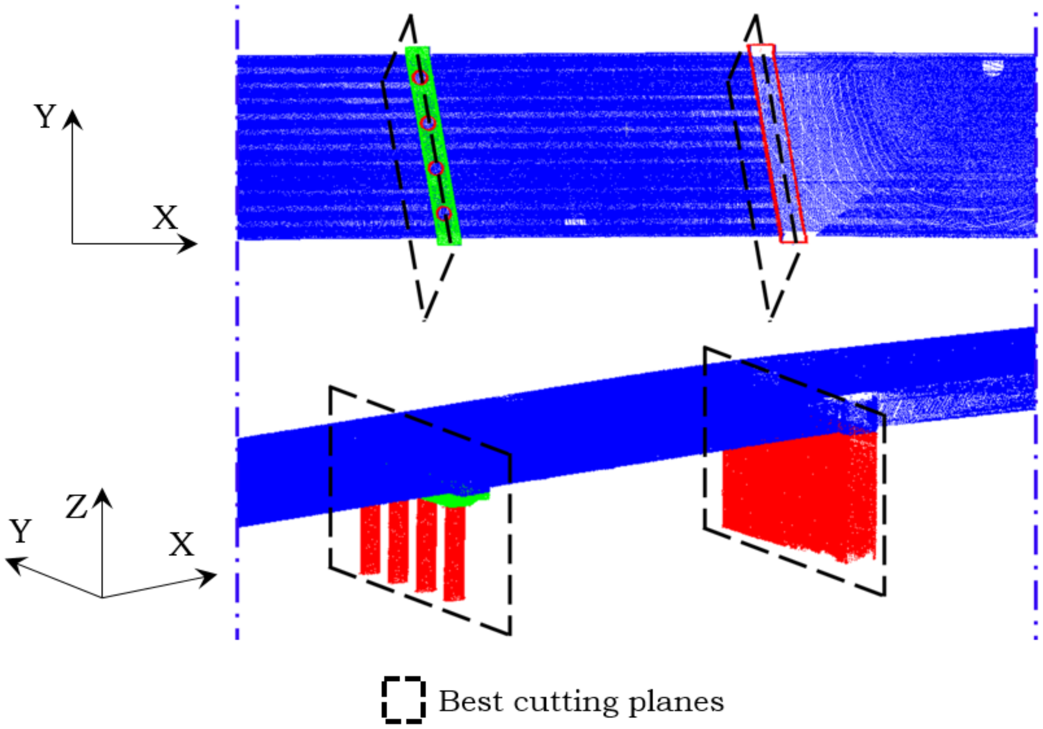


Figure 4.13 Best cutting planes

OBB replaces a point set with a parallelepiped of 12 edges and 8 vertices. Let $\{d_1, d_2, d_3, d_4\}$ and $\{p_1, p_2, p_3, p_4\}$ be the 4 upper vertices of the merged deck assembly $Deck$ and the bottom-half part of a pier cluster, repectively. Let M1, M2 be the mid-points of $line_{p1p2}$ and $line_{p3p4}$. MP1 and MP2 are two orthogonally projected points of M1 and M2 onto the plane pl_{d1-d3}. Then, d_5 and d_6 are two intersection points of $line_{d1d2}$ & $line_{MP1MP2}$ and $line_{d3d4}$ & $line_{MP1MP2}$ in the plane pl_{d1-d3}, respectively (Figure 4.14).

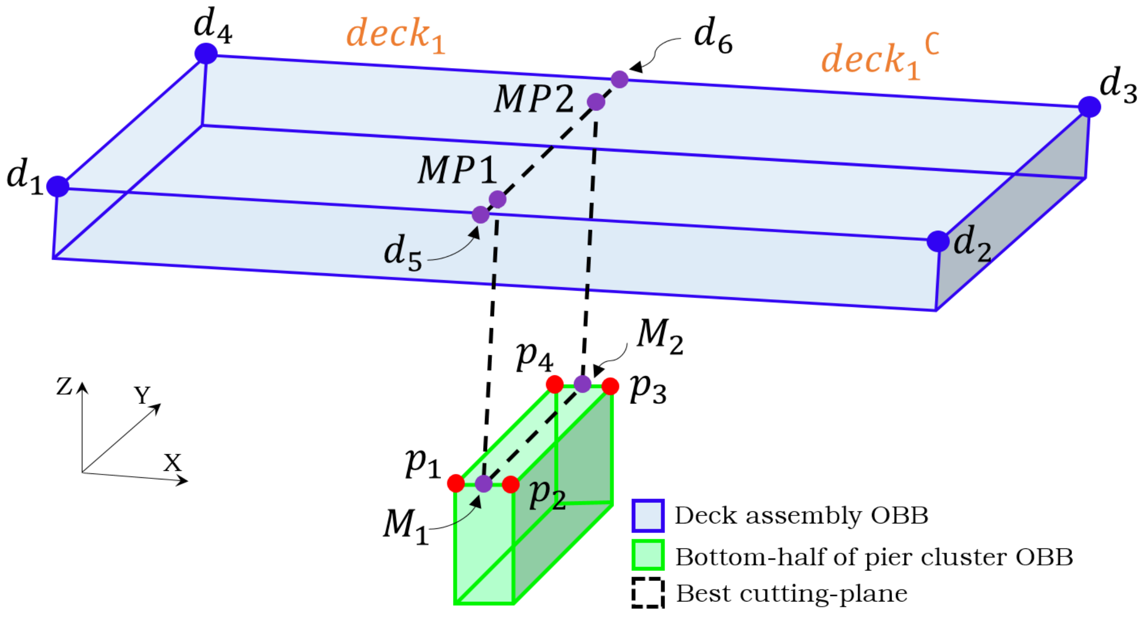


Figure 4.14 Find the best cutting plane

Specifically, given two straight lines of the form:

$$\begin{aligned} a_1x + b_1y + c_1 &= 0 \\ a_2x + b_2y + c_2 &= 0\,' \end{aligned} \qquad \text{(Eq. 4.11)}$$

the intersection point (x_i, y_i) is given by:

$$x_i = \frac{\begin{vmatrix} b_1 & c_1 \\ b_2 & c_2 \end{vmatrix}}{\begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix}}$$

and

$$y_i = \frac{\begin{vmatrix} c_1 & a_1 \\ c_2 & a_2 \end{vmatrix}}{\begin{vmatrix} a_1 & b_1 \\ a_2 & b_2 \end{vmatrix}}\,, \qquad \text{(Eq. 4.12)}$$

where the denominator is the determinant of (a_1, b_1) and (a_2, b_2). If this determinant is zero, the equations Eq. 4.12 are linearly dependent, indicating that there is no intersection. Herein, the determinant is assumed to be non-zero.

$Deck$ is then cut along the plane consisting of d_5, d_6, M1, and M2 so that the plane pl_{MP1-M2} is deemed the best cutting plane. Next, the method examines whether each point of the deck assembly lies inside the $ConvexHull_{\langle d1d4d6d5 \rangle}$:

if $\forall\, p_i \in \{p_{i\langle x,y \rangle} | Deck_{\langle x,y \rangle}\}$ is inside $Polygon_{\langle d1d4d6d5 \rangle}$: $\{p_i | Deck\} \in deck_1$, otherwise, $\{p_i|Deck\} \in {deck_1}^{C}$.

This is achieved using Hessian Normal Form (Vince, 2013), which provides a useful means of partitioning space into two zones: one is for points above a given line in the partition that includes that normal vector, and the other one is for points in the opposite partition. As shown in Figure 4.15, a point (x, y) on the line satisfies $ax + by + c = 0$. A point (x, y) in the partition in the direction of the normal vector satisfies $ax + by + c > 0$. A point (x, y) in the partition opposite to the direction of the normal vector satisfies $ax + by + c < 0$.

This process is recursively performed until the entire deck assembly is segmented into multiple segments $\{deck_\omega : \omega = 1,2, \dots (m + 1)\}$, where m is the number of pier clusters (equal to the number of pier assemblies) (Figure 4.16).

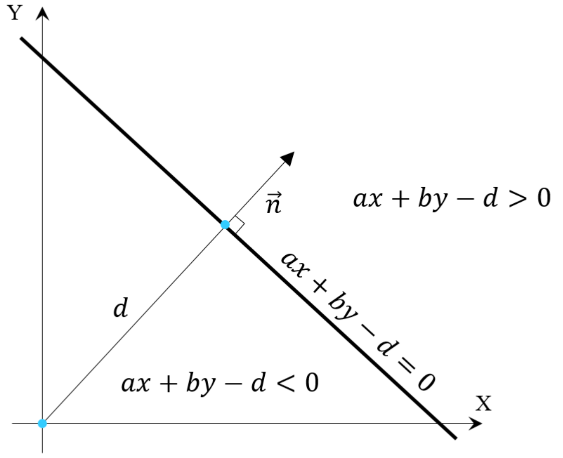


Figure 4.15 Partition space into two zone using Hessian Normal Form

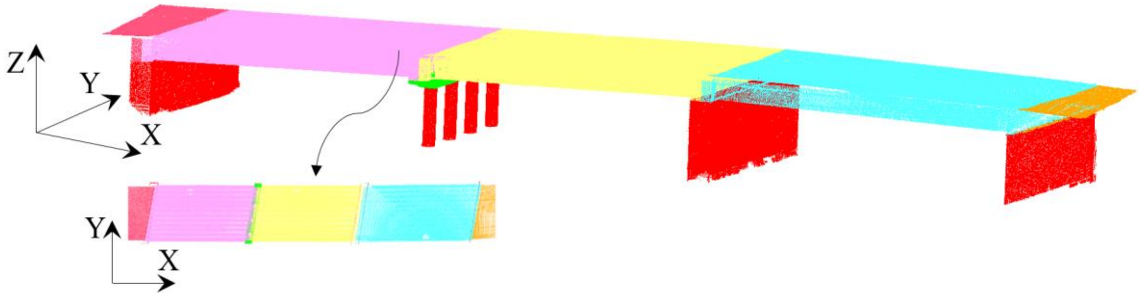


Figure 4.16 Split the whole deck assembly into $\{deck_\omega\}$

4.2.5.2. Step 4.2 – Girder detection in the deck assembly segment

In this step, the method detects girders in each deck assembly segment. It starts by rotating $deck_\omega$ around its Y-axis using:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\xi) & 0 & \sin(\xi) & 0 \\ 0 & 1 & 0 & 0 \\ -\sin(\xi) & 0 & \cos(\xi) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}, \qquad \text{(Eq. 4.13)}$$

where (x', y', z') are the updated point coordinates at each rotated angle ξ of point (x, y, z), until $deck_\omega$ reaches the best projection view, because the original projection results of $deck_\omega$ might be “muddy” due to a curved bridge elevation. Rotation is conducted through a grid search (step=0.01°) over a range of angles $\{\xi\}$, where $\{\xi\}$ = [-3.4°, 3.4°], deduced from the general longitudinal maximum gradient (6%) (Highways England, 2018c). A density histogram $\mathcal{H}_Z$ along the Z-axis is employed for evaluating whether a best rotation has been reached. The best rotation angle is determined using:

$$\breve{\xi} = \operatorname{argmax}\{std_{\mathcal{H}_Z(deck_\omega)}\}, \qquad \text{(Eq. 4.14)}$$

where $std_{\mathcal{H}_Z(deck_\omega)}$ is the standard deviation of the point counts in the bins. Empirical studies revealed that the best projection determination is not sensitive to the bin size (varies from #bin=10 to #bin=1000, μ=2.70°, σ=0.05°) (Figure 4.17). The bin size is then derived using (Eq. 4.9). $std_{\mathcal{H}_Z(deck_\omega)}$ is a stronger indicator than simply the maximum point count bin, because the elevation of the girder depends on that of the slab (see A13, Chapter 3, Section 3.4). The best projection may not necessarily be given by the bin with the maximum point count, which may possibly result from a concentration of unevenly distributed points. The best projection view can be found once the standard deviation of the histogram bins on the Z-axis reaches its maximum. Figure 4.18 demonstrates an example where the best rotation ($\breve{\xi}$=2.7°) is obtained when the biggest $std_{\mathcal{H}_Z(deck_\omega)}$ (1178) returns.

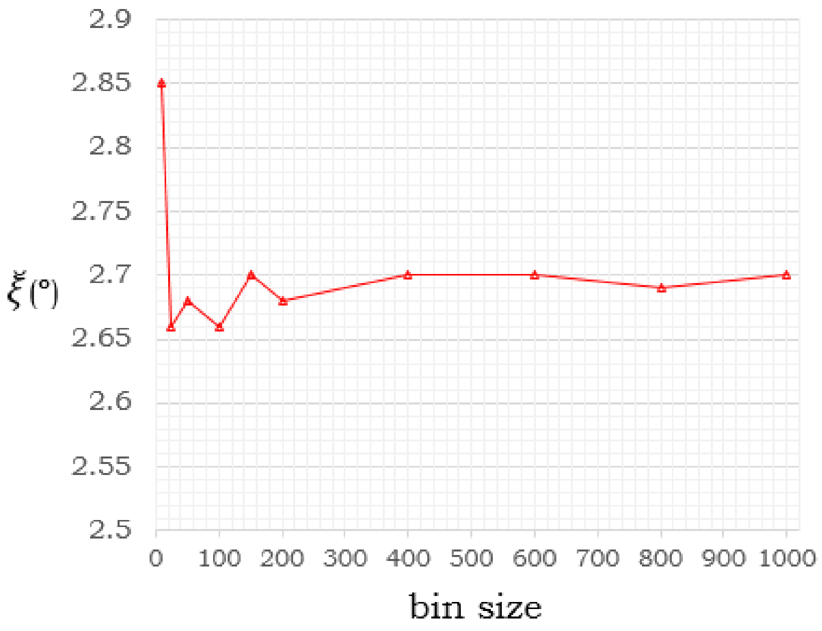


Figure 4.17 Best rotation found vs. Different bin size

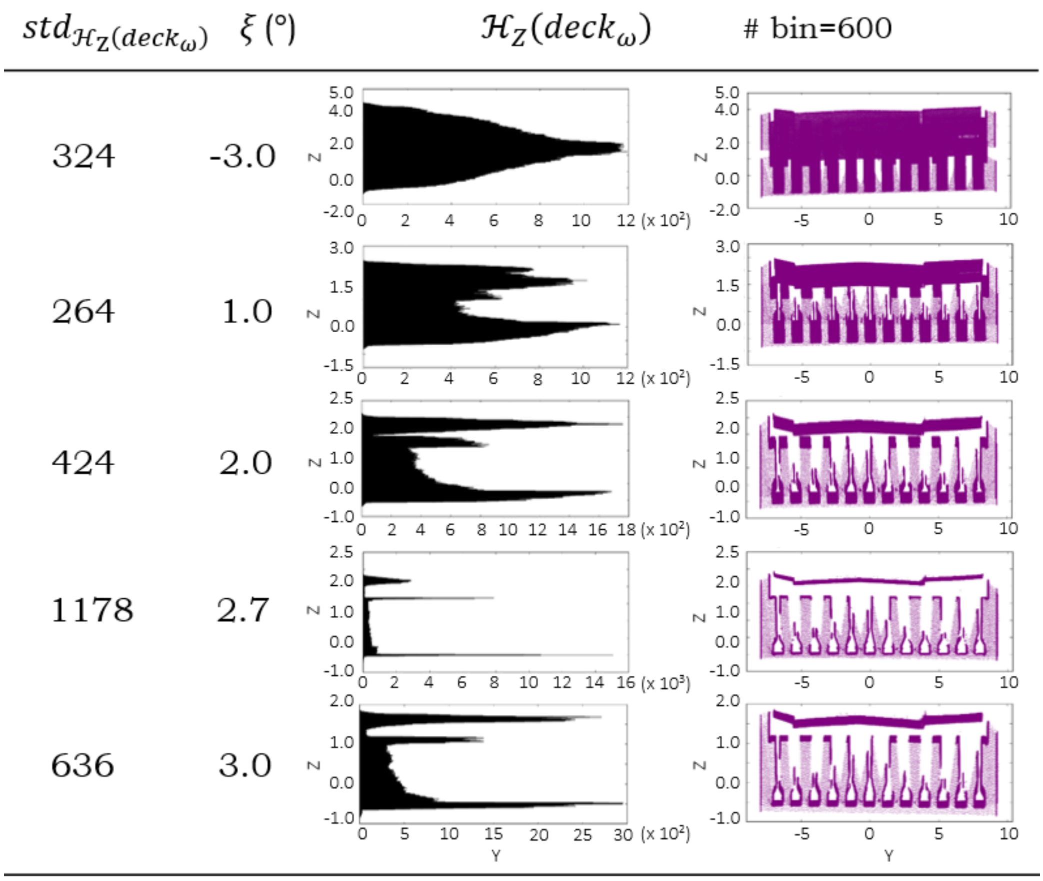


Figure 4.18 Best rotation search in $deck_\omega$

Next, only the bottom ρ_4 (%) points of $deck_{\omega(\xi)}$ (denoted $b_{deck_{\omega(\xi)}}$) are used for girder detection, where $\rho_4 = \frac{\rho_1 - \rho_{3a}}{\rho_1}$, with the thickness of the deck assembly as well as that of the slab being estimated to be roughly ρ_1 (obtained from Step 1) and ρ_{3a} (obtained from Step 3.1), respectively. The removal of the deck assembly's upper part (top (1- ρ_4)) is crucial as many more points are captured from the deck external surface, overpowering the girder points and rendering the geometric features uninformative. The extremities of $deck_{\omega(\xi)}$ are also excluded to avoid noise from bridge accessory components. Density histograms $\mathcal{H}_Y$ are drawn along the Y-axis of $b_{deck_{\omega(\xi)}}$ using (Eq. 4.9) ($Cluster \leftarrow b_{deck_{\omega(\xi)}}$) followed by generating the normalized probability of the point density (Figure 4.19). The density probability is uniformly distributed with significantly lower variance when there is no girder (slab bridge) while significant peaks can be observed in the distribution with non-trivial variance when girders exist (beam-slab bridge). For the latter, a binary list (0, 1) is created after thresholding out all small counts (small-count bin ← 0, big-count bin ← 1). This list is further de-noised through a simple k-NN filter, which works as a voting scheme. It checks the label of neighbouring bins, and then assigns a candidate label to the investigated bin. This process is iteratively performed until optimal clusters are returned, meaning that the "1" chunks have similar length because the girder section type is identical over a specific span (see A4, Chapter 3, Section 3.4). The bottom flange can infer a collection of possible girder section types for Y, U or SY beams (Figure 4.20) so that it is not enough to segment the girders from the slab using the bottom flange information alone. The height information is necessary as well. The method uses the best project view to infer the web depth (the height of the girder), which is then used to separate the girders from the slab.

All the over-segments from Step 1 to Step 4 are merged as per their class labels. The four-step top-down recursive detection method then terminates.

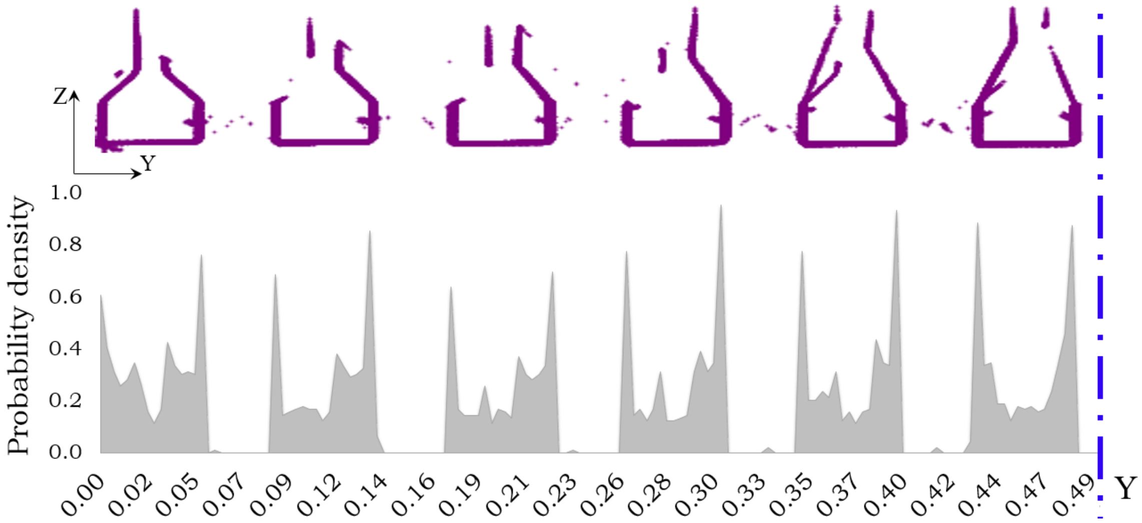


Figure 4.19 Girder segmentation

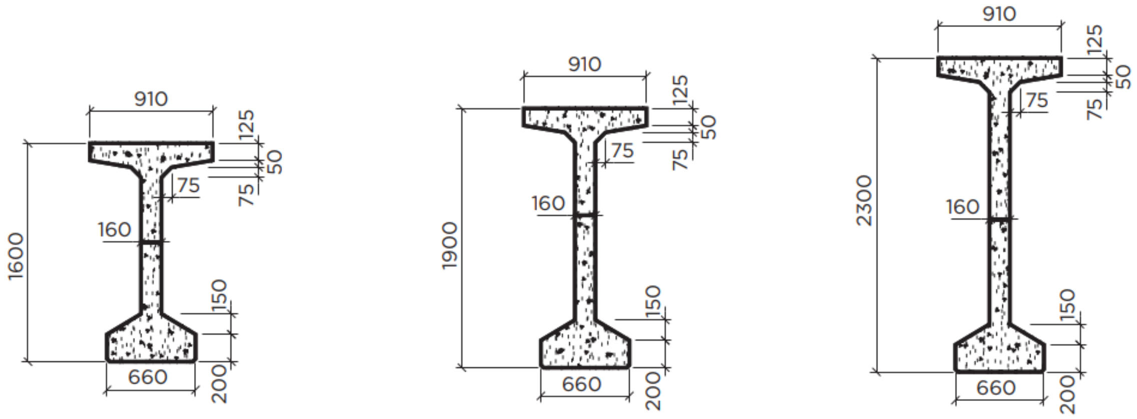


Figure 4.20 Different types of girder section with the same flange width

4.2.6. Summary

This chapter presents the first process of the proposed framework in more detail. It derives the novel top-down object detection method able to detect four types of bridge components in real PCD. The key idea behind this method is its reliance on slicing to deal with the skew complexity of real bridge geometries and to quickly locate the targeted objects. The author explains how the method first separates the deck assembly from pier assemblies, followed by detecting and segmenting pier caps using their surface normals, and girders using oriented bounding boxes and density histograms. Finally, the method merges over-segments into individually labelled point clusters. The novelty of this method lies in the fact that it bypasses altogether the generation of low-level shape primitives such as surfaces, and directly produces the key components of RC bridges in the form of labelled point clusters.

The outputs of this chapter are four types of bridge component in the form of labelled point clusters. The experiments, results, and discussions of this method are presented in Chapter 6, Section 6.2.4. The author elaborates in the following chapter how to fit IFC objects to the four types of labelled point cluster.

5. Fitting IFC Objects to Labelled Point Clusters

The **objective** of this chapter is to provide an automatic method to fit 3D solid models using current IFC standards to the labelled point clusters making up a bridge. This objective is achieved by answering the **research question**: How to effectively extract the geometric features of the four types of labelled point clusters of real bridge PCD and model them to the corresponding IFC standards?

5.1. Introduction

The author aims to (1) extract the geometric features of four types of labelled point clusters constituting a bridge, and (2) represent these features using current IFC standards in order to convert the point clusters into 3D solid models in this chapter. To this end, the author proposes a novel object fitting method. The novelty of this method lies in the fact that it uses multiple slice models to approximate the real bridge geometry.

The author contends this chapter is both causal and exploratory in nature. This means that, on one hand, this chapter proposes a method to solve the problem of bridge gDT modelling from labelled point clusters. On the other hand, this chapter also seeks to improve our understanding of the specific challenges of bridge gDT modelling and the extent to which it has been resolved.

Chapter 4 can automate the object detection step through a top-down object detection algorithm and directly produce labelled point clusters of four component types. However, the problem of automatic conversion from the labelled point clusters into 3D solid IFC models, remains unsolved. This chapter aims to solve this problem in this regard.

The challenges exhibited in bridge gDT generation are due to the irregular geometry of existing bridges. The geometric definition for buildings, for example, is basically developed on a grid system, whereas bridges are defined with curved horizontal alignments, vertical elevations and varying cross-sections. In addition, the as-weathered and as-damaged conditions further increase the geometry complexity, which renders simplistic modelling methods ineffective when subject to real bridge components. This is especially true because existing studies have attempted to fit generic geometric primitives that make up a bridge, from labelled point clusters (Zhang et al., 2014). These studies fail to justify that the models generated by their proposed methods are indeed in line with the as-is reality. The 3D shapes produced by these methods are too ideal to depict the real geometry of bridge objects such that the models do not overlap the point clusters. In addition, no method has explicitly explained how to convert bridge point clusters into IFC objects that can be exchanged between different partners in a project. And no method has provided a sound validation stage to evaluate the quality of the generated gDT.

To address the above-mentioned challenges, this chapter intends to provide a novel object fitting method that can automatically and rapidly generate the gDT of an existing RC bridge in IFC format, using the four types of point cluster constituting this bridge.

5.2. Proposed method

5.2.1. Overview

The inputs of the proposed method in this chapter are the refined point clusters generated from the previous step, i.e. object detection. The outputs of this chapter are two IFC files, containing various *IfcObjects* (components and attributes) making up a bridge gDT and corresponding to two different levels of detail (LOD 200 and LOD 250—300).

Figure 5.1 illustrates the workflow of the proposed method in this chapter. The method consists of two major steps, namely Step 1, geometric feature extraction and shape detection in point clusters of the four types of structural component; and Step 2, *IfcObjects* fitting for the extracted features and identified shapes. In addition, for one set of bridge point cluster data, the proposed method produces two .ifc files at two different levels of detail (LOD 200 gDT file and LOD 250—300 gDT file), containing the four components making up the bridge. A LOD 200 bridge gDT uses an Oriented Bounding Box (OBB) to represent each component, whereas a LOD 250—300 bridge gDT represents each component in a stacked way. The geometry of a deck slab point cluster is approximated using multiple oriented slice models along with its horizontal alignment. The geometry of a pier cap point cluster is represented by extruding its XY-plane outline. For a pier point cluster, the method first checks its shape and then decides whether to represent it as a generic shape primitive or to represent it using stacked slices. Last, for a girder point cluster, the method leverages template matching to fit it to a specific profile from a precast concrete catalogue. The method encodes all the extracted geometric features used to describe the shape of a component with the current IFC standards (IFC4 Add2).

The **hypothesis** of this chapter proposed in this chapter is: the slicing-based bridge-component fitting method can generate high quality gDT of an existing RC bridge in IFC format and there is no significant difference in the modelling accuracy for different RC bridges. In addition, the modelling time is much less compared to the current practice. This hypothesis will be tested with a PCD collection of ten highway RC bridges in the UK.

The reason for generating a bridge gDT with different resolutions is that the appropriate level of geometric detail required for different tasks is different, depending on the context it will be used for. Specifically, asset owners need to review their requirements both for new and existing projects. The project team needs to agree what information at what detail and quality level needs to be provided at what stage of the project. Both coordination needs and phase handover needs are to be considered. To support these processes, there is a need to have a common ground for open discussion. LOD concept can be used both as a basis for contract documents and as a tool to communicate needs and wants. It is therefore important that the end-users define the LOD that is required at each stage of development of the project according to the EURs. However, at present, there is no standardised definition for the information exchange or for specific LOD, other than the suggestion that they should be aligned with end-users' decision points and should be consistent across all appointments. A definition and specification of the level of geometric detail required for (as-is) gDT generation in accordance with the EURs is highly needed but is outside the scope of this thesis. Therefore, the author proposes a method to generate a bridge gDT with different resolutions based on the existing very broad guidance discussed in the following section such that it is more flexible to adapt to current and future needs.

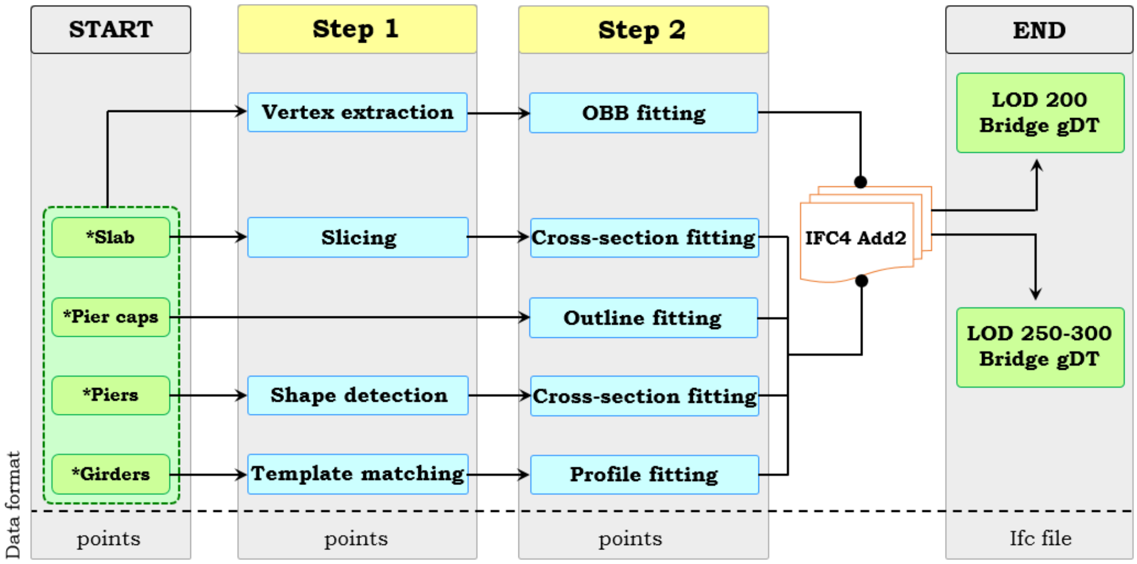


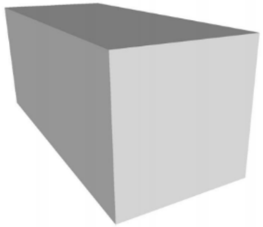
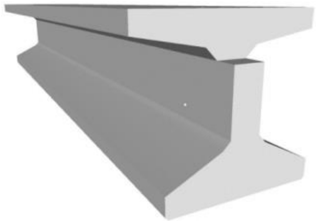
Figure 5.1 Workflow of the proposed IFC fitting method

5.2.2. Level of Detail (LOD)

By definition, the level of detail (LOD) is how much detail is included in the graphical representation of an object. It defines how the 3D geometry of an asset can achieve different levels of refinement. By contrast, the Level of Development is how much detail is included in both the graphical representation and the other information, such as properties, material, cost, etc. It represents the degree to which project team members may rely on the information when using the model. As the standardized specification for (as-is) gDT generation according to the EURs is missing in literature, the author leveraged the LOD specification suggested by BIMForum (2018) as guidance. Table 5.1 illustrates an example of the interpretation of the LOD for a highway bridge component: a concrete precast girder. Herein, LOD specifically refers to the level of geometry refinement of a bridge component can achieve, is used as a measure of the service level required.

Simple model with lower LOD, such as LOD 200, is graphically represented as a generic system, object, or assembly with approximate size, shape, location, and orientation. LOD 200 is approved to use in analysis, cost estimating, and scheduling during design stage or to reflect the approximate reality of the sensed infrastructure. It is especially useful when a quick 3D visualization of the schematic layout is preferred rather than the model accuracy and precision, as per the needs of end-users. The LOD increases as the project requirement proceeds, often developing from a simple design intent model through to a detailed virtual construction model, then an as-is DT. For example, the model element of LOD 300 is graphically represented as a specific system, object, or assembly accurate in terms of quantity, size, shape, location, and orientation. This level is approved to support process such as analysis, cost estimating, bidding, scheduling during construction and post-construction stage. LOD 300 should reflect the accurate geometry of a sensed infrastructure. However, LOD 300 does not include information such as detailing, fabrication, installation, and detailed assemblies, which are necessary to reflect the actual status of an existing infrastructure. LOD 350 and higher LODs contain enriched information that reflect the as-is status of an existing infrastructure. However, various additional sensors are required to capture this embedded information that is invisible to a laser sensor. Extracting this information is beyond the scope of this thesis. This thesis therefore only focuses on generating LODs that can be achieved through laser scanning alone.

The proposed method in this chapter first generates a bridge gDT that is in line with the LOD 200: (1) a model in a coarse level representation of the underlying point clusters. This model representation uses the Oriented Bounding Box (OBB) of each point cluster to generate *IfcObjects*. Details are given in Section 5.2.4. Then, the method generates a bridge gDT with a LOD that is higher than LOD 200 but may not necessarily in line with LOD 300. Thus, herein, the author names it a LOD 250—300 gDT: (2) a geometrically more detailed model with a better approximation of the real geometry. In general, this model representation uses stacked slice models to describe point clusters to generate *IfcObjects*. Details are given in Section 5.2.5.

LOD	Interpretation	Element modelling to include	Schema
100	Elements are not geometric representations. Examples are information attached to other model elements or symbols showing the existence of a component but not its shape, size, or precise location. Any information derived from LOD 100 elements must be considered approximate.		
200	Elements are generic placeholders. They may be recognizable as the components they represent, or they may be volumes for space reservation. Any information derived from LOD 200 elements must be considered approximate.	• Type of structural concrete system • Approximate geometry (e.g. depth) of structural elements	
300	The quantity, size, shape, location, and orientation of the element as designed can be measured directly from the model without referring to non-modelled information such as notes or dimension call-outs. The project origin is defined, and the element is located accurately with respect to the project origin.	• Specific sizes and locations of main concrete structural members modelled per defined structural grid with correct orientation • Concrete defined per spec (strength, air entrainment, aggregate size, etc.) • All sloping surfaces included in model element with exception of elements affected by manufacturer selection	

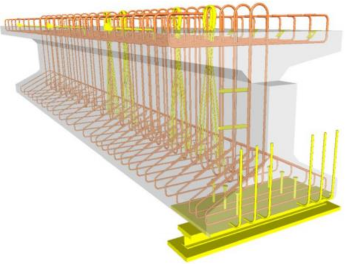
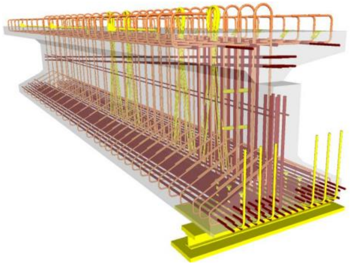
350	Parts necessary for coordination of the element with nearby or attached elements are modelled. These parts will include such items as supports and connections. The quantity, size, shape, location, and orientation of the element as designed can be measured directly from the model without referring to non-modelled information such as notes or dimension call-outs.	• Reinforcing Post-tension profiles and strand locations • Reinforcement called out, modelled if required by the BXP, typically only in congested areas • Chamfer • Pour joints and sequences to help identify reinforcing lap splice locations, scheduling, etc. • Expansion Joints • Lifting devices • Embeds and anchor rods • Post-tension profile and strands modelled if required by the BXP • Penetrations for items such as MEP • Any permanent forming or shoring components	
400	Elements are modelled at sufficient detail and accuracy for fabrication of the represented component. The quantity, size, shape, location, and orientation of the element as designed can be measured directly from the model without referring to non-modelled information such as notes or dimension call-outs.	• All reinforcement including post tension elements detailed and modelled	

Table 5.1 LOD Specification for Highway Bridge Precast Structural Girder (BIMForum, 2018)

5.2.3. MVD and IFC Entities

As earlier mentioned in the thesis (Chapter 2, Section 2.2.2), IFC documentation is highly focused on buildings. To date, IFC is not fully ready for sharing data models of infrastructure linear projects. IFC-Bridge is an extension of the IFC standard for the bridge sector, which is in ongoing development. Whilst IFC has traditionally focused on buildings, it was agreed that for purposes of management, less is more, as the barrier to entry for software vendors who already support IFC must be kept at a minimum to achieve industry adoption. Hence, it is preferred to use existing applicable data structures and IFC entities whenever possible (buildingSMART, 2018). For instance, the *IfcBeam* data type can be used to define a bridge girder in the same way as in buildings.

As discussed previously in the thesis, MVD refers to a smaller subset of the model data schema that provides a complete representation of the information needed for a specific data exchange scenario (Chapter 2, Section 2.2.2). Developing the MVD for bridges is beyond the scope of this thesis but has been done by others. The author leverages the MVD proposed by Sacks et al. (2018), which proposes a binding to the IFC4 Add2 standard for exchanging bridge DTs and defines the mandatory IFC entities in a bridge inspection scenario. Table 5.2 shows part of the essential IFC entities used in this chapter. The author elaborates other entities used in each step in each component model generation.

Information Item	Description	IFC4 Add2
Project	Bridge maintenance/or rehabilitation project	*IfcProject*
Site	A geometrical definition of the bridge site including representative coordinates and a polygon showing the site borders and defining its area. The site definition will include a list of all the bridges (by their names) included in the site.	*IfcSite*
Bridge	A bridge included in the site.	*IfcBuilding*
Slab	This particular element is for the slab part of a slab or beam-slab bridge, i.e., the bridge deck is composed only of a slab without girders.	*IfcSlab*
Pier cap	A transverse beam connecting the main bridge columns at a pier supporting line.	*IfcBeam*
Pier	A bridge upright support element for the bridge superstructure.	*IfcColumn*
Girder	The main bridge deck girder is transferring the deck load to the supports.	*IfcBeam*

Table 5.2 IFC entities concepts used in this thesis

5.2.4. Aggregation

An aggregation indicates an internal unordered part composition relationship between the whole structures, referred to as the “composite”, and the subordinate components, referred to as the “parts”. For instance, in building construction, aggregation can be used on spatial elements to indicate a spatial structure such as a storey within a building. A building storey is the aggregation of the elements in the specified storey. Similarly, for a bridge construction it may or may not be useful to use a storey, so a bridge can be the aggregation of the bridge components, including the slab, abutments, piers, pier caps and so on. Figure 5.2 illustrates the IFC bridge model aggregation used in this chapter. Bridge components are assigned to IFC entities in a hierarchical way by using the aggregation relationship *IfcRelAggregates*.

The following texts elaborate how to use the presented IFC structure to describe a LOD 200 bridge gDT.

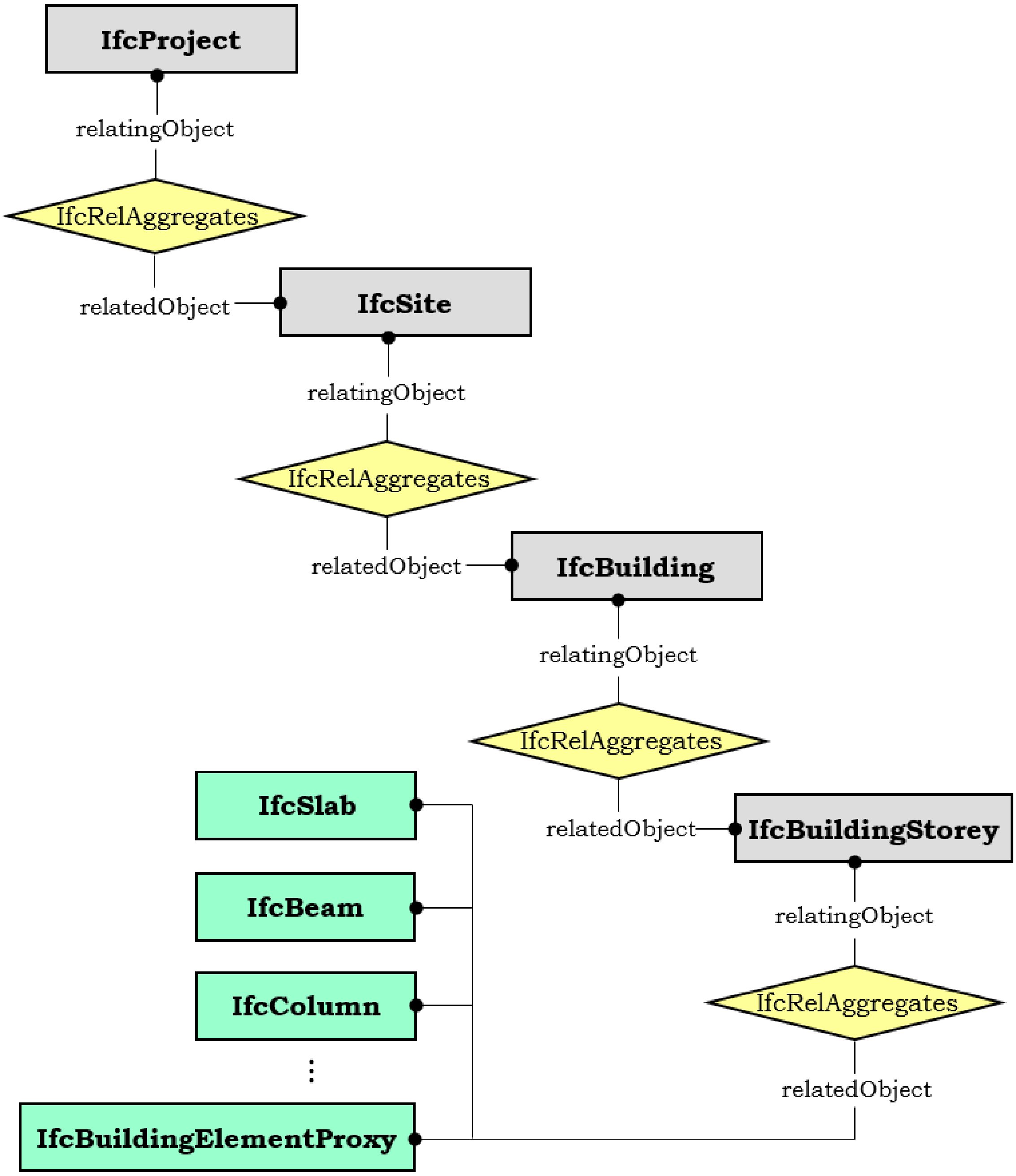


Figure 5.2 IFC Bridge model hierarchy aggregation

5.2.5. LOD 200 gDT generation

The objective of this section is to generate a bridge gDT in LOD 200, meaning that all bridge components are represented as generic placeholders with approximate geometry (Table 5.1).

The author uses TSR (Tessellated Surface Representation) to create the OBB representation for each model element. The reason to choose TSR is because the OBB of a point set is a parallelepiped of 12 edges, 8 vertices and 6 faces. It is not a simple orthogonal box which can be described using the *IfcBoundingBox*. Specifically, *IfcBoundingBox* defines an orthogonal box oriented parallel to the axes of the geometric coordinate system. So, an oriented orthogonal bounding box can be represented using *IfcBoundingBox* and its direction information in IFC, whilst TSR is an explicit and easy way to present an OBB as the vertex information can directly be used. The parallelepiped geometry can be represented using the tessellated geometry model through *IfcShapeRepresentation*, expressing it as a triangulated tessellation. The faces are constructed by implicit polylines defined by three Cartesian points (vertices) constituting half of each face of the OBB. Figure 5.3 shows an example of the *IfcPolyline* model.

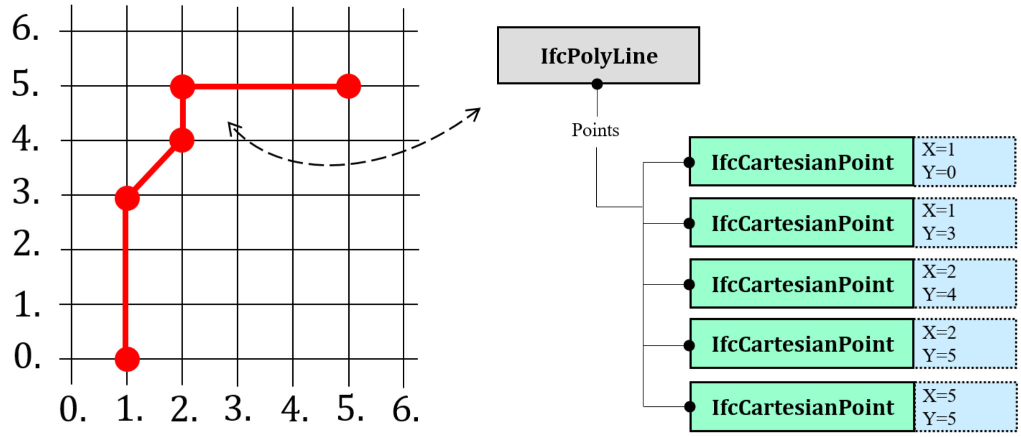
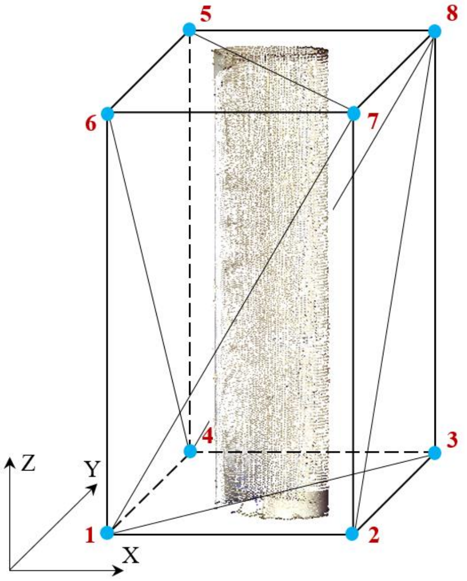


Figure 5.3 Example of *IfcPolyline*

The coordinates of each vertex are provided by an index into an ordered list of Cartesian points *IfcCartesianPointList3D*. For example, as shown in Figure 5.4, ((p1x, p1y, p1z) … (p8x, p8y, p8z)) is the list of the 8 vertex coordinates of an OBB. Then, the author uses *IfcTriangulatedFaceSet* to present the tessellated face set with all faces bounded by the triangles. Since an OBB has 6 faces, each consisting of 2 triangles, there are 6x2=12 items in the *IfcTriangulatedFaceSet.* Figure 5.4 demonstrates a snippet of IFC codes describing an OBB.



```
#100=   IFCCARTESIANPOINTLIST3D
        (((p1x,p1y,p1z), (p2x,p2y, p2z),
        (p3x,p3y,p3z),(p4x,p4y,p4z),
        (p5x,p5y,p5z),(p6x,p6y,p6z),
        (p7x,p7y,p7z),(p8x,p8y,p8z)));
#101=   IFCTRIANGULATEDFACESET(#100,$,
        $,
        ((1,2,3),(1,3,4),
        (5,6,7),(5,7,8),
        (1,4,6),(4,5,6),
        (2,7,8),(2,3,8),
        (1,6,7),(1,2,7),
        (4,5,8),(3,4,8)),$);
#102=   IFCSHAPEREPRESENTATION(#1,'Bod
        y','Tessellation',(#101));
```

Figure 5.4 Triangulated face set geometry of a pier point cluster OBB (left) and snippet of IFC code

The author assigns a specific IFC entity to one corresponding point cluster based on its semantic label. Specifically, *IfcSlab* is used for the Slab, *IfcBeam* for both pier cap and girder point clusters, and *IfcColumn* for pier clusters. IFC places emphasis on property sets. These are sets of properties that are used to define contextual properties, performance, material, and other implicit attributes, which are the basic elements of IFC models. IFC has a flexible extension mechanism that allows custom defined attributes through *IfcPropertyset* without modifying the underlying schema. *IfcPropertyset* is a set of IFC properties which stores the actual data as triple including name, data type, and value. For LOD 200 bridge gDT generation, the author introduces the property set *Pset_BoundingBoxProperties*, in which the method adds the attributes such as the length, width, and height of each OBB and composes them into an *IfcPropertyset.* Figure 5.5 shows a code snippet of the IFC properties of an OBB. This way, each point cluster generated from the previous object detection step is transformed into an OBB in IFC format. The whole bridge gDT consists of a set of OBBs (Figure 5.6), blocks which only provide the location and orientation information of each component, so that the geometric shape is a rough approximation in nature.

```
/****************************************/
/*    Pset_BoundingBoxProperties    */
/****************************************/
#105=    IFCPROPERTYSINGLEVALUE('OBB
         Length:',$,IFCLENGTHMEASURE(78.52),$);
#106=    IFCPROPERTYSINGLEVALUE('OBB
         Width:',$,IFCLENGTHMEASURE(2.80),$);
#107=    IFCPROPERTYSINGLEVALUE('OBB
         Height:',$,IFCLENGTHMEASURE(10.60),$);
#108=    IFCPROPERTYSET('6aa7134da20b403eafee',$,'Pset_BoundingBoxProperties
         ',$,
         (#105,#106,#107));
#109=    IFCRELDEFINESBYPROPERTIES('1e83f1978df846ccb496',$,$,$,(#104),#10
         8);
```

Figure 5.5 Snippet of the IFC code of properties of an OBB

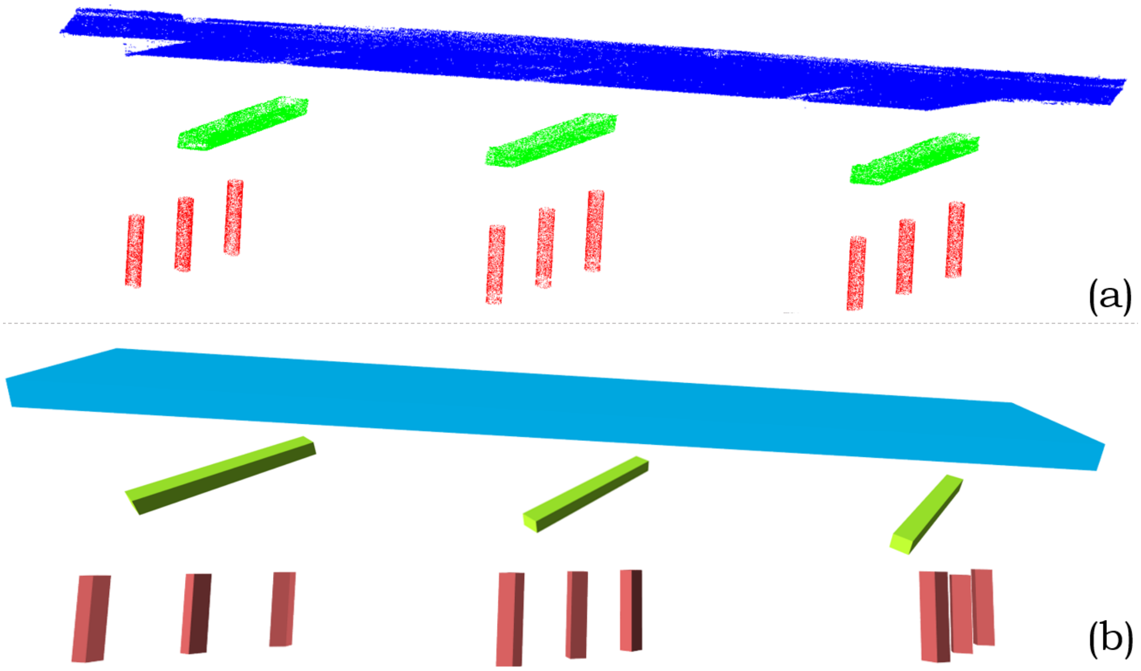


Figure 5.6 (a) Labelled clusters generated from the previous step; (b) OBB of each labelled point cluster in IFC format

5.2.6. LOD 250—300 gDT generation

The objective of this section is to generate a bridge gDT with a resolution higher than LOD 200, but not fully LOD 300. The author uses SSR (Swept Solid Representation or Extrusion) to create the stacked slices for each model element. Extruded objects are often used when one dimension of a component is significantly larger than the other two, or when the cross-section of the component is considered to be constant along the path. Most of the bridge components can be modelled using extruded objects, including the deck slab, girders, piers, and so on. Solid extrusions are preferred wherever possible if the cross-section in each slice model is deemed to be constant. Thus, SSR is mainly used for the model generation in the proposed object fitting method. The general thrust behind the LOD 250—300 representation is that the geometry of a bridge component can be approximated using multiple stacked slices. For example, the horizontal alignment information of the deck slab can be derived from multiple straight alignments of deck segments. In the following text, the author elaborates how to model each of the four types of labelled point cluster.

5.2.6.1. Slab – *IfcSlab*

The topology of a bridge usually depends on its horizontal and vertical alignment, such as the straightness and flatness of the deck. Real-world bridges are neither straight nor flat. To circumvent or be compatible with the existing constraints of road geometry, many highway bridges carrying roads are on a curved alignment and the supporting structure follows that curved alignment (Highways England, 2018c). The alignment information (horizontal and vertical alignment) is an important reference system associated with linear construction such as roads, rails, and bridges. A single alignment may have 1) a horizontal alignment defined in the XY plane of the global coordinate system; 2) an accompanying vertical alignment, defined along the horizontal alignment in the Z coordinate space; and 3) a 3D alignment, computed from the horizontal and vertical alignment.

The presented method aims to approximate the real horizontal (and/or vertical) alignment by using multiple straight segments, such that different horizontal alignment segments can be concatenated to a single horizontal alignment, with the same also true for the vertical alignment. This information can be assigned in the future into the *IfcAlignment* entity as the list of slab segments generated from the proposed method can deduce the necessary information required for *IfcAlignment* (Chapter 7, Section 7.5).

Specifically, the author leverages a similar but not identical slicing method to that proposed in Chapter 4 to slice the deck slab into *J* slices. The slicing does not take a parallel pattern but is rather oriented along the normal direction of the slab curved alignment. According to Kobryń (2017), the author assumes that a circular curve is used for the horizontal alignment of bridges investigated in this research (see A8 and A9, Chapter 3, Section 3.4), such that the general function of the horizontal alignment is a degree 2 parabola (Figure 5.7). This assumption is based on the highway bridge design rule that it is preferable to locate bridges on the tangent positions of the alignment. Large horizontal curves should be avoided on bridges whenever possible. Yet, often, it is necessary to locate a bridge on a curve due to road geometry and on-site constraints. Where a curve is necessary, a simple curve should be used on the bridge and any curvature or super-elevation transitions placed on the approaching roadway (Highways England, 2018b).

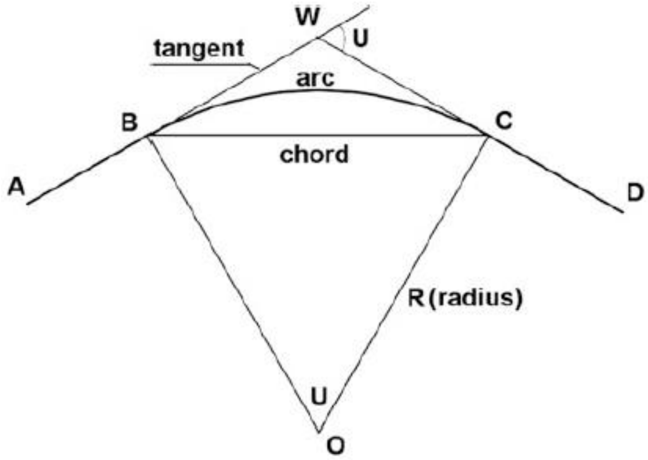


Figure 5.7 Circular curve for horizontal alignment (Kobryń, 2017)

The deck slab point cluster detected from the previous step normally contains most of the scanned points of an entire bridge point cloud, attributed to its large upper and bottom surface being exposed to the laser sensor. First, in order to be cost-efficient, the proposed method does not use all of the slab points, only 10% of them being randomly chosen for fitting a parabola. To this end, the author projects the randomly down-sampled deck slab point cluster onto the XY-plane followed by fitting a unique second-degree polynomial to the projected points by minimizing the square error:

$$E = \sum_{i=0}^{k} |p(x_i) - y_i|^2 \tag{Eq. 5.1}$$

in the system of linear equations:

$$\begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_k \end{bmatrix} = \begin{bmatrix} x_0^n & x_0^{n-1} & x_0^{n-2} & \cdots & x_0 & 1 \\ x_1^n & x_1^{n-1} & x_1^{n-2} & \cdots & x_1 & 1 \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ x_k^n & x_k^{n-1} & x_k^{n-2} & \cdots & x_k & 1 \end{bmatrix} \begin{bmatrix} a_k \\ a_{k-1} \\ \vdots \\ a_0 \end{bmatrix}, \tag{Eq. 5.2}$$

where in matrix notation, the equation for a polynomial fit is given by

$$y = \mathrm{X}a. \tag{Eq. 5.3}$$

This can be solved by premultiplying by the transpose of X^{T}, i.e.

$$\mathrm{X}^{\mathrm{T}}y = \mathrm{X}^{\mathrm{T}}\mathrm{X}a. \tag{Eq. 5.4}$$

The author then yields this system for a_k for a second-degree polynomial to construct the interpolant $p(x)$ by inverting directly the matrix equation in (Eq. 5.4) , i.e.

$$a = (\mathrm{X}^{\mathrm{T}}\mathrm{X})^{-1}\mathrm{X}^{\mathrm{T}}y. \tag{Eq. 5.5}$$

Then, the author acquires the parabola of the horizontal alignment of the deck slab $f(x) = Ax^2 + Bx + C$ with A, B, $C \in \mathbb{R}$, $a \neq 0$.

Next, the author computes the tangent at each interpolant of the parabola (Figure 5.8). The derivative of the parabola gives the slope of the line tangent: $\text{tangent}_j = f(x)' = 2Ax + B$. The normal is given by $\text{normal}_j = \frac{-1}{\text{tangent}_j}$. The deck slab is then segmented along the direction of the normal of each interpolated position into J slices (Figure 5.8).

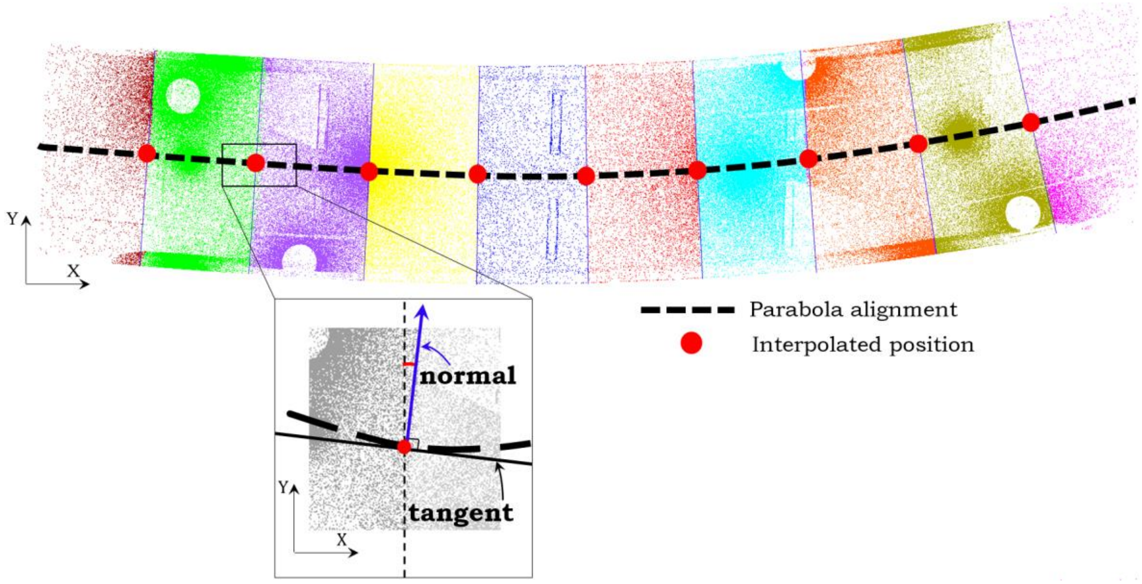


Figure 5.8 Slicing deck slab along the normal of the interpolated positions

Then, the author assumes that each slice is straight along the tangent direction of this slice and that the cross-section of each slice is constant (see A10 and A11, Chapter 3, Section 3.4), despite the possible variance of the cross-section along the central axis of the whole bridge deck slab. This way, the problem of modelling the whole deck slab is transformed into modelling each straight slab slice. The author elaborates this in the following. For each slice, the method first rotates the slice around the Z-axis using:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(-\varphi_j) & \sin(-\varphi_j) & 0 & 0 \\ -\sin(-\varphi_j) & \cos(-\varphi_j) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}, \qquad \text{(Eq. 5.6)}$$

where the rotated angle φ_j is derived from the angle between the normal direction of the alignment of the slice j and the global Y-axis. Specifically, the normal direction of each slice is computed using the mid-x value of each slice j. The updated slice points (x', y', z') are used to define the cross-section of each slice. The author uses a 2D *ConcaveHull* α-shape (Moreira & Santos, 2006) to describe the outline of the slice cross-section. Each concave hull of the local XY-plane projection of the slice j is stored as a 2D Cartesian point *IfcCartesianPoint*. Given a point set S, the α-shape degenerates to S when α approaches 0 while it is the convex hull of point set S when α increases towards infinity. This can be expressed as:

$$\lim_{\alpha\to 0} S_\alpha = S \text{ and } \lim_{\alpha\to\infty} S_\alpha = \mathrm{conv}S. \qquad \text{(Eq. 5.7)}$$

Using an appropriate value of α between these limits creates a shape that is not necessarily convex, and may contain many holes, but broadly resembles the "shape" implied by the distribution of the set of points. These *IfcCartesianPoint* elements map the cross-section with a list of *IfcPolyline* objects (Figure 5.9). A 2D profile *IfcArbitraryClosedProfileDef* is therefore used to describe the slice cross-section. The slab slice geometry is then represented using an extruded geometry model through *IfcExtrudedAreaSolid* and *IfcShapeRepresentation*, expressing it as a Swept Solid.

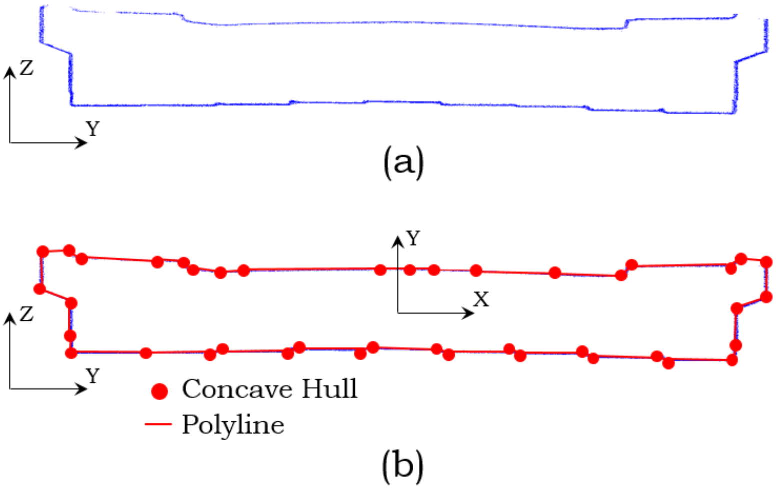


Figure 5.9 (a) Cross-section of a deck slab slice *j*; (b) concave hulls of the local XY-plane of slice *j*

The extruded area solid defines the extrusion of a 2D area (given by a profile definition) by two attributes. One is the Extruded Direction, defining the direction in which the profile is to be swept; the other one is the Depth, defining the distance over which the profile is to be swept (Figure 5.10). The extruded direction is derived from the tangent direction at the mid-x value position of each slice. The depth is derived from the maximum and minimum x'-coordinates of each oriented slab slice.

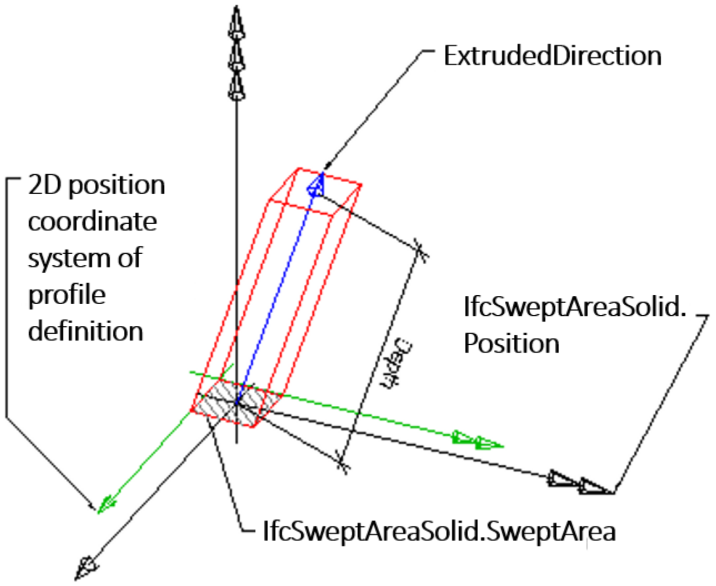

ExtrudedDirection
2D position coordinate system of profile definition
IfcSweptAreaSolid. Position
Depth
IfcSweptAreaSolid.SweptArea

Figure 5.10 An extruded area solid defined by direction and depth of the 2D area extrusion (buildingSMART, 2016b)

It is worth noting that, the extruded direction might be ignored if the slice number is adequate. This concept comes from Cavalieri's principle, which serves as the theoretical guidance of the proposed method in this research. Assume an arbitrary 3D solid along the X-axis, extending from $x = a$ to $x = b$, as shown in the Figure 5.11. Let $A(x)$ give the area of a cross-section of the slide perpendicular to the X-axis at x, so if the solid is sliced into thin parallel slices, each of width Δx, then the approximate volume of each slice equals $A(x) \cdot \Delta x$. With thinner slices, the approximation becomes more precise. Suppose the solid is divided into n equally thick slices and define the usual partition: $\Delta x = \frac{b-a}{n}$, and $x_i = a + i\Delta x$, for $i = 1,2, \dots, n$, then Cavalieri's principle states that:

$$V = \lim_{n\to\infty} \sum_{i=1}^{n} A(x_i)\Delta x = \int_a^b A(x)dx \qquad \text{(Eq. 5.8)}$$

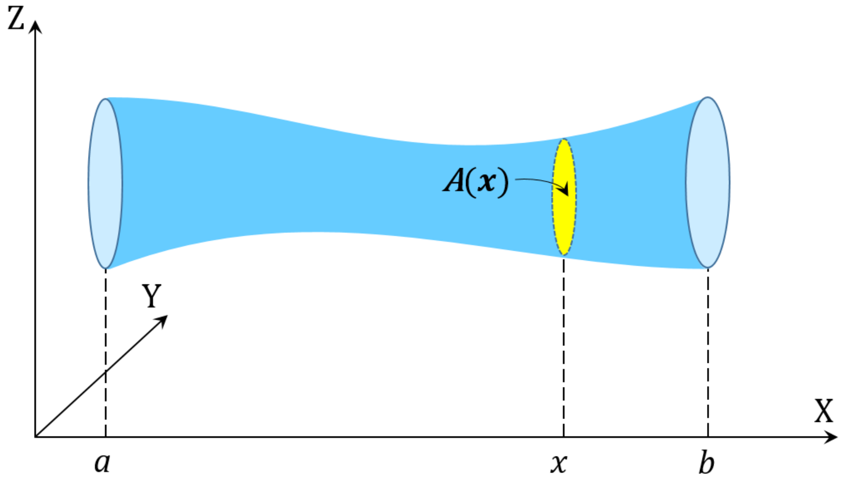


Figure 5.11 Arbitrary 3D solid the along X-axis with a cross-section $A(x)$

Similarly, using the slicing strategy, for instance, given a deck slab interval $[a, b]$, the volume of the deck slab of a bridge roughly aligned along the X-axis can be expressed by Cavalieri's formula as $V = \int_a^b A(x)dx$, using its cross-section and the slice thickness (Figure 5.12).

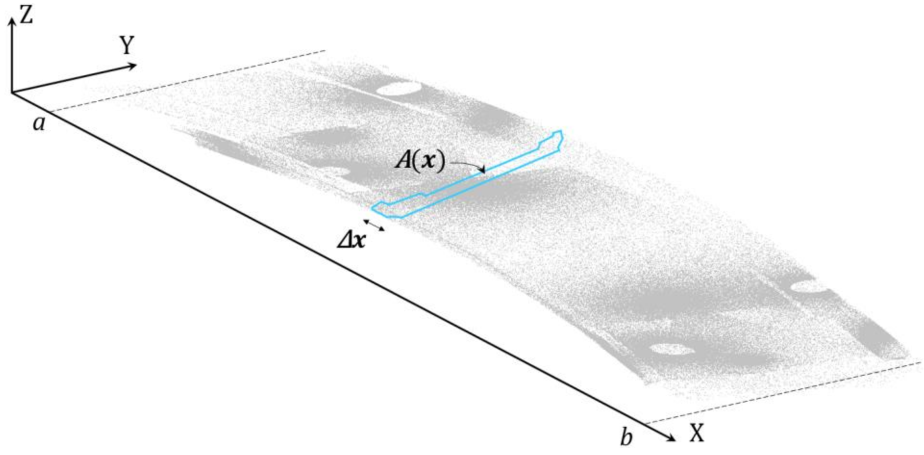


Figure 5.12 Deck slab along the X-axis with a cross-section $A(x)$

Figure 5.13 shows an example of a snippet of the IFC codes of a slab slice. The cross-section of this slab slice consists of 92 concave hulls, which are connected by 93 polylines. For LOD 250—300 bridge gDT generation, the author introduces the property set *Pset_SlabSliceProperties*, in which the method can add the attributes of each slab slice. Normally, a gDT object should include a "shape" property completed with a float alphanumeric value representing the characteristic of the object shape. If the value is not known or not available, the property is defined as “N/A”. The author has not specified any attribute with specific numbers (e.g. Figure 5.13 #109--#111) but demonstrates the possibility of flexibly adding any attribute for further use and composing them into an *IfcPropertyset* (e.g. Figure 5.13 #112). The attributes could be the area of the extruded section, the volume of the slice, and so on, which are required by the end-users for structural analysis or other purposes in an inspection process. Figure 5.14 illustrates an example of deck slab slice models of a bridge used in this research (number of slices = 20).

```
/*************************************/
/*              Slab 1               */
/*************************************/
#100001=    IFCCARTESIANPOINT((-1655.0,269561.6));
…
#100090=    IFCCARTESIANPOINT((-390.7,269624.0));
#100091=    IFCCARTESIANPOINT((-1243.2,269571.9));
#100092=    IFCCARTESIANPOINT((-1431.6,269563.1));
#101=       IFCPOLYLINE((#100001,#100002,#100003,#100004,#100005,#100006
            ,#100007,#100008,#100009,#100010,#100011,#100012,#100013,#10
            0014,#100015,#100016,#100017,#100018,#100019,#100020,#100021
            ,#100022,#100023,#100024,#100025,#100026,#100027,#100028,#10
            0029,#100030,#100031,#100032,#100033,#100034,#100035,#100036
            ,#100037,#100038,#100039,#100040,#100041,#100042,#100043,#10
            0044,#100045,#100046,#100047,#100048,#100049,#100050,#100051
            ,#100052,#100053,#100054,#100055,#100056,#100057,#100058,#10
            0059,#100060,#100061,#100062,#100063,#100064,#100065,#100066
            ,#100067,#100068,#100069,#100070,#100071,#100072,#100073,#10
            0074,#100075,#100076,#100077,#100078,#100079,#100080,#100081
            ,#100082,#100083,#100084,#100085,#100086,#100087,#100088,#10
            0089,#100090,#100091,#100092,#100001));
#102=       IFCARBITRARYCLOSEDPROFILEDEF(.AREA.,'deckSlab',#101);
#103=       IFCCARTESIANPOINT((37646.700000000004,0.,0.));
#104=       IFCSLAB('7IfdS9ZAQku4vN074Zp8',$,'deckSlab',$,'deckSlab',$,#107,'de
            ckSlab',$);
#105=       IFCEXTRUDEDAREASOLID(#102,#108,#114,3904.3);
#106=       IFCSHAPEREPRESENTATION(#1,'Body','SweptSolid',(#105));
#107=       IFCPRODUCTDEFINITIONSHAPE($,$,(#106));
#108=       IFCAXIS2PLACEMENT3D(#103,#2,#3);
#109=       IFCPROPERTYSINGLEVALUE('Property A:',$,IFCIDENTIFIER('N/A'),$);
#110=       IFCPROPERTYSINGLEVALUE('Property B:',$,IFCIDENTIFIER('N/A'),$);
#111=       IFCPROPERTYSINGLEVALUE('Property C:',$,IFCIDENTIFIER('N/A'),$);
#112=       IFCPROPERTYSET('Q4aFfLsjxKvYYYQNpxfR',$,'Pset_SlabSlicePropertie
            s',$,(#109, #110, #111));
#113=       IFCRELDEFINESBYPROPERTIES('IzbZOghGrptNbGu3FayF',$,$,$,(#10
            4),#112);
#114=       IFCDIRECTION((-0.02355817626597581,0.,1.));
```

Figure 5.13 Snippet of the IFC codes of a slab slice

As shown in (Figure 5.14 (c)), the deck slab model contains undulations. This is attributed to the nature of the *ConcaveHull* representation. Convex hulls are the outmost vertices whereas the concave hulls are a list of the boundary vertices of a set of points, which capture all the subtle changes in the slab surfaces. This excessive LOD representation may not be necessary because the carriageways are expected to be kept as smooth as possible so that bridge users should not perceive obvious surface undulations. The end-users of the resulting gDTs may not prefer these details either. However, at the same time, existing potholes and any big defects should not be missed. It is a trade-off between the reality and the digital representation, which could be possible to control using a user-defined smooth technique but is beyond the scope of this research due to limited time.

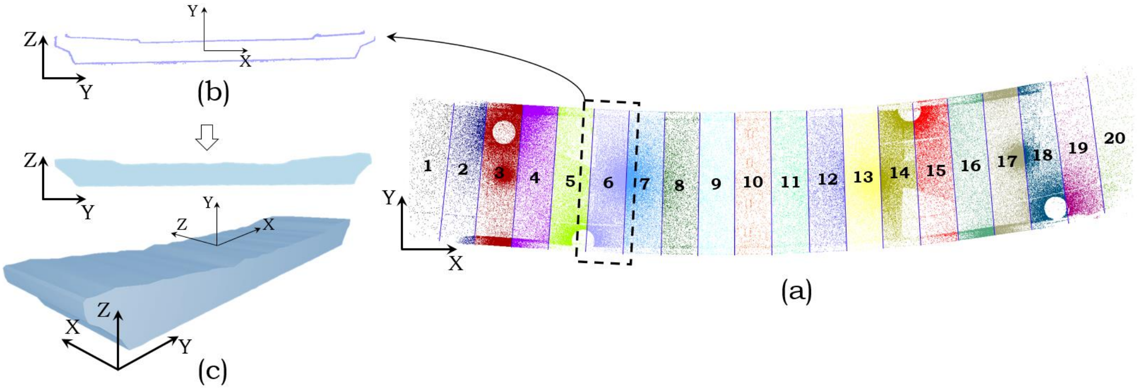


Figure 5.14 Slab slice modelling (a) slice the slab point cluster into 20 slices; (b) the cross-section of slice 6; (c) IfcSlab of slice 6

Last, it is worth noting that this method does not maintain the tangential continuity between the slice models. This is to say, a horizontal alignment is assumed to be gap free. However, the proposed approximation method provides the possibility for achieving gap-less segments in the future which will keep the tangential continuity of the horizontal alignment, because the starting point of a subsequent segment can be matched to the end point of the previous segment. This way, the connectivity between continuous segments could be tangential and can be mapped to *IfcAlignment* (Chapter 7, Section 7.5).

5.2.6.2. Pier cap – *IfcBeam*

Similar to how the slab slice is extruded, when modelling a pier cap point cluster, the author projects its points onto the XY-plane. Then, the author uses a 2D *ConcaveHull* shape (Eq. 5.4) to describe the projected contour such that each concave hull of the local XY-plane projection of the pier cap is stored in a 2D Cartesian point *IfcCartesianPoint* followed by mapping the contour with a list of *IfcPolyLine* objects (Figure 5.15). Like the slab slice, a pier cap is also represented as a Swept Solid through *IfcArbitraryClosedProfileDef* and *IfcExtrudedAreaSolid*.

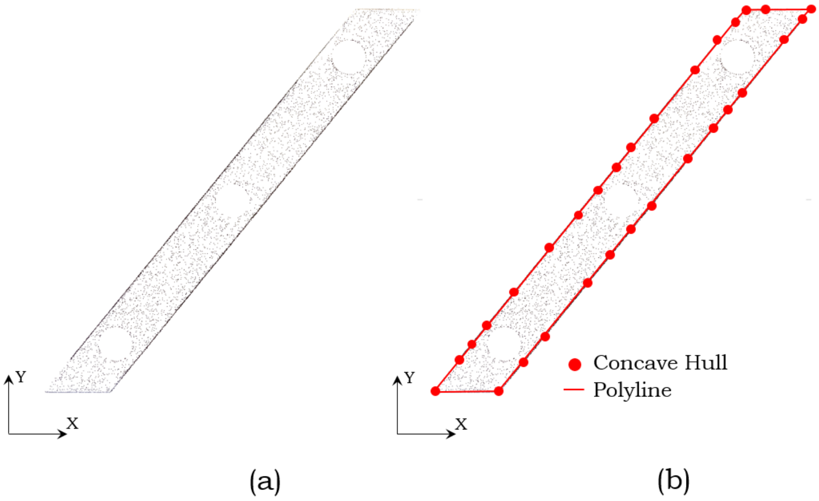


Figure 5.15 (a) XY-plane projection of a pier cap point cluster; (b) concave hulls of the projected points

As shown in Figure 5.10, the extruded direction is assumed to be vertical for pier caps and the depth is defined as the height of the pier cap (see A12, Chapter 3, Section 3.4), which is calculated using the maximum and minimum of its z-coordinates. Likewise, the author introduces the property set *Pset_PierCapProperties,* for which the method does not specify but can flexibly add attributes for future use. Figure 5.16 shows an example of a snippet of the IFC codes of a pier cap. The contour of the projected XY-plane points consists of 17 concave hulls, which are connected by 18 polylines.

```
/************************************/
/*             PierCap 17             */
/************************************/
#1700001=    IFCCARTESIANPOINT((37715.199,-4361.1));
…
#1700015=    IFCCARTESIANPOINT((38047.401,-4305.3));
#1700016=    IFCCARTESIANPOINT((38001.301,-4355.8));
#1700017=    IFCCARTESIANPOINT((37910.9,-4359.7));
#1701=       IFCPOLYLINE((#1700001,#1700002,#1700003,#1700004,#1700005,#
             1700006,#1700007,#1700008,#1700009,#1700010,#1700011,#1700
             012,#1700013,#1700014,#1700015,#1700016,#1700017,#1700001));
#1702=       IFCARBITRARYCLOSEDPROFILEDEF(.AREA.,'pierCap',#1701);
#1703=       IFCCARTESIANPOINT((0.,0.,268577.0));
#1704=       IFCBEAM('gtVtUBuk6Oquo7onZ870',$,'pierCap',$,'pierCap',$,#1707,'
             pierCap',$);
#1705=       IFCEXTRUDEDAREASOLID(#1702,#1708,#4,993.4);
#1706=       IFCSHAPEREPRESENTATION(#1,'Body','SweptSolid',(#1705));
#1707=       IFCPRODUCTDEFINITIONSHAPE($,$,(#1706));
#1708=       IFCAXIS2PLACEMENT3D(#1703,#4,$);
#1709=       IFCPROPERTYSINGLEVALUE('Property A:',$,IFCIDENTIFIER('N/A'),$);
#1710=       IFCPROPERTYSINGLEVALUE('Property B:',$,IFCIDENTIFIER('N/A'),$);
#1711=       IFCPROPERTYSINGLEVALUE('Property C:',$,IFCIDENTIFIER('N/A'),$);
#1712=       IFCPROPERTYSET('jSLgcUGBp89gePkkhcAz',$,'Pset_PierCapPropertie
             s',$,(#1709,#1710,#1711));
#1713=       IFCRELDEFINESBYPROPERTIES('KbnwJsHxsmauW6a8cBjs',$,$,$,(#
             1704),#1712);
```

Figure 5.16 Snippet of the IFC codes of a pier cap

5.2.6.3. Pier – *IfcColumn*

Piers support the weight of a bridge against gravity and serve as retaining walls to resist lateral movement. Defining a generic parametric pier object is difficult because piers can take many configurations. In general, the shape of a pier is defined by the shape of its cross-section, whose scale may vary over its height. Figure 5.17 illustrates a collection of the most typical cross-section shapes of piers for modern highway bridges, taken from the Bridge Engineering Handbook – Substructure Design (Wai-Fah & Lian, 2014). However, in reality, piers can take many other irregular shapes.

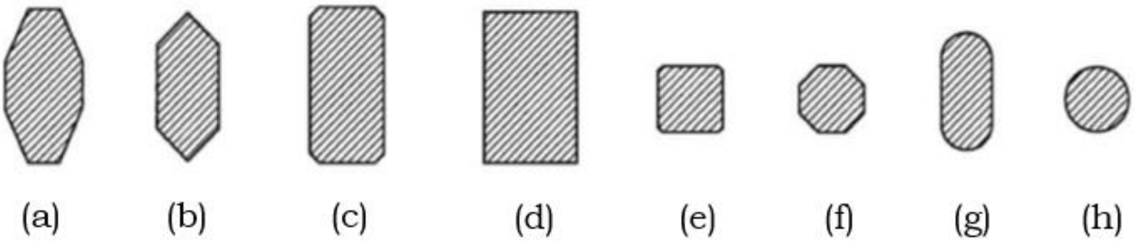


Figure 5.17 Typical cross-section shapes of piers for overcrossings or viaducts on land (Wai-Fah & Lian, 2014)

To simplify the problem, the author groups the cross-section of pier shapes listed in Figure 5.17 into 3 classes of primitives: circular (for cylindrical piers), quadrilateral (for quasi-cuboid or quasi-trapezoidal prism piers), and the others. This research does not tackle the “terminator” shape (Figure 5.17 (g)). Thus, the 3 groups are

- *Shape group* 1 – Circular (Figure 5.17 (h));
- *Shape group* 2 – Quadrilateral (Figure 5.17 (d));
- *Shape group* 3 – Other shapes: the rest, i.e. Figure 5.17 (a), (b), (c), (e), and (f).

The chamfered-edge shapes (Figure 5.17 (c), (e), and (f)) are considered as “other” shapes. A shape detection method is needed to identify the cross-section shape of piers. The shape detection method should be (1) invariant under scaling, rotation, and translation, and (2) robust under geometrical distortions and occlusions.

The author conducted an initial try using the Ramer-Douglas-Peucker contour approximation method (Douglas & Peucker, 2011). As the name suggests, contour approximation is an algorithm for reducing the number of points in a curve with a reduced set of points. This algorithm is commonly known as the split-and-merge algorithm. Contour approximation is predicated on the assumption that a curve can be approximated by a series of short line segments. This leads to a resulting approximated curve that consists of a subset of points that were defined by the original curve. The shape is determined using the number of the contour vertices. For example, if the approximated contour has three vertices, then it must be a triangle. If a contour has four vertices, then it must be either a square or a rectangle; if a contour has five vertices, the shape is labelled as a pentagon, and so on. Obviously, this method assumes in the first place that the investigated shapes are basic shapes which can be recognized using contour properties. Second, this method assumes that the contour can be cleanly segmented from the background. Both assumptions are unrealistic because (1) one expects to describe the real geometries rather than generic shapes which means that a custom object detector is needed for more advanced and as-weathered shapes, or shapes that have substantial variances in how they appear; and (2) real PCD is very noisy. This method is likely to fail if one cannot smooth the jagged edges resulted after projecting the piers onto 2D planes. Figure 5.18 illustrates examples of shape detection results using the split-and-merge contour approximation method. As shown, all shapes are recognized as one generic 2D shape and not all of them are correctly labelled (e.g. the parallel quadrilateral pier cap was wrongly identified as a circle). This demonstrates that this contour approximation method is very sensitive to the jagged outline, which can be attributed to the finite size of a square pixel when a high contrast edge appears. An alternative way is needed which can better describe the cross-section shapes.

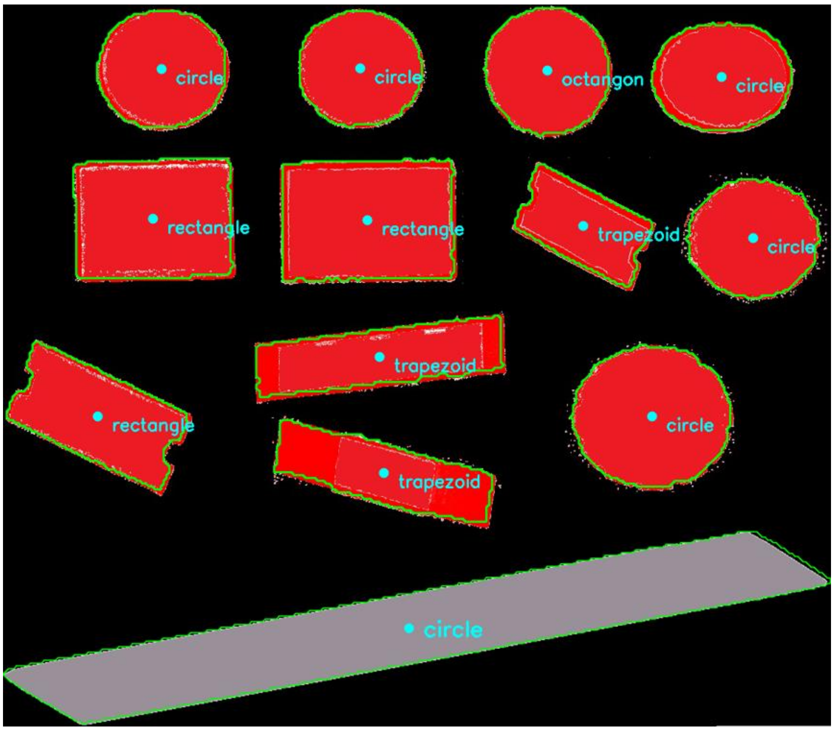


Figure 5.18 Shape detection using Ramer-Douglas-Peucker contour approximation

The author then uses another simple but very efficient fuzzy-logic-based shape descriptor, which is similar to the method presented by Fonseca & Jorge (2000), to extract geometric features followed by identifying the cross-section shape of a pier. The reason for using fuzzy-logic is to remedy the jagged edges resulting from the point-cloud-based projection because the boundaries are not smooth due to inevitable noise, which renders edge-based shape detectors unreliable. Unlike simplified scenarios and synthetic data, real objects embedded in point clouds are similar to hand-drawn geometric shapes that usually contain imperfections or distortions. These properties increase the difficulty of shape detection for real objects underlying in PCD. The presented fuzzy-logic-based method can handle imprecision and ambiguity in imperfect point cloud projection in a natural manner, thereby recognizing cross-section shapes independently of noise, edge effect, size, unevenly distributed points, and occlusions. The author elaborates this method in the following.

The fuzzy-logic pier cross-section shape detection method is based on two main ideas:

- It follows a top-down strategy to identify shapes, using the global geometric properties extracted from the projection contour of a pier point cluster;
- It leverages a set of filters to specify shapes through distinctive criteria.

Piers are not necessarily always perfectly vertical. They may be slightly inclined. First, the author projects a pier point cluster points onto the global XY-plane followed by calculating the perimeter of the projected points (denoted P_{ch}) and the bounded area (denoted A_{ch}) using their concave hulls. Second, the author computes the area of the enclosing rectangle of the concave hulls, i.e., the 2D oriented-bounding-box (denoted A_{er}) and the area of their largest-quadrilateral (denoted A_{lq}). Figure 5.19 (a) and (b) illustrate examples of a cylindrical pier and a trapezoidal prism pier, respectively. As shown in the figures, the cross-section of a cylinder is close to a circle while the cross-section of a trapezoidal prism pier is close to a rectangle. The figures also show that if the cross-section is detected as a circle, then the perimeter of the concave hulls P_{ch} (Figure 5.19 (a.3)), the enclosing rectangle A_{er} (Figure 5.19 (a.4)), and the largest quadrilateral A_{lq} (Figure 5.19 (a.5)) are distinctly different from each other, whereas if the cross-section is a quadrilateral, these three geometric features are similar to each other (Figure 5.19 (b)).

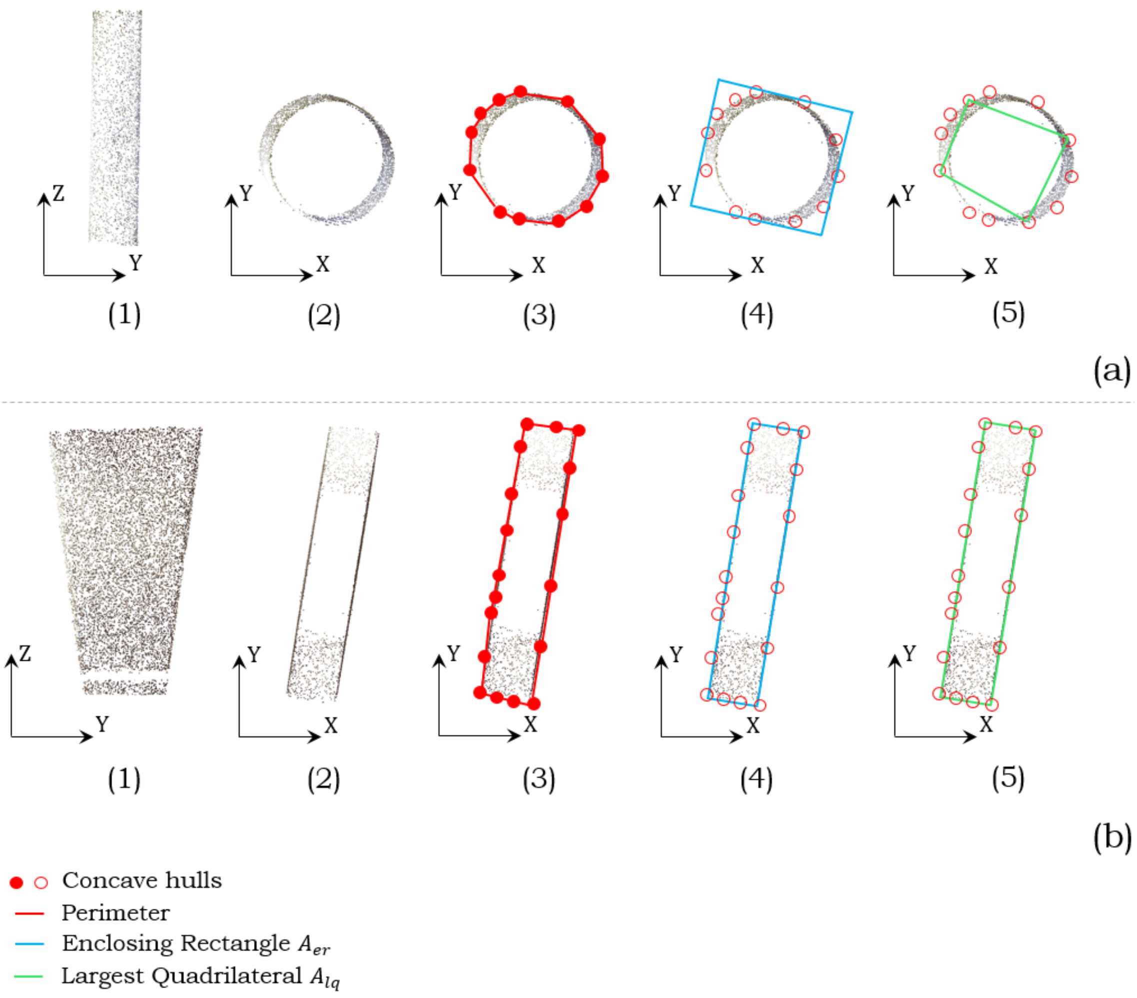


Figure 5.19 (1) YZ-plane projection; (2) XY-plane projection; (3) concave hulls of XY-plane projected points; (4) enclosing rectangle of concave hulls; (5) largest quadrilateral of concave hulls

Specifically, define the thinness ratio as ${P_{ch}}^2/A_{ch}$:

if ${P_{ch}}^2/A_{ch} \cong 4\pi$, (Eq. 5.9)

then, the cross-section ← circle,

where ${P_{ch}}^2$ is the squared perimeter of the concave hulls. The thinness of a circle is minimal, since it is the planar figure with the smallest perimeter enclosing a given area, yielding a value around 4π. Next:

else if $A_{ch}/A_{er} \cong A_{lq}/A_{er} \cong 1$, (Eq. 5.10)

then, the cross-section ← rectangle.

That is to say, for quadrilateral cross-section shapes, the area of the concave hulls is very close to that of the enclosing rectangle, and the latter is very close to the largest quadrilateral. The ratios of A_{ch}/A_{er} and A_{lq}/A_{er} are close to unity. Otherwise, the cross-section takes another shape. Specifically, the author computes A_{lq} in the following way.

To find the maximum quadrilateral area inside the polygon consisting of the concave hulls of the projected pier points, the author lists all possible combinations of 4 points making up a quadrilateral from the concave hull. For each combination, the author first computes the 4 angles in a clockwise order using coordinates of 3 points. For example, as shown in Figure 5.20, angle α can be calculated using point D, point A, and point B. Then, the author uses Bretschneider's formula (Eq. 5.8) to calculate the area of a quadrilateral:

$$A_q = \sqrt{(p-a)(p-b)(p-c)(p-d) - abcd \cdot \cos^2(\tfrac{\alpha+\gamma}{2})}\ , \tag{Eq. 5.11}$$

where p is the semi-perimeter. This formula can also be expressed as:

$$A_q = S_{\Delta ADB} + S_{\Delta BDC} = \frac{ad\sin\alpha}{2} + \frac{bc\sin\gamma}{2}\ . \tag{Eq. 5.12}$$

The largest quadrilateral A_{lq} is the maximum value of A_q found.

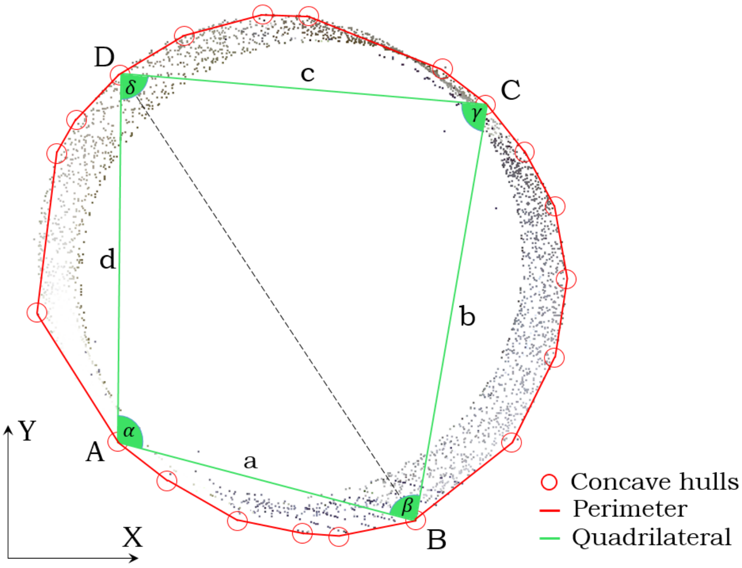


Figure 5.20 A quadrilateral inside concave hulls of the projected points of a cylindrical pier

This fuzzy-logic method classifies the pier cross-section shapes into three shape groups as previously mentioned, namely: *Shape group* 1 – Circular; *Shape group* 2 – Quadrilateral; and *Shape group* 3 – Other shapes. The goals are (1) to define a component using minimum geometric information, and (2) to represent this geometric information in IFC format. For a shape which is identified as *group 1* shape, i.e. circular, the method then describes the pier using a relatively small number of parameters. Else, the method conducts a slicing procedure followed by using 2D *ConcaveHull* to describe the cross-section. The author elaborates the steps to model these classified shapes into 3D *IfcObjects* in the following text.

- **Cylindrical pier**

If a pier cross-section shape is identified as a circle from the fuzzy-logic method, it is a cylindrical pier, and a 2D *IfcCircleProfileDef* is used to describe the circle. To define a cylinder in a 3D space, a minimum of three parameters are necessary: radius (or diameter), location (or position), and direction. There are different ways to detect these parameters for a cylinder, such as the conventionally used RANSAC algorithm and HT. To keep consistent and efficient, the author leverages a practical slicing method. A slicing procedure is conducted along the Z-axis with the slice number set at 20. Multiple slices are used to approximate the global topology of a cylindrical pier. For each slice, the author calculates its radius (or diameter) and ignores the inclination. Then, the radius of the entire cylinder is calculated by averaging the radii obtained from the multiple slices. The averaged radius value is then stored in *IfcCircleProfileDef* as an attribute Radius. To generate an *IfcColumn* instance, two other parameters are needed: location and direction. An *IfcAxis2Placement3D* is used to define a location point and the orientation. The coordinates of the location point are stored in a 3D Cartesian point *IfcCartesianPoint* as an attribute Position. The pier direction information in the 3D coordinates system is stored in *IfcDirection*. The *IfcCartesianPoint* is represented by the bottom slice centre coordinates of the cylinder, i.e. point A (x_A, y_A, z_A) in Figure 5.21. The *IfcDirection* is defined by the vector computed by the bottom and upper slice centre of the cylinder, i.e. point A (x_A, y_A, z_A) and point B (x_B, y_B, z_B):

$$\overrightarrow{AB} = (x_B - x_A, y_B - y_A, z_B - z_A), \qquad \text{(Eq. 5.13)}$$

$$\left|\overrightarrow{AB}\right| = \sqrt{(x_B - x_A)^2 + (y_B - y_A)^2 + \ (z_B - z_A)^2}\,, \qquad \text{(Eq. 5.14)}$$

then the unity direction vector is derived by

$$\frac{\overrightarrow{AB}}{\left|\overrightarrow{AB}\right|} = \left(\frac{x_B - x_A}{\left|\overrightarrow{AB}\right|}, \frac{y_B - y_A}{\left|\overrightarrow{AB}\right|}, \frac{z_B - z_A}{\left|\overrightarrow{AB}\right|}\right). \qquad \text{(Eq. 5.15)}$$

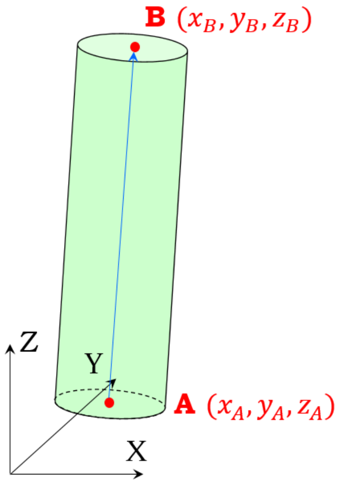


Figure 5.21 The direction of the cylinder is defined by point A and point B

Similar to the deck slab and pier cap, the geometry of a cylindrical pier is represented using the extruded model through *IfcExtrudedAreaSolid* and *IfcShapeRepresentation*, expressing it as a Swept Solid along its extruded direction *IfcDirection* (Eq. 5.15). The author introduces the property set *Pset_CylinderProperties*, in which four attributes are defined: Position, Direction, Diameter, and Length (Eq. 5.14). Again, other attributes can be added flexibly for future use. The method then composes them into an *IfcPropertyset*. Figure 5.22 shows a snippet of the IFC codes of a cylindrical pier.

```
/***********************************/
/*            Cylinder 19          */
/***********************************/
#1900=    IFCCIRCLEPROFILEDEF(.AREA.,'pier',$,397.70000000000005);
#1901=    IFCCARTESIANPOINT((18954.050000000003,-
          2968.4999999999995,264760.2));
#1902=    IFCDIRECTION((0.016032865740065953,-
          0.011407500907438025,0.9998063893270576));
#1903=    IFCAXIS2PLACEMENT3D(#1901,#1902,$);
#1904=    IFCCOLUMN('JKAMd3zbiOhZAjYECZ1K',$,'pier',$,'pier',$,#1906,'pier',.C
          OLUMN.);
#1905=    IFCSHAPEREPRESENTATION(#1,'Body','SweptSolid',(#1907));
#1906=    IFCPRODUCTDEFINITIONSHAPE($,$,(#1905));
#1907=    IFCEXTRUDEDAREASOLID(#1900,#1903,#4,3848.3450806807855);
#1908=    IFCPROPERTYSINGLEVALUE('Position:',$,IFCIDENTIFIER('18.95, -2.97,
          264.76'),$);
#1909=    IFCPROPERTYSINGLEVALUE('Direction:',$,IFCIDENTIFIER('0.016, -
          0.0114, 0.9998'),$);
#1910=    IFCPROPERTYSINGLEVALUE('Diameter:',$,IFCIDENTIFIER('0.8'),$);
#1911=    IFCPROPERTYSINGLEVALUE('Length:',$,IFCIDENTIFIER('3.85'),$);
#1912=    IFCPROPERTYSET('6pl77kdcIvgUBvcxllWC',$,'Pset_CylinderProperties',$
          ,(#1908,#1909,#1910,#1911));
#1913=    IFCRELDEFINESBYPROPERTIES('LtGdrB7m6PPc7uUC7eYV',$,$,$,(#190
          4),#1912);
```

Figure 5.22 Snippet of the IFC codes of a cylinder

- **Quadrilateral and other piers**

The modelling strategy is different if a pier cross-section shape is identified as a quadrilateral or other shape. A quadrilateral cross-section may represent a quasi-cuboid pier or a quasi-trapezoidal prism pier. The author uses a stacked representation to approximate the overall pier shape along its Z-axis through multiple slice models. For each slice, the author applies the same method used for modelling the pier cap. Each slice of a quadrilateral-cross-section pier is considered to be a pier cap, so that again a 2D *ConcaveHull* shape is used to describe the cross-section outline of the pier slice. Figure 5.23 shows an example of a stacked representation of a pier using *IfcArbitraryClosedProfileDef* and *IfcExtrudedAreaSolid* through the *ConcaveHull* shape in each slice. Figure 5.23 (a) demonstrates that the pier consists of multiple extruded slice models. Figure 5.23 (b) shows the cross-section of one slice. Figure 5.24 illustrates stacked piers in an entire bridge gDT. Figure 5.25 shows a snippet of IFC codes of a pier slice.

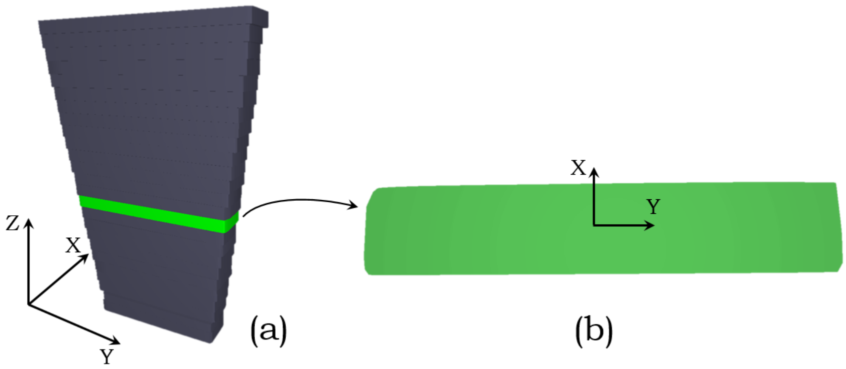


Figure 5.23 A stacked representation of a pier using multiple slice models

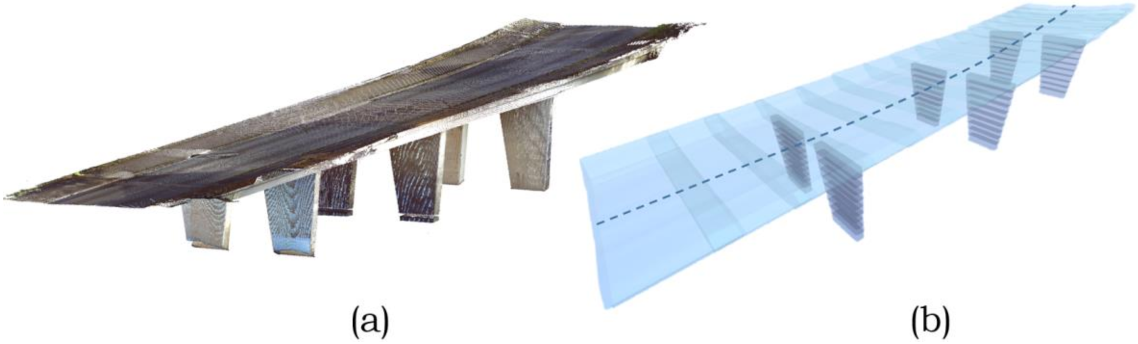


Figure 5.24 Modelling a curved slab bridge (a) point cloud data; (b) stacked representation

```
/***********************************/
/*            PierSlice 31          */
/***********************************/
#3100001=   IFCCARTESIANPOINT((-55679.82,15741.73));
…
#3100011=   IFCCARTESIANPOINT((-55170.78,13220.57));
#3100012=   IFCCARTESIANPOINT((-55181.75,13205.47));
#3101=      IFCPOLYLINE((#3100001,#3100002,#3100003,#3100004,#3100005,#
            3100006,#3100007,#3100008,#3100009,#3100010,#3100011,#3100
            012,#3100001));
#3102=      IFCARBITRARYCLOSEDPROFILEDEF(.AREA.,'pier',#3101);
#3103=      IFCCARTESIANPOINT((0.,0.,150149.10));
#3108=      IFCAXIS2PLACEMENT3D(#3103,#4,$);
#3105=      IFCEXTRUDEDAREASOLID(#3102,#3108,#4,269.41);
#3106=      IFCSHAPEREPRESENTATION(#1,'Body','SweptSolid',(#3105));
#3107=      IFCPRODUCTDEFINITIONSHAPE($,$,(#3106));
#3104=      IFCCOLUMN('4c544bd436a04908906c',$,'pier',$,'pier',$,#3107,'pier',$
            );
#3109=      IFCPROPERTYSINGLEVALUE('Property A:',$,IFCIDENTIFIER('N/A'),$);
#3110=      IFCPROPERTYSINGLEVALUE('Property B:',$,IFCIDENTIFIER('N/A'),$);
#3111=      IFCPROPERTYSINGLEVALUE('Property C:',$,IFCIDENTIFIER('N/A'),$);
#3112=      IFCPROPERTYSET('b8ff2dafd88346768c43',$,'Pset_PierSlicePropertie
            s',$,(#3109,#3110,#3111));
#3113=      IFCRELDEFINESBYPROPERTIES('b35e7fdfa72a484d84e5',$,$,$,(#31
            04),#3112);
```

Figure 5.25 Snippet of the IFC codes of a pier slice

5.2.6.4. Girder – *IfcBeam*

In this section, the author aims to generate the girder with its actual type, which is defined by its profile. Although the majority of bridge construction work is carried out on-site, certain structural component types such as precast beams and precast piers are very routinely used in the US and the UK (Calavera et al., 2004). In addition, 86% of beam-slab bridges to be built in the near future in the UK select precast concrete components for the primary structural elements (Kim et al., 2016). The author therefore assumes that the girders studied in this research are precast, standardized bridge beams (see A15, Chapter 3, Section 3.4). Precast bridge girders are a type of pre-stressed beams capable of spanning long distances, with maximum spans up to 50 m (depending on the load). They can be used for purposes such as fly over bridges on motor ways, and pedestrian bridges, among many others.

The author suggests a template matching method to find the best-match girder type in existing precast bridge beam catalogues. The author leverages the girder sections provided by the standard products of American Association of State Highway and Transportation Officials (AASHTO) and the Bridge Beam Manual provided by BANAGHER Precast Concrete (BANAGHER, 2018), which is the largest precast concrete Bridge Beam manufacturer in Ireland and the UK. In the previous chapter, the author presented an object detection method to detect girders in the deck assembly segment (Chapter 4, Section 4.2.5). The specific girder type in each span is inferred using three criteria:

- Span length sl
- Girder bottom flange b_f
- Web depth d

The span length sl can narrow down a possible range of girder types. This is because often, the creation of a typical girder section begins with the calculation of the structure depth for a given span length (AASHTO, 2017). Appendix E lists minimum structural depth for various structural spans. Then, the girder bottom flange b_f is the averaged flange of the segmented girders in each span derived from the method presented in Chapter 4. Likewise, the web depth d is also deduced from the method presented in Chapter 4. It is extracted along the best projection view. b_f and d can be used to select a specific girder type from the possible girder types. This procedure is iterated until all spans of a bridges are checked. Figure 5.26 illustrates an example of girder type determination using the abovementioned criteria, where sl_3 ≈28m, $\overline{b_f}$ ≈760mm, d ≈1600mm (Figure 5.26 (a)). The closest precast girder type found in the Bridge Beam Manual (BANAGHER, 2018) is type SY2 from SY Beams (Figure 5.26 (b)).

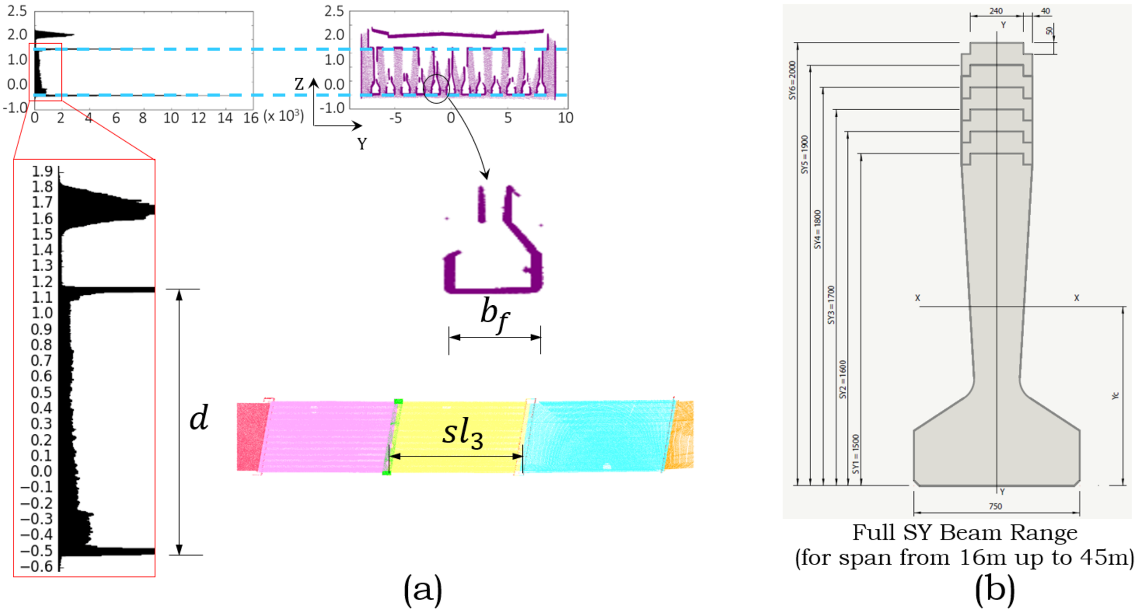


Figure 5.26 Girder type template matching (a) criteria extraction; (b) best matching type from catalogue

Once the girder type is determined, the author next describes the profile using IFC standards. Similar to how the slab slice and pier caps are extruded, when modelling a girder point cluster the author uses the feature points of the selected girder section to describe its geometry. For instance, using the above-mentioned criteria, a girder point cluster is matched with a standard pre-stressed wide flange concrete girder, e.g. WF50G (Figure 5.27 (a)). Then, given the coordinates of the starting middle bottom point (green point in Figure 5.27 (b)) which is derived from each segmented girder point cluster, and the dimensions of WF50G, then, each feature point (red point in Figure 5.27 (b)) can be defined accordingly with the exact coordinate information (the minor chamfer is neglected).

This template matching method does not require all feature points to be detected. It can produce complete girder dimensions based on several matched criteria. This is important as many girder points may be missing in the data.

The robustness and efficiency of this method with regard to the data defects such as occlusions and sparseness will be tested in experiments.

Then, the author stores the coordinates of each feature point in a 2D Cartesian point *IfcCartesianPoint* in its local XY-coordinates, followed by mapping the contour with a list of *IfcPolyline* objects (Figure 5.27 (b)). A 2D profile *IfcArbitraryClosedProfileDef* is used to describe the girder profile. The girder is then represented using the extruded geometry model through *IfcExtrudedAreaSolid* and *IfcShapeRepresentation*, expressing it as a Swept Solid. Assume the girders in each span are straight (see A14, Chapter 3, Section 3.4), the extruded direction is defined by the starting and end middle bottom points of a girder point cluster. As the girder point clusters have already been segmented, it is easy to extract the starting and end middle bottom points by using the two extremities. Next, the author introduces the property set *Pset_GirderProperties,* in which the attributes such as Girder Type, Length, and Slope are given. The length and slope information of a girder can be calculated using its OBB representation or the starting and end middle bottom points. Again, the method can provide additional flexibly undefined attributes for future use. Figure 5.28 shows an example of a snippet of IFC codes.

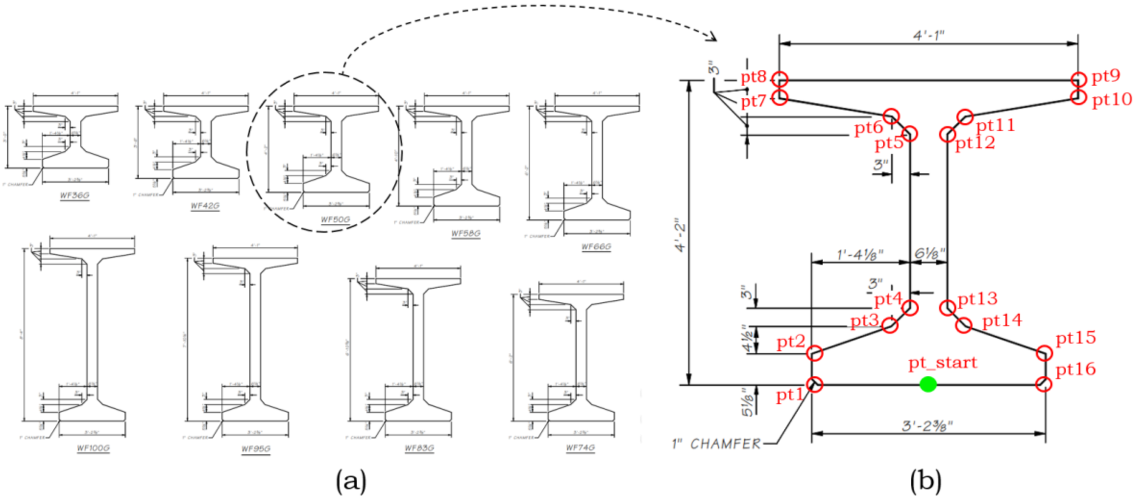


Figure 5.27 Standard pre-stressed wide flange concrete girders (WSDoT, 2009); (b) WF42G and the feature points (in total 16 points)

```
/************************************/
/*             Girder 22              */
/************************************/
#2200001 =    IFCCARTESIANPOINT((-25550.1375,230100.0));
#2200002 =    IFCCARTESIANPOINT((-25550.1375,230230.175));
#2200003 =    IFCCARTESIANPOINT((-25883.5125,230344.475));
#2200004 =    IFCCARTESIANPOINT((-25959.7125,230420.675));
#2200005 =    IFCCARTESIANPOINT((-25959.7125,230938.2));
#2200006 =    IFCCARTESIANPOINT((-25883.5125,231014.4));
#2200007 =    IFCCARTESIANPOINT((-25415.2,231090.6));
#2200008 =    IFCCARTESIANPOINT((-25415.2,231166.8));
#2200009 =    IFCCARTESIANPOINT((-26659.8,231166.8));
#2200010 =    IFCCARTESIANPOINT((-26659.8,231090.6));
#2200011 =    IFCCARTESIANPOINT((-26191.4875,231014.4));
#2200012 =    IFCCARTESIANPOINT((-26115.2875,230938.2));
#2200013 =    IFCCARTESIANPOINT((-26115.2875,230420.675));
#2200014 =    IFCCARTESIANPOINT((-26191.4875,230344.475));
#2200015 =    IFCCARTESIANPOINT((-26524.8625,230230.175));
#2200016 =    IFCCARTESIANPOINT((-26524.8625,230100.0));
#2201 =       IFCPOLYLINE((#2200001,#2200002,#2200003,#2200004,#2200005
              ,#2200006,#2200007,#2200008,#2200009,#2200010,#2200011,#2
              200012,#2200013,#2200014,#2200015,#2200016,#2200001));
#2202 =       IFCARBITRARYCLOSEDPROFILEDEF(.AREA.,'WF42G',#2201);
#2203 =       IFCCARTESIANPOINT((-29401.50,0.,0.));
#2208 =       IFCAXIS2PLACEMENT3D(#2203,#2,#3);
#2205 =       IFCEXTRUDEDAREASOLID(#2202,#2208,#2218,17829.);
#2206 =       IFCSHAPEREPRESENTATION(#1,'Body','SweptSolid',(#2205));
#2209 =       IFCPRODUCTDEFINITIONSHAPE($,$,(#2206));
#2204 =       IFCBEAM('397973f40134471ab2ef',$,'girder',$,'girder',$,#2209,'girde
              r',$);
#2210 =       IFCPROPERTYSINGLEVALUE('Girder
              Type:',$,IFCIDENTIFIER('WF42G'),$);
#2211 =       IFCPROPERTYSINGLEVALUE('Length:',$,IFCIDENTIFIER('17.83
              m'),$);
#2212 =       IFCPROPERTYSINGLEVALUE('Slope:',$,IFCIDENTIFIER('1.07'),$);
#2213 =       IFCPROPERTYSINGLEVALUE('Property
              A:',$,IFCIDENTIFIER('N/A'),$);
#2214 =       IFCPROPERTYSINGLEVALUE('Property
              B:',$,IFCIDENTIFIER('N/A'),$);
#2215 =       IFCPROPERTYSINGLEVALUE('Property
              C:',$,IFCIDENTIFIER('N/A'),$);
#2216=        IFCPROPERTYSET('c30de12f55284474b738',$,'Pset_GirderPropertie
              s',$,(#2210,#2211,#2212,#2213,#2214,#2215));
#2217=        IFCRELDEFINESBYPROPERTIES('d91e0869ebd44c55ae1d',$,$,$,(#
              2204),#2216);
#2218=        IFCDIRECTION((0.025,0.01,0.98));
```

Figure 5.28 Snippet of the IFC codes of a girder

5.2.7. Summary

This chapter presents the second process of the proposed framework in more detail. It derives a novel object fitting method able to generate the gDT of an existing bridge in IFC format, using the four types of point clusters making up the bridge. The author suggests a method to generate a bridge gDT in two different resolutions: a LOD 200 gDT, which uses OBBs to represent bridge components, and a LOD 250—300 gDT, which uses a stacked slice representation. For the latter, a slicing-based method is proposed to approximate the as-is shapes of the underlying bridge components in point clusters. This method consists of two steps. First, it follows a slicing strategy; for each slice of a point cluster of the four component types, it extracts geometric features using *ConcaveHull*, and detects shapes using fuzzy-logic in point clusters. Second, it performs runtime fitting with the *IfcObjects* of the extracted features. The novelty of this method lies in the fact that multiple local topological configurations derived from the slicing scheme provide good characterization to approximate the global topology of the underlying bridge in a point cloud.

The outputs of this chapter are two IFC files of a bridge, corresponding to two different LOD. The experiments, results, and discussions of this method are presented in Chapter 6, Section 6.2.5.

6. Experiments & Results

This chapter starts with the data collection activities (Section 6.1), research platform development (Section 6.2.1), prototype implementation (Section 6.2.2), experimental data preparation (Section 6.2.3), and experiment details and results (Section 6.2.4 and Section 6.2.5).

6.1. Data collection and Samples

In order to test the hypothesis of this thesis (Chapter 2, Section 2.4) as well as the hypothesis for each proposed method (Chapter 4, Section 4.2.6 and Chapter 5, Section 5.2.7), the author used a FARO Focus 3D X330 Terrestrial Laser Scanner (2013) (ranging error ±2 mm at 10 m, self-levelling: accuracy 0.015° (range ±5°)) to collect PCD of ten RC highway bridges around the city of Cambridge, UK.

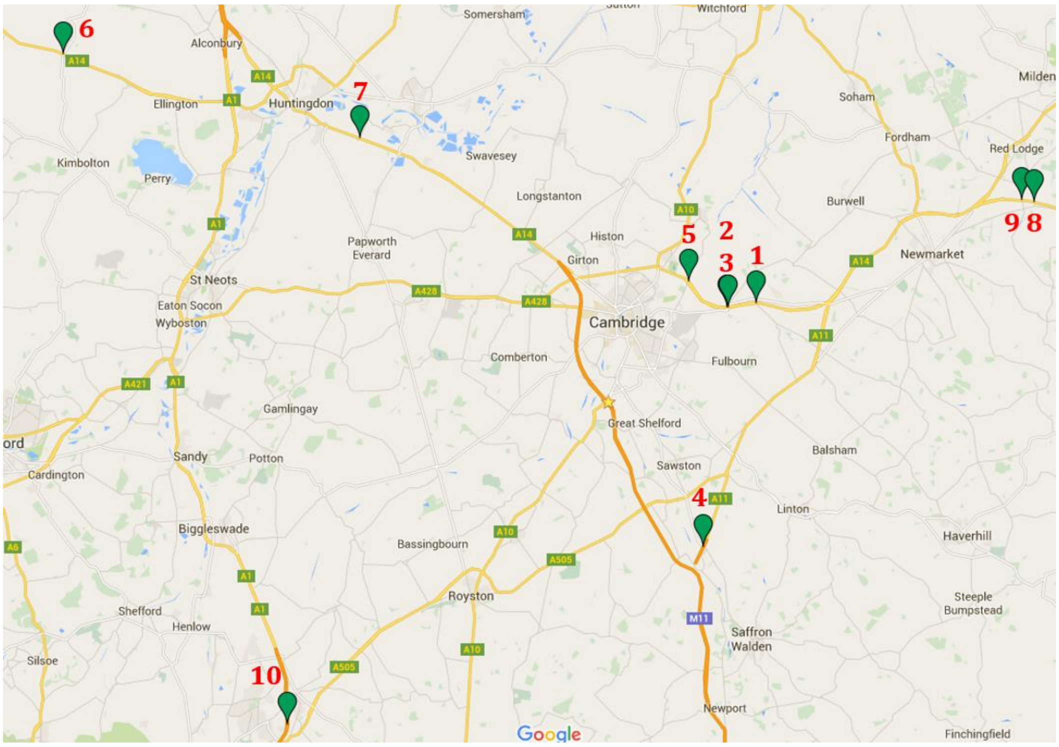


Figure 6.1 Locations of the ten RC bridges collected in this thesis

Table 6.1 shows the Global Positioning System (GPS), vertical clearance (denoted VC), and other bridge data. The author took an average of 17 scans per bridge to ensure that every visible bridge surface was scanned. This was achieved through multiple scans from different vantage points in order to minimize occlusions and ensure a complete data set with points on all of a bridge's visible surfaces. Table 6.2 shows the scan stations of each bridge (left column) and the resulting registered PCD (right column). The PCD is available at: http://doi.org/10.5281/zenodo.1233844.

The distance between the captured scan points was set to be in 7.67 mm over a scan distance of 10 m (except for inaccessible standpoints). The average on-site scanning time was 2.82 hours per bridge. After the on-site scanning, the author registered all raw scans using the FARO Scene software (2012) using natural features. Note that natural references such as planes and corners were used for automatic registration alignments as it was impractical to place artificial spheres on-site. On average, the registration time was 10.6 hours per bridge. The registration quality was good. The achieved averaged relative registration accuracy (denoted $\overline{\text{acc}}$ in Table 6.1) was 96.7%. This was estimated from the averaged fraction between pair reference-point distances in the registered scan data and the corresponding on-site pair point distances using a measuring tape, independent of other error sources:

$$\overline{\text{acc}} = \frac{1}{M}\sum_{i=1}^{M}\left(\frac{\text{pair reference}-\text{point distance}}{\text{on}-\text{site pair point distance}}\right)_i, \qquad \text{(Eq. 6.1)}$$

where M is the number of investigated pair-wise distances. The author selected three pairs of distances to calculate this value for every bridge. The spatial completeness of the datasets was computed based on rough estimation of the occlusion-data ratio (Table 6.1).

$$\text{Completeness} = \frac{\text{occluded area}}{\text{total area}}, \qquad \text{(Eq. 6.2)}$$

Although occlusions are inevitable in some cases due to on-site vegetation, trees and barriers, the key features, edges, and boundary points of every bridge are visible in the results and the occluded areas are very limited, being inferior to 5% of the total surface for almost all the bridges, except *Bridge 7*. Nearly 8% of the surface points in *Bridge 7* were not captured, which is due to limited line-of-sight to the girder areas underneath the deck. The large size of the excluded data (non-data) is mainly due to the manual removal of the on-site traffic noise, trees, large ground surfaces, ramps, and abutments. This activity will be presented later in this chapter.

Note that several factors may affect the measuring accuracy, such as low temperatures, which may condense elements inside the scanner. As the author conduced this data-collection work during the cold winter season (February and March 2016), the scanner needed to be warmed up before every task until its internal temperature stabilized. The author also used the built-in inclinometer to store the inclination of the levelled scanner.

	Bridge 1		***Bridge 2***		***Bridge 3***		***Bridge 4***		***Bridge 5***	
GPS	52.216821, 0.243050		52.21415, 0.215174		52.21424, 0.216835		52.07585, 0.193352		52.22903, 0.179660	
Scanning	3.5 hrs	18 scans	3.3 hrs	17 scans	3.2 hrs	19 scans	4 hrs	25 scans	3.2 hrs	16 scans
Registration	14 hrs	$\overline{acc}$: 0.98	8 hrs	$\overline{acc}$: 0.96	12 hrs	$\overline{acc}$: 0.97	14 hrs	$\overline{acc}$: 0.96	6 hrs	$\overline{acc}$: 0.97
Completeness	97.9%		98.0%		98.3%		95.0%		98.7%	
Original Size	26,235,309		11,852,062		12,284,105		12,655,082		10,157,137	
Non-data ratio	23.7%		31.4%		43.5%		36.8%		55.1%	
VS (m)	span 1	span 2	span 1	span 2	span 1	span 2	span 1		span 1	
	5.00	5.20	5.21	5.30	4.96	5.15	5.90		5.36	

	Bridge 6		***Bridge 7***		***Bridge 8***		***Bridge 9***		***Bridge 10***	
GPS	52.35976, -0.406092		52.31170, -0.12829		52.27482, 0.503124		52.27626, 0.491071		51.97319, -0.19567	
Scanning	2.5 hrs	17 scans	2 hrs	14 scans	2.3 hrs	16 scans	2.2 hrs	16 scans	2 hrs	16 scans
Registration	8 hrs	$\overline{acc}$: 0.96	6 hrs	$\overline{acc}$: 0.97	6 hrs	$\overline{acc}$: 0.96	24 hrs	$\overline{acc}$: 0.96	8 hrs	$\overline{acc}$: 0.98
Completeness	98.2%		92.2%		97.8%		98.8%		98.5%	
Original Size	79,787,269		53,708,475		81,610,875		80,915,621		77,688,416	
Non-data ratio	14.5%		42.8%		11.5%		14.2%		11.1%	
VS (m)	span 1	span 2	span 1	span 2	span 1	span 2	span 1	span 2	span 1	span 2
	5.89	5.60	5.20	5.40	5.68	6.05	5.32	5.60	5.27	5.15

The data can be accessed at the following link: http://doi.org/10.5281/zenodo.1233844

Table 6.1 Metadata of ten RC bridge point cloud datasets

Figure 6.2 and Figure 6.3 show the on-site data collection activities. The data collection team consisted of two members. The team took personal protective equipment (PPE) and warning cones to ensure on-site safety. The PPE included high-visibility vest, helmet, work gloves, and safety shoes. The team also placed several large warning cones 10—20 meters away from the site. A 5 kg weight was hung on the bottom of the scanner tripod to ensure its stability. A flashing red-light and a high-visibility vest were mounted on the top of the tripod to raise the visibility of the scanner. For future data collection practice, the author recommends the following points.

- Lane Closure

It is highly recommended that inspectors request for a lane closure for future practice as it is impractical for inspectors to set up a large tripod in the shoulders or refuge areas of a highway, which are usually small, narrow, and difficult to manoeuvre in. Despite multiple safety measures being taken into account (placing large cones, hanging weights and mounting a flashing light on the tripod), cars and heavy trucks ignored these warning signs, passing by with high speed which created huge turbulence that put the scanner at risk. With traffic control, a laser scanner could be placed in the middle of the lanes to ensure a better quality of scan result. Given the on-site scanning time is limited (2.82 hours/bridge), the author believes that temporary lane closure will not affect the traffic too much.

- Data Collection Plan

It is also highly recommended that inspectors make a detailed data collection plan including safety measurements, data requirements, equipment requirements, and on-site precautions, etc., before going on-site. Inspectors should maximize the advantages of daytime natural lighting and avoid cold seasons and bad weather. Cloudy days are preferable. The collected data must be crosschecked before leaving the site.

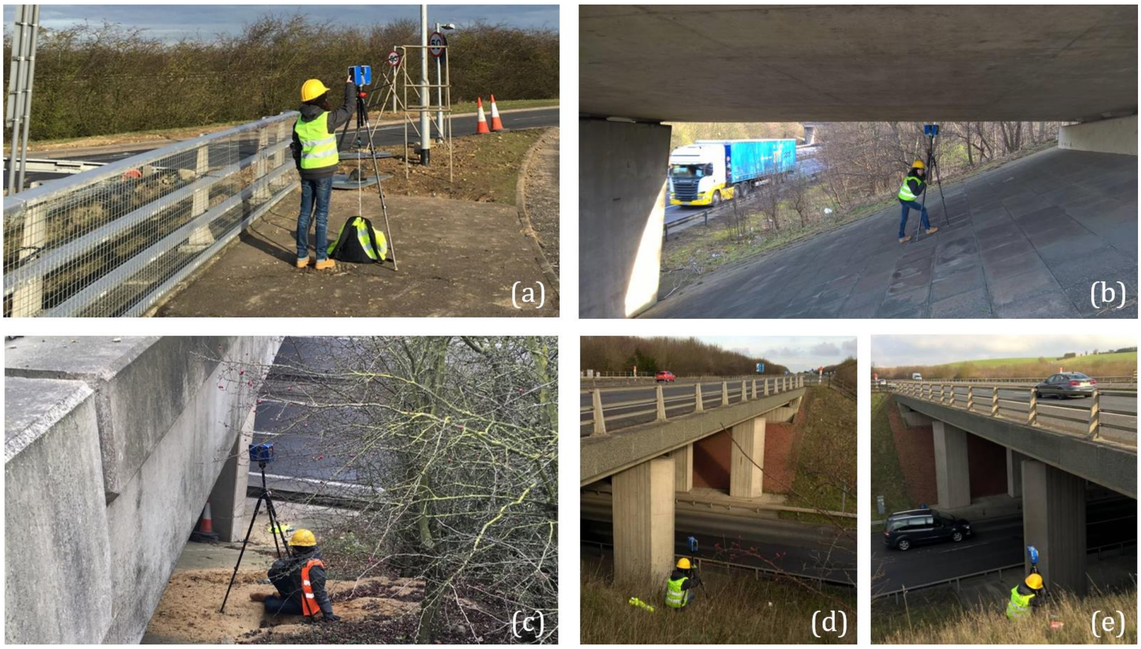


Figure 6.2 On-site scanning (a) on top of the deck; (b) on the abutment; (c) (d) (e) on the ramp

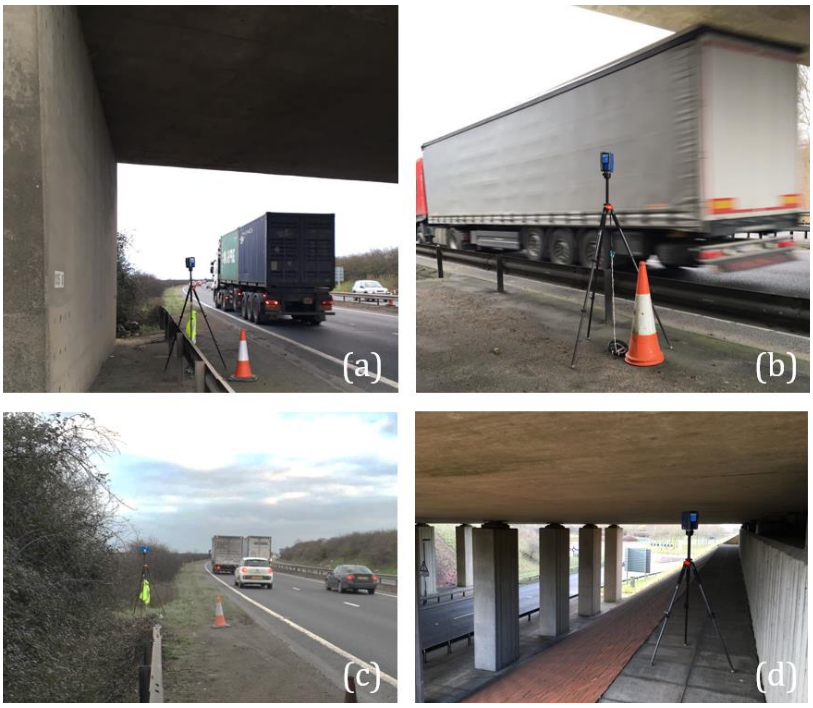
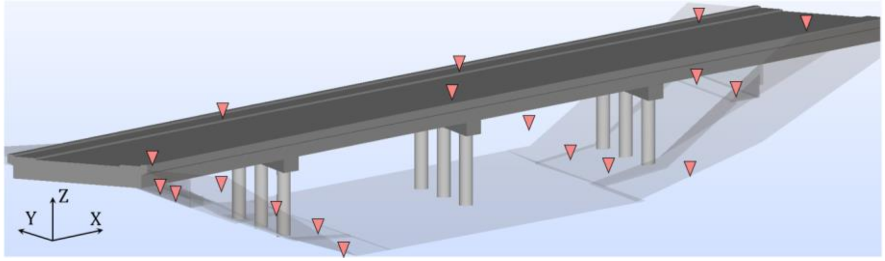
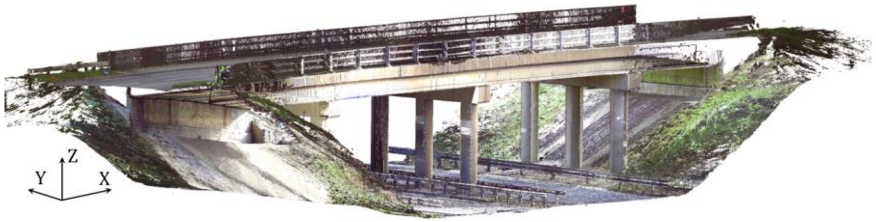
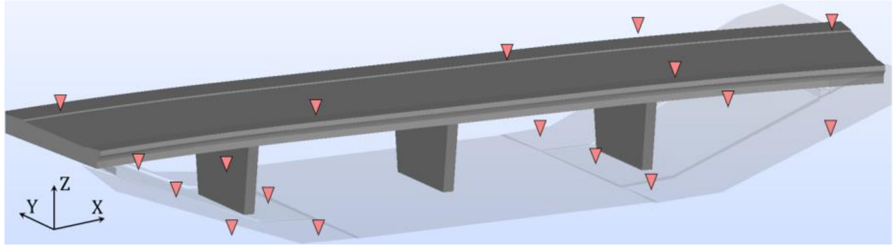
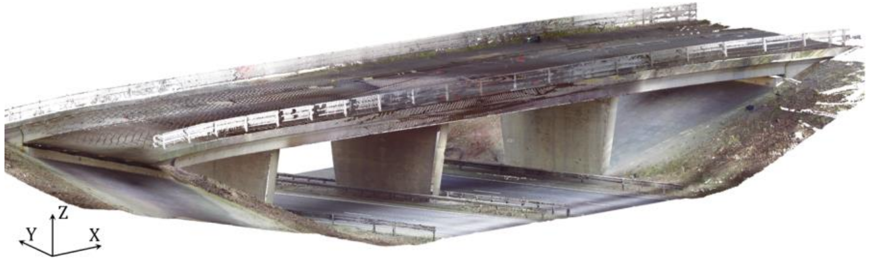
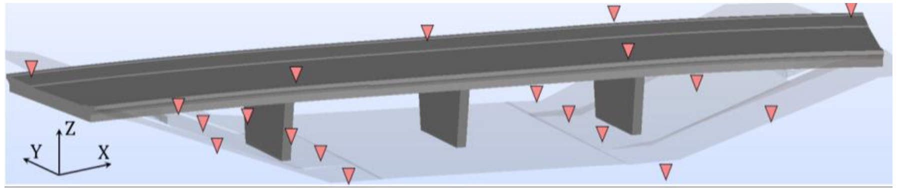
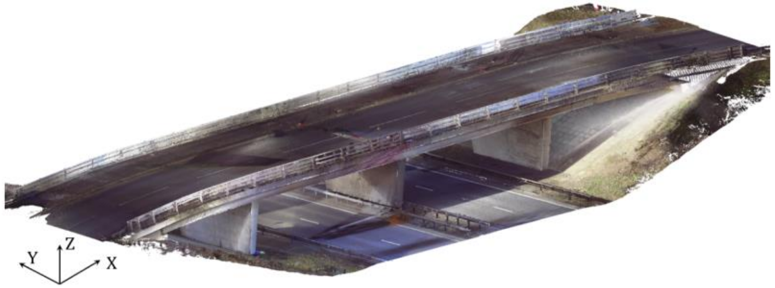
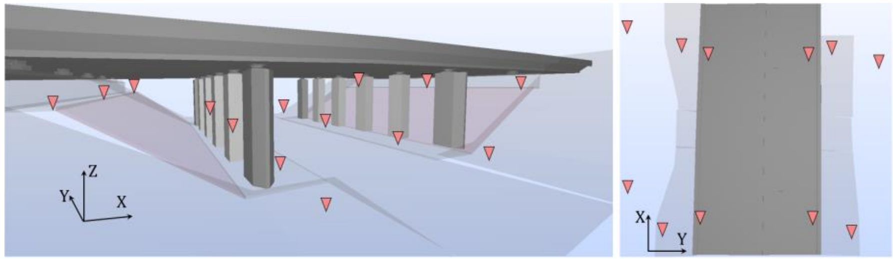
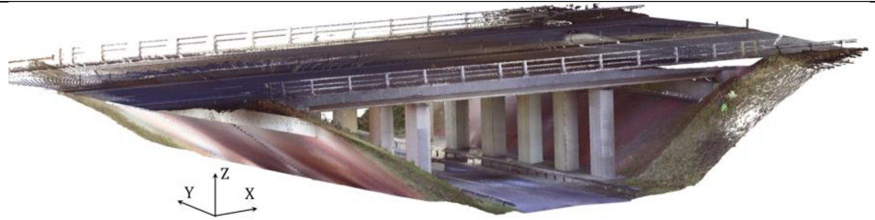
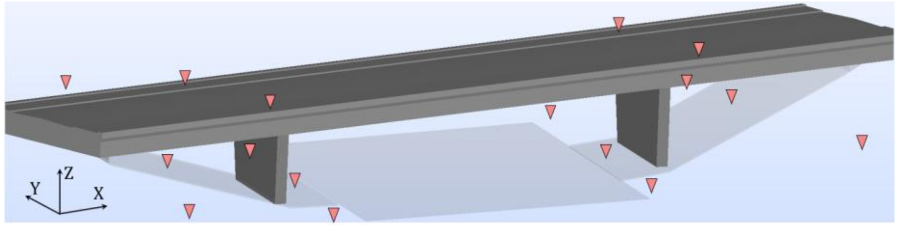
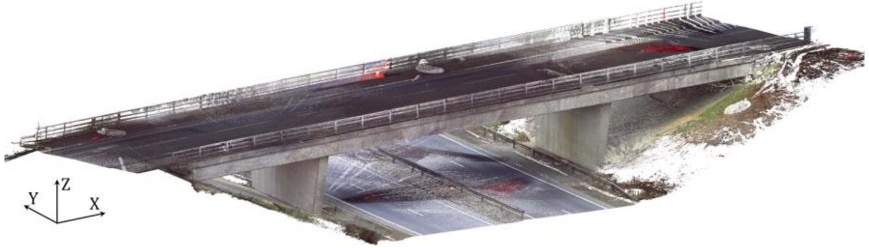
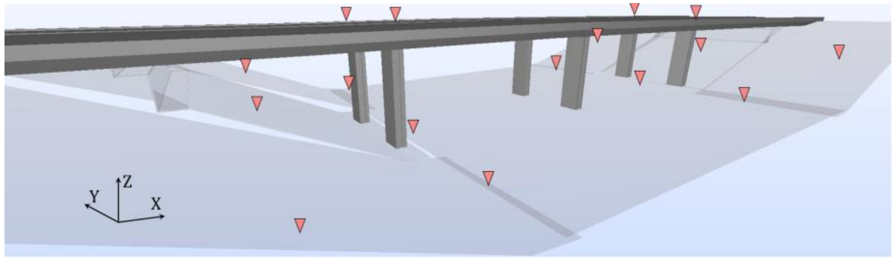
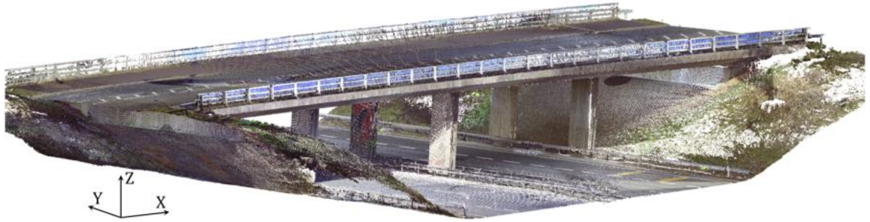
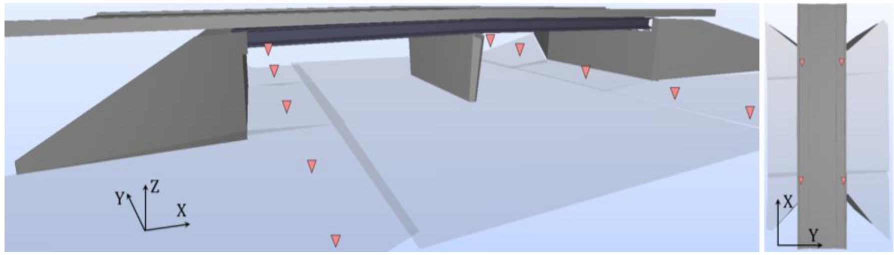
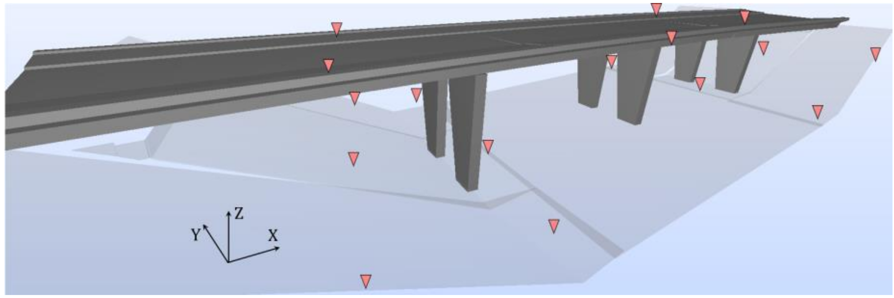
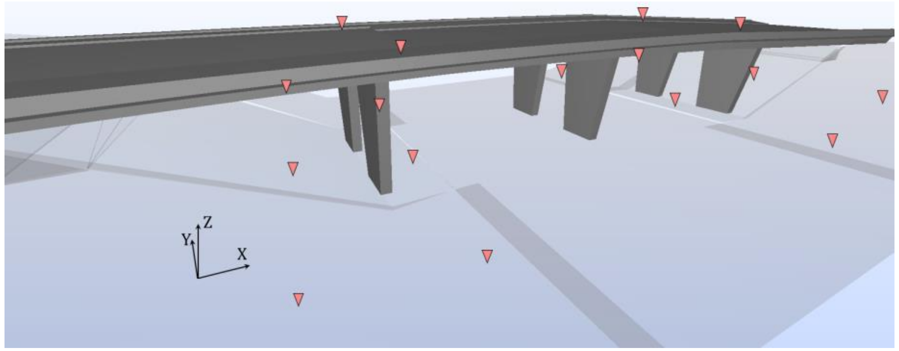


Figure 6.3 On-site scanning (a) (b) tripod warning; (c) behind the thorns; (d) on the platform

	Scan Stations	Registered PCD
Bridge 1		
Bridge 2		
Bridge 3		

Bridge 4		
Bridge 5		
Bridge 6		

Bridge 7		
Bridge 8		
Bridge 9		

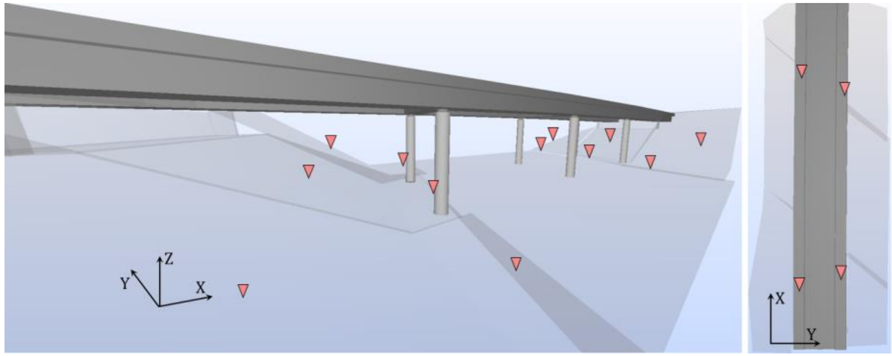
Bridge 10		

Table 6.2 Point cloud datasets of ten RC highway bridges and their scan stations

The sample type used to test the proposed object detection method (Chapter 4) as well as the object fitting method (Chapter 5) are bridge structural components. The author calculated the sample size (SS) using:

$$\mathrm{SS} = \frac{4(Z_{crit})^2 p(1-p)}{D^2}, \qquad \text{(Eq. 6.3)}$$

where Z_{crit} is the standard normal deviation corresponding to the selected significance criteria and confidence interval (CI), and p is a pre-study estimate of the proportion to be measured, also called the category proportion, which is given by the component proportion of the whole slab and beam-slab bridge population in the UK. To estimate the value of p, the author randomly selected 100 slab and beam-slab bridges on the UK's highways from Google Maps (Figure 6.4) and calculated the number of the four types of structural component appearing in each bridge. D is the total width of CI, also called the error limit (EL). Table 6.3 shows the calculated sample size with regard to different confidence levels (CL) and error limits. As the calculated SS decreases, the margin of error grows, so, more bridge data (from 27 more bridges) is needed to achieve a good CL (90%) with a relatively small EL (0.2).

However, the cost and risk of data collection is extremely high, as the author must operate a laser scanner next to a live carriageway and face significant traffic hazards. This thesis therefore considers this proof of concept study validated if it achieves a low performance variance over the ten bridge datasets (i.e. CL=90%, EL=0.4). To the author's best knowledge, this research has the largest PCD collection of real-world RC bridges.

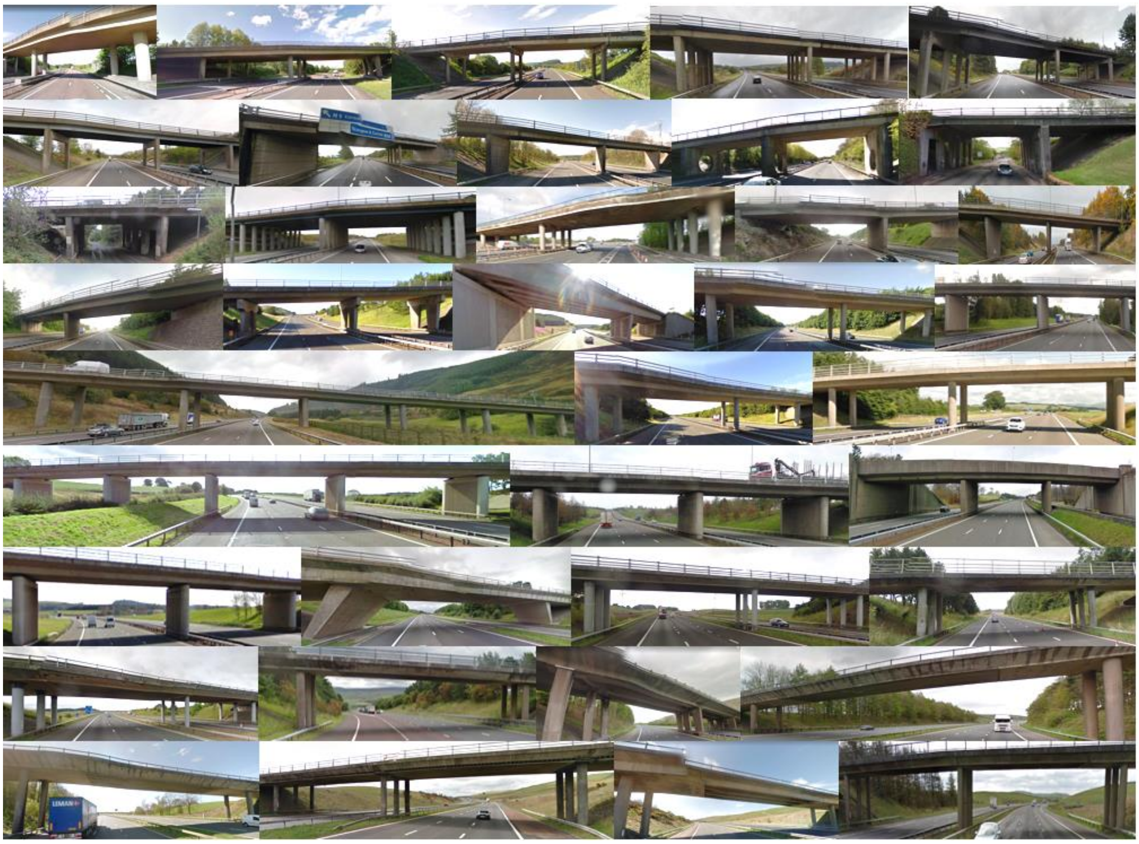


Figure 6.4 Subset of 100 UK bridges used for the sample size determination

CL	Z_{crit}	EL	Slabs	Pier caps	Piers	Girders
90%	1.645	0.1	147	32	208	107
		0.2	37	8	52	27
		0.3	17	4	24	12
		0.4	10	2	13	7
92%	1.75	0.1	166	36	234	121
		0.2	42	9	59	31
		0.3	19	4	27	14
		0.4	11	3	15	8
95%	1.96	0.1	208	45	296	152
		0.2	52	12	74	38
		0.3	24	5	33	17
		0.4	13	3	19	10

Table 6.3 Sample size determination

6.2. Research activities

6.2.1. Research platform

The Construction IT group at the University of Cambridge has developed a research coding platform that provides students with a barebones open source code they can use to get quick access to powerful libraries and develop software prototypes. This platform is called Gygax. It can simultaneously access PCD, IFC files, images, videos and other file types. It provides the ability to load them in memory, visualize and process them simultaneously. It is based on the .NET framework and is written in C#.

The Gygax solution contains three different types of projects (in different programming languages): GygaxCore (C#), GygaxVisu (C#), and PclWrapper (C++). GygaxCore defines basic functionalities such as the data structures and interfaces of this platform. GygaxVisu supports visualization. Helix Toolkit (2016) is used to visualize all supported data sources in a single 3D space without having major performance loss. Helix Toolkit is an open source 3D library that provides all major functions required for working in 3D space along with good performance in the DirectX version. Several wrappers were developed to enable Gygax to use open source libraries which are available in other languages. For example, OpenCV in combination with EmguCV as C# wrapper is used for image and video processing. Likewise, Point Cloud Library (PCL, 2018) is a popular open-source library for point cloud processing but is basically available in C++. Hence, a PclWrapper is integrated into Gygax as one project which can access to PCL functionality in a C# environment. Gygax uses IFC Engine DLL (RDF Ltd., 2017) to read and visualize IFC files on a logical and geometrical level.

All the above-mentioned components were combined in the Gygax platform which is available on GitHub (https://github.com/ph463/Gygax/). The algorithms developed for validating the proposed framework were implemented on Gygax into a proof of concept prototype in a desktop computer with a specification as shown in Table 6.4.

	PC Configuration
Motherboard	Asus Z79 Pro Gamer Intel Z97
CPU	Intel Core i7-4790K 4.00GHz
Memory	TeamGroup Vulcan Gold 32GB
SSD hard disk	Samsung 500GB SSD 2.5”
Graphics card	Sapphire Radeon R9 270X Boost

Table 6.4 The workstation configuration of this research

6.2.2. Implementation

6.2.2.1. Data cleaning-up and alignment

As the collected data contains points from live traffic, the raw PCD is extremely noisy. Data cleaning-up was conducted before running any experiments. The author developed a user-defined 2D clipping polygon function on Gygax to manually delete irrelevant points such as the on-site traffic, vegetation, ground surface and so on.

The current graphics view of 3D points need to be mapped onto a 2D screen by controlling the viewpoint to conduct the clipping. The viewing characteristics used to display a 3D world point onto a 2D screen are controlled by a set of graphics properties. The 3D points in the world coordinates are first transformed into a camera space followed by a perspective projection to convert the camera coordinates into the screen coordinates. This was achieved by using the Viewport3DX in Helix Toolkit. Viewport is a virtual camera which specifies the current viewport by calculating the azimuth and elevation with respect to the axis origin. Azimuth is a polar angle in the XY-plane, with positive angles indicating counter clockwise rotation of the viewpoint. Elevation is the angle above (positive angle) or below (negative angle) of the XY-plane (Figure 6.5). The current viewport of Viewport3DX then uses the projection matrix to project the 3D points in camera coordinates (x, y, z) onto the 2D screen (x', y'):

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}, \qquad \text{(Eq. 6.4)}$$

so that $x' = f\frac{x}{z}$ and $y' = f\frac{y}{z}$, where f is the distance from the screen to the origin. Note that (x', y') is still in physical units and needs to be converted into pixel units. The whole process of mapping points in a virtual 3D space to a 2D screen is similar to taking a picture of a real-world object. One needs to first convert the world coordinates to camera coordinates, then to projection coordinates, and finally to pixel coordinates or image coordinates (Eq. 6.5).

$$p(x_{pix}, y_{pix}) = \frac{1}{Z} K[\mathrm{I} \quad 0][\mathrm{R}|\mathrm{T}] \begin{bmatrix} x_w \\ y_w \\ z_w \\ 1 \end{bmatrix}, \tag{Eq. 6.5}$$

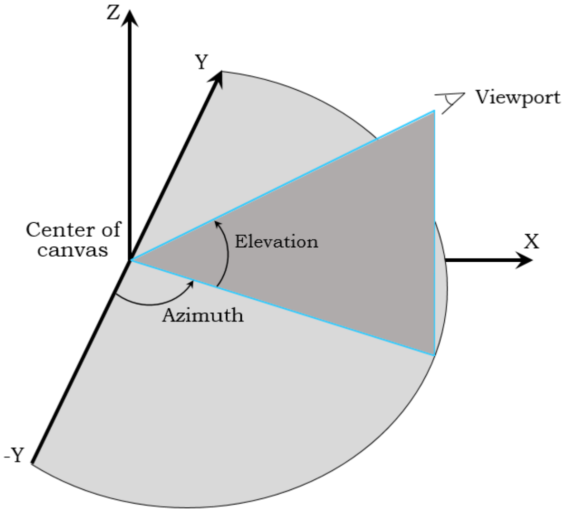


Figure 6.5 Azimuth and elevation

Then, the cropping function works such that one can perform multiple clicks to draw a closed 2D polygon to select points of interest. This cropping function checks each screen point if it is inside the 2D polygon. If so, the corresponding 3D points are colour coded and displayed on the screen, then going to be discarded. The process is repeatedly performed until the irrelevant points such as the on-site traffic, vegetation, ground surface, and ramps are all removed (Figure 6.6).

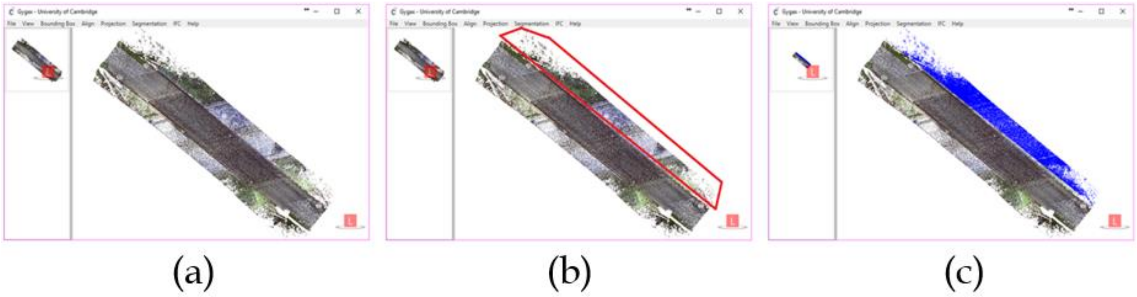


Figure 6.6 Point cloud cropping (a) insert a raw point cloud into Gygax; (b) use clipping polygons to select region of interest; (c) selected points are coloured and removed

Note that this is the only required manual work. The proposed four-step object-detection method in Chapter 4 is fully automatic without any human intervention, because it is easy for a modeller to delete irrelevant points than a computer. The author chose not to focus on automating this step given how little impact it would have on the modeller's time and effort. By contrast, it is not such easy for a modeller to precisely segment a point cloud of complex geometries on a 2D computer monitor.

The author then randomly down-sampled the cropped bridge PCD and produced a down-sampled RC bridge PCD with the key components containing less than 1 million points. The reason for down sampling is that the original size was not used for manual gDT generation as it is difficult for commercial software to handle large datasets. Thus, the bridge PCD is down sampled in order to compare the processing time between the manual and automated gDT generation. Next, the author used PCA to align the cropped bridge PCD so that the major axes of the bridge were positioned roughly parallel to the global axes X-Y-Z (Figure 6.7). None of the bridges could be positioned exactly parallel to the axes due to their real-world skewed geometry. The rough alignment was achieved using the Statistic Analysis in Accord.NET, which provides numerous statistical analyses, machine learning, image processing, and computer vision methods for .NET applications.

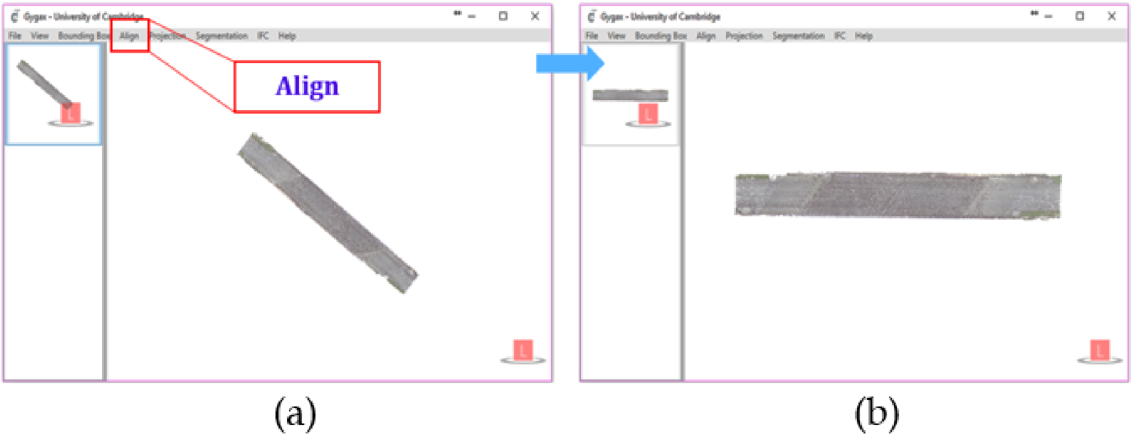


Figure 6.7 PCA alignment (a) before alignment; (b) after alignment

6.2.2.2. Object detection

The proposed 4-step object detection method (Chapter 4) was implemented on Gygax as four different classes. A Unified Modelling Language (UML) Diagram of the object detection is shown in Figure 6.8. The author created a class called SegmentationController, which is used for undertaking the logic of the object detection method and the user interface. The author also generated a base class BaseSegmentation, defining properties related to the segmented data used by subclasses, and abstract methods implemented by subclasses. In addition, the BaseSegmentation class includes a segmentation configuration class called SegmentationCongfig, which is used for reading the pre-set configuration parameters in Extensible Markup Language (XML) for each step as well as providing information to the method logic. That is to say, once a user clicks the interface button to call the SegmentationController class, the methods in this class will create a subclass of BaseSegmentation using different set of parameters. They will also execute the startSegment method in different subclasses through SegmentController class. Specifically, there are four subclasses inherited from the superclass BaseSegmentation in the object detection module. They are SegmentationStepOne, SegmentationStepTwo, SegmentationStepThree, and SegmentationStepFour, corresponding to the four steps in the object detection method. A user clicks an interface button to call methods of each subclass of BaseSegmentation in order to perform each step of the proposed method. The segmented results of each step were then saved to specified folders in directory.

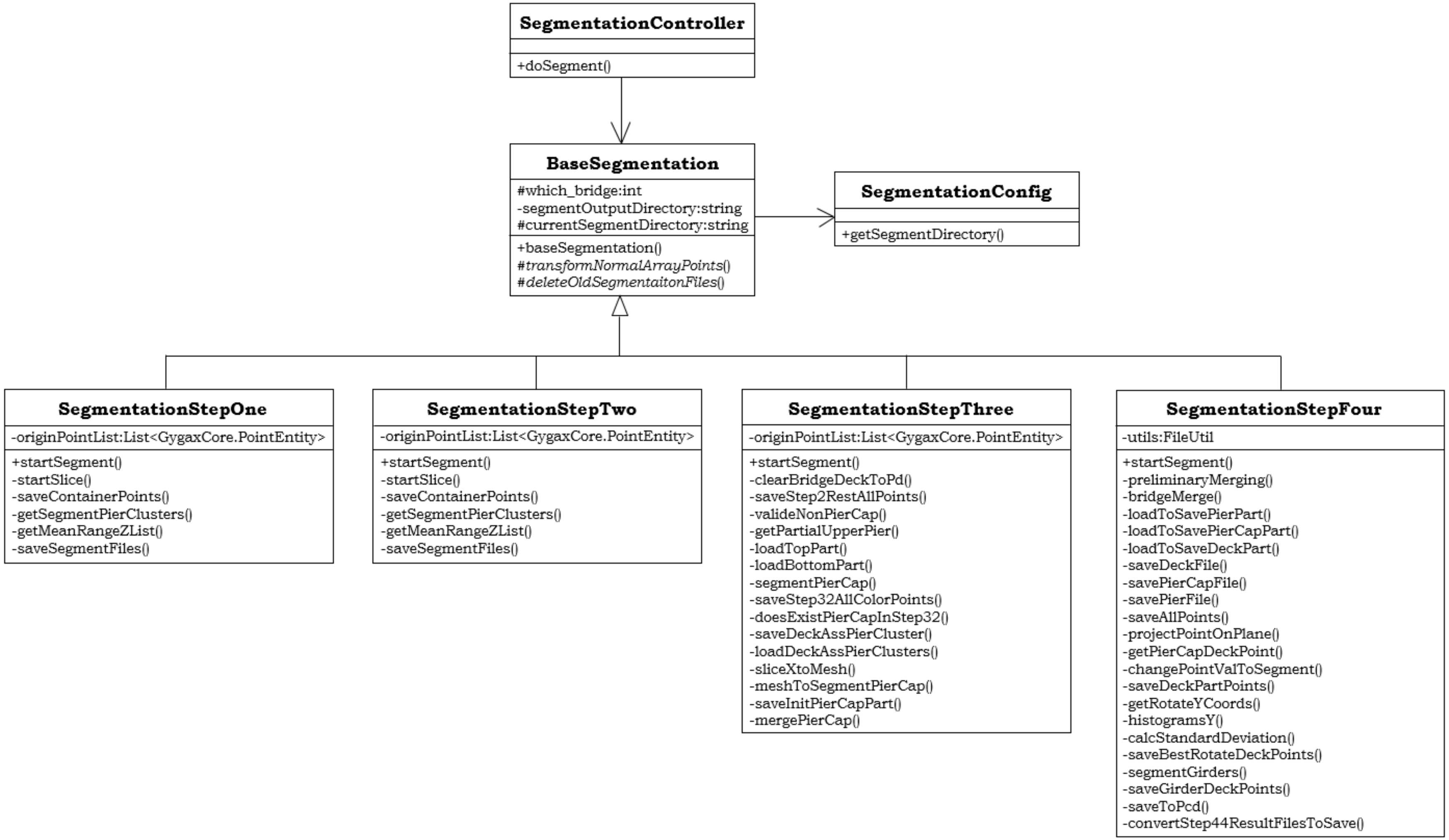
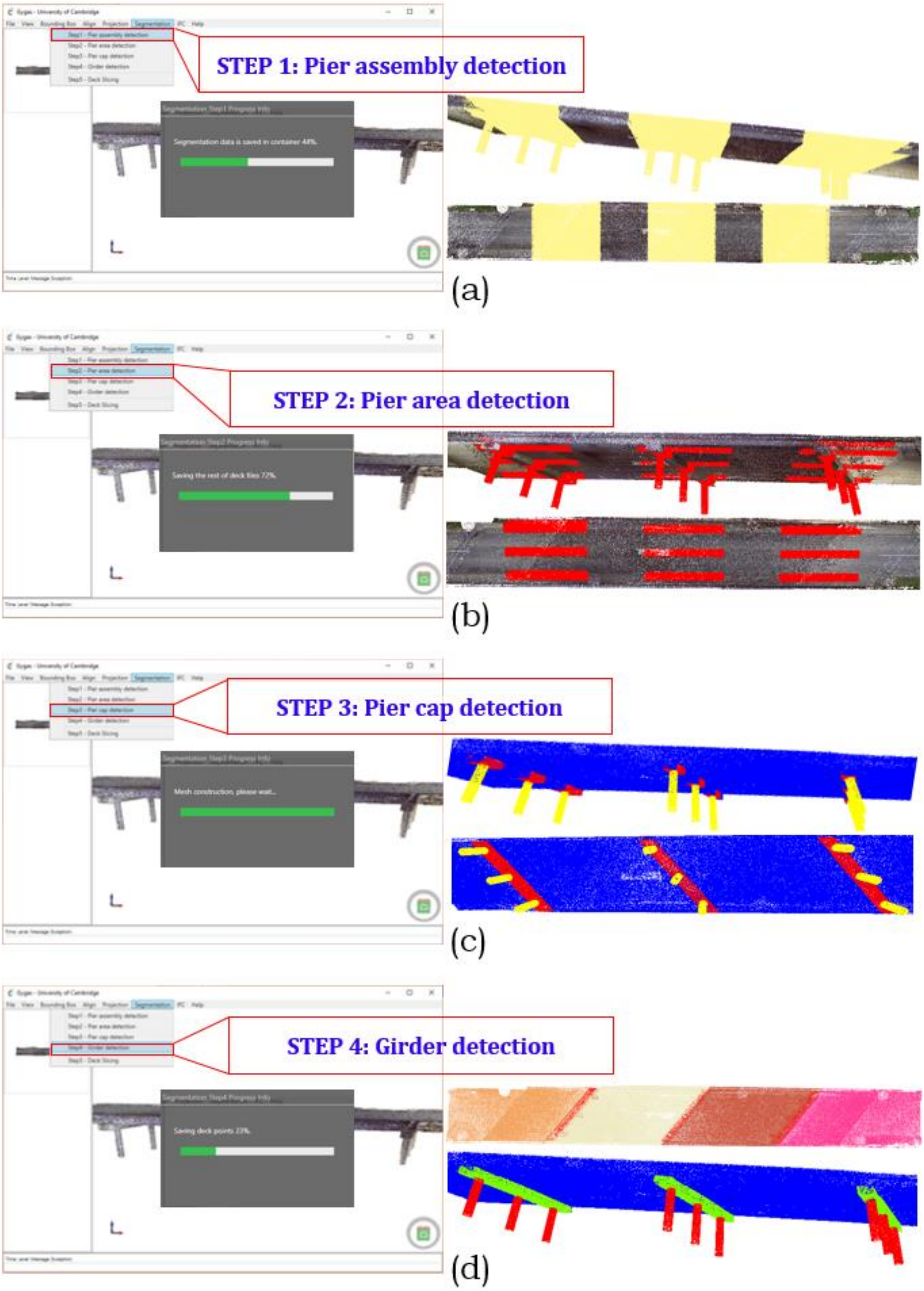


Figure 6.8 UML diagram of object detection module

The author shows 2 bridge examples: *Bridge 1* and *Bridge 7*. *Bridge 1* contains deck slab, pier caps, and piers while *Bridge 7* contains deck slab, girders, and pier. Figure 6.9 illustrates the implementation of *Bridge 1*. First, the author started with a roughly aligned point cloud produced previously using PCA, followed by conducting Step 1 using the Tab “Segmentation → Step 1 – Pier assembly detection”. Three pier assembly clusters were detected and encoded in yellow (Figure 6.9 (a)). Then, the author conducted Step 2 by using each pier assembly to detect pier areas through the Tab “Segmentation → Step 2 – Pier area detection”. Nine pier areas were detected (three for each pier assembly) and encoded in red (Figure 6.9 (b)). Next, the author conducted Step 3 by using each pier area to detect pier caps through the Tab “Segmentation → Step 3 – Pier cap detection”. Figure 6.9 (c) shows that the method first detected a pier cap part on top of each pier area and continued extracting entire pier caps if they were present. Finally, the author conducted Step 4 to detect girders through the Tab “Segmentation → Step 4 – Girder detection”. As shown in Figure 6.9 (d), the method first segmented the merge deck into four segments (spans) followed by detecting the girders in each segment. The object detection process terminated, and three types of labelled point cluster making up *Bridge 1* were produced.

Likewise, the author conducted the same workflow to detect components in *Bridge 7*. Figure 6.10 (c) shows that there was no pier cap detected in *Bridge 7*, as it has a wall-like pier which does not need a pier cap to distribute the loading from the deck. Figure 6.10 (d) shows that the method segmented the merged deck assembly into two segments followed by detecting all the girders underneath each segment. Finally, three types of labelled point cluster making up *Bridge 7* were produced.



Gygax - University of Cambridge
File View Bounding Box Align Projection Segmentation IFC Help
Step1 - Pier assembly detection
Step2 - Pier area detection
Step3 - Pier cap detection
Step4 - Girder detection
Step5 - Deck Slicing
Time Level Message Exception
STEP 1: Pier assembly detection
(a)
STEP 2: Pier area detection
(b)
STEP 3: Pier cap detection
(c)
STEP 4: Girder detection
(d)

Segmentation Step1 Progress Info

Segmentation data is saved in container 44%.

Segmentation Step2 Progress Info

Saving the rest of deck files 72%.

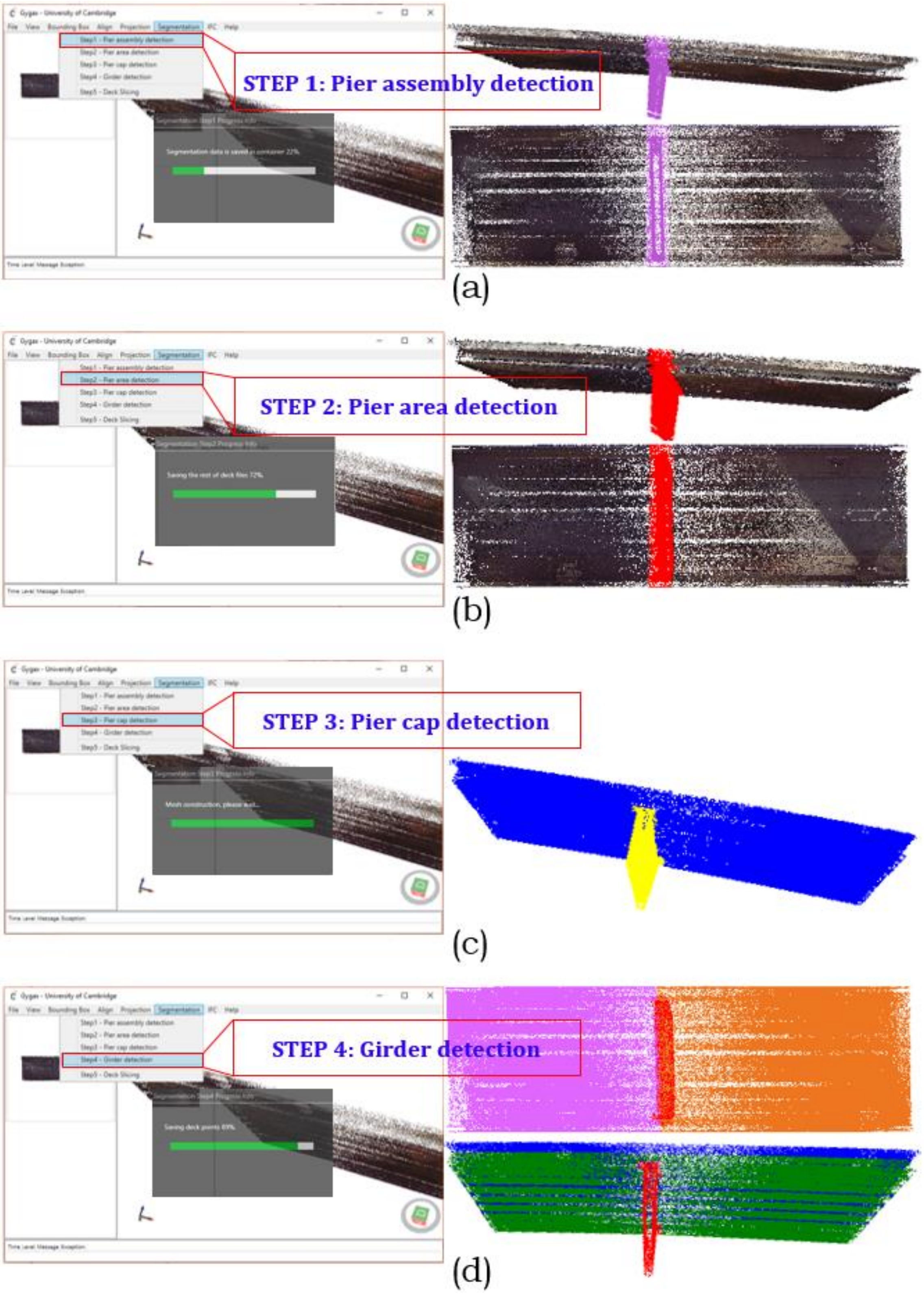
Segmentation Step3 Progress Info

Mesh construction, please wait...

Segmentation Step4 Progress Info

Saving deck points 23%.

Figure 6.9 Implementation of the 4-step detection method on *Bridge 1*



Gygax - University of Cambridge
File View Bounding Box Align Projection Segmentation IFC Help
Step1 - Pier assembly detection
Step2 - Pier area detection
Step3 - Pier cap detection
Step4 - Girder detection
Step5 - Deck Slicing
Time Level Message Exception
STEP 1: Pier assembly detection
(a)
STEP 2: Pier area detection
(b)
STEP 3: Pier cap detection
(c)
STEP 4: Girder detection
(d)

Segmentation Step1 Progress Info

Segmentation data is saved in container 22%

Segmentation Step2 Progress Info

Saving the rest of deck files 72%

Segmentation Step3 Progress Info

Mesh construction, please wait...

Segmentation Step4 Progress Info

Saving deck points 89%

Figure 6.10 Implementation of the 4-step detection method on *Bridge 7*

6.2.2.3. IFC object fitting

The proposed IFC object fitting method (Chapter 5) was implemented on Gygax as two different classes. A UML Diagram of the object detection is shown in Figure 6.11. The author created a class called IFCGeneratorProvider, which is used for undertaking the logic of the IFC object fitting method and the user interface. The IFCGeneratorProvider class includes an Enumeration class called FineLevel, representing the specific LOD for a generated IFC file. The author also generated a base class IFCBaseGenerator, which provides methods of reading IFC template files, methods of getting object vertices, and other methods implemented by subclasses. In addition, the IFCBaseGenerator includes a configuration class called IFCGenerationConfig, which is used for reading the pre-set configuration parameters in XML used for generating IFC files. That is to say, once a user clicks the interface button to call the IFCGeneratorProvider class, the methods in this class will create a subclass of IFCBaseGenerator according to different set of parameters. They will also execute the startGenerateIFCFile method in different subclasses. Specifically, the author first generated a subclass called DeckSegmentation inherited from the BaseSegmentation. This is used to segment the deck assembly into multiple oriented segments. Then, there are two subclasses inherited from the superclass IFCBaseGenerator in the IFC object fitting module. They are IFCLoD200Generator and IFCLoD250300Generator, corresponding to the two IFC files with different LOD. A user clicks an interface button to call functions of each subclass of IFCBaseGenerator in order to generate a specific IFC LOD file. The generated files were then saved in specified folder in directory.

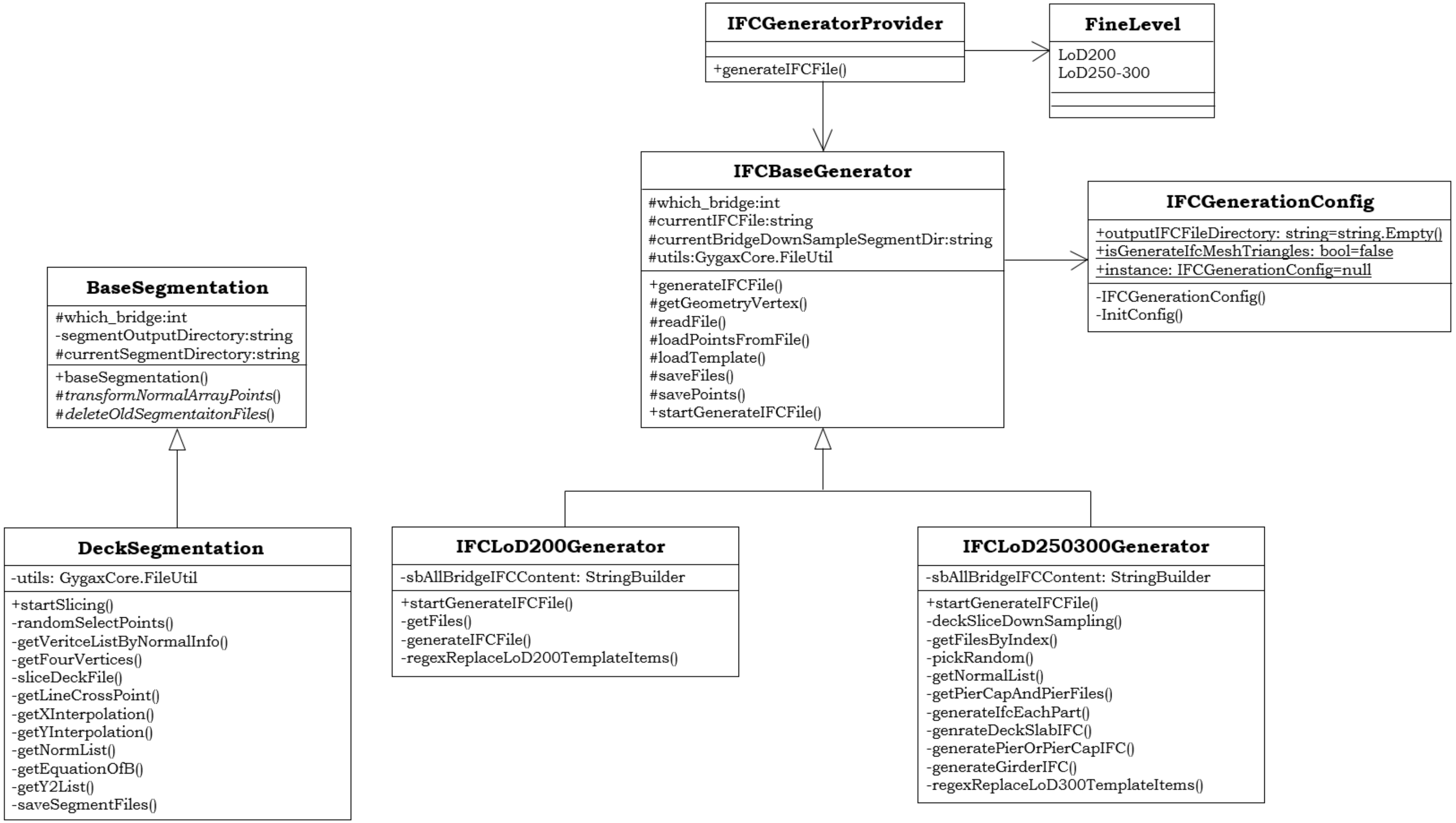


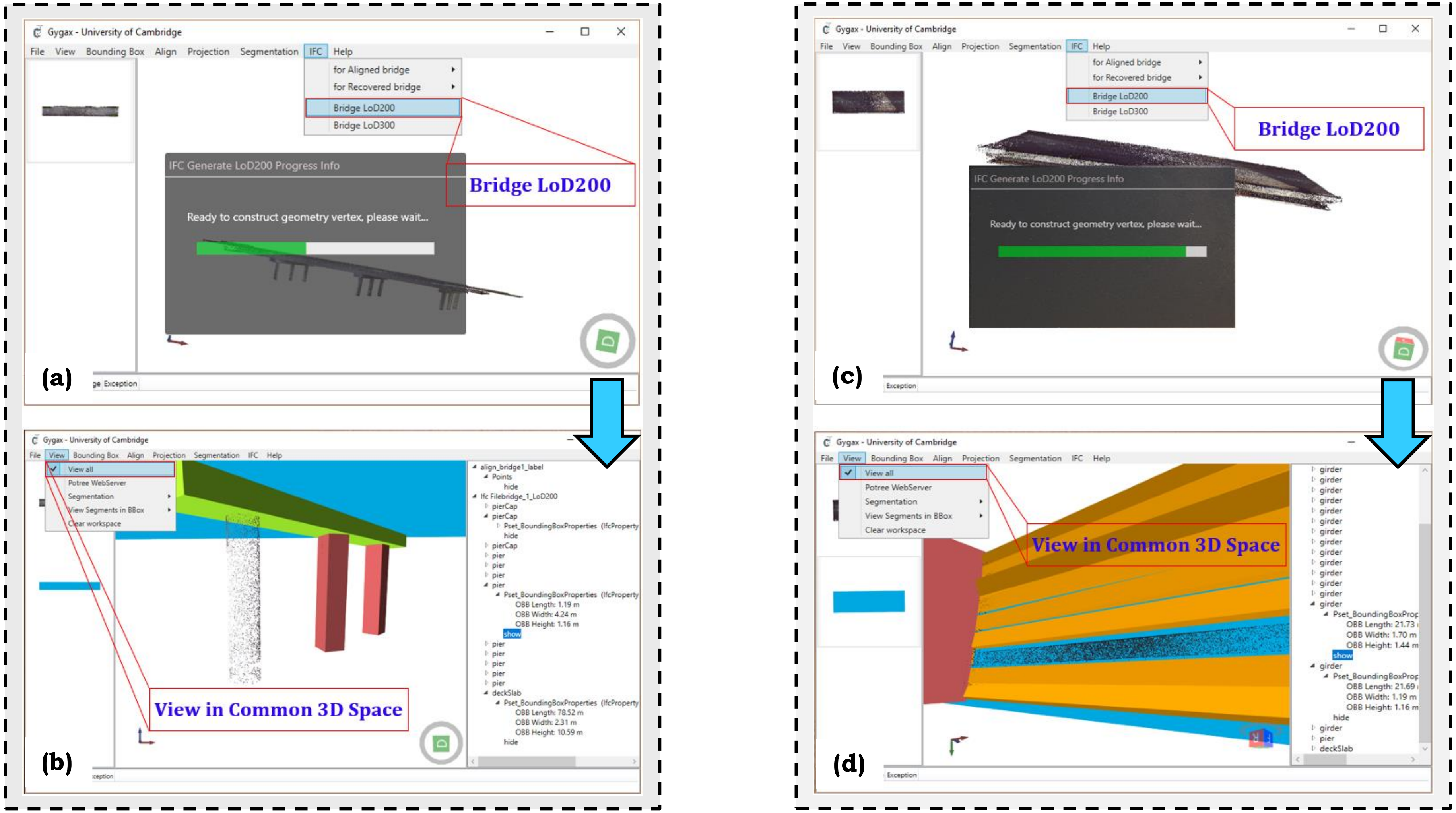
Figure 6.11 UML diagram of IFC object fitting module

The author shows three bridge examples: *Bridge 1*, *Bridge 7*, and *Bridge 9*, of IFC object fitting using the proposed method. The blue colour represents the deck slab, green represents pier caps, red represents piers, and orange represents girders. *Bridge 1* contains deck slab, pier caps and piers, *Bridge 7* contains desk slab, girders and pier, and *Bridge 9* contains deck slab, piers as well as obvious curved horizontal alignment. For LOD 200, the author only demonstrates *Bridge 1* and *Bridge 7* while for LOD 250—300, the author shows all 3 bridges.

- **LOD 200**

Figure 6.12 illustrates the implementation of *Bridge 1* and *Bridge 7* for LOD 200 gDT generation. First, a bridge PCD was loaded followed by clicking the Tab "IFC → Bridge LoD200" (Figure 6.12 (a) and (c)) to start the runtime IFC object fitting process. Once the fitting was done, one can load the generated LOD 200 .ifc file in the local directory, and its corresponding point cloud as well. Then, the author used the Tab "View → View all" to simultaneously show both the LOD 200 bridge gDT in IFC format and the overlaid PCD in the main window of the interface (Figure 6.12 (b) and (d)). This common view functionality is used to show the fitting result of the method. LOD 200 gDTs use OBBs to describe the point clusters.

The elements shown in the main window were listed in the right-hand side of the interface. It is a tree view that maps the logical tree structure of an IFC model. Each node lists the corresponding property details. The property set and the attributes of the modelled component (dimensions of an OBB such as length, width, and height) were represented as child nodes of a model element node in the 3D view. In addition, as illustrated in Figure 6.12 (b), one can hide an OBB of a cylindrical pier while its corresponding points were shown. Similarly, Figure 6.12 (d) illustrates a hidden girder of *Bridge 7* while its points were shown.


(a)
Gygax - University of Cambridge
File View Bounding Box Align Projection Segmentation IFC Help
for Aligned bridge
for Recovered bridge
Bridge LoD200
Bridge LoD300
IFC Generate LoD200 Progress Info
ge Exception
(b)
Gygax - University of Cambridge
File View Bounding Box Align Projection Segmentation IFC Help
View all
Potree WebServer
Segmentation
View Segments in BBox
Clear workspace
ception
(c)
Gygax - University of Cambridge
File View Bounding Box Align Projection Segmentation IFC Help
for Aligned bridge
for Recovered bridge
Bridge LoD200
Bridge LoD300
IFC Generate LoD200 Progress Info
Exception
(d)
Gygax - University of Cambridge
File View Bounding Box Align Projection Segmentation IFC Help
View all
Potree WebServer
Segmentation
View Segments in BBox
Clear workspace
Exception

Bridge LoD200

Ready to construct geometry vertex, please wait...

View in Common 3D Space

```
align_bridge1_label
  Points
    hide
Ifc Filebridge_1_LoD200
  pierCap
  pierCap
    Pset_BoundingBoxProperties (IfcProperty
    hide
  pierCap
  pier
  pier
  pier
  pier
    Pset_BoundingBoxProperties (IfcProperty
      OBB Length: 1.19 m
      OBB Width: 4.24 m
      OBB Height: 1.16 m
    show
  pier
  pier
  pier
  pier
  pier
  deckSlab
    Pset_BoundingBoxProperties (IfcProperty
      OBB Length: 78.52 m
      OBB Width: 2.31 m
      OBB Height: 10.59 m
    hide
```

```
girder
girder
girder
girder
girder
girder
girder
girder
girder
girder
girder
girder
girder
girder
  Pset_BoundingBoxProp
    OBB Length: 21.73
    OBB Width: 1.70 m
    OBB Height: 1.44 m
  show
girder
  Pset_BoundingBoxProp
    OBB Length: 21.69
    OBB Width: 1.19 m
    OBB Height: 1.16 m
  hide
girder
pier
deckSlab
```

Figure 6.12 Implementation of the LOD 200 model generation method: (a) generate *Bridge 1* OBBs; (b) overlap OBBs and *Bridge 1* PCD; (c) generate *Bridge 7* OBBs; (d) overlap OBBs and *Bridge 7* PCD

- **LOD 250—300**

Figure 6.13 illustrates the implementation of *Bridge 1*, *Bridge 7*, and *Bridge 9* for LOD 250—300 gDT generation. First, similar to the LOD 200 gDT process, the author loaded a bridge PCD that needed to be modelled using the Tab "IFC → Bridge LoD300" to start the runtime IFC fitting process. Then, both the generated LOD 250—300 gDT and the point cloud were simultaneously displayed in the main window using the Tab "View → View all". Figure 6.13 (a) shows a hidden deck slab slice in the LOD 300 gDT of *Bridge* 1. The pier caps and cylindrical piers were represented as entire components. Figure 6.13 (b) shows that the pier of *Bridge 7* was represented as a stacked slice model. The girders were represented as entire components with detailed dimensions. Figure 6.13 (c) shows the deck slab of *Bridge 9* was represented in oriented slice models that approximate well the geometry of the skewed horizontal alignment. One of the slice models was hidden while its points were shown. Again, all the attributes (both fixed and uncertain) of the displayed components in the main window were listed as child nodes in the window at the right-hand side.

The following section presents the ground truth preparation work for object detection and for IFC object fitting, respectively. The resulting ground truth data will be used to compare against the results generated from the proposed methods.

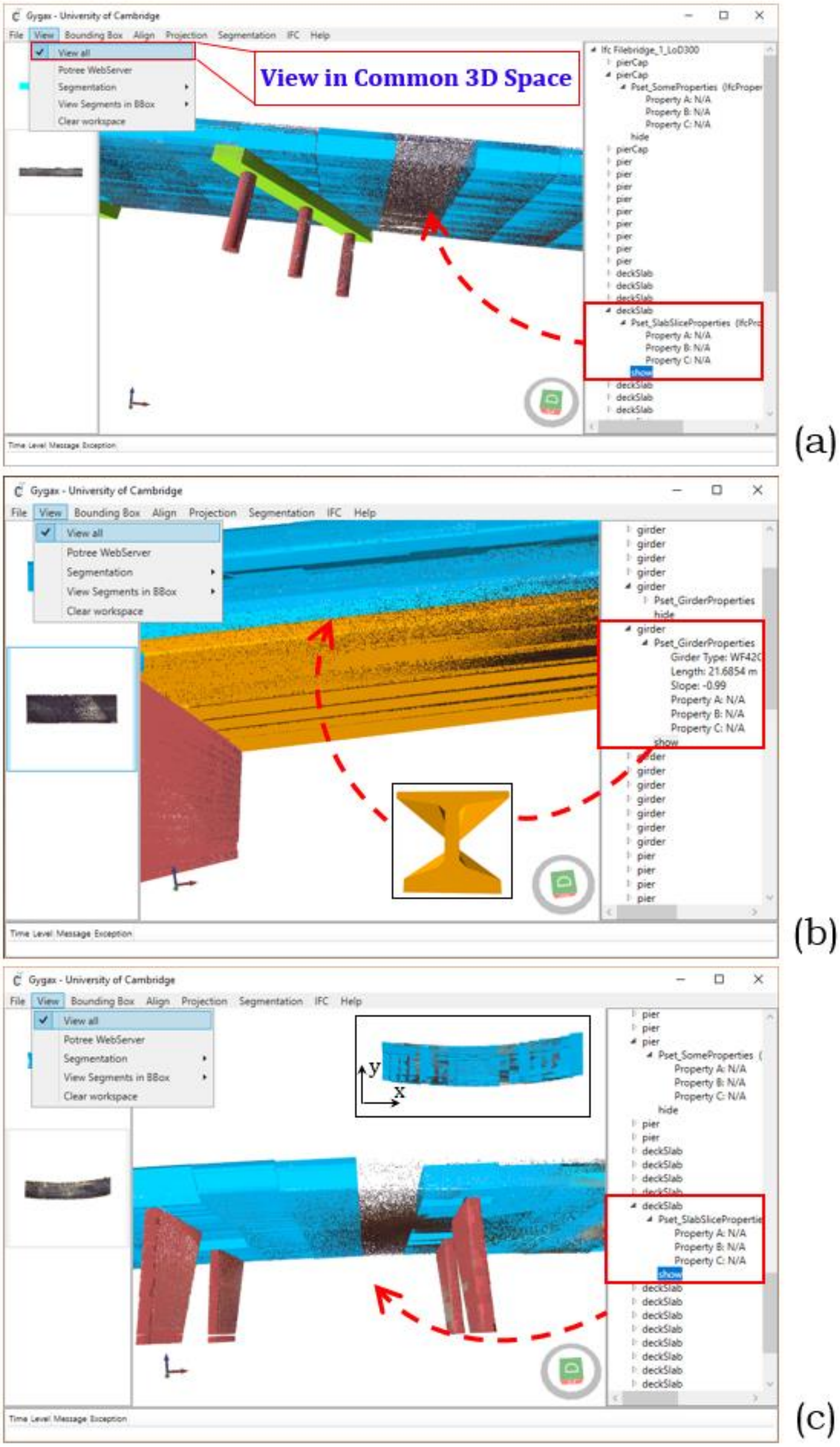


Figure 6.13 Implementation of the LOD 250—300 model generation method and overlap models and the PCD for: (a) *Bridge 1*; (b) *Bridge 7*; and (c) *Bridge 9*

6.2.3. Data preparation

6.2.3.1. Ground Truth 1 – Object detection – Chapter 4

Three ground-truth (GT) datasets were created by manually conducting Step 1 (i.e. *GT A*), Step 2 (i.e. *GT B*), and the entire solution (i.e. *GT C*) to evaluate the proposed object detection method (Section 6.2.4). The GT datasets are optimally desired outputs to compare against those generated from the proposed method.

GT A: A given bridge point cloud input was segmented into two clusters: deck assembly and pier assembly. The individual points were assigned their corresponding point-wise labels.

GT B: A pier assembly point cloud input was segmented into two clusters: deck assembly and pier area. The individual points were assigned their corresponding point-wise labels.

GT C: A given bridge point cloud input was segmented into individual point clusters as per their semantic class: structural components, including slab, piers, pier caps (if they exist), and girders (if they exist).

Figure 6.14 illustrates an example of the Ground Truth 1 preparation for one bridge (*Bridge 1*). The author prepared the three GT datasets which were used to compare against the results generated by the proposed method. The points in each GT point cluster were also assigned their corresponding point-wise labels (colour coded). In addition, each of these point clusters was bounded by an oriented-bounding-box (hereafter GTBBox) (Figure 6.14 (d)). Both the GTBBox and manually labelled points served as reference for comparison.

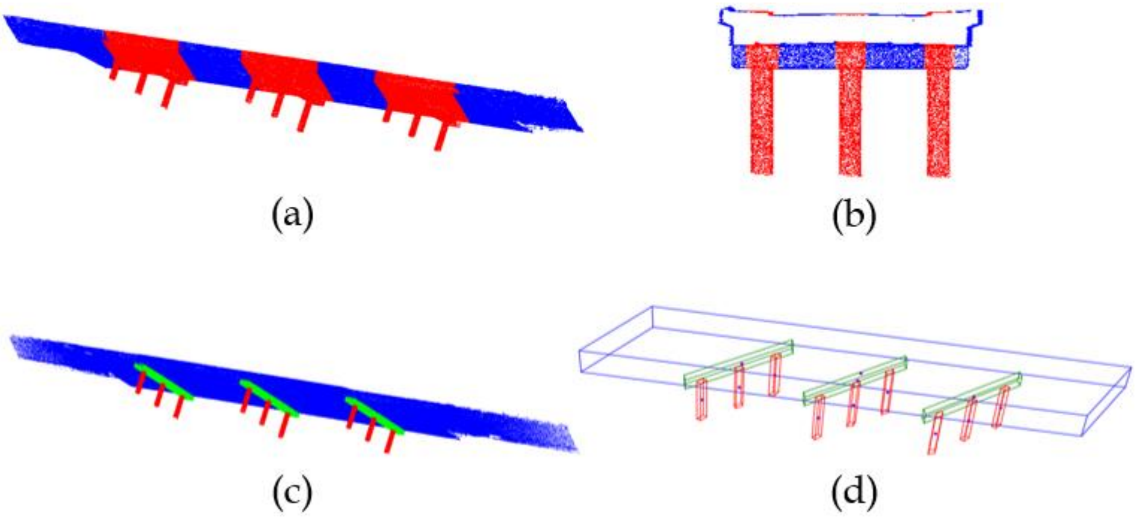


Figure 6.14 Ground Truth 1: (a) *GT A* for Step 1 – manual pier assembly detection; (b) *GT B* for Step 2 – manual pier area detection; (c) *GT C* for the entire method – manual structural component detection; (d) *GT C* for the entire method - OBB of each structural component, i.e. GTBBox

6.2.3.2. Ground Truth 2 – IFC object fitting – Chapter 5

To evaluate the proposed IFC object fitting method (Chapter 5), GT gDTs were manually generated using Autodesk Revit 2016. The author manually created two different sets of models to compare against two sets of resulting models generated from the proposed method with different resolution:

GT D: The four types of bridge structural component in this set of models were represented using their tightest cuboids (Table 6.5). These models were in line with and were compared against the automatically generated LOD 200 gDTs (Chapter 6, Section 6.2.5.3). The manual modelling time was recorded. The average time spent on creating one such bridge gDT in *GT D* was 0.92±0.79 hours (around 55.2 minutes).

GT E: The four types of bridge structural components in this set of models were represented within their precise dimensions (Table 6.6). Note that, the Revit 3D representations of these bridges are still approximate in the sense that they only describe the smooth surfaces and shape of each component. This is because potential potholes or defects in the bridge surfaces are not apparent in the point clouds. Modellers cannot easily perceive these damages. In addition, Revit structure modelling is not capable to model the defects. These *GT E* models were considered in line with and were compared against the automatically generated LOD 250—300 gDTs (Chapter 6, Section 6.2.5.4). The average time spent on creating one such bridge gDT of *GT E* was 27.6±16.4 hours (around 1656 minutes).

It is worth noting that, given that different modellers possess different discretion, experience and knowledge base, the quality of the generated GT gDT may be inconsistent. This point of view is confirmed in the experimental results. In addition, it is clear to see that the higher the resolution of gDT is, the more time is required. This point is also confirmed in the experimental results.

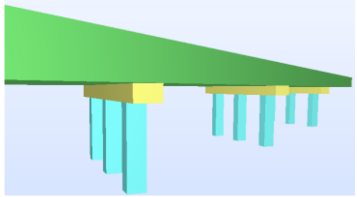
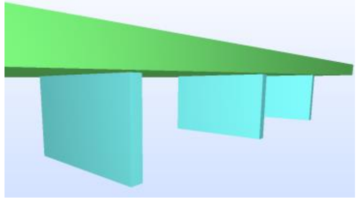
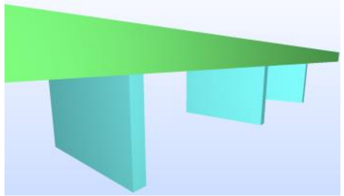
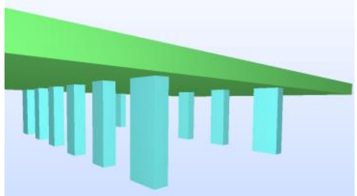
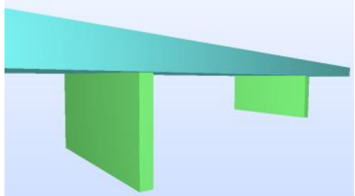
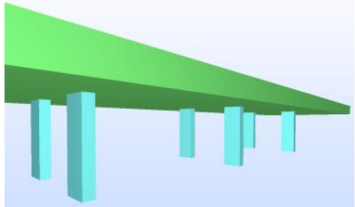
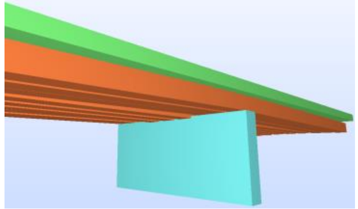
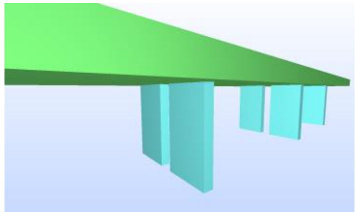
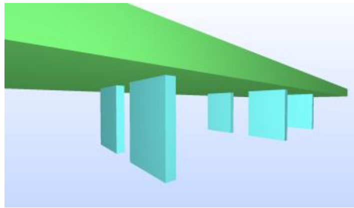
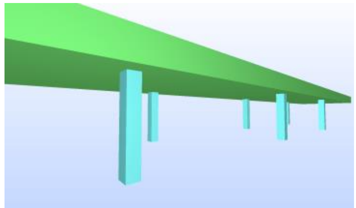
	Bridge 1	***Bridge 2***	***Bridge 3***	***Bridge 4***	***Bridge 5***
gDT					
# of points used for modelling	488,453	500,000	500,000	498,579	500,000
Time (h)	1.1	0.8	0.75	1.5	0.6
	Bridge 6	***Bridge 7***	***Bridge 8***	***Bridge 9***	***Bridge 10***
gDT					
# of points used for modelling	500,000	454,985	500,000	500,000	875,036
Time (h)	0.9	1.75	0.75	0.5	0.55

Table 6.5 Manual gDT modelling using Autodesk Revit 2016 – *GT D* (LOD 200)

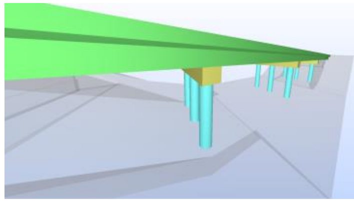
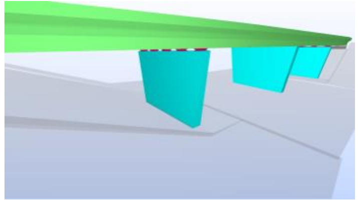
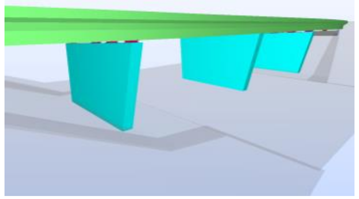
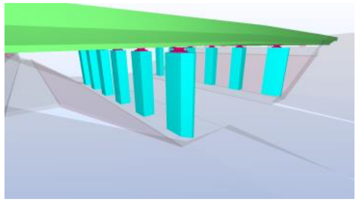
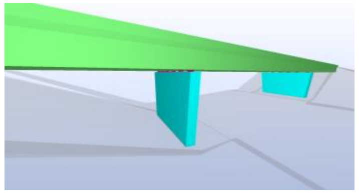
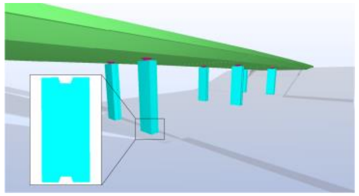
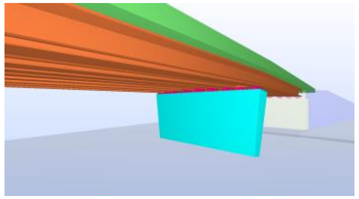
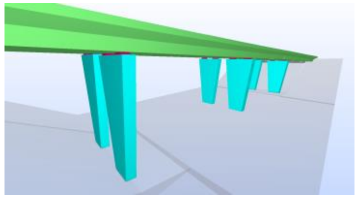
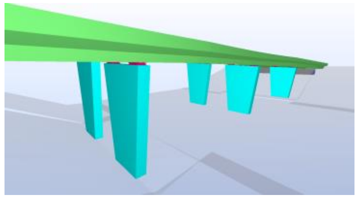
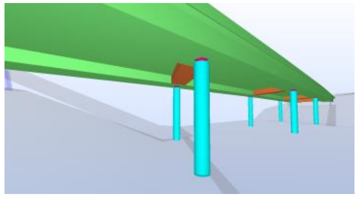
	Bridge 1	***Bridge 2***	***Bridge 3***	***Bridge 4***	***Bridge 5***
gDT					
# of points used for modelling	26,650,420	12,567,781	15,664,533	11,391,402	10,190,169
Time (h)	50	31	30	26	22
	Bridge 6	***Bridge 7***	***Bridge 8***	***Bridge 9***	***Bridge 10***
gDT					
# of points used for modelling	19,998,887	13,643,711	19,767,176	21,897,480	20,389,248
Time (h)	25	27	23	20	22

Table 6.6 Manual gDT modelling using Autodesk Revit 2016 – *GT E* (LOD 250—300)

6.2.4. Experiments of object detection – Chapter 4

This section presents the experiments that aim to answer **research question 1** (Chapter 2, Section 2.4): How to detect four types of object in imperfect RC bridge PCD where occlusions and non-uniformly distributed points exist, in form of labelled point clusters making up a bridge? The following text explains the design and execution of the experiments for the proposed object detection method presented in Chapter 4. The author elaborates how the experiments were done, what the results were, and why. The analyses consist of two parts. The first part (Section 6.2.4.1) is to experimentally define the optimal values of the two key discriminative parameters (ρ_1 and ρ_2) at the level of individual point clusters in Steps 1 and 2, respectively. Then, the author derives the optimal values of the other three hyper-parameters ($\rho_{3\mathrm{a}}$, $\rho_{3\mathrm{b}}$, and ρ_4). The second part (Section 6.2.4.2) is to assess the proposed method at the level of bridge structural components using both bounding-box-wise and point-wise performance metrics.

6.2.4.1. Estimation of hyper-parameters

In this section, the two hyper-parameters ρ_1 and ρ_2 are estimated by implementing the developed software prototype, and then the detection results of Step 1 and Step 2 are compared against *GT A* and *GT B* (Section 6.2.2.2), respectively.

Denote "*S*" as a specific point cluster, where $S \in \{\alpha_M, {\alpha_M}^C\}$ in Step 1 and $S \in \{D_{PC_M}, \beta_{MP}\}$ in Step 2. The author defined the following point-wise performance metrics Precision (Pr), Recall (R) and F1-score (F1) as:

$$\mathrm{Pr_s} = \frac{\mathrm{TP_s}}{\mathrm{TP_s+FP_s}} = \frac{\text{\# of correctly labelled points in cluster s}}{\text{total \# of points in cluster s}}, \qquad \text{(Eq. 6.6)}$$

$$\mathrm{R_s} = \frac{\mathrm{TP_s}}{\mathrm{TP_s+FN_s}} = \frac{\text{\# of correctly labelled points in cluster s}}{\text{total \# of points in GT cluster s}}, \qquad \text{(Eq. 6.7)}$$

$$\mathrm{F1_s} = 2 * \frac{\mathrm{Pr_s * R_s}}{\mathrm{Pr_s + R_s}}, \qquad \text{(Eq. 6.8)}$$

where TP refers to True Positive, FP refers to False Positive, and FN refers to False Negative. The values of ρ_1 and ρ_2 vary for different bridge configurations. To learn how sensitive the outputs of the first and the second steps of the proposed method are to ρ_1 and ρ_2, the author conducted a grid-search over the entire range of values of ρ_1 as well as ρ_2 within the value space (0, 1), and computed the empirical receiver operating characteristic (ROC), which depicts the trade-off between the true positive rate (TPR) and the false positive rate (FPR). The TPR (Eq. 6.8) is known as the sensitivity or probability of detection, being the equivalent of the Recall while the FPR (Eq. 6.9) is known as the probability of false alarm (1-specificity), where:

$$\mathrm{TPR_s} = \text{sensitity} = \frac{\mathrm{TP_s}}{\mathrm{TP_s+FN_s}}, \qquad \text{(Eq. 6.9)}$$

$$\mathrm{FPR_s} = 1 - \text{specificity} = \frac{\mathrm{FP_s}}{\mathrm{FP_s+TN_s}}, \qquad \text{(Eq. 6.10)}$$

where TN refers to True Negative. A too small or too big (close to 1) ρ_1 may lead the method to consider a whole RC bridge as either a pier assembly or deck assembly. By intuition, a relatively big value of ρ_1, for example, 0.5, can be used to extract pier assemblies in Step 1 because, normally, the height of a pier should be much larger than that of the deck. Yet, to keep a necessary vertical clearance, it is almost impossible for the thickness of the deck to be as much as 0.5 times the height of the pier assembly. The author therefore should find a reasonable value of ρ_1, which is both theoretically and realistically sound. The optimal values of ρ_1^* and ρ_2^* were identified when the distance to the perfect classification in the ROC (i.e. FPR=0, TPR=1) was minimized:

$$\rho_k^* = \operatorname{argmin}\left(d_{\rho_k}\right), \tag{Eq. 6.11}$$

where $d_{\rho_k} = \sqrt{(1-\text{sensitivity})^2 + (1-\text{specificity})^2}$, for $k \in \{1,2\}$, where k represents the number of the Step.

The optimal thickness ratio ρ_1 for each bridge was then found using (Eq. 6.11. Figure 6.15 shows an example of the ROC curve of *Bridge 1*, which suggests that the feature detector $range_{j\langle z\rangle}$ can be used to effectively discriminate between its deck assembly and pier assembly. Assuming that the samples follow a t-distribution (small sample size) with 9 degrees of freedom (i.e. n-1, where n=10), the 95% Confidence Interval (CI) critical value was derived from the t-table as 2.262 (Appendix D) for calculating the true sampling distribution mean $\mu_{\overline{\rho_1^*}}$. The optimal ρ_1 was then computed as 0.27±0.03, i.e. $\overline{\rho_1^*}$±t(s/$\sqrt{n}$), where $\overline{\rho_1^*}$ is the sample mean and s is the sample standard deviation (Table 6.7).

Bridge	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	$\overline{\rho_1^*}$	***s***
ρ_1^*	0.31	0.25	0.25	0.25	0.25	0.25	0.31	0.20	0.30	0.30	0.27	0.03

Table 6.7 Determination of ρ_1^*

The author also computed $\mu_{\overline{\rho_1^*}}$ using the bootstrapping technique (Efron, 1979) which resamples the data by replacement with a same sample size of 10 followed by repeating 1000 times. The author provided a snippet of data from the bootstrapping resampling procedure in Table 6.8. The 95% CL upper bound was estimated to be 0.29, which is in line with that of the t-statistic. The author chose the upper bound of the t-statistic to set ρ_1^* as 0.30 rather than its lower bound because all indicators such as Pr, R, F1, FPR, and d_{ρ_1} had a good balance when ρ_1=0.30 ($\overline{\mathrm{F1}}_{0.3}$=0.84, $\overline{\mathrm{F1}}_{0.24}$=0.74). More importantly, setting a bigger value for ρ_1^* can avoid extracting too many FPs ($\overline{\mathrm{FPR}}_{0.3}$=0, $\overline{\mathrm{FPR}}_{0.24}$=0.06). Figure 6.16 shows the ROC curve of Step 1 of all bridge samples. The Area Under the Curve (AUC) measures the degree of usefulness of the detector. It tells how much the detector is capable of distinguishing between two classes. The AUC of some bridges are bigger than that of the others. This is because the manual operation of different bridges in *GT A* is not consistent. However, the AUC of all bridges were bigger than 0.5, meaning that the proposed detector distinguishes deck assembly and pier assembly better than the random choice (AUC = 0.5).

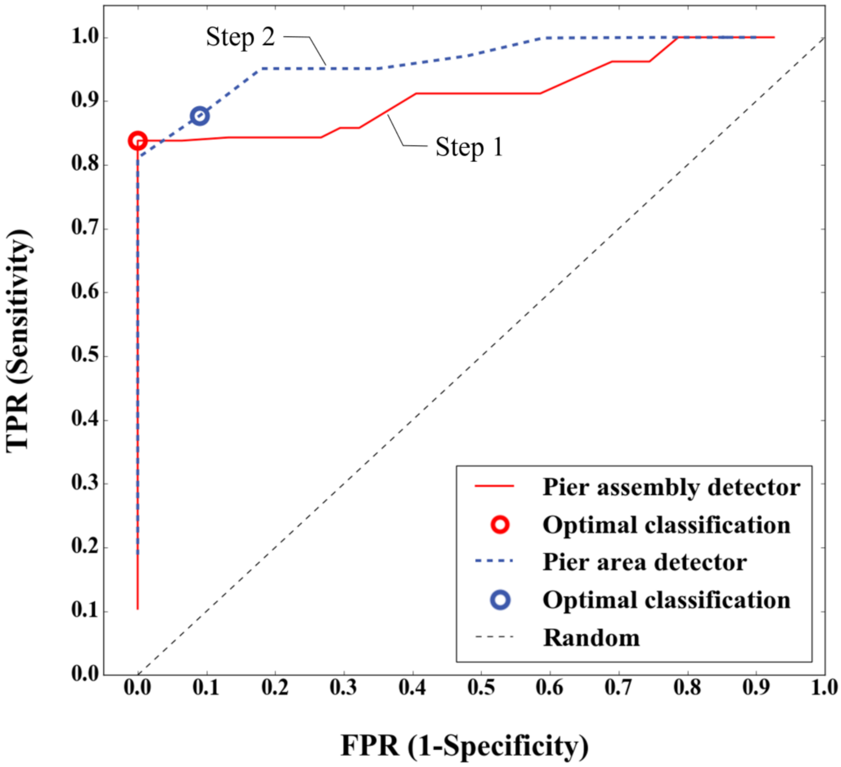


Figure 6.15 ROC of detector $range_{j\langle z\rangle}$ of pier assembly and pier area (e.g. *Bridge 1*)

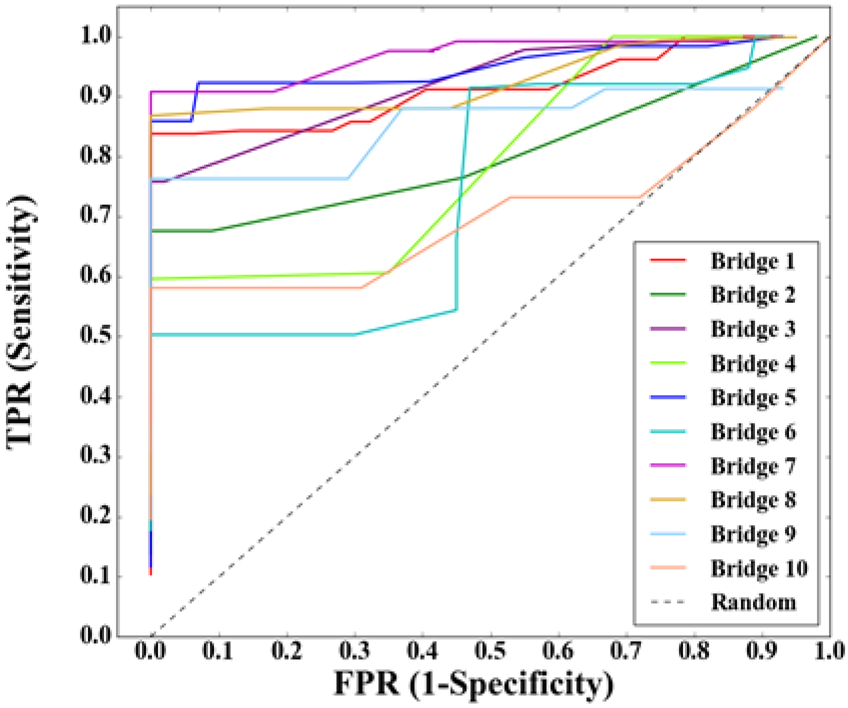


Figure 6.16 ROC of detector $range_{j\langle z\rangle}$ of pier assembly of all bridge samples

#1000 times	**resample size = 10, the same as the original sample size**										***μ***	***s***
1	0.3	0.3	0.31	0.25	0.2	0.25	0.2	0.25	0.25	0.25	0.256	0.0351
2	0.3	0.3	0.25	0.31	0.25	0.2	0.25	0.25	0.3	0.25	0.266	0.0329
3	0.31	0.31	0.25	0.3	0.31	0.31	0.31	0.25	0.31	0.31	0.297	0.0246
4	0.25	0.2	0.3	0.25	0.25	0.3	0.25	0.25	0.2	0.25	0.25	0.0333
5	0.25	0.31	0.31	0.2	0.25	0.25	0.25	0.2	0.2	0.25	0.247	0.0403
...												
595	0.25	0.25	0.25	0.3	0.2	0.31	0.25	0.25	0.2	0.25	0.25	0.0269
596	0.25	0.31	0.3	0.3	0.25	0.2	0.2	0.25	0.25	0.25	0.31	0.0372
597	0.3	0.25	0.25	0.25	0.25	0.25	0.31	0.31	0.3	0.3	0.25	0.0416
598	0.25	0.2	0.25	0.31	0.3	0.25	0.25	0.25	0.3	0.25	0.2	0.0302
599	0.3	0.31	0.25	0.2	0.25	0.2	0.25	0.25	0.31	0.3	0.31	0.0269
...												
995	0.2	0.31	0.25	0.25	0.31	0.31	0.25	0.3	0.25	0.3	0.273	0.0281
996	0.25	0.25	0.3	0.25	0.25	0.25	0.3	0.25	0.3	0.25	0.265	0.0236
997	0.25	0.31	0.31	0.31	0.31	0.2	0.3	0.25	0.3	0.31	0.285	0.0364
998	0.31	0.25	0.3	0.25	0.3	0.2	0.3	0.25	0.25	0.31	0.272	0.0346
999	0.25	0.25	0.31	0.25	0.31	0.25	0.3	0.25	0.25	0.25	0.267	0.0269
1000	0.31	0.25	0.31	0.3	0.31	0.31	0.31	0.3	0.25	0.2	0.285	0.0374
											lower	**upper**
								2.5%*1000=25 (CL=95%)			0.245	0.289

Table 6.8 Bootstrapping resampling

The optimal ρ_2 was then also grid searched in the same way followed by plotting ROC (Figure 6.17) at various threshold settings of ρ_2 for a pier assembly sample. The 95% CI of t-statistic for ρ_2 was derived to be 0.36±0.03. Likewise, the author chose to set the upper bound of ρ_2 as 0.39. Once ρ_1^* and ρ_2^* were obtained, the author calculated $\rho_{3b} = \frac{\rho_1^*}{\rho_2^*}$ ≈0.80. The ROC curves of Step 2 of all pier assembly samples are shown in Figure 6.17. Table 6.9, Table 6.10, and Table 6.11 illustrate the detailed grid search results of ρ_1^* and ρ_2^* in Step 1 and Step 2, respectively. The best set of Pr, R, F, FPR, and $d_{\rho 1}$ are highlighted.

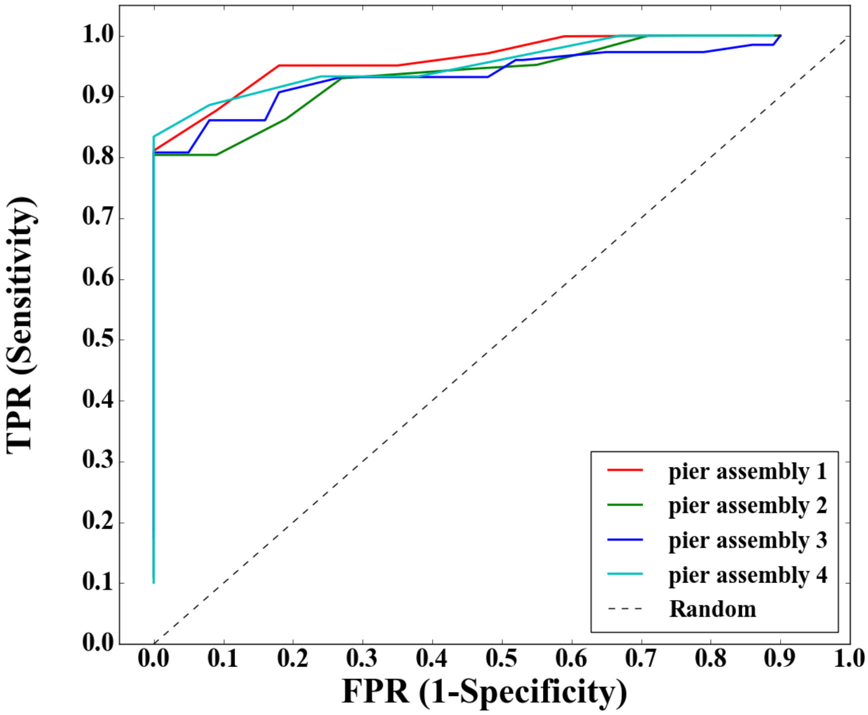


Figure 6.17 ROC of detector $range_{j\langle z\rangle}$ of pier assembly of all pier assembly samples

	Bridge 1					***Bridge 2***					***Bridge 3***					***Bridge 4***					***Bridge 5***				
$\boldsymbol{\rho 1}$	Pr	R	F1	FPR	$d_{\rho 1}$	Pr	R	F1	FPR	$d_{\rho 1}$	Pr	R	F1	FPR	$d_{\rho 1}$	Pr	R	F1	FPR	$d_{\rho 1}$	Pr	R	F1	FPR	$d_{\rho 1}$
0.10	0.52	1.00	0.69	0.86	0.86	-	-	-	-	-	-	-	-	-	-	0.33	1.00	0.49	0.87	0.87	-	-	-	-	-
0.15	0.53	1.00	0.69	0.85	0.85	0.30	1.00	0.46	0.98	0.98	0.25	1.00	0.40	0.92	0.92	0.35	1.00	0.52	0.78	0.78	-	-	-	-	-
0.20	0.57	0.96	0.72	0.69	0.69	0.75	0.68	0.71	0.09	0.34	0.26	1.00	0.41	0.90	0.90	1.00	0.61	0.50	0.35	0.53	0.29	1.00	0.45	0.93	0.93
0.25	0.98	0.84	0.91	0.01	0.16	1.00	0.68	0.81	0.00	0.32	0.91	0.76	0.83	0.02	0.24	1.00	0.60	0.75	0.00	0.40	0.99	0.86	0.92	0.00	0.14
0.30	1.00	0.84	0.91	0.00	0.16	1.00	0.68	0.81	0.00	0.32	1.00	0.76	0.86	0.00	0.24	1.00	0.60	0.75	0.00	0.40	0.99	0.86	0.92	0.00	0.14
0.35	1.00	0.31	0.47	0.00	0.69	1.00	0.67	0.78	0.00	0.33	1.00	0.43	0.60	0.00	0.57	1.00	0.60	0.75	0.00	0.40	0.99	0.86	0.92	0.00	0.14
0.40	1.00	0.31	0.47	0.00	0.69	1.00	0.55	0.71	0.00	0.45	1.00	0.43	0.60	0.00	0.57	1.00	0.60	0.75	0.00	0.40	0.99	0.86	0.92	0.00	0.14
0.45	1.00	0.31	0.47	0.00	0.69	1.00	0.55	0.71	0.00	0.45	1.00	0.43	0.60	0.00	0.57	1.00	0.60	0.75	0.00	0.40	0.99	0.86	0.92	0.00	0.14
0.50	1.00	0.31	0.47	0.00	0.69	1.00	0.55	0.71	0.00	0.45	1.00	0.43	0.60	0.00	0.57	1.00	0.60	0.75	0.00	0.40	0.99	0.86	0.92	0.00	0.14
0.55	1.00	0.31	0.47	0.00	0.69	1.00	0.55	0.71	0.00	0.45	1.00	0.43	0.60	0.00	0.57	1.00	0.60	0.75	0.00	0.40	0.99	0.86	0.92	0.00	0.14
0.60	1.00	0.29	0.44	0.00	0.71	1.00	0.55	0.71	0.00	0.45	1.00	0.43	0.60	0.00	0.57	1.00	0.60	0.75	0.00	0.40	0.99	0.86	0.92	0.00	0.14
0.65	1.00	0.29	0.44	0.00	0.71	1.00	0.55	0.71	0.00	0.45	1.00	0.43	0.60	0.00	0.57	1.00	0.60	0.75	0.00	0.40	0.99	0.86	0.92	0.00	0.14
0.70	1.00	0.29	0.44	0.00	0.71	1.00	0.55	0.71	0.00	0.45	1.00	0.43	0.60	0.00	0.57	1.00	0.60	0.75	0.00	0.40	0.99	0.86	0.92	0.00	0.14
0.75	1.00	0.29	0.44	0.00	0.71	1.00	0.55	0.71	0.00	0.45	1.00	0.43	0.60	0.00	0.57	1.00	0.60	0.75	0.00	0.40	0.99	0.86	0.92	0.00	0.14
0.80	1.00	0.29	0.44	0.00	0.71	1.00	0.54	0.71	0.00	0.46	1.00	0.43	0.60	0.00	0.57	1.00	0.60	0.75	0.00	0.40	1.00	0.83	0.91	0.00	0.17
0.85	1.00	0.29	0.44	0.00	0.71	1.00	0.42	0.59	0.00	0.58	1.00	0.43	0.60	0.00	0.57	1.00	0.60	0.75	0.00	0.40	1.00	0.78	0.87	0.00	0.22
0.90	1.00	0.10	0.19	0.00	0.90	1.00	0.42	0.59	0.00	0.58	1.00	0.43	0.60	0.00	0.57	1.00	0.50	0.66	0.00	0.50	1.00	0.44	0.62	0.00	0.56
0.95	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 6.9 Grid search results of ρ_1^* (*Bridge 1-Bridge 5*)

	Bridge 6					***Bridge 7***					***Bridge 8***					***Bridge 9***					***Bridge 10***				
$\boldsymbol{\rho 1}$	Pr	R	F1	FPR	$d_{\rho 1}$	Pr	R	F1	FPR	$d_{\rho 1}$	Pr	R	F1	FPR	$d_{\rho 1}$	Pr	R	F1	FPR	$d_{\rho 1}$	Pr	R	F1	FPR	$d_{\rho 1}$
0.10	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
0.15	0.28	1.00	0.44	0.91	0.91	-	-	-	-	-	0.35	1.00	0.52	0.95	0.95	-	-	-	-	-	-	-	-	-	-
0.20	0.37	0.50	0.43	0.30	0.68	0.09	0.99	0.17	0.54	0.54	1.00	0.87	0.93	0.00	0.13	0.31	0.91	0.46	0.91	0.91	0.09	0.88	0.17	0.89	0.90
0.25	1.00	0.50	0.67	0.00	0.50	0.09	0.98	0.17	0.54	0.54	1.00	0.87	0.93	0.00	0.13	0.95	0.76	0.85	0.02	0.24	0.59	0.58	0.58	0.04	0.42
0.30	1.00	0.50	0.67	0.00	0.50	1.00	0.91	0.95	0.00	0.09	1.00	0.83	0.91	0.00	0.17	1.00	0.76	0.87	0.00	0.24	0.97	0.58	0.73	0.00	0.42
0.35	1.00	0.50	0.67	0.00	0.50	1.00	0.91	0.95	0.00	0.09	1.00	0.83	0.91	0.00	0.17	1.00	0.76	0.87	0.00	0.24	0.97	0.58	0.73	0.00	0.42
0.40	1.00	0.50	0.67	0.00	0.50	1.00	0.91	0.95	0.00	0.09	1.00	0.83	0.91	0.00	0.17	1.00	0.76	0.87	0.00	0.24	0.97	0.58	0.73	0.00	0.42
0.45	1.00	0.50	0.67	0.00	0.50	1.00	0.91	0.95	0.00	0.09	1.00	0.83	0.91	0.00	0.17	1.00	0.76	0.87	0.00	0.24	0.97	0.58	0.73	0.00	0.42
0.50	1.00	0.50	0.67	0.00	0.50	1.00	0.85	0.95	0.00	0.15	1.00	0.72	0.84	0.00	0.28	1.00	0.76	0.87	0.00	0.24	0.97	0.58	0.73	0.00	0.42
0.55	1.00	0.50	0.67	0.00	0.50	1.00	0.85	0.95	0.00	0.15	1.00	0.72	0.84	0.00	0.28	1.00	0.76	0.87	0.00	0.24	0.97	0.58	0.73	0.00	0.42
0.60	1.00	0.50	0.67	0.00	0.50	1.00	0.45	0.62	0.00	0.55	1.00	0.72	0.84	0.00	0.28	1.00	0.76	0.87	0.00	0.24	0.97	0.58	0.73	0.00	0.42
0.65	1.00	0.50	0.67	0.00	0.50	1.00	0.45	0.62	0.00	0.55	1.00	0.72	0.84	0.00	0.28	1.00	0.76	0.87	0.00	0.24	0.97	0.58	0.73	0.00	0.42
0.70	1.00	0.50	0.67	0.00	0.50	1.00	0.45	0.62	0.00	0.55	1.00	0.72	0.84	0.00	0.28	1.00	0.76	0.87	0.00	0.28	0.97	0.58	0.73	0.00	0.42
0.75	1.00	0.50	0.67	0.00	0.50	1.00	0.45	0.62	0.00	0.55	1.00	0.72	0.84	0.00	0.28	1.00	0.76	0.87	0.00	0.24	0.97	0.58	0.73	0.00	0.42
0.80	1.00	0.50	0.67	0.00	0.50	1.00	0.45	0.62	0.00	0.55	1.00	0.72	0.84	0.00	0.28	1.00	0.76	0.87	0.00	0.24	0.97	0.58	0.73	0.00	0.42
0.85	1.00	0.50	0.67	0.00	0.50	1.00	0.39	0.56	0.00	0.61	1.00	0.72	0.84	0.00	0.28	1.00	0.72	0.87	0.00	0.28	0.97	0.58	0.73	0.00	0.42
0.90	1.00	0.18	0.30	0.00	0.82	1.00	0.39	0.56	0.00	0.61	1.00	0.24	0.38	0.00	0.76	1.00	0.52	0.69	0.00	0.48	0.97	0.58	0.73	0.00	0.42
0.95	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1.00	0.39	0.56	0.00	0.61	0.75	0.47	0.58	0.02	0.53

Table 6.10 Grid search results of $\boldsymbol{\rho_1^*}$ (*Bridge 6-Bridge 10*)

	Pier assembly 1					***Pier assembly 2***					***Pier assembly 3***					***Pier assembly 4***				
$\boldsymbol{\rho 2}$	Pr	R	F1	FPR	$d_{\rho 2}$	Pr	R	F1	FPR	$d_{\rho 2}$	Pr	R	F1	FPR	$d_{\rho 2}$	Pr	R	F1	FPR	$d_{\rho 2}$
0.10	0.47	1.00	0.64	0.90	0.90	0.41	1.00	0.58	0.86	0.86	-	-	-	-	-	-	-	-	-	-
0.15	0.48	1.00	0.65	0.85	0.85	0.43	1.00	0.60	0.80	0.80	0.45	0.99	0.62	0.89	0.89	-	-	-	-	-
0.20	0.48	1.00	0.65	0.86	0.86	0.43	1.00	0.60	0.79	0.79	0.45	0.99	0.62	0.89	0.89	-	-	-	-	-
0.25	0.50	1.00	0.67	0.78	0.78	0.46	1.00	0.63	0.71	0.71	0.46	0.99	0.63	0.86	0.86	0.49	1.00	0.66	0.87	0.87
0.28	0.50	1.00	0.67	0.78	0.78	0.46	1.00	0.63	0.71	0.71	0.46	0.99	0.63	0.86	0.86	0.49	1.00	0.65	0.89	0.89
0.30	0.50	1.00	0.67	0.78	0.78	0.46	1.00	0.63	0.71	0.71	0.46	0.99	0.63	0.86	0.86	0.49	1.00	0.66	0.87	0.87
0.32	0.57	1.00	0.73	0.59	0.59	0.73	0.86	0.79	0.19	0.23	0.48	0.97	0.64	0.79	0.79	0.50	1.00	0.67	0.85	0.85
0.34	0.89	0.88	0.88	0.09	0.15	1.00	0.80	0.89	0.00	0.20	0.88	0.86	0.87	0.08	0.16	0.54	1.00	0.70	0.72	0.72
0.35	1.00	0.81	0.90	0.00	0.19	1.00	0.80	0.89	0.00	0.20	0.92	0.81	0.86	0.05	0.20	0.54	1.00	0.70	0.72	0.72
0.40	1.00	0.81	0.90	0.00	0.19	1.00	0.80	0.89	0.00	0.20	0.99	0.81	0.89	0.00	0.19	1.00	0.83	0.91	0.00	0.17
0.50	1.00	0.77	0.87	0.00	0.23	1.00	0.80	0.89	0.00	0.20	0.99	0.77	0.87	0.00	0.23	1.00	0.83	0.91	0.00	0.17
0.60	1.00	0.77	0.87	0.00	0.23	1.00	0.80	0.89	0.00	0.20	0.99	0.77	0.87	0.00	0.23	1.00	0.83	0.91	0.00	0.17
0.70	1.00	0.74	0.85	0.00	0.26	1.00	0.72	0.84	0.00	0.28	0.99	0.73	0.84	0.00	0.27	1.00	0.83	0.91	0.00	0.17
0.80	1.00	0.74	0.85	0.00	0.26	1.00	0.72	0.84	0.00	0.28	0.99	0.73	0.84	0.00	0.27	1.00	0.83	0.91	0.00	0.17
0.90	1.00	0.74	0.85	0.00	0.26	1.00	0.72	0.84	0.00	0.28	0.99	0.73	0.84	0.00	0.27	1.00	0.83	0.91	0.00	0.17
0.95	1.00	0.69	0.81	0.00	0.31	1.00	0.56	0.72	0.00	0.44	1.00	0.71	0.83	0.00	0.29	1.00	0.83	0.91	0.00	0.17
0.96	1.00	0.69	0.81	0.00	0.31	1.00	0.24	0.39	0.00	0.76	1.00	0.55	0.71	0.00	0.45	1.00	0.83	0.91	0.00	0.17
0.98	1.00	0.19	0.32	0.00	0.81	1.00	0.11	0.20	0.00	0.89	1.00	0.17	0.30	0.00	0.83	0.98	0.34	0.50	0.01	0.66

Table 6.11 Grid search results of ρ_2^* (pier assembly samples)

The author then estimated the value of ρ_{3a}, which is the slab thickness ratio estimation used to remove the upper slab surface points from the pier area(s) $\{\beta_{mp}\}$ (Chapter 4, Section 4.2.4). The pier area(s) $\{\beta_{mp}\}$ are relatively small regions compared to an entire bridge. It is reasonable to consider the vertical elevation of $\{\beta_{mp}\}$ constant so that the probability of its point density can be presented. Suppose the distribution of the points along Z-axis of $\{\beta_{mp}\}$ is a collection of probabilities of the locations on the pier area surface where the points are located. The sum of all the probabilities in the distribution must be equal to 1. The density estimations of the pier area sample from slab bridges showed an obvious void space between the top surface and the bottom surface of a slab (Figure 6.18 (a)). The method estimated the 95% CI for the normalized slab bottom level was 0.84±0.06. Likewise, the method computed the density estimations of all the pier area samples from beam-slab bridges (Figure 6.18 (b)). This value was estimated to be 0.76±0.04. These statistics suggested that for slab and beam-slab bridges, the top 10% β_{mp} points include the upper slab surface so as they can be removed and classified into the deck assembly. This slab thickness ratio ρ_{3a} is especially used for beam-slab bridges in Step 4 for girder segmentation. Note that only one bridge from the ten bridge datasets is beam-slab bridge (i.e. *Bridge 7*). The author thus added one additional beam-slab for statistics estimation (i.e. *Bridge 11*). This bridge was used for algorithm development and parameter estimation. The author chose the slab bottom level estimation of beam-slab bridge to estimate the ρ_{3a} as 0.28 and 0.2 (1-(0.76±0.04)). Taking into account the shallow girder and the effect of the transverse gradient, ρ_{3a} was set to be 0.2 so that then, $\rho_4 = \frac{\rho_1^* - \rho_{3a}}{\rho_1^*} \approx 0.33$.

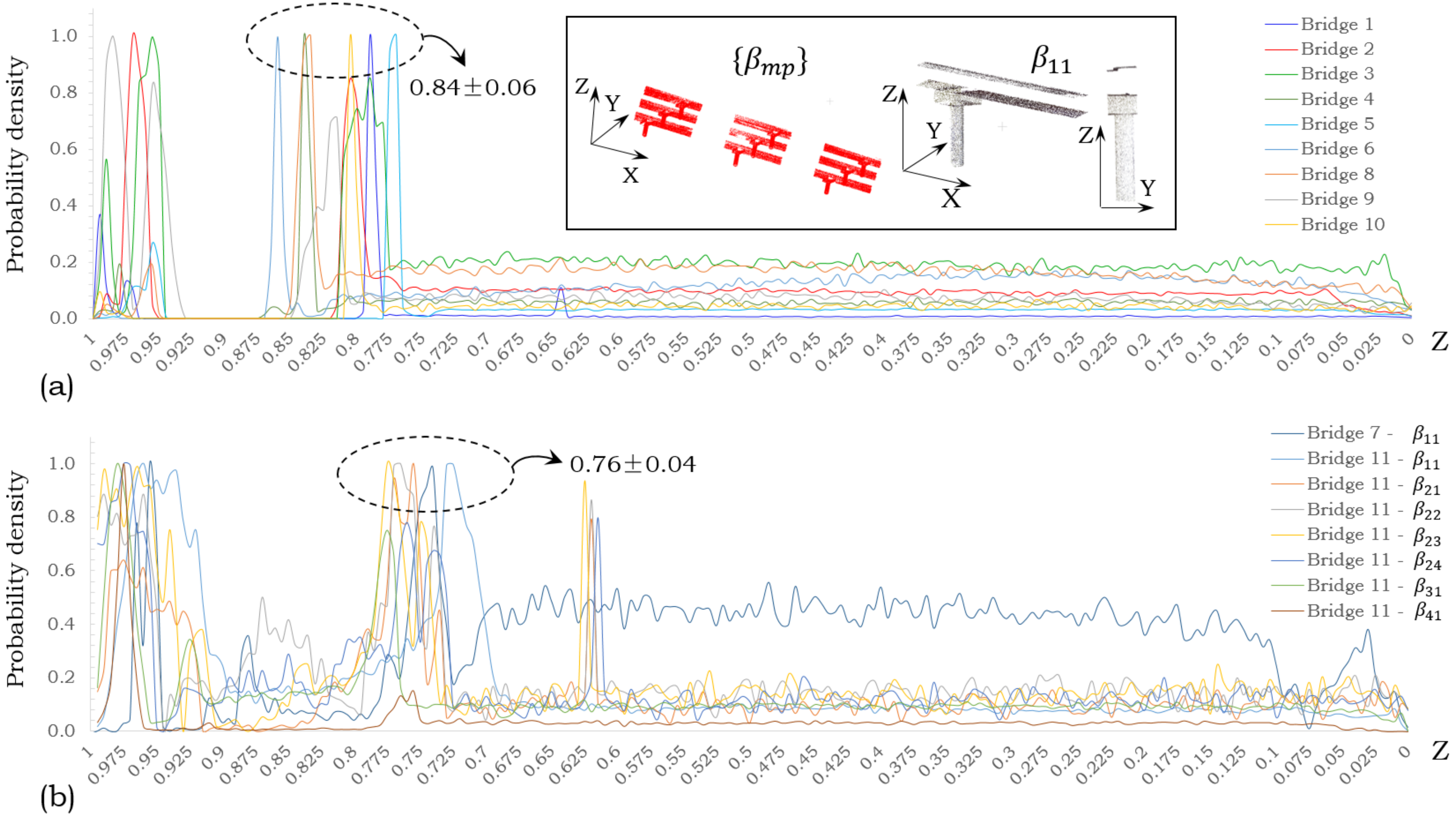


Figure 6.18 Density estimations of pier area samples of (a) slab bridges (e.g. β_{11}) and (b) beam-slab bridges

6.2.4.2. Method validation and Results

The author evaluated the proposed method of the prototype on the level of structural components with the optimal hyper-parameters identified in the previous section. Then the author compared the results against *GT C* (Section 6.2.2.2). The method generated an oriented-bounding-box for each segmented point cluster (hereafter AutoBBox) and assigned a semantic instance label to each point.

The author first compared AutoBBoxes against GTBBoxes and evaluated the proposed method's performance using the following three criteria. For a specific point cluster generated from the proposed method, let C_{auto} and C_{gt} be the centers of its AutoBBox and its GTBBox (if it exists), respectively, and $d(C_{auto}, C_{gt})$ be the Euclidean distance between C_{auto} and C_{gt}.

C1. GTBBox of the specific point cluster exists;

C2. C_{auto} is inside the corresponding GTBBox;

C3. $\varepsilon = \frac{d(C_{auto}, C_{gt})}{min(l_{gt}, w_{gt}, h_{gt})}$ <50%, where l_{gt}, w_{gt}, h_{gt} are the length, width and height of the GTBBox of the point cluster, respectively.

The point cluster is correctly detected by the AutoBBox and was assigned one to TP if all the above three conditions are satisfied; one to FP if C1 is false but an AutoBBox is generated; one to FN if C1 is true but at least one of C2 and C3 is not satisfied. The Pr, R, and F1 were generated using the values of TP, FN and FP, where

$$\mathrm{Pr}_{\mathrm{bridge_j}} = \frac{\text{\# of correctly detected point clusters}}{\text{total \# of AutoBBoxes for a bridge PCD}}, \quad \text{(Eq. 6.12)}$$

$$\mathrm{R}_{\mathrm{bridge\_j}} = \frac{\text{\# of correctly detected point clusters}}{\text{total \# of GTBBoxes for a bridge PCD}}, \quad \text{(Eq. 6.13)}$$

$$\mathrm{F1}_{\mathrm{bridge\_j}} = 2 * \frac{\mathrm{Pr}_{\mathrm{bridge\_j}} * \mathrm{R}_{\mathrm{bridge\_j}}}{\mathrm{Pr}_{\mathrm{bridge\_j}} + \mathrm{R}_{\mathrm{bridge\_j}}}, \quad \text{(Eq. 6.14)}$$

for $\mathrm{j} \in \{1,2,3 \ldots ,9,10\}$. Table 6.12 illustrates the point-wise and bounding-box-wise detection results. As shown clearly, Bridge 9 slab contains obvious skew. The average Pr, R and F1 of bounding-box-wise component detection for all ten bridges were 100%, 98.5%, and 99.2% (Table 6.13), respectively. All of the components were correctly detected except the point cluster of pierCap 1 ($\boldsymbol{\varepsilon}$=71.9%) and pierCap 2 (ε=81.9%) of Bridge 1 (Table 6.15). Yet, few points in these clusters were detected as FP ($\mathrm{FDR}_{\mathrm{pierCap1}}$=4.4%, $\mathrm{FDR}_{\mathrm{pierCap2}}$=8.6%), where the false discovery rate (FDR) for each point cluster is computed as:

$$\mathrm{FDR_s} = \frac{\mathrm{FP_s}}{\mathrm{FP_s} + \mathrm{TP_s}}. \qquad \text{(Eq. 6.15)}$$

Therefore, although bounding-box-wise metrics can give a general picture of the proposed performance, they are too sensitive to the locations of misclassified points, which largely affected the values of $d(C_{auto}, C_{gt})$.

The author repeated the system evaluation with point-wise metrics: Eq. 6.11 – Eq. 6.13. Herein, the "S" in Eq. 6.15 – Eq. 6.18 refers to any specific final point cluster generated from the proposed solution. For a specific bridge point cloud, the author computed the micro-average scores. In micro-average, the author summed up individual values of TP, FP and FN from all point clusters to obtain the statistics:

$$\mathrm{Pr_{micro}} = \frac{\sum_{s=1}^{|S|} \mathrm{TP_s}}{\sum_{s=1}^{|S|} \mathrm{TP_s} + \sum_{s=1}^{|S|} \mathrm{FP_s}}, \qquad \text{(Eq. 6.16)}$$

$$\mathrm{R_{micro}} = \frac{\sum_{s=1}^{|S|} \mathrm{TP_s}}{\sum_{s=1}^{|S|} \mathrm{TP_s} + \sum_{s=1}^{|S|} \mathrm{FN_s}}, \qquad \text{(Eq. 6.17)}$$

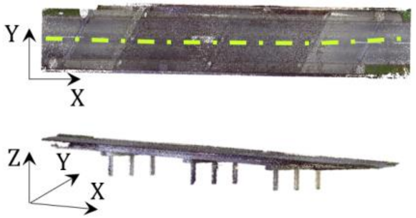
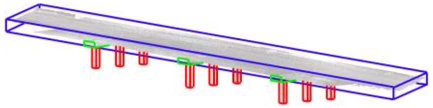
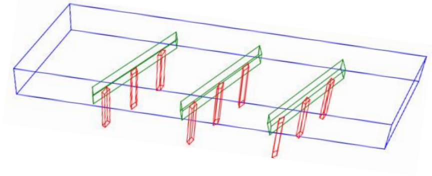
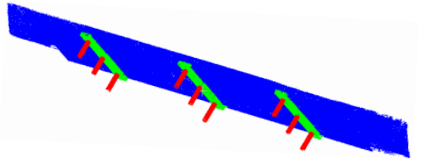
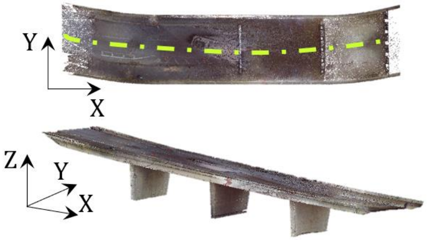
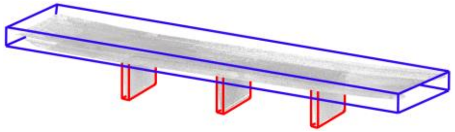
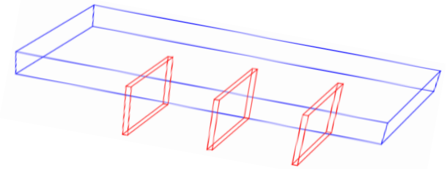
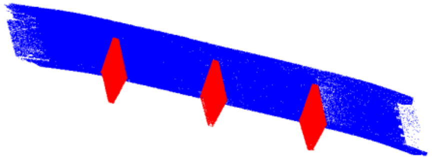
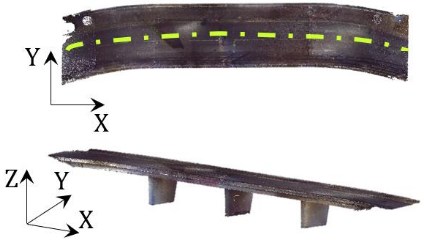
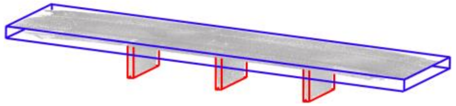
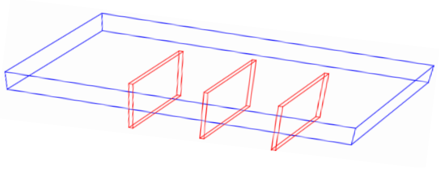
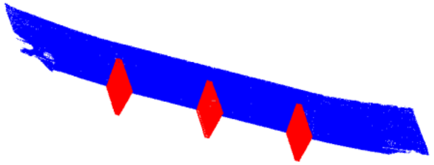
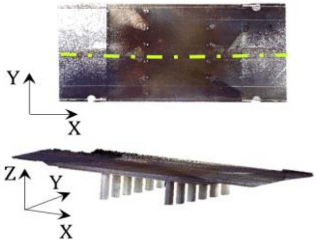
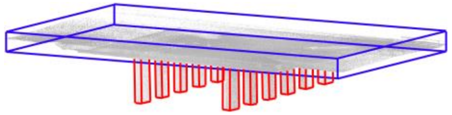
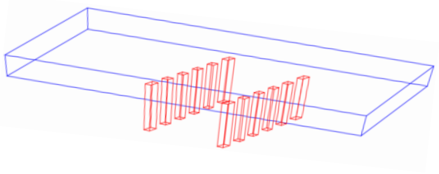
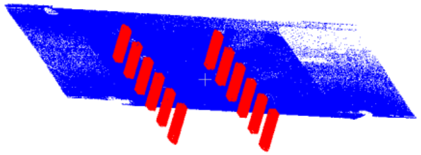
where |s| is the number of generated point clusters in this given bridge point cloud. The micro-average F1-score is simply the harmonic mean of $\mathrm{Pr_{micro}}$ and $\mathrm{R_{micro}}$. In macro-average, the author took the average of the Precision and Recall of all point clusters:

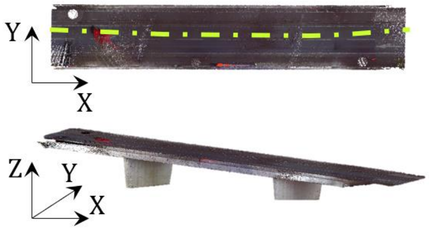
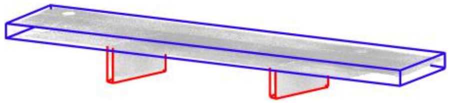
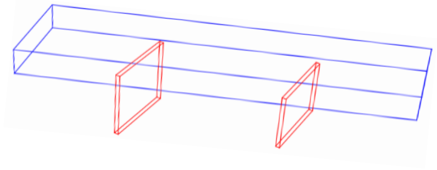
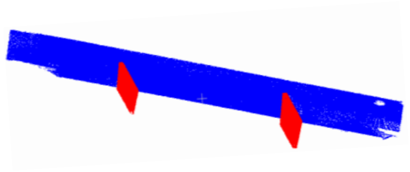
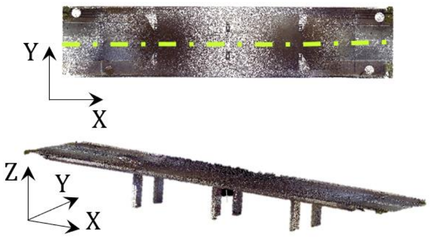
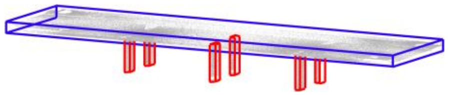
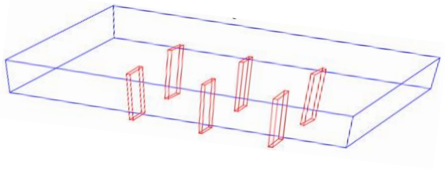
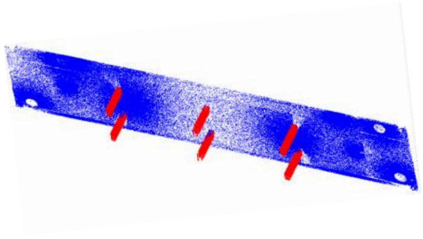
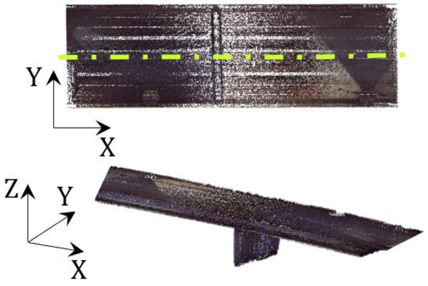
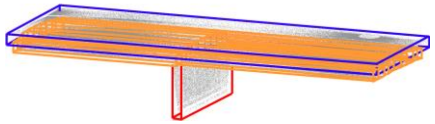
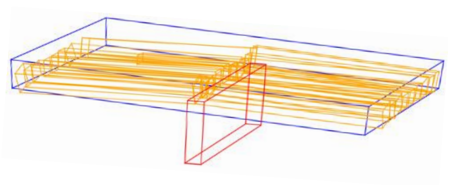
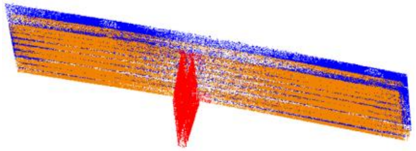
$$\mathrm{Pr_{macro}} = \frac{\sum_{s=1}^{|S|} \mathrm{Pr_s}}{|S|}, \qquad \text{(Eq. 6.18)}$$

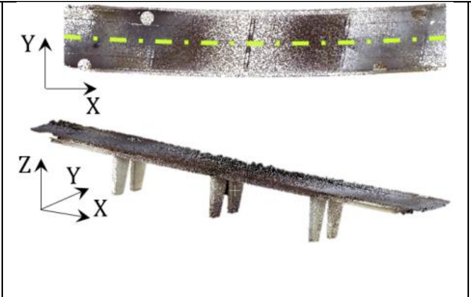
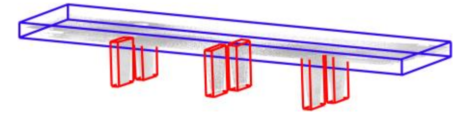
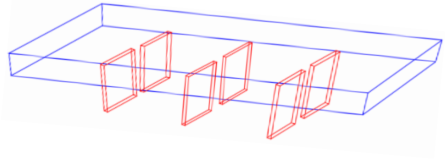
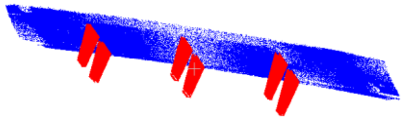
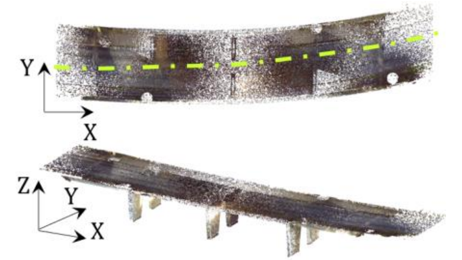
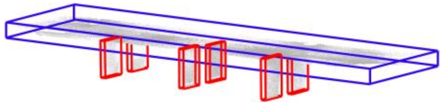
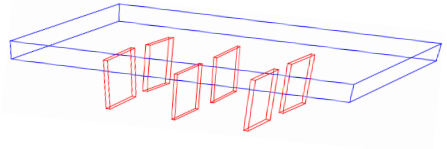
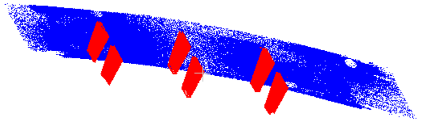
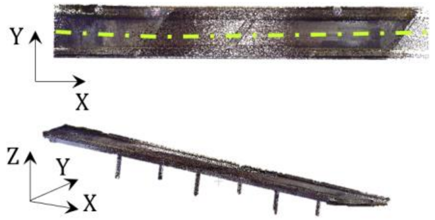
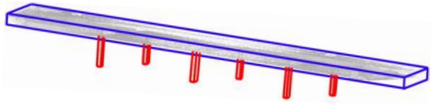
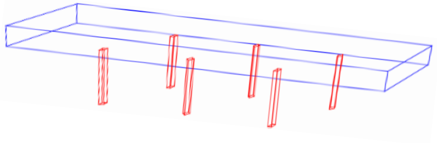
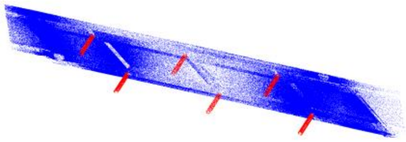
$$\mathrm{R}_{\mathrm{macro}} = \frac{\sum_{s=1}^{|S|} \mathrm{R}_{s}}{|S|}. \qquad \text{(Eq. 6.19)}$$

Likewise, the macro-average F1 is the harmonic mean of $\mathrm{Pr}_{\mathrm{macro}}$ and $\mathrm{R}_{\mathrm{macro}}$. Assumptions A3a – A3c were justified as the method recognizes that there was no pier cap in the pier assemblies of *Bridge 2, 3, 5 and 7* (wall-type-pier) and *Bridge 4, 6, 8, 9* and 10 (multiple columns without cap). The method correctly identified the pier caps in *Bridge 1*.

The author summarized the point-wise micro- and macro-average evaluation results in Table 6.14. On average, the proposed method achieved remarkable performance: the micro-average of P/R/F1 was 98.4% (for a multi-class case, the micro-averages option yields results in a mathematically equivalent definition for precision and recall, thus equivalent F1-score). Among these, the highest was rounded up to 100% and the lowest was 89.1%. The macro-average ones were 99.2%, 97.3% and 98.1%, respectively.

	Input	GTBBox	AutoBBox	Detection Results
Bridge 1				
Bridge 2				
Bridge 3				
Bridge 4				

Bridge 5				
Bridge 6				
Bridge 7				

Bridge 8				
Bridge 9				
Bridge 10				

🟦 ☐ slab 🟥 ☐ pier 🟩 ☐ pier cap 🟧 ☐ girder

Table 6.12 Detection results and GTBBoxes & AutoBBoxes of point clusters

Bridge	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***	**Avg.**
FN	2	0	0	0	0	0	0	0	0	0	
FP	0	0	0	0	0	0	0	0	0	0	
TP	11	4	4	13	3	7	20	7	7	7	
Pr	100%	100%	100%	100%	100%	100%	100%	100%	100%	100%	**100%**
R	84.6%	100%	100%	100%	100%	100%	100%	100%	100%	100%	**98.5%**
F1	91.7%	100%	100%	100%	100%	100%	100%	100%	100%	100%	**99.2%**

Table 6.13 Bounding-box-wise component detection performance

Bridge	**#TP**	**#FP**	$\mathbf{P_{r_{macro}}}$	$\mathbf{R_{macro}}$	$\mathbf{F1_{macro}}$
1	485960	2493	98.6%	98.5%	98.5%
2	495161	4839	99.7%	96.9%	98.2%
3	496514	3486	99.8%	97.0%	98.4%
4	498359	220	99.9%	99.9%	99.9%
5	496744	3256	99.7%	97.4%	98.5%
6	498491	1509	100%	98.3%	99.1%
7	405573	49412	94.3%	89.2%	90.8%
8	497649	2351	100%	98.7%	99.3%
9	497223	2777	100%	98.1%	99.0%
10	874802	234	100%	99.1%	99.5%
Avg.			**99.2%**	**97.3%**	**98.1%**

Table 6.14 Point-wise component detection performance

Table 6.15 Detailed point-wise performance evaluation results of ten bridge datasets

	Segment	#GT	#Auto	#TP	#FN	#FP	Pr	R	F1	FDR (%)	$d(C_{auto}, C_{gt})$	ε (%)
Bridge 1	slab	423526	422496	422048	1478	448	99.9%	99.7%	99.8%	0.11	0.00	0.0
	pierCap 1	12780	13113	12541	239	572	95.6%	98.1%	96.9%	4.36	0.65	71.9
	pierCap 2	8547	9347	8547	0	800	91.4%	100%	95.5%	8.56	0.72	81.9
	pierCap 3	11878	12342	11669	209	673	94.5%	98.2%	96.4%	5.45	0.16	17.6
	pier 11	3687	3650	3650	37	0	100%	99.0%	99.5%	0	0.01	1.8
	pier 12	4159	4107	4107	52	0	100%	98.7%	99.4%	0	0.02	2.9
	pier 13	3722	3668	3668	54	0	100%	98.5%	99.3%	0	0.03	3.1
	pier 21	3068	3035	3035	33	0	100%	98.9%	99.5%	0	0.02	2.1
	pier 22	2893	2870	2870	23	0	100%	99.2%	99.6%	0	0.02	2.8
	pier 23	3437	3382	3382	55	0	100%	98.4%	99.2%	0	0.02	2.9
	pier 31	3470	3450	3450	20	0	100%	99.4%	99.7%	0	0.01	0.9
	pier 32	3719	3471	3471	248	0	100%	93.3%	96.6%	0	0.13	15.7
	pier 33	3567	3522	3522	45	0	100%	98.7%	99.4%	0	0.03	3.4
	micro-avg	488453	488453	485960	2493	2493	**99.5%**	**99.5%**	**99.5%**			
Bridge 2	slab	388527	393358	388526	2	844	99.8%	100%	99.9%	1.23	0.09	5.2
	pier 11	40574	37937	37935	2639	0	100%	99.1%	99.6%	0.01	0.15	17.1
	pier 21	32378	31806	31804	574	0	100%	99.4%	99.7%	0.01	0.07	7.8
	pier 31	38521	36899	36897	1624	1	100%	99.2%	99.6%	0.01	0.14	15.3
	micro-avg	500000	500000	495161	4839	4839	**99.0%**	**99.0%**	**99.0%**			
Bridge 3	slab	413961	417439	413959	2	3480	99.2%	100%	99.6%	0.83	0.04	2.4
	pier 11	30230	28875	28873	1357	2	100%	95.5%	97.7%	0.01	0.11	12.2
	pier 21	25193	24283	24281	912	2	100%	96.4%	98.2%	0.01	0.06	7.2
	pier 31	30616	29403	29401	1215	2	100%	96.0%	98.0%	0.01	0.09	10.1
	micro-avg	500000	500000	496514	3486	3486	**99.3%**	**99.3%**	**99.3%**			

Bridge 4	slab	394609	394641	394527	82	114	100%	100%	100%	0.03	0.09	3.4
	pier 11	9309	9330	9307	2	23	99.8%	100%	99.9%	0.25	0.09	8.1
	pier 12	9233	9225	9224	9	1	100%	100%	99.9%	0.01	0.02	1.5
	pier 13	9199	9200	9196	3	4	100%	100%	100%	0.04	0.00	0.1
	pier 14	9059	9063	9057	2	6	100%	100%	100%	0.07	0.00	0.2
	pier 15	8339	8330	8328	11	2	100%	100%	100%	0.02	0.01	1.1
	pier 16	8381	8404	8379	2	25	100%	100%	99.8%	0.30	0.01	1.2
	pier 21	9096	9121	9094	2	27	100%	100%	100%	0.30	0.11	10.5
	pier 22	8601	8571	8569	32	2	100%	99.6%	99.8%	0.02	0.06	4.4
	pier 23	8142	8137	8135	7	2	100%	100%	99.9%	0.02	0.01	0.9
	pier 24	8354	8342	8340	14	2	100%	99.8%	99.9%	0.02	0.02	2.3
	pier 25	7824	7774	7772	52	2	100%	99%	99.7%	0.03	0.03	2.9
	pier 26	8433	8441	8431	2	10	100%	100%	99.9%	0.12	0.00	0.0
	micro-avg	498579	498579	498359	220	220	**100%**	**100%**	**100%**			
Bridge 5	slab	415505	418755	415503	2	3252	99.2%	100%	99.6%	0.78	0.07	3.3
	pier 11	44994	43546	43544	1450	2	100%	96.8%	98.4%	0	0.16	15.7
	pier 21	39501	37699	37697	1804	2	100%	95.4%	97.7%	0.01	0.14	14.5
	micro-avg	500000	500000	496744	3256	3256	**99.3%**	**99.3%**	**99.3%**			
Bridge 6	Slab	398271	399766	398269	2	1497	99.6%	100%	99.8%	0.37	0.06	3.8
	pier 11	24651	24169	24167	484	2	100%	98.0%	99.0%	0.01	0.13	16.2
	pier 12	19491	19391	19389	102	2	100%	99.5%	99.7%	0.01	0.04	5.2
	pier 21	6226	6002	6000	226	2	100%	96.4%	98.1%	0.03	0.15	19.8
	pier 22	5701	5518	5516	185	2	100%	96.8%	98.3%	0.04	0.15	19.5
	pier 31	24919	24793	24791	128	2	100%	99.5%	99.7%	0.01	0.05	6.2
	pier 32	20741	20361	20359	382	2	100%	98.2%	99.1%	0.01	0.16	17.4
	micro-avg	500000	500000	498491	1509	1509	**99.7%**	**99.7%**	**99.7%**			
Bridge 7	slab	187043	213807	176043	11000	37764	82.3%	94.3%	87.7%	17.7	0.50	37.0

	pier 11	20250	20821	20248	2	573	97.2%	100%	99.1%	2.75	0.04	3.7
	girder 11	7804	9757	7696	108	2061	78.9%	98.6%	87.9%	21.1	0.31	30.5
	girder 12	7324	7214	7211	113	3	100%	98.1%	99.0%	0.04	0.03	2.4
	girder 13	12875	9648	9645	3230	3	100%	74.9%	85.7%	0.03	0.18	13.2
	girder 14	20027	16029	16027	4000	2	100%	79.9%	88.8%	0.01	0.12	7.1
	girder 15	27968	23965	23963	4005	2	100%	85.7%	92.3%	0.01	0.16	13.3
	girder 16	14345	11632	11630	2715	2	100%	81.0%	89.5%	0.02	0.21	17.4
	girder 17	10245	9196	9194	1051	2	100%	89.6%	94.5%	0.02	0.18	12.6
	girder 18	7589	7560	7557	32	3	100%	99.3%	99.7%	0.04	0.09	12.3
	girder 19	9057	12436	8968	89	3468	72.1%	98.9%	83.8%	27.9	0.24	22.3
	girder 21	10857	14042	10773	84	3269	76.7%	98.5%	86.6%	23.3	0.23	21.3
	girder 22	9730	9636	9632	98	4	100%	98.8%	99.4%	0.04	0.02	1.1
	girder 23	14723	11214	11212	3511	2	100%	75.8%	86.3%	0.02	0.20	16.2
	girder 24	22325	16977	16974	5351	3	100%	76.0%	86.3%	0.02	0.07	3.9
	girder 25	31032	22667	22664	8368	3	100%	72.9%	84.3%	0.01	0.08	3.5
	girder 26	15997	11624	11620	4377	4	100%	72.6%	84.1%	0.03	0.12	10.9
	girder 27	10167	9138	9133	1034	5	100%	89.8%	94.6%	0.05	0.14	10.6
	girder 28	6898	6812	6803	95	9	100%	98.3%	99.1%	0.13	0.06	7.1
	girder 29	8729	10810	8580	149	2230	79.4%	95.5%	86.5%	20.6	0.28	27.9
	micro-avg	454985	454985	405573	49412	49412	**89.1%**	**89.1%**	**89.1%**			
Bridge 8	slab	350859	353202	350857	2	2345	99.3%	100%	99.7%	0.66	0.51	36.7
	pier 11	37438	37076	37075	363	1	100%	99.0%	99.5%	0	0.02	2.5
	pier 12	34057	33681	33680	377	1	100%	98.9%	99.4%	0	0.02	2.0
	pier 21	9933	9820	9819	114	1	100%	98.9%	99.4%	0.01	0.06	8.1
	pier 22	9444	9369	9368	76	1	100%	99.2%	99.6%	0.01	0.04	5.7
	pier 31	28273	27015	27014	1259	1	100%	95.5%	97.7%	0	0.05	5.6
	pier 32	29996	29837	29836	160	1	100%	99.5%	99.7%	0	0.04	4.5

	micro-avg	500000	500000	497649	2351	2351	**99.5%**	**99.5%**	**99.5%**			
Bridge 9	slab	367606	370369	367601	5	2768	99.3%	100%	99.6%	0.75	0.00	0.2
	pier 11	7955	7923	7774	181	1	100%	97.7%	98.8%	0.01	0.15	16.3
	pier 12	34584	34417	33423	1161	1	100%	96.6%	98.3%	0	0.08	8.1
	pier 21	12004	11963	11726	278	1	100%	97.7%	98.8%	0.01	0.17	25.2
	pier 22	10769	10721	10511	258	1	100%	97.6%	98.8%	0.01	0.17	25.0
	pier 31	41572	41785	41115	457	1	100%	99%	99.4%	0	0.13	14.6
	pier 32	25510	25513	25073	437	4	100%	98%	99%	0.02	0.18	20.3
	micro-avg	500000	500000	497223	2777	2777	**99.4%**	**99.4%**	**99.4%**			
Bridge 10	deck-slab	854710	854920	854700	10	220	100%	100%	100%	0.03	0.00	0.0
	pier 11	3990	3888	3887	103	1	100%	97.4%	98.7%	0.03	0.04	5.6
	pier 21	4822	4828	4820	2	8	99.8%	100%	99.9%	0.17	0.01	1.2
	pier 31	1317	1298	1297	20	1	100%	98.5%	99.2%	0.08	0.03	4.9
	pier 41	1182	1182	1180	2	2	100%	99.8%	99.8%	0.17	0.00	1.1
	pier 51	3319	3290	3289	30	1	100%	99.1%	99.5%	0.03	0.02	3.7
	pier 61	5696	5630	5629	67	1	100%	98.8%	99.4%	0.02	0.03	5.7
	micro-avg	875036	875036	874802	234	234	**100%**	**100%**	**100%**			

The processing time for each bridge point cloud was on average 8.33±1.45 minutes (for a point cloud with less than one million points) (Table 6.16), including all four major steps of the proposed method.

Bridge ID	**# point**	**Processing time (minutes)**
1	488453	9.67
2	500000	8.33
3	500000	7.33
4	498579	8.12
5	500000	8.83
6	500000	8.62
7	454985	8.13
8	500000	8.77
9	500000	8.53
10	875036	6.98
Avg.	531705	**8.33**

Table 6.16 Processing time of ten down-sampled bridge datasets

To learn how many occlusions are exactly acceptable, the author re-conducted experiments using *Bridge 1* by creating arbitrary occlusions in slab, pier caps, and piers, respectively while others remain unchanged. Then the author combined all these occlusions. The occlusion level was estimated to be 30—40%. Table 6.17 and Table 6.18 show that the method achieved high detection performance, despite the presence of large occlusions in the data. However, it is not encouraging to process such a high occlusion level data in real applications. In addition, 3D scans typically produce a non-uniform sampling on the surface of scanned objects, which can be due to the distance from the shape to the scanner position and the scanner orientation, as well as the geometric features of the shape. For instance, Figure 6.19 illustrates the non-uniformly distributed points on the $deck_2$ of *Bridge 8*. The experiment of the best projection search proved that the method is robust to very unevenly distributed points between the upper and lower slab. There could be some extremely non-uniformly-distributed-points scenarios that our method may not accommodate. However, the author believes that a carefully planned and elaborately designed scanning process could eliminate these cases. Specially targeted LS techniques or settings are required for these challenging regions.

The method of Laefer & Truong-Hong (2017) can be considered in this endeavour.

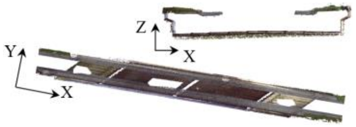
<table>
<tr><th colspan="3">Slab</th><th colspan="3">Pier caps</th><th colspan="3">Piers</th><th colspan="3">Combination</th></tr>
<tr><td colspan="3"> </td><td colspan="3"> </td><td colspan="3"> </td><td colspan="3"> </td></tr>
<tr><td colspan="2">#pt</td><td>321966</td><td colspan="2">#pt</td><td>25923</td><td colspan="2">#pt</td><td>21645</td><td colspan="2">#pt</td><td>369534</td></tr>
<tr><td>Pr</td><td>R</td><td>F1</td><td>Pr</td><td>R</td><td>F1</td><td>Pr</td><td>R</td><td>F1</td><td>Pr</td><td>R</td><td>F1</td></tr>
<tr><td>99.4%</td><td>97.1%</td><td>98.2%</td><td>92.6%</td><td>88.9%</td><td>90.7%</td><td>100%</td><td>98.4%</td><td>99.2%</td><td>96.6%</td><td>96.6%</td><td>96.6%</td></tr>
</table>

Table 6.17 Effect of large occlusions on Bridge 1

Segment	#GT	#Auto	#TP	#FN	#FP	Pr	R	F1
slab	321966	323306	312476	9490	10830	96.7%	97.1%	96.9%
pierCap 1	9899	9439	9043	856	396	95.8%	91.4%	93.5%
pierCap 2	6322	6532	5944	378	588	91.0%	94.0%	92.5%
pierCap 3	9702	9205	8835	867	370	96.0%	91.1%	93.5%
pier 11	2510	2520	2473	37	47	98.1%	98.5%	98.3%
pier 12	2731	2689	2673	58	16	99.4%	97.9%	98.6%
pier 13	2450	2413	2396	54	17	99.3%	97.8%	98.5%
pier 21	2230	2233	2197	33	36	98.4%	98.5%	98.5%
pier 22	2051	2033	2016	35	17	99.2%	98.3%	98.7%
pier 23	2411	2135	2110	301	25	98.8%	87.5%	92.8%
pier 31	2483	2491	2483	0	8	99.7%	100%	99.8%
pier 32	2455	2234	2209	246	25	98.9%	90.0%	94.2%
pier 33	2324	2304	2279	45	25	98.9%	98.1%	98.5%
micro-avg	369534	369534	354855	12400	12400	**96.6%**	**96.6%**	**96.6%**
macro-avg						**97.7%**	**95.4%**	**96.5%**

Table 6.18 Performance of the effect of large occlusions on Bridge 1 (combination)

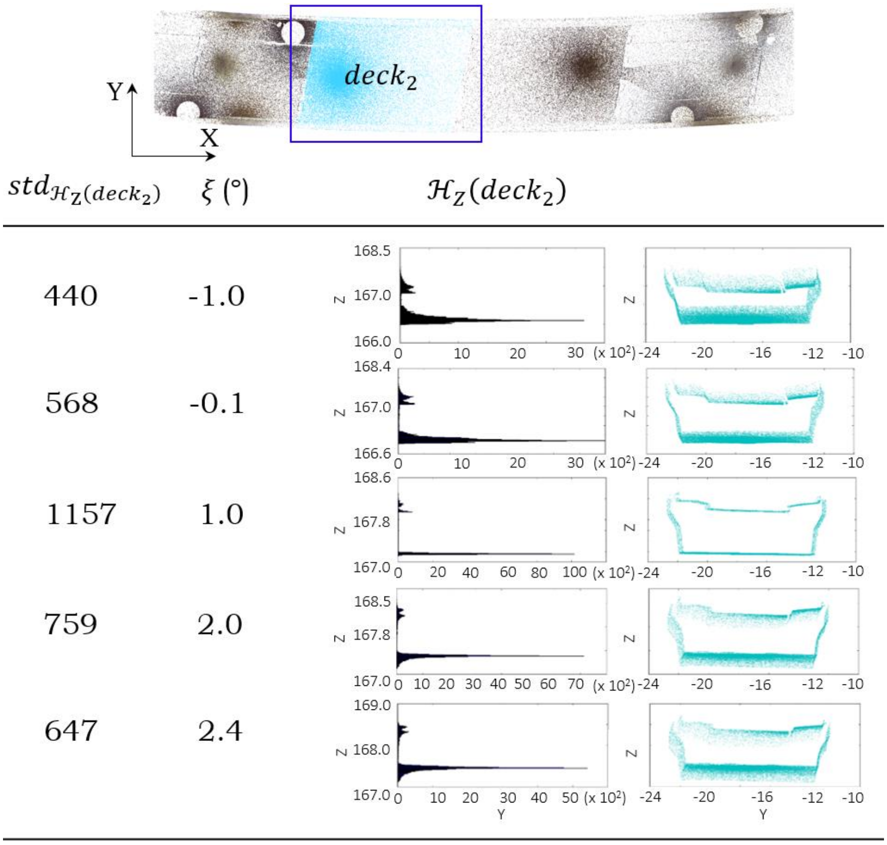


Figure 6.19 Best projection search when points are very unevenly distributed

6.2.4.3. Discussion

Although the method achieved high detection rates with the PCD of all ten RC bridges in both the bounding-box-wise and the point-wise assessments, the FDR of some point clusters revealed that the proportion of the FP points is not insignificant (Table 6.15). This is especially true for Bridge 7. There were a few components that reached very high detection precision, such as the pier (97.2%) and many girders, such as girder 14 (100%), girder 15 (100%), girder 22 (100%), and girder 23 (100%), among others. They contributed to maintain a good macro-average precision. This is technically true as across all point clusters, the macro-average precision was 94.3% (Table 6.14). However, some points were not properly classified. Normally, a slab cluster should contain the most populated labels in a bridge point cloud since it has the most points. For Bridge 7, the method had a misclassification for the slab point cluster (FP_{slab}=37764, FDR_{slab}=17.7%) (Table 6.15). The FDR was also not trivial for girder 11 (21.1%), girder 19 (27.9%), girder 21 (23.3%), and girder 29 (20.6%). The micro-average metrics adequately captured class imbalance issues and brought the overall precision average down to 89.1%.

There are two main reasons for the reduced classification performance in Bridge 7. First, the significant parabolic vertical alignment of the roadway in each deck segment of this bridge made the segmentation less accurate. Future work should develop a further deck-segment slicing procedure in Step 4 to alleviate the impact of parabolic curves. Second, the girders were placed so close to each other that it was difficult for a scan sensor to see the gaps between adjacent girders. Around 8% of the surface of Bridge 7 was occluded (Table 6.1) mainly due to the fact that the points on the webs of the girders were missing. As a result, these regions were quite ambiguous, making it difficult to detect an individual girder by the proposed method. The points between adjacent girders were misclassified as slab.

It is worth noting that our method is efficient for a certain type of RC bridge: the typical RC slab bridges and beam-slab bridges. The experiments proved that this method fills gap 1 (Chapter 2, Section 2.3) and can deal with some very common and important types of highway bridges. This method could likely be scaled up for more complicated bridges. Additional procedures that follow the same approach can be integrated into this method to detect other elements, such as abutments, bearings, handrails, etc. Future work can also be built on Step 4 to detect and segment smaller components in bridges with more complex superstructure geometries such as grid-beams and cross-beams. The method developed for reconstructing gridded steel structures (Gyetvai et al., 2018; Laefer & Truong-Hong, 2017) can be integrated.

In summary, the experimental results prove that the proposed method can detect four types of bridge objects in the form of point clusters in real PCD featuring defects (occlusions and varying point density). **Research question 1** has been addressed.

6.2.5. Experiments of IFC object fitting – Chapter 5

This section presents the experiments that aim to answer **research question 2** (Chapter 2, Section 2.4): How to reconstruct labelled point clusters in arbitrary shapes of real bridge PCD, into 3D solid models in IFC format? Same for **research question 3**: How to evaluate the accuracy of a bridge gDT reconstructed from a point cloud? The following text explains the design and execution of the experiments for the proposed IFC object fitting method presented in Chapter 5. The author elaborates how the experiments were done, what the results were, and why. In Section 6.2.5.1, the author explains the input data preparation followed by showing the runtime fitting results in Section 6.2.5.2. Then, in Section 6.2.5.3 and Section 6.2.5.4, the author elaborates how to evaluate the LOD 200 and LOD 250—300 gDTs generated by the proposed method, respectively.

6.2.5.1. Input data preparation

In order to test the hypothesis of the proposed IFC object fitting method, the author first prepared the data which serves as the input for the method. The inputs are the idealized outputs generated from Chapter 4, namely the refined labelled point clusters. The reason of refinement is that the labelled point clusters generated from the object detection step are imperfect, some FP points retained around boundaries between adjacent point clusters (Figure 6.20). These FP points will affect the effectiveness of the proposed IFC object fitting method. They may create incorrect OBB and concave hulls, so they should be removed before applying the proposed method. To do so, the author used the developed cropping function presented earlier in Section 6.2.2.1 to manually select the FP points of each point cluster and then, remove them. This process was repeatedly performed until FP points of all point clusters were removed.

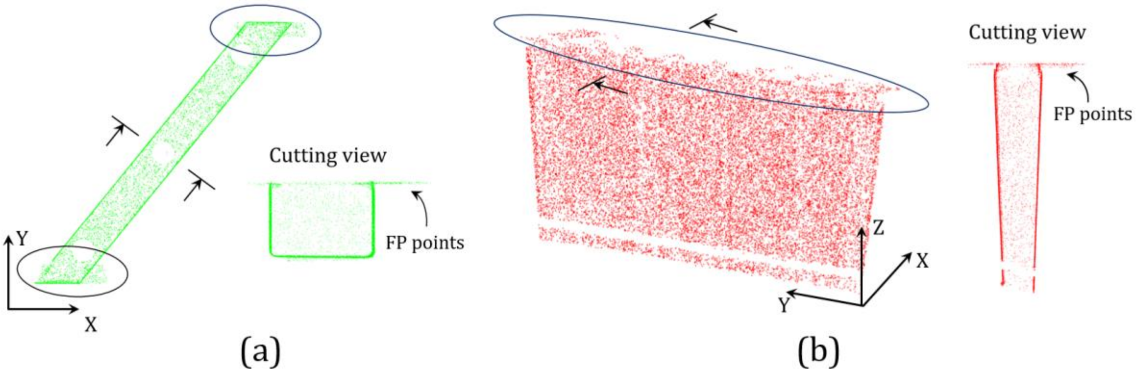


Figure 6.20 (a) A pier cap point cluster of *Bridge 1* and its FP points; (2) The pier point cluster of *Bridge 7* and its FP points

The proposed method was evaluated on the level of different resolutions, i.e. LOD. The author compared the automatically generated LOD 200 gDTs against *GT D* (Section 6.2.2.3, Table 6.4) and compared the automatically generated LOD 250—300 gDTs against *GT E* (Section 6.2.2.3, Table 6.5).

6.2.5.2. Results

For each bridge, the author manually prepared the input data for the proposed method. Table 6.19 and Table 6.20 illustrate the results of the LOD 200 and LOD 250—300 gDTs generated by the proposed method, respectively. For LOD 250—300, the number of deck slab slices and pier slices were both set to be 20. The value α in the 2D *ConcaveHull* α-shape algorithm was set to be 0.98.

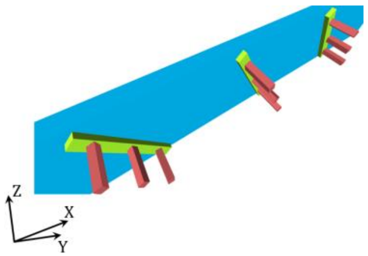
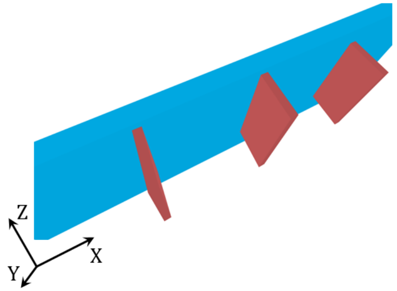
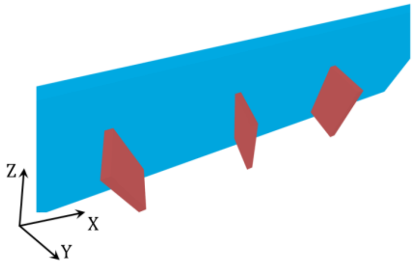
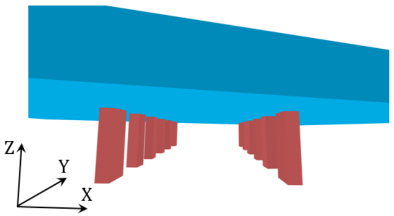
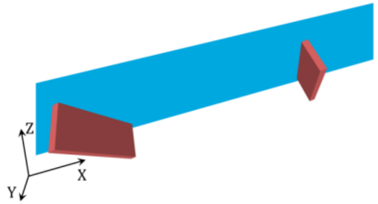
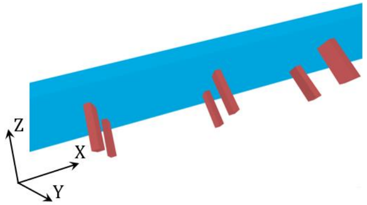
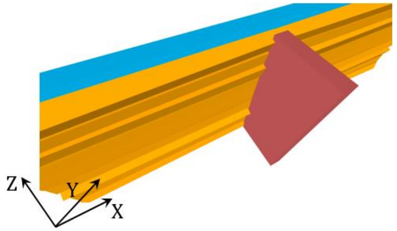
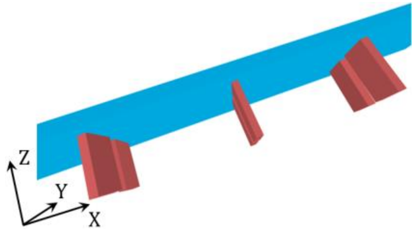
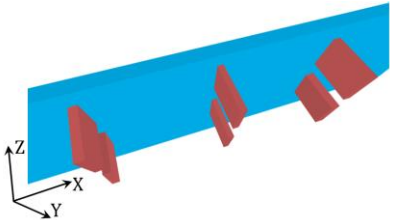
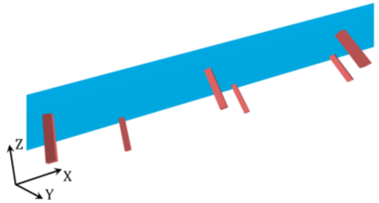
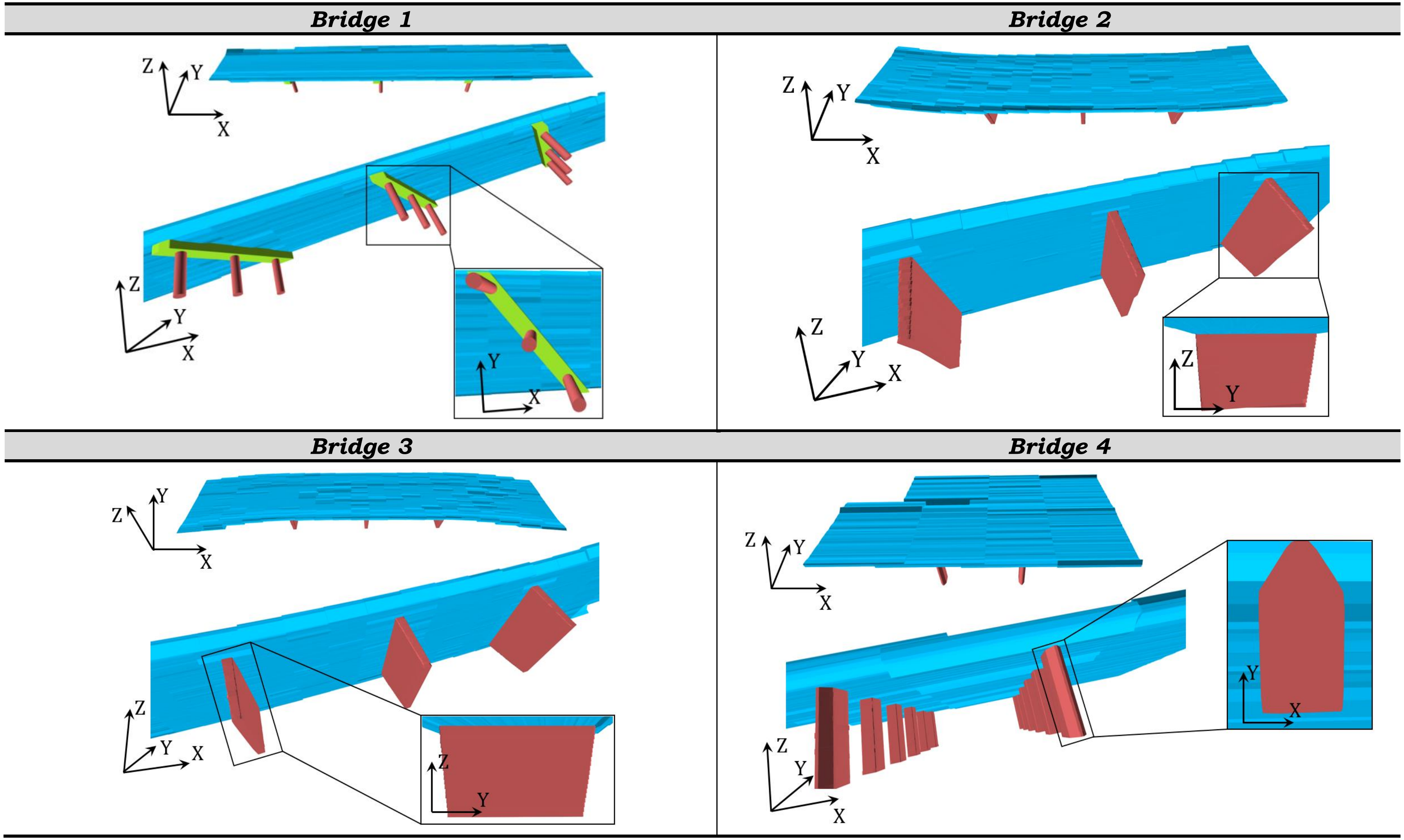
Bridge 1	Bridge 2	Bridge 3	Bridge 4	Bridge 5
				
Bridge 6	**Bridge 7**	**Bridge 8**	**Bridge 9**	**Bridge 10**
				

Table 6.19 LOD 200 bridge gDTs

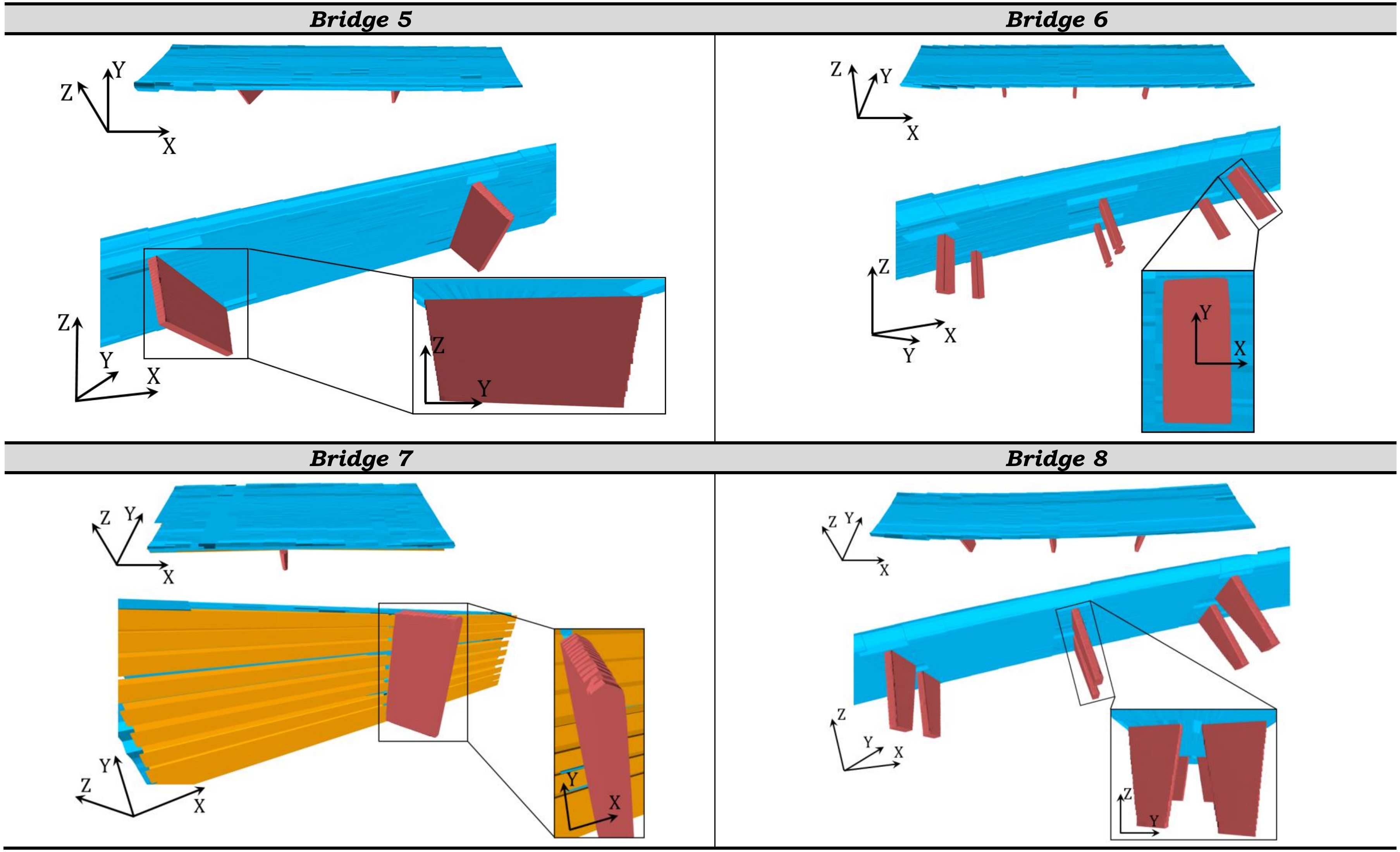


Bridge 1

Bridge 2

Bridge 3

Bridge 4



Bridge 5

Bridge 6

Bridge 7

Bridge 8

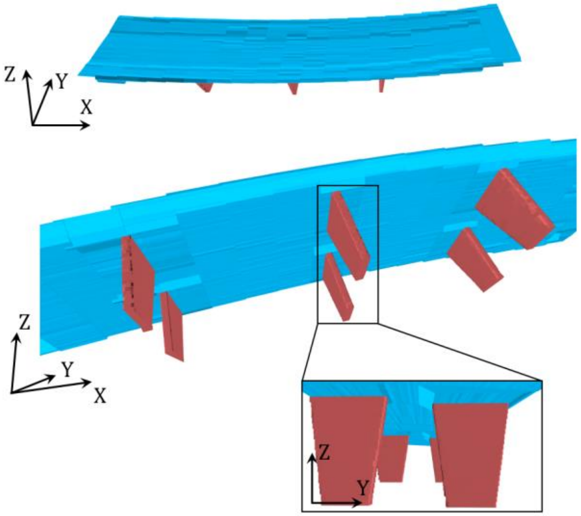
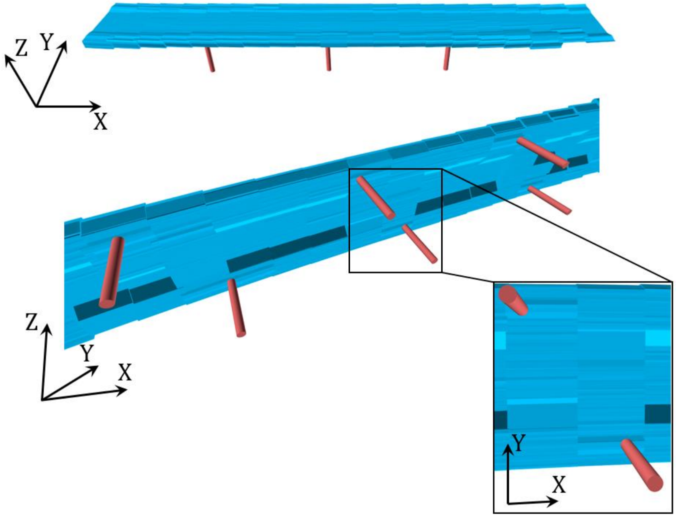
Bridge 9	***Bridge 10***
	

Table 6.20 LOD 250—300 bridge gDTs (# of deck slab slices=20, # of pier slices=20, α=0.98)

The time spent on runtime fitting for each bridge is shown in Table 6.21. For a bridge dataset of four types of point cluster containing less than one million points together, the average processing time was 10.2±4.2 seconds for LOD 200 gDT generation and 37.8±28.4 seconds for LOD 250—300 gDT generation. The time spent on generating the gDT of *Bridge 10* (16.3 seconds for LOD 200, and 65.5 seconds for LOD 250—300) was higher than for the other bridges. This is because *Bridge 10* has more points than the others. It is reasonable to have this result: the more points the bridge point clusters have, the longer the processing time. The time spent on generating the LOD 250—300 gDTs for *Bridge 4* and for *Bridge 6* was 53.7% and 22.5% higher than the average, respectively. This is mainly due to the large sparse regions in the slab point clusters of *Bridge 4* and *Bridge 6*. Large low-density areas slow down the running time.

In summary, compared to the time spent on manually generating *GT D* (0.92 hours = 3312 seconds) and *GT E* (27.6 hours = 99360 seconds) (Section 6.2.2.3), the time cost of the automatic method is trivial. This means an impressive direct time saving of 99.7% and 100%, respectively.

Bridge ID	**#pt**	**Processing time (seconds)**	
		LOD 200	**LOD 250—300**
1	488453	10.1	25.5
2	500000	10.3	25.3
3	500000	9.5	23.7
4	498579	9.5	58.1
5	500000	9.3	23.5
6	500000	9.5	46.3
7	454985	8.2	31.1
8	500000	9.3	39.8
9	500000	10.0	37.3
10	875036	16.3	65.5
Avg.	531705	**10.2**	**37.8**

Table 6.21 Fitting time of ten down-sampled bridge datasets

6.2.5.3. Evaluation of LOD 200 gDTs

In LOD 200 gDTs, the vertices of each bridge component represented in OBB are defined as:

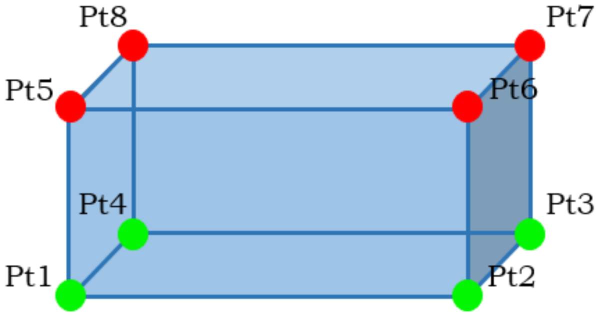


Figure 6.21 Vertices of an OBB

The author computed the volume (Vol) and the centroid of each GT bounding box (hereafter gtBBox) Vol_{gt} and C_{gtBBox}, and the automated bounding box (hereafter autoBBox) Vol_{auto} and C_{autoBBox}. The results are shown in Table 6.22. The Euclidean distance E_{dc} and the false volume ratio (FVR) between each C_{gtBBox} and the corresponding C_{autoBBox} were also computed. The FVR was calculated using:

$$\mathrm{FVR} = \frac{|Vol_{auto} - Vol_{gt}|}{Vol_{gt}}. \quad \text{(Eq. 6.20)}$$

As shown in Table 6.22, for each bridge, $\overline{E_{dc}}$ was small. *Bridge 6* had the minimum $\overline{E_{dc}}$ (0.03 m) while *Bridge 7* had the maximum (0.23 m). The average value of $\overline{E_{dc}}$ for ten bridges was 0.11 m. That is to say, in average, the centroid of an autoBBox deviated 11 cm from the centre of its corresponding gtBBox. By contrast, the FVR for all bridges maintained higher than 10%. *Bridge 5* had the minimum $\overline{\mathrm{FVR}}$ (10.5%) while *Bridge 7* had the maximum (24.1%). The average value of $\overline{\mathrm{FVR}}$ for ten bridges was 16.5%. That is to say, in average, the volume of an autoBBox was 16.5% larger than the volume of its corresponding gtBBox.

Specifically, for *Bridge 1*, the $\mathrm{FVR}_{\mathrm{pierCap1}}$, $\mathrm{FVR}_{\mathrm{pierCap2}}$, and $\mathrm{FVR}_{\mathrm{pierCap3}}$ of the autoBBoxes were 29.4%, 24.3%, and 24.0%, respectively. However, their centroids deviated by only 19 cm, 5 cm, and 2 cm from the GT centroids. Likewise, the $\mathrm{FVR}_{\mathrm{slab}}$ of *Bridge 2* was 44% while its C_{autoBBox} deviated 16 cm from its C_{gtBBox}. Compared to the minimal dimension of the slab of *Bridge 2*, with a slab thickness of 1.60 m, this deviation is a relatively minor issue (10%). The big false volume added to the autoBBox of the deck slab was attributed to the thicker slab bounding box generated from the method (i.e. h_{gt}=1.6 m, h_{auto}=2.3 m). For *Bridge 3, Bridge 4, and Bridge 5*, the results of both their $\overline{E_{dc}}$ (9 cm, 17cm, and 12 cm) and $\overline{\text{FVR}}$ (12%, 10.8%, and 10.5%) were generally good. The centroids of some autoBBoxes in *Bridge 4* had bigger deviations. The $\overline{E_{dc}}$ of *Bridge 6* was quite small (0.03 cm) whereas its $\overline{\text{FVR}}$ (22.6%) was not trivial. This is due to the dimensions of the piers of *Bridge 6* being small, so minor vertex displacement may lead to a large added false volume. The FVR_{slab} of *Bridge 7* was 46.3% while its C_{autoBBox} deviated only 6 cm from its C_{gtBBox}. The E_{dc} of most of the girders in *Bridge 7* was relatively large, which raised the average of E_{dc} up to 0.23 m. This is mainly because the girders generated from the method are longer than the GT ones. *Bridge 7* contains around 8% occlusion due to limited line-of-sight to the web of the girders as well as the extremities. The occluded girders render the GT gDTs (i.e. *GT D*) biased as a modeller can infer the hidden information based on his or her knowledge. The topologies of *Bridge 8* and *Bridge 9* were similar, but the $\overline{E_{dc}}$ of *Bridge 9* (18 cm) was bigger than that of *Bridge 8* (7 cm). This is due to the centroids of some autoBBoxes of *Bridge 9* having non-trivial deviations from their gtBBoxes. Finally, *Bridge 10* had a small $\overline{E_{dc}}$ (7 cm) while the abnormal FVR of pier 12 (40.8%) led the $\overline{\text{FVR}}$ (18.7%) to be not insignificant.

The author also computed the average point-to-point (P2P) distance, which is the Euclidean distance between each vertex of the automated gDT object and that of the GT one for each component. Note that the P2P values in the Table 6.22 were the average P2P value of the component vertices (8 vertices). The overall average $\overline{\text{P2P}}$ for all the ten bridges was 23 cm. The largest $\overline{\text{P2P}}$ found was 35 cm for *Bridge 7* while the smallest was 15 cm for *Bridge 8*.

Table 6.22 Comparison of LOD 200 Auto gDTs and *GT D* gDTs (*Bridge 1—Bridge 10*)

		***GT D* gDTs**		**Auto gDTs**				
Bridge 1	Component	Vol_{gt}	C_{gtBBox}	Vol_{auto}	C_{autoBBox}	E_{dc}	FVR	P2P
		(m^3)		(m^3)		(m)	(%)	(m)
	slab	1333.1	(41.51, -0.025, 270.21)	1483.7	(41.58, -0.01, 270.24)	0.08	11.3	0.15
	pierCap1	11.9	(21.74, 0.33, 269.03)	15.4	(21.76, 0.37, 269.21)	0.19	29.4	0.27
	pierCap2	11.9	(40.85, 0.09, 269.03)	14.8	(40.86, 0.10, 269.08)	0.05	24.3	0.13
	pierCap3	12.5	(59.96, -0.13, 268.96)	15.5	(59.94, -0.14, 268.97)	0.02	24.0	0.12
	pier 11	2.75	(18.98, -2.99, 266.70)	3.04	(18.98, -2.99, 266.70)	0.01	10.6	0.26
	pier 12	2.77	(21.71, 0.34, 266.68)	2.95	(21.69, 0.35, 266.69)	0.02	6.5	0.07
	pier 13	2.78	(24.43, 3.66, 266.68)	3.03	(24.40, 3.67, 266.69)	0.04	9.0	0.54
	pier 21	3.01	(38.10, -3.20, 266.52)	3.54	(38.10, -3.20, 266.48)	0.03	17.6	0.20
	pier 22	3.05	(40.83, 0.12, 266.49)	3.61	(40.82, 0.16, 266.48)	0.04	18.4	0.54
	pier 23	3.08	(43.52, 3.46, 266.47)	3.86	(43.52, 3.47, 266.48)	0.01	25.3	0.08
	pier 31	2.94	(57.21, -3.42, 266.51)	3.44	(57.20, -3.41, 266.48)	0.03	17.0	0.24
	pier 32	2.97	(59.92, -0.065, 266.50)	3.42	(59.90, -0.082, 266.37)	0.13	15.2	0.29
	pier 33	2.95	(62.71, 3.26, 266.51)	3.53	(62.63, 3.25, 266.49)	0.09	19.7	0.32
	avg					**0.06**	**17.6**	**0.23**
Bridge 2	Component	Vol_{gt}	C_{gtBBox}	Vol_{auto}	C_{autoBBox}	E_{dc}	FVR	P2P
		(m^3)		(m^3)		(m)	(%)	(m)
	slab	1473.3	(-5.59, 1.99, -1.34)	2121.3	(-5.52, 2.04, -1.20)	0.16	44.0	0.49
	pier 11	39.8	(-18.79, 1.02, -5.07)	42.7	(-18.72, 0.995, -5.19)	0.14	7.29	0.23
	pier 21	40.3	(-3.83, 0.998, -4.78)	44.1	(-3.76, 0.98, -4.81)	0.08	9.43	0.15
	pier 31	40.6	(11.19, 0.98, -4.49)	43	(11.25, 0.97, -4.46)	0.07	5.91	0.19

	avg					**0.11**	**16.7**	**0.26**
Bridge 3	Component	Vol_{gt}	C_{gtBBox}	Vol_{auto}	C_{autoBBox}	E_{dc}	FVR	P2P
		(m^3)		(m^3)		(m)	(%)	(m)
	slab	1748.5	(0.15, 0.07, -0.60)	1879.6	(0.17, 0.21, -0.65)	0.15	7.5	0.21
	pier 11	37.6	(-15.07, 1.75, -4.03)	42.7	(-15.04, 1.76, -4.06)	0.05	13.6	0.10
	pier 21	37.7	(1.87, -0.023, -4.02)	43.7	(1.80, -0.023, -3.96)	0.09	15.9	0.26
	pier 31	37.7	(14.96, 1.95, -4.02)	41.8	(15.01, 1.95, -4.05)	0.06	10.9	0.10
	avg					**0.09**	**12.0**	**0.17**
Bridge 4	Component	Vol_{gt}	C_{gtBBox}	Vol_{auto}	C_{autoBBox}	E_{dc}	FVR	P2P
		(m^3)		(m^3)		(m)	(%)	(m)
	slab	2079.5	(4.20, -0.08, -0.65)	2605.7	(4.25, -0.26, -0.65)	0.19	22.7	0.31
	pier 11	13.56	(-1.93, -11.95, -4.13)	14.7	(-1.91, -11.95, -4.28)	0.15	8.41	0.17
	pier 12	9.71	(-1.91, -7.03, -4.09)	10.9	(-1.91, -7.02, -4.26)	0.18	12.3	0.20
	pier 13	9.62	(-1.89, -2.37, -4.08)	10.5	(-1.88, -2.36, -4.24)	0.17	9.15	0.22
	pier 14	9.46	(-1.87, 2.27, -4.02)	10.2	(-1.87, 2.27, -4.14)	0.12	7.82	0.13
	pier 15	9.35	(-1.84, 6.89, -3.97)	10.5	(-1.84, 6.89, -4.18)	0.21	12.3	0.17
	pier 16	12.8	(-1.83, 11.84, -3.93)	14.8	(-1.79, 11.84, -3.93)	0.09	15.6	0.13
	pier 21	13.11	(12.69, -12.03, -4.26)	14.7	(12.67, -12.02, -4.40)	0.15	12.1	0.15
	pier 22	9.21	(12.72, -7.1, -4.21)	9.78	(12.71, -7.09, -4.45)	0.24	6.19	0.18
	pier 23	8.99	(12.74, -2.45, -4.20)	10.2	(12.76, -2.45, -4.35)	0.16	13.5	0.21
	pier 24	8.89	(12.76, 2.19, -4.16)	9.5	(12.77, 2.20, -4.32)	0.17	6.86	0.15
	pier 25	8.76	(12.78, 6.83, -4.10)	9.5	(12.78, 6.84, -4.42)	0.32	8.45	0.17
	pier 26	11.97	(12.82, 11.77, -4.06)	12.6	(12.81, 11.76, -4.13)	0.07	5.26	0.24
	avg					**0.17**	**10.8**	**0.19**
Bridge 5	Component	Vol_{gt}	C_{gtBBox}	Vol_{auto}	C_{autoBBox}	E_{dc}	FVR	P2P
		(m^3)		(m^3)		(m)	(%)	(m)

	slab	1644	(-1.21, -4.05, -0.61)	1863	(-1.15, -4.14, -0.57)	0.11	13.3	0.31
	pier 11	53.5	(-18.19, -4.07, -4.17)	58.2	(-18.19, -4.10, -4.28)	0.12	8.8	0.15
	pier 12	47.4	(13.00, -4.12, -4.08)	51.9	(13.07, -4.12, -4.20)	0.14	9.5	0.22
	avg					**0.12**	**10.5**	**0.22**
Bridge 6	Component	Vol_{gt}	C_{gtBBox}	Vol_{auto}	C_{autoBBox}	E_{dc}	FVR	P2P
		(m³)		(m³)		(m)	(%)	(m)
	slab	1698	(-34.04, -11.97, 257.1)	2197	(-34.05, -11.91, 256.88)	0.19	29.4	0.26
	pier 11	6.5	(-49.59, -14.81, 253.6)	7.9	(-49.59, -14.80, 253.49)	0.11	21.5	0.23
	pier 12	6.4	(-49.56, -8.81, 253.65)	7.7	(-49.57, -8.81, 253.75)	0.10	20.3	0.14
	pier 21	7	(-34.02, -14.85, 253.5)	7.5	(-34.02, -14.85, 253.38)	0.08	7.14	0.24
	pier 22	6.7	(-34.00, -8.85, 253.5)	7.7	(-34.00, -8.86, 253.50)	0.01	14.9	0.12
	pier 31	7	(-18.43, -14.90, 253.4)	8.8	(-18.37, -14.90, 253.31)	0.07	25.7	0.40
	pier 32	7	(-18.40, -8.90, 253.35)	8.9	(-18.42, -8.91, 253.36)	0.02	27.1	0.12
	avg					**0.03**	**22.6**	**0.22**
Bridge 7	Component	Vol_{gt}	C_{gtBBox}	Vol_{auto}	C_{autoBBox}	E_{dc}	FVR	P2P
		(m³)		(m³)		(m)	(%)	(m)
	slab	369.1	(-8.85, -20.63, 231.65)	539.9	(-8.81, -20.66, 231.61)	0.06	46.3	0.48
	pier 11	72.2	(-10.65, -20.63, 227.7)	75.5	(-10.59, -20.65, 227.69)	0.07	4.57	0.10
	girder 11	18.1	(-19.47, -26.00, 230.7)	26.9	(-19.57, -26.15, 230.70)	0.19	29.3	0.39
	girder 12	20	(-19.47, -24.83, 230.6)	31.3	(-19.61, -24.93, 230.65)	0.20	35.0	0.32
	girder 13	20.1	(-19.45, -23.64, 230.8)	41.5	(-19.63, -23.72, 230.61)	0.29	37.8	0.39
	girder 14	20.5	(-19.51, -22.14, 230.7)	29.6	(-19.69, -22.14, 230.79)	0.22	31.7	0.27
	girder 15	20.5	(-19.53, -20.64, 230.7)	26.3	(-19.76, -20.65, 230.68)	0.24	28.3	0.32
	girder 16	20.5	(-19.57, -19.1, 230.73)	26.3	(-19.78, -19.12, 230.83)	0.24	28.3	0.32
	girder 17	20.5	(-19.65, -17.66, 230.8)	41	(-19.85, -17.62, 230.71)	0.22	36.1	0.29
	girder 18	21.8	(-19.66, -16.42, 230.8)	23.7	(-19.87, -16.41, 230.56)	0.32	8.72	0.38
	girder 19	19.1	(-19.73, -15.30, 230.7)	52.8	(-19.88, -15.36, 230.83)	0.18	29.8	0.28
	girder 21	21.7	(0.09, -26.00, 230.65)	28.2	(0.33, -26.11, 230.71)	0.28	30.0	0.43

	girder 22	21.7	(0.055, -24.86, 230.73)	23	(0.31, -24.86, 230.65)	0.26	11.5	0.29
	girder 23	25	(0.065, -23.62, 230.80)	31.4	(0.29, -23.59, 230.65)	0.27	25.6	0.46
	girder 24	24.6	(0.015, -22.12, 230.73)	30.2	(0.24, -22.14, 230.84)	0.25	22.8	0.35
	girder 25	24.6	(0.005, -20.64, 230.73)	31	(0.22, -20.64, 230.85)	0.24	26.0	0.36
	girder 26	24.6	(-0.045, -19.14, 230.9)	28.8	(0.16, -19.14, 230.77)	0.22	17.1	0.40
	girder 27	24.6	(-0.065, -17.64, 230.8)	26.2	(0.12, -17.73, 230.65)	0.25	6.5	0.41
	girder 28	26.1	(-0.115, -16.42, 230.9)	29.2	(0.11, -16.40, 230.59)	0.35	11.9	0.41
	girder 29	22.8	(-0.22, -15.23, 230.70)	30.5	(0.04, -15.29, 230.71)	0.26	14.9	0.37
	avg					**0.23**	**24.1**	**0.35**
Bridge 8	Component	Vol_{gt}	C_{gtBBox}	Vol_{auto}	C_{autoBBox}	E_{dc}	FVR	P2P
		(m^3)		(m^3)		(m)	(%)	(m)
	slab	936.5	(-3.91, -16.96, 167.63)	961.2	(-3.89, -16.99, 167.60)	0.04	2.64	0.19
	pier 11	11.4	(-19.94, -19.40, 164.0)	18.2	(-19.92, -19.34, 163.99)	0.07	20.1	0.22
	pier 12	11.4	(-19.24, -14.97, 164.0)	16.3	(-19.23, -14.96, 163.95)	0.02	9.7	0.13
	pier 21	11.9	(-5.15, -19.70, 164.0)	13.7	(-5.17, -19.75, 164.10)	0.11	15.1	0.14
	pier 22	11.9	(-4.43, -15.30, 164.1)	13.6	(-4.44, -15.27, 164.00)	0.06	14.3	0.08
	pier 31	13.3	(9.80, -19.44, 163.90)	18.4	(9.74, -19.54, 164.96)	0.13	38.4	0.19
	pier 32	13.5	(10.45, -15.02, 163.9)	150	(10.49, -15.02, 163.89)	0.04	11.1	0.12
	avg					**0.07**	**16.0**	**0.15**
Bridge 9	Component	Vol_{gt}	C_{gtBBox}	Vol_{auto}	C_{autoBBox}	E_{dc}	FVR	P2P
		(m^3)		(m^3)		(m)	(%)	(m)
	slab	2014.4	(-38.71, 19.48, 156.20)	2014.4	(-38.57, 19.44, 156.15)	0.16	0	0.55
	pier 11	13.9	(-55.45, 14.46, 152.70)	18.5	(-55.44, 14.50, 152.85)	0.16	33.1	0.21
	pier 12	13.9	(-55.21, 20.96, 152.75)	21.2	(-55.21, 20.77, 152.70)	0.20	18.0	0.30
	pier 21	14.5	(-40.79, 14.13, 152.70)	15.5	(-40.79, 14.14, 152.95)	0.25	6.9	0.27
	pier 22	14.1	(-40.58, 20.66, 152.70)	15.1	(-40.58, 20.72, 152.75)	0.08	7.1	0.16
	pier 31	16	(-26.13, 14.95, 152.55)	23	(-26.13, 14.89, 152.78)	0.23	43.8	0.32

	pier 32	16.3	(-25.89, 21.42, 152.55)	16.9	(-25.76, 21.49, 152.60)	0.15	3.68	0.26
	avg					**0.18**	**16.1**	**0.30**
Bridge 10	Component	Vol_{gt}	C_{gtBBox}	Vol_{auto}	C_{autoBBox}	E_{dc}	FVR	P2P
		(m^3)		(m^3)		(m)	(%)	(m)
	slab	1431	(30.20, -7.01, 156.00)	1778.4	(30.25, -7.03, 155.91)	0.10	24.3	0.32
	pier 11	3.54	(8.63, -10.88, 152.45)	3.38	(8.62, -10.87, 152.48)	0.03	4.52	0.35
	pier 12	3.33	(14.66, -3.00, 152.55)	5	(14.64, -3.01, 152.65)	0.10	40.8	0.13
	pier 21	3.56	(27.30, -10.95, 152.55)	4.14	(27.31, -10.97, 152.60)	0.05	16.3	0.27
	pier 22	3.33	(33.32, -3.08, 152.65)	3.55	(33.26, -3.15, 152.75)	0.14	6.61	0.35
	pier 31	3.78	(46.02, -11.04, 152.45)	4.77	(46.03, -11.04, 152.48)	0.03	26.2	0.11
	pier 32	3.56	(52.05, -3.14, 152.55)	4	(52.06, -3.16, 152.60)	0.06	12.4	0.34
	avg					**0.07**	**18.7**	**0.27**

6.2.5.4. Evaluation of LOD 250—300 gDTs

While the LOD 200 gDTs were evaluated using vertex-based metrics, it is difficult to use the same metrics to evaluate the automatically generated LOD 250—300 gDTs, because the vertices of the GT models and that of the automated ones do not correspond. This is attributed to the nature of the *ConcaveHull* algorithms used in the proposed solution. Normally, the number of hulls found in the automated models by the proposed method is much more than that of the component vertices of the GT models. The surfaces of the GT models, i.e. GT E, are smoothed planes without local undulations. In addition, during the manual model generation stage, the author simplified the underlying shape of each segmented point cluster of a bridge point cloud. This is true because when a modeller uses a modelling software interface to assist the act of creating an 3D object embedded in PCD, almost every object description is approximate in the sense that it describes the geometry or the shape of the 3D object only to the extent that inputting this description the modelling software module produces a 3D model of acceptable quality. To this end, the author chose the Hausdorff distance-based metrics to evaluate the automated LOD 250—300 gDTs.

One central problem in computer graphics and computer vision is measuring the extent to which one shape differs from another. The Hausdorff distance is a commonly used shape comparison method that can measure the difference between two different representations of the same 3D object when generating the LOD for efficient display of complex 3D models (Aspert et al., 2002; Cignoni et al., 1998; Huttenlocher et al., 1993). The author conducted a Hausdorff-based cloud-to-cloud (C2C) distance evaluation to detect changes between the GT models ($GT\ E$) and the automated ones. Given two finite points $A = \{a_1, \dots, a_p\}$ and $B = \{b_1, \dots, b_q\}$, the Hausdorff distance is defined as:

$$H(A, B) = \max(h(A, B), h(B, A)), \qquad \text{(Eq. 6.21)}$$

$$h(A, B) = \max_{a \in A} \min_{b \in B} \|a - b\|_2, \qquad \text{(Eq. 6.22)}$$

where $\|.\|_2$ denotes the usual Euclidean norm on the points of A and B. The function $h(A,B)$ is called the directed Hausdorff distance from A to B. It determines the point $a \in A$ that is farthest from any point of B and measures the distance from a to its nearest neighbour in B (using $\|.\|$), that is, $h(A,B)$ ranks each point of A based on its distance to the nearest point of B and uses the largest ranked point as the distance (the most mismatched point of A). The Hausdoff distance $H(A,B)$ is the maximum of $h(A,B)$ and $h(B,A)$. In other words, it measures the degree of mismatch between A and B by determining the distance of the point of A that is farthest from any point of B and vice versa.

Note that the GT models $GT\ E$ might contain more components than the four types of components generated by the proposed method, such as bearings. Hence, the author first removed these irrelevant component models from $GT\ E$ and kept only the four structural component types studied in this research (slab, pier, pier cap, and girder).

The author then converted both IFC-based GT models and automated models into points. This was achieved by converting the geometry in .ifc format into Wavefront .obj file format using IfcOpenShell (2018), which is an open source ifc toolkit and geometry engine that helps users and software developers to work with the IFC file format. The .obj file format is open and has been adopted by many 3D graphic application vendors. The .obj format is a simple data format which represents only the 3D geometry information, such as the vertex position (used in the 3D modelling process), vertex normal, the faces that define each polygon as a list of vertices, and the UV position (used in the process of projecting a 2D image to the surface of a 3D model for texture mapping). The automatically generated LOD 250—300 gDTs were converted into their corresponding .obj files followed by randomly point sampling from the generated polygons using a similar method suggested in (Cignoni et al., 1998). The random point number from the .obj polygons was in line with the original size of the point cloud before down-sampling (Table 6.1). The author sampled both $GT\ E$ models and the resulting gDTs to generate two sets of point clouds: $GT\ E$ PCDs and Auto PCDs, from their .obj files (Table 6.23).

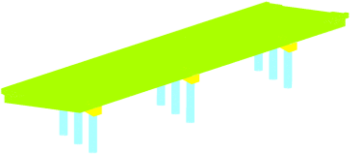
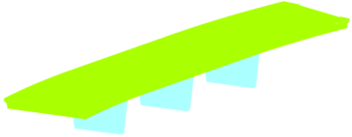
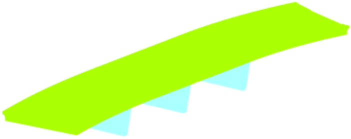
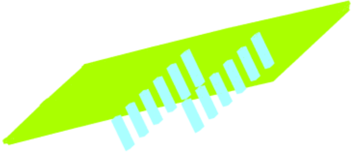
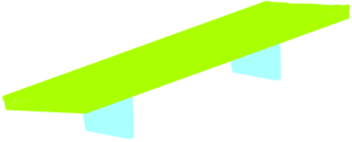
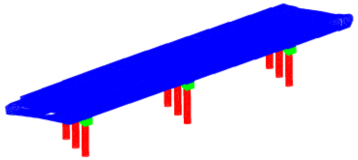
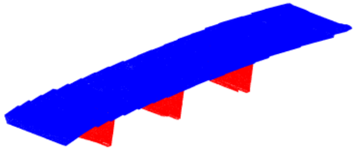
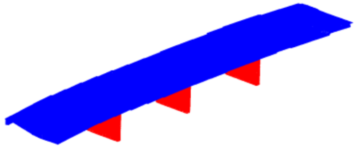
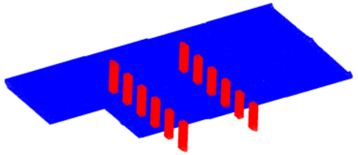
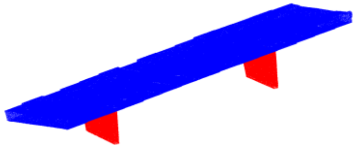
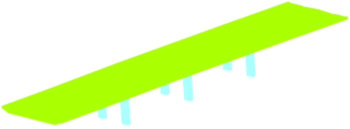
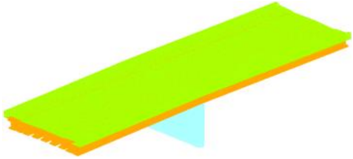
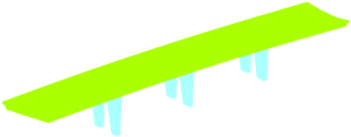
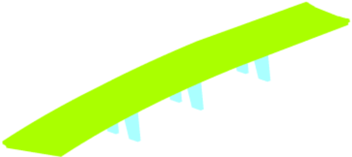
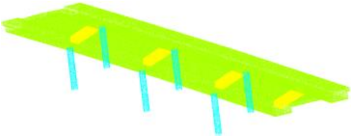
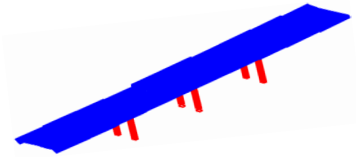
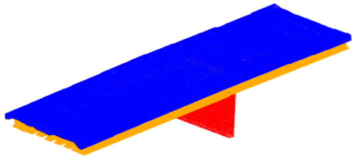
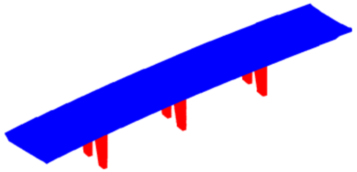
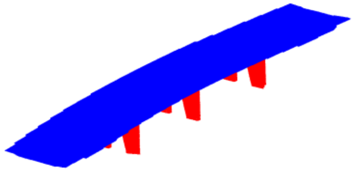
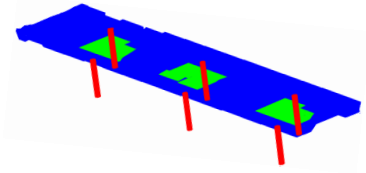
	Bridge 1	***Bridge 2***	***Bridge 3***	***Bridge 4***	***Bridge 5***
GT E PCD					
Auto PCD					
	Bridge 6	***Bridge 7***	***Bridge 8***	***Bridge 9***	***Bridge 10***
GT E PCD					
Auto PCD					

Table 6.23 Sampled point clouds of *GT E* and of Automated LOD 250—300 bridge gDTs

The issue that needs to be noticed is that the nearest neighbour is rarely in reality the actual nearest point on the surface represented by the cloud. This is especially true if the reference point cloud is non-uniformly distributed or contains occlusions. That is why the author used clouds that were as dense as possible to compute the distances. To conduct the distance calculations, for each bridge, the real-world point cloud and the sampled points of the GT model ($GT\,E$ PCD) as well as the automated one (Auto PCD) were all kept with an order of magnitude at least 4 million points. However, the defects in real-world points cannot be totally avoided. In this scenario, a local distance strategy was leveraged to compute a local model using neighbouring points to get a better estimation of the “real” distance (Figure 6.22). The effectiveness of this method is statistically more or less dependent on the point cloud sampling and on how appropriate the local surface approximation is. The author used a quadratic model Q, where Q can be expressed as:

$$f(x, y, z) = ax^2 + by^2 + cz^2 + dxy + exz + fyz + gx + hy + i + j = 0 \qquad \text{(Eq. 6.23)}$$

to fit the neighbouring points in the reference point cloud on a smooth surface within a radius of 0.3 m. This means, in reality, it is not correct to compute the distance of a single point, but one should take into account a local tendency. To this end, the distance in each sphere of radius equal to 0.3 m is the smaller distance between the directed Hausdorff distance (nearest neighbour distance) and the distance to the local model. Specifically, given a point $q_i = (x_o, y_o, z_o)$ of the compared point cloud that is not on the quadratic model Q, the Euclidean distance from this point q_i to Q is expressed as:

$$d(q_i, Q) = \min\{\|x - p\| : f(p) = 0\}. \qquad \text{(Eq. 6.24)}$$

Hence, the estimated average local distance from a compared point cloud to a reference point cloud is:

$$\overline{\mathrm{dist}} = \tfrac{1}{n}\textstyle\sum_{i=1}^{n} \min\{d(q_i, Q)\}. \qquad \text{(Eq. 6.25)}$$

The overall estimated distance between a compared point cloud and a reference point cloud is the bigger one of the mutual $\overline{\text{dist}}$, that is:

$$\text{C2C} = \max(\overline{\text{dist}}_{A/B}, \overline{\text{dist}}_{B/A}). \qquad \text{(Eq. 6.26)}$$

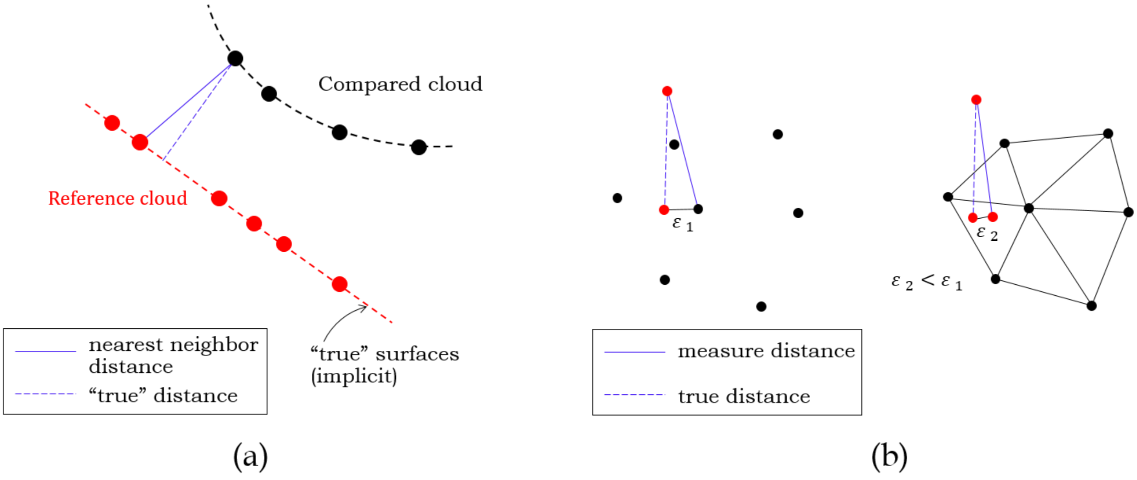


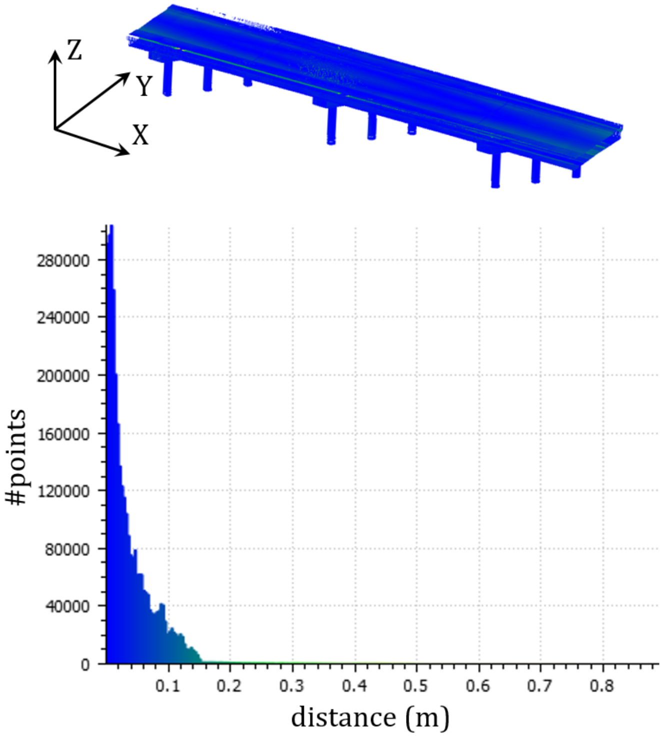
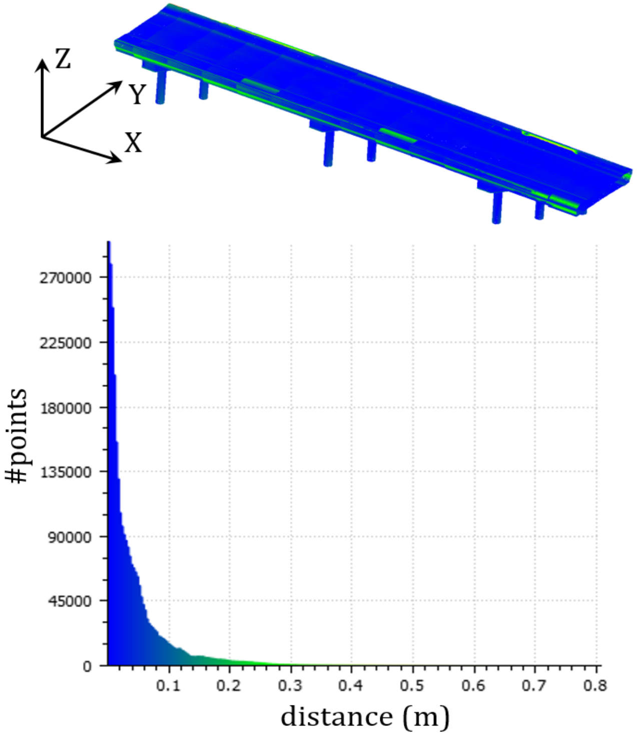
Figure 6.22 Comparison of cloud-to-cloud distance using (a) global model; and (b) local surface model

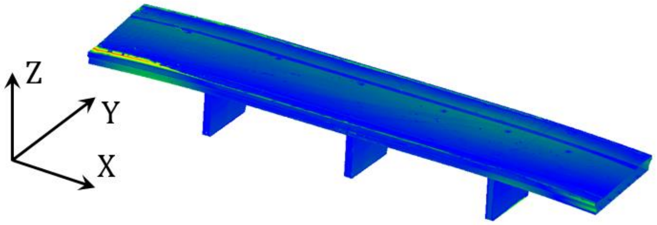
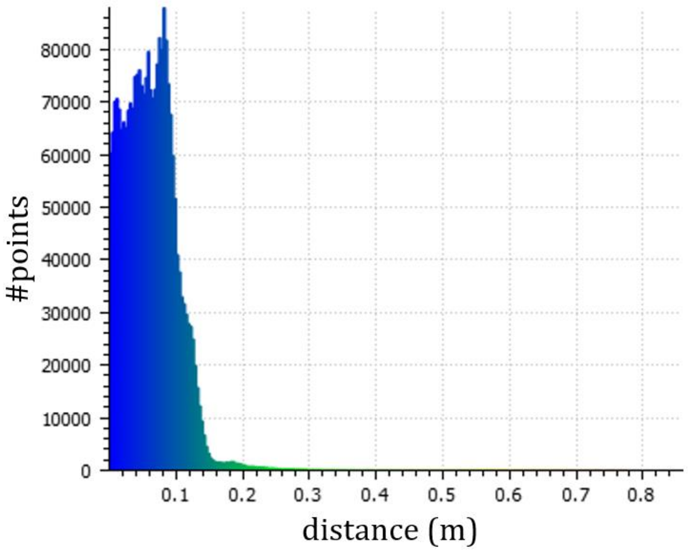
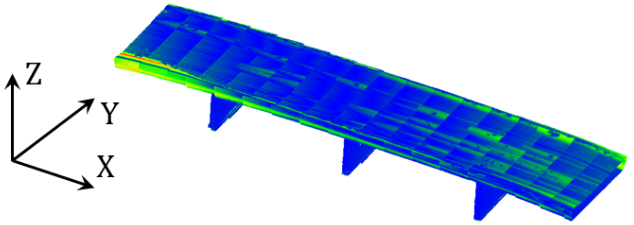
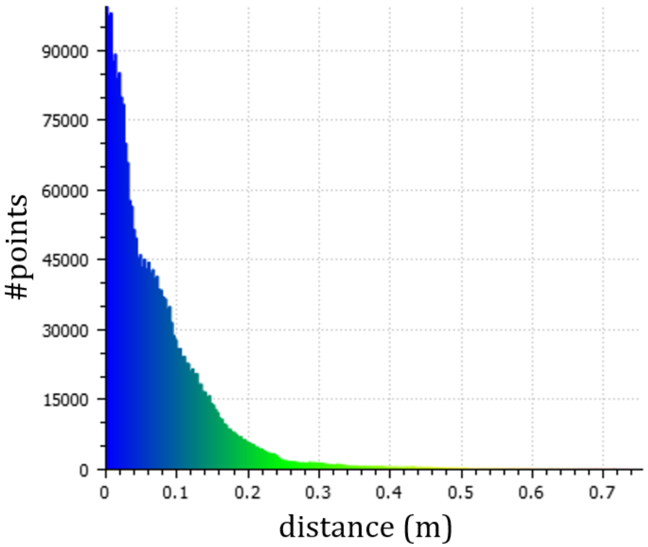
Table 6.24 demonstrates all experimental results of the ten bridge datasets, where the author summarized the C2C distance of:

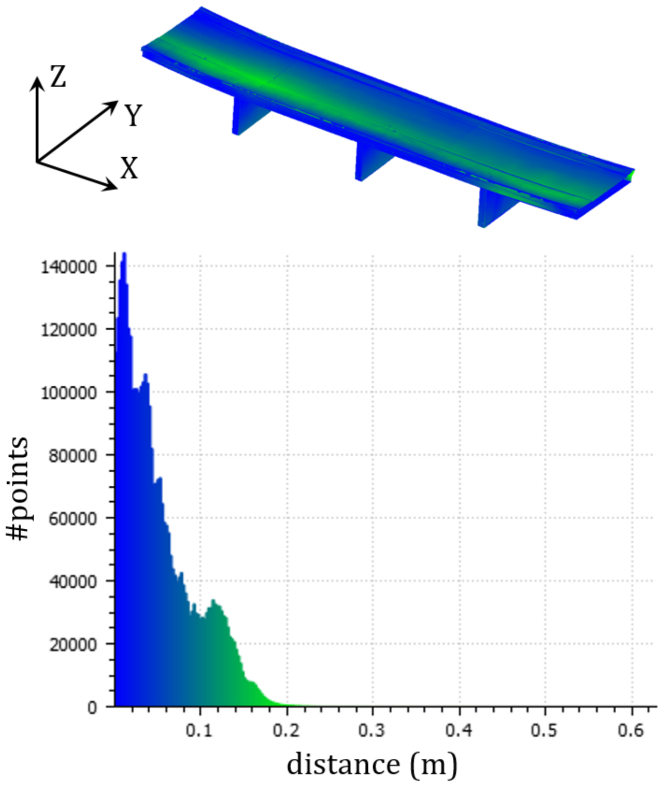
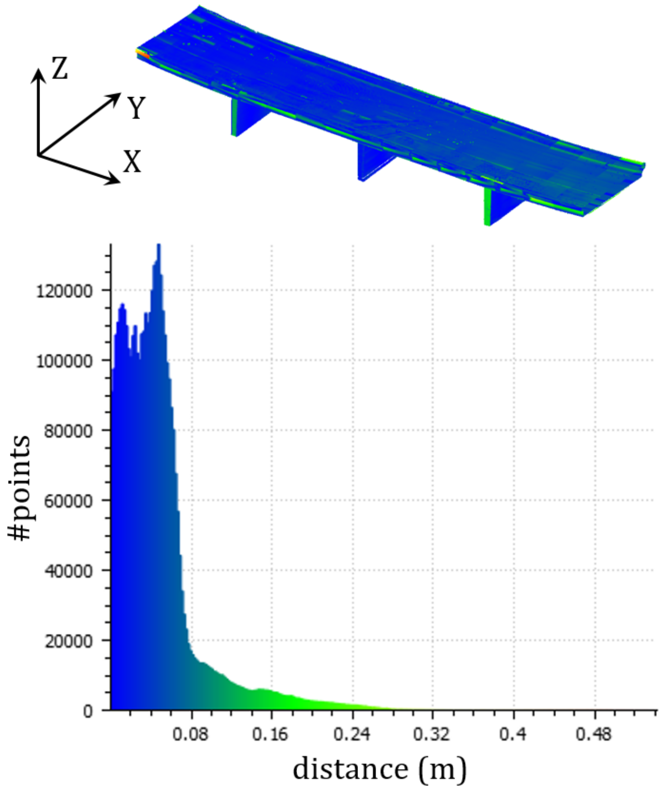
- *GT E* PCDs against the real world PCDs (i.e. *GT E*/Real & Real/*GT E*); and between the
- Auto PCDs against the real world PCDs (i.e. Auto/Real & Real/Auto).

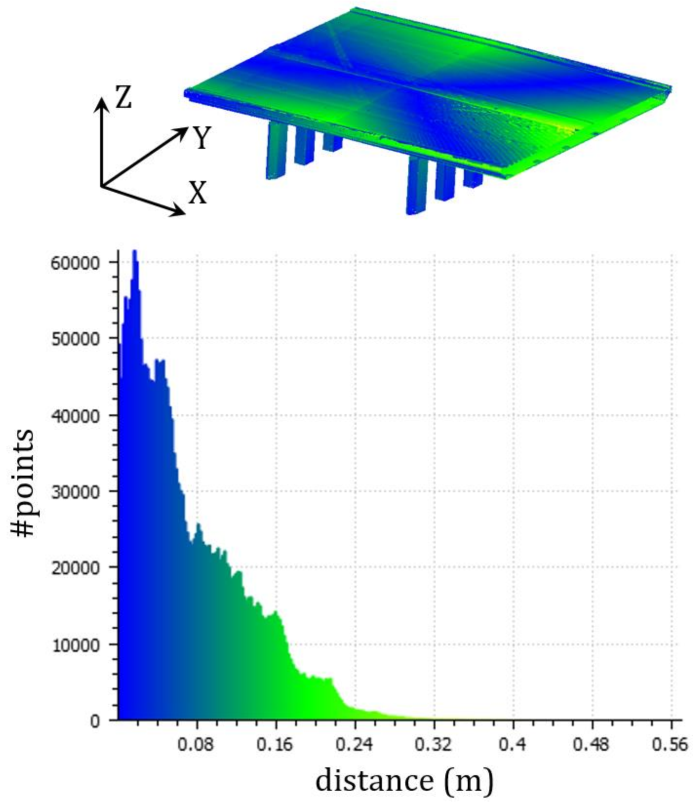
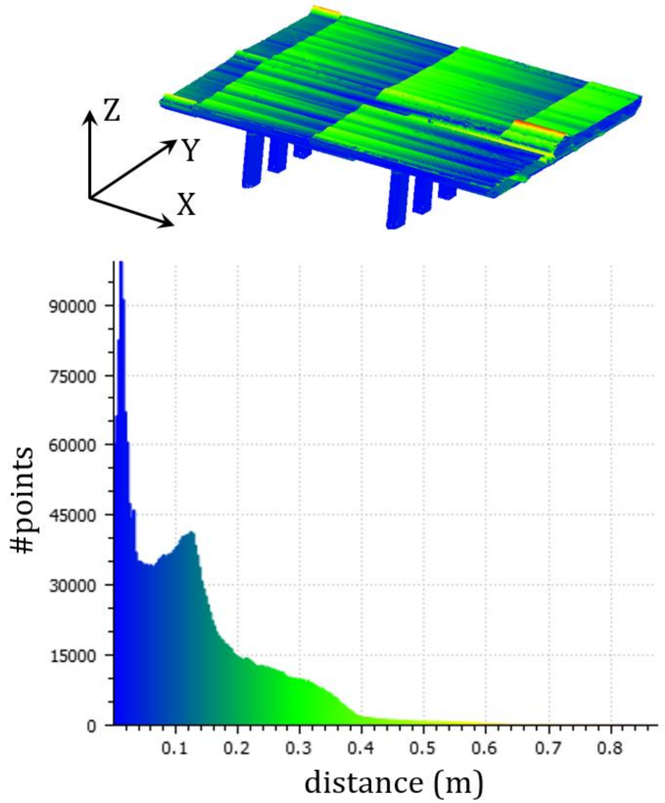
An automated model is deemed better modelled if its C2C (denoted C2C_{Auto}) is smaller compared with that of the manual model (denoted C2C_{GT}), and vice versa. Table 6.24 also illustrates the C2C distance in colour scalar field in the reference point cloud and the histograms of the C2C distribution using the RGB colour map, where the horizontal axis presents the C2C distance in meters while the vertical axis presents the point counts. The RGB colour was scaled from B (small distance) to R (big distance).

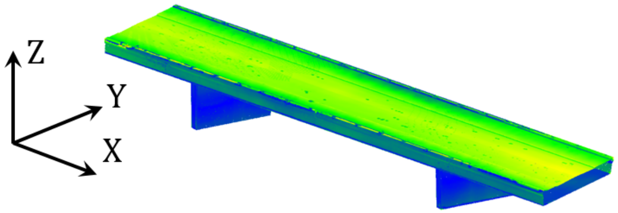
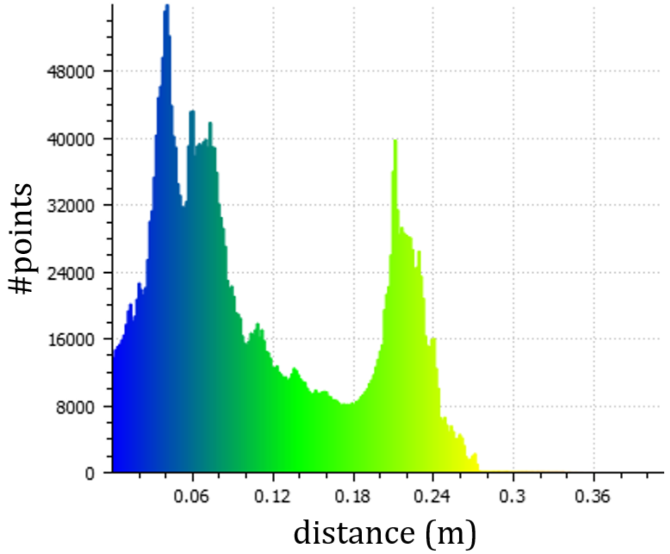
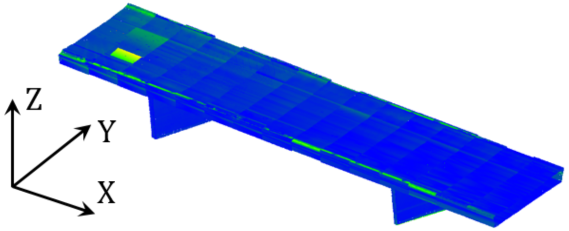
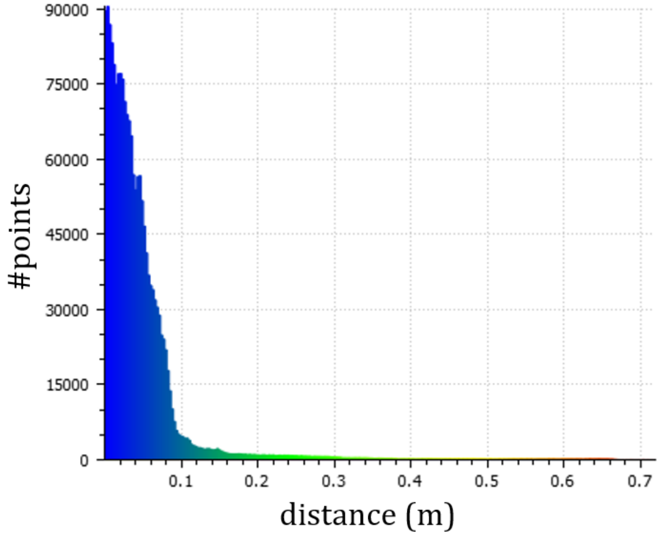
Table 6.24 Comparison of C2C between *GT E* PCD and Auto PCD against Real world PCD (*Bridge 1—Bridge 10*)

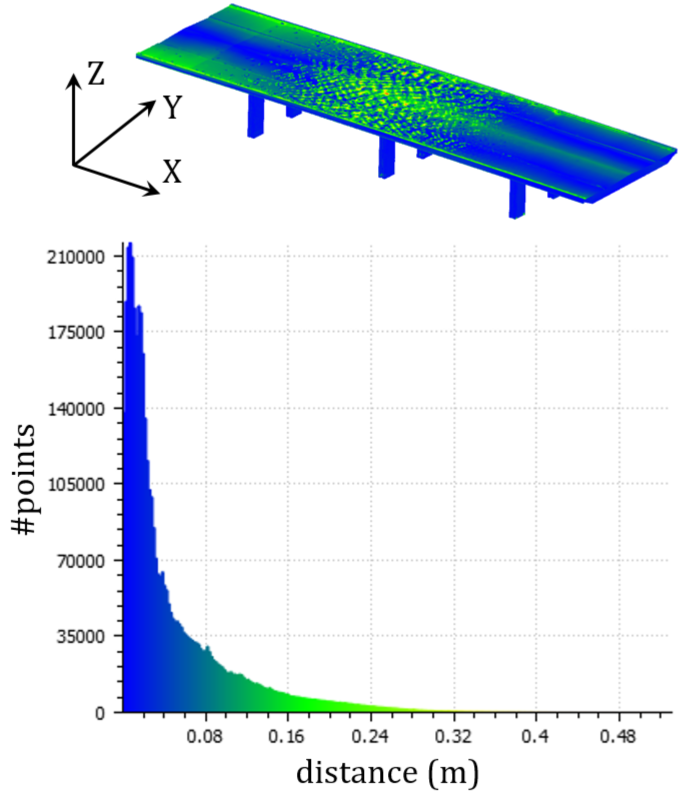
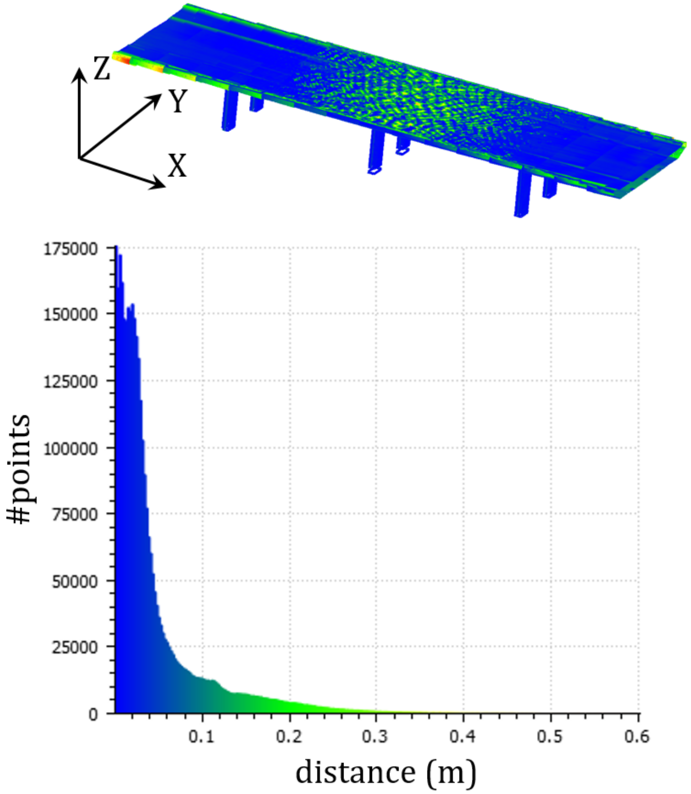
Bridge 1	***GT E* PCD & Real World PCD**		**Auto PCD & Real World PCD**	
A/B & B/A	*GT E*/Real	Real/*GT E*	Auto/Real	Real/Auto
$\overline{\text{dist}}$ (m)	0.040	0.040	0.043	0.043
std (m)	0.051	0.040	0.059	0.054
C2C (m)	**0.040**		0.043	
Histogram colour map				

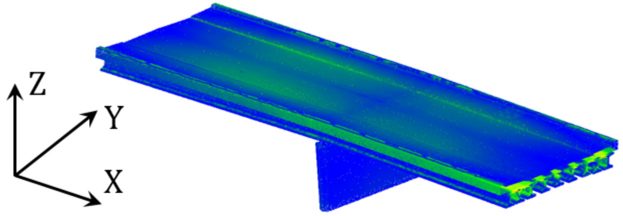
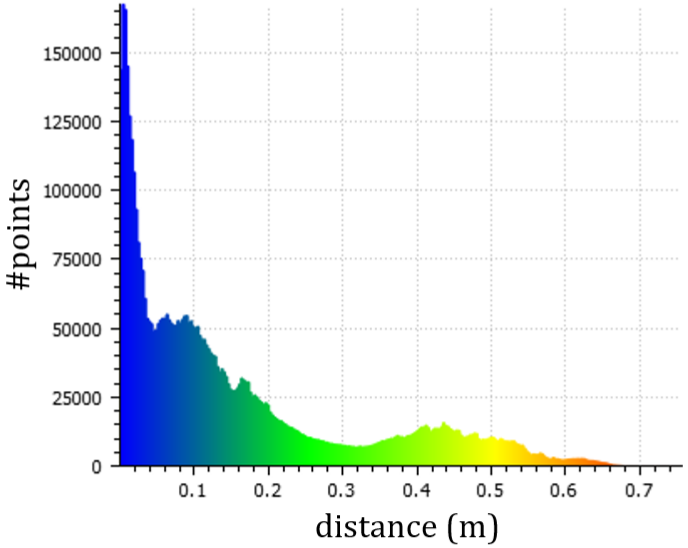
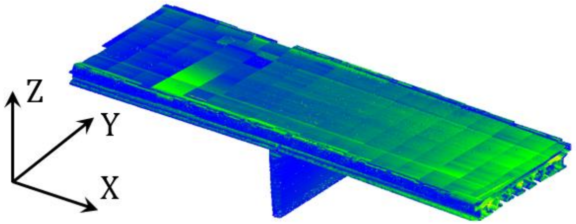
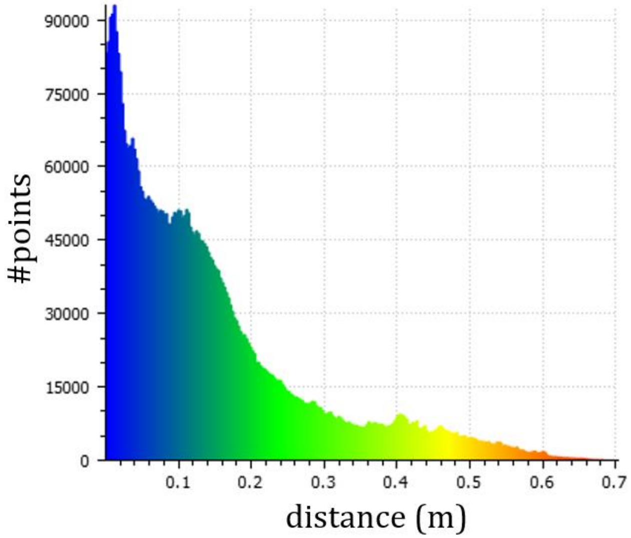
Bridge 2	***GT E* PCD & Real World PCD**		**Auto PCD & Real World PCD**	
A/B & B/A	*GT E*/Real	Real/*GT E*	Auto/Real	Real/Auto
$\overline{dist}$ (m)	0.064	0.060	0.073	0.067
std (m)	0.050	0.036	0.075	0.071
C2C (m)	**0.064**		0.073	
Histogram colour map	 		 	

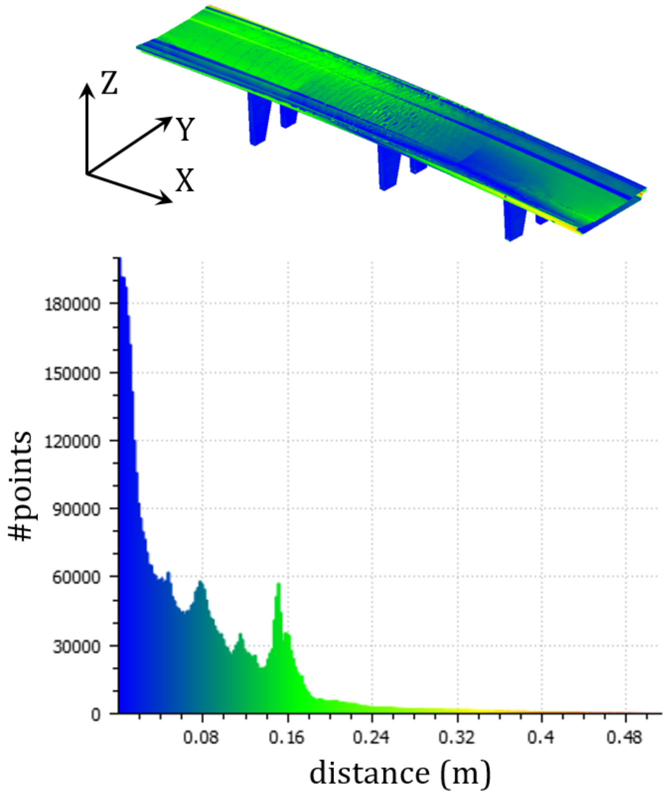
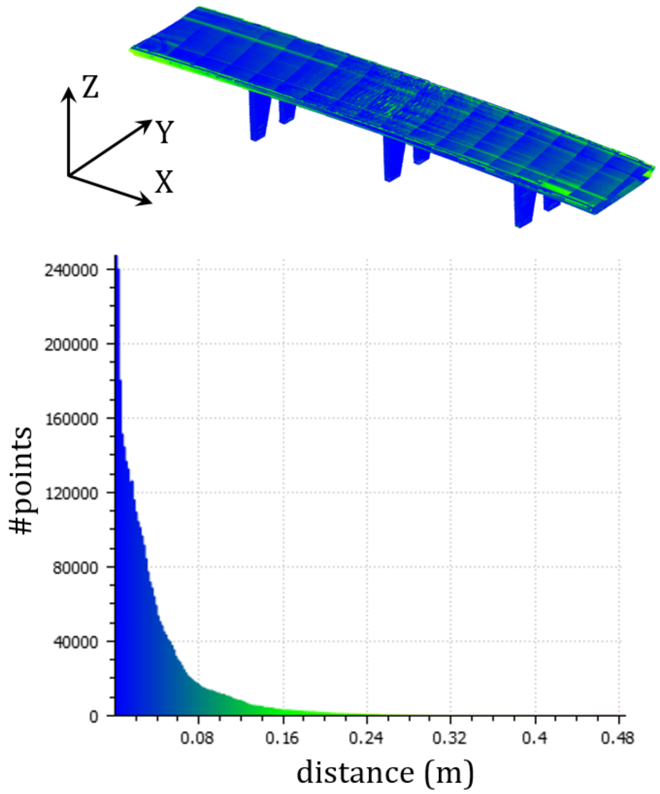
Bridge 3	***GT E* PCD & Real World PCD**		**Auto PCD & Real World PCD**	
A/B & B/A	*GT E*/Real	Real/*GT E*	Auto/Real	Real/Auto
$\overline{\text{dist}}$ (m)	0.052	0.050	0.044	0.047
std (m)	0.043	0.042	0.039	0.043
C2C (m)	0.052		**0.047**	
Histogram colour map				

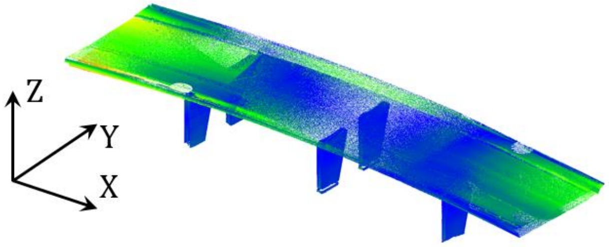
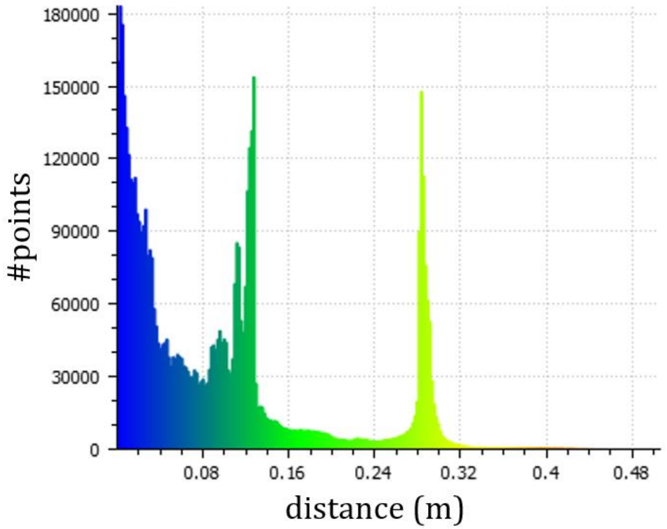
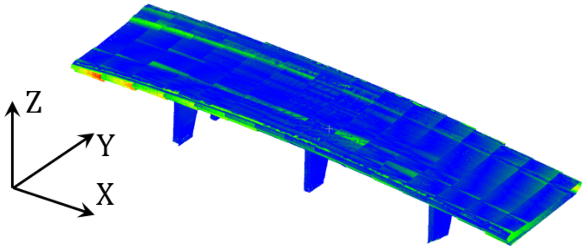
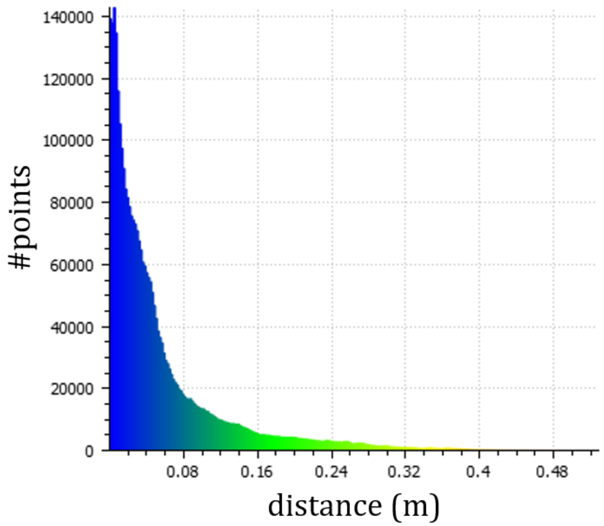
Bridge 4	***GT E* PCD & Real World PCD**		**Auto PCD & Real World PCD**	
A/B & B/A	*GT E*/Real	Real/*GT E*	Auto/Real	Real/Auto
$\overline{\mathrm{dist}}$ (m)	0.073	0.065	0.094	0.074
std (m)	0.059	0.059	0.120	0.095
C2C (m)	**0.073**		0.094	
Histogram colour map				

Bridge 5	***GT E* PCD & Real World PCD**		**Auto PCD & Real World PCD**	
A/B & B/A	*GT E*/Real	Real/*GT E*	Auto/Real	Real/Auto
$\overline{\text{dist}}$ (m)	0.109	0.098	0.049	0.036
std (m)	0.075	0.072	0.071	0.039
C2C (m)	0.109		**0.049**	
Histogram colour map	  		  	

Bridge 6	***GT E* PCD & Real World PCD**		**Auto PCD & Real World PCD**	
A/B & B/A	*GT E*/Real	Real/*GT E*	Auto/Real	Real/Auto
$\overline{\text{dist}}$ (m)	0.049	0.023	0.046	0.042
std (m)	0.058	0.020	0.058	0.05
C2C (m)	0.049		**0.046**	
Histogram colour map				

Bridge 7	***GT E* PCD & Real World PCD**		**Auto PCD & Real World PCD**	
A/B & B/A	*GT E*/Real	Real/*GT E*	Auto/Real	Real/Auto
$\overline{\text{dist}}$ (m)	0.157	0.042	0.125	0.055
std (m)	0.159	0.037	0.113	0.055
C2C (m)	0.157		**0.125**	
Histogram colour map	 		 	

Bridge 8	***GT E* PCD & Real World PCD**		**Auto PCD & Real World PCD**	
A/B & B/A	*GT E*/Real	Real/*GT E*	Auto/Real	Real/Auto
$\overline{\text{dist}}$ (m)	0.072	0.064	0.037	0.030
std (m)	0.072	0.102	0.043	0.050
C2C (m)	0.072		**0.037**	
Histogram colour map				

Bridge 9	***GT E* PCD & Real World PCD**		**Auto PCD & Real World PCD**	
A/B & B/A	*GT E*/Real	Real/*GT E*	Auto/Real	Real/Auto
$\overline{\text{dist}}$ (m)	0.076	0.098	0.056	0.044
std (m)	0.071	0.093	0.067	0.056
C2C (m)	0.098		**0.056**	
Histogram colour map	 		 	

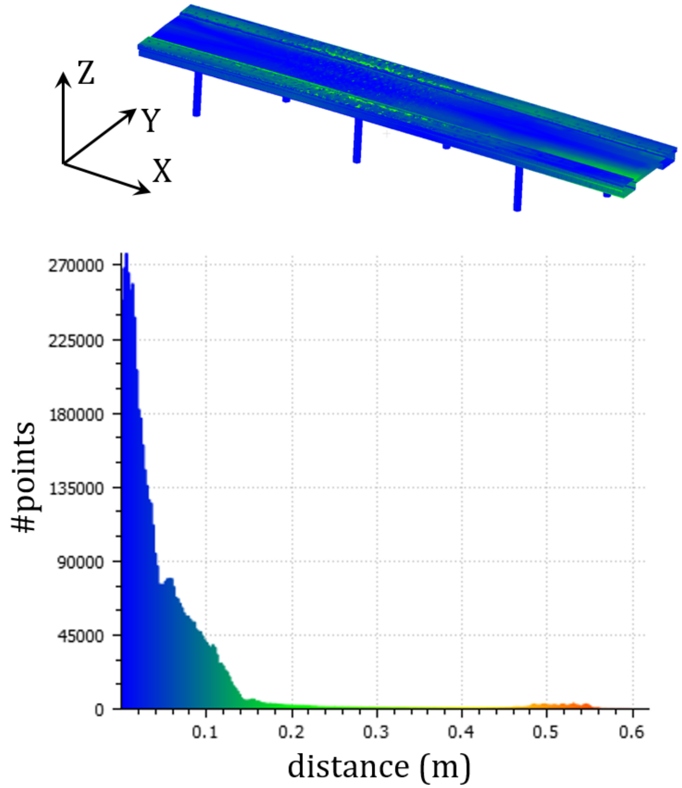
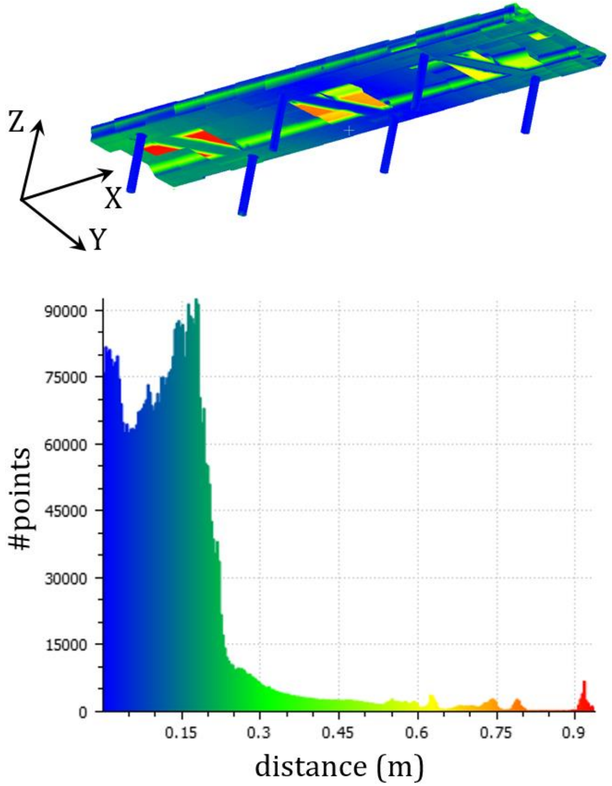
Bridge 10	***GT E* PCD & Real World PCD**		**Auto PCD & Real World PCD**	
A/B & B/A	*GT E*/Real	Real/*GT E*	Auto/Real	Real/Auto
$\overline{\text{dist}}$ (m)	0.055	0.036	0.135	0.080
std (m)	0.080	0.035	0.162	0.143
C2C (m)	**0.055**		0.135	
Histogram colour map				

Table 6.25 summarizes the number of matched points (in percentage) of each bridge derived from their automated gDTs, compared to the corresponding real-world point cloud. The term “matched” here, is defined at different levels, i.e. C2C<10 cm, C2C<7.5 cm, C2C<5 cm, and C2C<2.5 cm.

C2C	<10cm	<7.5cm	<5cm	<2.5cm
Bridge 1	89.1%	83.6%	73.2%	53.3%
Bridge 2	73.8%	62.8%	47.2%	29.1%
Bridge 3	94.6%	90.3%	69.5%	34.9%
Bridge 4	59.3%	53.2%	46.7%	37.7%
Bridge 5	95.8%	89.7%	75.4%	43.0%
Bridge 6	87.2%	82.3%	75.0%	50.7%
Bridge 7	56.2%	49.7%	40.5%	28.7%
Bridge 8	93.8%	89.5%	77.1%	55.2%
Bridge 9	83.4%	77.3%	66.5%	43.8%
Bridge 10	52.7%	47.0%	44.4%	36.6%
Avg.	**78.6%**	**72.5%**	**61.6%**	**41.3%**

Table 6.25 C2C in percentage of points between Auto PCDs and Real World PCDs

For all the bridge data, on average, almost 80% (78.6%) of points representing the automated gDTs had a C2C distance less than 10 cm, 72.5% inferior to 7.5 cm, 61.6% inferior to 5 cm, and 41.3% inferior to 2.5 cm. Specifically, the C2C distance of more than 80% of Auto points in *Bridge 1*, *3*, *5*, 6, *8* and *9* were found less than 10cm. Among them, *Bridge 3, 5* and *8* reached 94.6%, 95.8% and 93.8%, respectively. In addition, the C2C of more than 50% points of *Bridge 1* (53.3%), *Bridge 6* (50.7%), and *Bridge 8* (55.2%) were found inferior to 2.5 cm. The results demonstrated that the proposed method was able to generate the as-is geometry of some existing bridges within a tolerable margin of error, as the majority of the Auto points were aligned with the real-world points. However, the errors were non-trivial for some challenging bridges, such as *Bridge 10.*

6.2.5.5. Discussion

The results of LOD 200 gDTs in Table 6.22 showed that the generated components are generic placeholders, which can represent the approximate geometry of bridge components. The vertex-based metrics revealed that the proposed method tends to generate larger OBBs than the GT ones while the centroids of the autoBBoxes did not shift away too much from that of the corresponding gtBBoxes. These LOD 200 gDTs can be used in scenarios such as quick visualization, tender preparation, resource management, and cost planning, but cannot be used for maintenance operations.

The results of the LOD 250—300 model in Table 6.24 showed that the proposed method can produce well-modelled bridges from point clusters. This finding was drawn based on a distance-based quantitative measurement. For example, some bridges had very limited $\mathrm{C2C_{Auto}}$, such as *Bridges 3, 5,* and *8*. The overall $\overline{\mathrm{C2C}}_{\mathrm{Auto}}$ of ten bridge automated gDTs was 7.05 cm while the $\overline{\mathrm{C2C}}_{\mathrm{GT}}$ was 7.69 cm.

In total, six out of ten bridge PCDs, were modelled better by using the proposed methods than by manual modelling (for each bridge, the better C2C result was highlighted in green). The C2C of the rest four Auto PCDs were found close to that of their corresponding *GT E* ones. In addition, more than 40% of the points in all of the bridge Auto PCDs were ideally matched to their real-world PCDs. The C2C of more than 40% of points sampled from the automated gDTs were inferior to 2.5 cm against their real-world PCDs. The C2C of the majority of all Auto points (78.6%) were less than 10 cm. This implies that, in general, the method is capable of converting bridge point clusters into 3D solid models with full details.

The $\mathrm{C2C_{Auto}}$ of Bridges *3*, *5*, 6, 7, *8*, and *9* were smaller than those produced by the corresponding $\mathrm{C2C_{GT}}$s. However, not all the automated models were ideally modelled. The results also revealed that the major defects of the automated gDTs were due to sparse and occluded input points used in the method. When the input points are not sufficiently distributed on the bridge surface, the *ConcaveHull* algorithm cannot extract the hulls properly and is highly likely to generate local "holes" on the surface. This was especially true for *Bridges 2*, *4*, *5*, *6*, *8*, and 9, although the "holes" did not affect the overall performance of the proposed method for all of these bridges, and the $\mathrm{C2C_{Auto}}$ of some bridges remained small (*Bridges 5*, *6*, and *8*). The issues of local "holes" can be alleviated by using denser input data. The more points there are, the longer the time taken for fitting. Given the trivial time needed to run this method, the author contends that it will be a minor issue to add more input points for this method (one of the future work).

There were also some challenging scenarios, such as *Bridge 7* and *Bridge 10*. For *Bridge 7*, both the $\mathrm{C2C_{GT}}$ and that of the $\mathrm{C2C_{Auto}}$ were very big. This is because the web points of *Bridge 7*'s girders were largely missing in the real points. However, neither the manual operation nor the automated method was affected by these missing points and both generated the girders with full dimensions. The $\mathrm{C2C_{Auto}}$ was slightly smaller than the $\mathrm{C2C_{GT}}$, so the automated model was better modelled. By contrast, for *Bridge 10*, the GT model ($\mathrm{C2C_{GT}}$=5.5 cm) was much better modelled than the automated one ($\mathrm{C2C_{Auto}}$=13.5 cm). This is mainly because of the complex geometries underneath the deck slab of *Bridge 10*. The geometries of the oriented pier caps embedded in the deck slab were difficult to be effectively described using the proposed method.

The results of the $\mathrm{C2C_{GT}}$ also demonstrated that the performance of a human modeller is not always consistent. Although the models were all created by the same modeller, the accuracy rate varied. For example, the GT models of some bridges such as *Bridges 1*, *2*, *3*, *6* and *10* were very well modelled. The $\mathrm{C2C_{GT}}$ values were 4 cm, 6.4 cm, 5.2 cm, 4.9 cm and 5.5 cm, respectively. In contrast, *Bridge 5* had major manual modelling errors. Its entire upper deck slab surface was modelled higher than expected. This problem meant that the sampled deck slab points of the GT model largely drifted away from the real points. This, in turn, generated a large $\mathrm{C2C_{GT}}$. The results of *Bridge 5* demonstrated that the performance of manual modelling has an inevitable deviation, which is mainly due to the modeller's experience and judgement. Although the proposed method did not always outperform the manual method, it is more reliable and consistent.

Most of the automated gDTs were very well modelled and the results were consistent. There were no abnormal deviations. This demonstrated that the proposed method can rapidly reconstruct real labelled point clusters into accurate 3D solid models. In addition, those bridges whose horizontal alignments are strongly curved, e.g. *Bridge 8* and *9*, were better modelled than their GT models. This indicated that the proposed method performs better at modelling curved bridges than a human modeller does.

In summary, the experimental results prove that the proposed method can reconstruct four types of labelled point cluster making up a bridge into 3D solid models in IFC format. In addition, the modelling accuracy was evaluated using a distance-based quantitative metric. **Research questions 2 and 3** have been addressed.

6.3. Summary

This chapter presents a detailed description of the experiments conducted during this PhD thesis as well as all the results. The author starts by presenting the on-site data collection activities. Ten point cloud datasets of highway RC bridges around Cambridgeshire are collected for developing algorithms and testing the hypothesis of this research. All developed technologies have been integrated into a robust software prototype, which provides a handy user interface to enhance modellers' user experience. The author conducts the experiments on the methods proposed in Chapter 4 and 5 with the software prototype, respectively. The detected bridge labelled point clusters are compared against the GT point clusters and evaluated using standardized classification metrics. The method exhibits impressive performance on ten bridge point clouds in terms of Pr, R, and F1. This supports the hypothesis of Chapter 4. Then, the author validates the resulting gDTs at different resolution levels. The LOD 200 gDTs are evaluated using vertex-based metrics while LOD 250—300 gDTs are evaluated using C2C distance-based metrics. Both sets of gDTs approximate the real geometries of bridges. The accuracy of LOD 250—300 gDTs is consistent. On average, the LOD 250—300 gDTs achieve higher accuracy than the manually generated GT gDTs. This supports the hypothesis of Chapter 5. Interpretations of the detailed outcomes are given in Section 7.1.

7. CONCLUSIONS

In this PhD thesis, the author has answered the following **research questions**: **(1)** how to detect four types of RC bridge components in real PCD with occlusions and non-uniform distribution points, in the form of labelled clusters? **(2)** how to extract and use the geometric features to reconstruct the labelled point clusters in arbitrary shapes of real bridge PCD, into 3D solid models in IFC format? and **(3)** how to evaluate the accuracy of the resulting bridge gDTs? In this final chapter, the author first provides detailed interpretations of outcomes of each method in Section 7.1. The author then discusses in Section 7.2 the pros and cons of each method as well as that of the overall framework followed by discussing in Section 7.3 the implications of this research for the current practice as well as to the society. Finally, in Section 7.4, the author focuses on directions and recommendations for future research that stems from this PhD thesis.

7.1. Outcomes Interpretation

To answer research question 1, the author developed a slicing-based top-down object detection method in Chapter 4 to detect four component types in the form of point clusters in real bridge PCD featuring defects. To answer research question 2, the author followed the slicing strategy and proposed an IFC object fitting method in Chapter 5 to generate the gDT of an existing RC bridge, using the four types of point clusters. To answer research question 3, the author suggested using distance-based metrics in Chapter 6 to evaluate the spatial accuracy of the automated gDTs. The following texts demonstrate how much the research questions have been addressed through interpreting the experiments outcomes in detail.

- **Chapter 4 Results**

In Section 6.2.4.2, the author evaluated the proposed object detection method with the optimal parameters selected experimentally. The evaluation included two parts: a bounding-box-wise evaluation, and a point-wise evaluation.

The bounding-box-wise evaluation examined whether a detected point cluster is a TP or FP or FN using three criteria. The overall average Pr, R and F1 of bounding-box-wise detection were 100%, 98.5%, and 99.2%, respectively. All of the components of the ten bridge point clouds were detected in the form of labelled point clusters, except two components (pierCap1 and pierCap2) of *Bridge 1*, which were detected as FNs, because the values of ε of these two components were superior to 50%, i.e. they did not meet criterion C1 ($\varepsilon_{\text{pierCap1}}$=71.9%, $\varepsilon_{\text{pierCap2}}$=81.9%). However, there were not too many FP points in these two point clusters: only 4.4% and 8.6% points were FPs for pierCap 1 and pierCap 2, respectively. Compared to pierCap 3, whose FDR was 5.5% while $\varepsilon_{\text{pierCap3}}$ was only 17.6% (pierCap 3 was successfully detected as TP), the large ε value for pierCap 1 and pierCap 2 was attributed to the distant FPs. This means, for example, if there is only 1 FP point in a component and this FP point is distant from the TP points, then it is highly likely to detect this component as a FN. Hence, although bound-box-wise metrics give a quick impression of the detection results, they are too sensitive to the FP points, leading to imprecise performance assessment. The point-wise evaluation aims to give a precise description of the performance in this regard.

The point-wise evaluation consists of two average calculations for Pr, R, and F1: micro- and macro-averages. The former aggregates the contributions of all classes (component types) to compute the average metric whist the latter computes the metric independently for each class and then takes the average (hence treating all classes equally). In a multi-class classification setup, the micro-average is preferable if one suspects there is class imbalance (many more instances of one class than of other classes). This is especially true in bridge PCD, as normally a deck slab point cluster should contain much more points than other point clusters. However, this is not always true as any component could have unexpected occlusions and/or low point density so that one component does not necessarily have more points than the others. The author, therefore, used both calculations.

All bridges had both very high micro-average Pr/R/F1 (higher than 99%) and macro-average Pr, R, and F1 (higher than 95%) except *Bridge 7* whose detection rates (89.1% $\mathrm{F1}_{\mathrm{micro}}$, 90.8% $\mathrm{F}_{\mathrm{macro}}$) were lower. The overall micro-average Pr/R/F1 of the ten bridges was 98.4% and the overall macro-averages Pr, R, and F1 of the ten bridges were 99.2%, 97.3%, and 98.1%, respectively.

Bridge 7 had both lower micro- and macro- averages as its components had many either FP or FN points, especially the slab point cluster and the many girders. It is too soon to claim that the method performs less well on this bridge type than the slab bridge type, as *Bridge 7* is the only beam-slab bridge in the datasets. The author analysed in Section 6.2.4.3 that *Bridge 7* contains a significant parabolic vertical alignment which renders the method less effective. This is true because even though *Bridge 7* was segmented into two spans, the vertical alignment of each span is strongly curved. This characteristic led to an imperfect separation between the slab and the girders. The author contended that a further slicing procedure should be added in case the vertical alignment of the deck of a bridge is strongly curved. Another reason is that many girder points of *Bridge 7* were missing due to limited line-of-sight of the laser sensor and the small spacing between adjacent girders. This makes the method less effective to identify the girder height as very few points were captured for the upper flange. This problem can be overcome or alleviated through well-designed scanning with specific equipment.

Both *Bridge 8* and *Bridge 9* have obvious curved horizontal alignment, and the slab points of *Bridge 8* were very unevenly distributed. The method still exceled on these bridges and achieved high detection rate, with 99.5% $\mathrm{F1_{micro}}$, and 99.3% $\mathrm{F_{macro}}$ for *Bridge 8*, and 99.4% $\mathrm{F1_{micro}}$ and 99.0% $\mathrm{F_{macro}}$ for *Bridge 9.*

In general, the proposed object detection method can directly detect four types of bridge component in the form of point clusters. It has been proven to be robust on all the bridge point clouds, despite some remaining issues. It can tackle the natural skewness regardless of the shape of a component (e.g., *Bridges 8* and *9*). It can deal with common defects such as occlusions and varying point density (e.g., *Bridges 4, 6,* and *8*). The method has been proven to be consistent and reliable as the detection performance for all bridge data had a small variance (3.14% $\mathrm{F1_{micro-std}}$). The author therefore concluded that **research question 1** has been addressed.

- **Chapter 5 Results**

In Section 6.2.5.3 and Section 6.2.5.4, the author evaluated the proposed IFC object fitting method at different resolution levels with different metrics. The author used vertex-based metrics to evaluate the LOD 200 gDTs. The author first computed E_{dc}, which is the Euclidean distance between the centroid C_{gtBBox} of each GT gDTs and the automated one C_{autoBBox}. The overall average $\overline{E_{dc}}$ of all components of all the ten bridge datasets was 11.3 cm. Most bridges had small $\overline{E_{dc}}$ values whereas only a few of them were large, raising the overall $\overline{E_{dc}}$. Then, the author computed the FVR, which is the volume added or reduced compared to the GT volume. The results of the FVR of all components revealed that the method tends to generate larger bounding boxes than the manually generated ones. The overall average $\overline{\mathrm{FVR}}$ of all components of all the ten bridge datasets was 16.5%, meaning that 16.5% volume was added to every component by the proposed method. This is because manual model generation is not as accurate as the proposed method. Manual modelling is based on the subjective judgement of the modeller's naked eye and knowledge. A manual bounding box does not necessarily include all points belonging to a component because a modeller could easily miss certain points in the curved boundary surfaces (e.g. $\overline{\mathrm{FVR}}$=24.1% *Bridge 7*). Thus, it is easy to increase the false volume of, especially for components whose dimensions are big, such as a slab. Slabs are the components which had the biggest FVR in almost all bridge datasets. This can be attributed to the large dimension of the slab such that even a minor increase in the slab thickness could lead to a large $\mathrm{FVR_{slab}}$ (e.g. $\mathrm{FVR_{slab}}$=44% *Bridge 2,*

$\mathrm{FVR_{slab}}$=22.6% *Bridge 6*, and $\mathrm{FVR_{slab}}$=24.1% *Bridge 7*). Although the method generated bigger OBBs, the limited E_{dc} of the components indicated that the objects in gDTs were not displaced. In addition, the author computed the P2P distance, which is the Euclidean distance between each autoBBox vertex and the corresponding gtBBox vertex. The overall average $\overline{\mathrm{P2P}}$ of all components of all the ten bridge datasets was 23 cm. This indicated that the general location information of components was captured.

The statistics of the vertices and the P2P distance indicated that the generated LOD 200 gDTs can describe the approximate geometry and the direction of the structural components. These models can be used for examining the presence of components. The generated bounding boxes may serve as volumes for space occupancy. Although the resolution of a LOD 200 gDTs is not high enough for structural assessment, the approximate geometric information provided by the generic placeholders can be used for (1) preparing the tender and contract documents, including all survey or retrofit work needed to prepare quantities and guideline costings; (2) establishing stockpiles of materials; and (3) doing project control, and proactively planning for the technical demands of maintenance process. The following text interprets the evaluation results of the LOD 250—300 gDTs.

The author used Hausdorff distance-based metrics to evaluate the LOD 250—300 gDTs. To do so, the author computed the C2C distance between the *GT E* PCD (point cloud sampled from the GT gDT) and the real PCD, and the C2C distance between the Auto PCD (point cloud sampled from the resulting automated gDT) and the real PCD. The overall estimated distance between a compared point cloud (*GT E* PCD or Auto PCD) and a reference point cloud (real PCD) is the bigger one of the mutual $\overline{\mathrm{dist}}$, that is:

$\mathrm{C2C_{GT}} = \max(\overline{\mathrm{dist}}_{GT\ E\ /\mathrm{real}}, \overline{\mathrm{dist}}_{\mathrm{real}/GT\ E})$ for comparing the *GT E* PCD and the real PCD, and

$\mathrm{C2C_{Auto}} = \max(\overline{\mathrm{dist}}_{\mathrm{Auto}\ /\mathrm{real}}, \overline{\mathrm{dist}}_{\mathrm{real/Auto}})$ for comparing the Auto PCD and the real PCD.

The $\mathrm{C2C_{Auto}}$ values of six out of ten bridge gDTs automatically generated by the proposed method were smaller than their corresponding $\mathrm{C2C_{GT}}$ values. This means that most of the automatically generated gDTs were better modelled compared to the GT ones. Specifically, the $\mathrm{C2C_{Auto}}$ values of *Bridges 3*, *5*, *6*, *7*, *8*, *9* were smaller than their $\mathrm{C2C_{GT}}$ values.

The Auto PCD of *Bridge 3* contains only a small portion of mismatched points. The deck slab points were very well matched with minor mismatches found on the boundaries and some pier points. By contrast, more deck slab points in the *GT E* PCD of *Bridge 3* were mismatched. This demonstrated that the proposed method outperforms the manual operation.

Similarly, the whole deck slab surface points in *Bridge 5* were found to be mismatched to the real-world points. The upper slab surface was modelled a couple of centimetres higher than the real points. By contrast, the very small $\mathrm{C2C_{Auto}}$ (4.9 cm) indicated that the Auto PCD has very few mismatched points. Mismatched points were attributed to the local “holes” generated on the deck slab surface and at the boundaries. The rest of the points sampled from the automated gDT were ideally matched to the real-world PCD. This again demonstrated that the proposed method outperforms the manual operation. This also suggested that the quality of the manually generated gDTs is not consistent. The GT model accuracy largely depends on the rigorousness and discretion of the modeller.

Both the manual operation and the proposed method modelled *Bridge 6* to a very good quality. The modelling accuracy of the automated gDT ($\mathrm{C2C_{Auto}}$= 4.6 cm) was slightly better than the manual one ($\mathrm{C2C_{GT}}$ = 4.9 cm), despite some points still being mismatched. Most of these mismatched points in Auto PCD were concentrated in the middle part of the bridge, where the real-world PCD were very sparse, increasing the C2C. The proposed method also generated many rugged surfaces which might not actually exist in the real bridge. In addition, the piers in the automated gDT were not accurately modelled (see *Bridge 6* in Table 6.6). The actual cross-section of the pier of *Bridge 6* is not a simple rectangle. The proposed method did not properly capture the exact concave geometry of the pier. This problem could potentially be solved by using a smaller alpha value in the *ConcaveHull* algorithm.

Next, although the $\mathrm{C2C_{Auto}}$ (12.5 cm) of *Bridge 7* was smaller than its $\mathrm{C2C_{GT}}$ (15.7 cm), it was not insignificant. Both $\mathrm{C2C_{Auto}}$ and $\mathrm{C2C_{GT}}$ of *Bridge 7* were very big. It is not surprising that this was mainly due to the largely missing girder points in the real PCD of *Bridge 7*, whereas the missing points did not actually affect the manual operation and the proposed method. This is to say, both the modeller and the proposed method used engineering knowledge and inference to overcome the problem of occlusions, and successfully produced the girders with completed dimensions. This explains why both C2C distances of the *GT E* PCD and Auto PCD to the real-world PCD were very big. The relatively smaller $\mathrm{C2C_{Auto}}$ suggested that the automated model of *Bridge 7* was better.

The proposed method excelled at modelling *Bridge 8*. The $\mathrm{C2C_{Auto}}$ (3.7 cm) was much smaller than its $\mathrm{C2C_{GT}}$ (7.2 cm). Only a small portion of mismatched points were found at the boundaries of the extremities, where the transition curve begins. This is because these regions were occluded by on-site trees so that many real points were missing. By contrast, the mismatched points in the *GT E* PCD were mostly found on the upper surface of the deck slab and the boundaries. The results demonstrated again that the proposed method outperforms manual modelling, which exhibits difficulty in modelling the curved alignment deck slab, where the proposed method excels.

The topology of *Bridge 9* is similar to that of *Bridge 8*. *Bridge 9* contains an even stronger curved horizontal alignment than *Bridge 8* does. This challenge increased the difficulty of manual gDT generation as there were more mismatched points in the *GT E* PCD of *Bridge 9* ($\mathrm{C2C_{GT}}$ = 9.8 cm) than that of *Bridge 8* ($\mathrm{C2C_{GT}}$ = 7.2 cm). Most of the mismatched points in *GT E* PCD were concentrated in the two extremities of the curved deck slab surface, where both the horizontal and vertical curves became severe. This again suggested that it is quite difficult to model curved deck slab manually. By contrast, the proposed method was not affected by the curvature and continued modelling *Bridge 9* effectively ($\mathrm{C2C_{Auto}}$ = 5.6 cm), despite there being a small portion of mismatches at the boundaries. The following text discusses on the results of the other bridges, whose GT gDTs were better than the resulting automated gDTs ($\mathrm{C2C_{GT}}$ < $\mathrm{C2C_{Auto}}$).

For *Bridge 1*, the accuracy of the manual gDT ($\mathrm{C2C_{GT}}$ = 4.0 cm) and that of the Auto gDT were ($\mathrm{C2C_{Auto}}$ = 4.3 cm) very close to each other. The C2C of the GT gDT was slightly smaller. In general, both gDTs were very well modelled. Almost all the *GT E* points as well as the Auto points were correctly matched to the real points. Only a very limited number of mismatches were found in the Auto PCD at the boundaries where the real points were missing due to occlusions.

Similarly, both the GT gDT and the Auto gDT of *Bridge 2* were well modelled. The C2C of the GT gDT ($\mathrm{C2C_{GT}}$ = 6.4 cm) was slightly smaller than that of the Auto gDT ($\mathrm{C2C_{Auto}}$ = 7.3 cm). Most of the mismatched points in Auto PCD were locally concentrated at the boundaries or on the undulated deck slab surface.

A big portion of the slab points in the input data of *Bridge 4* was very sparse. The method didn't extract enough concave hulls to capture the slab geometry in that occluded region so that the Auto gDT was incomplete, and no points were sampled in that region. The author therefore evaluated *Bridge 4* (Table 6.24) after removing the partially modelled slice to avoid incorrect calculation of the C2C distance. The results showed that many points (the big green region shown in Table 6.24) derived from the Auto gDT were still distant from the real-world points ($\mathrm{C2C_{Auto}}$ = 9.4 cm). This was attributed to the sparse and non-uniformly distributed input data (of the upper slab surface points) used for the proposed method. These characteristics of the input data rendered the proposed method ineffective to capture slab geometries so that many grooves were generated on the surface, enlarging the local C2C distances. This problem can be avoided by using denser input data. By contrast, the manual gDT was better modelled ($\mathrm{C2C_{GT}}$ = 7.3 cm), but there were still many mismatched points in the deck slab. This was due to the varying deck slopes, which were difficult to describe well manually.

Bridge 10 was the most challenging case. The accuracy of the Auto gDT ($\mathrm{C2C_{Auto}}$ = 13.5 cm) was not as good as the GT gDT ($\mathrm{C2C_{GT}}$ = 5.5 cm). Many mismatched points in the Auto PCD were found under the deck slab. This is due to the complex geometry of the superstructure. *Bridge 10* is a diaphragm bridge, containing upstand diaphragms (embedded pier caps), which lie on the same level as the integrated beams. The upstand diaphragms are oriented based on the pairwise piers. The proposed method did not properly capture and describe these complex geometries of the upstand diaphragms. Thus, the Auto points were not well matched to the real-world points, leading to a large $\mathrm{C2C_{Auto}}$. This problem might be remedied by increasing the number of slices such that the diaphragm geometries can be better approximated by the method. The result of *Bridge 10* demonstrated that human modellers can deal with some really challenging scenarios that the current automated method cannot handle.

Last, the GT gDT generation (Table 6.5 and Table 6.6) revealed that the time spent on different bridges varies, depending on the complexity of the bridge and the quality of the raw bridge point cloud. The more complex the geometries, the more time is required to create a gDT. Likewise, the worse the quality of a bridge point cloud, the more manual hours are needed. The average modelling time spent on generating *GT E* was 99360 seconds (27.6 hours). By contrast, the performance of the proposed method was generally robust with regard to different bridges as well as to various data defects. Assuming that a total of 1 hour is required for manual cropping (data preparation for object detection) and manual refinement (data preparation for IFC object fitting), the overall time spent on generating a LOD 250—300 bridge gDT using the proposed framework was 4137.6 seconds (object detection 8.33 minutes + object fitting 37.8 seconds + manual work 1h) on average. A direct time savings of at least 95.8% was realized. **Research questions 2 and 3** have been answered.

7.2. Contributions & Limitations

- **Object detection method**

The impressive detection rates (micro-average F1 98.4%) acquired on ten highway RC bridge point clouds suggested that the proposed top-down slicing-based object detection method has addressed the research question 1 very well (Chapter 2, Section 2.4). The expected contributions achieved are in the following.

Con 1.1. The method dramatically reduces computational costs by breaking down a large set of point cloud into subsets. This way, large-scale object detection efficiency can be significantly improved without sacrificing precision. **Con 1.2.** The method is very efficient. It follows a top-down strategy in which high-level engineering information serves as guidance to directly produce four types of point clusters constituting a bridge point cloud without clustering low-level surfaces. **Con 1.3.** The method is robust as the variance of detection performance of ten highway RC bridge point clouds is small. **Con 1.4.** The method can deal with real-world bridges with varying elevation and curved horizontal alignment (*Bridges 8* and *9*). **Con 1.5.** The method can handle challenging scenarios, such as occlusions and locally variable point densities (*Bridges 4*, *6*, *8*, and *9*). However, the proposed object detection is not intended to be a cure-all. The limitations of this method are listed as following.

Pro 1.1. The bridge configurations studied are limited. More RC bridge datasets with various configurations, especially those with different girder and pier cap types, should be included and investigated in future studies. **Pro 1.2.** The method is not suitable for diaphragm bridges whose upstand diaphragms lie on the same level as the integrated lateral beams, such as *Bridge 10.* Note that *Bridge 10* was only segmented into two classes: deck and piers. However, the lateral beams and the oriented pier caps cannot be properly detected by the proposed method. **Pro 1.3.** The method cannot effectively tackle bridges whose vertical alignments are strongly curved (*Bridge 7*). **Pro 1.4.** The detection performance decreases when girder spacing becomes quite small, as many points might not be captured (*Bridge 7*). In this case, the method cannot have enough informative features to infer the girder type. **Pro 1.5.** The method can only deal with the four most important and highly detectable bridge components.

The hypothesis of the proposed method (Chapter 4, Section 4.2.1) has been experimentally validated. The results of the ten real-world bridge point clouds demonstrated that the detection of the four most important types of the structural component of typical RC bridges can be supported using the proposed solution. The first task of gDT generation (EURs 1 and 3) can be automated so that the overall modelling cost and effort will be reduced.

- **IFC object fitting method**

The high modelling accuracy ($\overline{\mathrm{C2C}}_{\mathrm{Auto}}$ 7.05 cm) acquired for the resulting gDTs of the ten highway RC bridges suggested that the proposed slicing-based object fitting method has addressed research questions 2 and 3 well (Chapter 2, Section 2.4). The expected contributions achieved are in the following.

Con 2.1. The method achieves enormous time savings for both LOD 200 and LOD 250—300 gDT generation, significantly outperforming the current manual practice. **Con 2.2.** The slicing-based object fitting method can accurately describe bridge geometries. **Con 2.3.** The method explicitly describes the component geometries using current IFC standards so that the resulting gDTs can be exchanged between project participants and be visualized with various IFC viewers. **Con 2.4.** The method can tackle curved horizontal alignments (*Bridge 8* and *Bridge 9*) as well as occlusions (*Bridge 7*). It can infer the girder type with full dimensions using the limited information from the occluded data. **Con 2.5.** Compared to manual operation, the method is robust and less liable to human errors. However, some problems remain unsolved, and are listed in the following.

Pro 2.1. The method is exploratory and imperfect. It tends to generate gaps and overlaps between two adjacent deck slabs when the alignment becomes strongly curved (Figure 7.1). **Pro 2.2.** The method cannot perfectly model a pier cap if its cross-section varies along the Z-axis (extruded direction). **Pro 2.3.** The method tends to generate undulating-surface sliced models especially when the input data is sparse (*Bridges 4 and 6*). However, the undulations may not exist in reality. This over-detailed information might not be necessary for end users. **Pro 2.4.** The method cannot describe the fine geometric details of a pier cross-section shape (*Bridge 6*). **Pro 2.5.** The method cannot properly capture geometric features to generate a model if there are not enough points in the input data (*Bridge 4*).

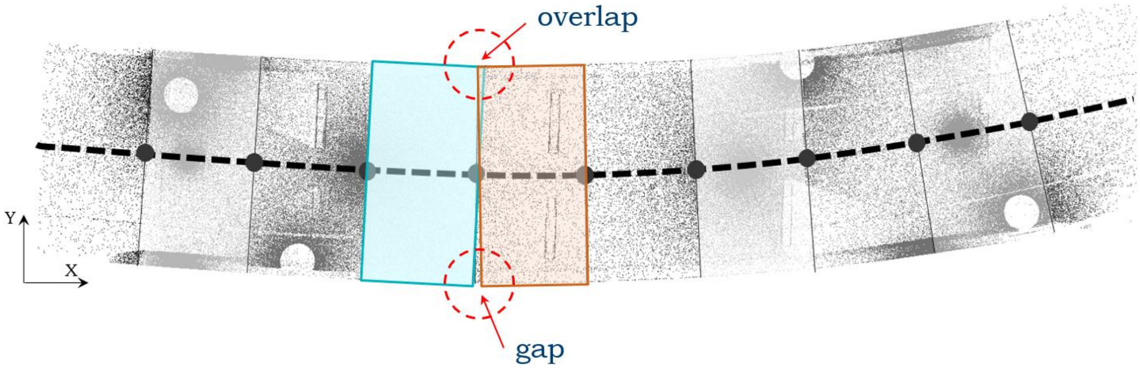


Figure 7.1 overlap and gap of the connection of two adjacent deck slabs

The hypothesis of the proposed method (Chapter 5, Section 5.2.1) has been experimentally validated. The resulting models of ten bridge point cluster datasets demonstrated that the four most important and detectable bridge component types can be modelled accurately using the proposed IFC object fitting method. The second task of gDT generation (EURs 1, 2, and 6) can be automated. Given that EUR 2 (Chapter 1, Section 1.3.2 and Appendix A) is the most time-consuming part in manual practice (95% manual modelling time), the overall gDT generation pipeline will be streamlined, and the modelling cost and effort will be reduced accordingly.

- **Overall framework**

The overall framework has been experimentally validated using ten real bridge point clouds. The bridge types studied in this thesis are the most representative ones in the UK (73%), although the configurations of these bridges are similar. The consistent results on these bridges suggest that the framework can reduce a lot of tedious repetitive manual work on modelling similar bridges.

Con 3.1. This thesis presents a novel top-down framework, which has been experimentally validated to be robust and efficient at detecting four structural component types in RC bridge PCD and generating their 3D solid models using current IFC standards. This is the first framework of its kind to achieve such high and reliable performance of gDT generation of existing bridges. It provides a solid foundation for future work in generating enriched bridge DTs. This research will move forward the state of DT research as well as promote the understanding of DT. **Con 3.2.** The proposed framework has proven experimentally how top-down reasoning can improve the efficiency and accuracy of detection and modelling. The outcome interpretations strongly suggest that data deficits like occlusions and varying point density, and challenging scenarios like complex bridge geometries can be overcome by using top-down engineering knowledge. **Con 3.3.** It is the first framework of its kind to suggest a cloud-to-cloud distance-based quantitative metric to evaluate the accuracy of the resulting automated gDTs and the GT gDTs, benchmarking against the real PCD.

Pro 3.1. However, the ten bridge datasets are not enough to fully validate the proposed framework. More bridge datasets with various configurations are needed to enhance the statistical validity of the framework with an increased confidence level and a reduced statistical margin of error. **Pro 3.2.** In addition, if the point cluster input data for the object fitting method contains too few points, then the performance of the framework will be affected. **Pro 3.3.** Finally, the framework cannot deal with a bridge with diaphragm bridges.

7.3. Impact for practice and society

The author concludes by briefly highlighting some of the implications of this research for the current practice of data collection, bridge gDT generation, bridge management as well as the social impacts of public safety and public expenses.

First, the proposed gDT generation framework will promote the proliferation of the non-contact TLS technique in the AEC/FM sector. The adoption of TLS methods can improve inspector safety by reducing exposure time spent working alongside traffic and at height.

Second, the enormous time savings on bridge gDT generation (95.8%) demonstrated the powerful computing performance of the proposed framework, significantly overriding the current manual practice. The use of this framework will decrease the repetitive work of manual gDT generation and enhance work efficiency. The proposed framework also outperforms the manual practice in terms of modelling accuracy. It overcomes the existing limitations while providing a reliable basis that can be integrated with future innovative technologies for up-to-date information of bridge condition. This way, the framework will contribute to largely streamline the whole PCD-to-gDT process so that the cost and benefit ratio will be improved. The total costs will be cut down as well.

Then, it is potentially possible to seamlessly integrate the technologies developed in this research into the BMS currently used in practice in the future. The greatest benefit is that the BMS databases will be enriched with information that is more accurate, more detailed, and more accessible. Bridge inspectors can view the complete status of a bridge using its DT, including its design, up-to-date status, works performed or planned, directly on their desktop. The benefits of this research are also projected to reduce costs for bridge owners by automating some procedures (both on-site and off-site) of the entire inspection process, enabling end-users to make accurate condition assessments and timely maintenance decisions. The BMS will be enhanced by virtue of using DT as the kernel over time, once bridge DTs become readily available with highway agencies.

Another benefit derived from this research is the enhancement of public safety. The vital consideration during the entire lifetime of a bridge is its health.

This research derives the knowledge needed for the first part of a National Digital Twin agenda, that is the ability to generate geometric virtual copies of existing structural. It is the prerequisite of the next step, that is to enrich these virtual copies with structured and unstructured data. This supports the generation of automated texturing mapping and condition assessment tools for bridge inspectors. Bridge health monitoring will benefit from the wide adoption of bridge DTs in the BMS, which provides the ability to compare the (as-is) DTs at any time in the BMS database in an automated manner. This will provide inspectors with a rapid, yet quantitative judgment of the condition of deteriorated bridges. This judgment can protect the general public, who could be in danger from a potential bridge collapse.

Last, but not least, the DT technologies derived from this research have the potentials to update current BMS, which can then support the preventative maintenance by the highway authority, by preventing costly downtime and outages. Inspecting and monitoring existing bridges in a virtual 3D space by using the PCD to automatically generate its DT can boost the work efficiency. This means that prioritization of bridge repair and maintenance programs can be quantitatively rationalized, resulting in considerable economic savings.

7.4. Recommendations for the future

The initially stated overarching objective of this research was to devise, implement, and benchmark a novel top-down framework that can generate the gDT of an existing RC bridge in IFC format, using PCD. The author believes she has largely achieved this in Chapter 6 by addressing three research questions, while recognizing the limitations of the proposed framework.

There are several gaps in the knowledge around bridge gDT in research that follow from the findings of this thesis and would benefit from further research, to extend and further enhance the framework the author has developed here:

First, it is difficult to find the accurate spine-curved alignment of existing bridges, whose deformed shape and as-is condition will furthermore increase this challenge. The author assumes the horizontal and vertical alignment are gap free in this research. However, a parallel slicing method provides the possibility for achieving gap-less alignment which will keep its tangential continuity. Gap-less slice segment models can be achieved by matching the end point (EndPoint) of the current segment and the starting point (StartPoint) of a subsequent segment. The horizontal alignment *IfcAlignment2DHorizontal* as well as the vertical alignment *IfcAlignment2DVertical* can be combined into a list of straight lines List<*IfcAlignment2DHorizontalSegment*>
and List<*IfcAlignment2DVerticalSegment*>, respectively. This development will be in accordance with the *IfcAlignment* model (Figure 7.2). Once the alignment curve is defined with curvature horizontally and vertically. The deck slab model can then be constructed by sweeping along a horizontal curve using *IfcSectionedSolidHorizontal* (buildingSMART, 2014) or along an arbitrary 3D curve using *IfcFixedReferenceSweptAreaSolid*. This way, the gap and overlap issue (Section 7.2, Pro 2.4) can be solved. In addition, it is highly likely to have the same cross-sections for adjacent slice models. A method for how to average the geometries of cross-sections to produce a common one used for several adjacent slices would be useful. Further slicing procedures should be conducted to tackle bridges with strongly curved vertical alignments.

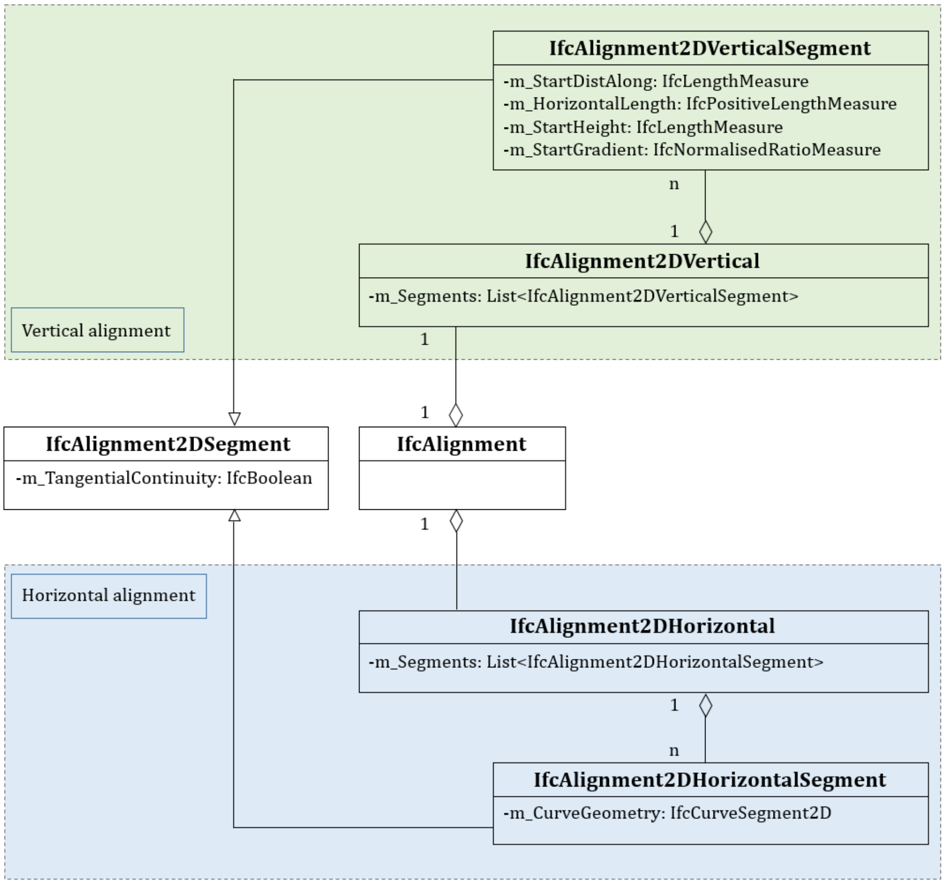

IfcAlignment2DVerticalSegment
-m_StartDistAlong: IfcLengthMeasure
-m_HorizontalLength: IfcPositiveLengthMeasure
-m_StartHeight: IfcLengthMeasure
-m_StartGradient: IfcNormalisedRatioMeasure
n
1
IfcAlignment2DVertical
-m_Segments: List<IfcAlignment2DVerticalSegment>
Vertical alignment
1
1
IfcAlignment2DSegment
-m_TangentialContinuity: IfcBoolean
IfcAlignment
1
Horizontal alignment
IfcAlignment2DHorizontal
-m_Segments: List<IfcAlignment2DHorizontalSegment>
1
n
IfcAlignment2DHorizontalSegment
-m_CurveGeometry: IfcCurveSegment2D

Figure 7.2 IFC alignment model (adapted from Amann et al., 2015)

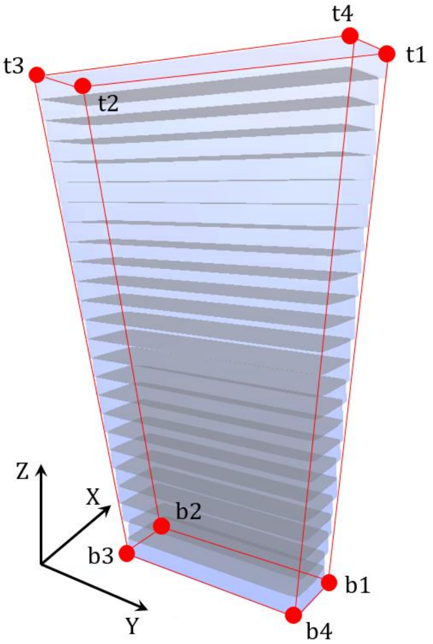


Figure 7.3 A single pier model defined by 8 vertices

Second, the method does not work perfectly for a pier cap whose outline shape or scale varies along the Z-axis, i.e. the extruded direction. This problem may be solved by adding a slicing procedure at the beginning. The slicing could be either along the Z-axis or along the local y-axis.

Third, it is preferable to describe a pier as a single solid model instead of using a stacked slice representation. Research is needed to develop a method which can replace the stacked slices by eight vertices: four upper vertices and four bottom vertices, respectively (Figure 7.3). A full analysis is required to quantify the fitting accuracy.

Fourth, enlarging the bridge database with various configurations and detecting more components is needed. Further research might detect secondary components, such as Category 2 and 3 elements (Appendix C), including transverse girder (C1/2), abutment (C2), wing walls (C2), safety barriers/handrails, and so on. Research could explore how to integrate additional layers to infer more components. Specifically, a concrete deck can have a very large transverse span between main girders. This large unsupported width can lead to buckling of the slab in compression unless it is prevented from occurring by placing transverse girders. This knowledge could be integrated into Step 4 of the object detection method to infer the presence of the traverse girder. Abutment and wing walls are located at the two extremities of a bridge. They are either vertical or inclined and have large exposed surface, which can be recognized using existing plane detection method. The safety barriers/handrails of a bridge are located above the deck upper surface or on top of the wing wall. They are often standardized tubes with much smaller size and volume compared to the structural components. The detection of these elements could be done by adding a step before the Step1 of the object detection method.

Next, more methodology work is needed on how to deal with diaphragm bridges with complex superstructure geometries. For example, it may be possible to add more detection procedures in Step 4 of the object detection method.

Then, research to develop ad-hoc scanning or specific methodology in order to tackle the problem of limited line-of-sight to the small girder spacing and the underneath deck area would be very beneficial.

Further exploration of the influence of the parameters to the overall performance is necessary. Does the number of slices impact the performance and if so, by how much? Would it be better to increase or decrease the alpha value of *ConcaveHull* and if so, by how much? Would it be better to describe the exact geometry of the model surface or smooth the unnecessary undulations and if so, by how much? And how to smooth? All these in-depth exploratory works must be guided by high-level and detailed LOD of (as-is) gDTs, which however, are missing in the literature.

Furthermore, using denser input data to feed the IFC object fitting method would result in an even more accurate gDT. This hypothesis could be tested by further research devised to identify the relationship between the size of the input point cluster and the quality of the resulting gDT.

Finally, this thesis only meets the EURs 1, 2, 3, and 6 (Section 1.3.2) to a certain extend. However, other mandatory EURs. i.e. EURs 4 and 5, must be met in the future in order to generate genuinely semantically enriched bridge DTs.

8. References

AASHTO. (2011). *The Manual for Bridge Evaluation. American Association of State Highway and Transportation Officials*. https://doi.org/10.1017/CBO9781107415324.004

AASHTO. AASHTO LRFD Bridge Design Specifications, American Association of State Highway and Transportation Officials, 8th Edition § (2017).

AASHTOWare. (2018). AASHTOWare. Available at https://www.aashtoware.org/, accessed 18 September 2018.

Adan, A., & Huber, D. (2011). 3D reconstruction of interior wall surfaces under occlusion and clutter. *Proceedings - 2011 International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission, 3DIMPVT 2011*, 275–281. https://doi.org/10.1109/3DIMPVT.2011.42

Agapaki, E., & Brilakis, I. (2018). State-of-Practice on As-Is Modelling of Industrial Facilities. In *EG-ICE 25th International Workshop on Intelligent Computing in Engineering* (pp. 103–124). https://doi.org/10.1007/978-3-319-91635-4_6

Ahmed, M. F., Haas, C. T., & Haas, R. (2014). Automatic Detection of Cylindrical Objects in Built Facilities. *Journal of Computing in Civil Engineering*, *28*(3), 1–11. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000329.

Akenine-Möller, T., Haines, E., Hoffman, N., Pesce, A., Iwanicki, M., & Hillaire, S. (2018). *Real-time rendering: Fourth edition.*

Amann, J., Singer, D., & Borrmannm André. (2015). Extension of the upcoming IFC alignment standard with cross sections for road design. In *Proceedings of the ICCBEI.* Tokyo, Japan.

Anil, E. B., Akinci, B., & Huber, D. (2011). Representation Requirements of As-Is Building Information Models Generated from Laser Scanned Point Cloud Data. In *Proceedings of the International Symposium on Automation and Robotics in Construction (ISARC)*. Seoul, Korea. https://doi.org/10.22260/ISARC2011/0063

Anil, E. B., Tang, P., Akinci, B., & Huber, D. (2013). Deviation analysis method for the assessment of the quality of the as-is Building Information Models generated from point cloud data. *Automation in Construction*, *35*, 507–516. https://doi.org/10.1016/j.autcon.2013.06.003

Arikan, M., Schwärzler, M., Flöry, S., Wimmer, M., & Maierhofer, S. (2013). O-snap:Optimization-based snapping for modeling architecture. *ACM Transactions on Graphics*, *32*(1), 1–15. https://doi.org/10.1145/2421636.2421642

Armeni, I., Sener, O., Zamir, A. R., Jiang, H., Brilakis, I., Fischer, M., & Savarese, S. (2016). 3D Semantic Parsing of Large-Scale Indoor Spaces. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1534–1543). IEEE. https://doi.org/10.1109/CVPR.2016.170

ASCE. (2013). 2013 Report Card for America's Infrastructure, Bridges. *ASCE.*

ASCE. (2017). 2017 Report Card for America's Infrastructure, Bridges. *ASCE.*

Aspert, N., Santa-Cruz, D., & Ebrahimi, T. (2002). MESH: Measuring errors between surfaces using the Hausdorff distance. In *Proceedings - 2002 IEEE International Conference on Multimedia and Expo, ICME 2002*. https://doi.org/10.1109/ICME.2002.1035879

Autodesk Revit 2016. (2016). *Autodesk Revit 2016. Available at https://www.autodesk.co.uk/products/revit/overview, accessed 18 September, 2018.*

Ballard, D. H. (1981). Generalizing the Hough transform to detect arbitrary shapes. *Pattern Recognition*. https://doi.org/10.1016/0031-3203(81)90009-1

BANAGHER. (2018). Bridge Beam Manual, 2nd Edition. Available at https://bancrete.com/bridge-beam-manual/, accessed 18 September, 2018.

Barbosa, F., Woetzel, J., Mischke, J., João Ribeirinho, M., Sridhar, M., Parsons, M., … Brown, S. (2017). Reinventing Construction: A Route to Higher Productivity. *Mckinsey & Company*. Retrieved from https://www.mckinsey.com/~/media/McKinsey/Industries/Capital Projects and Infrastructure/Our Insights/Reinventing construction through a productivity revolution/MGI-Reinventing-construction-A-route-to-higher-productivity-Full-report.ashx

BBC News. (2018a). Italy bridge collapse: What we know so far. Retrieved from https://www.bbc.co.uk/news/world-europe-45193452

BBC News. (2018b). Mamia bridge collapse: Death toll expected to rise from six. Retrieved from https://www.bbc.co.uk/news/world-us-canada-43434746

Belsky, M. (2015). A framework for semantic enrichment of IFC building models. *PhD Thesis*.

Belsky, M., Eastman, C., Sacks, R., Venugopal, M., Aram, S., & Yang, D. (2014). Interoperability for precast concrete building models. *PCI JOURNAL*. https://doi.org/10.15554/pcij.03012014.144.155

Belsky, M., Sacks, R., & Brilakis, I. (2016). Semantic Enrichment for Building Information Modeling. *Computer-Aided Civil and Infrastructure Engineering*, *31*(4), 261–274. https://doi.org/10.1111/mice.12128

Belton, D., & Lichti, D. (2006). Classification and segmentation of terrestrial laser scanner point clouds using local variance information. *ISPRS Commission V Symposium 'Image Engineering and Vision Metrology*, *Part 5*, 44–49.

Besl, P. J., & Jain, R. C. (1988). Segmentation Through Variable-Order Surface Fitting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*. https://doi.org/10.1109/34.3881

BIMForum. (2018). Level of Development Specification. Retrieved from https://bimforum.org/wp-content/uploads/2018/07/BIMForum-LOD-2018_Spec-Part-1_and_Guide_PUB-DRAFT.pdf

Borenstein, E., & Ullman, S. (2008). Combined Top-Down / Bottom-Up Segmentation, *30*(12), 1–17.

Borrmann, A., Amann, J., Chipman, T., Hyvärinen, J., Liebich, T., Muhič, L., … Scarponcini, P. (2017). IFC Infra Overall Architecture Project Documentation and Guidelines. Technical Report. *BuildingSMART*. Retrieved from https://buildingsmart.org/wp-content/uploads/2017/07/08_bSI_OverallArchitecure_Guidelines_final.pdf

Borrmann, A., Beetz, J., Koch, C., Liebich, T., & Muhic, S. (2018). Industry Foundation Classes: A Standardized Data Model for the Vendor-Neutral Exchange of Digital Building Models. In A. Borrmann, M. König, C. Koch, & J. Beetz (Eds.), *Building Information Modeling: Technology Foundations and Industry Practice* (pp. 81–126). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-92862-3_5

Borrmann, D., Elseberg, J., Lingemann, K., & Nüchter, A. (2011). The 3D Hough Transform for plane detection in point clouds: A review and a new accumulator design. *3D Research*, *2*(2), 3. https://doi.org/10.1007/3DRes.02(2011)3

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*(Part B), 201–213. https://doi.org/10.1016/j.autcon.2014.05.014

Bosché, F., Haas, C. T., & Akinci, B. (2009). Automated Recognition of 3D CAD Objects in Site Laser Scans for Project 3D Status Visualization and Performance Control. *Journal of Computing in Civil Engineering*, *23*(6), 311–318. https://doi.org/10.1061/(ASCE)0887-3801(2009)23:6(311)

Brilakis, I., Lourakis, M., Sacks, R., Savarese, S., Christodoulou, S., Teizer, J., & Makhmalbaf, A. (2010). Toward automated generation of parametric BIMs based on hybrid video and laser scanning data. *Advanced Engineering Informatics*, *24*(4), 456–465. https://doi.org/10.1016/j.aei.2010.06.006

Buckley, B., & Logan, K. (2017). The Business Value of BIM for Infrastructure 2017. *Dodge Data & Analytics*, 1–68. Retrieved from https://www2.deloitte.com/content/dam/Deloitte/us/Documents/finance/us-fas-bim-infrastructure.pdf

Budroni, A., & Boehm, J. (2009). Toward automatic reconstruction of interiors from laser data. In *3D-ARCH*.

Budroni, A., & Boehm, J. (2010). Automated 3D Reconstruction of Interiors from Point Clouds. *International Journal of Architectural Computing*, *08*(01), 55–73. https://doi.org/10.1260/1478-0771.8.1.55

buildingSMART. (2014). IfcSectionedSolidHorizontal. *IFC4x1*. Retrieved from http://www.buildingsmart-tech.org/ifc/IFC4x1/RC3/html/schema/ifcgeometricmodelresource/lexical/ifcsectionedsolidhorizontal.htm

buildingSMART. (2016a). IfcAdvancedBrep. *IFC4 Add2*. Retrieved from http://www.buildingsmart-tech.org/mvd/IFC4Add2TC1/DTV/1.1/html/schema/ifcgeometricmodelresource/lexical/ifcadvancedbrep.htm

buildingSMART. (2016b). IfcExtrudedAreaSolid. *IFC4 Add2*. Retrieved from http://www.buildingsmart-tech.org/ifc/IFC2x3/TC1/html/ifcgeometricmodelresource/lexical/ifcextrudedareasolid.htm

buildingSMART. (2018). International home of openBIM. Retrieved from http://www.buildingsmart-tech.org/

Calavera, J., De Chefdebien, A., Fernández-Ordóñez, D., Gasperi, A., Ley, J., Mönnig, F., … Van Acker, A. (2004). Precast Concrete Bridges: State-of-the-art report. *The International Federation for Structural Concrete: FIB, Task Group 6.4*. Retrieved from https://www.istructe.org/fibuk/files/fib_bull29_nmg.pdf

Carr, J. C., Beatson, R. K., McCallum, B. C., Fright, W. R., McLennan, T. J., & Mitchell, T. J. (2003). Smooth surface reconstruction from noisy range data. In *Proceedings of the 1st international conference on Computer graphics and interactive techniques in Austalasia and South East Asia - GRAPHITE ’03* (pp. 119–297). https://doi.org/10.1145/604492.604495

Chen, J., Zhang, C., & Tang, P. (2017). Geometry-based optimized point cloud compression methodology for construction and infrastructure management. In *Congress on Computing in Civil Engineering, Proceedings* (pp. 377–385).

Cignoni, P., Rocchini, C., & Scopigno, R. (1998). Metro: Measuring Error on Simplified Surfaces. *Computer Graphics Forum*. https://doi.org/10.1111/1467-8659.00236

ClearEdge3D. (2017). Structure Modelling Tools. Available at https://www.clearedge3d.com/, accessed 18 September, 2018.

Corbetta, M., & Shulman, G. L. (2002). Control of goal-directed and stimulus-driven attention in the brain. *Nature Reviews Neuroscience*, *3*(3), 201–215. https://doi.org/10.1038/nrn755

Deng, Y., Cheng, J. C. P., & Anumba, C. (2016). Mapping between BIM and 3D GIS in different levels of detail using schema mediation and instance comparison. *Automation in Construction*, *67*(July), 1–21. https://doi.org/10.1016/j.autcon.2016.03.006

Department for Transport. (2011). *Tackling £1 billion cost of motorway closures. Available at https://www.gov.uk/government/news/tackling-1billion-cost-of-motorway-closures, accessed 18 September, 2018.*

Design Manual for Roads and Bridges. (2007). Inspection of Highway Structures. Available at http://www.standardsforhighways.co.uk/ha/standards/dmrb/vol3/index.htm, accessed 18 September, 2018.

Díaz-Vilariño, L., Khoshelham, K., Martínez-Sánchez, J., & Arias, P. (2015). 3D modeling of building indoor spaces and closed doors from imagery and point clouds. *Sensors (Basel, Switzerland)*, *15*(2), 3491–3512. https://doi.org/10.3390/s150203491

Dimitrov, A., & Golparvar-Fard, M. (2015). Segmentation of building point cloud models including detailed architectural/structural features and MEP systems. *Automation in Construction*, *51*(C), 32–45. https://doi.org/10.1016/j.autcon.2014.12.015

Dimitrov, A., Gu, R., & Golparvar-Fard, M. (2016). Non-Uniform B-Spline Surface Fitting from Unordered 3D Point Clouds for As-Built Modeling.

Computer-Aided Civil and Infrastructure Engineering, *31*(7), 483–498. https://doi.org/10.1111/mice.12192

Dore, C., & Murphy, M. (2014). Semi-automatic generation of as-built BIM façade geometry from laser and image data. *Journal of Information Technology in Construction*, *19*(January), 20–46.

Dorninger, P., & Pfeifer, N. (2008). A Comprehensive Automated 3D Approach for Building Extraction, Reconstruction, and Regularization from Airborne Laser Scanning Point Clouds. *Sensors*, *8*(11), 7323–7343. https://doi.org/10.3390/s8117323

Douglas, D. H., & Peucker, T. K. (2011). Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or its Caricature. In *Classics in Cartography: Reflections on Influential Articles from Cartographica*. https://doi.org/10.1002/9780470669488.ch2

Douillard, B., Underwood, J., Kuntz, N., Vlaskine, V., Quadros, a., Morton, P., & Frenkel, a. (2011). On the segmentation of 3D LIDAR point clouds. *2011 IEEE International Conference on Robotics and Automation*, 2798–2805. https://doi.org/10.1109/ICRA.2011.5979818

Duda, R. O., & Hart, P. E. (1972). Use of the Hough transformation to detect lines and curves in pictures. *Communications of the ACM*. https://doi.org/10.1145/361237.361242

Eastman, C. M., Panushev, I., Sacks, R., Venugopal, M., Aram, V., & See, R. (2011). A Guide for Development and Preparation of a National BIM Exchange Standard. Technical report. Retrieved from http://dcom.arch.gatech.edu/pcibim/documents/IDM-MVD_Development_Guide_v4.pdf

Eastman, C., Teicholz, P., Sacks, R., & Liston, K. (2011). *BIM handbook: A guide to building information modeling for owners, managers, designers, engineers and contractors*. *Wiley* (Vol. 2). https://doi.org/10.1002/9780470261309

Eck, M., DeRose, T., Duchamp, T., Hoppe, H., Lounsbery, M., & Stuetzle, W. (1995). Multiresolution analysis of arbitrary meshes. *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH '95*. https://doi.org/10.1145/218380.218440

Efron, B. (1979). Bootstrap Methods: Another Look at the Jackknife. *The Annals of Statistics*. https://doi.org/10.1214/aos/1176344552

Engel, A. K., Fries, P., & Singer, W. (2001). Dynamic predictions: Oscillations and synchrony in top–down processing. *Nature Reviews Neuroscience*, *2*(10), 704–716. https://doi.org/10.1038/35094565

Fanning, B., Clevenger, C. M., Ozbek, M. E., & Mahmoud, H. (2015). Implementing BIM on Infrastructure: Comparison of Two Bridge Construction Projects. *Practice Periodical on Structural Design and Construction*, *20*(4), 04014044. https://doi.org/10.1061/(ASCE)SC.1943-5576.0000239

Faro. (2012). FARO SCENE, FARO's 3D Documentation Software for terrestrial and handheld Scanners. Available at https://www.faro.com/en-gb/products/construction-bim-cim/faro-scene/, accessed 18 September, 2018.

Faro. (2013). THE NEW FARO LASER SCANNER FOCUS3D X 330: THE PERFECT INSTRUMENT FOR 3D DOCUMENTATION AND LAND SURVEYING. Available at https://www.faro.com/en-gb/news/the-new-faro-laser-scanner-focus3d-x-330-the-perfect-instrument-for-3d-documentation-and-land-surveying-.

FHWA. (2018). Bridge Preservation Guide Maintaining a Resilient Infrastructure to Preserve Mobility. *Federal Highway Administration.* Retrieved from https://www.fhwa.dot.gov/bridge/preservation/guide/guide.pdf

FHWA - Federal Highway Administration. (2012a). Bridge Inspector' s Reference Manual, *1*. Retrieved from https://www.dot.state.mn.us/bridge/pdf/insp/birm/birmchapt0-cover.pdf

FHWA - Federal Highway Administration. (2012b). Estimated 2012 Costs to Replace or Rehabilitate Structurally Deficient Bridges. Available at https://www.fhwa.dot.gov/bridge/nbi/sd2012.cfm, accessed 18 September, 2018.

Fischler, M. a., & Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, *24*(6), 381–395. https://doi.org/10.1145/358669.358692

Flaig, K. D., & Lark, R. J. (2000). The development of UK bridge management systems. *Proceedings of the Institution of Civil Engineers - Transport*, *141*(2), 99–106. https://doi.org/10.1680/tran.2000.141.2.99

Foltz, L. B. (2000). 3D Laser Scanner Provides Benefits for PennDOT Bridge and Rockface Surveys. *Professional Surveyor*, *20*(5), 22–28.

Fonseca, M. J., & Jorge, J. A. (2000). Using fuzzy logic to recognize geometric shapes interactively. In *Ninth IEEE International Conference on Fuzzy Systems. FUZZ- IEEE 2000 (Cat. No.00CH37063)* (Vol. 1, pp. 291–296). IEEE. https://doi.org/10.1109/FUZZY.2000.838674

Gerardo-castro, M. P., Peynot, T., Ramos, F., & Fitch, R. (2014). Robust Multiple-Sensing-Modality Data Fusion using Gaussian Process Implicit Surfaces. *IEEE 17th Int. Conf. on Information Fusion*, 1–8. https://doi.org/10.13140/2.1.4799.8082

Gerardo-Castro, M., Peynot, T., & Ramos, F. (2013). Laser-Radar Data Fusion with Gaussian Process Implicit Surfaces. *Proc. of Int. Conf. on Field And Service Robotics*. https://doi.org/10.1007/978-3-319-07488-7_20

Gerbert, P., Castagnino, S., Rothballer, C., Renz, A., & Filitz, R. (2016). Digital in Engineering and Construction. *Boston Consulting Group,The Transformative Power of Building Informaiton Modeling*. Retrieved from http://futureofconstruction.org/content/uploads/2016/09/BCG-Digital-in-Engineering-and-Construction-Mar-2016.pdf

Graham, W. (2014). Structural Health Monitoring of Bridges. *PhD Thesis*.

Gyetvai, N., Truong-Hong, L., & Laefer, D. F. (2018). Laser scan-based structural assessment of wrought iron bridges: Guinness Bridge, Ireland. *Proceedings of the Institution of Civil Engineers - Engineering History and Heritage*, *171*(2), 76–89. https://doi.org/10.1680/jenhh.17.00018

HA. (1980). Section 41(1), Highways Act 1980. Available at https://www.legislation.gov.uk/ukpga/1980/66/section/41, accessed 18 September, 2018.

Hähnel, D., Burgard, W., & Thrun, S. (2003). Learning compact 3D models of indoor and outdoor environments with a mobile robot. *Robotics and Autonomous Systems*, *44*(1), 15–27. https://doi.org/10.1016/S0921-8890(03)00007-1

Hassanein, A. S., Mohammad, S., Sameer, M., & Ragab, M. E. (2015). A Survey on Hough Transform, Theory, Techniques and Applications. *International Journal of Computer Science Issues*, *IJCSI 12*(1).

Highways England. (1994). Expansion Joints for Use in Highway Bridge Decks. Retrieved from http://www.standardsforhighways.co.uk/ha/standards/dmrb/vol2/section3/bd3394.pdf

Highways England. (2018a). Design Manual for Roads and Bridges: Volume 3 Highway Structures: Inspection & Maintenance. Available at http://www.standardsforhighways.co.uk/ha/standards/dmrb/vol3/index.htm, accessed 18 September, 2018.

Highways England. (2018b). Design Manual for Roads and Bridges: Volume 6 Road Geometry. Available at http://www.standardsforhighways.co.uk/ha/standards/dmrb/vol6/index.htm, accessed 18 September, 2018.

Highways England. (2018c). Design Manual for Roads and Bridges. Retrieved from http://www.standardsforhighways.co.uk/ha/standards/dmrb/index.htm

HM Government. (2013). Construction 2025. *Industrial Strategy: Government and Industry in Partnership*. Retrieved from https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/210099/bis-13-955-construction-2025-industrial-strategy.pdf

Hobi, M. L., & Ginzler, C. (2012). Accuracy Assessment of Digital Surface Models Based on WorldView-2 and ADS80 Stereo Remote Sensing Data. *Sensors*, *12*(5), 6347–6368. https://doi.org/10.3390/s120506347

Holance, Objorke, Ikeough, & Iluvata82. (2016). Helix Toolkit - 3D toolkit for .NET. Available at https://github.com/helix-toolkit/helix-toolkit, accessed 18 September, 2018.

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J., & Stuetzle, W. (1992). Surface reconstruction from unorganized points. *ACM SIGGRAPH Computer Graphics*, *26*(2), 71–78. https://doi.org/10.1145/142920.134011

Hough, P. V. C. (1959). Machine analysis of bubble chamber pictures. *2nd International Conference on High-Energy Accelerators and Instrumentation.*

Hough, P. V. C. (1962). Method and means for recognizing complex patterns. *US Patent 3,069,654*. https://doi.org/10.1007/s10811-008-9353-1

Hsiao, E., & Hebert, M. (2012). Occlusion reasoning for object detection under arbitrary viewpoint. In *2012 IEEE Conference on Computer Vision and Pattern Recognition* (pp. 3146–3153). IEEE. https://doi.org/10.1109/CVPR.2012.6248048

Hüthwohl, P., Brilakis, I., Borrmann, A., & Sacks, R. (2018). Integrating RC Bridge Defect Information into BIM Models. *Journal of Computing in Civil Engineering*, *32*(3), 04018013. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000744

Huttenlocher, D. P., Klanderman, G. A., & Rucklidge, W. J. (1993). Comparing images using the Hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *15*(9), 850–863. https://doi.org/10.1109/34.232073

IfcOpenShell.org. (2018). IfcOpenShell, the open source ifc toolkit and geometry engine. Available at http://ifcopenshell.org/, accessed 18 September, 2018.

Ismail, A., Srewil, Y., Scherer, R., & Mansperger, T. (2016). Semantic Enrichment and Multimodel Data Exchange Approach for CFD Analysis of Bridges. In *Proceedings of the 23rd International Workshop of the European Group for Intelligent Computing in Engineering, EG-ICE*.

Jeong, W., Kähler, K., Haber, J., & Seidel, H. (2002). Automatic generation of subdivision surface head models from point cloud data. In *Proceedings of Graphics Interface* (pp. 181–188). https://doi.org/10.20380/GI2002.21

Ji, Y., Borrmann, A., Beetz, J., & Obergrießer, M. (2013). Exchange of Parametric Bridge Models Using a Neutral Data Format. *Journal of Computing in Civil Engineering*, *27*(6), 593–606. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000286

Jung, J., Hong, S., Jeong, S., Kim, S., Cho, H., Hong, S., & Heo, J. (2014). Productive modeling for development of as-built BIM of existing indoor structures. *Automation in Construction*, *42*, 68–77. https://doi.org/10.1016/j.autcon.2014.02.021

Kawashima, K., Kanai, S., & Date, H. (2014). As-built modeling of piping system from terrestrial laser-scanned point clouds using normal-based region growing. *Journal of Computational Design and Engineering*, *1*(1), 13–26. https://doi.org/10.7315/JCDE.2014.002

Kedar, A. (2016). *SeeBridge Project Document: Criteria for evaluation of complete SeeBridge System*. Retrieved from https://technionmail-my.sharepoint.com/personal/cvsacks_technion_ac_il/Documents/Virtual Construction Lab/SeeBridge/WP1/Criteria/Deliverable12SeeBridgeEvaluationCriteria . pdf?slrid=578a909e-00ff-6000-c3fd-8c935ec273ce

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., McGovern, S., Belsky, M., Middleton, C., & Brilakis, I. (2016). A Suitability Analysis of Precast Components for Standardized Bridge Construction in the United Kingdom. *Procedia Engineering*, *164*, 188–195. https://doi.org/10.1016/j.proeng.2016.11.609

Klasing, K., Althoff, D., Wollherr, D., & Buss, M. (2009). Comparison of surface normal estimation methods for range sensing applications. In *2009 IEEE International Conference on Robotics and Automation* (pp. 3206–3211). IEEE. https://doi.org/10.1007/978-3-319-53727-6

Kobryń, A. (2017). *Transition Curves for Highway Geometric Design. Springer Tracts on Transportations and Traffic* (Vol. 14). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-53727-6

Koch, C., Paal, S. G., Rashidi, A., Zhu, Z., König, M., & Brilakis, I. (2014). Achievements and Challenges in Machine Vision-Based Inspection of Large Concrete Structures. *Advances in Structural Engineering*, *17*(3), 303–318. https://doi.org/10.1260/1369-4332.17.3.303

Kokkinos, I., Maragos, P., & Yuille, A. (2006). Bottom-Up &amp; Top-down Object Detection using Primal Sketch Features and Graphical Models. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2 (CVPR'06)* (Vol. 2, pp. 1893–1900). IEEE. https://doi.org/10.1109/CVPR.2006.74

Koppula, H. S., Anand, A., Joachims, T., & Saxena, A. (2011). Semantic Labeling of 3D Point Clouds for Indoor Scenes. *Neural Information Processing Systems*, 1–9.

Kwon, S.-W., Bosche, F., Kim, C., Haas, C. T., & Liapi, K. A. (2004). Fitting range data to primitives for rapid local 3D modeling using sparse range point clouds. *Automation in Construction*, *13*(1), 67–81. https://doi.org/10.1016/j.autcon.2003.08.007

Laefer, D. F., & Truong-Hong, L. (2017). Toward automatic generation of 3D steel structures for building information modelling. *Automation in Construction*, *74*, 66–77. https://doi.org/10.1016/j.autcon.2016.11.011

Laing, R., Leon, M., Mahdjoubi, L., & Scott, J. (2014). Integrating Rapid 3D Data Collection Techniques to Support BIM Design Decision Making. *Procedia Environmental Sciences*, *22*, 120–130. https://doi.org/10.1016/j.proenv.2014.11.012

Lebegue, E., Fiês, B., & Gual, J. (2012). IFC-Bridge V3 Data Model - IFC4, Edition R1.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444. https://doi.org/10.1038/nature14539

Lee, G., Mathews, N., & Yoders, J. (2014). The business value of BIM for construction in major global markets. *McGraw Hill Construction, SmartMarket Report*, 60. Retrieved from http://static.autodesk.net/dc/content/dam/autodesk/www/solutions/building-information-modeling/construction/business-value-of-bim-for-construction-in-global-markets.pdf

Lee, S.-H., & Kim, B.-G. (2011). IFC Extension for Road Structures and Digital Modeling. *Procedia Engineering*, *14*, 1037–1042. https://doi.org/10.1016/j.proeng.2011.07.130

Limberger, F. A., & Oliveira, M. M. (2015). Real-time detection of planar regions in unorganized point clouds. *Pattern Recognition*, *48*(6), 2043–2053. https://doi.org/10.1016/j.patcog.2014.12.020

Liu, X., Eybpoosh, M., & Akinci, B. (2012). Developing As-Built Building Information Model Using Construction Process History Captured by a Laser Scanner and a Camera. In *Construction Research Congress 2012* (pp. 1232–1241). Reston, VA: American Society of Civil Engineers. https://doi.org/10.1061/9780784412329.124

Liu, Y.-S., & Ramani, K. (2009). Robust principal axes determination for point-based shapes using least median of squares. *Computer-Aided Design*, *41*(4), 293–305. https://doi.org/10.1016/j.cad.2008.10.012

Luximon, Y., Ball, R., & Justice, L. (2012). The 3D Chinese head and face modeling. *Computer-Aided Design*, *44*(1), 40–47. https://doi.org/10.1016/j.cad.2011.01.011

Ma, L., Sacks, R., Kattel, U., & Bloch, T. (2018). 3D Object Classification Using Geometric Features and Pairwise Relationships. *Computer-Aided Civil and Infrastructure Engineering*, *33*(2), 152–164. https://doi.org/10.1111/mice.12336

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

Marton, Z. C., Rusu, R. B., & Beetz, M. (2009). On fast surface reconstruction methods for large and noisy point clouds. In *2009 IEEE International Conference on Robotics and Automation* (Vol. 269, pp. 3218–3223). IEEE. https://doi.org/10.1109/ROBOT.2009.5152628

Maturana, D., & Scherer, S. (2015). VoxNet: A 3D Convolutional Neural Network for real-time object recognition. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 922–928). IEEE. https://doi.org/10.1109/IROS.2015.7353481

McNell, D., Allison, H., Black, W., Cukrow, M., Harrison, K., Sherred, C., … Wilts, D. (2011). Building Information Modeling (BIM) Guide. *InfoComm International*. Retrieved from https://www.yumpu.com/en/document/view/8168798/building-information-modeling-bim-infocomm-international

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

Moreira, A., & Santos, M. Y. (2006). Concave Hull: A k-Nearest Neighbours Approach for The Computation of The Region Occupied By A Set of Points. *Proceedings of the 2nd International Conference on Computer Graphics Theory and Applications (GRAPP 2007), Barcelona, Spain*. Retrieved from http://hdl.handle.net/1822/6429

Mukhopadhyay, P., & Chaudhuri, B. B. (2015). A survey of Hough Transform. *Pattern Recognition*. https://doi.org/10.1016/j.patcog.2014.08.027

NAO. (2018). A Short Guide to the Department for Transport. *National Audit Office*. Retrieved from https://www.nao.org.uk/wp-content/uploads/2018/02/A-Short-Guide-to-the-Department-for-Transport-2017.pdf

Network Rail. (2015). Network Rail Bridge List. Available at https://www.whatdotheyknow.com/request/list_of_bridges_on_network_rail, accessed 20 September, 2018.

Nüchter, A., & Hertzberg, J. (2008). Towards semantic maps for mobile robots. *Robotics and Autonomous Systems*, *56*(11), 915–926. https://doi.org/10.1016/j.robot.2008.08.001

Ochmann, S., Vock, R., Wessel, R., & Klein, R. (2016). Automatic reconstruction of parametric building models from indoor point clouds. *Computers & Graphics*, *54*, 94–103. https://doi.org/10.1016/j.cag.2015.07.008

Oesau, S., Lafarge, F., & Alliez, P. (2014). Indoor scene reconstruction using feature sensitive primitive extraction and graph-cut. *ISPRS Journal of Photogrammetry and Remote Sensing*, *90*, 68–82. https://doi.org/10.1016/j.isprsjprs.2014.02.004

Okorn, B., Xiong, X., Akinci, B., & Huber, D. (2010). Toward Automated Modeling of Floor Plans. *Proceedings of the Symposium on 3D Data Processing, Visualization and Transmission*, *2*. Retrieved from https://pdfs.semanticscholar.org/3aac/ae3356e52a12c6e74926a289cc4495bc9bdf.pdf?_ga=2.145761012.1346194459.1537454782-2044273777.1532971434

Panushev, I., Eastman, C., Sacks, R., Venugopal, M., & Aram, V. (2010). Development of the national BIM standard (NBIMS) for precast/prestressed concrete. In *Proceedings of the 27th International Conference on Information Technology in Construction CIB W78*. Retrieved from http://itc.scix.net/data/works/att/w78-2010-18.pdf

Park, H. S., Lee, H. M., Adeli, H., & Lee, I. (2007). A New Approach for Health Monitoring of Structures: Terrestrial Laser Scanning. *Computer-Aided Civil* *and Infrastructure Engineering*, *22*(1), 19–30. https://doi.org/10.1111/j.1467-8667.2006.00466.x

Park, S. W., Park, H. S., Kim, J. H., & Adeli, H. (2015). 3D displacement measurement model for health monitoring of structures using a motion capture system. *Measurement*, *59*, 352–362. https://doi.org/10.1016/j.measurement.2014.09.063

Parrott, A., & Lane, W. (2017). Industry 4.0 and the digital twin. *Deloitte University Press*. Retrieved from https://www2.deloitte.com/insights/us/en/focus/industry-4-0/digital-twin-technology-smart-factory.html

Patil, A. K., Holi, P., Lee, S. K., & Chai, Y. H. (2017). An adaptive approach for the reconstruction and modeling of as-built 3D pipelines from point clouds. *Automation in Construction*, *75*, 65–78. https://doi.org/10.1016/j.autcon.2016.12.002

Pătrăucean, V., Armeni, I., Nahangi, M., Yeung, J., Brilakis, I., & Haas, C. (2015). State of research in automatic as-built modelling. *Advanced Engineering Informatics*, *29*(2), 162–171. https://doi.org/10.1016/j.aei.2015.01.001

PCL. (2018). Point Cloud Library. Available at http://pointclouds.org/, accessed 20 September, 2018.

Perez-Gallardo, Y., Cuadrado, J. L. L., Crespo, Á. G., & de Jesús, C. G. (2017). GEODIM: A Semantic Model-Based System for 3D Recognition of Industrial Scenes. In *Intelligent Systems Reference Library* (pp. 137–159). https://doi.org/10.1007/978-3-319-51905-0_7

Pfeifer, N., & Briese, C. (2007). Laser scanning – principles and applications. In *Non-Destructive Techniques for the Evaluation of Structures and Infrastructure* (pp. 7–35). https://doi.org/10.3997/2214-4609.201403279

Pinheiro, P. O., Lin, T.-Y., Collobert, R., & Dollàr, P. (2016). Learning to Refine Object Segments. *ArXiv Preprint*. Retrieved from http://arxiv.org/abs/1603.08695

Pu, S., & Vosselman, G. (2009). Knowledge based reconstruction of building models from terrestrial laser scanning data. *ISPRS Journal of Photogrammetry and Remote Sensing*, *64*(6), 575–584. https://doi.org/10.1016/j.isprsjprs.2009.04.001

Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2016). PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *CVPR 2017*. Retrieved from http://arxiv.org/abs/1612.00593

Qi, C. R., Su, H., Niessner, M., Dai, A., Yan, M., & Guibas, L. J. (2016). Volumetric and Multi-View CNNs for Object Classification on 3D Data. *2016,Computer Vision and Pattern Recognition (CVPR), IEEE*. Retrieved from http://arxiv.org/abs/1604.03265

Rabbani, T. (2006). Automatic Reconstruction of Industrial Installations Using Point Clouds and Images. *Publications on Geodesy*, *62*(May), 7401–7410. Retrieved from uuid:83466092-f531-41e2-8ef3-5985b608aeac

Rabbani, T., van den Heuvel, F. a, & Vosselman, G. (2006). Segmentation of point clouds using smoothness constraint. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences - Commission V Symposium "Image Engineering and Vision Metrology,"* *36*(5), 248–253. Retrieved from http://www.isprs.org/proceedings/XXXVI/part5/paper/RABB_639.pdf

Randall, T. (2013). Client Guide to 3D Scanning and Data Capture. *BIM Task Group*. Retrieved from http://bim-level2.org/globalassets/pdfs/clients-guide-to-3d-scanning-and-data-capture.pdf

Rashidi, M., Samali, B., & Sharafi, P. (2016). A new model for bridge management: Part A: condition assessment and priority ranking of bridges. *Australian Journal of Civil Engineering*, *14*(1), 35–45. https://doi.org/10.1080/14488353.2015.1092641

Rasmussen, C. E., & Williams, C. K. I. (2004). *Gaussian processes for machine learning. International journal of neural systems*. https://doi.org/10.1142/S0129065704001899

RDF Ltd. (2017). IFC Engine DLL. Available at http://rdf.bg/ifc-engine-dll.html, accessed 20 September, 2018.

Ricci, A. (1973). A constructive geometry for computer graphics. *The Computer Journal*, *16*(2), 157–160. https://doi.org/10.1093/comjnl/16.2.157

Riveiro, B., DeJong, M. J., & Conde, B. (2016). Automated processing of large point clouds for structural health monitoring of masonry arch bridges. *Automation in Construction*, *72*(3), 258–268. https://doi.org/10.1016/j.autcon.2016.02.009

Rusu, R. B., Blodow, N., Marton, Z. C., & Beetz, M. (2009). Close-range Scene Segmentation and Reconstruction of 3D Point Cloud Maps for Mobile Manipulation in Human Environments. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '09)*, 1–6.

Rusu, R. B., Marton, Z. C., Blodow, N., Dolha, M., & Beetz, M. (2008). Towards 3D Point cloud based object maps for household environments. *Robotics and Autonomous Systems*, *56*(11), 927–941. https://doi.org/10.1016/j.robot.2008.08.005

Rvachev, V. L. (1963). An analytical description of certain geometric objects. *Reports of Ukrainian Academy of Sciences*, *153*(4), 765–767.

Rvachev, V. L. (1974). Methods of Logic Algebra in Mathematical Physics. *Naukova Dumka*, *Kiev*.

Rvachev, V. L. (1982). Theory of R-functions and some applications. *Naukova Dumka*, *Kiev*.

Sacks, R., Kaner, I., Eastman, C. M., & Jeong, Y.-S. (2010). The Rosewood experiment — Building information modeling and interoperability for architectural precast facades. *Automation in Construction*, *19*(4), 419–432. https://doi.org/10.1016/j.autcon.2009.11.012

Sacks, R., Kedar, A., Borrmann, A., Ma, L., Brilakis, I., Hüthwohl, P., … Muhic, S. (2018). SeeBridge as next generation bridge inspection: Overview, Information Delivery Manual and Model View Definition. *Automation in Construction*, *90*, 134–145. https://doi.org/10.1016/j.autcon.2018.02.033

Sampath, A., & Jie Shan. (2010). Segmentation and Reconstruction of Polyhedral Building Roofs From Aerial Lidar Point Clouds. *IEEE Transactions on Geoscience and Remote Sensing*, *48*(3), 1554–1567. https://doi.org/10.1109/TGRS.2009.2030180

Schnabel, R., Wahl, R., & Klein, R. (2007). Efficient RANSAC for Point-Cloud Shape Detection. *Computer Graphics Forum*, *26*(2), 214–226. https://doi.org/10.1111/j.1467-8659.2007.01016.x

Schulte, B. A. (2003). Scientific Writing &amp; the Scientific Method: Parallel "Hourglass" Structure in Form &amp; Content. *The American Biology Teacher*, *65*(8), 591–594. https://doi.org/10.2307/4451568

SeeBridge. (2014). Infravation Project: Automated Compilation of Semantically Rich BIM Models of Bridges. Available at https://seebridge.net.technion.ac.il/, accessed 20 September, 2018.

Son, H., Kim, C., & Kim, C. (2013). KNOWLEDGE-BASED APPROACH FOR 3D RECONSTRUCTION OF AS-BUILT INDUSTRIAL PLANT MODELS FROM LASER-SCAN DATA. *ISARC*. https://doi.org/10.22260/ISARC2013/0096

Son, H., Kim, C., & Kim, C. (2015). 3D reconstruction of as-built industrial instrumentation models from laser-scan data and a 3D CAD database based on prior knowledge. *Automation in Construction*, *49*, 193–200. https://doi.org/10.1016/j.autcon.2014.08.007

Song, X., & Jüttler, B. (2009). Modeling and 3D object reconstruction by implicitly defined surfaces with sharp features. *Computers & Graphics*, *33*(3), 321–330. https://doi.org/10.1016/j.cag.2009.03.021

Su, Y., Bethel, J., & Hu, S. (2016). Octree-based segmentation for terrestrial LiDAR point cloud data in industrial applications. *ISPRS Journal of Photogrammetry and Remote Sensing*, *113*, 59–74. https://doi.org/10.1016/j.isprsjprs.2016.01.001

Tang, P. (2009). Extraction of Surveying Goals from Point Clouds Obtained from Laser Scanners to Support Bridge Inspection. *PhD Thesis*, (August). https://doi.org/1877822251

Tang, P., Akinci, B., & Garrett, J. H. (2007). Laser Scanning for Bridge Inspection and Management. *IABSE Symposium Report*, *93*(18), 17–24. https://doi.org/10.2749/222137807796120283

Tang, P., Huber, D., Akinci, B., Lipman, R., & Lytle, A. (2010). Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques. *Automation in Construction*, *19*(7), 829–843. https://doi.org/10.1016/j.autcon.2010.06.007

Tarsha-Kurdi, F., Landes, T., & Grussenmeyer, P. (2007). Hough-Transform and Extended Ransac Algorithms for Automatic Detection of 3D Building Roof Planes From Lidar Data. *ISPRS Workshop on Laser Scanning 2007 and SilviLaser 2007*, *XXXVI*(1), 407–412. Retrieved from http://www.isprs.org/proceedings/XXXVI/3-W52/final_papers/Tarsha-Kurdi_2007.pdf

Tatarchenko, M., Dosovitskiy, A., & Brox, T. (2017). Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. arXiv preprint arXiv:1703.09438.

Thompson, W. B., Owen, J. C., de St. Germain, H. J., Stark, S. R., & Henderson, T. C. (1999). Feature-based reverse engineering of mechanical parts. *IEEE Transactions on Robotics and Automation*, *15*(1), 57–66. https://doi.org/10.1109/70.744602

Tombari, F., & Di Stefano, L. (2010). Object Recognition in 3D Scenes with Occlusions and Clutter by Hough Voting. In *2010 Fourth Pacific-Rim Symposium on Image and Video Technology* (pp. 349–355). IEEE. https://doi.org/10.1109/PSIVT.2010.65

Torr, P. H. S., & Zisserman, A. (2000). MLESAC: A New Robust Estimator with Application to Estimating Image Geometry. *Computer Vision and Image Understanding*, *78*(1), 138–156. https://doi.org/10.1006/cviu.1999.0832

Trimble. (2017). Creating a 3D CAD Model From a laser Scanned Point Cloud (Overview). Available at https://constructible.trimble.com/construction-industry/how-to-create-a-3d-cad-model-from-a-laser-scanned-point-cloud-overview, accessed 20 September, 2018.

Trimble. (2018). RealWorks. Available at https://geospatial.trimble.com/products-and-solutions/trimble-realworks, 20 September, 2018. *Geospatial*.

Truong-Hong, L., & Laefer, D. F. (2015). Quantitative evaluation strategies for urban 3D model generation from remote sensing data. *Computers & Graphics*, *49*, 82–91. https://doi.org/10.1016/j.cag.2015.03.001

Truong-Hong, L., Laefer, D. F., Hinks, T., & Carr, H. (2012). Flying Voxel Method with Delaunay Triangulation Criterion for Façade/Feature Detection for Computation. *Journal of Computing in Civil Engineering*, *26*(6), 691–707. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000188

Truong-Hong, L., Laefer, D. F., Hinks, T., & Carr, H. (2013). Combining an Angle Criterion with Voxelization and the Flying Voxel Method in Reconstructing Building Models from LiDAR Data. *Computer-Aided Civil and Infrastructure Engineering*, *28*(2), 112–129. https://doi.org/10.1111/j.1467-8667.2012.00761.x

U.S. DoT. (2015). 2015 Status of the Nation's Highways, Bridges, and Transit: Conditions Performance. Technical report. Available at https://www.fhwa.dot.gov/policy/2015cpr/es.cfm#5h, 20 September, 2018.

Uijlings, J. R. R., van de Sande, K. E. A., Gevers, T., & Smeulders, A. W. M. (2013). Selective Search for Object Recognition. *International Journal of Computer Vision*, *104*(2), 154–171. https://doi.org/10.1007/s11263-013-0620-5

UK Roads Liaison Group. (2013). Management of Highway Structures - Code of Practice. Available at http://www.ukroadsliaisongroup.org/en/utilities/document-summary.cfm?docid=28EAC85C-CD3D-48FC-887E22845F326CBD, accessed 20 September, 2018.

Valero, E., Adán, A., & Bosché, F. (2016). Semantic 3D Reconstruction of Furnished Interiors Using Laser Scanning and RFID Technology. *Journal of Computing in Civil Engineering*, *30*(4), 04015053. https://doi.org/10.1061/(ASCE)CP.1943-5487.0000525

Valero, E., Adán, A., & Cerrada, C. (2012). Automatic Method for Building Indoor Boundary Models from Dense Point Clouds Collected by Laser Scanners. *Sensors*, *12*(12), 16099–16115. https://doi.org/10.3390/s121216099

Vassou, V. (2010). Structures Condition Suevey of Borough Principal Road Network. Technical report. *London Bridges Engineering Group, Bridge Condition Indicators Project.* Retrieved from http://www.bridgeforum.org/bof/meetings/bof33/BCI Study_Report_Final.pdf

Venugopal, M., Eastman, C. M., Sacks, R., & Teizer, J. (2012). Semantics of model views for information exchanges using the industry foundation class schema. *Advanced Engineering Informatics*, *26*(2), 411–428. https://doi.org/10.1016/j.aei.2012.01.005

Vince, J. (2013). *Calculus for Computer Graphics*. London: Springer London. https://doi.org/10.1007/978-1-4471-5466-2

Vo, A.-V., Truong-Hong, L., Laefer, D. F., & Bertolotto, M. (2015). Octree-based region growing for point cloud segmentation. *ISPRS Journal of Photogrammetry and Remote Sensing*, *104*, 88–100. https://doi.org/10.1016/j.isprsjprs.2015.01.011

Vosselman, G. (2009). Advanced Point Cloud Processing. *In Photogrammetric Week'09*, (July), 137–146.

Vosselman, G., Gorte, B., Sithole, G., & Rabbani, T. (2004). Recognising Structure in Laser Scanner Point Clouds. *Information Sciences*. https://doi.org/10.1002/bip.360320508

Wai-Fah, C., & Lian, D. (2014). *Bridge Engineering Handbook, Construction and Maintenance. CRC Press.*

Walsh, S. B., Borello, D. J., Guldur, B., & Hajjar, J. F. (2013). Data Processing of Point Clouds for Object Detection for Structural Engineering Applications. *Computer-Aided Civil and Infrastructure Engineering*, *28*(7), 495–508. https://doi.org/10.1111/mice.12016

Wang, C., Cho, Y. K., & Kim, C. (2015). Automatic BIM component extraction from point clouds of existing buildings for sustainability applications. *Automation in Construction*, *56*, 1–13. https://doi.org/10.1016/j.autcon.2015.04.001

Watt, A. (2000). *3D Computer Graphics. Addison-Wesley.*

Webb, G. T., Vardanega, P. J., Fidler, P. R. A., & Middleton, C. R. (2014). Analysis of Structural Health Monitoring Data from Hammersmith Flyover. *Journal of Bridge Engineering*, *19*(6), 05014003. https://doi.org/10.1061/(ASCE)BE.1943-5592.0000587

Weisstein, E. W. (2018a). Ellipsoid. Available at http://mathworld.wolfram.com/Ellipsoid.html, accessed 20 September, 2018. *MathWorld - A Wolfram Web Resource.*

Weisstein, E. W. (2018b). Elliptic Paraboloid. Available at http://mathworld.wolfram.com/EllipticParaboloid.html, accessed 20 September, 2018. *MathWorld - A Wolfram Web Resource.*

Weisstein, E. W. (2018c). Hyperbolic Paraboloid. Available at http://mathworld.wolfram.com/HyperbolicParaboloid.html, accessed 20 September, 2018. *MathWorld - A Wolfram Web Resource.*

Weisstein, E. W. (2018d). Torus. Available at http://mathworld.wolfram.com/Torus.html, accessed 20 September, 2018. *MathWorld - A Wolfram Web Resource.*

West, J., & Brereton, D. (2013). *A framework for best practice in financial risk assessment, governance and disclosure.* (G. C. National Climate Change Adaptation Research Facility, Ed.). Australia.

West, T. D., & Blackburn, M. (2017). Is Digital Thread/Digital Twin Affordable? A Systemic Assessment of the Cost of DoD's Latest Manhattan Project. *Procedia Computer Science*, *114*, 47–56. https://doi.org/10.1016/j.procs.2017.09.003

Wikipedia. (2018). Constructive Solid Geometry. Available at https://en.wikipedia.org/wiki/Constructive_solid_geometry, accessed 20 September, 2018.

World Economic Forum. (2016). Shaping the Future of Construction. *A Breakthrough in Mindset and Technology*. Retrieved from http://www3.weforum.org/docs/WEF_Shaping_the_Future_of_Construction_full_report__.pdf

WSDoT. (2009). Precast Prestressed Wide Flange Girders. Available at http://www.wsdot.wa.gov/eesc/bridge/designmemos/14-2009.htm, accessed 20 September, 2018.

Xiao, J., & Furukawa, Y. (2014). Reconstructing the World's Museums. *International Journal of Computer Vision*, *110*(3), 243–258. https://doi.org/10.1007/s11263-014-0711-y

Xiao, J., Zhang, J., Adler, B., Zhang, H., & Zhang, J. (2013). Three-dimensional point cloud plane segmentation in both structured and unstructured environments. *Robotics and Autonomous Systems*, *61*(12), 1641–1652. https://doi.org/10.1016/j.robot.2013.07.001

Xiong, X., Adan, A., Akinci, B., & Huber, D. (2013). Automatic creation of semantically rich 3D building models from laser scanner data. *Automation in Construction*, *31*, 325–337. https://doi.org/10.1016/j.autcon.2012.10.006

Xiong, X., & Huber, D. (2010). Using Context to Create Semantic 3D Models of Indoor Environments. In *Procedings of the British Machine Vision Conference 2010* (p. 45.1-45.11). British Machine Vision Association. https://doi.org/10.5244/C.24.45

Xu, Y., Tuttas, S., Hoegner, L., & Stilla, U. (2018). Voxel-based segmentation of 3D point clouds from construction sites using a probabilistic connectivity model. *Pattern Recognition Letters*, *102*, 67–74. https://doi.org/10.1016/j.patrec.2017.12.016

Yabuki, N., Lebegue, E., Gual, J., Shitani, T., & Zhantao, L. (2006). International Collaboration for Developing the Bridge Product Model "IFC-Bridge." In *Proceedings of the 2006 Joint International Conference on Computing and Decision Making in Civil and Building Engineering*.

Yan, J., Jiang, W., & Shan, J. (2012). QUALITY ANALYSIS ON RANSAC-BASED ROOF FACETS EXTRACTION FROM AIRBORNE LIDAR DATA. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XXXIX-B3*, 367–372. https://doi.org/10.5194/isprsarchives-XXXIX-B3-367-2012

Yen, K. S., Lasky, T. A., & Ravani, B. (2014). Cost-Benefit Analysis of Mobile Terrestrial Laser Scanning Applications for Highway Infrastructure. *Journal of Infrastructure Systems*, *20*(4), 04014022. https://doi.org/10.1061/(ASCE)IS.1943-555X.0000192

Yue, K., Huber, D., Akinci, B., & Krishnamurti, R. (2006). The ASDMCon Project: The Challenge of Detecting Defects on Construction Sites. In *Third International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT'06)* (pp. 1048–1055). IEEE. https://doi.org/10.1109/3DPVT.2006.134

Zhang, C., & Tang, P. (2015). Visual Complexity Analysis of Sparse Imageries for Automatic Laser Scan Planning in Dynamic Environments. In *Computing in Civil Engineering 2015* (pp. 271–279). Reston, VA: American Society of Civil Engineers. https://doi.org/10.1061/9780784479247.034

Zhang, G., Vela, P. A., & Brilakis, I. (2014). Automatic Generation of As-Built Geometric Civil Infrastructure Models from Point Cloud Data. In *Computing in Civil and Building Engineering (2014)* (pp. 406–413). Reston, VA: American Society of Civil Engineers. https://doi.org/10.1061/9780784413616.051

Zhang, G., Vela, P. A., Karasev, P., & Brilakis, I. (2015). A Sparsity-Inducing Optimization-Based Algorithm for Planar Patches Extraction from Noisy Point-Cloud Data. *Computer-Aided Civil and Infrastructure Engineering*, *30*(2), 85–102. https://doi.org/10.1111/mice.12063

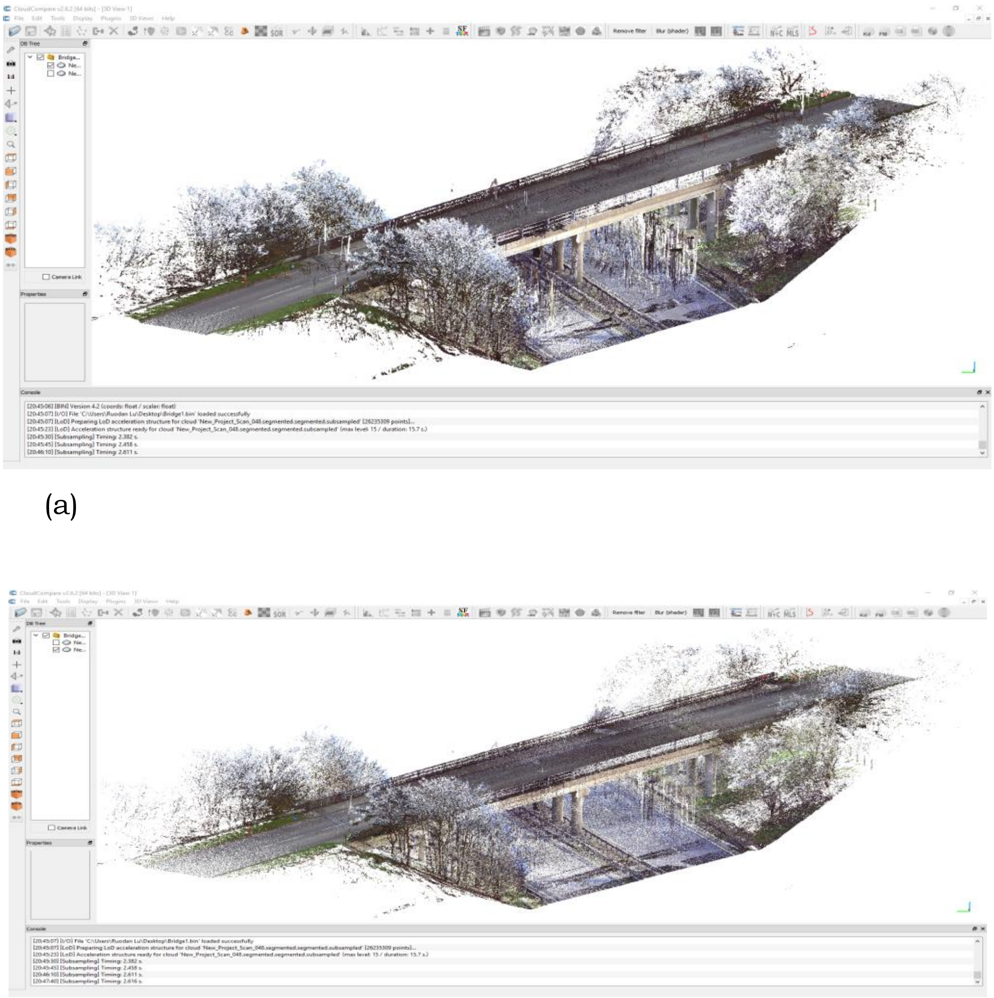
9. APPENDIX

Appendix A

8-step manual bridge DT generation from a registered PCD

Step 1. After registering the raw scans, the registered raw PCD of a RC bridge is imported into CloudCompare. The format of the raw PCD can be .bin, .pcd, .txt and so on, depending on different data outputs from the registration phase (Figure 9.1 (a)).

Step 2. The PCD is sub-sampled using the Cloud Sub Sampling functionality in CloudCompare (sampling method: random) and then the sub-sampled result is saved. The reason for down-sampling the original PCD is that current modelling software such as Revit is an in-memory system which slows down significantly or even collapses when working with large PCD. This requires sub-sampling the original PCD in advance (Figure 9.1 (b)).



```
[20:45:06] [BIN] Version 4.2 (coords: float / scalar: float)
[20:45:07] [I/O] File 'C:\Users\Ruodan Lu\Desktop\Bridge1.bin' loaded successfully
[20:45:07] [LoD] Preparing LoD acceleration structure for cloud 'New_Project_Scan_048.segmented.segmented.subsampled' [26235309 points]...
[20:45:23] [LoD] Acceleration structure ready for cloud 'New_Project_Scan_048.segmented.segmented.subsampled' (max level: 15 / duration: 15.7 s.)
[20:45:30] [Subsampling] Timing: 2.382 s.
[20:45:45] [Subsampling] Timing: 2.458 s.
[20:46:10] [Subsampling] Timing: 2.611 s.
```

(a)

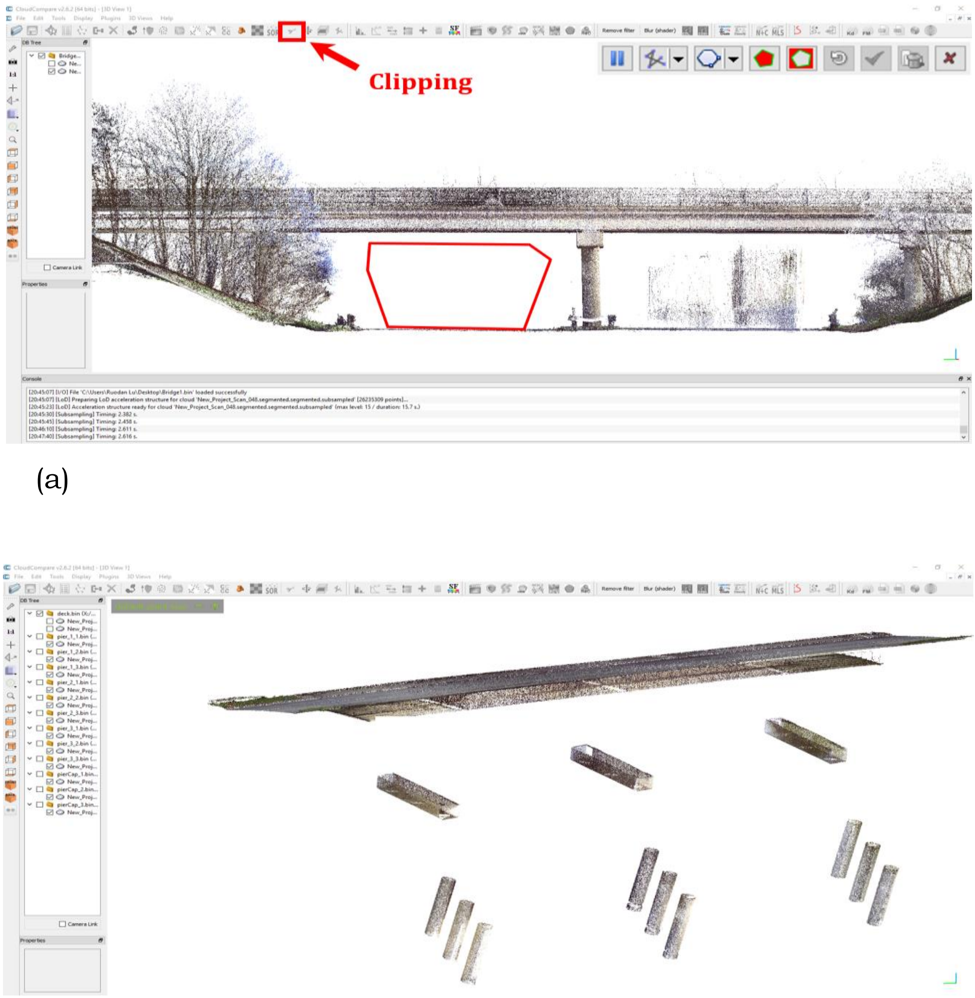
```
[20:45:07] [I/O] File 'C:\Users\Ruodan Lu\Desktop\Bridge1.bin' loaded successfully
[20:45:07] [LoD] Preparing LoD acceleration structure for cloud 'New_Project_Scan_048.segmented.segmented.subsampled' [26235309 points]...
[20:45:23] [LoD] Acceleration structure ready for cloud 'New_Project_Scan_048.segmented.segmented.subsampled' (max level: 15 / duration: 15.7 s.)
[20:45:30] [Subsampling] Timing: 2.382 s.
[20:45:45] [Subsampling] Timing: 2.458 s.
[20:46:10] [Subsampling] Timing: 2.611 s.
[20:47:40] [Subsampling] Timing: 2.616 s.
```

(b)

Figure 9.1 (a) Step 1: Insert a raw PCD into CloudCompare; (b) Step 2: Sub-sample the raw PCD

Step 3. The sub-sampled data is cropped, which aims to remove irrelevant points such as trees, vegetation, road, traffic, etc. To do so, a modeller repeatedly selects regions of interest to delete by creating polygons through CloudCompare's clipping functionality. The remaining points contain the components that need to be modelled in a given modelling project (Figure 9.2 (a)).

Step 4. The clipping functionality is again repeatedly used in order to segment the cropped point cloud into individual point-clusters, which correspond to the components making up a bridge. To this end, a modeller needs to repeatedly rotate the cropped PCD to find the best view, and segment along the best view. Each segmented sub-point-cluster is saved into an .e57 file (Figure 9.2 (b)).



```
[20:45:07] [I/O] File 'C:\Users\Ruodan Lu\Desktop\Bridge1.bin' loaded successfully
[20:45:07] [LoD] Preparing LoD acceleration structure for cloud 'New_Project_Scan_048.segmented.segmented.subsampled' [26235309 points]...
[20:45:23] [LoD] Acceleration structure ready for cloud 'New_Project_Scan_048.segmented.segmented.subsampled' (max level: 15 / duration: 15.7 s.)
[20:45:30] [Subsampling] Timing: 2.382 s.
[20:45:45] [Subsampling] Timing: 2.458 s.
[20:46:10] [Subsampling] Timing: 2.611 s.
[20:47:40] [Subsampling] Timing: 2.616 s.
```

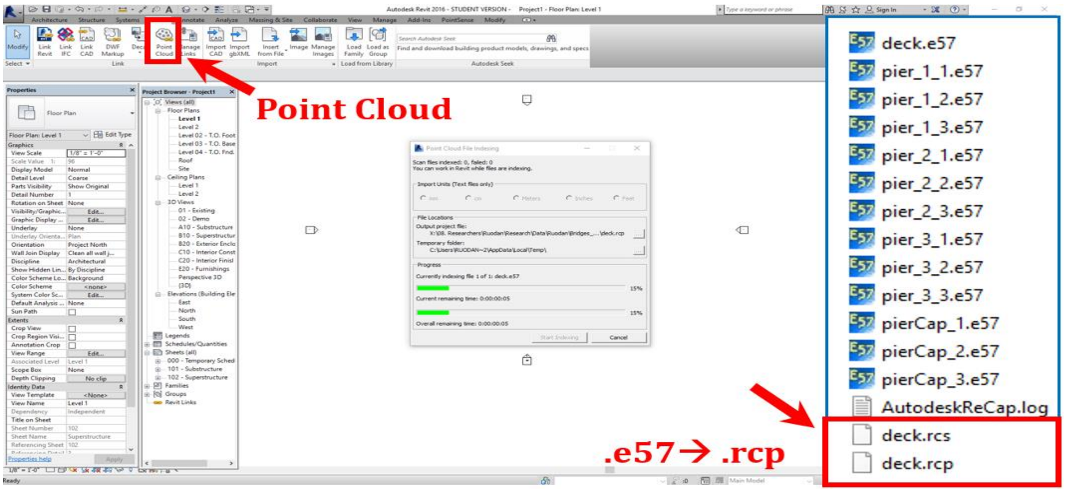
(a)

(b)

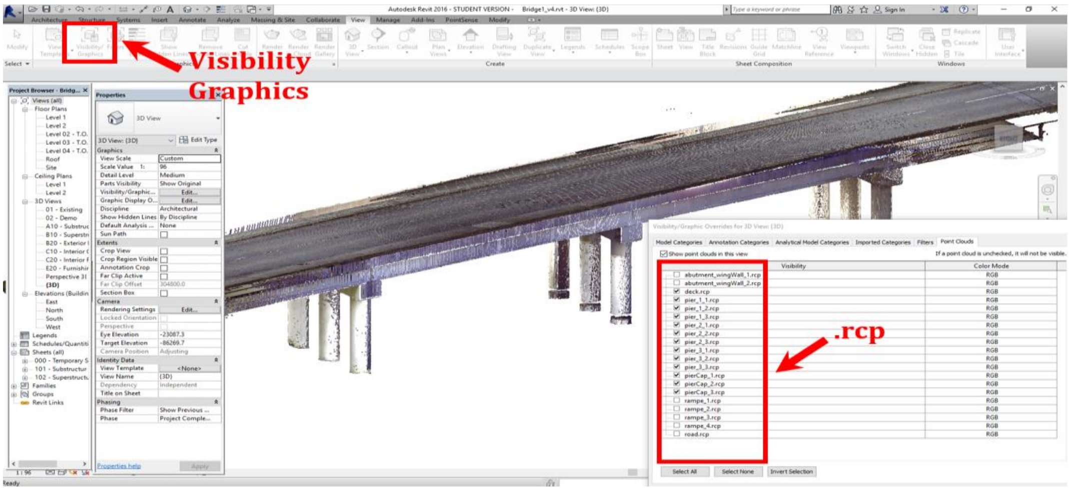
Figure 9.2 (a) Step 3: Remove irrelevant points; (b) Step 4: Segmentation into sub-parts

Step 5. A Revit project is opened and a point cluster in .e57 file is imported by clicking the Insert Point Cloud tab. The .e57 file needs to be first converted into an .rcp file by an indexing process, and then positioned by shared coordinates. This procedure is repeated until all .e57 files are indexed and a set of .rcs and .rcp files are created (Figure 9.3 (a)).

Step 6. Once all point cloud projects (*.rcp) have been created, the visibility of each point-cluster can be controlled by the Visibility Graphics tab (Figure 9.3 (b)).



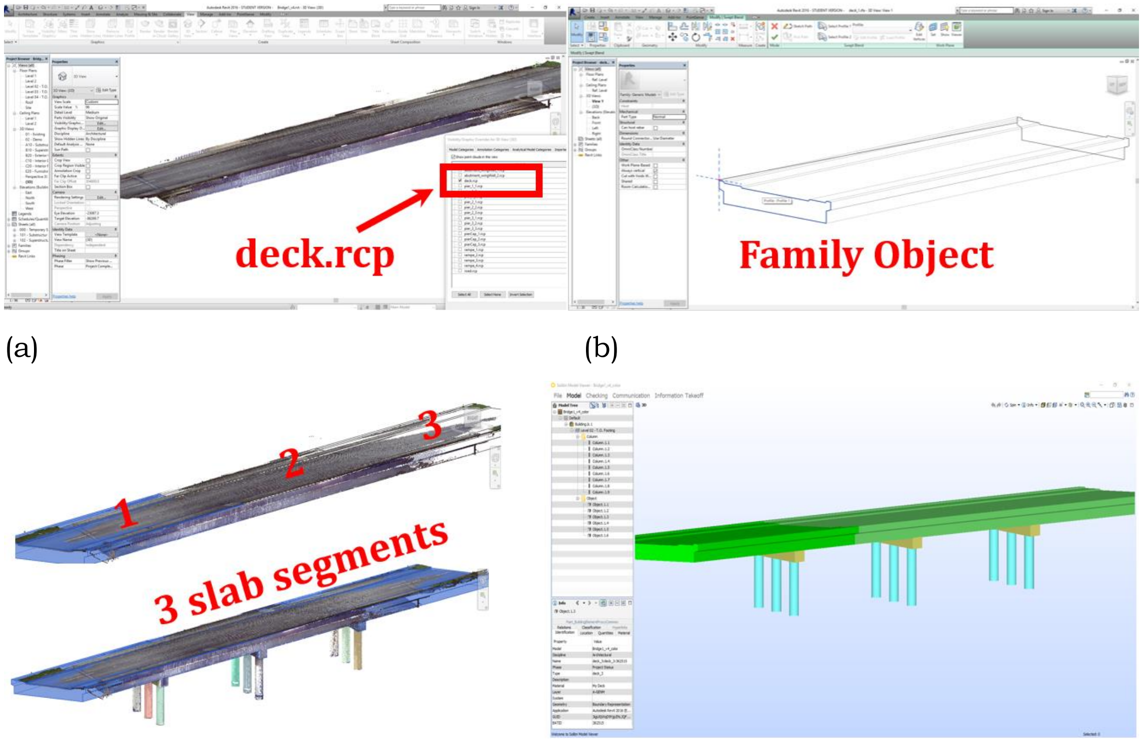
(a)



(b)

Figure 9.3 (a) Step 5: Import point clusters into Revit; (b) Step 6: One cluster shown, others hidden

Step 7. The point-clusters are modelled one by one. To do so, only the point cluster being modelled is displayed on the screen, with the others hidden (Figure 9.4 (a)). Based on the geometric nature of the current point cluster, a modeller uses his or her engineering knowledge and modelling experience to decide the object's type and to fit the point cluster with (1) a generic shape from the built-in shape library, or (2) a manually created customized shape using Revit Family editor. For complicated point cluster like the deck slab, a modeller needs to fit it using multiple customized shapes by manually generating Family objects (Figure 9.4 (b)) so that its overall topology can be approximated. For example, a deck slab is modelled by several individual free-form-shaped slab segments (Figure 9.4 (c)). Up until this step, the EURs 1 and 2, i.e. description of the exact shape and geometric attributes of a component, are met.



(a) (b)

(c) (d)

Figure 9.4 (a) Step 7: Deck slab cluster shown, others hidden; (b) Step 7: Edit slab segment as Family object; (c) Step 7: Deck slab consists of 3 slab segments; (d) Step 8: Export Revit project into IFC and visualize in Solibri

Step 8. Finally, the Revit modelling project can be exported into an .ifc file after a manual semantic enrichment process. To do so, a modeller can label each component with its real-world taxonomy (EUR 3) and then choose a specific IFC setup (IFC2x3 or IFC4) to create an .ifc file. The final IFC file can be visualized in Solibri Model Viewer or any other IFC viewer

(Figure 9.4 (d)). Up until the end, only EURs 1, 2, 3 and 6 are satisfied whereas EURs 4 and 5 are not.

Appendix B

SeeBridge Project Overview

Figure 9.5 highlights the scope this thesis addresses in the context of the SeeBridge project.

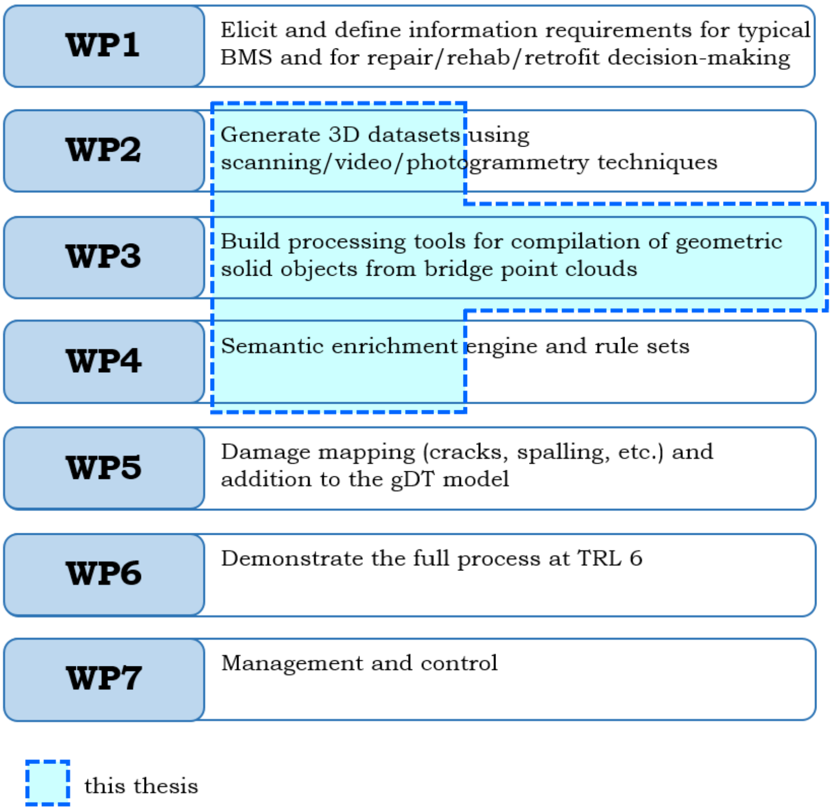

WP1
Elicit and define information requirements for typical BMS and for repair/rehab/retrofit decision-making
WP2
Generate 3D datasets using scanning/video/photogrammetry techniques
WP3
Build processing tools for compilation of geometric solid objects from bridge point clouds
WP4
Semantic enrichment engine and rule sets
WP5
Damage mapping (cracks, spalling, etc.) and addition to the gDT model
WP6
Demonstrate the full process at TRL 6
WP7
Management and control
this thesis

Figure 9.5 The scope of this thesis in the context of the SeeBridge project

WP2 - Spatial & visual raw data collection with existing rapid and non-contact survey technologies such as LS, video/photogrammetry, etc.

WP3 – Development of a bridge object detection and modelling software tool for automated compilation of solid 3D geometry from the PCD, i.e. for gDT generation.

WP4 – Development of a rule-processing expert system for semantic enrichment of the solid geometry model to generate a bridge DT. This has two aspects: (a) classification of bridge objects, and (b) deduction of supplementary information concerning material types, internal component geometry, and topological and aggregation relationships.

WP5 – Development of a damage measurement tool for damage identification, classification and spatial/visual properties, and measurement and integration of this information with the bridge gDT.

Appendix C

SeeBridge Prototype Element Identification Evaluation Categories

🟧 Normally this element always exist in this type of Bridge

🟦 Optional Element sometime exist in this type of Bridge

🟥 Other optional elements

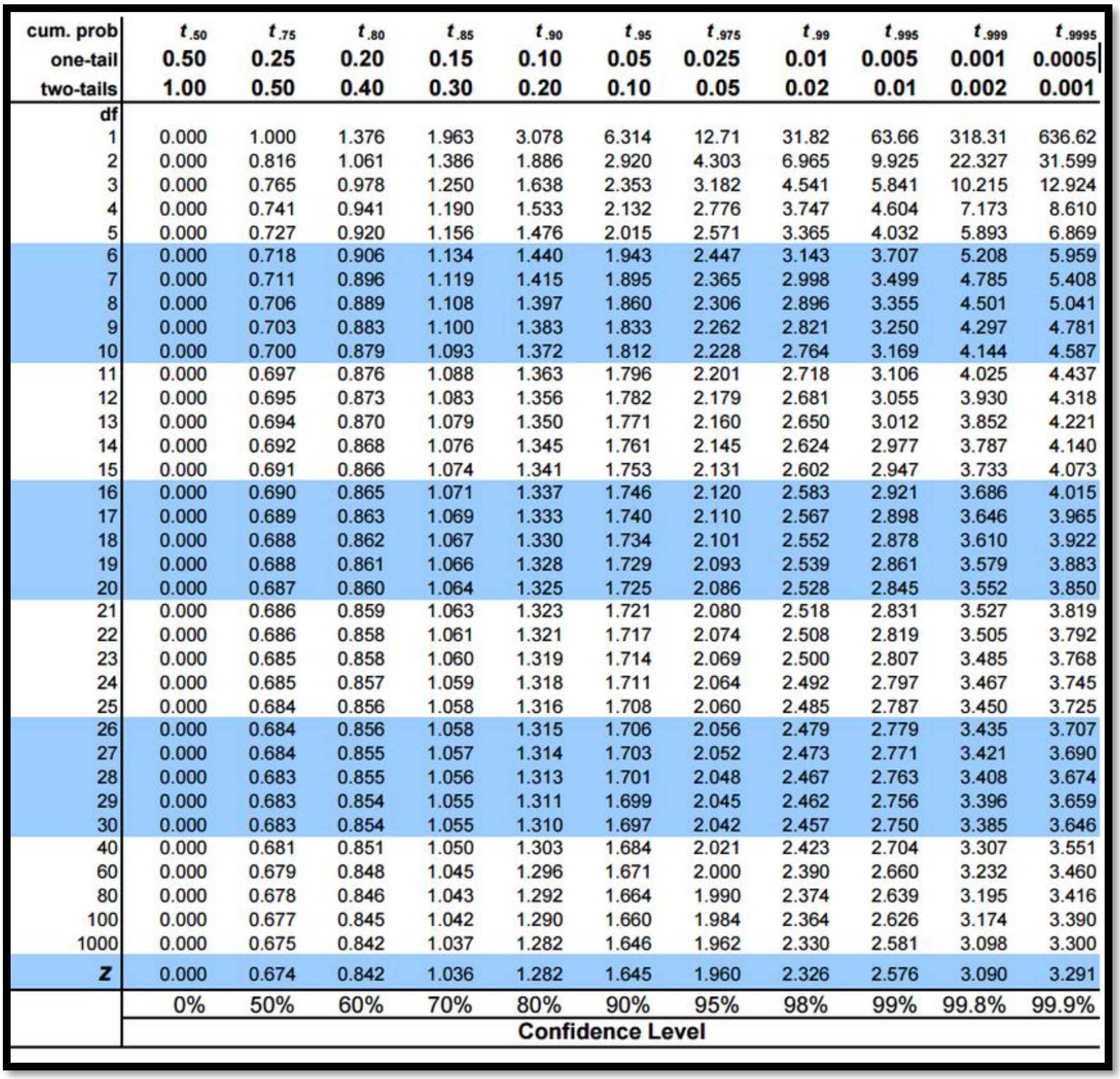
Otherwise, this elements do not exist in this type of Bridge

C1: Element evaluation category 1
C2: Element evaluation category 2
C3: Element evaluation category 3
C4: Element evaluation category 4
C5: Element evaluation category 5

Slab Bridge	**Steel Beam Girder**	**Concrete Beam Girder Bridge**		**Main Structure**		
Monolithic Slab Bridge	At/Below deck surface	Box Girder (ext. & int.)	At/Below deck surface	**Bridge Type**	**Element description**	**Element Group**
C1	C1		C1	111	Primary Girders	**Deck/ Superstructure**
				112	Slab	
		C1		131	Box	
	C1/2	C1/2	C1/2	201	Transverse Beam/Diaphragm	
	C1	C1	C1	301	Deck slab	
C1	C1	C1	C1	202	Half Joints	
	C1			203	Tie beam/rod/hangers/deck stiffeners	
C1	C1	C1	C1	204	Cantilever	
		C2		205	Blisters	
		C2		206	Saddles	
				207	Main Cables	
				208	Cable hangers	
				209	Cable anchors	
				210	Cable spreader	
C2	C2	C2	C2	401	Shear Keys	**Substructure**
C5	C5	C5	C5	402	Foundations/Pile caps/Piles	
C2	C2	C2	C2	403	Abutments/Arch springing/End walls	
				404	Head wall/Spandrel wall	
C1	C1	C1	C1	405	Pier/Column/Arch support/Pylon	
C1	C1	C1	C1	406	Cross-head/Capping beam	
C3	C3	C3	C3	407	Bearings	
C2	C2	C2	C2	408	Bearing Plinth/Pedestal/Shelf	
				409	Cables Anchors Blocks/chambers	
C3	C3	C3	C3	501	Superstructure Drainages	**Durability protection Elements**
C3/5	C3/5	C3/5	C3/5	502	Substructure Drainage/Drainage channel	
C5	C5	C5	C5	503	Waterproofing (all elements)	
C2	C2	C2	C2	504	Bridge deck Expansion Joints	
C3	C3	C3	C3	505	Finishes & protective coatings: Superstructure	
C3	C3	C3	C3	506	Finishes & protective coatings: Substructure	
C3	C3	C3	C3	507	Finishes & protective coatings: parapets/safety barriers	
C3	C3	C3	C3	601	Access/Walkways/Stairs	**Safety Elements**
C2	C2	C2	C2	602	Safety Barriers/handrails	
C3	C3	C3	C3	603	Carriageway surfacing	
C3	C3	C3	C3	604	Footway/verge/shoulders/footbridge surfacing	
C3	C3	C3	C3	701	Invert/River bed	**Other Elements**
C3	C3	C3	C3	702	Aprons/parapets/edge beams	
C3	C3	C3	C3	703	Fenders/cutwaters/collision protection	
C3	C3	C3	C3	704	River Training works	
C3	C3	C3	C3	705	Revetment/batter paving	
C2	C2	C2	C2	706	Wing walls	
C3	C3	C3	C3	707	Retaining Walls	
C3	C3	C3	C3	708	Embankments	
C3	C3	C3	C3	709	Expansion Joints	
C5	C5	C5	C5	710	Approach slabs	
C3	C3	C3	C3	711	Curbs	
				712	Machinery	
C4	C4	C4	C4	901	Approach ramps retaining walls	**Miscellaneous Elements**
C4	C4	C4	C4	902	Signs	
C4	C4	C4	C4	903	Lighting	

Appendix D

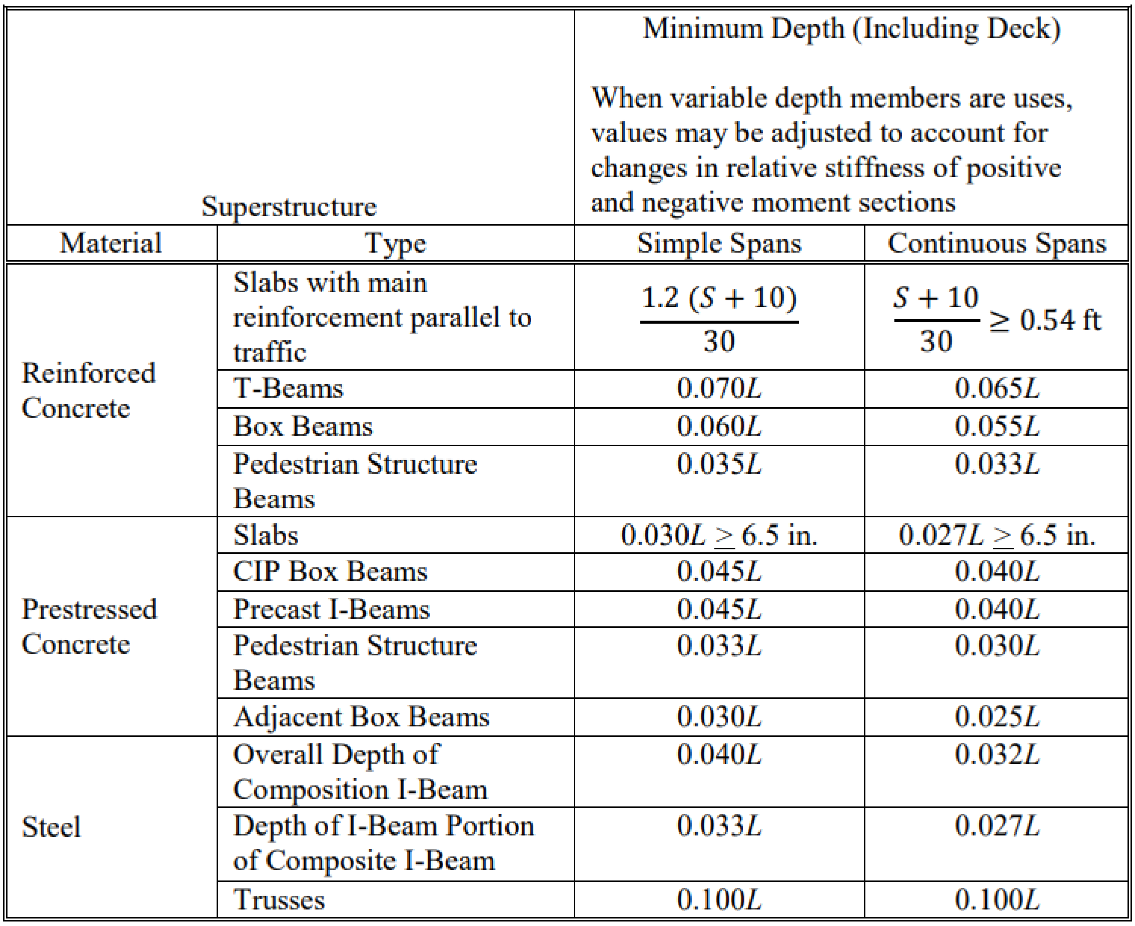
T-table



cum. prob	$t_{.50}$	$t_{.75}$	$t_{.80}$	$t_{.85}$	$t_{.90}$	$t_{.95}$	$t_{.975}$	$t_{.99}$	$t_{.995}$	$t_{.999}$	$t_{.9995}$
one-tail	**0.50**	**0.25**	**0.20**	**0.15**	**0.10**	**0.05**	**0.025**	**0.01**	**0.005**	**0.001**	**0.0005**
two-tails	**1.00**	**0.50**	**0.40**	**0.30**	**0.20**	**0.10**	**0.05**	**0.02**	**0.01**	**0.002**	**0.001**
df											
1	0.000	1.000	1.376	1.963	3.078	6.314	12.71	31.82	63.66	318.31	636.62
2	0.000	0.816	1.061	1.386	1.886	2.920	4.303	6.965	9.925	22.327	31.599
3	0.000	0.765	0.978	1.250	1.638	2.353	3.182	4.541	5.841	10.215	12.924
4	0.000	0.741	0.941	1.190	1.533	2.132	2.776	3.747	4.604	7.173	8.610
5	0.000	0.727	0.920	1.156	1.476	2.015	2.571	3.365	4.032	5.893	6.869
6	0.000	0.718	0.906	1.134	1.440	1.943	2.447	3.143	3.707	5.208	5.959
7	0.000	0.711	0.896	1.119	1.415	1.895	2.365	2.998	3.499	4.785	5.408
8	0.000	0.706	0.889	1.108	1.397	1.860	2.306	2.896	3.355	4.501	5.041
9	0.000	0.703	0.883	1.100	1.383	1.833	2.262	2.821	3.250	4.297	4.781
10	0.000	0.700	0.879	1.093	1.372	1.812	2.228	2.764	3.169	4.144	4.587
11	0.000	0.697	0.876	1.088	1.363	1.796	2.201	2.718	3.106	4.025	4.437
12	0.000	0.695	0.873	1.083	1.356	1.782	2.179	2.681	3.055	3.930	4.318
13	0.000	0.694	0.870	1.079	1.350	1.771	2.160	2.650	3.012	3.852	4.221
14	0.000	0.692	0.868	1.076	1.345	1.761	2.145	2.624	2.977	3.787	4.140
15	0.000	0.691	0.866	1.074	1.341	1.753	2.131	2.602	2.947	3.733	4.073
16	0.000	0.690	0.865	1.071	1.337	1.746	2.120	2.583	2.921	3.686	4.015
17	0.000	0.689	0.863	1.069	1.333	1.740	2.110	2.567	2.898	3.646	3.965
18	0.000	0.688	0.862	1.067	1.330	1.734	2.101	2.552	2.878	3.610	3.922
19	0.000	0.688	0.861	1.066	1.328	1.729	2.093	2.539	2.861	3.579	3.883
20	0.000	0.687	0.860	1.064	1.325	1.725	2.086	2.528	2.845	3.552	3.850
21	0.000	0.686	0.859	1.063	1.323	1.721	2.080	2.518	2.831	3.527	3.819
22	0.000	0.686	0.858	1.061	1.321	1.717	2.074	2.508	2.819	3.505	3.792
23	0.000	0.685	0.858	1.060	1.319	1.714	2.069	2.500	2.807	3.485	3.768
24	0.000	0.685	0.857	1.059	1.318	1.711	2.064	2.492	2.797	3.467	3.745
25	0.000	0.684	0.856	1.058	1.316	1.708	2.060	2.485	2.787	3.450	3.725
26	0.000	0.684	0.856	1.058	1.315	1.706	2.056	2.479	2.779	3.435	3.707
27	0.000	0.684	0.855	1.057	1.314	1.703	2.052	2.473	2.771	3.421	3.690
28	0.000	0.683	0.855	1.056	1.313	1.701	2.048	2.467	2.763	3.408	3.674
29	0.000	0.683	0.854	1.055	1.311	1.699	2.045	2.462	2.756	3.396	3.659
30	0.000	0.683	0.854	1.055	1.310	1.697	2.042	2.457	2.750	3.385	3.646
40	0.000	0.681	0.851	1.050	1.303	1.684	2.021	2.423	2.704	3.307	3.551
60	0.000	0.679	0.848	1.045	1.296	1.671	2.000	2.390	2.660	3.232	3.460
80	0.000	0.678	0.846	1.043	1.292	1.664	1.990	2.374	2.639	3.195	3.416
100	0.000	0.677	0.845	1.042	1.290	1.660	1.984	2.364	2.626	3.174	3.390
1000	0.000	0.675	0.842	1.037	1.282	1.646	1.962	2.330	2.581	3.098	3.300
z	0.000	0.674	0.842	1.036	1.282	1.645	1.960	2.326	2.576	3.090	3.291
	0%	50%	60%	70%	80%	90%	95%	98%	99%	99.8%	99.9%
	Confidence Level										

Appendix E

Traditional Minimum Depth for Constant Depth Superstructures (AASHTO, 2017)



Superstructure		Minimum Depth (Including Deck) When variable depth members are uses, values may be adjusted to account for changes in relative stiffness of positive and negative moment sections	
Material	Type	Simple Spans	Continuous Spans
Reinforced Concrete	Slabs with main reinforcement parallel to traffic	$\frac{1.2\,(S+10)}{30}$	$\frac{S+10}{30} \geq 0.54$ ft
	T-Beams	$0.070L$	$0.065L$
	Box Beams	$0.060L$	$0.055L$
	Pedestrian Structure Beams	$0.035L$	$0.033L$
Prestressed Concrete	Slabs	$0.030L \geq 6.5$ in.	$0.027L \geq 6.5$ in.
	CIP Box Beams	$0.045L$	$0.040L$
	Precast I-Beams	$0.045L$	$0.040L$
	Pedestrian Structure Beams	$0.033L$	$0.030L$
	Adjacent Box Beams	$0.030L$	$0.025L$
Steel	Overall Depth of Composition I-Beam	$0.040L$	$0.032L$
	Depth of I-Beam Portion of Composite I-Beam	$0.033L$	$0.027L$
	Trusses	$0.100L$	$0.100L$

Ruodan Lu - September 2018